Gate Sizing and Replication to Minimize the Effects of Virtual Ground Parasitic Resistances in MTCMOS Designs

Chanseok Hwang, Changwoo Kang, Massoud Pedram
University of Southern California, Department of EE
Los Angeles, CA 90089
{chanseoh, ckang, pedram}@usc.edu

Abstract

The Multi-Threshold CMOS (MTCMOS) design technique achieves a significant reduction in sub-threshold leakage currents during the circuit sleep (standby) mode by adding high-V_{th} power switches (sleep transistors) to low-V_{th} logic cells. During the active mode of the circuit, the high-V_{th} transistors and the virtual ground network may be modeled as resistors, which in turn cause voltage of the virtual ground node to rise, thereby, degrading the switching speed of the logic cells. This paper introduces a new design methodology that minimizes the impact of virtual ground parasitic resistances on the performance of an MTCMOS circuit by using gate resizing and logic restructuring (i.e., gate replication.) Experimental results show that the proposed techniques are highly effective in making the MTCMOS circuits robust with respect to such parasitic resistance effects.

1. Introduction

MTCMOS techniques are in wide use especially for mobile battery-powered electronic systems. This is because the power consumption in these systems can be dramatically reduced during the standby-mode by adding high-V_{th} transistors to the bottom terminal of the pull-down network (PDN) of the (typically) low-V_{th} logic cells in the circuit (cf. Figure 1a.) The clock speed of the circuit is high because during the active mode of circuit operation, all switching logic cells have low-V_{th} transistors. At the same time, the subthreshold conduction leakage in the standby mode is low because of the stacked high-V_{th} sleep transistor connected to the bottom of the PDN of all logic cells in the circuit. The key to the success of this MTCMOS scheme is that (a) the sleep transistor is large enough so that its on-resistance in the active mode of the circuit operation is small enough such that it can have only a little impact on the switching speed of non-MTCMOS cells (say it can results in 5% or less delay penalty in the active mode); b) V_{th} of the sleep transistor (and to a much lesser degree its size) are chosen such that the leakage in the sleep mode of the MTCMOS logic cells is significantly smaller (e.g., by one order of magnitude) than that of the non-MTCMOS cells; c) The larger the size of the sleep transistor, the larger its layout area and the power dissipation to turn off or on this sleep transistor when transitioning in and out of the sleep mode. Clearly, it is challenging to meet all of the abovementioned goals and one must tradeoff lower leakage for higher area and dynamic power dissipation penalty or vice versa.

Due to aforesaid tradeoffs, sleep transistors present a non-negligible resistance on the PDN of the MTCMOS logic cells in the circuits during their active mode. In addition, it is not advisable to use a single sleep transistor for all the MTCMOS logic cells in the netlist since such a transistor will be very large, and at the same time, currents coming down thru the PDN's of all MTCMOS logic cells must be channeled thru this single sleep transistor. The use of a single exit terminal for the ground currents implies that the virtual ground network will have to be larger and more expensive in terms of its layout cost compared to the actual ground network (which is likely connected to multiple GND pins on the package.) Therefore, it is commonplace to split the sleep transistor into a series of parallel connected smaller sleep transistors and then route the current for a subset of the MTCMOS logic cells thru each one of these sleep transistors. This solution has the added benefit of distributed sleep transistor implementation (many small sleep transistors rather than one very large sleep transistor) and lower layout cost for the virtual ground network (wire segments in the virtual ground network can be made narrower and thinner than the wire segments in the actual ground network.) Examples of distributed sleep transistor layout designs are discussed in [1][2]. From now on, we will consider such distributed sleep transistor realizations.

To support the distributed sleep transistor layout design, we need a virtual ground network, which connects the bottom terminals of all the PDN's in the MTCMOS logic cells that are assigned to a sleep transistor to the drain of that sleep transistor. These interconnect lines have parasitic resistances and contribute capacitances to the virtual ground network (cf. Figure 2a.)

