Stand-by Power Minimization through Simultaneous Threshold Voltage Selection and Circuit Sizing

Supamas Sirichotiyakul, Tim Edwards, Chanhee Oh, Jingyan Zuo,
Abhijit Dharchoudhury, Rajendran Panda, and David Blaauw
Advanced Tools, Motorola Inc., Austin, TX
david_blaauw@email.mot.com

Abstract

We present a new approach for estimation and optimization of the average stand-by power dissipation in large MOS digital circuits. To overcome the complexity of state dependence in average leakage estimation, we introduce the concept of "dominant leakage states" and use state probabilities. Our method achieves speed-ups of 3 to 4 orders of magnitude over exhaustive SPICE simulations while maintaining accuracies within 9% of SPICE. This accurate estimation is used in a new sensitivity-based leakage and performance optimization approach for circuits using dual V_t processes. In tests on a variety of industrial circuits, this approach was able to obtain 81-100% of the performance achievable with all low V_t transistors, but with 1/3 to 1/6 the stand-by current.

Keywords

Low-power-design, Dual-V_t, Leakage

1. Introduction and Prior Work

There is a growing need to analyze and optimize the stand-by component of power in digital circuits being designed for portable and battery-powered applications. Since these circuits remain in stand-by mode significantly longer than in active mode, their stand-by switching current has a major impact on battery life. Because of this, stringent specifications are being placed on the stand-by (or leakage) current drawn by such circuits. Reductions in operation voltage have accentuated the leakage current problem. As the power supply voltage is reduced, the threshold voltage of transistors is scaled down to maintain a constant switching speed. Since reducing the threshold voltage increases the leakage of a circuit exponentially, circuits operating with low supply voltages (such as 1V or below) obtain very low switching power but suffer from high leakage power.

To address the simultaneous constraints on circuit performance and leakage current for portable applications, dual-threshold[1] processes have come into use, allowing the circuit designer to choose the appropriate threshold voltage for each device. In a dual-threshold (dual V_t) process, an additional mask layer is used to assign either a high or low V_t to each transistor. Other approaches for leakage reduction, such as substrate-bias management[2] and insertion of special stand-by mode shut-off transistors[3], have also been proposed. However, these methods significantly increase design complexity.

Transistor Type	Switching Delay (norm)	Leakage Current (norm)
High-V_t	1.0	1.0
Low-V_t	0.53	33.2

Table 1: Performance and Leakage Current for High and Low V_t Transistors

Table 1 shows the performance and leakage current trade-off for high and low V_t transistors in a 0.25 micron industrial dual-V_t process at 0.9 Volts. Considering the high leakage current of low V_t transistors, a very careful analysis must be made to determine which transistors are set to low V_t such that the overall leakage current is not unduly increased.

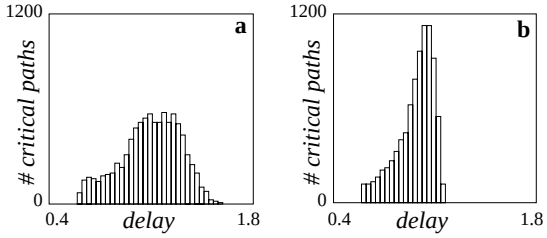


Figure 1. Path delay distribution of a circuit before and after size optimization.

The traditional approach to V_t selection for a circuit relies on the observation that a circuit's overall performance is often limited by a few critical paths. Transistors and gates along these critical paths are set to low-V_t while their transistor sizes are held fixed. By assigning a few transistors on the critical paths of the circuit to low V_t, overall circuit performance can be improved significantly while leakage current is kept within bounds. An example of the path delay distribution of a synthesized circuit is shown in Figure 1 (a), where the circuit's performance can be increased by 19% by speeding up only 15% of the total paths in the circuit. This approach was used in the PowerPC$_{TM}$ 750[1], and similar algorithms were proposed in [4] and [5]. While this approach provides good results for many circuits, it has difficulty optimizing circuits that are carefully balanced using post-synthesis optimization techniques such as transistor sizing. Figure 1 (b) shows the path distribution for the same circuit after transistor sizing has been applied. Further increasing the performance of this balanced circuit requires that

DAC 99, New Orleans, Louisiana

transistors on a large portion of all paths to be set to low V_t, resulting in a far less favorable trade-off between performance and leakage current.

