

# Minimization of Dynamic and Static Power Through Joint Assignment of Threshold Voltages and Sizing Optimization

David Nguyen, Abhijit Davare, Michael Orshansky,  
David Chinnery, Brandon Thompson, and Kurt Keutzer  
University of California at Berkeley

{nguyendt, davare, omisha, chinnery, thompsbj, keutzer}@eecs.berkeley.edu

## ABSTRACT

We describe an optimization strategy for minimizing total power consumption using dual threshold voltage ( $V_{th}$ ) technology. Significant power savings are possible by simultaneous assignment of  $V_{th}$  with gate sizing. We propose an efficient algorithm based on linear programming that jointly performs  $V_{th}$  assignment and gate sizing to minimize total power under delay constraints. First, linear programming assigns the optimal amounts of slack to gates based on power-delay sensitivity. Then, an optimal gate configuration, in terms of  $V_{th}$  and transistor sizes, is selected by an exhaustive local search. Benchmark results for the algorithm show 32% reduction in power consumption on average, compared to sizing only power minimization. There is up to a 57% reduction for some circuits. The flow can be extended to dual supply voltage libraries to yield further power savings.

## Categories and Subject Descriptors

B.6.3 [Design Aids]

## General Terms

Algorithms, Performance, Design

## Keywords

Dual Threshold, Dual Supply Voltage, Sizing, Simultaneous

## 1. INTRODUCTION

In deep submicron technologies, minimization of leakage power becomes the dominant concern as we try to combat the increase in the overall circuit power consumption. At the 90nm technology node, leakage power may make up 42% of total power [4]. The primary reason for this increase in leakage power is the reduction of threshold voltage ( $V_{th}$ ) of devices, and an exponential increase in leakage current that it causes.

A potent way to reduce leakage power consumption is to raise the  $V_{th}$  of some gates. A higher  $V_{th}$  reduces the subthreshold leakage

current of a gate but increases its delay. However, with careful optimization, power consumption can be reduced without penalizing the performance by only increasing the delay of gates that have timing slack. Having an additional  $V_{th}$  on the chip requires an extra mask, and while costly, this practice is justified in the production of high-volume high-end parts. Transistor sizing is also a powerful tool for power minimization. Gates with timing slack can be downsized leading to reduced area, capacitance, and power.

To achieve greatest power savings, the optimization algorithm must be able to simultaneously assign  $V_{th}$  to transistors and size them. This, however, is a difficult optimization problem. The discrete nature of  $V_{th}$  assignment and sizing makes the problem combinatorial. In addition, accurate models of power and circuit delay in terms of optimization variables are non-linear. Thus, in its most general formulation, the optimization problem is an integer (or mixed-integer, if we allow for a continuous range of transistor sizes) non-linear programming problem. Neither of these problems can be solved efficiently.

Previous research has studied  $V_{th}$  assignment separately from transistor sizing [11][12]. A simultaneous  $V_{th}$  and sizing optimization introduced in [7] uses sensitivity-based size updating, which is bound to lead to sub-optimal results. Another algorithm relies on a binary search approach but is very computationally expensive [8]. Previous approaches also did not minimize total power, limiting themselves to stand-by power minimization using dual  $V_{th}$  [11][12]. The reality is that leakage power is now a large component of total power even in the active mode of a chip's operation [6][9]. The work in [8] does acknowledge the need to minimize total power, but it suffers from other drawbacks, such as the lack of global circuit view during  $V_{th}$  assignment.

Our approach explicitly attempts to reduce total power by a simultaneous optimal assignment of  $V_{th}$  and transistor sizing. The basic idea of our approach is to convert the power minimization problem into the problem of optimal slack assignment to gates in the circuit. However, the slack assignment is such that the overall power savings, due to  $V_{th}$  change and/or resizing, are maximized. Formulating the problem in this way casts the combinatorial assignment problem to a linear slack assignment problem, which is then solved efficiently.

The result of our method is sizing and  $V_{th}$  assignment such that delay on a particular gate is increased, but the final circuit meets timing constraints while minimizing power. By assigning slack to a gate, the slack of gates in the transitive fan-in and transitive fan-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ISLPED'03, August 25-27, 2003, Seoul, Korea.