In this paper we propose a new design methodology that minimizes the impact of virtual ground parasitic resistances on the performance of an MTCMOS by using gate resizing and logic restructuring (i.e., gate replication.) The main idea is to utilize the relationship between the performance degradation and the size of low-V_{th} logic cells due to the virtual ground resistances. We show formulations and simulation results to prove the relationship which is then exploited by our proposed gate resizing and replication techniques.

2. Background

Figure 1(a) depicts a logic block, *LB*, in which a group of low-V_{th} logic cells are first connected to the virtual ground node and then through a high-V_{th} sleep transistor, S, to the actual ground, GND [3]. Figure 1(b) models the virtual ground interconnection and the high-V_{th} sleep transistor, which behaves like a linear resistor in the active mode of the circuit operation, as resistors R_i and R_s, respectively. The virtual ground is at voltage V_x above the actual ground, i.e.,

$$V_x = I \cdot (R_s + R_i) \tag{1}$$

where I is the current flowing through the virtual ground sub-network and the sleep transistor. The voltage drop across R, i.e., $R_s + R_i$, reduces the gate over-drive voltage of MTCMOS logic cells (i.e., their V_{GS} value) from V_{dd} to $V_{dd} - V_x$.

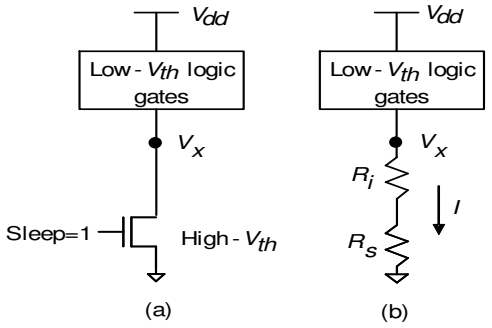

Figure 1: (a) Basic MTCMOS circuit structure, (b) The circuit model with sleep transistor and virtual ground interconnect modeled as resistors

When sleep transistors are absent, the 50%-input to 50%-output propagation delay for a CMOS logic cell may be expressed as

$$T_{pd} \propto \frac{C_L V_{dd}}{(V_{dd} - V_{tL})^{\alpha}} \tag{2}$$

where C_L is the load capacitance at the cell output, V_{tL} is the threshold voltage of the low-V_{th} cells, and α is the velocity saturation index (from the well-known alpha-power model.) In the presence of a sleep transistor and virtual ground interconnects, the gate propagation delay increases to

$$T_{pd\text{-}sleep} \propto \frac{C_L V_{dd}}{(V_{dd} - V_X - V_{tL})^{\alpha}} \tag{3}$$

As shown in Equation 3, when V_X increases, the propagation delay of a low-V_{th} gate gets worse. Here, we rewrite Equation 3 as

$$T_{pd\text{-}sleep} \propto \frac{C_L V_{dd}}{(V_{dd} - I(R_s + R_i) - V_{tL})^{\alpha}} \tag{4}$$

where I is the total saturation current that is sourced by low-V_{th} logic cells into the sleep transistor based on the assumption that the logic cells switch simultaneously. The saturation current of logic cell, *j*, is expressed as

$$I_j = \frac{\mu_n C_{ox}}{2} \frac{W}{L} (V_{dd} - V_X - V_{tL})^2 \tag{5a}$$

$$I = \sum_{k \in LB} I_k = I_j + \sum_{k \in LB \wedge k \neq j} I_k \triangleq I_j + I_{rem} \tag{5b}$$

Based on the above equations, we can see that the delay degradation of a low-V_{th} logic cell, *j*, relates to the size of the sleep transistor, $(W/L)_S \propto 1/R_s$, resistance of the virtual ground sub-network, R_i, the current that is sourced into the sleep transistor by all other cells assigned to it, I_{rem}, and the size of the cell itself, $(W/L)_j$. In other word, the amount of voltage drop on virtual ground depends on these four factors. In this paper, we study in detail the relationship between the performance degradation of a low-V_{th} gate and its size on the variation of virtual ground resistances.