In order to obtain a better trade-off between the performance and leakage of a design, the assignment of low and high V_t transistors must be performed while simultaneously adjusting transistor sizes. If, in a well-balanced circuit, the V_t of a transistor on the critical path is lowered while keeping the transistor sizes fixed, the path will become unduly fast, thereby making the sizes sub-optimal. Also, the gate capacitance of a transistor increases by approximately 8-10% as its V_t is lowered, slowing other paths passing through this transistor's gate node. Setting a transistor to low-V_t without subsequently adjusting the transistor sizes in the circuit can actually degrade the performance of the circuit while increasing leakage. In the approach proposed in this paper, we consider both V_t selections and transistor sizes of the circuit simultaneously. Our results show significant improvements in circuit performance or leakage current when transistor sizes and threshold voltages are optimized simultaneously, as compared to performing V_t selection with fixed sizes. The previous methods in [4] and [5] did not consider changing transistor sizes during optimization.

Another critical issue in leakage current optimization is obtaining an accurate and meaningful metric for the leakage current of a circuit which can be efficiently calculated and used in an optimization engine. The leakage current of a circuit is highly dependent on the state of the circuit. Figure 2 shows the leakage current for all states of a 3-input NAND gate. For this gate, the highest leakage current is 99 times greater than the lowest. When considering the current of a circuit as a whole, the correlation between the states of the gates must be considered. Furthermore, the state of a circuit's inputs is typically partially defined when the device enters stand-by mode. This partially-defined state is referred to as the sleep state. Previous approaches such as [6] have focused on calculating the maximum leakage across all permutations of the unspecified inputs. However, a device will enter sleep mode many times during the lifetime of its battery, each time with a random setting for the unspecified input signals. To obtain a reliable measure of the expected or mean lifetime of the battery, the average, rather than the maximum leakage of a circuit must be calculated. Previous approaches for calculating the maximum leakage of a circuit also suffer from inherent computational complexity, making them unsuitable for use in an optimization engine.

Leakage current calculation is further complicated by the highly non-linear behavior of the drain current of a device with respect to source/drain voltages. However, accurate SPICE-like simulation using non-linear models is very expensive, and becomes infeasible for repeated evaluation of large circuits in an optimization framework. In view of this, previous works used simpler but inaccurate models for leakage estimation, such as a gate-level[7][8] model or a stack-based model ignoring the voltage drops across the ON transistors in the stack[4][5][6]. These procedures can result in significant error as revealed in our experiments. The method proposed in this paper uses non-linear simulation with accurate leakage models. Simulation complexity is overcome through a series of techniques -- (i) eliminating the need to simulate the entire network, instead simulating only one DC-connected component (DCC) at a time and combining the results using state probabilities (calculated using the propagation of input state probabilities while accounting for first order spatial correlations[9]), (ii) further reducing individual DCCs using state information, the concept of dominant leakage states, and graph reduction techniques, and (iii) specially modifying the non-linear simulation for leakage simulation using pre-characterized tables. The techniques described here have been implemented in a tool called Duet, which is being used to optimize a variety of industrial circuits designed in submicron dual-V_t processes.

2. Leakage Measurement

In this work, we consider only subthreshold leakage current ($\mathbf{I_{sub}}$), the current through the channel at $\mathbf{V_{gs} < V_t}$. Junction leakage (reverse currents in source/drain junctions with the bulk) is 2 to 3 orders smaller than $\mathbf{I_{sub}}$ and is ignored. Likewise, the reverse junction current between well and bulk is ignored, as it is significantly smaller and is usually not a target for optimization at the circuit level.

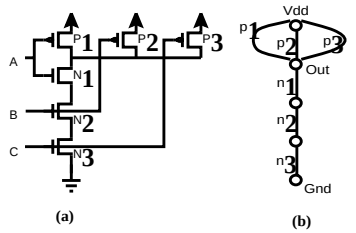
The average subthreshold leakage of a circuit is obtained from the leakage of individual DCCs simulated in various states, and from their state probabilities calculated using primary input probabilities. DCC-by-DCC evaluations eliminate the need to do non-linear simulation of the circuit as a whole, and the probabilistic approach eliminates the need to do simulations over all 2^n input combinations (where n is the number of circuit inputs). Moreover, the DCC leakages are used in transistor cost calculations during V_t selection.

