Static Power Optimization of Deep Submicron CMOS Circuits for Dual V_T Technology

Qi Wang and Sarma B. K. Vrudhula

Center for Low Power Electronics
ECE Department, University of Arizona
Tucson, AZ 85721

Abstract

In this paper we address the problem of delay constrained minimization of leakage power of CMOS digital circuits for dual V_T technology. A novel and efficient heuristic alogrithm based on circuit graph enumeration is proposed. The experimental results on the MCNC91 benchmark circuits show that up to an order of magnitude power reduction can be achieved without any increase in delay.

1 Introduction

CMOS has long been considered the technology of choice for low power applications. The continuous shrinking of feature sizes has made it possible to achieve ever greater integration of complex functions on a single chip. This capability has also fueled an explosive growth in the market for high performance portable computing and communications systems. However, the higher chip densities have resulted in a one to two orders of magnitude increase in the power consumption of many high-end processors. The point is rapidly being reached where reduction of power consumption becomes the single most important hurdle that designers and manufacturers must face.

Power consumption in CMOS circuits can be expressed as the sum of the (average) *switching power* (P_{sw}), the *short-circuit power* (P_{sc}) and the *leakage power* (P_{leak}). P_{sw} is due to the charging and discharging of load capacitances as logic gates transition between 0 and 1. It is typically expressed as $C_L V_{dd}^2 E(t)$, where C_L is the load capacitance, V_{dd} is the supply voltage and $E(t)$ is the expected number of times that the gate switches. P_{sc} is due the existence of a conducting path between the V_{dd} and ground during the brief period when a gate switches, and P_{leak} is due to the leakage current caused by the stored charge in the drain junctions leaking away and due to devices that conduct while in the off-state (subthreshold conduction). With relatively larger devices, i.e., significantly larger than $1\mu m$, P_{sw} is the dominant component. The quadratic dependence of P_{sw} on V_{dd} indicates that reducing the supply voltage will have the greatest impact on reducing P_{sw}. Additionally, C_L is reduced by the reduced dimensions of the devices and by circuit design and layout techniques. Finally, a significant number of results have been reported on techniques to reduce the switching activity ($E(t)$).

Scaling down the supply voltage has the most significant impact on the power dissipation. This also avoids hot-carrier effects in short channel devices. However, the threshold voltage V_T has also to be scaled down because otherwise it has a much greater detrimental impact on the delay when small geometry devices are used [4]. Thus scaling V_T by the same factor as V_{dd} is needed so as not to adversely impact delay. However, reducing V_T in small geometry MOSFETs results in a exponential increase in the stand-by current [1]. Simulation results given in [5] show that the power dissipation due to the standby current dominates the switching power at low threshold voltages. The standby current increases as the subthreshold swing increases, and the subthreshold swing increases with increased doping density and reduced gate length, both of which happen for small geometry devices.

It is now clear that optimal design of CMOS circuits that employ submicron devices requiring operation at low voltages involves a number of complex tradeoffs, involving device dimensions, the supply voltage, and the threshold voltage. One relatively recent development is the use of multiple threshold voltage CMOS (MTCMOS) [10], which is relatively easy to implement. If only two threshold voltages are considered (dual threshold CMOS - DTCMOS), then the threshold voltage of an appropriate subset of the devices can be assigned the higher threshold voltage by including an extra implant step [10].

In [9], an approach to simultaneously optimize the supply voltage, threshold voltage and transistor sizes assuming MTCMOS is presented. Their approach is based on first assigning delays to all the gates without violating the cycle time constraint. Then the optimal supply and threshold voltage and transistor width of each gate are determined so as to minimize power consumption. Although their algorithm has the flexibility of assigning different threshold voltages, the results reported were based on a single threshold voltage. Their results indicate significant reduction in total power dissipation and energy. It must be pointed out that their estimation of the leakage power does not take in account the signal probabilities, which could result in significant errors in the estimates.

In [10], an MTCMOS circuit structure is proposed and analyzed. The circuit consists of a network of transistors that have low V_T. The network is connected to ground through a high V_T *gating* transistor that is off during the inactive period and on during the active period. In this way the standby current is reduced. This approach introduces some complications for circuit design. For example, reverse conduction paths may exist which tend to reduce the noise margins or in the worst case, result in complete failure of the gate. Additionally, the leakage power dissipated when the system is in the active mode will not be reduced. Finally, extra chip area is required for the high V_T gating transistors and the associated routing of wires.

The availability of two or more threshold voltages on the same chip provides a new opportunity for circuit designers to make tradeoffs between power and delay. In this paper the problem of optimal assignment of threshold voltages to transistors in a CMOS logic circuit is defined, and an efficient algorithm for its solution is given.

This problem will be referred to as the *Dual V_T Selection* problem, since only two threshold voltages, a low V_T and a high V_T, are considered.

The rest of the paper is organized as follows. Section 2 contains a summary of the power and delay models recently investigated by other researchers. In Section 3, a formal statement of the *Dual V_T Selection* problem is given. A new algorithm to solve this problem is described in Section 4. Some implementation issues are discussed in Section 5. The effectiveness of the proposed algorithm is examined by carrying out extensive experiments on MCNC91 benchmark circuits. The results of these experiments are given in Section 6. Finally, conclusions and directions for future work are discussed in Section 7.

2 Preliminaries

In this section background material on models used for estimating power dissipation for short channel MOSFETs and the models used to compute delay are presented.