3. Prior Work Review

Optimal sizing of the sleep transistors has been actively researched in MTCMOS designs. In [4], sleep transistor are modeled as resistors and then this model is used to bound the performance penalty for the worst case input vector. In [5], the authors size the sleep transistor of each cell to limit the performance degradation. Next, they merge sleep transistors whose discharge current patterns are mutually exclusive based on a unit delay model. In [6], the authors use a more precise delay model to accomplish the same objectives. In [7], the authors adopt an approach for reducing the transition time from the sleep mode to active mode of a circuit block while assuring power integrity for the rest of the system by restricting the current that flows to ground during the transition. In other approaches [2][6][8], the authors use placement information for optimizing sizes of the sleep transistors or by clustering gates and assigning them to some sleep transistor. In [8], the authors identify gates that are non-

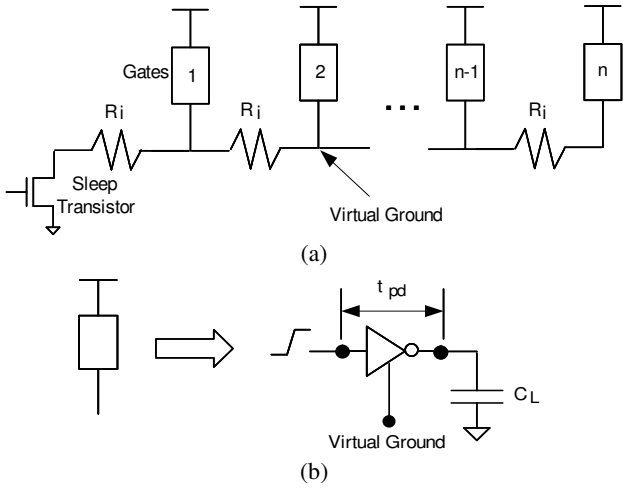

Figure2: (a) MTCMOS circuit structure, and (b) Simulation model for an inverter

critical and allow a larger increase in the delay of such gates in order to further reduce the overall circuit leakage.

Unlike the previous approaches, we attempt to minimize the performance degradation of low-V_{th} gates due to the voltage drops on its virtual ground network by gate resizing and logic restructuring without increasing the size of sleep transistors.

4. Key Observations

In the distributed sleep transistor layout design for the row-based standard cell design such as the one discussed in [2], sleep transistors are inserted on a row-by-row basis at the boundaries of each row. Therefore, logic cells in a row see different resistances for the virtual ground interconnect depending on their positions on the row as depicted in Figure 2(a).

Let T_{close} (T_{far}) be the propagation delays of a low-V_{th} logic cell when the cell is placed close to (far from) a sleep transistor, respectively. β is the far-end to close-end delay ratio, which characterizes the performance degradation of a logic cell due to a virtual ground interconnect, i.e.,

$$\beta = \frac{T_{far}}{T_{close}} \qquad (6)$$

We perform transistor-level simulations based on the MTCMOS circuit shown in Figure 2 to observe the variation of β with respect to the length of the virtual ground interconnects and the size of logic cells. We use five inverters discharging simultaneously, i.e., n=5 in Figure 2(a). Each inverter is separated from its neighbors by R_i and has a load capacitance of four times a minimum-size inverter (FO4.) We set the size of the sleep transistor so that its "on" resistance, R_s, is about 100Ω. Under these conditions, we measure the input-to-output propagation delay t_{PD} of gate$_1$ and gate$_5$ with the sleep transistor fully turned on. Subsequently, β of the logic cell is obtained by the ratio of $t_{PD}(5)$ to $t_{PD}(1)$.