Copyright 2003 ACM 1-58113-682-X/03/0008...\$5.00.

out may be affected. Delay assignments have to be made while ensuring that the slack of other gates is not unduly reduced, so that the entire circuit meets performance constraints. The zero slack algorithm approach [7] shows how to estimate the amount of slack that can be added to each gate so that the circuit remains delay ‘safe’, e.g. still meets its timing constraints. In another approach [2], delay allocation is formulated as a graph-theoretical problem. Independent delay allocations to gates in a circuit are chosen via the notion of a maximally-weighted independent set, with weights representing power reduction per unit delay increase. Our algorithm utilizes the idea of power-delay sensitivities to drive the linear program that allocates the optimal delays to gates depending on their ability to convert slack to power. At the same time, the entire circuit is guaranteed to remain delay safe.

This paper is organized as follows. In Section 2, we describe the implementation of the algorithm and the modeling details. In Section 3, we give the results of running the tests using our optimization algorithm. And we conclude in Section 4.

## 2. IMPLEMENTATION

Our algorithm uses a three-phase strategy to find a circuit configuration with the lower total power consumption. In *Phase I*, we create the maximum amount of slack possible in the circuit, which is achieved by performing sizing optimization for minimum delay at fixed low  $V_{th}$ . In *Phase II*, the available slack in the circuit is distributed to individual gates such that the overall timing performance of the circuit is maintained and the overall power savings are maximized. In *Phase III*, every gate is resized and/or assigned to an alternative  $V_{th}$  with its delay increase not exceeding the assigned slack.

### 2.1 Gate modeling

The gate library that is used to test the performance of our optimization algorithm consists of gates with six discrete sizes with balanced P:N width ratios. While  $V_{th}$  allocation can be done at the transistor level, for simplicity we assume the granularity of  $V_{th}$  assignment is at the gate level. For static timing analysis, the gate delay is the worst case of SPICE characterized rise and fall delays for a given load. Increasing the flexibility of the library, with transistor level sizing and threshold assignment, simply increases the size of the library. Any standard library description in which gate delays are dependent on load capacitance can be used, including separate rise and fall delays. The lookup time for alternative gate configurations increases linearly with the number of alternatives for a gate.

The total power consumption for a gate is

$$(1) \quad P_{total} = P_{static} + P_{dynamic}$$

We assume that short-circuit power is only a small component of the dynamic power consumption, and so ignore it. Then the dynamic power consumption for a gate is

$$(2) \quad P_{dynamic} = 0.5\alpha f V_{DD}^2 (C_{load} + C_{internal})$$

where  $\alpha$  is the gate’s switching activity,  $f$  is the clock frequency,  $C_{load}$  is the load capacitance, and  $C_{internal}$  is the gate’s internal capacitance. The input capacitance of a gate loads its fanins.

Gate leakage is very small in this 0.18 $\mu$ m technology, so only subthreshold leakage is considered to determine static power consumption. Subthreshold leakage was characterized for the dominant leakage states, which have been shown to be responsible for the overwhelming portion (e.g. 95%) of the subthreshold leakage current [10]. Then the static power consumption of a gate is given by

$$(3) \quad P_{static} = (1 - \alpha) \sum_i P_{leak,i} \beta_i$$

where  $P_{leak,i}$  is the leakage current for a gate in dominant leakage state  $i$ , and  $\beta_i$  is the probability that the gate is in that dominant state. The total circuit power consumption is computed by summing over all gates.

Switching activity and dominant leakage state probabilities were calculated by random simulation. Input probabilities were independent with equal probability of being high or low.

### 2.2 Multi-Stage Optimization Flow

The flow begins with optimizing a given circuit for maximum speed using sizing. The available slack is then distributed over the gates based on gate power-delay sensitivities. For each gate a configuration that maximizes the utilization of the assigned delay is then found.

