Duet: An Accurate Leakage Estimation and Optimization Tool for Dual-V_t Circuits

Supamas Sirichotiyakul, *Member, IEEE*, Tim Edwards, Chanhee Oh, *Member, IEEE*, Rajendran Panda, *Member, IEEE*, and David Blaauw, *Member, IEEE*

***Abstract*—We present a new approach for the estimation and optimization of standby power dissipation in large MOS digital circuits. We first introduce a new approach for accurate and efficient calculation of the average standby or leakage current in large digital circuits by introducing the concepts of "dominant leakage states" and the use of state probabilities. Combined with graph reduction techniques and simplified nonlinear simulation, our method achieves speedups of three to four orders of magnitude over exhaustive SPICE simulations while maintaining very good accuracy. The leakage current calculation is then utilized in a new leakage and performance optimization algorithm for circuits using dual V_t processes. Our approach is the first to consider the assignment of both the V_t and the width of a transistor, simultaneously. Our optimization approach uses incremental calculation of leakage and performance sensitivities and can take into account a partially defined circuit state constraint for the standby mode of the device. In tests on a variety of industrial circuits, our optimization 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 standby current. We also show that knowledge of the standby state of the device enhances the leakage/performance tradeoff.**

***Index Terms*—Critical-path, dual-V_t, high-performance, leakage, low-power design, low-power dissipation, low-voltage, performance-tradeoffs.**

I. Introduction and Prior Work

THERE is a growing need to analyze and optimize the stand-by component of power in digital circuits designed for portable and battery-powered applications. Since these circuits remain in standby (or sleep) mode significantly longer than in active mode, their standby current and not their active switching current, determines battery life. Because of this, stringent specifications are being placed on the standby (or leakage) current drawn by such devices. Very high-performance circuits, such as microprocessors and microcontrollers, are designed with their switching power as a primary concern. Circuits for use in portable applications are also constrained by their leakage power. Reductions in operating 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 device exponentially, circuits operating with low-supply voltages (such as 1 V or below) have very low switching power but suffer from high-leakage power.

To address the simultaneous constraints on circuit performance and leakage current for portable applications, a careful tradeoff must be made in the selection of the threshold voltage (V_t). For designs where performance and leakage current constraints cannot be met simultaneously with a single V_t for all devices, dual-threshold [1], [2] processes have come into use, allowing the circuit designer to choose the appropriate V_t (high or low) 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 [3] and insertion of special standby mode shut-off transistors [4], [22], have also been proposed. However, these methods significantly increase design complexity.

Table I shows the performance and leakage current tradeoff for high- and low-V_t transistors in a 0.25 μm industrial dual-V_t process at 0.9 V. Considering the high-leakage current of low-V_t transistors, a very careful analysis must be made to determine which transistors are assigned a low V_t, such that the overall leakage current is not unduly increased. Setting too many devices to low V_t quickly results in a significant increase in the overall leakage of the circuit.

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 distribution of a synthesized circuit is shown in Fig. 1(a), where the circuit's performance can be increased by 19% through speeding up only 15% of the total paths in the circuit. This approach was used in the PowerPC 750 [2] and similar algorithms were proposed in [5] and [7]. 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. Fig. 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 transistors on a large portion of all paths are set to low V_t resulting in a far less favorable tradeoff between performance and leakage current.

Manuscript received January 12, 2002; revised September 7, 2001.

S. Sirichotiyakul is with Sun Microsystems, Moston, MA 01824 USA.

T. Edwards, C. Oh, and R. Panda are with Advanced Tools, Motorola Incorporated, Austin, TX 78729 USA.

D. Blaauw is with the EECS Department, University of Michigan, Ann Arbor, MI 48104 USA (e-mail: blaauw@umich.edu).

Publisher Item Identifier S 1063-8210(02)10695-0.

TABLE I
PERFORMANCE AND LEAKAGE CURRENT FOR A HIGH AND LOW V_t TRANSISTOR

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

In order to obtain a better tradeoff between 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 size fixed, the path will become unnecessarily fast, thereby making the sizes suboptimal. Also, a lower V_t causes the formation of the channel to occur earlier during the input transition due to the earlier onset of strong inversion with a reduced V_t. This results in an increase of approximately 8–10% of the average gate capacitance of the transistor gate as seen by the input driver. Therefore, when a transistor is assigned a low V_t, the increase of its gate capacitance can adversely affect the performance of other paths loaded by this transistor's gate node. Hence, setting a transistor to low-V_t without subsequently adjusting transistor sizes in the circuit can actually degrade the performance of the circuit while increasing leakage. The transistor sizes must be adjusted simultaneously with the assignment of V_t values to obtain an optimal solution.

In the approach proposed in this paper, we consider both V_t selections and transistor sizes of the circuit simultaneously. Our approach uses two new techniques. First, the V_t selection uses leakage and performance sensitivities to accurately determine the impact of changing a transistor's V_t. Second, the V_t selection process incrementally adjusts the transistor sizes of the circuit after each change in V_t. We present results that show a significant improvement 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 [5], [7] and [22] did not consider changing transistor sizes during optimization.

Our optimization approach assumes that the actual high- and low-V_t values on a device type are fixed. This assumption simplifies the threshold voltage selection problem to one with a discrete domain with only two choices: high V_t or low V_t. However, the approach can be easily extended to technologies with multiple discrete V_t values. Also, we assume that each transistor can be individually assigned a V_t value (high or low). However, our approach can easily be extended for stack-based or gate-based optimization by doing size adjustment and V_t selection of a predefined group of transistors, where a group consists of transistors in the same stack or in the same gate. Stack-based or gate-based optimizations are preferable if there is a manufacturing process limitation on assigning different V_t values to closely-spaced transistors. Gate-based optimization is also more suitable for a standard cell design methodology.

A 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. Fig. 2 shows the leakage current for all states of a three-input NAND gate. For this gate, the highest leakage current is 99 times greater than the lowest, clearly indicating a strong dependence of the leakage on the circuit state. When considering the leakage current of the circuit as a whole, the correlation between the states of the gates must also be considered. Furthermore, the state of a circuit's inputs is typically partially defined when the device enters standby mode. This partially defined state is referred to as the *sleep state*. Previous approaches such as [8] 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 for the expected or mean lifetime of the battery, the average, rather than the maximum leakage of a circuit must be calculated. The approaches for calculating the maximum leakage of a circuit also suffer from inherent computational complexity, making them unsuitable for use in an optimization engine.

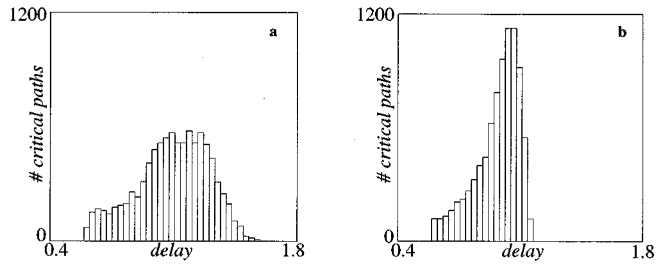
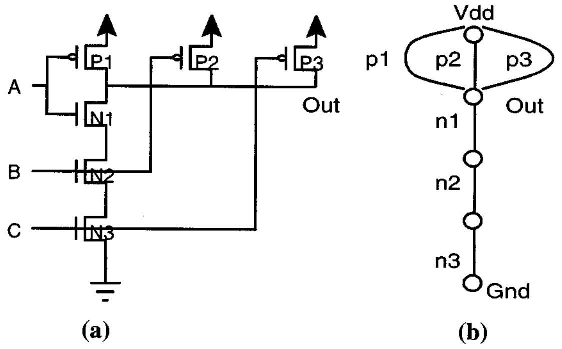


Fig. 1. Path delay distribution of a circuit before and after size optimization.