For each DCC, only a small subset of all possible states is evaluated for leakage. This approach is based on the notion of *dominant leakage states* and on graph reduction using state information, as discussed in Section 2.1. Each state in the dominant leakage set of each DCC is simulated using an efficient and accurate leakage model, described in Section 2.2.

2.1 Dominant Leakage States

The leakage of a gate is significantly less in some states than in others. A state with more than one transistor OFF in a path from Vdd to Gnd (a *Vdd-Gnd path*) is far less leaky than a state with *only one* transistor OFF in any Vdd-Gnd path. We call these latter states *dominant leakage states*. The set of dominant leakage states is usually small compared with the set of all possible states. The key idea is to ignore the leakage of insignificant (non-dominant) states in the average leakage calculation without losing significant accuracy. For example, D={011, 101, 110, 111}, the set of four dominant leakage states for the NAND gate of Figure

2, accounts for 95.3% of the total leakage (assuming equal state probabilities). Hence, simulating only half the states incurs an error of only 4.7%. This trade-off becomes even more attractive for DCCs with a large number of inputs.



State (ABC)	Leakage (nA)	Leaking Transistors
000	0.095	N1, N2, N3
001	0.195	N1, N2
010	0.195	N1, N3
011	**1.874**	**N1**
100	0.185	N2, N3
101	**1.220**	**N2**
110	**1.140**	**N3**
111	**9.410**	**P1, P2, P3**

Figure 2. 3-input NAND gate, its graph representation, and its leakage current

We give the following definitions:

Let **G(V, E)** be the graph representing a DCC in the circuit, such that each $v \in \boldsymbol{V}$ represents a node in the DCC, and each $e \in \boldsymbol{E}$ represents a transistor in the DCC whose drain and source nodes are the endpoints of e. Since G represents a DCC, it has only one (connected) component.

A *disconnecting set* of edges in a connected graph G is any set of edges in G whose removal results in more than one connected component. If $\boldsymbol{F} \subseteq E(\boldsymbol{G})$ is a disconnecting set, **G - F** has more than one component. For instance, in Figure 2 (b), {n1, n3, p1, p2} is a disconnecting set of the graph **G**.

A cutset of **G** is defined as a minimal disconnecting set of **G**. Since it is minimal, a cutset always leaves a graph with exactly two components. Given a non-empty set $S \subset V(G)$, **[S, $\overline{\mathbf{S}}$]** denotes a cutset of **G**, the set of edges each having one end point in **S** and the other in $\overline{\mathbf{S}}$. In Figure 2 (b), {n3} is a cutset of **G**. We also define that Vdd is always in **S**.

Let **B** be the set of all possible Boolean states for a gate's inputs. An edge is called an *OFF-edge* if its corresponding transistor is OFF in a given state. Given $b \in \boldsymbol{B}$, let OFF(*b*) denote the set of OFF-edges for state b. For instance, OFF(010) in Figure 2(b) is {n1, n3, p2}.

Let LEAK(*b*) be the set of transistors that contributes to subthreshold leakage in state b. It is clear that $LEAK(b) \subseteq OFF(b)$ and that the endpoints of each edge of LEAK(*b*) are in different components of **G** - LEAK(*b*). If both the drain and source nodes of a transistor are in the same component of **G** - LEAK(b), then there is a conducting path between them consisting of other transistors in the component. Such a transistor will not contribute to subthreshold leakage. In our example, LEAK(010) is {n1}.

We define a state b to be a *dominant leakage state* if LEAK(b) is minimal, i.e. if there exists no other state $a \in \boldsymbol{B}$ such that $LEAK(a) \subset LEAK(b)$. If b is a dominant leakage state, LEAK(*b*) is called a dominant leakage set. For instance, in Figure 2(b) there is no state whose LEAK set is a subset of LEAK(011) = {n1}. So 011 is a dominant leakage state, while 010 whose LEAK set is {n1, n3} is not.

