Leakage Control Through Fine-Grained Placement and Sizing of Sleep Transistors

Vishal Khandelwal, *Member, IEEE*, and Ankur Srivastava, *Member, IEEE*

ABSTRACT

Multi-Threshold CMOS (MTCMOS) technology has become a popular technique for standby power reduction. Sleep transistor insertion in circuits is an effective application of MTCMOS technology for reducing leakage power. In this paper we present a fine grained approach where each gate in the circuit is provided an independent sleep transistor. Key advantages of this approach include better circuit slack utilization and improvements in ground bounce related signal integrity (which is a major disadvantage in clustering based approaches). To this end, we propose an optimal polynomial time fine grained sleep transistor sizing algorithm. We also prove the selective sleep transistor placement problem as NP-Complete and propose an effective heuristic. Finally, in order to reduce the sleep transistor area penalty, we propose a placement area constrained sleep transistor sizing formulation. Our experiments show that on an average the sleep transistor placement and optimal sizing algorithm gave 50.9% and 46.5% savings in leakage power as compared to the conventional fixed delay penalty algorithms for 5% and 7% circuit slowdown respectively. Moreover the post placement area penalty was less than 5% which is comparable to clustering schemes [11].

I. Introduction and Background

In recent years, technology scaling has increased the role of leakage power in the overall power consumption of circuits. Multi-threshold CMOS (MTCMOS) has emerged as an effective technique for reducing sub-threshold currents in the standby mode while maintaining circuit performance. MTCMOS technology essentially places a sleep transistor on gates and puts them in sleep mode when the circuit is non-operational. State of the art techniques in leakage optimization using MTCMOS essentially assign a sleep transistor to each gate and size them such that all gates have a fixed slowdown. This is followed by a clustering approach that groups together gates with mutually exclusive switching patterns to share a sleep transistor thereby reducing the area penalty and leakage. There are several problems in this approach that we address in our work. Firstly, the traditional approach sizes the sleep transistors such that all gates have the same slowdown. It does not investigate the possibility of slowing down non-critical gates more than critical gates for better improvements in leakage. Secondly, it has been shown that clustering MTCMOS gates has adverse effects on signal integrity due to ground bounce issues [11], [2], [10]. In this paper we address these issues by developing a fine grained methodology for MTCMOS based leakage optimization.

As shown in figure 1(a), low V_t logic modules or gates are connected to the virtual supply rails through high V_t sleep transistors [14] which behave similar to a linear resistor in active mode as shown in figure 1(b). The high threshold sleep transistor is controlled using the $Sleep$ signal and limits the leakage current to a low value in the standby mode. In this work we have chosen to use both nMOS and pMOS sleep transistors. The proposed scheme can be easily modified to work with only a single set of sleep transistors.
The load dependent delay d^i of a gate i in the absence of a sleep transistor can be expressed as

$$d^i \propto \frac{C_L V_{dd}}{(V_{dd} - V_{tL})^\alpha} \quad (1)$$

where C_L is the load capacitance at the gate output, V_{tL} is the low voltage threshold = 350 mV, V_{dd} = 1.8 V and α is the velocity saturation index ($\approx$ 1.3 in 0.18-μm CMOS technology). In the presence of a sleep transistor, the propagation delay of a gate can be expressed as

$$d^i_{sleep} = \frac{K C_L V_{dd}}{(V_{dd} - 2V_x - V_{tL})^\alpha} \quad (2)$$

where V_x is the potential of the virtual rails as shown in figure 1 and K is the proportionality constant. We can see that now in the active mode, the logic gate operates not at the true supply rails (V_{dd}), but at the virtual supply rails that are offset by a magnitude V_x from the supply rails on either side. Hence, the effective supply voltage seen by the gate is V_{dd} - $2V_x$.

Let us suppose $I_{sleep_{ON}}$ is the current flowing in the gate during active mode of operation. During this mode, the sleep transistor is in the linear region of operation. Using the basic device equations for a transistor in linear region, the drain to source current in the sleep transistor (which is the same as $I_{sleep_{ON}}$) is given by

$$I_{sleep_{ON}} = \mu_n C_{ox} (W/L)_{sleep} ((V_{dd} - V_{tH}) V_x - \frac{V_x^2}{2}) \quad (3)$$

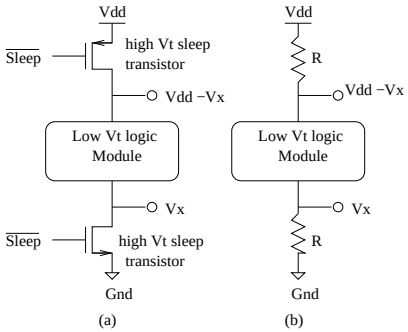

Fig. 1. Sleep Transistor in MTCMOS Circuits

$$I_{sleep_{ON}} \simeq \mu_n C_{ox}(W/L)_{sleep}(V_{dd} - V_{tH})V_x \quad (4)$$

The sub-threshold leakage current I_{leak} in the sleep mode will be determined by the sleep transistor and is expressed as given by [13]

$$I_{leak} = \mu_n C_{ox}(W/L)_{sleep} e^{1.8} V_T^2 e^{\frac{V_{gs}-V_{th}}{nV_T}}(1 - e^{\frac{-V_{ds}}{V_T}}) \quad (5)$$

where μ_n is the N-mobility , C_{ox} is the oxide capacitance, V_{tH} is the high threshold voltage (= 500 mV), V_T is the thermal voltage = 26mV and n is the sub-threshold swing parameter.

Equation 2 establishes a relation between delay of a gate d^i_{sleep} and V_x. By replacing V_x in equation 4 in terms of d^i_{sleep} (using equation 2), we get a dependence between $(W/L)_{sleep}$ and d^i_{sleep} (assuming the ON current is constant for each gate). Thus, a range of $(W/L)_{sleep}$ for the sleep transistor would correspond to a range of gate delays. Finally, $(W/L)_{sleep}$ in equation 5 can be replaced in terms of d^i_{sleep}, hence establishing a relationship between gate delay and gate leakage. The final relation between leakage and delay can be expressed as

$$I_{leak} = \mu_n C_{ox} e^{1.8} V_T^2 e^{\frac{V_{gs}-V_{th}}{nV_T}}(1 - e^{\frac{-V_{ds}}{V_T}}) \times \frac{I_{sleep_{ON}}}{\mu_n C_{ox}(V_{dd} - V_{tH})} \times \frac{{d_{sleep}}^{1/\alpha}}{(V_{dd} - V_{tL}){d_{sleep}}^{1/\alpha} - (KC_L V_{dd})^{1/\alpha}} \quad (6)$$

This relationship exists for only those gates that have a sleep transistor assigned to them. Note that the moment a sleep transistor is assigned, some delay penalty is incurred. The range of delay that a gate can have is decided by the range of the acceptable $(W/L)_{sleep}$. The objective of sleep transistor sizing is to decide the best values of $(W/L)_{sleep}$ for all sleep transistors such that the global delay constraint is satisfied and the total leakage is minimized.