Leakage current calculation is complicated by the highly nonlinear behavior of the drain current of a device with respect to source/drain voltages. Several simple models for subthreshold operation which are efficient for circuit simulation have been proposed in [9]–[13]. Nevertheless, accurate SPICE-like simulation using these nonlinear models is still 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 [14], [15] model or a stack-based model ignoring the voltage drops across the ON transistors in the stack [5], [7], and [8]. These procedures can result in significant error as revealed in our experiments. In contrast, the method proposed here uses nonlinear simulation with accurate leakage models similar to those in [9]–[13]. Simulation complexity is overcome through a series of techniques. 1) 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, 2) further reducing individual DCCs using state information, the concept of dominant leakage states and graph reduction techniques, and 3) specially modifying the nonlinear simulation for leakage simulation using precharacterized tables. Furthermore, sleep-state information is seamlessly handled by this approach and contributes to a further improvement in optimization results. The techniques described here have been implemented in a tool called Duet, which has been used to optimize a variety of in-



State (ABC)	Leakage Current (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**

Fig. 2. Three-input NAND gate, its graph representation and its leakage current.

dustrial circuits designed in deep submicron dual-V_t processes. The leakage current calculation in this tool can be used to analyze circuit-leakage current at different process corners. The analysis is usually done at the best case process corner to obtain the worst-leakage current.

The remainder of this paper is organized as follows. In Section II, we present the approach for average leakage estimation. In Section III, we present the algorithm for simultaneous V_t selection and transistor sizing. In Section IV, we present benchmark results, and in Section V we present our conclusions.

II. Leakage Estimation Approach

In this work, we consider only subthreshold leakage current ($\mathbf{I_{sub}}$), the current through the channel at $\mathbf{V_{gs}} < \mathbf{V_t}$. Junction leakage (reverse currents in source/drain junction diodes with the bulk) is two to three 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.

In our approach, the complexity of average leakage estimation is reduced through a sequence of steps, presented below in a top-down manner.

1) The average 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 nonlinear 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 merit calculations during V_t selection.
2) 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 II-A.
3) Each state in the dominant leakage set of a DCC is simulated using an efficient and accurate leakage model, described in Section II-B.

A. Dominant Leakage States

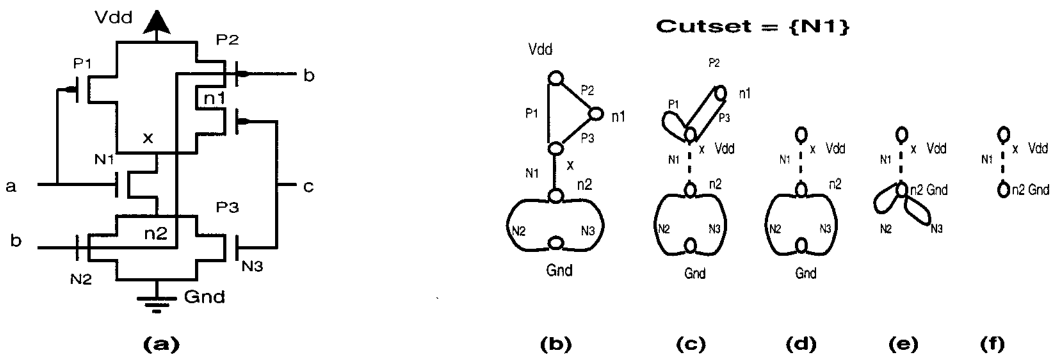
In experiments using SPICE simulations of several gates overall possible states, we observed that the leakage of a gate is significantly less in some states than in others. A state with more than one transistor OFF in a path from Vdd to Gnd (a *Vdd-Gnd path.*) is far less leaky than a state with *only one* transistor OFF in any Vdd-Gnd path. We call these latter states *dominant leakage states*. The set of dominant leakage states is usually very small compared with the set of all possible states. The key idea is to ignore the leakage of insignificant (nondominant) states in the average leakage calculation without losing significant accuracy. For example, SPICE simulations of a three-input NAND gate in Fig. 2 show that the exact average leakage is 1.78925 nA, assuming equal probabilities for all states. The set of dominant leakage states for this gate is $D = \{011, 101, 110, 111\}$. Considering only these four states, the average leakage is 1.7055 nA, only 4.68% less than the exact average. Note that such accuracy is obtained by simulating only four out of the eight possible states. This tradeoff becomes even more attractive for DCCs with a large number of inputs.

Before showing how to find the dominant leakage states, we give several definitions.

Let $\mathbf{G}(\mathbf{V}, \mathbf{E})$ be the graph representing a DCC in the circuit, such that each $v \in V$ represents a node in the DCC and each $e \in E$ represents a transistor in the DCC whose drain and source nodes are the endpoints of e. Since $\mathbf{G}$ represents a DCC, it has only one (connected) component. Also $|\mathbf{V}| > 1$, as there are at least two nodes, Vdd and Gnd.

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 $\mathbf{F} \subseteq E(\mathbf{G})$ is a disconnecting set, $\mathbf{G}(\mathbf{V}, \mathbf{E} - \mathbf{F})$ has more than one component. For instance, in Fig. 2(b), {n1, n3, p1, p2} is a disconnecting set of the graph $\mathbf{G}$.

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



S Set of Partition	Cutset	Cutset input (abc)	Dominant Leakage State (abc)
{Vdd}	{P1, P2}	11u	110
{Vdd, n1}	{P1, P3}	1u1	101
{Vdd, x, n1}	{N1}	0uu	01x
{Vdd, x, n1, n2}	{N2, N3}	u00	100

Fig. 3. Dominant-leakage state generation for an OAI gate.

Let $\mathbf{B}$ 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 $b \in B$, let OFF(b) denote the set of OFF-edges for state b. For instance, OFF(**010**) in Fig. 2(b) is {n1, n3, p2}. It is imperative that OFF(b) is a disconnecting set for $\mathbf{G}$ such that Vdd and Gnd lie in different components of $\mathbf{G}(\mathbf{V}, \mathbf{E} - \text{OFF}(b))$. Otherwise, there will be a short circuit in state b.

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 $\mathbf{G}(\mathbf{V}, \mathbf{E} - \text{OFF}(b))\underline{\text{C}}$. If both the drain and source nodes of a transistor are in the same component of $\mathbf{G}(\mathbf{V}, \mathbf{E} - \text{OFF}(b))$, then there is a conducting path between them consisting of other transistors in the component. Such a transistor will not contribute to subthreshold leakage. LEAK(b) is also a disconnecting set of $\mathbf{G}$. In our example, LEAK(**010**) is {n1, n3}.