2.1 Leakage Power Model of MOSFET's

For a single NMOS(PMOS) device, the Berkeley Short-Channel IGFET model (BSIM) [1] is used to estimate the leakage power dissipation. In the BSIM model, the threshold voltage is expressed as:

$$V_T = V_{T,0} - \eta V_{dd}, \tag{1}$$

$$V_{T,0} = V_{FB} + \phi_s + k_1\sqrt{\phi_s - V_{BS}} - k_2(\phi_s - V_{BS}), \tag{2}$$

where V_{FB} is the flatband voltage, ϕ_s is two times the Fermi potential, V_{BS} is the substrate reverse bias, k_1 is the body-effect factor, and k_2 and η model the threshold lowering effects of short channel MOSFET's. The leakage current for NMOS transistors working in the week inversion region, i.e. $V_{gs} = 0$, is given by

$$I_s = I_0 exp((V_{gs} - V_T)/nV_t)(1 - exp(-V_{ds}/V_t)), \tag{3}$$

where V_t is the thermal voltage ($\approx 25mV$ at room temperature), n is the subthreshold slope coefficient, and $I_0 = \mu_0 C_{ox}(W/L)V_t^2 e^{1.8}$. The leakage current formula for a PMOS device is similar.

Equation (3) gives a simple formula for the leakage current for a single NMOS device. In CMOS logic gates consisting of series-parallel networks of PMOS and NMOS devices, the leakage current through devices in parallel can be taken to be the sum of the individual leakage currents. However, the leakage current through a series of MOSFETs requires careful analysis of different combinations of the on and off devices. In [5], simple analytical formulas for the leakage current through a stack of one, two and three MOSFETs are given. In addition, the leakage current for stacked NMOS devices is related to the single NMOS leakage current as follows.

$$I_{s1} : I_{s2} : I_{s3} = e^{\frac{\eta V_{dd}}{n V_t}} : 1 : 0.56, \tag{4}$$

where I_{s_i} (i=1,2,3) is the leakage current for i stacked MOSFET's. I_{s1} is given by (3). Equation (4) shows that the leakage power of a CMOS gate depends on the state of inputs and the threshold value of the corresponding transistor. With this in mind, consider a 2-input CMOS NAND gate shown in Figure 1. For simplicity the PMOS and NMOS transistors driven by the the same input are assumed to have the identical threshold voltages, although different NMOS transistors can have different threshold voltages. The leakage power of the gate under different input combinations is summarized in Table 1.

$I^A_{s1,n}$ and $I^B_{s1,n}$ are the leakage currents for the single NMOS device of input A and B respectively. These may be different due to their different threshold voltages. $I^A_{s1,p}$ is the leakage current of the single PMOS device of input A. This value can also be different from $I^A_{s1,n}$ since the two transistors may have different sizes. $I^{AB}_{s2,n}$ is the leakage current when both NMOS devices A and B are off and the output C is high. Although A and B may have different threshold voltages, for simplicity, $I^{AB}_{s2,n}$ is taken to be the smaller of $I^A_{s2,n}$ and $I^B_{s2,n}$. This is a conservative approximation but will not lead to significant errors since from (4) the leakage power of two series connected MOSFETs is much less than that of a single MOSFET. All these quantities can be obtained directly from (4).

The overall average leakage power dissipation can be expressed as follows:

$$P = V_{dd} * [p(\overline{AB}) * I^{AB}_{s2,n} + p(\overline{A}B) * I^A_{s1,n} + p(A\overline{B}) * I^B_{s1,n} + p(AB) * (I^A_{s1,p} + I^B_{s1,p})] \tag{5}$$

where $p(\bullet)$ are the signal probabilities for the different input combinations. To accurately estimate the leakage power, the exact probabilities for each combination have to be found. This may be achieved using BDDs. However, in most practical cases, the signal probabilities at the gate inputs and outputs are obtained by either local probability propagation or by logic simulation.

2.2 Delay Model

2.2.1 Gate Delay Model

An accurate and computationally efficient model for a short channel MOSFET is described in [4]. The model, called the nth power law, is an extension of the alpha-power law model [3], but is much more accurate. The nth power law model has been shown to accurately represent the $I - V$ characteristics of short channel MOSFET's down to 0.25-μm channel length. The CMOS inverter propagation delay and output transition delay formula derived from the MOSFET's model predicts the circuit behavior for modern submicrometer designs very well. For CMOS gate delays, it was found that N series-connected MOSFET's (SCMS) would show less than N times the delay compared to a single MOSFET for submicrometer designs [3]. That is,

$$\frac{delay(SCMS)}{delay(inverter)} = 1 + \zeta\ (N - 1) \tag{6}$$

where ζ is a technology dependent parameter and $0 < \zeta < 1$ for most current submicrometer technologies.

To compute the circuit delay, standard static timing analysis is used. For each gate n of a circuit we define three values: $AT(n)$, $RT(n)$ and $S(n)$, which are the arrival time, required time and slack for the gate n. The arrival time $AT(n)$ is the worst delay from the primary inputs to gate n. Given the arrival time at primary inputs, the arrival time of gate n is obtained by

$$AT(n) = \max_{\substack{\forall\ i \\ i\ \in\ fanins\ of\ n}} (AT(i) + d_i(n)), \tag{7}$$

where $d_i(n)$ is the pin-to-pin delay from the input i of gate n to the output of gate n. This quantity is computed using the nth power law model. Note that the interconnect delay is not considered in the delay computation. The required time is the latest time the signal has to arrive at the output of gate n. Given the required time at each primary output, the required time at the output of gate n is obtained by

$$RT(n) = \min_{\substack{\forall\ j \\ j\ \in\ fanouts\ of\ n}} (RT(j) - d_n(j)), \tag{8}$$

where $d_n(j)$ is the pin-to-pin delay from the inut of gate j that is fed from gate n to the output of gate j. This delay is also computed

using the nth power law model. The slack is defined as $S(n) = RT(n) - AT(n)$. The set of gates that has the minimal slack value constitute the critical path of the circuit. If no gate in the circuit has a negative slack, then timing constraints are satisfied [11].