#### 2.2.1 Phase I – Sizing for Minimum Delay

In this phase, we generate a circuit configuration that results in the most available slack. To accomplish this, the threshold voltages on all gates are set to a low  $V_{th}$  and all gates are sized for minimum delay. Sizing can be carried out using any number of available sizing optimizers. To make our algorithm self-contained, we implemented our own version of the sizing optimization algorithm. The algorithm uses a sensitivity-based optimization approach inspired by TILOS [3]: it sets gate widths such that the derivatives of path delays with respect to gate widths are zero.

#### 2.2.2 Phase II – Slack Redistribution using Linear Programming

In this phase, we distribute the available slack in the circuit to individual gates. The simplest strategy of assigning slack is to assign equal slack to every gate along a path. However, this approach introduces two major sources of non-optimality. First, this method does not recognize that the total amount of extra delay assigned to gates in the entire circuit may depend on how total slack is partitioned amongst all gates. Second, some gates are more efficient in converting extra delay to power reduction, and should receive more assigned delay.

We use linear programming to distribute slack to gates with the objective of maximizing total power reduction. The optimization is based on the concept of power-delay sensitivity. A gate’s power-delay sensitivity  $\sigma(i)$  is power reduction per unit of added delay:

$$(4) \quad \sigma(i) = \frac{\Delta P(i)}{\Delta D(i)}$$

$$\max\left(\sum_i \sigma(i)d(i), \forall i \in C\right)$$

- 1)  $at(i) \geq d_{nom}(i) + d(i) + at(z), \forall i \in C, z \in fanin(i)$
- 2)  $at(j) \leq rt(C), \forall j \in outputs(C)$
- 3)  $d(i) \geq 0, \forall i \in C$
- 4)  $d(i) \leq Upper\_bound(i), \forall i \in C$

**Figure 1: Linear programming for delay redistribution**

where  $\Delta P$  is the reduction in power, and  $\Delta D$  is the change in delay. Gates with higher sensitivity can more efficiently utilize a given amount of added delay compared to gates with lower sensitivity. Then, the power reduction for an added amount of delay  $d(i)$  to gate  $i$  is

$$(5) \quad \Delta P(i) = d(i)\sigma(i)$$

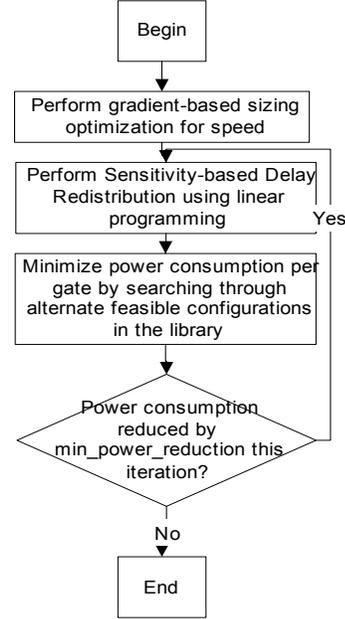
The objective of the linear program is to maximize the sum of power savings over all gates:

$$(6) \quad \max \sum_{i \in Gates} \Delta P(i) = \max \sum_{i \in Gates} d(i) \frac{\Delta P(i)}{\Delta D(i)}$$

In reality, the power-delay trade-off for a gate is not a linear function. Thus, the sensitivities as defined above are accurate only within a small range of delay, where the linear approximation is valid. Hence the LP-based optimization can only be reasonably formulated for small ranges of delay. In order to optimize the circuit across a large range of gate delays, we need to use an iterative approach in which the values of gate's power-delay sensitivities are re-computed at every delay iteration point.

In order to derive the power-delay trade-off curve, every possible gate configuration is first considered. Then, the configurations that give the optimal power-delay trade-off and their sensitivities are used in optimization. In this context, gate configuration is any combination of valid transistor sizes and valid threshold voltages. For each of the valid alternative gate configurations, the decrease in power consumption ( $\Delta P$ ) and the change in delay ( $\Delta D$ ) are calculated. For example, we may compute the sensitivity of changing the gate from all transistors having low  $V_{th}$  to the configuration where all transistors have high  $V_{th}$ . In this manner, we simultaneously make sizing and threshold assignments.