We define a state b to be a *dominant-leakage state* if LEAK(b) is minimal, i.e., if there exists no other state $a \in B$ 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 Fig. 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 $\mathbf{G}$ and is, hence, a cutset $[\mathbf{S}, \bar{\mathbf{S}}]$ of $\mathbf{G}$, such that Vdd is in $\mathbf{S}$ and Gnd is in $\bar{\mathbf{S}}$. That is, when b is a dominant leakage state, $\mathbf{G}(\mathbf{V}, \mathbf{E} - \text{LEAK}(b))$ 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 partitions Vdd and Gnd into different partitions and 2) all of its edges can be logically OFF at the same time.

The breadth-first traversal starts with an initial partitioning $[\mathbf{S}, \bar{\mathbf{S}}]$ of the nodes wherein only the Vdd node is in $\mathbf{S}$. Nodes are then recursively added to $\mathbf{S}$ until all nodes but the Gnd node are included in $\mathbf{S}$. Partitions that create more than two connected components are not considered. This guarantees that condition 1 is satisfied. At each point in the traversal, partition duplication is detected. Fig. 3 shows an OAI gate, its $\mathbf{S}$ partitions and its cutsets.

For each generated cutset, we assert a partial 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 two is satisfied. For example, if in Fig. 3(a) inputs a and b are inversely related, then the cutset {P1, P2} is infeasible.

Note, that a cutset defines a partial input state since it only asserts input nodes that control the cutset edges. To simulate the leakage current for the cutset, a fully-defined input state is needed. Simple enumeration of undefined input variables to obtain a set of fully defined input states for a cutset could lead to a very large number of circuit states. For the cutset $\{N1\}$ in Fig. 3, the following four-input states would need to be simulated: $\{\bar{a}bc, \bar{a}b\bar{c}, \bar{a}\bar{b}c, \bar{a}\bar{b}\bar{c}\}$. Therefore, we expand the partially defined cutset state such that only circuit states are generated for leakage simulation that maximize the leakage current for the cutset. The procedure for deriving a set of fully defined circuit states from a partially defined cutset state is shown below.

For each feasible cutset generated as follows.

1) Assert the cutset inputs and add their values to the *known set*. For example, in Fig. 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*, merge the two end nodes of the edge. A native type transistor is a PMOS (NMOS) transistor whose drain and source nodes are in the $\mathbf{S}(\bar{\mathbf{S}})$ partition. If a transistor is of native type, there will be no V_t drop across the transistor when it is ON. Therefore, its source and drain nodes will be at equal potentials in the DC leakage simulation and the nodes in the graph can be merged. In our example, the known set is $\{a = 0\}$, so P1 is ON and is of native type. Nodes Vdd and X are merged.

Fig. 3(b) and (c) show the graph representations of the OAI gate before and after the merge.

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 Fig. 3(c), P1, P2, and P3 are in loops. They are removed as shown in Fig. 3(d).

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

a) 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 Fig. 3(d), turning on N2 is a feasible assertion. The known set becomes $\{a = 0, b = 1\}$. Since N2 is now ON and is of native type, nodes Gnd and n2 are merged as shown in Fig. 3(e). Now both N2 and N3 form loops, so they can be removed as shown in Fig. 3(f).

b) If an assertion of the gate node is not feasible, add the transistor gate node to 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. In our example, the set of dominant leakage states for the cutset $\{N1\}$ is $\{abc = 01X\}$.

States generated from different cutsets may be duplicates or may dominate one another. These cases are eliminated after a complete set of states is generated using every cutset.

Each state in the dominant leakage set will be simulated using the simulation engine described in Section II-B. Since the cost of simulation depends on the size of the DCC being solved, for each state we simulate with the reduced graph of each state instead of the full graph. In our example, the graph in Fig. 3(f) will be used by the simulation engine. If the standby (sleep) states of any of the primary inputs are known, we propagate those values to internal nodes of the circuit and use a graph reduction technique similar to that described in Step 2) before finding the full set of states. This additional information further reduces the total number of dominant leakage states.

B. Leakage Model and Leakage Simulation Engine

At the core of this approach is an accurate transistor-level leakage simulator which efficiently evaluates the leakage current of a given DCC for a given state. A number of methods have been proposed for quickly calculating an approximate leakage number for a stack of transistors [5], [7], [8], and [21]. These methods precalibrate 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. They avoid the need to perform iterations over the nonlinear device models in order to converge on the node voltages and hence, they have very fast run times. However, the drain to source current (I_{ds}) of a device is highly nonlinear with the drain to source voltage (V_{ds}). A linear-scaling factor to account for the stack depth of the leaking transistors 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 Fig. 2, for instance, a V_t drop develops across transistor N1 when the state **100** is applied. 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 μ process.

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}. Our device current model is specifically targeted at efficient leakage simulation in the course of leakage optimization through simultaneous transistor sizing and V_t selection. With V_{t0}, L and other process parameters fixed, the drain current of a given type of MOS device is described with the nonlinear function $I_d = f(V_d, V_s, V_g, W)$, where V_d, V_s, V_g are the drain, source and gate voltages and W is the width of the device. As the V_g for a device is either Vdd or 0 during leakage simulation, I_d is captured in two three-dimensional (3-D) tables, one for each value of V_g. These tables are derived through precharacterization using SPICE simulations with accurate models. When the reduced graph contains only Vdd and Gnd nodes as in Fig. 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. Since the computationally expensive model evaluation calculations are shifted from within the iterations to the characterization phase, simulation speeds improve dramatically. Table II shows the comparison between our fast leakage simulation and SPICE for a three-input NAND gate. The table shows that accuracies within 2% of SPICE are achieved for all possible states of the gate.

C. Calculation of State Probabilities

In [16], the authors describe a procedure for propagating the probabilities of primary inputs to all internal (and primary output) signals while accounting for the first-order spatial correlations between signals. The procedure uses correlation coefficients between every pair of signals. The coefficients are propagated along with the probabilities. We apply this procedure to calculate the probabilities and correlation coefficients of all signals in the circuit and thereby calculate the state probabilities of the DCCs. The Boolean functions for the DCC nodes are extracted and represented as SP-BDDs [17]. Since the original approach can only be applied to acyclic graphs, the procedure is modified as follows to handle sequential circuits. First, the network is represented as a directed graph wherein the nodes are the primary inputs, primary outputs, or DCCs of the circuit and the edges are the connections between them. Second, the directed graph is levelized to identify the back-edges (feedback edges from higher level nodes to lower level nodes). The back-edges are assigned initial values for signal probabilities and correlation coefficients. The procedure

TABLE II
NAND3 LEAKAGE MEASUREMENT RESULTS USING NEWTON–RAPHSON

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

in [16] is then applied in an iterative loop on the whole circuit until the updated values for the back-edges converge, or until a maximum limit on the number of iterations is reached.