3 Problem Definition

A combinational circuit is represented by a directed acyclic graph (DAG) $G = (V, E)$. Each node and edge in G corresponds to a gate and a connection in the circuit respectively. The general form of the dual-V_T optimization problem is to assign one of two threshold voltages, $V_{T,high}$ and $V_{T,low}$, to each transistor such that some cost function is optimized subject to constraints. Since the PMOS and NMOS transistors that are connected to the same signal have the same threshold voltage, the different threshold voltages of the transistors are represented by labeling each connection $e_{ij} \in E$ by x_{ij}, where $x_{ij} = 0$ ($x_{ij} = 1$) means that the PMOS and NMOS transistors that are driven by edge e_{ij} have a $V_T = V_{T,high}$ ($V_T = V_{T,low}$).

The dual-V_T selection problem can be viewed in one of two ways - either delay can be optimized subject to constraints on power or visa versa. The algorithm presented here attempts to reduced the standby power subject to the constraint of not increasing the delay. Thus, the procedure starts with a combinational circuit where all the devices are assumed to have their threshold voltage set to $V_{T,low}$ and selects a subset of devices whose threshold voltage will be changed to $V_{T,high}$, without increasing the delay. This is formally expressed as follows: *Given a combinational circuit, represented as a DAG $G = (V, E)$, and with all the devices having their threshold voltage set to $V_{T,low}$,*

$$maximize \sum_{\forall\, e_{ij} \in E} x_{ij} \Delta P_{e_{ij}}$$

$$subject\ to\ \ x_{ij} = 0, 1 \text{ and } S(n) \geq 0,\ \forall\, n \in V$$

where $\Delta P_{e_{ij}}$ is the reduction in the leakage power when the threshold voltage associated with edge e_{ij} is changed from $V_{T,low}$ to $V_{T,high}$. The required time for each primary output is defined to be the worst case delay of the original circuit where all the devices have their threshold voltage set to $V_{T,low}$. Thus, the initial circuit is the fastest implementation with all the other parameters being fixed. This is a constrained 0-1 programming problem with non-linear constraint functions. In the following section an efficient heuristic procedure to solve this problem is described. The effectiveness of the algorithm will demonstrated through experiments on the MCNC91 benchmark circuits. Note: The proofs of the Lemmas are simple and are ommitted here in the interest of brevity.

4 The Algorithm

Definition 1 *Let e_{ij} be an edge of $G = (V, E)$. e_{ij} is said to be feasible iff changing the threshold voltage of e_{ij} from $V_{T,low}$ to $V_{T,hig}$, does not result in making the slack of gates i and j negative.*

Given the delay information of the gates, the feasibility of an edge is determined as follows. An edge e_{ij} is feasible if its threshold voltage is $V_{T,low}$ and $\epsilon_f < S(j)$ and $\epsilon_b < S(i)$ where $\epsilon_f = AT(i) + d_i^H(j) - AT(j)$, and $\epsilon_b = RT(i) + d_i^H(j) - RT(j)$. $d_i^H(j)$ is the pin-to-pin delay when the threshold voltage of edge e_{ij} is $V_{T,high}$. Since non-feasible connections are guaranteed not be included in any selection, only feasible connections need to be considered.[1]

[1] Note for the of simplicity, although the output rise and fall delays have not been differentiated so far in the presentation, they are accounted for in the implementation.

Definition 2 *A weight $w(e_{ij})$ is assigned to each edge e_{ij} in $G = (V, E)$ as follows:*

$$w(e_{ij}) = \left\{ \begin{array}{ll} \Delta\ P_{e_{ij}} & \textit{if } e_{ij} \textit{ is feasible} \\ 0, & \textit{if } e_{ij} \textit{ is not feasible.} \end{array} \right\} \quad (9)$$

A solution to the dual-V_T problem is to identify the largest subset $S_H \subseteq E$ ($S_L = E - S_H$), such that changing the threshold voltage of the edges in S_H to $V_{T,high}$ will not violate the delay constraints. The heuristic procedure to be described consists of two steps. First, instead of finding the largest feasible subset of edges, a *maximal* feasible subset is determined. That is, a subset that has the property that if another edge is added to it, it is no longer feasible, i.e., it violates the delay constraint. A maximal feasible subset is a *locally* optimal solution. To escape from this with the intent of finding a possibly better one, a second step, called *swapping*, is carried out. This swaps elements from S_L and S_H to increase the weight of the maximal set. The weight of a set of edges is the total power reduction when the threshold voltage of all edges in the set is changed from $V_{T,low}$ to $V_{T,high}$.