We look at a fine-grained sleep transistor placement and sizing methodology in this work. Our approach controls the placement and sizing of sleep transistors selectively in gate level circuits. In this work we propose an optimal polynomial time formulation for the fine-grained sleep transistor sizing problem. The key idea is to identify gates that are non-critical and increase their delay more than that of the critical gates such that the overall leakage is minimized and the delay constraint is satisfied. Similar ideas about using non-uniform delay degradation have been proposed earlier in [8], [9]. We also address the sleep transistor placement and sizing problem and prove it to be NP-Complete. A dynamic programming based heuristic is proposed to solve the same. Since the fine-grained scheme has potential for a very high area penalty, we present a standard cell placement driven sizing methodology. We show that the area penalty incurred by our proposed scheme is comparable with that of other existing clustering based schemes that consider standard cell placement [11]. The work presented is this paper is orthogonal to the existing body of work [8], [11], [3] in terms of proposing to leverage the timing slack in the design for sizing sleep transistors in an automated MTCMOS design methodology.

Our experimental results show that on an average the Optimal Sizing Algorithm gave 50.9% and 46.5% savings in leakage power as compared with the conventional *fixed* delay penalty algorithms for 5% and 7% global circuit slowdown respectively. Moreover, the area penalty in our approach was minimal (less than 5%).

The rest of the paper is organized as follows: section II discusses the issue of noise margins and signal integrity, section III describes our polynomial time optimal sizing algorithm, section IV describes the selective placement and sizing heuristic, section V discusses the proposed standard cell placement based sizing formulation, section VI discusses our experimental results and section VII contains the conclusions drawn from this work.

II. Signal Integrity and Noise Margin Issues

The existing literature on MTCMOS circuits [11], [8], [6], [7], [10] presents clustering based approaches for sleep transistor insertion. Clustering of gates has adverse effects on circuit performance due to the virtual ground bounce problem [11], [10], [2]. In a MTCMOS scheme, the logic gates operate at virtual supply rails which tend to bounce with the charging/discharging of the logic gates. This varying voltage depends on the number of gates within the cluster that are switching simultaneously at a given point in time and their corresponding discharge currents. Ground bounce can have a severe effect on gate speed and noise immunity.

In contrast, our fine-grained scheme does not share the virtual ground across multiple gates and we do see adverse effects on performance due to ground bounce issues. Since the sleep transistor at each gate is individually sized, the speed degradation and noise margins can be accurately controlled. In [10], the authors suggest that the ground bounce problem can be eliminated by using fine-grained sleep transistors as opposed to clustered sleep transistors.

In [6], the authors discuss the presence of reverse conduction paths in clustered MTCMOS logic blocks. These reduce the noise margins of the circuit and in the worst case cause the circuit to fail logically. Furthermore, as pointed out in [2], fine-grained sleep transistor placement allows us to guarantee circuit speed. This is inherently true because if sleep transistors are placed at gate level, we can exactly calculate the worst case delay of each gate through a gate-level simulation.

III. Optimal MTCMOS Sizing

We propose a novel polynomial time optimal sizing algorithm that tries to maximize the utilization of the existing slack in the circuit for additional savings in leakage power. The sizing algorithm is independent of the placement problem and can be defined as follows:

Given a circuit along with the location of the sleep transistors, the arrival time at each primary input and a required time constraint at each primary output, optimally size the sleep transistors for minimal leakage while satisfying the delay constraints on the circuit.

We propose a sizing formulation that has a convex and separable objective function under a set of linear constraints (hence optimally solvable in polynomial time [4]). We exploit the relationship between gate delay and transistor size formulated in equations 2 and 4. We use the load dependent delay model as given by equation 2. Essentially we budget the delay of each gate (on which there is a sleep transistor) such that the total leakage is minimized and the delay constraint is satisfied. For each gate with a sleep transistor, there exists a range of delays which is decided by the range of sleep transistor sizes that are of interest. We optimally assign delay budgets to each gate with sleep transistor in the circuit such that our objective function, which is the sum of the leakage power over all gates is minimized.

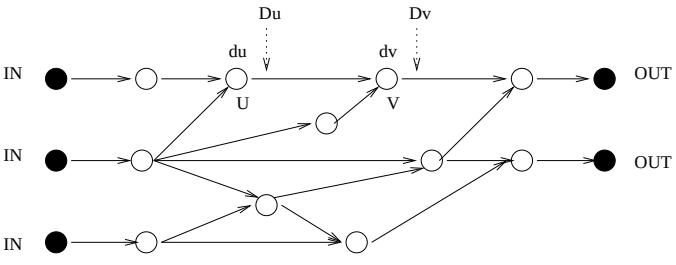

Fig. 2. DAG representation

We represent our circuit as a directed acyclic graph (DAG) $G(V, E)$ as shown in figure 2. Each node in the DAG represents a gate in the circuit. We are also provided with a set S of gates which have sleep transistor. We add a dummy *IN* node before each of the primary inputs which are shown as the black nodes marked *IN* in figure 2. We also add a similar dummy node *OUT* after each of the primary outputs which are shown as the black nodes marked *OUT* in figure 2. For each node u, we associate a variable d_u which represents the delay of that node. We also associate another variable D_u with each node which represents the arrival time at the output of node u. Now we consider two nodes u and v as shown in figure 2. Their corresponding variables have also been shown in the figure. The timing constraints on G(V,E) can be modeled as

$$d_v - D_v + D_u \leq 0 \quad \forall\ e(u,v) \in E \tag{7}$$

$$d^i_{min} \leq d_i \leq d^i_{max} \quad \forall\ vertex\ i \in V, i \in S \tag{8}$$