The methods presented in [5] and [14] use statistical simulation to measure the average leakage of an entire circuit. These statistical methods are based on Monte Carlo experiments, where in each iteration, a randomly picked circuit state is applied and the leakage of the entire circuit is computed. The iteration is repeated until the average leakage of the circuit, computed over all applied vectors, converges. However, a leakage-optimization tool relies on accurate estimation of leakage for the individual DCCs rather than estimates for the total circuit leakage when calculating leakage sensitivities. While Monte Carlo-based methods may converge quickly on the circuit's total leakage, they are not effective in determining the average leakage of individual DCCs. Since the leakage of an individual DCC can vary widely over its circuit states (as much as 99X in a NAND3 gate, as seen in Table II), it is necessary to simulate a substantial portion of its states to obtain an accurate average leakage of each gate. Since the number of gates in a circuit is usually very large, it would require an extremely large number of random global circuit vectors to sufficiently cover the states of the individual DCCs and for their average leakage to converge. For this reason, the probabilistic approach is more effective than statistical simulation for optimization purposes. The probabilistic approach described above accounts for the leakage of *every* state of *every* DCC, provided such leakage is significant (i.e., dominant). Thus, fewer states are explored than by statistical simulation, but with a very high confidence on the accuracy of the estimated average leakage.

III. SIMULTANEOUS V_t SELECTION AND SIZING OPTIMIZATION

We now describe the method of optimization in Duet wherein we determine the size (width) and threshold voltage for each transistor (or each group of transistors) in a given circuit such that its area, performance, and leakage current are optimal. We cast the problem as a constrained optimization problem and pose the question, "What are the sizes and threshold voltages for the transistors that obtain the best performance/leakage tradeoff for the circuit without exceeding a given total area?" By repeatedly answering this question for different circuit areas, a 3-D tradeoff between area, performance, and leakage can be generated. Both the performance of the circuit and its leakage vary nonlinearly with device widths and their V_t's. Moreover, the width domain is continuous while the V_t domain is discrete as only two choices are available: high V_t or low V_t. Thus, finding an exact optimum solution would require solving an integer nonlinear program. This is prohibitively expensive even for circuits of moderate size. Hence, we need to take a heuristic approach.

Let $C(W, V)$ denote a circuit with transistor widths $W = [W_i]$, $w_{i,\min} \leq w_i \leq w_{i,\max}$ and subthreshold voltage levels $V = [v_i]$, $v_i = \{0, 1\}$, where w_i is the width of the ith transistor and V_i represents its threshold voltage level (low if 0 and high if 1). For the set of all optimal solutions with a given total area ($\sum w_i = \text{constant}$), 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 (smallest delay). These two solutions are illustrated in Fig. 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 Fig. 4(a), the intermediate solutions on the horizontal segment between the two boundary points are generated by repeating two basic steps 1) changing the V_t of some transistors to their low value and 2) rebalancing the circuit by reducing the sizes of a selected set of transistors and resizing the circuit back to its original area. These two steps are repeated until the performance target is met. Fig. 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 sensitivities account for the effects of both the increased drive strength and the increased gate capacitance on critical and near-critical paths in the circuit. By lowering V_t in this step, circuit delay decreases while area remains the same and leakage increases (Fig. 4, graph segment labeled V_t *lowering*). The second step is aimed at recovering additional performance by redistributing the area optimally after the V_t changes. In the first part of this step, the sizes of a selected set of transistors are reduced, causing a possible slight increase in delay, and a decrease in both the total circuit area and leakage (Fig. 4, graph segment labeled *Size Reduction*). In the second part, the circuit is resized to its original area, causing a decrease in delay, an increase in total circuit area to its original size and an increase in leakage (Fig. 4, graph segment labeled *Resizing*). This two-tier approach is warranted by the differing natures of the domains of transistor size and V_t, the former being continuous and the latter discrete. The steps are detailed in the following sections.

A. Threshold Voltage Selection

The difficulty in optimizing the threshold voltage of transistors in a circuit is that lowering the threshold voltage of a particular transistor has both positive and negative impact on circuit performance. On the one hand, the drive strength of the transistor is significantly increased, resulting in a much faster switching delay. On the other hand, the gate capacitance of the transistor is increased (due to the increase in the length of time

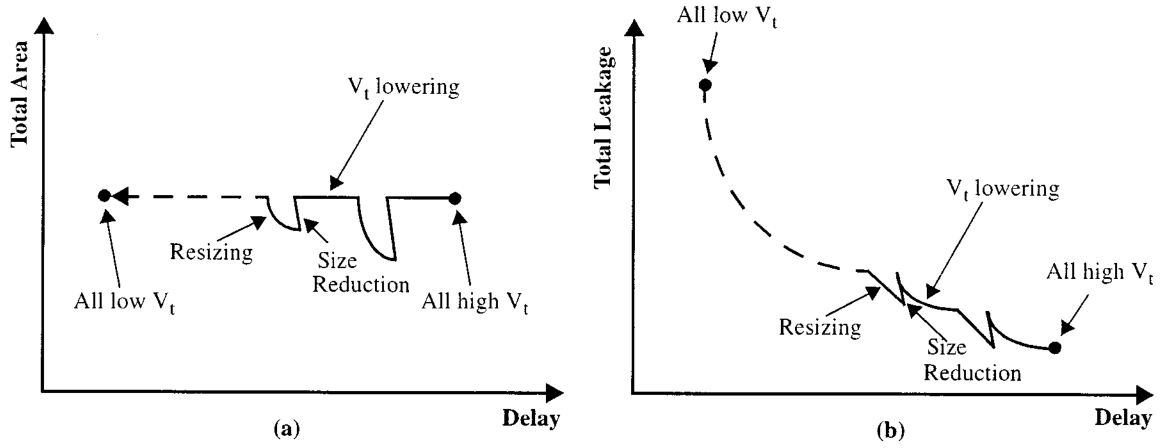


Fig. 4. V_t selection and redistribution of area, in two domains.

during which the channel is formed) 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 [5] 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 suboptimal solution.

We propose an iterative approach that uses a merit function to evaluate the increase in total leakage with respect to the performance gain of the whole circuit. In each iteration, the merit function is calculated for all transistors in the circuit and the transistor with the best merit is selected and is set to low V_t. The circuit sizes are then rebalanced as explained in Section III-B, the circuit timing and transistor merit is incrementally recalculated and the procedure is repeated. The merit function is shown below

$$\text{Merit}(T) = \frac{\Delta I_{\text{sub}}(T)}{\Delta D(T)}$$
$$\text{where } \Delta D(T) = \sum_{\text{arcs}}^{\alpha} \Delta d_\alpha(T) \cdot \frac{1}{k + \text{Min(slacks)} - \text{slack}_\alpha}.$$

The V_t change of a transistor directly impacts the delay of a number of timing arcs in its own gate and in the gate driving its gate node, due to added capacitive loading. The impact of the V_t change of a transistor $\mathbf{T}$ on a particular timing arc 2 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 weighting function $1/(k + \text{Min(slacks)} - \text{slack}_\alpha)$, where k is a small negative number and Min (slacks) 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, a small two–input NAUD circuit is shown in Fig. 5(a), with its associated four-timing arcs listed in Fig. 5(b). If we consider changing the V_t of transistor N2, this will affect the two-top-timing arcs: $A(R) \rightarrow \text{Out}(F)$ and $B(R) \rightarrow \text{Out}(F)$. The weighted sum $\Delta\mathbf{D}(\mathbf{N2})$ is now computed as the sum of the sensitivities of the two-timing arcs weighted by their criticality. If either timing arc lies on the critical path and is improved in performance by lowering the V_t of transistor N2, but if the other timing arc lies on a near-critical path and 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 calculation of $\Delta\mathbf{I_{sub}}(\mathbf{T})$ and $\Delta\mathbf{D}(\mathbf{T})$ is explained in more detail below.