The details of the LP formulation for a given circuit  $C$  are shown in Figure 1, where:  $d(i)$  is the additional delay assigned to gate  $i$ ,  $at(i)$  is the arrival time at the output of gate  $i$ ,  $rt(C)$  is the required time for the given circuit  $C$ ,  $d_{nom}(i)$  is the gate delay in its current configuration,  $fanin(i)$  is all the gates in the fanin of  $i$ ,  $outputs(C)$  are the outputs of the given circuit  $C$ , and  $Upper\_bound(i)$  is the change in delay required to realize the maximum power sensitivity  $\sigma(i)$ . The first line is the objective function and lines (1-4) are the constraints. In addition to the upper bound constraints (4), the constraints are imposed to model the circuit and specify timing constraints. The arrival time constraints ensure that the arrival time at the fan out of each gate is at least the arrival time at the fan in plus the delay of that gate (1). The delay assignments will affect the arrival times of signals within the circuit, but the arrival times at the outputs should remain below the given required time.



**Figure 2: The power minimization flow.**

The required time ensures that the arrival times at all the primary outputs arrive before the required time (2). Constraints (1) and (2) jointly guarantee a “safe” delay assignment. The amount of delay assigned to a gate must not be negative, since the circuit is already in its fastest configuration. Therefore, the additional delay assigned to each gate must be positive (3). This formulation is guaranteed to converge if the input circuit was feasible since in the worst case, all delays can be assigned to 0.

### 2.2.3 Phase III – Threshold Voltage and Sizing Assignment

After the safe delay assignments have been made for all gates, sizing and threshold levels can now be assigned to gates independently. For each gate, the configuration that maximizes power reduction while not increasing delay by more than the assigned delay is chosen. Since the number of alternative configurations for a given gate is typically small, this phase does not limit the performance of our algorithm.

The extra delay that the gate accrues as a result of resizing and/or  $V_{th}$  reassignment may be smaller than the actual delay allowed to the gate by LP. Alternatively, the allowed delay may have been too small to allow any change in gate's configuration. In either case, some extra delay assigned to the gate remains unused. Since the method is iterative, however, the unused assigned delay is reclaimed and reassigned in the next iteration.

As the method continues, the overall power consumption of the circuit monotonically decreases with run-time. The iterations are terminated when the change in power consumption between consecutive iterations is smaller than a predetermined constant.

## 2.3 Extension to Dual Supply

Dual supply technologies can be used to significantly reduce dynamic power without penalizing overall performance. Our slack redistribution approach can also be applied to dual-Vdd libraries to provide substantial power savings.

$$\begin{aligned} & \max(\sum_i \Delta Power_{supply}(i)s(i), \forall i \in C \\ & s(i) \geq s(j), \forall i \in C, \forall j \in fanin(i) \\ & s(i)\Delta delay_{supply}(i) \leq assigned\_delay(i) \end{aligned}$$

**Figure 3: ILP formulation for low-supply assignment subject to topological constraints.**

The assignment of dual-Vdd gates in a CMOS-based circuit is, however, a topologically constrained problem. High supply gates are able to drive both low supply and high supply gates. However, low supply gates are not able to drive high supply gates without the addition of asynchronous level converters. The addition of asynchronous level converters is subject to various robustness and noise margin issues. If asynchronous level converters are not permitted, then assigning gates to low supply becomes very topologically constrained. In particular, all gates which have been assigned to a low supply voltage can only contain low supply gates in their transitive fan out.

In our approach, the initial configuration must be the fastest configuration, in order to have the maximum amount of delay to redistribute. Therefore, all gates are initially assigned high-Vdd. Gates are then chosen to be assigned a low-Vdd value if feasible. The topological and feasibility constraints to assign supply voltages can be encoded using an Integer Linear Programming (ILP) formulation.

The Integer Linear Formulation for the assignment of dual-Vdd gates in a circuit is shown in Figure 3, where:  $s(i)$  is a 0-1 integer variable for gate  $i$ , 1 indicates low supply and 0 indicates high supply,  $\Delta Power_{supply}(i)$  is the reduction in power consumption if the gate  $i$  is switched from high supply to low supply,  $fanin(i)$  is the set of fanin gates for the given gate  $i$ ,  $\Delta delay_{supply}(i)$  is the increase in delay for gate  $i$  if the supply assignment is changed from high to low,  $assigned\_delay(i)$  is the additional delay assigned to this gate during the linear programming stage of the flow.