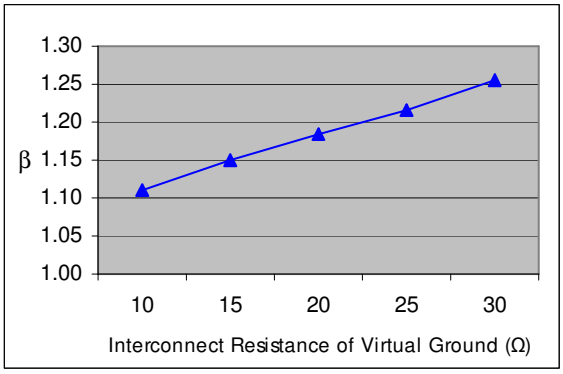

Figure 3: β-variation on the growth of length (resistance) of the virtual ground interconnects

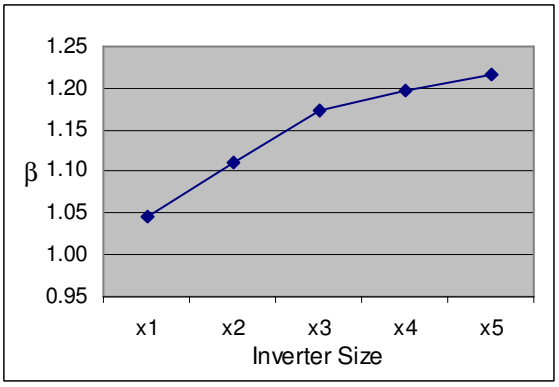

Figure 4: β-variation on different sizes of an inverter

As shown in Figure 3, β increases linearly as a function of R_i due to the voltage drop in the virtual ground interconnect. β goes up to a value of 1.25 by increasing R_i from 10 to 30Ω. This means that the performance degradation of the gate can be as high as 25% as a function of its distance from the sleep transistor.

Figure 4 shows the dependence of β on the size of the inverter. Here R_i is set to 10 Ω (this is a reasonable value based on a wire resistance of 0.05Ω/μm and a row length of >1000μm for a circuit of the size >50K gates in a 90nm CMOS technology.) As expected, β increases for a larger inverter (since R_s becomes smaller, the contribution of R_i to the delay goes up.) For example, the performance degradation for an inverter of size 4x is nearly four times greater than that for the minimum size inverter (size 1x.)

As we can see from the simulation results, the performance degradation of low-V_{th} logic cells in

MTCMOS circuits is strongly affected by their distance from the sleep transistor (and hence the resistance of virtual ground segment between these cells and the sleep transistor) and the size of the cells. The performance degradation of logic cells may greatly affect the overall circuit performance if logic cells with large β values happen to lie on the timing critical paths of the circuit. Furthermore, since the β value for each cell in the netlist is known only during/after placement, a path which is non-timing-critical during the logic design stage can easily become timing-critical if one or more of its logic cells assume large β values during the placement stage.

To address these problems, one may place cells in the timing-critical paths or those paths that are close to becoming timing-critical, closer to the sleep transistors. Alternatively, one may size up those gates that get a large β value so as to compensate for the performance loss due to higher virtual ground resistance by reducing the driver resistance. However, these approaches can perturb the placement solution, which may adversely impact the timing closure of the design.

5. Problem Formulation

We set to optimize a MTCMOS circuit to become robust with respect to the voltage drop variations on its virtual ground network. The key advantage of such an optimized circuit is that one need not worry as much about issues related to the virtual ground voltage drops and the resulting performance degradations during the subsequent optimizations (for example, during the cell placement step.) We achieve this goal by appropriate gate sizing and logic restructuring of the original circuit.