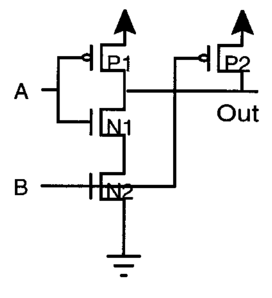
The factor $\Delta\mathbf{d}_\alpha(\mathbf{T})$ is the change of the delay of timing arc α in the circuit due to the change in the V_t of the transistor T. This is calculated using an analytical function based on the Elmore delay model similar to that in [18]. Changing the V_t of a transistor has two pertinent effects.

1) The effective resistance of the transistor is reduced. Its impact is simply expressed as $\Delta\mathbf{d}_{\alpha,1}(\mathbf{T}) = \Delta\mathbf{R}(\mathbf{T})^*\mathbf{C}_{\mathbf{drain},\alpha}(\mathbf{T})$, where $\Delta\mathbf{R}(\mathbf{T})$ is the change in effective resistance of transistor T and $\mathbf{C}_{\mathbf{drain},\alpha}(\mathbf{T})$ is the total switched (Elmore) capacitance from the drain terminal of transistor T to the output node of timing arc α.
2) The gate capacitance of the transistor is increased. If a timing arc α ends at the gate node of transistor T, then the impact on timing due to a V_t change of the gate of transistor T is expressed as $\Delta\mathbf{d}_{\alpha,2}(\mathbf{T}) = \mathbf{R}_{\mathbf{effect},\alpha}{}^*\Delta\mathbf{C}_{\mathbf{gate}}(\mathbf{T})$, where $\mathbf{R}_{\mathbf{effect},\alpha}$ is the effective resistance of timing arc α and $\Delta\mathbf{C}_{\mathbf{gate}}(\mathbf{T})$ is the change in the gate capacitance of transistor T

The total impact of the V_t change is: $\Delta d_\alpha(\mathbf{T}) = \Delta d_{\alpha,1}(\mathbf{T}) + \Delta d_{\alpha,2}(\mathbf{T})$.

Since $\Delta\mathbf{d}_\alpha(\mathbf{T})$ is calculated analytically, the evaluation of $\Delta\mathbf{D}(\mathbf{T})$ is extremely efficient. The Elmore delay model uses extensive *a priori* SPICE simulation to calibrate the effective resistance of a transistor as a function of its width, input slope, output load, and position relative to the switching transistor. 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 [19].

The factor $\Delta\mathbf{I_{sub}}(\mathbf{T})$ 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, estimating the



(a)

A	B	Out
R	1	F
1	R	F
F	1	R
1	F	R

(b)

Fig. 5. A two-input NAND2 gate and its four-timing arcs.

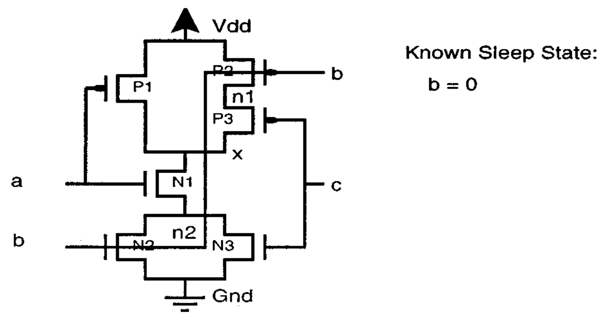


Fig. 6. Use of sleep state information.

average leakage of the DCC using the procedure in Section II, and taking the difference with the leakage of the DCC before the V_t was lowered. The dominant leakage states of a DCC are independent of the V_t settings in the circuit and therefore, are not recalculated. Furthermore, only those dominant leakage states containing transistor T in their reduced graphs need to be resimulated. This significantly reduces the cost of calculating $\Delta\mathbf{I_{sub}}(\mathbf{T})$ and makes it feasible to calculate $\Delta\mathbf{I_{sub}}(\mathbf{T})$ for each transistor in each iteration of optimization.

After the merit function is calculated for all transistors, the transistor with the minimum merit 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. In the case where multiple transistors have Merit $(\mathbf{T_i}) = 0$, the transistor that has the maximum $\Delta\mathbf{D}(\mathbf{T})$ is selected. In the special case, where lowering the V_t of any transistor in the circuit only increases the delay of the circuit (i.e., $\Delta\mathbf{D}(\mathbf{T}) < 0$ for all $\mathbf{T_i}$), the optimization stops and the current solution is reported as the best possible solution for the circuit.

The proposed approach can be easily extended to standard cell designs. In such cases, the standard cell library usually has two versions of a cell: a low-V_t cell (all transistors in the cell are low V_t) and a high-V_t cell (all transistors in the cell are high V_t); or four versions of a cell: a low-V_t cell, a high-V_t cell, a high-V_t PMOS/low-V_t NMOS cell (all PMOS transistors in the cell are high V_t and all NMOS transistors are low V_t) and a low-V_t PMOS/high-V_t NMOS cell. In each iteration, a cell is selected to be changed to its lower-leakage version based on the merit function: Merit $(C) = (\Delta I_{\text{sub}}(C))/(\Delta D(C))$, where the values $\Delta\mathbf{I_{sub}}(\mathbf{C})$ and $\Delta\mathbf{d}_\alpha(\mathbf{C})$ for each cell $\mathbf{C}$ are calculated with respect to the change of a cell, not the change of an individual transistor.

B. 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. Both the speed and gate capacitance of such transistors are changed, affecting all incident timing paths and increasing the load on nearby transistors. The decrease in delay results in excess area which can be recovered from now-oversized devices. The increase in capacitance may result in undersized devices which require additional area in order to meet their timing requirements. By shifting the excess area to undersized regions, we can improve performance without an area penalty. We call the process of adjusting device widths while maintaining total circuit area *rebalancing* and we accomplish it in two steps 1) reducing selected transistor widths and 2) resizing the entire circuit back to its original area.