The objective function in this ILP approach attempts to maximize the power reduction obtained by supply assignments. The first constraint ensures the topological restriction that low supply gates only have low supply gates in their transitive fan out. The second constraint encodes the requirement that the total delay of the gate given a supply change must not exceed the delay allocated to the gate in the previously performed LP stage in the flow.

Two alternatives exist for modifying the flow to include the ILP stage of Vdd assignment. The first alternative is to include the original linear programming stage. However, due to the relative severity of the topological constraints, very few gates will be allocated enough slack to undergo a supply transition. The second alternative that was evaluated was to perform a special LP stage before the ILP supply assignment where delay is assigned based on topological location only. This method separates supply assignment from threshold and sizing assignment, but works better in practice.

**Table 1: Efficient power reduction for the benchmark set using the proposed flow.**

| Circuit | #gates | Power, fastest config | optim. with Sizing (uW) | optim. with Vth / sizing (uW) | CPU time (s) |
|---------|--------|-----------------------|-------------------------|-------------------------------|--------------|
| c17     | 13     | 0.41                  | 0.41                    | 0.41                          | 0.17         |
| c432    | 268    | 1.70                  | 1.00                    | 0.71                          | 3.17         |
| c880    | 641    | 5.62                  | 2.36                    | 1.93                          | 8.35         |
| c499    | 653    | 5.48                  | 4.40                    | 2.17                          | 7.32         |
| c1355   | 717    | 5.36                  | 3.61                    | 2.35                          | 5.98         |
| c1908   | 1263   | 14.72                 | 4.54                    | 3.76                          | 20.52        |
| c2670   | 2181   | 22.80                 | 6.95                    | 4.12                          | 36.28        |
| c3540   | 2604   | 27.87                 | 9.41                    | 4.05                          | 42.39        |
| c6288   | 2736   | 30.51                 | 20.00                   | 13.95                         | 39.53        |
| c5315   | 3876   | 36.93                 | 13.78                   | 8.52                          | 51.18        |
| c7552   | 5437   | 62.83                 | 21.85                   | 14.27                         | 86.00        |

### 3. RESULTS

To test the algorithm, a library with a range of gate widths and thresholds was characterized in a 0.18um process. The logic gates in the library are inverter, NAND2, NAND3, NAND4, NOR2, NOR3 and NOR4. The benchmark gate net lists were mapped to this library.

Gates have six discrete sizes, ranging from 0.18um to 1.80um for an inverter. Vth allocation can be done on a transistor level, but for simplicity we assume the granularity of Vth assignment is at the gate level. For NMOS (PMOS) transistors, the high threshold voltage is 0.45V (-0.45V) and the low threshold voltage is 0.30V (-0.30V). The possible gate supply voltages are 1.8V and 1.2V.

Our proposed flow was applied to the 11 circuits in the ISCAS85 benchmark set. First, the circuits were sized for maximum speed using a gradient-based optimizer, similar to TILOS [3]. Based on these parameters, the initial power consumption of each circuit was recorded. Power optimization was then performed with the chosen library and cycle time relaxation percentage. The iteration cutoff percentage parameter was set to 2% unless otherwise noted. Final power consumption was recorded, and the CPU time was recorded as well. The runtime numbers were obtained on an Intel Pentium 3 Mobile processor running at 1GHz with 256 MB of RAM.

Table 1 presents the absolute power consumption for each of the benchmark circuits before optimization, after application of the flow with only sizing available, and after the application of the flow with both sizing and dual-Vth options available. The CPU runtimes taken for the application of the flow with the dual-Vth/sizing library are shown.