Let *C'* be the new circuit after gate resizing and logic restructuring of some original circuit *C*. $D(\chi)$ is the critical path delay of circuit χ. Let δ_i denote the *sensitivity* of a logic cell i to the voltage drop variation on the virtual ground network, which is expressed as

$$\delta_i = T_{far,i} - T_{close,i} \tag{7a}$$

In Equation 7, the sensitivity of a logic cell represents the worst-case increase in propagation delay of the cell due to the voltage drop on its virtual ground connection to the sleep transistor. Since each cell exhibits a different value of δ depending on its size and logic function, the *minimum-sum* optimization version of the problem[1] can be stated as follows:

$$\begin{aligned} &Minimize \quad \Upsilon \triangleq \sum_{i=1}^{N} \delta_i \\ &s.t. \qquad D(C') \le D(C) \end{aligned} \tag{7b}$$

We optimize the size of gates in a given MTCMOS circuit in order to reduce the sensitivity of the circuit to the voltage drop variation in the virtual ground network. However, we are not willing to sacrifice any performance loss with respect to the original circuit. The assumption is that we have a cell library where, for each logic function, there exist multiple library cells matching that function, each having a different size.

6. Proposed Solution

In this section, we describe two algorithms to minimize the sensitivity of a MCTMOS circuit while satisfying the delay constraint. In our approach, we first apply our optimization algorithm for gate sizing to a given circuit and then execute a simple but effective logic restructuring technique for timing-critical paths in the circuit.

6.1 Gate Sizing Algorithm

To reduce the sensitivity of a circuit without timing violations, we resize gates by using the slack, which is defined as the required time minus the arrival time. Our approach is based on the gate sizing algorithm developed in [9] and detailed in Figure 5. The arrival time and required time are first propagated in the circuit and the slack is calculated. All possible gate sizing choices are evaluated for each gate. The move that has the best *fitness* value is selected as the best move among the candidates. The fitness is defined as follows:

$$Fitness = \begin{cases} 0 & \text{if } \Delta\Upsilon > 0 \text{ or } S_0 + \Delta S < 0 \\ |\Delta\Upsilon| \cdot \Phi\left(\dfrac{\Delta S}{S_0}\right) & \text{otherwise} \end{cases}$$

where

$$\Phi(x) = \begin{cases} 1+x & \text{if } x \ge 0 \\ \dfrac{1}{1-x} & \text{otherwise} \end{cases}$$

In this formulation, S_0 and ΔS are the minimum slack and a variation of the minimum slack as a result of a candidate move. To get the variation of the minimum slack, we search only in a local neighborhood, which is defined as gates within a user-specified level from the source of the move. In [9], the concept of a local neighborhood is proved as an effective way to quantify the benefit of each move. It has been observed that slack change outside the neighborhood tends to be very small and can be neglected for fitness calculation. The fitness function balances the gain in the sensitivity, $\Delta\Upsilon$, with a delay dependent function ϕ that acts as a benefit/penalty function. It takes

[1] There is a *minimax* version of the problem that we do not address in this paper.

```
GSMT Algorithm (C: circuit, L: technology library)
C_cur=C;
Calculate_Slacks(C_cur);
newcost= Cost(C_cur);
loop {
  prevcost = newcost;
  foreach logic cell i ∈ C_cur {
    i.move = 0;
    i.fitness = EvalFitness(C_cur);
    foreach library cell j ∈ L realizing logic cell I {
      tempfit = EvalFitness(C_cur[i ← j])
      if (tempfit > i.fitness) {
        i.move= j;
        i.fitness = tempfit;
      }
    }
  }
  moved = GenApplyMoveSequence(C_cur);
  C_cur = UpdateNetlist(moved, C_cur);
  Calculate_Slacks(C_cur);
  newcost = Cost(C_cur);
} until Convergence(prevcost, newcost)
return C_cur;
```

Figure 5: Gate sizing algorithm for MTCMOS circuits

into account how much sensitivity and slack are won or lost, and how critical the node is.