$$d_i = d^i_{cons} \quad \forall\ vertex\ i \in V, i \notin S \tag{9}$$

$$D^i_{IN} = T^i_{arrival} \quad \forall\ vertex\ i \in IN \tag{10}$$

$$D^i_{OUT} \leq T^i_{con} \quad \forall\ vertex\ i \in OUT \tag{11}$$

$$d^i_{IN} = 0 \quad \forall\ vertex\ i \in IN \tag{12}$$

$$d^i_{OUT} = 0 \quad \forall\ vertex\ i \in OUT \tag{13}$$

For all the IN nodes, the D_{IN} values have been set to the corresponding arrival time values $T_{arrival}$ for the signals. The delay of the IN nodes denoted by d_{IN} have been set to zero. Similarly for the OUT nodes, their delay d_{OUT} values have been set to zero and the corresponding D_{OUT} values have been set to be less than or equal to the required time constraint T_{con} at the corresponding primary output node. The above formulation assigns delay budgets to all the gates of the circuit which belong to the set S (have sleep transistors) in such a way as to utilize as much slack available. The gates which do not have a sleep transistor ($\notin S$) have a constant delay given by d_{cons}. Each sleep transistor has a range of sizes it can take which implies that the corresponding gate has a range of possible delay budget which is denoted by the [d_{min},d_{max}]. The objective of optimal sizing is to minimize the total leakage power of the circuit which can be represented as

$$min(\Sigma_{i \in V} P_i) = min(\Sigma_{i \in V} V_{dd} I^i_{leak}) \tag{14}$$

As illustrated before the dependence between leakage and gate delay (on which there is a sleep transistor) is given as follows

$$I_{leak} = \mu_n C_{ox} e^{1.8} V_T^2 e^{\frac{V_{gs}-V_{th}}{nV_T}} (1 - e^{\frac{-V_{ds}}{V_T}}) \times \frac{I_{sleep_{ON}}}{\mu_n C_{ox}(V_{dd} - V_{tH})} \frac{d^{i^{1/\alpha}}}{(V_{dd} - V_{tL}) d^{i^{1/\alpha}} - (K C_L V_{dd})^{1/\alpha}} \tag{15}$$

Theorem: Optimal Sizing Algorithm is polynomial time solvable

Proof: From [4], we know that a problem formulation with a convex separable objective function under a set of linear constraints is polynomial time solvable. Let us consider the minimization objective function in equation 14. We can see that it is a separable function since each term P_i depends only on the variable d^i as seen from equation 15. Hence, grouping together all the other constant symbols into constants K_1 , K_2 and K_3, we can represent P_i from equations 14 and 15 as

$$P_i = \frac{K_1 d^{i^{1/\alpha}}}{K_2 d^{i^{1/\alpha}} - K_3^{1/\alpha}} \tag{16}$$

$$= \frac{1}{K_4 - K_5 d^{i^{-1/\alpha}}} \tag{17}$$

where K_4 and K_5 are positive constants. We know that $-1/\alpha = -0.77$ and therefore $d^{i^{-1/\alpha}}$ is a convex function. A function $f(x)$ is a strictly convex function if $f(\lambda x_1 + (1-\lambda)x_2) < \lambda f(x_1) + (1-\lambda) f(x_2)$. Simple manipulations can show that $d^{i^{-1/\alpha}}$ is strictly convex. Since K_5 is a positive constant, $K_5 d^{i^{-1/\alpha}}$ is also a strictly convex function. Therefore, $-K_5 d^{i^{-1/\alpha}}$ is a concave function. K_4 is a positive constant, hence $K_4 - K_5 d^{i^{-1/\alpha}}$ is a concave function which by definition is positive as well. Therefore, $P_i = \frac{1}{K_4 - K_5 d^{i^{-1/\alpha}}}$ is a convex function (inverse of a positive concave function). Thereby we have shown that the objective function is convex separable. Hence, according to the result from [4], Optimal Sizing Algorithm is polynomial time solvable.

IV. Selective Sleep Transistor Placement and Sizing

In delay critical circuits, it is not possible to place a sleep transistor at each gate. The proposed fine-grained scheme allows us to selectively place sleep transistors at some of the gates while meeting the delay constraint. In this section we propose a novel heuristic for selectively placing and sizing the sleep transistors. Formally, the problem can be defined as follows

Given a gate level design represented as a Directed Acyclic Graph (DAG) and a delay constraint at the output, selectively place and size sleep transistors at individual gates such that the overall leakage is minimized and the delay constraint is satisfied.

Theorem: The optimal sleep transistor placement and sizing problem is NP-Complete.
Proof: Detailed proof is omitted for brevity.

We propose a heuristic to solve this problem. Before we delve into the details of the heuristic, we address another important issue. Our selective placement formulation causes interaction between gates which are MTCMOS and CMOS. During the standby state, the MTCMOS gate output can get into a floating state [7] resulting in short circuit current dissipation. In [7], the authors propose to use a leakage feedback gate structure as shown in figure 3. We note that there are two additional high threshold voltage transistors (labeled Helper X and Y) connected in parallel with the sleep transistors. The helper transistors alongwith the extra inverter connected to the feedback from the output (that drives these helper transistors) help preserve the output logic state of the MTCMOS gate. Hence, at every MTCMOS-CMOS interface we can use this leakage feedback gate structure for the MTCMOS so that we can safely drive a CMOS gate at its output without creating any short circuit currents. We will refer to the leakage feedback structure as sleep transistor of Type II throughout this paper. The leakage feedback gate that we have proposed to use in this work requires both nMOS and pMOS transistors to function correctly. The proposed scheme is independent of the exact structure and other possible designs can also be used.

