Cell Selection from Technology Libraries for Minimizing Power

Yumin Zhang
Synopsys, Inc.
700 East Middlefield Road
Mountain View, CA 94043
yumin@synopsys.com

Xiaobo (Sharon) Hu Danny Z. Chen
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556, USA
{shu, chen}@cse.nd.edu

Abstract

In this paper we present a new library-oriented cell selection approach to minimize power consumption of combinational circuits. Our unified Mixed Integer-Linear-Programming (MILP) formulation selects library cells with different gate sizes, supply voltages and threshold voltages simultaneously during technology mapping. Experimental results on benchmarks mapped to an industrial library show that our technique achieves 19% more power saving in less CPU time comparing with other approaches.

1 Introduction

One effective method to reduce power in standard-cell style design is to judiciously select library cells during technology mapping at the gate level. Standard cell libraries may contain various cells that implement the same Boolean function but have different sizes, supply voltages and threshold voltages [12, 13, 14]. In this paper, we aim to solve the following problem: Given a technology mapped combinational circuit under timing constraints, select those cells in a given library such that the power consumption is minimized without violating the delay constraints. This problem is an NP-complete problem given the fact that only discrete gate sizes and voltage values are allowed in today's technology libraries.

Approaches for optimizing performance in general cannot be used directly to optimize power since delay and power behave rather differently as gate sizes and voltages change. Towards power minimization, several approaches have been proposed in literature to select cells with one or two kinds of variations. However, several drawbacks prevent them from achieving high power reduction. For example, approaches in [1, 4, 12, 15] lack a global view of circuits due to the backward traversal method used. [3, 10] use linear or piece-wise linear functions to model the relation between the delays and sizes of gates, which can introduce considerable errors. [2, 7] use a locally computed weight function for cell selection, which may not accurately capture power saving globally. Above all, it is not clear how the above methods can be extended to select cells efficiently from libraries with all three kinds of variations since gate size, supply voltage and threshold voltage affect the power and delay differently. To our best knowledge, no papers have attempted to solve this library-oriented cell selection power optimization problem.

In this paper, we describe a unified Mixed Integer-Linear-Programming (MILP) approach to solve the gate-level low power cell selection problem. Our main contributions are summarized below:

Improved power and delay model: we consider wire delays and capacitances, and take into account different pin-to-pin delays and input capacitances.

A unique MILP formulation: we not only consider three types of variations simultaneously but also integrate level shifter insertion into the process. We also achieve an significant reduction of the numbers of constraints and variables in the MILP.

Efficient and effective heuristics: LP relaxation and novel approximation heuristics are used to obtain reliable solutions in short CPU time.

Practical experimental results: experiments are conducted on ISCAS'85 benchmarks with a real-world library. The results show that our approach achieves 19% more power saving in less CPU time.

2 Preliminaries

As given in the IBM library databook, the dynamic power consumption of a cell v can be computed by

$$P_d(v) = f \cdot V_{dd}^2 \cdot \sum_j \alpha(j)(C_p(j) + C_w(j) + C_t(v)) \quad (1)$$

where f is the clock frequency, V_{dd} is the supply voltage, $\alpha(j)$, $C_p(j)$ and $C_w(j)$ are the switching activity, input and wire capacitances of v's one fanin node, j, and $C_t(v)$ is v's internal capacitance. The static power consumption is mainly due to the sub-threshold current in a well-designed CMOS circuit [8]. The sub-threshold current is computed as in [9]

$$I_{sub} = I_0 \cdot (1 - e^{-V_{ds}/V_T}) \cdot e^{(V_{gs} - V_{th})/nV_T} \quad (2)$$

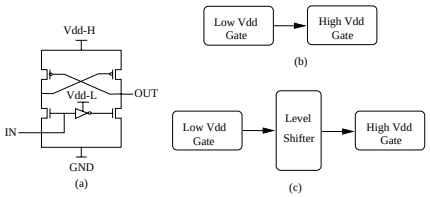

Figure 1: (a) A conventional level shifter. (b) Direct connection of a V_{ddl} gate to a V_{ddh} gate. (c) A level shifter in-between to connect the two gates.