By our definition, a dominant leakage set is a minimal disconnecting set of **G**, and is hence a cutset [**S**, $\overline{\mathbf{S}}$] of **G**, such that Vdd is in **S** and Gnd is in $\overline{\mathbf{S}}$. That is, when *b* is a dominant leakage state, **G** - LEAK(*b*) has exactly two components, with Vdd and Gnd in different components.

We will now show how to efficiently obtain the dominant leakage sets. We start with the graph of a DCC and systematically generate its cutsets using a breadth-first traversal. A cutset is qualified as a dominant leakage set only if: 1) removing its edges separates Vdd and Gnd into different partitions, and 2) all of its edges can be logically OFF at the same time.

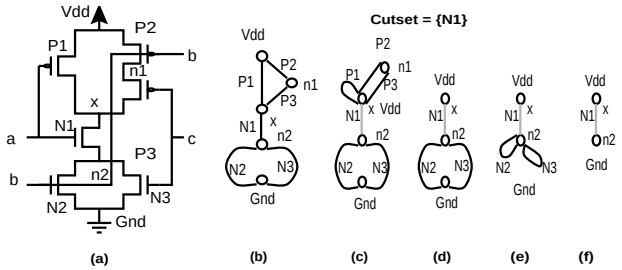


Figure 3. Dominant leakage state generation for an OAI gate.

The breadth-first traversal starts with an initial partitioning [**S**, $\overline{\mathbf{S}}$] of the nodes wherein only the Vdd node is in **S**. Nodes are then recursively added to **S** until all nodes but the Gnd node are included in **S**. Partitions that create more than 2 connected components are not considered. This guarantees that condition 1 is satisfied. At each point in the traversal, partition duplication is detected. For each generated cutset, we assert the input vector such that all edges in the cutset are logically OFF. If an assertion fails, the cutset is rejected as infeasible, guaranteeing that condition 2 is satisfied. For example, in Figure 3(a), if inputs a and b are inversely related then the cutset {P1, P2} is infeasible.

Note that a cutset only partially defines the input state. The full set of states that need to be simulated for each cutset is found using the following procedure:

For each feasible cutset generated:

1. Assert the cutset inputs and add their values to the known set. For example, in Figure 3, when the cutset is {N1} the known set is {a=0}.

2. Reduce the graph as follows:

(i) If an edge is logically ON and is of *native type* (it corresponds to a PMOS (NMOS) transistor and both its drain and source are in the **S** ($\overline{\mathbf{S}}$) partition), merge the two end nodes of the edge. In Figure 3, P1 is ON and is of native type, and nodes Vdd and X are merged.

(ii) If, as a result of Step (i), an edge lies in a loop which does not contain any edges in a Vdd-Gnd path, remove the edge from the graph. From Figure 3 (c), P1, P2, and P3 are in loops, and are removed in Figure 3 (d).

3. For each transistor in the reduced graph whose input logic value is not defined:

(i) If a *feasible* assertion on the transistor gate node can be made, perform the assertion, add the node value to the known set, and reduce the graph as described in Step 2. An assertion on a transistor is said to be *feasible* if it turns ON that transistor and does not turn OFF any other

transistors in the reduced graph. A feasible assertion is guaranteed to maximize the leakage of the cutset since it does not turn any transistor OFF. In Figure 3 (d), turning on N2 is a feasible assertion. The known set becomes {a=0, b=1}. Further reductions are made in (e) and (f).

(ii) If an assertion of the gate node is not feasible, add the transistor gate node to a set called the *permute set*.

4. When the reduced graph does not contain any transistors in an undefined state, a full set of dominant leakage states for the cutset is created by enumerating all input permutations of nodes in the permute set.

Each state in the dominant leakage set will be simulated using the simulation engine described in Section 2.3. For each state we simulate with the reduced graph of each state instead of the full graph.