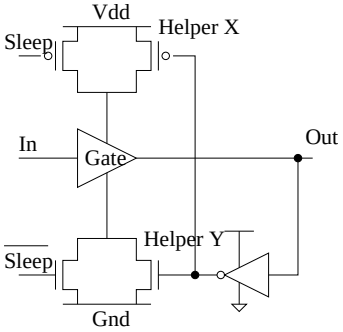

Fig. 3. Leakage Feedback Gate

A. Heuristic for Selective Placement and Sizing

The proposed two-step heuristic is a dynamic programming formulation. The arrival time at the output of gate g can be atmost equal to the required time at that gate for correct operation. In the first step, we traverse the circuit topologically from primary outputs to primary inputs generating a leakage versus required time trade-off curve at the input of each gate. At any gate g, every point on its trade-off curve represents one solution of sleep transistor placement (and its size) at each gate in the fanout-cone of gate g. While generating this trade-off curve at gate g, we consider all possible combinations of sleep transistor placement (and sizing) at gate g (and implicitly consider all combinations at the fanout gates of gate g as well). In the second step, we move topologically from the primary inputs to the primary outputs. At each gate g, we decide whether a sleep transistor gets placed (and its size) based on the trade-off curve at that gate and the arrival time constraints posed by the fanin gates of gate g (which have already been assigned a sleep transistor placement solution). We choose the minimum leakage solution that satisfies this arrival time constraint.

In the first step the heuristic traverses the circuit topologically from the primary outputs to the primary inputs generating a leakage versus required time trade-off curve at each gate. Let us suppose that we are at gate A as shown in figure 4 that has two fanout gates (gates P and Q). Since we traverse the graph topologically, we would have already generated the trade-off curve at these gates which are shown as Curve P and Q in the figure. First, we need to combine these trade-off curves at the fanout gates into a single curve at the output of gate A which would represent the trade-off curve between leakage and required time for the fanout gates of gate A (shown in figure 4 as curve X). We note that a point on the curve P (or Q) signifies the sleep transistor placement solution at each gate in the fanout-cone of gate P (or Q). This solution includes the placement of sleep transistors (alongwith their type I or II) and their respective sizes. Let us now understand how to merge curves P and Q to get curve X. Each point on the merged curve X would correspond to a specific point on both curves P and Q. Suppose we are trying to merge two points, one with required time $T1$ (and leakage $LP1$) on curve P and other with required time $T2$ (and leakage $LQ2$) on curve Q as shown in the figure. The resulting required time T and leakage L will be given as:

$$T = min(T1, T2) \tag{18}$$

$$L = LP1 + LQ2 \tag{19}$$

Let us suppose that we are generating the point at required time $T0$ as shown on curve X. The idea is to get a minimum leakage solution at this value of required time at the output of gate A. Hence, we need to evaluate all combinations of points with required times $T1$ and $T2$ on curves P and Q respectively such that $T0 \leq min(T1, T2)$. The combination that results in minimum leakage (using equation 19) will be chosen as the final solution. We now make an important

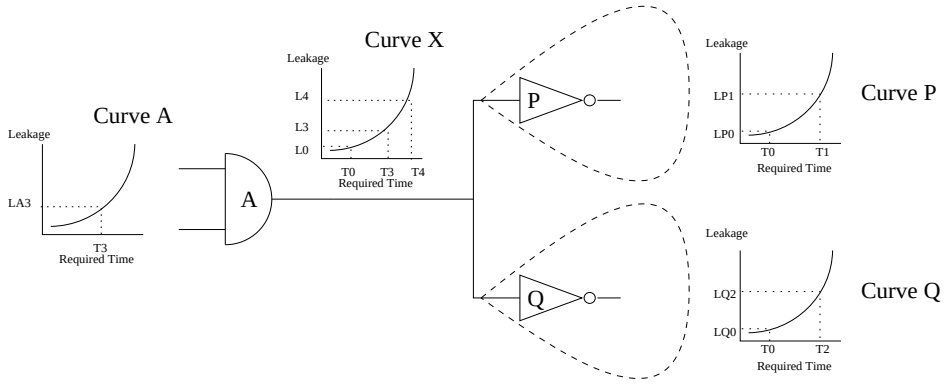

Fig. 4. Merging of Trade-Off Curves

observation: leakage has a monotonically increasing relationship with required time. Hence, the lowest leakage solution such that $T0 \leq min(T1, T2)$ will be when $T1 = T0$ and $T2 = T0$. Therefore, the solution at point $T0$ on curve X will be generated as follows:

$$To = min(T1 = T0, T2 = T0) \quad (20)$$

$$L0 = LP0 + LQ0 \quad (21)$$

Similarly we can generate any point on curve X (say with required time T) by merging points with required time T on curves P and Q. Therefore merging curves P and Q into curve X corresponds to a simple addition of curves P and Q for each value of required time. Now that we have generated curve X at the output of gate A, we need to combine this with the possible sleep transistor placements at gate A (namely Type I or II or no sleep transistor) and the possible sizes to generate the trade-off curve A at the input of gate A. Depending on the size of the sleep transistor placed at gate A, its delay D and leakage L will be decided (size corresponds to delay through equation 2 and 4 and leakage through equation 5). Additionally, to place a sleep transistor at a gate there is a minimum delay penalty that has to be incurred. Depending on the size of the sleep transistor, the delay can increase from this minimum delay penalty value. If no sleep transistor has been placed, then its leakage and delay are unchanged from their base case values for that particular gate. Let us suppose that we are trying to generate the point with required time $T3$ on curve A. Clearly, the only points on curve X that can be used to generate this point are those with required time $T \geq T3$. Let us suppose that we are at point $T4$ on curve X. The allowed value of gate delay D for this point to be translated to point $T3$ on curve A is given by:

$$D = T4 - T3 \quad (22)$$

If this value of gate delay D is greater than the minimum delay required to place a sleep transistor at gate A, then we can evaluate the size of the sleep transistor (of type I or II) for this value of gate delay D and its leakage L. Thus the leakage of the point $T3$ on curve A would now be:

$$LA3 = L4 + L \quad (23)$$

As discussed earlier in this section, we cannot drive a CMOS gates with an MTCMOS gate of type I due to the existence of short circuit currents at such interfaces. Hence, if the solution point $T4$ on curve X corresponds to any one of the fanout gates of gate A (namely gate P and Q) being a CMOS gate, then we can only place a Type II sleep transistor on gate A. The heuristic needs to make these considerations during this curve generation step. Hence, we evaluate all possible points with required time T on curve X to generate point $T3$ on curve A (which means $T \geq T3$). The minimum leakage point is chosen as the final solution for curve A with required time $T3$. Similarly, we generate every point on curve A from curve X. Hence, at the end of the first step of the heuristic we have a leakage versus required time curve at each gate of the circuit.