4.1 Construction of an Initial Solution

Definition 3 *A* **cut** *C of a directed acyclic graph $G = (V, E)$ is a partition of the nodes of V into two disjoint sets, i.e. $C = (S, \overline{S})$. The* **forward cut edges** *are those edges $e_{ij} \in E$ such that $i \in S$ and $j \in \overline{S}$. Similarly, the* **backward cut edges** *are all those edges $e_{ij} \in E$ such that $i \in \overline{S}$ and $j \in S$. A* **forward cut** *C_f of G is a cut where all cut edges are forward cut edges.*

Lemma 1 *Let C_f be a forward cut of a DAG $G = (V, E)$. Let e_1 and e_2 be any two edges in the set of forward cut edges. Then e_1 is neither in the transitive fanin cone nor the transitive fanout cone of e_2.*

Lemma 2 *Given a forward cut of the circuit graph G, changing the threshold voltage of all the forward cut edges that are feasible from $V_{T,low}$ to $V_{T,high}$ will not increase the circuit delay.*

In the rest of the paper, a forward cut will be referred to simply as a cut. A simple algorithm to find a good intial solution can obtained by iteratively finding the maximum weighted cut of the circuit graph until the weight of the cut becomes zero. Note that after changing the threshold voltages of all edges in a cut to $V_{T,high}$, the timing information of the circuit has to be updated and the edge weights have to be re-evaluated since their feasibility may have been changed. The problem of finding a maximum weighted cut of a graph is NP-Complete. A heuristic employed here is to define a special type of a cut which can be easily identified and where the total number of such cuts is sufficiently small that they can be enumerated. One such class of cuts is based on the topological level of the gates.

Given a combinational circuit, the level of a primary input is zero, and the level of a gate is the one more than maximum of the levels of all its fanin gates.

Definition 4 *Given a circuit graph $G = (V, E)$ corresponding to a levelized combinational circuit, the level k partition of G is a partion of V into $(S, \overline{S})$ such that (1) $\forall\ i \in S$, level(i) $\leq k$; (2) $\forall\ j \in \overline{S}$, level(j) $> k$; and (3) $0 \leq k < \max(level(n)),\ \forall\, v \in V$.*

Clearly, the level k partition is a forward cut. The procedure outlined in Figure 2 finds an initial solution by iteratively finding the maximum level-k cut of the current weighted circuit graph until no other cut with a non-zero weight can be found.

Figure 3 shows an example of how the procedure works. Assume that the circuit graph shown in Figure 3(a) is the current state.

The circuit has three level cuts and steps 6 through 9 of the procedure *initsolution* will result in a $level\ i-1$ cut with the maximum weight of 21. Recall that an edge of weight of zero means that the edge is either non-feasible or its threshold voltage is already $V_{T,high}$. Therefore all the edges in the level $i-1$ cut with positive weight (edges $\{e_{ac}, e_{bd}, e_{be}\}$), will be included in the intial solution and their threshold voltages will be changed to $V_{T,high}$ (steps12 and 13). At this point a static timing analysis is performed and the weights of the edges will be updated (step 14, 15). For example, after changing the weights of the edges in the $level\ i-1$ cut to $V_{T,high}$, the weight of the edge e_{gk} changes from 14 to 0. Thus edge e_{gk} is now unfeasible. The new weighted circuit graph is shown in Figure 3(b). The loop containing steps 3 through 16 will be repeated on the new circuit graph until the weights of all edges become zero.

Lemma 3 *Procedure initsolution (Figure 2) returns a* **maximal** *set of edges whose threshold voltages can be simultaneously changed to $V_{T,high}$ without increasing the delay.*

Corollary 1 *After the termination of the procedure initsolution, no edge in the circuit graph is feasible.*

4.2 Improving the initial Maximal Set by Swapping

The result of procedure *initsolution* (Figure 2) is a partition of the edges of the circuit graph into two disjoint sets, S_h and S_l, where S_h is the set of edges whose threshold voltages will be $V_{T,high}$ and S_l is the set of edges whose threshold voltages will be $V_{T,low}$. Lemma 3 states that the set of edges with $V_{T,high}$'s is maximal in the sense that no more edges can be added to S_h (from S_l) without increasing the delay. Since the objective function is non-negative, this set is a locally optimal solution.

To escape for a locally optimal solution, a *swapping* procedure is carried out. An outline of this procedure is shown in Figure 4. The basic idea here is to move edges from S_h with a total weight that is as small as possible into the set S_L, and to move edges from S_L into S_h whose total weight is as large as possible, without increasing the delay. Note that Corollary 1 states that none of the edges are feasible in the initial solution.

After some edge is moved from the set S_h into S_l, there may be some previously unfeasible nets that become feasible. These are potential candidates for being moved from S_l into S_h. The swapping is performed one edge at a time, with the edge having the smallest weight being moved from S_h to S_l (line 2). This is done to as to minimize the cost of the swap out operation. After setting the threshold voltage of the edge to be swapped out to $V_{T,low}$, an incremental timing analysis (refer to Section 5) is performed to identify all edges that have become feasible (line 5). Note unlike the situation with a cut, it may not be possible to swap in all the feasible nets because they may not be simultaneously feasible. A conservative approach is to define a gain associated with the edges that might be swapped in. This gain is the maximum weight among all the feasible edges (line 6). If the gain is greater than the cost, the swap in performed. Note that to guarantee that the delay is not increased, the feasible nets will have to be swapped in one by one followed by an incremental timing analysis. A good heuristic is to swap in the order of decreasing weight since the edge with a larger weight (gain) will be swapped in first. This is carried out by the procedure $applySwapInNets$, the details of which are omitted. The value returned by the procedure $applySwapInNets$ is the set of edges that will be moved into S_h. Finally, if the gain is less than the cost, the swap is not performed.

The overall dual V_T power minization algorithm is shown in Figure 5.