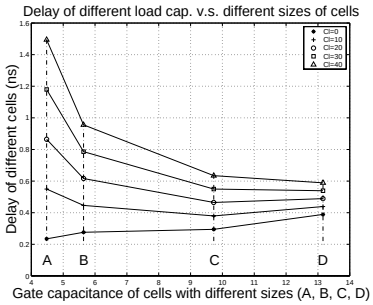

Figure 2: The delays at certain loads for cells with different sizes, A, B, C, and D.

where I_0 is a function of process technology and gate ratio, V_T is the thermal voltage and n is the sub-threshold swing coefficient. If the supply voltage of a gate is reduced to save power, a high DC current will flow through its fanouts with a higher supply voltage [12]. Level shifters are often used to avoid such high current, as shown in Figure 1.

In Figure 2, we depict the delays of four cells with different sizes for the AND logic in the IBM library. Each of the five curves depicts the delays of four different cells, A, B, C, and D at a certain load, from 0 to 40 standard-load (0.036pf). Cells A, B, C, and D have increased sizes. It is clear that the assumption used in [10], where the delay is a linear function of its size, can introduce quite some errors. Also, another widely used assumption, that smaller gates always have longer delays, is not necessarily true for real-world libraries. Section 4 will describe how our approach handles the smaller but faster cells. We will discuss how to handle pin-to-pin delay differences in Section 3.

For gate v, the available time is the longest time from the primary inputs to the output of v, and the required time is the earliest time at which the output of v must be ready in order to meet the delay constraint. The slack of v is defined as the difference of the required time and the available time of v. Intuitively, the slack time is the maximum amount of time that the gate can be slowed down without violating the timing constraint. The delay of a path is the sum of delays of every gates on that path.

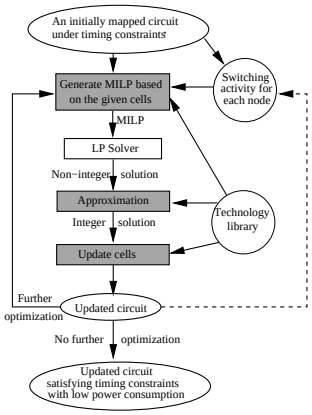

Figure 3: The flow of our cell selection approach.

3 Unified Problem Formulation

Our cell selection technique can be integrated into a logic synthesis process to optimize power without violating timing constraints posed on the circuits. The flow of our approach is depicted in Figure 3. The dash line in Figure 3 indicates that the update of switching activity of each node after each iteration is not implemented in our experiments. This practice is acceptable since our cell selection approach strives to balance delays of different paths [10]. In the following, we describe our modeling of a set of cells for a gate, a general MILP formulation of the cell selection problem with level shifter insertion integrated in, and how we handle pin-to-pin differences and the effect of interconnects on delay and power.

3.1 Modeling library cells

The delay and power for each cell can be estimated as described in Section 2. However, it is difficult to use close-form expressions for power optimizations to accurately capture the delay and power of those cells with different sizes and voltages for the same logic function. Here, we describe a way to represent such different cells in a library. Assume that gate v has $N(v)$ cells with different sizes, supply voltages and/or threshold voltages. We associate a variable $x_i(v)$ with the ith cell of gate v. If $x_i(v) = 1$, the ith cell of gate v is used in the design. For each gate, only one cell can be selected at a time. Since the number of different gate sizes is usually between 3 and 6, and most designs only employ dual supply voltages and dual threshold voltages, the number of x_i's is generally small.

For the ith cell, the delay at a given load, $T_i(v)$, can be looked up from the library databook and the power, $p_i(v)$, can be estimated as the sum of (1) and (2) if the clock frequency, supply and threshold voltages, and switching activities are given.