In the second step, the heuristic traverses the circuit topologically from the primary inputs to the primary outputs. Let us suppose that we are at a primary input gate G. We are given the arrival time A_x of the primary input signal X. In the first step of the heuristic, we had generated a leakage versus required time trade-off curve at each gate in the circuit. On this curve at the primary input gate G, we choose the final solution point to be the one corresponding to a required time equal to the signal arrival time A_x.

Whenever we chose a solution at any gate in the circuit, this solution imposes a choice of solution at each of the fanout gates. This is true because in the first step, each point generated on the trade-off curve at a gate was corresponding to one particular point on the trade-off curve at each of the fanout gates. As shown in figure 5, let us suppose that we are at gate A and we choose a solution R on the trade-off curve. This solution choice implies a placement of sleep transistor at gate A (either Type I or Type II or no sleep transistor) and its size. Additionally, this solution point corresponds to a solution point at each of the fanout gates P and Q. Hence, choosing a solution R at gate A corresponds to solutions X and Y at gates P and Q respectively.

In order to choose a solution at any gate A in the circuit as shown in figure 6, we look at the fan-in gates of gate A namely gates P and Q. Since the heuristic traverses in topological ordering, we have already decided the sleep transistor placement and sizing solution at these gates which is denoted in figure 6 by points X and Y on curves P and Q respectively. Each of

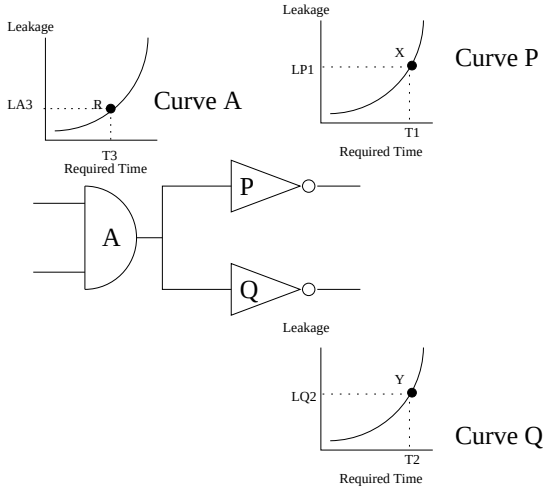

Fig. 5. Choosing a Solution

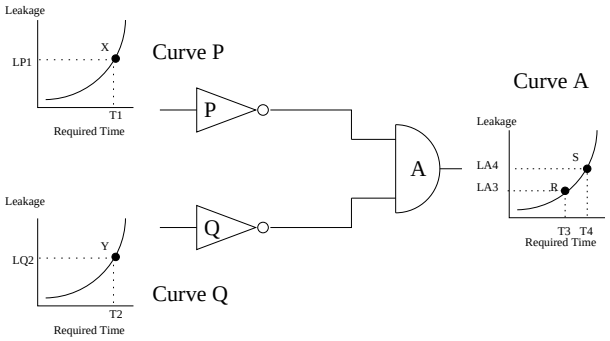

Fig. 6. Building the Solution

these solution points correspond to some point on the leakage versus required time curve at gate A (as explained earlier in this section). Let the points R (with required time $T3$) and S (with required time $T4$) be the points that are corresponding to solution points X and Y respectively. The final solution chosen at gate A must have a required time T that is atleast equal to the maximum of $T3$ and $T4$ in order to satisfy the timing constraints.

Algorithm 1: Heuristic for Selective Placement and Sizing

INPUT: *Circuit, PI arrival time values, PO required time values*
First Step
For each gate A (in topological order from PO to PI)
{
Merge the leakage vs required time curves at its fanout gates (to generate curve X)
Generate curve A at the input of gate A by considering all possible placements and sizing of sleep transistor
}
Second Step
{
Based on the solutions chosen at the fanin gates of gate A, calculate the arrival time constraint
Based on the solutions chosen at the fanin gates of gate A, determine if there is a sleep transistor type constraint
Choose a minimum leakage solution on curve A that satisfies both the arrival time and the sleep transistor type constraint
}
Post-Processing Step
Use the sleep transistor placement information from above to run optimal sizing algorithm

We know that an MTCMOS gate with a type I sleep transistor cannot drive a CMOS gate (without a sleep transistor). If the chosen solution at any of the fanin gates (solution X and Y for gate P and Q respectively as shown in the figure) is such that it places a sleep transistor of type I at the gate, an additional sleep transistor type constraint is imposed on the solution chosen at gate A. Since gate A is being driven by a gate that has a sleep transistor of type I, we cannot choose a solution that implies a CMOS gate at gate A. Therefore, at gate A the final solution chosen must have the *minimum* required time T such that the sleep transistor type constraint (if it exists) is satisfied and the timing constraints (given by equation 24) are met.

$$Required\ \ Time\ \ T \geq max(T3, T4) \tag{24}$$

Due to the above mentioned reasons, the final solution chosen at gate A may have a higher required time such that some of the inputs get some timing slack. This timing slack can later be used to resize the sleep transistor for further leakage reduction at that fanin gate. This completes the second step of the heuristic.

At the end of the two step heuristic, we have a placement of (sized) sleep transistors at the selected gates in the circuit. We resize these sleep transistors using the optimal sizing formulation as a post-processing step to further reduce the leakage power by utilizing the available slack. The overall heuristic can be given as in Algorithm 1.

V. Managing Area Penalty in Standard-Cell Placement

In this section we present a standard cell placement driven fine-grained sizing methodology that ensures that the area penalty incurred is comparable to existing clustering based schemes [11]. In [12], the authors propose a sleep transistor sizing methodology considering them to be distributed in rows in a fully placed circuit. They adopt a clustering based sleep transistor insertion methodology implementing them as special-purpose leakage control cells.