2.2 Leakage Model and Estimation Engine

A number of methods have been proposed for quickly calculating an approximate leakage number for a stack of transistors[4][5][6]. These methods pre-calibrate the leakage of a single transistor, and then apply a constant multiplier to reduce the leakage when more than one transistor is leaking in series. However, a linear scaling factor cannot accurately predict leakage over a range of transistor widths and stack topologies. These methods also ignore the V_t drop across transistors that are ON in series with transistors that are leaking. In our experiments, we found that ignoring this V_t drop over-estimates the leakage current by approximately 30% for typical gates in a 1.5 volt, 0.25 micron process.

State	Spice (nAmp)	Newton-Raphson (nAmp)	Diff (%)
000	0.095	0.093	-2.11
001	0.195	0.193	-1.03
010	0.195	0.193	-1.03
011	1.874	1.873	-0.05
100	0.185	0.180	-0.42
101	1.220	1.222	0.16
110	1.140	1.138	-0.18
111	9.410	9.412	0.02

Table 2: NAND3 Leakage Measurement Results Using Newton-Raphson

To obtain both a fast run time and an acceptable accuracy, our approach is based on Newton-Raphson iterations using fast table lookups of I_{ds}. The drain current of a given type of MOS device is described with the non-linear function $I_{ds} = f(V_d, V_s, W, V_g)$. As the V_g for a device is either Vdd or 0 during leakage simulation, I_{ds} is captured in two 3-dimensional tables, one for each value of V_g. These tables are derived through pre-characterization using SPICE simulations with accurate models. When the reduced graph contains only Vdd and Gnd nodes as in Figure 3(f), the state leakage is directly referenced from a table. Otherwise, KCL equations for the DCC are set up and the currents are solved through Newton-Raphson iterations and the tabular current model. Table 2 shows the comparison between our fast leakage simulation and SPICE for all possible states of a 3-input NAND.

3. Simultaneous V_t Selection and Sizing

We shall now describe the method of optimization wherein we determine the size (width) and threshold voltage for each transistor in a given circuit such that its area, performance, and leakage current are optimal. Both the performance of the circuit and its leakage vary non-linearly with device widths and their V_t's. Moreover, the width domain is continuous while the V_t domain is discrete. Thus, finding an exact optimum solution would require solving an integer non-linear program. This is prohibitively expensive even for circuits of moderate size. Hence, we need to take a heuristic approach.

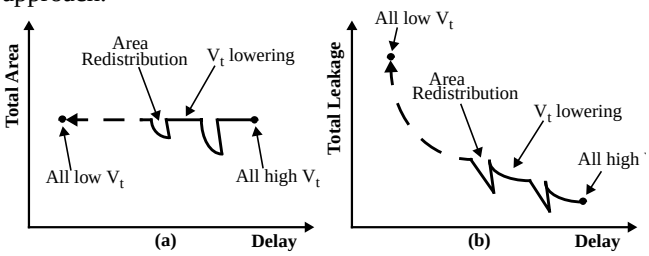


Figure 4. V_t selection and redistribution of area, two views

For the set of all optimal solutions with a given total area, there exists both an all-high V_t and an all-low V_t solution. The all-high V_t solution has the lowest leakage, while the all-low V_t solution has the best performance. These two solutions are illustrated in Figure 4 by the rightmost and leftmost points, respectively. Our approach explores optimal mixed-V_t solutions with leakage and performance lying between these bounds by moving horizontally (i.e. with fixed total area) from all-high to all-low V_t. As shown in Figure 4(a), the intermediate solutions are generated by repeating two basic steps: (1) changing the Vt of some transistors to their low value, and (2) resizing the circuit. Figure 4(b) shows the same segment in the leakage domain. The first step focuses on obtaining a maximum improvement in the speed of the circuit while incurring a minimum increase in its leakage through sensitivity-guided optimization. The second step is aimed at recovering additional performance by redistributing the area optimally after the V_t changes. The steps are detailed in the following sections.

3.1 Threshold Voltage Selection

Lowering the threshold voltage of a particular transistor has both a positive and a negative impact on circuit performance. The drive strength of the transistor is significantly increased, resulting in a must faster switching delay. Also, the gate capacitance of the transistor is increased, and paths that pass through the gate node of the transistor are slowed. Finally, the impact on leakage depends strongly on the location of a transistor within its gate and on the state probabilities of the gate. Therefore,

previous methods[4] which simply modify the transistors in a predetermined order, from output to input, do not adequately evaluate the impact of the transistor on the leakage and performance of the circuit and will result in a sub-optimal solution.