Figure 4 and 5 present the static and dynamic make up of overall power consumption for circuits sized for maximum speed (Figure 4), and then sizing and dual-Vth optimized for minimum power consumption (Figure 5). With our 0.18um library, static power accounts for about 59% of overall power while dynamic power accounts for about 41% in the speed optimized configuration. Once optimized for power, static power accounts for about 16% of overall power while dynamic power accounts for about 84%. There is a significant reduction in static power by using dual-Vth. On average, 88% of the power reduction was from static power consumption and 12% was from dynamic power reduction.

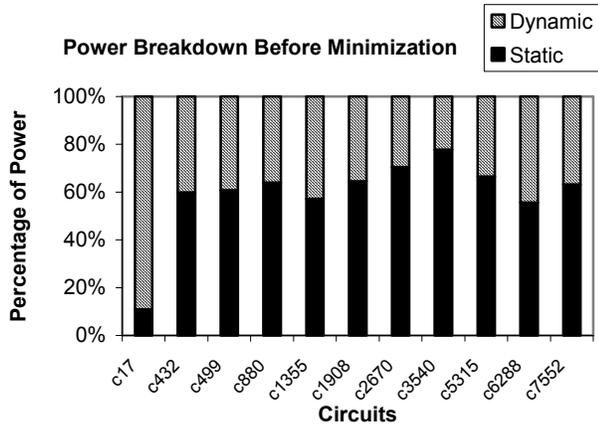


Figure 4: Percentage of static and dynamic power in overall power for speed optimized circuits

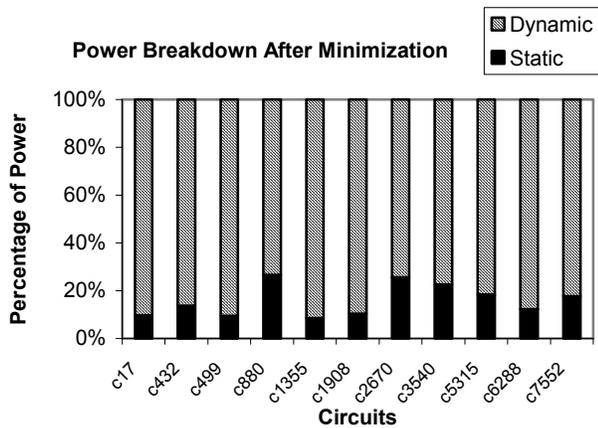


Figure 5: Percentage of static and dynamic power in overall power for power optimized circuits while maintaining cycle time of speed-optimized circuit.

In Figure 6, the power reduction from the fastest circuit configuration for both libraries is shown. The dual-Vth/sizing library yields better results for all of the circuits. The power reduction from the fastest configuration with the dual-Vth/sizing library ranges from 1.3% to 85.5% with an average reduction of 63.0%. The power reduction with the sizing library from the fastest configuration ranges from 0% to 69.5% with an average of 47.2%. Adding dual threshold voltages gives an additional 15.8% reduction in total power (32% lower power than sizing only).

As the required time of a circuit is increased, more slack is introduced into the circuit. This slack can be transformed into greater assigned delay for the gates in the circuit and can lead to lower power consumption.

Figure 7 shows the change in power consumption with the application of our algorithm as the required time is relaxed from 0% to 200%. This plot was obtained with the dual-Vth/sizing library on circuit c3540. The power consumption drops 32.1% with a 20% relaxation in the required time. At 100% cycle time relaxation, the power consumption drops by 56.3%, while further relaxation to 200% yields a 65.4% power drop.

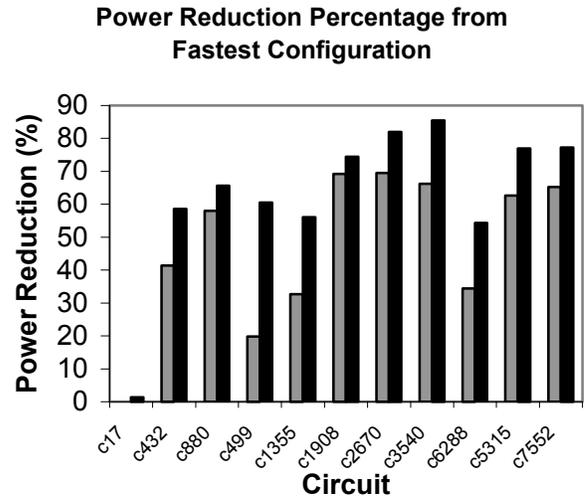


Figure 6: Percentage power reduction from speed optimized configuration.