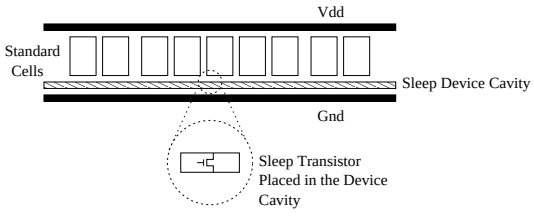

Fig. 7. Existing Scheme

Let us first understand that area penalty imposed by existing clustering-based schemes. Most of the MTCMOS schemes [3], [10], [8], [6], [7], [2] either do not explicitly analyze their area penalty or are based on custom designs. In [11], the authors present a clustering based MTCMOS insertion and sizing approach that utilizes a standard-cell placement design methodology. For every row of standard-cells in the design,

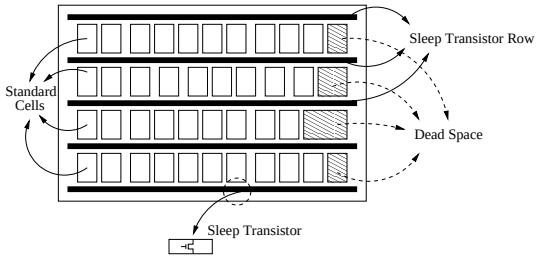

Fig. 8. Proposed Design Methodology

their scheme inserts a row of sleep transistors (which they call sleep device cavity) as shown in figure 7.

Our proposed scheme is similar in essence to that in [11] in terms of sleep transistor row insertion into the design. Let us consider a standard cell placement as shown in figure 8. We will consider inserting rows of sleep transistors of fixed height between every two standard cell rows as shown in figure 8. Hence, for an n row standard cell placement design, the total area penalty would correspond to that of $n+1$ sleep transistor rows of fixed height. Since our sleep transistor methodology places both $PMOS$ and $NMOS$ transistors, we would need a row of $PMOS$ and $NMOS$ transistors for each standard cell row. We counter this overhead by aligning our standard cell rows such that adjacent rows have $PMOS$ and $NMOS$ transistors on the same side. Hence our standard cell rows could be flipped to align their transistors in two possible orderings (PNNPPNN.. or NPPNNPP...). This will allow us to share rows of sleep transistors between two adjacent standard cell rows and keep our area penalty to only $n+1$ rows.

In our area-constrained approach, we first generate a solution for the placement of sleep transistors at the gates in the design under a delay penalty constraint (without any area constraint) using our placement and sizing heuristic from the previous section. Given the standard-cell placement of the circuit, the sleep transistors are inserted in the sleep transistor rows. We will now perform sizing of the sleep transistors such the total area does not exceed the available area in that row. This is done by modifying the optimal sizing formulation presented in section III by adding the area constraints. Each row of sleep transistors is shared between two standard cell rows. We also assign the same size to both the $PMOS$ and $NMOS$ sleep transistor at each gate. The additional area constraints can be generated as follows: For each gate, a particular value of gate delay allocation d^i corresponds to a specific value of leakage I^i_{leak} (through equation 6) and a sleep transistor size W^i/L^i (through equations 2 and 4). We will now utilize these relationship between gate leakage, delay and sleep transistor size to formulate our area-constrained sleep transistor sizing formulation. Combining equations 5 and 6, we can show that there is a convex relationship between the size of the sleep transistor (W^i/L^i) and its delay d^i. This is given by equation 25:

$$W^i/L^i = \frac{1}{K_1 - K_2 d^{i^{-1/\alpha}}} \tag{25}$$

where K_1 and K_2 are positive constants. We will assume that the length L of the sleep transistor is fixed to 2λ units (λ = 180nm for 0.18 micron technology). Hence the sizing is done through changing the W of each sleep transistor. If the height of each sleep transistor row is fixed to 2λ, the sizing is performed along the length of the row. The length of each sleep transistor row is fixed by our chip size (say $MaxLength$). Therefore, there is a limit on the maximum sizing that can be done in each row. Now in the formulation we assign a sizing variable W^i for each vertex i in the design. For each pair of adjacent standard cell rows K and $K+1$ that share a common sleep transistor row, we restrict the total size of the sleep transistors in that row to be atmost equal to $MaxLength$. The following constraints can be added to our optimal sizing formulation (section III) to enforce this area constraint:

$$\Sigma_{i\in K} W^i + \Sigma_{i\in(K+1)} W^i \leq MaxLength$$
$$\forall\ adjacent\ standard\ cell\ rows\ K\ and\ (K+1) \tag{26}$$

The convex relationship between transistor size W^i and its delay d^i (as given by equation 25), is an additional constraint in the sizing formulation now. Hence, we can add the constraints from equation 26 and 25 to the formulation for sizing as presented in section III. We want the sizing formulation to have linear constraints with a convex objective function. Hence, we impose the convex constraint from equation 25 as piecewise linear constraints. These additional constraints put an area restriction on each sleep transistor row during sizing. Hence using the placement information, we have ensured that our sizing algorithm does not result in an area explosion.

A. Other Area Considerations

The leakage feedback gate structure (Type II sleep transistors) has an extra area penalty (due to the helper transistors and the extra inverter). We need to account for this area overhead as well. The helper cells can be placed the in corresponding NMOS and PMOS sleep transistor cells (making the effective area of the sleep transistor double of its actual size). Since we already know the type of sleep transistors that have been placed at each gate before running the sizing formulation, we can consider this extra area (due to helper transistors) into the area constraint of equation 26. We also need to consider the inverter in the leakage feedback gate structure. From figure 8, we can see that in every standard-cell placement row, there is some unutilized space shown as dead space in the figure. This dead space can be used to acccomodate the extra inverter.

B. Routing of Control Signals

As shown in figure 9, each row of sleep transistors can be controlled through just one line of sleep signal running across the entire sleep transistor row. This can then be connected to the appropriate control signal line (shown running vertically in the figure). This illustrates that the routing overhead on control signals is not very complex in the proposed scheme.