5 Implementation

The bottleneck in the proposed method is the swapping operation. For each candidate edge to be swapped out, an incremental timing analysis is performed to find the new edges that become feasible. In the worst case, even the incremental timing analysis may have to traverse backward to the primary inputs and forward to the primary outputs in order to compute the changes in the slack values of all the affected gates. However, in practice, the effects of changing the threshold voltage of a single edge on the slack values of other gates in the circuit diminishes geometrically with the depth of the fanin and fanout cones [7]. Consequently, in the current implementation, timing analysis is performed within a window (number of levels around the node in consideration) of a specified size. Experimental results confirm that this simple heuristic significantly improves the running time for large circuits with little degradation on the reduction in the leakage power.

6 Experimental Results

The dual V_T power optimization algorithm shown in Figure 5 was implemented in Smalltalk. All experiments were run on a Sun Sparc4 machine with 64MB memory with the circuits from the MCNC91 benchmark suites. The typical $V_{T,high}$ and $V_{T,low}$ for current dual V_T digital CMOS process are 0.7 and 0.25 volts respectively [12, 13]. All the technology parameters for the power and delay model come from [1, 2, 3, 5]. The signal probabilities are obtained by logic simulations with randomly generated input patterns.

The leakage power model (4,5) is used to compute the leakage power for each gate. Since the signal probabilities and gate sizes will not be changed during the optimization, the leakage power reduction of each edge (by changing it from $V_{T,low}$ to $V_{T,high}$) need only be computed once. The timing constraints for a circuit is the worst case delay of the original circuit implementation, i.e. all transistors are low threshold voltages. Since accurate delay computation and timing analysis is used in the algorithm, the algorithm guarantees to produce a new implementation with a worst case delay that is the same as the original one. The experimental results are shown in Table 2.

The circuits shown in the Table 2 are sorted in increasing order of the number of connections of the each circuit, which roughly indicates the complexity of a circuit. The second (**N**) and third (**G**) columns show the total number of connections and gates of each circuit. The fourth column (**IP**) shows the leakage power in milliwatts before the optimization, i.e. all transistors are $V_{T,low}$. The sixth (**NP1**) through nineth (**C1**) columns show the results for window size (ref. Section 5) set to ∞, i.e. timing analysis is carried out on the entire circuit each time. The fifth column (**%Pd**) shows the leakage power as a percentage of the total power. The sixth (**(NP1)** column shows the leakage power in milli-watts after the optimization. The seventh (**%1**) column shows the leakage power reduction in percentage and the eighth (**X1**) shows the ratio of the reduced leakage power to the initial leakage power. The nineth (**C1**) column shows the CPU time in seconds. The tenth (**NP2**) through thirteenth (**C2**) columns show the experimental results for window size of 5. Finally, the last column shows the worst delay of each circuit when all transistors are chosen to have high threshold voltages.

From the Table 2, it can be seen that significant power reduction can be achieved using the proposed dual V_T power minimization algorithm. The power reduction can be up to an order of magnitude. By setting a window on the static timing analysis, the computation time for large circuits is reduced significantly with little penalty on the power reduction. For example, by setting window size to be 5, the CPU time for the circuit C6288 reduced almost 50% with only 3% decrease in the power reduction. Considering the limited computation power we have, the algorithm finished in reasonable time

for all circuits and should be suitable for real large designs. Finally, if the circuits are implemented with all high threshold voltage devices, the reduction in leakage power will be at least 2 to 3 orders of magnitude. But the last column shows that the average delay increase will be about 30%. The proposed algorithm can achieve significant power reduction without any delay penalties.

7 Conclusion and Future Work

Modern advanced digital CMOS dual V_T technology allows transistors with two different threshold voltage on the same chip. This provides another dimension for circuit optimization. In this paper we addressed the problem of leakage power reduction under delay constraints given a circuit implemented by all low V_T devices for dual V_T technology. A simple and efficient algorithm was presented and experimental results show that significant leakage power reduction can be achieved without any delay penalty.

The increasing use of dual and multiple V_T CMOS technology provides other opportunites for circuit optimization. Currently we are looking at the following problems: (1) delay optimization with minimum leakage power penalty given dual V_T technology; (2) simultaneously gate sizing and V_T selection for power delay tradeoffs; (3) including the effects of dual V_T on the short circuit power which is not considered in the power model in this paper.

8 Acknowledgement

This work was carried out at the Center for Low Power Electronics which is supported by the National Science Foundation, the Department of Commerce of the State of Arizona, and various companies in the microelectronics industry, including, Analog Devices, Analogy, Burr Brown, Hughes Aircraft, Intel, Microchip, Motorola, National Semiconductor, Rockwell, Sicom, SMI, Texas Instruments, and Western Design.

9 References

[1] B. Sheu, d. L. Scharfetter, P.K. Ko and M.C. Jeng "BSIM: Berkeley Short-Channel IGFET Model for MOS Transistors," *IEEE JSSC*, Vol. 22, No. 4, pp. 558-566, August 1987.

[2] T. Sakurai and A. R. Newton "Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas," *IEEE JSSC* Vol. 25, No. 2, pp. 584-594, April 1990.

[3] T. Sakurai and A. R. Newton "Delay Analysis of Series-Connected MOSFET Circuits," *IEEE JSSC*, Vol. 26, No. 2, pp. 122-131, February 1991.

[4] T. Sakurai and A. R. Newton "A Simple MOSFET Model for Circuit Analysis," *IEEE Trans. on Eletron Devices*, Vol. 38, No. 4, pp. 887-893, April 1991.