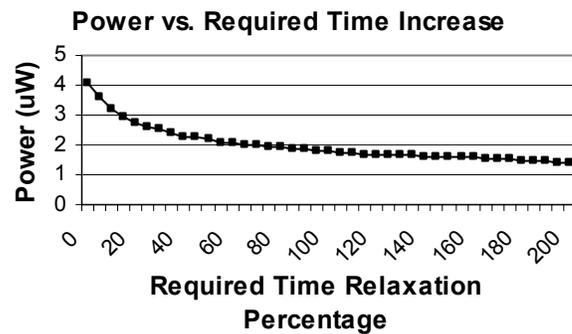


Figure 7: Power reduction tapers off as required time target is relaxed.

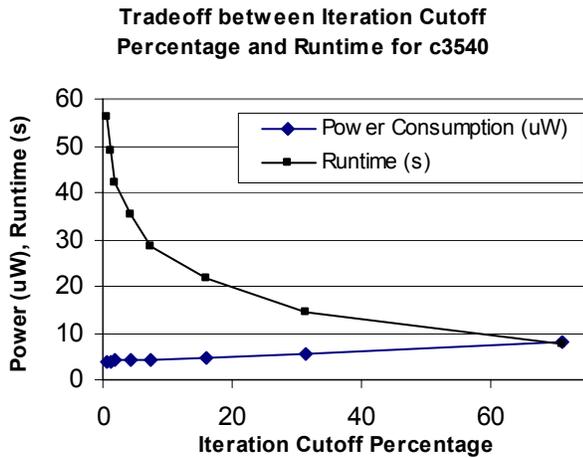
The iteration cutoff parameter is the loop termination criteria for the LP assignment and exhaustive library assignment steps. A higher number indicates that the iteration process is allowed to proceed for a fewer number of loop cycles. This will result in higher overall power consumption but will consume less CPU time. On the other hand, if the iteration cutoff parameter is set to a lower value, more iteration steps will occur, power consumption will drop further, at the expense of higher runtime.

Figure 8 shows this tradeoff between power consumption and runtime as dependent on the selected iteration cutoff percentage. This data was obtained from circuit c3540 with the dual-Vth/sizing library. The flexibility in our algorithm for terminating the iteration procedure allows efficient exploitation of the tradeoff between runtime and power reduction.

The concept of weighted slack distribution can be extended to topologically constrained dual-Vdd libraries by using an ILP formulation. To test this extension to our algorithm, a third library consisting of gates with dual-Vdd/dual-Vth/sizing was created. All 11 ISCAS85 benchmark circuits were re-evaluated using this new library and the extended version of the proposed flow.

**Table 2: Power reduction obtained with extension of the flow to handle dual-VDD/dual-Vth/sizing.**

| Circuit | #Gates | Power, fastest config (uW) | Power with Vdd/Vth/sizing (uW) | CPU runtime (s) |
|---------|--------|----------------------------|--------------------------------|-----------------|
| c17     | 13     | 0.41                       | 0.41                           | 0.19            |
| c432    | 268    | 1.70                       | 0.68                           | 2.82            |
| c880    | 641    | 5.62                       | 1.89                           | 8.39            |
| c499    | 653    | 5.48                       | 2.17                           | 8.46            |
| c1355   | 717    | 5.36                       | 2.12                           | 7.04            |
| c1908   | 1263   | 14.72                      | 3.52                           | 25.26           |
| c2670   | 2181   | 22.80                      | 4.15                           | 39.40           |
| c3540   | 2604   | 27.87                      | 3.88                           | 52.71           |
| c6288   | 2736   | 30.51                      | 13.36                          | 38.05           |
| c5315   | 3876   | 36.93                      | 8.56                           | 64.14           |
| c7552   | 5437   | 62.83                      | 13.98                          | 107.29          |



**Figure 8: Evaluating the tradeoff between power consumption and, CPU time, and iteration cutoff percentage.**

Table 2 shows the results obtained with the extended flow and the dual-Vdd/dual-Vth/sizing library. The column corresponding to the power consumption for the fastest configuration is shown again for reference. Due to the addition of an ILP step to each iteration, the average CPU time increases over the dual-Vth/sizing flow. However, power consumption is reduced further, on average, than with libraries without dual Vdd. In this set of benchmarks, there was an average of 74.5% power reduction from the fastest configuration. This is 11.5% greater than using sizing and dual Vth.