3.2 MILP formulation

If the timing constraint on a circuit is T_c, the delay of every path must always be no more than T_c regardless of the cells selected. To capture this requirement by enumerating every path, as did in [7], is not practical because the number of paths in a circuit can be huge. To overcome this difficulty, we rewrite the delay constraints similar to those used in [3]. That is, we transform constraints on paths to gates and interconnects. Denote a circuit with V gates and E interconnects as $G(V, E)$. We first introduce two super gates, IN and OUT, which are connected to all primary input pins and output pins, respectively. Associate variables $D(v)$ and $d(v)$ with each gate $v \in V$. Intuitively, $D(v)$ represents the available time at the output of v, while $d(v)$ corresponds to the delay increase of v after optimization. Then the delay constraints can be rewritten as:

$$D(OUT) - D(IN) \leq T_c \tag{3}$$

$$D(v) - D(u) - d(v) \geq T(v) \quad \forall e(u, v) \in E \tag{4}$$

where $T(v)$ is the delay of v before optimization. We have formally proved that the above constraints indeed correctly capture the timing constraints. Due to the space limit, the proof is omitted.

To resolve the dependency of delay and power of a gate on the sizes of its fanouts, which could be changed during the optimization, we adopt an iterative approach. During each iteration, we assume the capacitance at the fanouts of v to be a constant. In particular, we use the capacitance values prior to the optimization. Then, for each $x_i(v)$, $T_i(v)$ becomes a constant at this given load. As pointed out earlier, the power consumption $p_i(v)$ for the ith cell can also be treated as a constant. We formulate the cell selection power optimization problem as follows.

$$\text{Maximize:} \quad \sum_{v \in V} \sum_{i=1}^{N(v)} x_i(v)(p(v) - p_i(v)) \tag{5}$$

$$\text{Subject to:} \quad D(OUT) - D(IN) \leq T_c \tag{6}$$

$$D(v) - D(u) - \sum_{i=1}^{N(v)} x_i(v)(T_i(v) - T(v)) \geq T(v) \quad \forall e(u, v) \in E \tag{7}$$

$$\sum_{i=1}^{N(v)} x_i(v) \leq 1 \qquad x_i(v) = \{0, 1\} \ \forall v \in V \tag{8}$$

where $x_i(v)$'s and $D(v)$'s are variables and need to be determined, $T(v)$'s and $p(v)$'s are delay and power of cells used before the optimization, and $T_i(v)$'s and $p_i(v)$'s are delay and power of those cells represented by $x_i(v)$'s which will be checked. If $x_i(v)$ is 1, the ith cell is selected to be in the circuit.

We have pointed out in Section 2 that level shifters must be inserted if a V_{ddl} gate drives a V_{ddh} gate. Since level shifters introduce delay and power overhead into a circuit and have a direct impact on the cell selection decision, the insertion should be considered together with voltage scaling, rather than separately as in [2]. We associate a new variable $x_{ls}(v)$ with v to represent the insertion of a level shifter after v. The value of $x_{ls}(v)$ is 1 if and only if the supply voltage of v is scaled to V_{ddl} and the supply voltage of one of its fanouts is V_{ddh}. Denote the delay and power consumption of a level shifter as two constants provided from the technology library, T_{ls} and p_{ls}, respectively. The MILP formulation for the cell selection problem becomes

$$\text{Maximize:} \quad \sum_{v \in V} \sum_{i=1}^{N(v)} x_i(v)(p(v) - p_i(v)) - x_{ls}(v)p_{ls} \tag{9}$$

$$\text{Subject to:} \quad D(OUT) - D(IN) \leq T_c \tag{10}$$

$$D(v) - D(u) - \sum_{i=1}^{N(v)} x_i(v)(T_i(v) - T(v)) - x_{ls}(v)T_{ls} \geq T(v) \ \ \forall e(u, v) \in E \tag{11}$$

$$x_{ls}(v) \geq \sum x_j(v) - \sum x_k(w) \ \ \forall e(v, w) \in E \tag{12}$$

$$\sum_{i=1}^{N(v)} x_i(v) \leq 1, \ x_i(v), x_{ls}(v) = \{0, 1\} \ \forall v \in V \tag{13}$$

where $x_j(v)$'s and $x_k(w)$'s are the x variables associated with V_{ddl} cells for gate v and its fanouts w's.