VI. Experimental Results

The proposed schemes were implemented in SIS [5]. We integrated CPLEX with SIS to perform our optimal sleep tran-

Bench-mark	Initial Leakage	3% Fixed	3% Opt Siz	3% Heur	3% Place	5% Fixed	5% Opt Siz	5% Heur	5% Place	7% Fixed	7% Opt Siz	7% Heur	7% Place
C1908	31432	-	-	2807.2	2807.2	1913.6	960.6	960.6	960.6	1713.9	934.8	934.8	934.8
C432	13507	-	-	934.7	934.7	862.8	429.5	429.5	429.5	772.8	409.4	409.4	409.4
C880	19472	-	-	934.6	934.6	1311.0	651.4	651.4	651.4	1174.3	637.8	637.8	637.8
apex7	14515	-	-	606.1	606.1	930.3	450.1	450.1	450.1	833.2	441.2	441.2	441.2
comp	11184	-	-	466.8	466.8	631.4	316.5	316.5	316.5	565.5	307.9	307.9	307.9
count	8137	-	-	548.5	548.6	520.6	280.6	280.6	280.6	466.3	270.7	270.7	270.7
i5	19566	-	-	506.4	506.4	872.4	416.3	416.3	416.3	781.4	411.2	411.2	411.2
my_adder	10904	-	-	726.4	726.4	708.5	368.4	368.4	368.4	634.6	358.4	358.4	358.4
too_large	18346	-	-	690.7	690.7	1267.7	595.8	595.8	595.8	1135.4	589.1	589.1	589.1
x4	27416	-	-	728.2	728.2	1479.8	682.6	682.6	682.6	1325.3	677.1	677.1	677.1
C1355	27442.2	-	-	5208.9	5208.9	1513.5	1010.9	1010.9	1010.9	1355.6	920.7	920.7	920.7
C3540	63449.1	-	-	2954.3	2954.3	4169.4	1976.3	1976.3	1976.3	3734.3	1945.1	1945.1	1945.1
C5315	104238.7	-	-	2825.1	2825.1	5750.25	2659.6	2659.6	2659.6	5150.3	2640.3	2640.3	2640.3
i6	50751.3	-	-	1137.8	1137.8	2458.2	1137.8	1137.8	1137.8	2201.7	1107.2	1107.2	1107.2
i7	65689.5	-	-	1536.4	1536.4	3345.1	1506.4	1506.4	1506.4	2996.1	1505.9	1505.9	1505.9
i8	66516.7	-	-	2226.7	2226.7	3817.5	1728.1	1728.1	1728.1	3419.2	1724.7	1724.7	1724.7
i9	46577.7	-	-	1448.6	1448.6	2689.6	1224.6	1224.6	1224.6	2408.9	1220.7	1220.7	1220.7
frg2	60083.6	-	-	1745.6	1745.6	3460.8	1570.2	1570.2	1570.2	3099.7	1565.7	1565.7	1565.7
% Imprv. by Opt Sizing over Fixed Penalty Algo.							50.9				46.5		

TABLE I

EXPERIMENTAL RESULTS (LEAKAGE CURRENT IN PA)

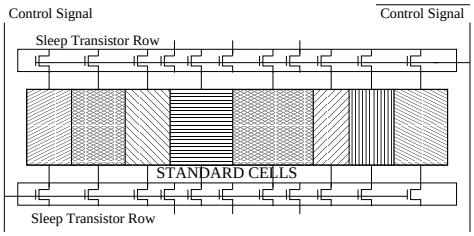

Fig. 9. Control Signal Routing

sistor sizing through the proposed formulation. The standard-cell placement of the benchmarks were generated using an academic placement tool CAPO [1]. We took V_{tH} = 500mV, V_{tL} = 350mV and $I_{sleep_{ON}}$ = 300 μA for all gates. Additionally, we imposed the condition that for each gate the delay penalty by placing a sleep transistor was allowed to vary between 4% and 15% (corresponding to $1 \leq (W/L)_{sleep} \leq 10$). This implies that conventional methodologies [11], [8], which place a sleep transistor at every gate will give a valid solution only if there is an acceptable performance loss $\geq 4\%$.

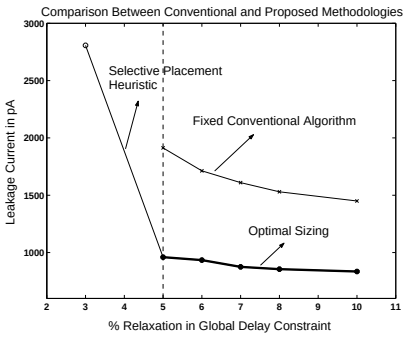

Fig. 10. Benchmark C1908

As shown in table I, for a 3% slowdown in performance, we cannot get a valid solution from the conventional fixed slowdown approach as well as the optimal sizing algorithm (with sleep transistors placed everywhere). On the other hand for all the benchmarks, the selective placement and sizing heuristic as well as the standard cell based placement and sizing methodology are able to significantly reduce leakage while still satisfying the delay constraints. These results prove our claim that for stringent required time constraints, selective placement and sizing of sleep transistors can effectively reduce leakage power.

Table I shows that the leakage values obtained from optimal sizing (assuming sleep transistor placed at each gate), the placement and sizing heuristic and the standard-cell driven placement and sizing formulation are the same for both 5% and 7% slowdown. The proposed heuristic is effective in placing a sleep transistor at all gates and sizes them to reach the optimal sizing solution. Furthermore, the standard-cell placement driven (area-constrained) scheme is able to utilize the available area to size the sleep transistors implying that delay constraint is more limiting than the area constraint. From figure 10, we can see that our results are much better than the conventional algorithms. The last row in table I shows that on an average the Optimal Sizing Algorithm gave 50.9% and 46.5% savings in leakage power as compared with the conventional *fixed* delay penalty algorithms for 5 and 7% slowdown respectively.

Table II shows the results obtained by running the selective placement placement heuristic for a 3% performance slowdown. Under these constraints, the conventional methodology of placing a sleep transistor at every gate would not yield a valid solution since we need a minimum 4% slowdown. From these results we can see that the selective placement and sizing methodology for sleep transistor insertion for stringent delay constraints is effective in providing significant leakage savings which is not possible using the conventional methodologies.

Lastly, we evaluate the area penalty imposed by our proposed fine-grained sleep transistor placement methodology. Our results have shown that the proposed scheme is able to effectively size the sleep transistors without imposing any additional area penalty as compared with the clustered

Bench-mark	Init Leak (10^{-12}A)	Final Leak (10^{-12}A)	Total Gates	Type I Sleep	Type II Sleep
C1908	31432	2807	349	306	18
C432	13507	935	147	102	27
C880	19472	934	225	216	2
apex7	14515	606	157	155	0
comp	11184	466	79	74	0
count	8137	548	78	62	7
i5	19566	506	244	236	2
my_adder	10904	726	128	104	11
too_large	18346	690	456	442	4
x4	27416	728	315	306	3
C1355	27442	5208	314	190	54
C3540	63449	2954	865	813	21
C5315	104238	2825	1193	1186	2
i6	50751	1137	510	506	0
i7	65689	1536	694	690	0
i8	66516	2226	792	783	4
i9	46577	1448	558	536	14
frg2	60083	1745	718	713	3

TABLE II
RESULTS FOR 3 PERCENT SLOWDOWN

approach [11]. If we have n standard cell rows each of height 20λ spaced 20λ apart, putting in $n+1$ sleep transistor rows of height 2λ each corresponds to an area penalty of roughly $(2\lambda(n+1))/(40\lambda n) = 5\%$. From our results, it can be seen that the area is each sleep transistor row is more than sufficient for placing and sizing sleep transistors.