The first step of rebalancing is reduction, the removal of excess area from devices, which are faster than necessary. Lowering the V_t of a transistor can easily change its speed by as much as 50% (depending on the process). If a circuit is balanced prior to a V_t change (all its timing paths are as near to their constraints as possible), such a localized speed increase almost invariably introduces imbalance among paths. Removing area from selected devices is the first step toward correcting the imbalance. The set of reduction candidates includes any device sharing a timing path with a V_t-changed device and the V_t-changed device itself. In the extreme case, all candidates in the set of reduction are set to their minimum sizes. While effective, this technique's aggressiveness unnecessarily shifts much of the work to the higher-complexity resizing phase. Instead, we can use the locality of the effect of a V_t change to isolate the reduction. Changes in the speed and capacitance of a device naturally have a greater effect on nearby devices, with diminishing strength further away as more intermediate devices buffer the effect. Therefore, we can identify a the core of influence of a V_t change 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, or optimally distributing excess area (the area gained during reduction) in order to decrease worst circuit delay. We use a delay/area sensitivity-based size optimization tool for the resizing step [20]. This tool balances the delay 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 crit-

TABLE III
BENCHMARK DETAILS, LEAKAGE MEASUREMENTS

Circuit Characteristics						**Leakage Current**			**Run-time**	
Name	**No. of input**	**No. of FETs**	**No. of circuit states**	**No. of dom. states**	**No. of solve states**	**Spice (nAmp)**	**Ours (nAmp)**	**Diff (%)**	**Spice (sec)**	**Ours (sec)**
blk	9	108	512	78	30	59.95	58.87	-1.79	375	< 1
bay	9	68	512	52	16	652.44	703.14	7.21	372	< 1
add1	10	244	1024	171	71	488.86	472.27	-3.39	2135	1
pla	12	1052	4096	809	246	3274.05	2987.40	-8.76	12415	2
add2	51	1090	2.3e15	646	270	N/A	554.61	N/A	N/A	2
add3	65	1256	3.69e19	1362	300	N/A	418.95	N/A	N/A	1
control1	91	5318	2.5e27	4387	966	N/A	5668.49	N/A	N/A	2

ical timing paths. This approach can also be extended to standard cell designs by selecting gates rather than transistors. The size reduction (or increase) of each cell then involves swapping the cell to its smaller (or larger) drive strength versions.

The computational complexity of the size-reduction step is linear, as the number of V_t changes is linear in circuit size and every V_t change incurs only a constant amount of size reduction work. It is key to note that the size reduction effort, limited to a fixed depth by the cone of influence, does not depend on the circuit size. Instead, it depends on the connectivity of the circuit and is, therefore, nearly constant. The resizing step, on the other hand, is applied globally and can be computationally expensive. However, we use incremental timing and incremental sensitivity calculation in this step, resulting in near-linear complexity. The complexity of the whole rebalancing step is, therefore, near linear.

C. Known State Information

Leakage power optimization is normally of concern in circuits designed for portable applications. These circuits spend the majority of time in sleep or standby mode. The leakage estimation of such a circuit in sleep mode only is therefore, a good approximation of the entire leakage power (which includes both standby mode and active mode leakage powers). In a sleep mode, a circuit is often brought to a fully or partially known input state through a series of one or more sleep operations. These known input states can be used to help both the leakage estimation and the leakage optimization. In our leakage estimation, known input-signal states are propagated to internal-circuit nodes, resulting in modified state probability for each gate input and more accurate leakage estimation. In the optimization approach, this additional information is seamlessly incorporated. Since a transistor that is ON in the sleep state does not contribute to the leakage of the gate, changing its V_t does not change the leakage of the circuit or changes it only by a negligible amount (due to the change in the V_t drop across the ON transistor that is in series with a leaking OFF transistor). For such a transistor, $\Delta\mathbf{I_{sub}}(\mathbf{T})$ will be zero (or near zero) and hence, its merit will be zero (or near zero), making it a likely candidate for V_t lowering. For example, if P2 in Fig. 6 is ON during sleep state, its V_t can be lowered without impacting the leakage current. Therefore, eliminating the impossible state of $b = 1$ improves the accuracy of leakage current estimation and the optimization quality. Runtime is also reduced, as fewer states need to be simulated. Note that known input states in active mode can be used in the same way if the circuit is being optimized for its active-mode leakage power.

IV. RESULTS

We have implemented the proposed leakage measurement and threshold voltage/size optimization algorithms in a tool called Duet. Duet has been used for industrial low-power DSP processor and micro controller design and has been successfully run on a large number of circuits. The results shown in this section are from a 0.18 μ process with $V_{t0,n} = 0.48$ and $V_{t0,p} = -0.52$ for high V_t devices and $V_{t0,n} = 0.33$ and $V_{t0,p} = -0.39$ for low V_t devices. The transistor length is calculated using a statistical model which takes into account the across-chip length variation.

A. Leakage Current Estimation

Table III 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 two and three show the number of inputs and the number of transistors in the circuit, respectively. Column four shows the number of circuit states which would have to be individually simulated in an exhaustive approach. Column five shows the number of dominant states and column six shows the actual number of states solved with our nonlinear solver. Columns 7–11, show the average leakage measurement results and run times.

The measurements in column seven (*leakage current–spice*) were obtained by exhaustively simulating each circuit over all possible input combinations and then taking the average

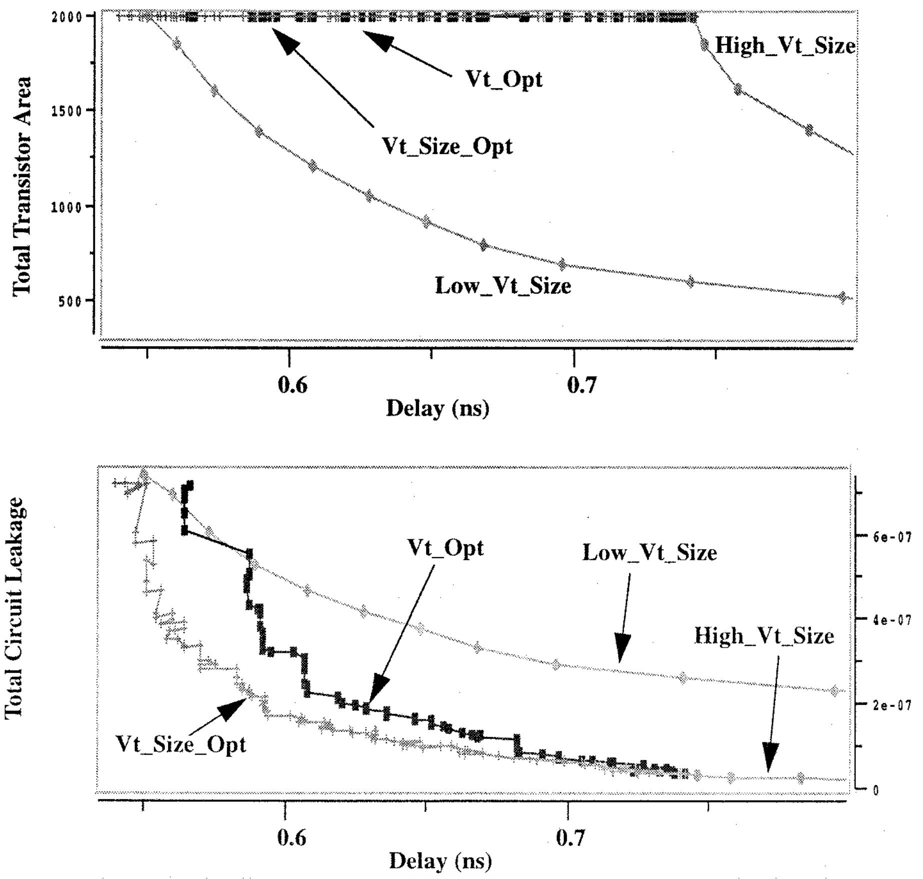


Fig. 7. Area versus delay and leakage versus delay tradeoff curves for circuit add1.

leakage. Thus, these values take into consideration the correlation between internal nodes and give the exact average leakage. The measurements in column eight (*leakage current–ours*) are from our approach and are compared in column nine with the SPICE measurements. For these circuits, our approach took less than two seconds (on a Sun Ultrasparc 60) to calculate the leakage. This amounts to a more than 6000x speedup over exhaustive SPICE simulation. Also, note that for the circuits *add2, add3,* and *control 1* , it is infeasible to run exhaustive SPICE simulations, as it would require 2.3e15, 3.69e19, and 2.5e27 simulation runs, respectively. The speedup in our approach demonstrates the efficacy of using the concept of dominant leakage states combined with state probabilities. The results also show the high accuracy of our method, which is within 9% of SPICE.