[5] R. X. Gu and M. I. Elmasry "Power Dissipation Analysis and Optimization of Deep Submicron CMOS Digital Circuits," *IEEE JSSC*, Vol. 31, No. 5, pp. 707-713, May 1996.

[6] J. P. Fishburn and A. E. Dunlop "TILOS: A Posynomial Programming Approach to Transistor Sizing," *Proc. of ICCAD'85*, pp. 326-328, Nov. 1985.

[7] O. Coudert "Gate Sizing: a General Purpose Optimization Approach," *Proc. of ED&TC'96*, Paris, France, March 1996.

[8] M. Borah, R. M. Owens, and M. J. Irwin "Transistor Sizing for Low Power CMOS Circuits" *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems* Vol. 15, No. 6, June 1996, pp. 665-671.

[9] P. Pant, V. De and A. Chatterjee "Device-Circuit Optimization for Minmal Energy and Power Consumption in CMOS Random Logic Networks," *Proc. of DAC'97*, Las Vegas, NV, June 1997.

[10] J. Kao, A. Chandrakasan, and D. Antoniandis "Transistor Sizing Issues and Tool For Multi-Threshold CMOS Technology," *Proc. of DAC'97*, Las Vegas, NV, June 1997.

[11] S. Devadas, A. Ghosh, and K. Keutzer, *Logic Synthesis*, McGraw-Hill, 1994.

[12] H.Y. Xie, Motorola Inc., *personal communications*, 1997.

[13] T. Dillinger, Rockwell Inc., *personal communications*, 1998.

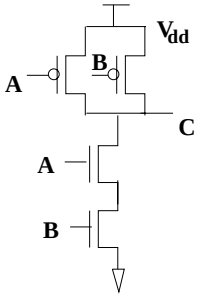

Figure 1: A 2-input CMOS NAND gate.

A	B	Total Leakage Current
0	0	$I^{AB}_{s2,n}$
0	1	$I^{A}_{s1,n}$
1	0	$I^{B}_{s1,n}$
1	1	$I^{A}_{s1,p} + I^{B}_{s1,p}$

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

procedure initSolution (G) {
/* G is the circuit graph with threshold voltages of all edges set to $V_{T,low}$, and each node has been labeled with the delay information and each edge has been labeled by a power reduction value that would result if its threshold voltage is made $V_{T,high}$.
MAXLEVEL is the maximum level of all nodes in G.*/
1. $solution = \emptyset$;
2. $stop$ = FALSE;
3. while($stop$ == FALSE) {
4. max=-1, $maxCut = \emptyset$;
5. initialize the edges weights;
6. for(k=0;$k <$ MAXLEVEL;k++) {
7. $levelCut$ = find level k cut of G_f;
8. if(total weights of $levelCut > max$) {
9. max = total weights of $levelCut$;
10. $maxcut = levelCut$;} }
11. if($maxcut$!= $\emptyset$) {
12. change positive weighted edges in $levelCut$ to $V_{T,high}$
13. $solution = solution \cup maxcut$;
14. update the delay information for each node in G;
15. re-evaluate the edge weights; }
16. else{ stop = TRUE;} }
17. return($solution$); }

Figure 2: Algorithm for finding the initial solution.

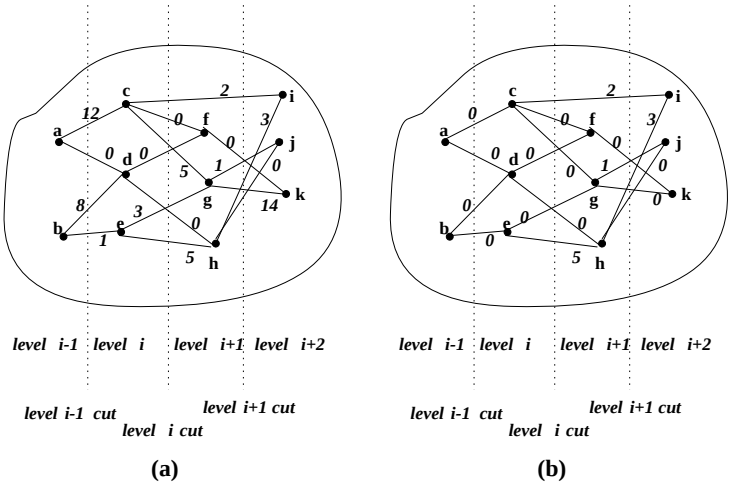

Figure 3: An example explaining the algorithm in Figure 2

procedure swap (G,I) {
/* G is the circuit graph with all edges of I are
$V_{T,high}$'s and the rest of edges are $V_{T,low}$'s. */
1. while (some edge in I not considered for swap) {
2. $swapOutNet$ = the edge in I with the smallest weight;
3. $cost$ = weight of $swapOutNet$;
4. set $swapOutNet$ to $V_{T,low}$;
5. $feasibleNets$ = timingAnalysis($swapOutNet$) ;
6. $gain$ = maximum weight of all edges in $feasibleNets$;
7. if ($gain > cost$) {
8. $swapInNets$ = $applySwapInNets(feasibleNets)$;
9. $I = I \cup swapInNets$; }
10. else {
11. set $swapOutNet$ to $V_{T,high}$
12. restore the original delay information; } }
13. resturn($solution$); }