#### 4. CONCLUSION

In this paper, we have presented a computationally efficient heuristic approach to minimize power consumption of combinational circuits with the simultaneous application of threshold and sizing. Our approach is based on the redistribution of slack using the concept of power sensitivity. Slack is assigned to gates that obtain the maximum power savings from a given delay increase. Gates then use assigned slack to reduce power consumption maximally. The flow iterates through these steps while significant power reduction is still obtained in each iteration. We have applied this approach to the complete ISCAS85 benchmark set using a dual-Vth library to show extensive power reduction while maintaining circuit speed. As the required time requirements are relaxed, further power savings can

be obtained since additional slack is introduced into the circuit. Since computation is bounded by the slack distribution time, and this is performed with a linear programming approach that runs quickly on these circuits, the program as a whole runs very efficiently; even large benchmarks finished within a minute. Since the appropriate use of this algorithm is to latch/FF bounded combinational blocks, we believe that the approach is computationally efficient enough to optimize industrial circuits. We believe that this flow can also be extended to handle dual voltage supply libraries, and provide further power reductions.

#### 5. REFERENCES

- [1] Brodersen, R., et al, "Methods for True Power Minimization," *International Conference on Computer-Aided Design*, 2002, pp. 35-40.
- [2] Chen, C., and Sarrafzadeh, M., "An Effective Algorithm for Gate-Level Power-Delay Tradeoff Using Two Voltages." *International Conference on Computer Design*, 1999, pp. 222 - 227.
- [3] Fishburn, J.P., and Dunlop, A.E., "TILOS: A Posynomial Programming Approach to Transistor Sizing," *International Conference on Computer-Aided Design*, 1985, pp. 326-328.
- [4] Kao J., Narendra, S., Chandrakasan A., "Subthreshold Leakage Modeling and Reduction Techniques," *International Conference on Computer-Aided Design*, 2002, pp. 141-149.
- [5] Karnik, T., "Total Power Optimization By Simultaneous Dual-Vt Allocation and Device Sizing in High Performance Microprocessors," *Design Automation Conference*, 2002, pp. 486-491.
- [6] Kim, C. and Roy, K., "Dynamic VTH Scaling Scheme for Active Leakage Power Reduction", *Design and Test European Conference*, 2002, pp. 163-167.
- [7] Nair, R., Berman, C.L., Hauge, P.S., Yoffa, E.J., "Generation of Performance Constraints for Layout," *IEEE Trans. on Computer-Aided Design*, Vol. 8 No. 8, August 1989, pp. 860-874.
- [8] Pant, P., Roy, R., and Chatterjee, A., "Dual-Threshold Voltage Assignment with Transistor Sizing for Low Power CMOS Circuits," *IEEE Trans. on VLSI*, Vol. 9, No. 2, 4, 2001, pp. 390-394.
- [9] Shrivastava A., and Sylvester D., "Minimizing Total Power by Simultaneous Vdd/Vth Assignment", *to appear at Asia and Pacific Design Automation Conference*, 2003.
- [10] Sirichotiyakul, S., et al, "Stand-by power minimization through simultaneous threshold voltage selection and circuit sizing," *Design Automation Conference*, 1999, pp. 436-441.
- [11] Wang, Q., and Vrudhula, S., "Static power optimization of deep submicron CMOS circuit for dual Vt technology," *International Conference on Computer-Aided Design*, 1998, pp. 490-496.
- [12] Wei, L., et al, "Mixed-Vth (MVT) CMOS circuit design methodology for low power applications," *Design Automation Conference*, 1999, pp. 430-435.