We propose the iterative use of a cost function which evaluates the increase in total leakage w.r.t. the performance gain of the whole circuit. In each iteration, the cost function is calculated for all transistors in the circuit, and the transistor with the minimum cost is selected and is set to low V_t. The circuit sizes are then rebalanced as explained in Section 3.2, the circuit timing and transistor cost are incrementally recalculated, and the procedure is repeated. The cost function is shown below.

$$Cost(T) = \frac{\Delta I_{sub}(T)}{\Delta D(T)}, \; where \; \Delta D(T) = \sum_{arcs}^{\alpha} \Delta d_{\alpha}(T) \cdot \frac{1}{k + Min(slacks) - slack_{\alpha}}$$

The V_t change of a transistor directly impacts the delay of a number of timing arcs in its gate and in the gate driving its gate node, due to added capacitive loading. The impact of the V_t change of a transistor **T** on a particular timing arc α is denoted by $\Delta\mathbf{d}_{\alpha}(\mathbf{T})$ in the above equation. The weighted sum of $\Delta\mathbf{d}_{\alpha}(\mathbf{T})$ is taken using the function 1/(k+Min(slack)-$slack_{\alpha}$), where k is a small negative number and Min(slack) is the critical slack in the circuit. This weighting function takes on the value 1/k for timing arcs on the critical path and quickly approaches zero for timing arcs that are less critical. The weighted sum ($\Delta\mathbf{D}(\mathbf{T})$) therefore captures the impact of lowering the V_t for transistor **T** on all affected paths in the circuit, weighted by their criticality. For example, if a critical path is improved in performance by lowering the V_t of transistor **T**, but a near-critical path is slowed down significantly, transistor **T** will not be selected for V_t lowering. By taking the ratio of $\Delta\mathbf{I}_{sub}(\mathbf{T})$ over $\Delta\mathbf{D}(\mathbf{T})$, the improvement in performance is weighted relative to the increase in leakage.

The factor $\Delta\mathbf{d}_{\alpha}(\mathbf{T})$ is the change of the delay of timing arc α in the circuit due to the change in the V_t of the transistor **T**. This is calculated using an analytical function based on the Elmore delay model similar to that in [10]. Since $\Delta\mathbf{d}_{\alpha}(\mathbf{T})$ is calculated analytically, the evaluation of $\Delta\mathbf{D}(\mathbf{T})$ is extremely efficient. Since the Elmore-based delay model is approximate, it is only used for calculating $\Delta\mathbf{d}_{\alpha}(\mathbf{T})$. The actual timing of the circuit and the value of $\mathbf{slack}_{\alpha}$ are based on an accurate regionwise quadratic delay model[11].

The factor $\Delta\mathbf{I}_{\mathbf{sub}}(\mathbf{T})$ is the change in leakage of the circuit due to the change in the V_t of transistor **T**. This is calculated numerically by lowering the V_t of the transistor and estimating the average leakage of the DCC using the procedure in Section 2. The dominant leakage states of a DCC are independent of the V_t settings in the circuit and are therefore not recalculated. Furthermore, only those dominant leakage states containing transistor **T** in their reduced graphs need to be re-simulated. This significantly reduces the cost of calculating $\Delta\mathbf{I}_{sub}(\mathbf{T})$.

After the cost function is calculated for all transistors, the transistor with the minimum cost is selected. The advantage of this approach is that in each iteration it selects the transistor which increases the circuit performance the most, relative to its increase in leakage, while taking into account both the increased drive strength and the increased capacitance on the performance of the circuit as a whole.

3.2 Rebalancing

As previously mentioned, a circuit's device sizes are no longer optimal once the V_t of one or more transistors has been lowered. We redistribute the transistor area of the circuit by 1) reducing selected transistor widths and 2) resizing the circuit back to its original area.