B. *Simultaneous V_t and Size Optimization*

We also benchmarked our simultaneous V_t selection and size optimization algorithm on the example circuits. Fig. 7 shows *area versus delay* and *leakage versus delay* tradeoff curves for example circuit *add1*.

Duet generates a series of tradeoff solutions and allows the user to interactively choose the final solution based on the leakage, delay, and area constraints. In Fig. 7, four different optimization scenarios are compared: 1) *High_Vt_Size*: All transistors are fixed at high V_t and the circuit is sized for performance using standard sizing algorithms. 2) *Low_Vt_Size*: All transistors are fixed at low V_t and the circuit is sized for performance. 3) *Vt_Size_Opt*: This is the approach proposed in this paper. Starting from the fastest all-high V_t solution, small sets of transistors are iteratively selected and changed to low V_t and circuit sizes are rebalanced. (4) *Vt_Size_Opt*: This is identical to *Vt_Size_Opt*, but no rebalancing is done after changes in V_t.

Quantitative optimization results are shown in Table IV. The columns *High_Vt_Size* and *Low_Vt_Size* show the delay and leakage of the circuit sized for performance with all high– and low-V_t transistors, respectively. Columns for *Vt_Opt* and *Vt_Size_Opt* are solutions from the Duet optimization which have reasonable tradeoffs in terms of delay and leakage. For *Vt_Size_Opt*, we also show the percentage of delay increase and the relative leakage-reduction factor (leakage of *Low_Vt_Size*/leakage of *Vt_Size_Opt*) with respect to the *Low_Vt_Size* case. In the case of circuit *control 2*, only the optimization with combined V_t assignment and transistor sizing was performed.

As expected, *High_Vt_Size* exhibits very low leakage but the circuit speed is also the slowest. On the other hand, *Low_Vt_Size* can achieve much faster circuit speed at the cost of a significantly higher leakage. In the *Vt_Size_Opt* curve, 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_Vt_Size*, but with considerably lower leakage.

TABLE IV
V_t AND SIZE OPTIMIZATION RESULT

		High_Vt_Size		Low_Vt_Size		Vt_Opt		Vt_Size_Opt	
Circuit name	No. of FETs	Delay (ns)	Leakage (uA)	Delay (ns)	Leakage (uA)	Delay (ns)	Leakage (uA)	Delay (% increase over Low_Vt) (ns)	Leakage (reduction factor over Low_Vt) (uA)
blk	108	0.33	0.002	0.25	0.048	0.25	0.037	0.25 (0%)	0.011 (4.4x)
bay	68	0.52	0.022	0.39	0.48	0.42	0.37	0.42 (3%)	0.14 (3.4x)
add1	244	0.74	0.03	0.55	0.74	0.59	0.32	0.59 (7%)	0.17 (4.4x)
pla	1052	1.09	0.09	0.74	1.98	0.77	0.55	0.77 (4%)	0.32 (6.2x)
add2	1090	1.58	0.018	1.18	0.40	1.25	0.25	1.25 (6%)	0.13 (3.1x)
add3	1256	1.68	0.027	1.21	0.60	1.31	0.35	1.31 (8%)	0.19 (3.2x)
control1	5318	1.53	0.13	0.96	2.84	1.14	1.38	1.14 (19%)	0.63 (4.5x)
control2	34992	6.48	1.32	5.79	28.33	N/A	N/A	5.88 (1.6%)	1.80 (15.7x)

A comparison of results from *Vt_Size_Opt* and *Vt_Opt* demonstrates the benefit of rebalancing the circuit after each V_t change operation. Table IV shows that *Vt_Size_Opt* can achieve the same delay target with 1.8–3.5 times less leakage than *Vt_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.

It is interesting to note that a mixed-V_t optimization can actually produce a faster circuit than an all-low-V_t circuit can, as shown in Fig. 7. This behavior is explained by the higher gate capacitance of low-V_t devices on noncritical paths that may increase the loading seen by timing critical paths, thus making the circuit slower than a mixed-V_t solution.

The run times for the optimization were also reasonable. A large circuit, *control1*, has 5318 transistors and was successfully optimized within 1.5 CPU h. In this circuit, the results for *Vt_Opt* and *Vt_Size_Opt* show significant increase in delay (19%) because the points were chosen such that they meet timing constraints while leakage increase is minimized.

Finally, Fig. 8 demonstrates how knowledge of the sleep state of a circuit can improve the circuit optimization. Curves SLEEP_MODE and ACTIVE_MODE are the cases where circuit *add2* is optimized for sleep mode operation (with sleep state information) and active mode operation (without sleep state information), respectively. Note that the leakage measurements for the ACTIVE_MODE curve account for the sleep states, while the optimization itself was done without sleep state information. The curve SLEEP_MODE generally has significantly lower leakage than the curve ACTIVE_MODE for a given delay. It is clear that the optimization benefits considerably from using sleep state information.

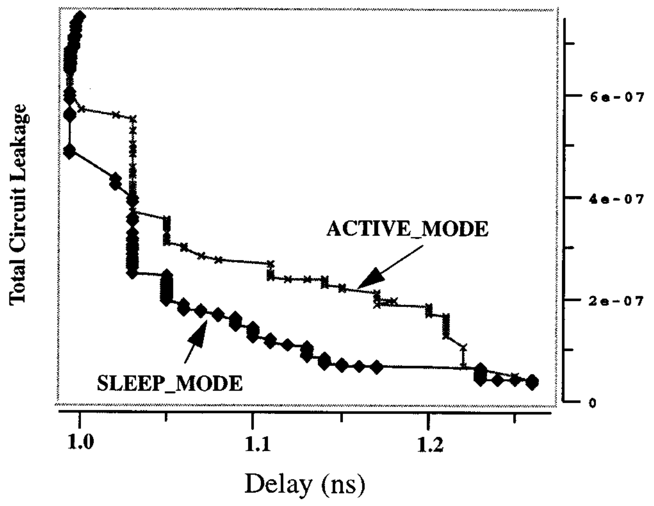


Fig. 8. Optimizations with- and without-sleep state information.

V. CONCLUSION

We have presented an efficient technique for accurately estimating the average standby 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 and seemlessly exploits sleep states to optimally tradeoff standby power and performance in dual V_t circuits. The benefits of combining V_t selection and transistor sizing over the earlier proposed approaches of doing only V_t selection were demonstrated. We have also shown the potential of using sleep state information during leakage optimization. Test results showing the accuracy and speedup improvement of our estimation procedure and the performance versus standby-power tradeoff were also given.