The runtime complexity for the proposed sizing and selective placement schemes is not very high. For very large designs, we can apply these schemes at the block level (or other partioning techniques can be used). The number of variables introduced per gate in the formulation is very low and hence the formulation does not explode with design size.

A. Comparison with Clustering Based MTCMOS Scheme [11]

We implemented a clustering strategy similar to that proposed in [11]. In order to make a fair comparison with our proposed fine-grained scheme, gates within each standard-cell placement row are clustered together based on their current discharge patterns. The authors in [11] reported in their results that if they increased the routing overhead cost to be dominant in the cost function, even their scheme clustered gates within the same sleep transistor row and formed larger number of sleep transistor clusters. We ran experiments assuming a 5% slowdown for each benchmark. The results are given in table III.

The performance of the fine-grained scheme is comparable to the clustering scheme both in terms of leakage and actual area penalty. Fine-grained sizing enables our scheme to significantly reduce the leakage even though we place more transistors compared to the clustering approach. Additionally, in this analysis we have ignored the effects of ground bounce and signal noise for the clustering scheme which if considered would worsen the clustering-based results.

In table III we have reported W/L numbers for the sleep transistors for the sake of direct comparison. The true area penalty imposed by both clustering and fine-grained schemes is equal to the area of the sleep transistor rows which is the same for both the schemes.

VII. CONCLUSION AND FUTURE WORK

In this work we have proposed a novel slack based approach for fine-grained sleep transistor placement and sizing. The proposed scheme has better control over noise immunity and signal integrity compared to existing clustering based schemes. The polynomial time optimal sizing formulation that we have proposed can be applied to get additional savings in leakage. We have shown through our experiments that selective placement and sizing of sleep transistors under very stringent delay constraints can lead to substantial leakage power reduction. Our standard-cell placement driven formulation for optimal sizing also gives the same leakage savings as the selective placement and sizing heuristic under an area penalty constraint which is comparable to the clustered schemes.

An interesting direction for future work is to investigate the possibility of sharing a sleep transistor row across multiple standard-cell placement rows considering the routing issues.

REFERENCES

[1] A. Caldwell, et al. "Can Recursive Bisection Alone Produce Routable Placements?". In *Proc. of DAC*, 2000.

[2] B. Calhoun, F. Honore and A. Chandrakasan. "Design Methodology for Fine-Grained Leakage Control in MTCMOS". In *Procs of ISLPED*, 2003.

[3] C. Long and L. He. "Distributed Sleep Transistor Network for Power Reduction". In *Procs of Design Automation Conference*, June 2003.

[4] D. Hochbaum and J. Shanthikumar. "Convex Separable Optimization is not much harder than Linear Optimization". In *Journal of the ACM*, vol. 37, No. 4, 1974.

[5] E.M. Sentovich, K.J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P.R. Stephan, R.K. Brayton, A.L. Sangiovanni-Vincentelli. *SIS: A System for Sequential Circuit Synthesis*. Memorandum No. UCB/ERL M92/41, Department of EECS. UC Berkeley, May 1992.

[6] J. Kao, A. Chandrakasan and D. Antoniadis. "Transistor Sizing Issues and Tool For Multi-Threshold CMOS Technology". In *Procs of Design Automation Conference*, June 1997.

[7] J. Kao and A. Chandrakasan. "MTCMOS Sequential Circuits". In *Procs of ESSDERC*, Sept 2003.

[8] J. Kao, S. Narendra and A. Chandrakasan. "MTCMOS Heirarchical Sizing Based on Mutual Exclusive Discharge Patterns". In *Procs of Design Automation Conference*, June 1998.

[9] K. Keutzer, et al. "Minimization of Dynamic and Static Power Through Joint Assignment of Threshold Voltages and Sizing Optimization". In *Proc. of ISLPED*, 2003.

[10] M. Stan. "Low Threshold CMOS circuits with Low Standby Current". In *Proc. of ISLPED*, 1998.

[11] Mohab Anis, et al. "Design and Optimization of Multithreshold CMOS (MTCMOS) Circuits". In *IEEE Transactions on CAD of Integrated Circuits and Systems*, October 2003.

[12] P. Babighian, L. Benini and E. Macii. "Sizing and Characterization of Leakage Control Cells for Layout-Aware Distributed Power-Gating". In *Procs of DATE*, 2004.

[13] S. Mukhopadhyay and K. Roy. "Modeling and Estimation of Total Leakage Current in Nano-scaled CMOS Devices Considering the Effect of Parameter Variation". In *Procs of ISLPED 2003*, Aug. 2003.

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

Bench-mark	Initial Leakage (10^{-12}A)	Fine-Grained Leakage (10^{-12}A)	Fine-Grained Area Σ W/L	Clustering Leakage (10^{-12}A)	Clustering Area Σ W/L	Number of Clusters
C1908	31432	960	542.7	775	438	73
C432	13507	429	242.6	403	228	38
C880	19472	651	368.0	637	360	60
apex7	14515	450	254.3	562	318	53
comp	11184	316	178.8	392	222	37
count	8137	280	158.5	286	162	27
i5	19566	416	235.2	902	510	85
my_adder	10904	368	208.1	392	222	37
too_large	18346	595	336.6	690	390	65
x4	27416	682	385.7	934	528	88
C1355	27442	1010	571.2	775.2	438	73
C3540	63449	1976	1115.7	1242.5	702	117
C5315	104238	2659	1502.6	2272.7	1284	214
i6	50751	1137	625.9	1497.4	846	141
i7	65689	1506	851.2	1529.3	864	144
i8	66516	1728	976.4	1433.7	810	135
i9	46577	1224	691.8	1189.5	672	112
frg2	60083	1570	887.1	1667.4	942	157

TABLE III

COMPARISON WITH CLUSTERING