We can identify a V_t change's cone of influence to a predetermined depth by following its device's connections into neighboring devices (to a specified depth) and recording their distance from the changed device. Next, we apply a width reduction to the marked set of devices based on their distances from the changed device. The changed device itself sees the greatest reduction, while the farthest devices see the smallest. We have determined experimentally that consideration of no more than three levels of logic in the cone of influence with a linear reduction gradient gives results equivalent to even the most aggressive reduction scheme.

The second step of rebalancing is resizing the circuit, re-inserting the area gained during reduction in order to decrease a circuit's worst delay. We use a delay/area sensitivity-based size optimization tool for the resizing step[12]. This tool balances the delays of all timing paths, thus minimizing total circuit area for a given performance. While the resizing phase initially focuses only on the obviously undersized devices affected during the reduction step, all devices in the circuit are candidates for resizing, and excess area is distributed across all critical timing paths.

4. Results

Duet, our implementation of the proposed leakage measurement and threshold voltage-size optimization algorithms, is currently being used for industrial low-power DSP processor design, and has been successfully run on a large number of circuits.

Table 3 gives the details of our benchmark circuits, as well as the average leakage measurement results obtained by SPICE and by our approach. Circuit *add1* is a 4-bit adder, *add2* is a 25-bit adder, *add3* is a 32-bit adder, *pla* is a PLA-type circuit, and the others are control circuits. Column 4 shows the number of circuit states which would have to be individually simulated in an exhaustive approach. Column 5 shows the actual number of states solved with our non-linear solver.

The measurements in column 6 (*Leakage Current - Spice*) were obtained by exhaustively simulating each circuit over

all possible input combinations and then taking the average leakage. The measurements in column 7 (*Leakage Current - Ours*) are from our approach, and are compared in column 8 with the SPICE measurements. For these circuits, our approach took less than 2 seconds (on a Sun Ultrasparc 60) to calculate the leakage. This amounts to a more than 6000x speed-up over exhaustive SPICE simulation. Also, note that for the circuits *add2*, *add3*, and *control1* it is infeasible to run exhaustive SPICE simulations, as it would require 2.3e15, 3.69e19, and 2.5e27 simulation runs, respectively. The results also show the high accuracy of our method, which is within 9% of SPICE.

Circuit Characteristics					Leakage Current		
Name	Inputs	FETs	Circuit states	Solve states	Spice (nAmp)	Ours (nAmp)	Diff (%)
blk	9	108	512	30	59.95	58.87	-1.79
bay	9	68	512	16	652.44	703.14	7.21
add1	10	244	1024	71	488.86	472.27	-3.39
pla	12	1052	4096	246	3274.05	2987.40	-8.76
add2	51	1090	2.3e15	270	N/A	554.61	N/A
add3	65	1256	3.69e19	300	N/A	418.95	N/A
control1	91	5318	2.5e27	966	N/A	5668.49	N/A

Table 3: Benchmark Details, Leakage Mesurements

We also benchmarked our simultaneous V_t selection and size optimization algorithm on the example circuits. Quantitative optimization results are shown in Table 4. The columns *High_V_t_Size* and *Low_V_t_Size* show the delay and leakage of the circuit sized for performance with all high and low V_t transistors, respectively. Columns *V_t_Opt* and *V_t_Size_Opt* are from solutions which are considered to have reasonable trade-offs in terms of delay and leakage.

	High_Vt_Size	Low_Vt_Size	Vt_Opt	Vt_Size_Opt	
Circuit Name	Delay/ Leakage (ns) / (uA)	Delay/ Leakage (ns) / (uA)	Delay/ Leakage (ns) / (uA)	Delay (% increase over Low_Vt) (ns)	Leakage (reduction factor over Low_Vt) (uA)
blk	0.33 / 0.002	0.25 / 0.048	0.25 / 0.037	0.25 (0%)	0.011 (4.4x)
bay	0.52 / 0.022	0.39 / 0.48	0.42 / 0.37	0.42 (3%)	0.14 (3.4x)
add1	0.74 / 0.03	0.55 / 0.74	0.59 / 0.32	0.59 (7%)	0.17 (4.4x)
pla	1.09 / 0.09	0.74 / 1.98	0.77 / 0.55	0.77 (4%)	0.32 (6.2x)
add2	1.58 / 0.018	1.18 / 0.40	1.25 / 0.25	1.25 (6%)	0.13 (3.1x)
add3	1.68 / 0.027	1.21 / 0.60	1.31 / 0.35	1.31 (8%)	0.19 (3.2x)
control1	1.53 / 0.13	0.96 / 2.84	1.14 / 1.38	1.14 (19%)	0.63 (4.5x)