After one iteration of solving the above MILP problem, the gate sizes, voltages and delays are updated according to the newly selected cells and a new MILP is formed and solved subsequently. The process continues until no more changes can be made to the circuit for power reduction.

3.3 Pin variance and interconnects

In real-world libraries, different input pins may have different delays. We extend the formulation in Section 3.2 to handle this variance. For gate v, if one of its input pins is connected to the output of gate u, we use $T(u, v)$ to represent the delay from this input pin to the output of v. Thus, $d(u, v)$ represents the corresponding delay increase after optimization. Substituting $T(u, v)$, $d(u, v)$, and $T_i(u, v)$ to the above MILP in (5)-(8) and (9)-(13), the new MILP correctly handles the pin-to-pin differences.

Interconnects introduce capacitances and affect the delay and power of a circuit. The effect of interconnects can be modeled as suggested by the IBM library databook by associating each interconnect

with a capacitance. The load capacitance of a gate, which affects the delay of the gate, can be computed as the sum of each fanout's input capacitance and interconnect capacitance. The interconnect effect on power is considered by the power formula in (1).

4 Solving the MILP Efficiently

Given the MILP formulation in (9)-(13), one may immediately point out that the number of variables and the number of constraints can be rather large for large circuits. In this section, we present several observations that are used to reduce the size of the MILP formulation. Then, an efficient heuristic is proposed to solve the MILP in short CPU time.

4.1 Reducing the size of the MILP

Given a circuit $G(V, E)$, if each gate has $|N(v)|$ different cells, it seems that the number of variables in the MILP is larger than $|V| \cdot |N(v)|$ and the number of constraints is larger than $|E| + |V|$. We reduce both these numbers as follows. The shifter variable $x_{ls}(v)$ can be eliminated if a gate has enough slack to use a cell with V_{ddl} but one of its fanouts does not. That is, $x_{ls}(v)$ can be replaced by $x_i(v)$'s corresponding to V_{ddl} cells. If the slack on a gate is not big enough for any slower cell from the library to be selected, then $x_i(v)$ can be omitted. (Only those cells with delay within the slack allowable range will have a chance to be selected.)

Maintaining a sorted cell list in the increasing order of delays for each gate can facilitate the reduction of x_i's. However, this sorted cell sequence changes for different load capacitances as different cells may experience different delay changes. Instead of enumerating all sequences for $N(v)$ cells, which is a factorial function of $N(v)$, the sorted sequences can be obtained in a systematic way. We first identify the at most $N(v)(N(v)-1)/2$ intersections between all $N(v)$ delay lines at different load capacitances. At each intersection, the sorted sequence of the $N(v)$ values changes, but the change only involves swapping the two intersecting lines. In this way, we can readily construct the sorted sequences of the delays for different cells at different load capacitances, and hence the MILP instance can be built efficiently.

4.2 A heuristic for solving the MILP

Since the cell selection problem on libraries with limited cells has been proved to be NP-complete, our MILP formulation is not likely to be solvable in polynomial time. To solve the MILP problem in (9)-(13) efficiently, we adopt an LP relaxation approach. That is, we solve the MILP problem by omitting the integer constraints on $x_i(v)$ and $x_{ls}(v)$ variables, and approximate the non-integer results with appropriate integers. Though this approach is often used in solving MILP problems, approximating non-integer solutions with integer ones is problem dependent. In our case, simply rounding the non-integer values to the closest integers can lead to unsafe circuit implementations (i.e., violating the timing constraints).