REFERENCES

[1] K. Fujii *et al.*, "A sub-1V triple-threshold CMOS/SIMOX circuit for active power reduction," in *ISSCC Dig. Tech. Papers*, Feb. 1998, p. 190.

[2] N. Rohrer *et al.*, "A 480 MHz RISC microprocessor in a 0.12 μm Leff CMOS technology with copper interconnects," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 1998.

[3] Y. Oowaki *et al.*, "A sub-0.1 μm circuit design with substrate-over-biasing," in *ISSCC Dig. Tech. Papers*, Feb. 1998, p. 88.

[4] J. Kao, A. Chandrakasan, and D. Antoniadis, "Transistor sizing issues and tool for multi-threshold CMOS technology," in *Proc. Design Automation Conf.*, 1997, pp. 409–414.

[5] L. Wei *et al.*, "Design and optimization of low voltage high performance dual threshold CMOS circuits," in *Proc. 35th Design Automation Conf.*, 1998, pp. 489–494.

[6] L. Wei , Z. Chen , and K. Roy, "Mixed-Vth (MVT) CMOS circuit design methodology for low power applications," in *Proc. 36th Design Automation Conf.*, 1999, pp. 430–435.

[7] Q. Wang *et al.*, "Static power optimization of deep submicron Cmos circuits for dual V_t technology," in *Proc. ICCAD*, 1998, pp. 490–496.

[8] Z. Chen *et al.*, "Estimation of standby leakage power in CMOS circuits considering accurate modeling of transistor stacks," in *Proc. Int. Symp. Low Power Electron. Design*, 1998, pp. 239–244.

[9] P. Antognetti, "CAD model for threshold and subthreshold conduction in MOSFET's," *IEEE J. Solid-State Circuits*, vol. SC-17, pp. 454–458, June 1982.

[10] B. J. Sheu *et al.*, "BSIM: Berkeley short-channel IGFET model for MOS transistors," *IEEE J.Solid-State Circuits*, vol. SC-22, pp. 558–566, June 1987.

[11] M. D. Godfrey, "CMOS device modeling for subthreshold circuits," *IEEE Trans. Circuits Syst. II*, vol. 39, Aug. 1992.

[12] E. A. Vittoz, "Analog VLSI signal processing: Why, where and how?," *J. VLSI Signal Processing*, vol. 8, pp. 27–44, 1994.

[13] M.-J. Chen *et al.*, "A three-parameters-only MOSFET subthreshold current CAD model considering back-gate bias and process variation," *IEEE Trans. Computer-Aided Design* , vol. 16, pp. 343–352, 1997.

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

[15] P. Pant *et al.*, "Device-circuit optimization for minimal energy and power consumption in CMOS random logic networks," in *Proc. 34th Design Automation Conf.*, June 1997, pp. 403–408.

[16] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Ricco, "Testability measures in pseudorandom testing," *IEEE Trans. Computer-Aided Design*, vol. 11, pp. 794–800, 1992.

[17] A. L. Glebov *et al.*, "Transistor reordering for low power CMOS gates using an SP-BDD representation," in *Proc. Int. Symp. Low Power Design*, 1995, pp. 161–166.

[18] J. P. Fishburn *et al.*, "TILOS: A posynomial programming approach to transistor sizing," in *Proc. ICCAD*, Nov. 1985, pp. 269–273.

[19] A. Dharchoudhury *et al.*, "Fast and accurate timing simulation with regionwise quadratic models of MOS I-V characteristics," in *ICCAD*, Nov. 1994, pp. 190–194.

[20] A. Dharchoudhury, D. Blaauw, T. Noston, S. Pullela, and T. Dunning, "Transistor-level sizing and timing verification of domino circuits in the powerPC™ microprocessor," in *ICCD*, Oct. 1997, pp. 143–148.

[21] M. C. Johnson *et al.*, "Models and algorithms for bounds on leakage in CMOS circuits," *IEEE Trans. Computer-Aided Design*, vol. 18, pp. 714–725, June 1999.

[22] M. C. Johnson, D. Somasekhar, and K. Roy, "Leakage control with efficient use of transistor stacks in single threshold CMOS," in *Proc. 36th Design Automation Conf.*, 1999, pp. 442–445.

Supamas Sirichotiyakul (M'99) received the B.Eng. degree from Chulalongkorn University, Bangkok, Thailand, and the M.S. degree in computer engineering from the University of Louisiana at Lafayette.

From 1995 to 2001, she worked in CAD Research and Development with the Advanced Tools Group at Motorola, Inc., Austin, TX. Currently, she is a Member of the Technical Staff in CAD with Sun Microsystems, Inc., Chelmsford, MA.

Tim Edwards received the M.A. degree in computer science from The University of Texas at Austin in 1994.

In 1995, he became a staff scientist with the Semiconductor Products Sector at Motorola, where he designs and develops CAD applications in the areas of design analysis and optimization.

Chanhee Oh (S'87–M'95) received the B.S. degree from Seoul National University, Korea, in 1987 and the M.S. and the Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin, in 1989 and 1995, respectively.

In 1997, he joined the Advanced Tools Group, Motorola Inc., Austin, TX, where he is currently a Senior Principal Engineer involved in the development of EDA tools and methodology for high-performance VLSI designs. Previously, he was with Advanced Micro Devices, Austin, TX. His research interests include signal integrity, reliability, timing analysis and optimization of VLSIs.

Rajendran Panda (S'94–M'96) received the B.E. (Honors) degree in electrical engineering from Madurai University, India, the LL.B. (Bachelor of Laws) degree from Bangalore University, India, and the Ph.D. degree in Computer Science from the University of Illinois at Urbana-Champaign, in 1978, 1984, and 1996, respectively.

From 1978 to 1991, he was with Bharat Heavy Electrical Industries, Bangalore. He is currently with Motorola, Inc., Austin, TX, managing the High Performance Design Technology group. His research focus has been on low power design, power supply and signal integrity issues, and circuit optimization. He has published extensively and holds many patents in these areas.

David Blaauw (M'94) received the B.S. degree in physics and computer science from Duke University in 1986 and the M.S. and Ph.D. degrees, both in Computer Science from the University of Illinois, Urbana-Champaign, in 1988 and 1991, respectively.

Until 1993, he was a Development Staff Member with the Engineering Accelerator Technology Division, IBM Corporation, Endicott, NY. From 1993 to 2001, he was Manager of the High Performance Design Technology group, Motorola, Inc., Austin, TX. Since 2001, he has been an Associate Professor with the University of Michigan, Ann Arbor. His research has focused on VLSI design and CAD with particular emphasis on circuit analysis and optimization problems for high performance and low-power designs.

Dr. Blaauw received the Best Paper Award at the ACM/IEEE Design Automation Conference in 2000. In 2000 and 2001, he was Technical Program Cochair as well as a member of the Executive Committee the ACM/IEEE Design Automation Conference. In 1999 and 2000, he was Technical Program Chair and General Chair for the International Symposium on Low-Power Electronic and Design.