Table 4: V_t and Size Optimization Results

As expected, *High_V_t_Size* exhibits very low leakage but the circuit speed is also the slowest. On the other hand, *Low_V_t_Size* can achieve much faster circuit speed at the cost of a significantly higher leakage. During the *V_t_Size_Opt* optimization, the circuit delay progressively improves and leakage increases as more and more transistors are changed to low V_t. The achieved circuit delay is very close to that of *Low_V_t_Size*, but with considerably lower leakage.

A comparison of results from *V_t_Size_Opt* and *V_t_Opt* demonstrates the benefit of rebalancing the circuit after each V_t change operation. Table 4 shows that V_t_Size_Opt can achieve the same delay target with 1.8 - 3.5 times less leakage than V_t_Opt in most cases. This supports our claim that circuit sizes are suboptimal after the V_t of a transistor is changed, and that localized reallocation of transistor sizes can alleviate this suboptimal size assignment.The run times for the optimization were also reasonable. The largest circuit, *control1*, has 5318 transistors and was successfully optimized within 1.5 CPU hours on a Sparc 60.

5. Conclusions

We have presented an efficient technique for accurately estimating the average stand-by power of MOS circuits using a variety of problem reduction techniques, including the notion of dominant leakage states. We have also given a simultaneous V_t selection and sizing optimization procedure that uses leakage and delay sensitivities to optimally trade-off stand-by power and performance in dual V_t circuits. The benefits of combining V_t selection and transistor sizing over the earlier approach of performing only V_t selection were demonstrated. Test results show the accuracy and performance improvements of our estimation procedure, and the performance vs. stand-by power trade-offs were also presented.

6. References

[1] N. Rohrer, et al. “A 480MHz RISC microprocessor in a 0.12 um Leff CMOS technology with copper interconnects”, IEEE International Solid-State Circuits Conference, 1998.

[2] Y. Oowaki, et al., “A Sub-0.1um Circuit Design with Substrate-over-Biasing”, ISSCC, page 88, February 1998.

[3] J. Kao, A. Chandrakasan, D. Antoniadis, “Transistor sizing issues and tool for multi-threshold CMOS technology”, Proc. Design Automation Conference, 1997

[4] L.Wei, et al. “Design and Optimization of Low Voltage High Performance Dual Threshold CMOS Circuits”, 35th Design Automation Conference, 1998

[5] Qi Wang, et al. “Static power optimization of deep submicron CMOS circuits for dual V_t technology,” ICCAD 1998.

[6] Z. Chen, et al. “Estimation of Standby Leakage Power in CMOS Circuits Considering Accurate Modeling of Transistor Stacks”, ISLPED, 1998.

[7] J. Halter and F.N. Najm, “A gate-level leakage power reduction method for ultra-low-power CMOS circuits,” Custom Integrated Circuit Conference, 1997.

[8] P. Pant, et. al., “Device-circuit optimization for minimal energy and power consumption in CMOS random logic networks,” 34th Design Automation Conference, June 1997.

[9] S.Ercolani, M.Favalli, M.Damiani, P.Olivo, B.Ricco. “Testability Measures in Pseudorandom Testing”, IEEE Trans. on CAD, 1992, v.11, n.6, pp.794-800.

[10] J.P. Fishburn, et al., “TILOS: A posynomial programming approach to transistor sizing,” ICCAD, Nov 1985

[11] A. Dharchoudhury, et al., “Fast and accurate timing simulation with regionwise quadratic models of MOS I-V characteristics,” ICCAD, Nov. 1994, pp190-194

[12] A. Dharchoudhury, et. al., “Transistor-level sizing and timing verification of domino circuits in the PowerPC$_{TM}$ microprocessor,” ICCD, October 1997.