After evaluating the fitness for all cells, the best move for each logic cell is selected based on its fitness value. Next, a sequence of moves is determined to maximize the overall fitness gain, which is achieved by the function *GenApplyMoveSequence* in Figure 5. In the function, we put cells into a heap where cells are sorted by their gain (highest gain move is root of the heap.) Next we extract the root cell from the heap. Whenever a move executes, we update gains of cells in the neighborhood of the move and, as a result, the heap is restructured. This process continues until the heap is empty. The running sum of the total fitness gains for the moves is constructed during this process to identify an optimal sequence of moves that produces the maximum total gain. Moves that are not part of the accepted move sequence are reversed and the circuit is updated by function *UpdateNetlist*. We go through multiples of this procedure until no further gain can be achieved, which is checked by function *Convergence* using the cost function defined as follows

$$Cost(circuit) = \begin{cases} +\infty & \text{if } S(circuit) < 0 \\ \Upsilon & \text{otherwise.} \end{cases}$$

where $S(circuit)$ is the minimum slack and Υ is the total sensitivity of the circuit as defined in Eqn. 7b.

6.2 Logic Restructuring

We can obtain an optimized circuit having lower sensitivity compared with the original circuit while satisfying the delay constraint by using the GSMT algorithm. However, due to the delay constraint, GSMT is somewhat constrained as to how much it can reduce the circuit sensitivity to the voltage drops on the virtual ground network. In this section, we present a simple, yet effective, logic restructuring mechanism to further reduce the sensitivity for the cells on the timing critical paths of the circuit.

We show a logic restructuring example in Figure 6 to illustrate our method. In this example, the second inverter of size 2x, which is driving two other inverters of size of 2x in Figure 6(a), is replaced by two inverters of size of 1x as shown in Figure 6(b). This *gate replication* method is effective due to the following reasons. 1) The sensitivity of a logic cell decreases as its size is reduced. 2) The delay of a path will remain the same if the size ratios of driver gates to fanout gates on the path remain unchanged. The first fact can be easily proved by using Equation 4 and 5, and the simulation results in Section 4. The second fact can be proved using the method of *logical effort* [10]. Based on this method, the *effort delays* of the two circuits in Figures 6(a) and 6(b) are same since the logical effort is independent of the size of the transistors in the circuit.

In this example, the sensitivity of the new circuit is reduced by the difference of the sensitivities of the two cells (i.e., $\Delta\Upsilon = \delta_{INVX2} - \delta_{INVX1}$.) The number of gates increases by $N + i$, where i is the number of replicated cells, while the total circuit area remains nearly the same because the sum of areas of two logic cells of size 1x is similar to that of the a logic cell of size 2x. We apply this restructuring method to all cells on the timing-critical paths of the target circuit.

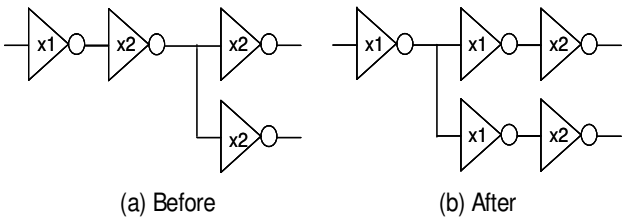

Figure 6: Logic restructuring

7. Experimental Results

Our experiments were done in the SIS environment. All benchmarks were first optimized using SIS script "script.rugged" and timing-driven technology mapped to an industrial-strength 100nm standard cell library. Next, we ran the GSMT algorithm and gate replication technique (GSMT+GR) to the mapped netlists. All

benchmarks were run on SUN Ultra Spark II machine. We took the high V_{th} values for PMOS and NMOS as -303mV and 260mV, and the low V_{th} values for PMOS and NMOS as -250mV and 200mV, respectively.

We run HSPICE for all library cells to obtain the sensitivity in the way that is used for the inverter in Figure 2 (R_i is set to 10Ω and n=5.) The obtained sensitivity of each gate in this simulation can be considered a worst-case value since it is based on the assumption that the gates are switching simultaneously and the length of virtual ground interconnect is the row length of a circuit. In addition, our simulation results show that the sensitivity of each logic cell is mainly related to the size of the logic cell and the maximum length of virtual ground interconnects. Therefore, we used fixed values for the input slew and output loading capacitance.