We apply the following approximation to the LP solution. For each gate v, let $\Delta T(u, v)$ be the delay increase. The value of $\Delta T(u, v)$ is computed by

$$\Delta T(u,v) = \sum_{i=1}^{N(v)} x_i(v)(T_i(u,v) - T(u,v)) + x_{ls}(v)T_{ls}$$

where the values of $x_i(v)$ and $x_{ls}(v)$ are obtained from the LP solution. If no cell in the library has the same delay as $T(u, v) + \Delta T(u, v)$, the cell which has the minimum power consumption among the cells with delays less than $T(u, v) + \Delta T(u, v)$ is selected to be the implementation for gate v. Supply voltage scaling and shifter insertion are decided in a backward topological-sort manner from primary outputs to avoid violating timing constraints or high DC current through high V_{dd} gates. Only when $\Delta T(u, v) \geq (T_i(u, v) - T(u, v)) + T_{sl}$ and the power consumption overhead caused by the insertion of a level shifter is less than the power saving of selecting the ith cell, the ith cell with low supply voltage can be selected to be the implementation for v. Since the delay of the new cell is always chosen to be smaller than the one obtained from the LP solution, the circuit will never violate the timing constraints after the approximation. The above approximation approach is very effective and efficient, as shown by experimental results in Section 5.

As we pointed out in Section 3, an iterative process is needed in order to account for the dependency of gate delay and power consumption on the changes of its fanout gates. That is, the slack, delay, capacitance, supply voltage, threshold voltage, and load capacitance of each gate are recomputed after each iteration of solving the LP. Then, a new MILP is formulated and solved by the above approximation.

In Section 2, we have pointed out that in some load capacitance range, smaller yet faster cells do exist in real-world libraries. This indicates that sometimes replacing a larger cell with a smaller one decreases the power *without* sacrificing the performance. Such a phenomenon cannot be properly captured by the model used in [2, 10] for gate resizing. To handle this case properly, we only need a small modification to our algorithm. At the beginning of every iteration, we examine each gate. If there exists a set of cells that have both smaller delay and smaller power consumption than the one used in the circuit, we

replace the cell in the circuit by the one with the smallest delay in the set. Subsequently, the load capacitance and delay for each gate are recalculated according to the cells used in the updated circuit. The replacement is repeated until no such smaller but faster cells can be found to replace cells in the circuit. Then, the MILP problem is formulated and solved as discussed above.

5 Experimental Results

We have applied our algorithms to the combinational circuits in the ISCAS'85 benchmark suite. The library used is the ASIC 5L tech library [5] (a $0.5\mu m$ CMOS library). Seven types of gates, INVERTER, BUFFER, NAND, AND, OR, NOR and XOR, were used in our experiments. Each gate type has four different performance levels corresponding to four different sizes of cells of the same logic. Each cell possesses different internal and input pins capacitances. Two commonly used values of supply voltage, 3.3V and 2.5V, are selected to be V_{ddh} and V_{ddl}, respectively, based on the library databook. The gate delay for a given load and power consumption can be readily estimated from the databook of the library. The platform is a Sun Ultra 5/10, with 440MHz clock and 256M memory, running SunOS 5.6. To compare different approaches based on the same libraries, we implemented the two most recent approaches in [2, 10] for selecting cells with different sizes (gate resizing), and the approach in [2] for selecting cells with different sizes and supply voltages (supply voltage scaling). The three approaches, approaches from [2, 10] and ours, are all implemented to the best as we can, and they share as many procedures as possible.