Figure 4: Algorithm for Swapping

procedure DualVT (G) {
/* G is the circuit graph with all edges are $V_{T,low}$'s */
1. read in the signal probability for each node;
2. compute the weights for each edge of G;
3. static timing analysis;
4. set the timing constraints as the worst cast delay of G;
5. I = *initialSolution*(G);
6. S = *swap* (G,I) ;
7. re-compute the total leakage power;
8. resturn(S); }

Figure 5: The dual V_T power minimization algorithm

					Win = ∞				Win = 5				DlyHigh
cktName	N	G	IP	%Pd	NP1	%1	X1	C1	NP2	%2	X2	C2	
t	15	7	0.54	40.63	0.15	-72.20%	3.6	0.2	0.15	-72.20%	3.6	0.2	1.37
C17	15	7	0.54	40.63	0.15	-72.20%	3.6	0.2	0.15	-72.20%	3.6	0.2	1.39
majority	22	12	0.85	19.39	0.38	-54.80%	2.24	0.3	0.38	-54.80%	2.24	0.3	1.36
b1	23	12	0.97	32.63	0.39	-59.50%	2.49	0.3	0.39	-59.50%	2.49	0.3	1.34
cm152a	47	25	1.81	22.20	0.95	-47.60%	1.91	0.5	0.95	-47.60%	1.91	0.5	1.27
cm82a	51	30	2.2	17.17	0.85	-61.40%	2.59	0.8	0.85	-61.40%	2.59	0.8	1.36
cm151a	52	28	1.87	16.39	1.03	-44.70%	1.82	0.5	1.03	-44.70%	1.82	0.5	1.33
cm42a	68	37	2.21	16.81	0.43	-80.50%	5.14	0.9	0.43	-80.50%	5.14	0.9	1.36
tcon	69	29	2.05	27.24	0.74	-64.00%	2.77	0.7	0.74	-64.00%	2.77	0.7	1.27
decod	75	33	1.39	18.15	0.74	-46.90%	1.88	0.8	0.74	-46.90%	1.88	0.8	1.45
z4ml	85	48	3.47	139.42	1.65	-52.30%	2.1	1.4	1.65	-52.30%	2.1	1.4	1.35
mux	90	47	4.09	16.05	2.73	-33.20%	1.5	0.9	2.73	-33.20%	1.5	0.9	1.36
cm163a	91	50	3.76	15.94	1.49	-60.40%	2.52	1.6	1.49	-60.40%	2.52	1.6	1.34
i1	96	47	3.21	19.95	0.58	-82.00%	5.53	1.2	0.58	-82.00%	5.53	1.2	1.33
cm85a	99	56	4.4	19.94	0.66	-84.90%	6.67	2.1	0.66	-84.90%	6.67	2.1	1.35
pm1	100	54	3.77	21.91	0.71	-81.20%	5.31	2.1	0.71	-81.20%	5.31	2.1	1.34
cm162a	101	61	4.59	22.01	1.45	-68.40%	3.17	2.1	1.45	-68.40%	3.17	2	1.36
x2	103	57	4.49	24.03	1.33	-70.40%	3.38	1.8	1.33	-70.40%	3.38	1.8	1.34
cm150a	106	63	5.12	17.68	2.74	-46.60%	1.87	1.1	2.74	-46.60%	1.87	1.1	1.39
cmb	113	64	4.3	15.17	0.75	-82.60%	5.73	2.6	0.75	-82.60%	5.73	2.3	1.35
parity	121	75	5.09	16.93	3.59	-29.50%	1.42	0.9	3.59	-29.50%	1.42	0.9	1.33
cu	123	66	4.29	17.34	0.4	-90.80%	10.73	3.5	0.4	-90.80%	10.73	3.3	1.36