Table 1 shows comparison results between the original circuits and the optimized ones obtained by running GSMT+GR in terms of the total area (A) and total sensitivity, Υ . As shown in Table 1, for the all benchmarks, we have significant reductions in the total sensitivity at a small increase in the total cell area. The maximum delay of the original circuit for every benchmark is not increased. In the last column, we show the sum of run times for GSMT+GR.

Table 1. Experimental Results of GSMT+GR.

Circuit	Original Circuit			Optimized Circuit		CPU
	Max Del.	Area	Tot. Sensit.	Area	Tot. Sensit.	(s)
C432	2.31	727.1	1469.0	739.6	919.2	4.5
C499	1.62	1574.2	3222.6	1593.2	2994.8	12.9
C880	1.78	1204.6	2474.2	1215.8	1517.1	5.3
C1355	1.85	1677.0	3349.7	1737.2	3107.5	14.8
C1908	2.28	2154.7	4399.8	2211.8	3063.7	13.2
C3540	3.28	4172.8	8437.3	4227.9	5254.4	35.5
C5315	2.87	6787.1	13705.7	6886.4	8155.6	20.4
C6288	8.18	6749.2	13098.8	7009.5	9965.7	73.4
C7552	3.46	9064.5	18571.2	9197.5	11688.1	22.8
Avg.		**1**	**1**	**1.02**	**0.74**	

8. Conclusions

We have studied the performance degradation of MTCMOS circuits due to the voltage drop on the virtual ground, and observed that the performance degradation is strongly related to the size of low-V_{th} gates. In order to utilize this relationship, we quantified the performance degradation of the gates as sensitivity and proposed new methods for gate resizing and logic restructuring. Experimental results show that the proposed techniques are highly effective in making the MTCMOS circuits robust with respect to the voltage drop variations on its virtual ground network.

9. References

[1] C. Long and L. He, "Distributed Sleep Transistor Network for Power reduction", In *Proceedings of the ACM/IEEE DAC*, 181-186, 2003.

[2] P. Babighian, L. Benini, A. Macii and E. Macii, "Post-Layout Leakage Power Minimization Based on Distributed Sleep Transistor Insertion", In *Proceedings of the ISLPED*, 138-143, 2004.

[3] S. Mutoh et al. "1-V Power Supply High Speed Digital Circuit Technology with Multithreshold-Voltages CMOS", In *IEEE JSSC*, vol. 30, no.8, Aug. 1995.

[4] J. Kao, A. Chandrakasan and D. Antoniadis, "Transistor Sizing Issues and Tool for Multi-Threshold CMOS Technology", In *Proceedings of the ACM/IEEE DAC*, 409-414, 1997.

[5] J. Kao, S. Narenda and A. Chandrakasan, "MTCMOS Hierarchical Sizing Based on Mutual Exclusive Discharge Patterns", In *Proceedings of the ACM/IEEE DAC*, 495-500, 1998.

[6] M. Anis, S. Areibi, M. Mahmoud and M. Elmasry, "Dynamic and Leakage Power Reduction in MTCMOS Circuits Using an Automated Efficient Gate Clustering Technique", In *Proceedings of the ACM/IEEE DAC*, 480-485, 2002.

[7] A. Abdollahi, F. Fallah and M. Pedram, "An Effective Power Mode Transition Technique in MTCMOS Circuits", In *Proceedings of the ACM/IEEE DAC*, 37-42, 2005.

[8] V. Khandelwal and A. Srivastava, "Leakage Control through Fine-Grained Placement and Sizing of Sleep Transistors", In *Proceedings of the ACM/IEEE ICCAD*, 533-536, 2004.

[9] O. Coudert, "Gate Sizing for Constrained Delay/Power/Area Optimization", In *IEEE Trans. on VLSI*, 465-472, Dec. 2005.

[10] Sproull, R. F., and I. E. Sutherland, "Logical Effort: Designing for Speed on the Back of an Envelop", *IEEE Advanced Research in VLSI*, C. Sequin (editor), MIT Press, 1991.