The ISCAS'85 benchmark circuits were initially technology mapped to the biggest size, performance-level-D 3.3V gates in the library with the Synopsys Design Compiler [11]. The switching activities for each net are collected after the mapping. Then, the optimization programs were applied. An LP solver, *oslbslv*, from IBM [6] was used to solve the linear programming problems. The power saving is measured as a percentage of the power consumption before any optimization (i.e., using only performance-level-D 3.3V gates). The CPU time includes the running time of all programs from reading input circuit netlist to reporting the results.

In Table 1, we summarize the number of iterations (#), percentage of power saving (%), and CPU time (s) obtained by the approach in [10] (COMP), in [2] (MWIS), and our approach (MILP) for gate resizing only. The data shows that our approach can save 79% power on average and it finishes within 7 minutes for large circuits (more than 4000 gates). Comparing with the approach in [10], our approach achieves 9% more power saving and takes 30% less CPU time. Comparing with the gate resizing approach in [2], our approach achieves 26% more power savings in 79% less CPU time.

A key issue is how well our approximation from the solutions of LP to solutions of MILP is. To find out this, we compare the solutions for the MILP obtained by our approximation with that from the LP on the 10 benchmark circuits. The data shows that our approximation results are within 97% of the LP solutions. Note that the solutions for an MILP cannot be better than that for the corresponding LP.

The power saving and CPU time for supply voltage scaling only by approach in [2] (MWIS-VS) and our approach (MILP-VS) are summarized in Table 2. The approach in [2] separates the level shifter insertion from the voltage scaling, while our approach integrates the voltage scaling and level shifter insertion together. As stated in [2], adding level shifters will decrease the power saving by 5% and increase the CPU time of their approach. Experimental data shows that our approach outperforms the approach in [2] for supply voltage scaling by more than 14% and the CPU time is 57% less on average.

The result of the simultaneous selection of cells with different sizes and supply voltages by our approach is also summarized in Table 2 (MILP-SIMU). The best possible result for the simultaneous gate resizing and supply voltage scaling in [2] would be the sum of the saving by gate resizing and supply voltage scaling minus the level shifter overhead (5%). Comparing with this best possible result by the approach in [2], our simultaneous cell selection is on average 19% better in power reduction.

The number of inserted level shifters due to supply voltage scaling by our approach is also summarized in Table 2 (Sh. #). According to the area information from the IBM library databook, the area overhead by inserting level shifters is about 2% on average. Since gate resizing decreases the area of a circuit, the overall area, considering some addition due to level shifter insertion, will decrease after cell selection for different sizes and supply voltages.

6 Conclusion

In the paper, we present a unified approach to simultaneously select library cells with three variations, gate sizes, supply voltages and threshold voltages, to optimize power at the gate level. The MILP formulation, being intuitive and easy to use, is also capable of combining cell selections with lever shifter insertions. Special efforts are given to solve the MILP effectively and efficiently. Experimental results on the

Table 1: Comparison of power savings and CPU time of three approaches on gate resizing

Bench	# of	COMP			MWIS			MILP		
marks	gates	(#)	(%)	(s)	(#)	(%)	(s)	(#)	p(%)	(s)
c432	216	5	69.13	4.73	38	53.22	2.82	5	81.72	6.02
c499	246	4	47.11	4.35	23	49.55	3.28	4	66.06	4.93
c880	435	8	73.72	13.94	48	66.08	9.70	7	78.93	12.77
c1355	590	14	68.94	31.29	21	25.12	10.44	8	84.16	42.73
c1908	1057	15	76.24	64.69	60	54.75	211.48	8	83.31	40.36
c2670	1400	7	76.00	50.07	90	72.60	136.94	4	78.66	36.00
c3540	1983	7	73.89	68.05	118	63.84	949.25	4	78.72	56.39
c5315	2973	15	73.96	290.62	85	66.90	708.08	5	77.70	199.82
c6288	2416	35	73.51	696.86	47	25.86	1133.67	16	85.28	394.17
c7552	4042	19	75.90	392.85	76	59.43	2197.37	7	80.56	318.29
average	1536	13	70.84	161.73	61	53.74	536.30	7	79.51	111.15