Table 2: Experimental Results

cktName	N	G	IP	%Pd	Win = ∞				Win = 5				DlyHigh
					NP1	%1	X1	C1	NP2	%2	X2	C2	
cc	128	69	4.68	29.84	2.01	-57.20%	2.33	2	2.01	-57.20%	2.33	2	1.33
cordic	137	81	5.81	18.95	2	-65.60%	2.91	2.1	2	-65.60%	2.91	2.1	1.36
pcle	141	77	5.09	18.70	0.71	-86.00%	7.17	3.5	0.71	-86.00%	7.17	3.5	1.35
sct	162	89	6.68	21.06	2.62	-60.90%	2.55	3.4	2.62	-60.90%	2.55	3.4	1.35
f51m	192	110	8	17.23	2.99	-62.70%	2.68	6.4	2.99	-62.70%	2.68	6.4	1.34
pcler8	198	110	7.9	34.53	1.51	-80.90%	5.23	4.3	1.51	-80.90%	5.23	4.2	1.32
comp	209	123	8.8	17.35	1.61	-81.70%	5.47	5.9	1.71	-80.50%	5.15	5.7	1.36
lal	209	115	8.48	24.08	1.32	-84.50%	6.42	5.7	1.32	-84.50%	6.42	5.5	1.35
unreg	220	121	9.34	20.49	4.8	-48.60%	1.95	2.8	4.8	-48.60%	1.95	2.7	1.36
b9	242	129	9.19	19.53	0.97	-89.50%	9.47	6.8	1.02	-88.90%	9.01	6.6	1.36
i3	264	132	11.51	16.85	11.21	-2.60%	1.03	1.7	11.21	-2.60%	1.03	1.7	1.39
c8	275	152	11.2	19.31	3.26	-70.90%	3.44	8.2	3.26	-70.90%	3.44	8.2	1.35
frg1	281	157	11.93	16.21	1.34	-88.80%	8.9	10.5	1.47	-87.70%	8.12	9.6	1.38
term1	326	184	13.75	19.10	1.87	-86.40%	7.35	14.3	1.87	-86.40%	7.35	13.5	1.35
cht	350	194	15.44	18.77	4.24	-72.50%	3.64	6.8	4.24	-72.50%	3.64	6.9	1.36
my_adder	368	222	16.59	21.42	1.96	-88.20%	8.46	17.3	1.96	-88.20%	8.46	14.7	1.35
9symml	382	213	15.99	17.74	1.83	-88.60%	8.74	19.4	1.83	-88.60%	8.74	17.8	1.35
i5	412	214	14.3	21.29	1.31	-90.90%	10.92	12.2	1.37	-90.40%	10.44	11.5	1.34
ttt2	416	224	16.49	21.25	1.21	-92.70%	13.63	17.4	1.21	-92.70%	13.63	16.8	1.36
i2	437	224	12.76	13.85	1.56	-87.80%	8.18	11.8	1.56	-87.80%	8.18	10.7	1.34
apex7	469	253	18.08	23.87	1.61	-91.10%	11.23	18.5	1.61	-91.10%	11.23	18.2	1.35
C432	470	270	19.61	20.95	5.9	-69.90%	3.32	29.2	5.9	-69.90%	3.32	27.6	1.36
x1	611	331	24.29	23.37	2.22	-90.90%	10.94	21.4	2.22	-90.90%	10.94	20.8	1.35
example2	634	328	22.42	20.54	1.54	-93.10%	14.56	23.9	1.65	-92.60%	13.59	23.6	1.34
too_large	674	378	25.18	20.76	2.57	-89.80%	9.8	49	2.87	-88.60%	8.77	41.4	1.34
alu2	695	387	28.62	30.01	3.68	-87.20%	7.78	85.5	4.02	-86.00%	7.12	58.4	1.35
x4	824	454	33.86	21.33	5.25	-84.50%	6.45	40.8	5.25	-84.50%	6.45	36	1.33
C880	837	475	35.21	20.18	3.66	-89.60%	9.62	152.3	3.67	-89.60%	9.59	99.9	1.37
i6	974	526	42.25	19.32	4.05	-90.40%	10.43	36	4.05	-90.40%	10.43	39.9	1.34
C1908	980	560	38.76	19.67	7.08	-81.70%	5.47	180	7.44	-80.80%	5.21	106.5	1.4
C499	1044	622	42.25	20.20	19.92	-52.80%	2.12	43.7	19.92	-52.80%	2.12	41.1	1.34
C1355	1044	622	42.51	19.72	19.7	-53.70%	2.16	42.2	19.7	-53.70%	2.16	42.3	1.47
vda	1101	611	49.31	53.29	4.38	-91.10%	11.26	73.8	4.49	-90.90%	10.98	72.9	1.35
i9	1280	697	60.94	17.66	14.62	-76.00%	4.17	114.8	14.61	-76.00%	4.17	106	1.47
i7	1360	786	62.42	20.52	8.05	-87.10%	7.75	63.8	8.05	-87.10%	7.75	62.5	1.38
rot	1416	755	54.14	22.68	2.3	-95.80%	23.54	123.9	2.47	-95.40%	21.92	107.8	1.39
alu4	1476	826	59.44	24.92	6.12	-89.70%	9.71	270.3	6.59	-88.90%	9.02	206.2	1.37
t481	1481	825	57.35	52.29	5.96	-89.60%	9.62	120.1	6.06	-89.40%	9.46	110.3	1.34
C2670	1552	828	62.33	17.51	2.59	-95.80%	24.07	242	2.67	-95.70%	23.34	199.7	1.35
apex6	1596	876	68.24	19.65	2.76	-95.90%	24.72	144.9	2.85	-95.80%	23.94	122.3	1.35
x3	1603	886	68.69	19.72	5.32	-92.30%	12.91	122.8	4.58	-93.30%	15	105.9	1.34
frg2	1741	913	67.73	21.71	6.19	-90.90%	10.94	189.2	6.19	-90.90%	10.94	162.9	1.35
k2	2004	1129	88.07	48.20	8.27	-90.60%	10.65	263.9	9.33	-89.40%	9.44	181.9	1.36
i8	2168	1190	94.24	24.44	17.32	-81.60%	5.44	241	17.32	-81.60%	5.44	230.2	1.35
C3540	2404	1339	102.32	21.60	7.47	-92.70%	13.7	565	8.28	-91.90%	12.36	457.4	1.36
dalu	2533	1430	105.51	26.95	6.66	-93.70%	15.84	591.8	7.29	-93.10%	14.47	464.8	1.39
pair	3251	1756	132.88	18.53	7.98	-94.00%	16.65	519.2	8.71	-93.40%	15.26	436.9	1.37
C5315	3281	1781	133.28	17.46	8.99	-93.30%	14.83	794.3	10.53	-92.10%	12.66	700.8	1.34
i10	4623	2522	188.28	24.94	7.99	-95.80%	23.56	2466.5	8.87	-95.30%	21.23	1588.2	1.34
C6288	5799	3406	234.63	18.16	79.92	-65.90%	2.94	15307.5	87.4	-62.80%	2.68	7944.6	1.35
des	7215	3989	313.72	21.94	24.18	-92.30%	12.97	2896.8	24.18	-92.30%	12.97	2445.6	1.36
Average	879	488	36.38	24.07	5.18	-76.10%	7.5	356.5	5.39	-75.80%	7.26	225.2	1.36

Table 2: Experimental results (cont'd).