Table 2: Power saving and CPU time for supply voltage scaling and our simultaneously cell selection

Benchmark	MWIS-VS			MILP-VS			MILP-SIMU		
circuits	(%)	(-5%)	(s)	(%)	(s)	(Sh. #)	(%)	(s)	(Sh. #)
c499	1.92	0	0.52	11.24	0.88	24	68.94	8.44	0
c880	15.64	10.64	3.24	34.70	2.59	14	85.18	22.41	20
c1908	23.14	18.14	89.66	18.38	8.89	7	87.92	175.87	21
c2670	18.33	13.33	41.76	31.04	19.01	21	86.83	88.87	20
c5315	15.57	10.57	210.26	30.53	113.84	32	85.79	576.12	84
average	14.92	10.53	69.09	25.18	29.04	20	82.93	174.34	29

ISCAS'85 benchmark circuits using an IBM library show that our approach achieves more power saving and takes less CPU time comparing with other known cell selection approaches.

7 Acknowledgment

This research was supported in part by the National Science Foundation under Grant CCR-9623585, CCR-9988468 and MIP-9701416, and by an External Research Program Grant from HP Lab, Bristol, England.

References

[1] R. Bahar, G. Hachtel, E. Macii and F. Somenzi, "A symbolic method to reduce power consumption of circuits contains false paths," *ICCAD'94*, pp.368-371.

[2] C. Chen and M. Sarrafzadeh, "Power reduction by simultaneous voltage scaling and gate resizing," *ASP-DAC'2000*, pp. 333-338.

[3] W. Chuang, S. Sapatnekar, and I. Hajj, "Delay and area optimization for discrete gate sizes under double-sided timing constraints," *CICC'93*, pp.9.4.1-9.4.4.

[4] M. Khellah and M. Elmasry, "Minimization of high performance submicron circuits using a dual-V_{dd} dual-V_{th} (DVDV) approach," *ISLPED'99*, pp.106-108.

[5] IBM ASIC Databook, *http://www.chips.ibm.com/techlib/products/asics/databooks.html.*

[6] IBM LP Solutions, *http://www6.software.ibm.com es/oslv2/features/lp.htm.*

[7] P. Pant, V. De, and A. Chatterjee, "Simultaneous power supply, threshold voltage, and transistor size optimization for low-power operation of CMOS circuits," *IEEE Transactions on VLSI*, vol. 6, no. 4, 1998, pp.538-545.

[8] J. Rabaey, *Digital Integrated Circuits*, Prentice-Hall, 1996.

[9] J. Sheu, *et. al.*, "BSIM: Berkeley short-channel IGFET model for MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 22, 1987, pp.558-566.

[10] V. Sundararajan and K. Parhi, "Low power gate resizing of combinational circuits by buffer-redistribution," *Proceedings of 20th Anniversary Conference on Advanced Research in VLSI*, 1999.

[11] *Synopsys Design Compiler*, Synopsys Inc., 1998.

[12] K. Usami and M. Horowitz, "Cluster voltage scaling technique for low power design," *International Symposium on Low Power Design*, 1995, pp.3-8.

[13] J. Wang, S. Shieh, J. Wang, and C. Yeh, "Design of standard cells used in low-power ASIC's exploiting the multiple-supply-voltage scheme," *11th Annual IEEE International ASIC Conference*, 1998, pp.119-123.

[14] Q. Wang and S. Vrudhula, "An investigation of power delay trade-offs for dual V_t CMOS circuits," *ICCD'99*, pp.556-562.

[15] L. Wei, Z. Chen, K. Roy, M. Johnson, Y. Ye, and V. De, "Design and optimization of dual-threshold circuits for low-voltage low-power applications," *IEEE Transactions on VLSI*, vol. 7, no. 1, 1999, pp.16-24.