Dynamic and Leakage Power Reduction in MTCMOS Circuits Using an Automated Efficient Gate Clustering Technique

Mohab Anis, Shawki Areibi*, Mohamed Mahmoud and Mohamed Elmasry

ECE Department, University of Waterloo, Waterloo, Ontario, Canada, N2L 3G1
email: manis,mohamedm,elmasry@vlsi.uwaterloo.ca
* School of Engineering, University of Guelph, Guelph, Ontario, Canada, N1G 2W1
email: sareibi@uoguelph.ca

ABSTRACT

Reducing power dissipation is one of the most principle subjects in VLSI design today. Scaling causes subthreshold leakage currents to become a large component of total power dissipation. This paper presents two techniques for efficient gate clustering in MTCMOS circuits by modeling the problem via Bin-Packing (BP) and Set-Partitioning (SP) techniques. An automated solution is presented, and both techniques are applied to six benchmarks to verify functionality. Both methodologies offer significant reduction in both dynamic and leakage power over previous techniques during the active and standby modes respectively. Furthermore, the SP technique takes the circuit's routing complexity into consideration which is critical for Deep Sub-Micron (DSM) implementations. Sufficient performance is achieved, while significantly reducing the overall sleep transistors' area. Results obtained indicate that our proposed techniques can achieve on average 90% savings for leakage power and 15% savings for dynamic power.

Categories & Subject Descriptors: B.7.1 [Integrated Circuits]: Types and Design Styles
General Terms: Design

1. INTRODUCTION

With the advent of technology, the reduction of the supply voltage V_{dd} has become vital to reduce dynamic power and to avoid reliability problems in Deep Sub-Micron (DSM) regimes. However, reducing V_{dd} alone causes serious degradation in the circuit's performance. One way to maintain performance is to scale down both V_{dd} and the threshold voltage V_{th}. However, reducing V_{th} increases the subthreshold leakage current exponentially. This problem escalates in DSM technologies. Multi-threshold CMOS (MTCMOS) technology has emerged as an increasingly popular technique to reduce leakage power during the standby mode, while attaining high speed in the active mode. Devices switching in the critical path are assigned low threshold voltage (LVT) while the others are high threshold voltage (HVT) to reduce leakage power [1],[2]. This technique requires accurate assignment of LVT and HVT, but has the advantage of preserved speed. Another way to implement MTCMOS technology is inserting a HVT device; called sleep transistor, in series to the normal LVT circuitry as shown in Figure 1(a) [3]. The sleep transistor is controlled by a controllable $SLEEP$ signal used for active/standby mode control ($SLEEP$=1,0 during standby and active modes respectively). Proper sleep transistor sizing is a key issue that affects the performance as well as the dynamic and leakage powers of the entire circuit. The design cycle is usually short, but at the expense of a slight speed loss.

In this paper we introduce two techniques that cluster logic gates at a fixed sleep transistor size, which will prove to be power efficient compared to the literature ([3] and [4]) while maintaining adequate performance.

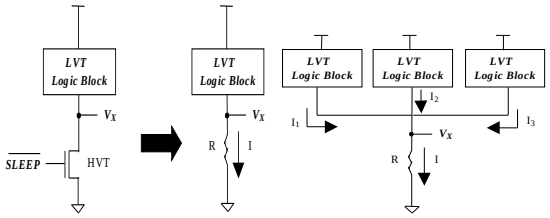
2. BACKGROUND

During the active mode, the sleep transistor could be realized as a resistor **R** as shown in Figure 1(a) [5]. This generates a small voltage drop V_X equal to $I \times R$, where I is the current flowing through the sleep transistor. The voltage drop across R, reduces the gate's driving capability from V_{dd} to V_{dd}-V_X which in turn degrades the gate's performance. Therefore, the resistor should be made small and consequently the size of the sleep transistor large which comes at the expence of area and power overhead. On the other hand, if the resistor is made large meaning that the sleep transistor is sized small, the circuit speed will degrade. This trade-off between achieving sufficient performance and low power values will become even more severe in the DSM regime. In DSM technologies, the supply voltage is scaled down aggressively, causing the resistance of the sleep transistor to increase dramatically, requiring even larger size sleep devices. This will cause leakage and dynamic power to significantly mountain in the standby and active modes respectively. Therefore, an important design criterion is sizing the sleep transistor to attain sufficient performance. In other words, the current "I" flowing through the sleep transistor must be satisfactory to achieve the required speed.

The worst case design scenario takes place if all the gates supported by the sleep transistor are simultaneously switching in time (Figure 1(b)). The sleep transistor exhibits maximum current then (I=I_1+I_2+I_3)(Case I). The sleep transistor is thus sized up to contain the high current. If the gates are discharging mutually exclusive, the sleep transistor is sized according to the maximum current

DAC 2002, June 10-14, 2002, New Orleans, Louisiana, USA.

of the mutually exclusive discharging gates ($I = \max\{I_1, I_2, I_3\}$) (Case II). The sleep transistor is a lot smaller in this case. If a current-time graph is constructed of the discharged currents, I_1, I_2 and I_3 would overlap in time in Case I. On the other hand, no overlap in time occurs for Case II. An intermediate case occurs when the discharged currents "partially" overlap, if the LVT Logic Blocks have slightly different discharge times.



(a) Sleep Transistor modeled as resistor

(b) Worst case discharging scenario

Figure 1: Sleep Transistor in MTCMOS Circuits

A single sleep transistor to support the whole circuit was proposed in [3]. In another work [4], the sleep transistor was sized according to an algorithm based on mutual exclusive discharge pattern. In [4], cascaded gates are clustered together because simultaneous current discharge can never take place. This methodology may be efficient for balanced circuits with tree configurations, where mutually exclusive discharging gates are easily detected. However, this methodology would not be efficient for circuits with complicated interconnections and unbalanced structures. Sleep transistor assignments can therefore be wasteful, and would cause dynamic and leakage power to rise. Finally, the sets of sleep transistors in [4] are merged into a single large sleep transistor to accommodate the whole circuit as in [3]. In addition to the drawbacks listed above, sharing a single sleep transistor for the whole circuit would increase the interconnect resistance for distant blocks. As a result, the sleep transistor would be sized even larger than expected to compensate for the added interconnect resistance. Excessively large sleep transistors again augment dynamic and leakage power as well as area. This drawback would be even more severe in DSM regimes, where interconnects would have a large impact on the circuit's performance [6]. Our proposed methodology in Section 4 solves this problem, and not only clusters gates with exclusive discharge patterns, but with "partially" overlapping discharged currents as well. The first step in our technique is to calculate the size of the sleep transistor.

3. SIZING THE SLEEP TRANSISTOR

To estimate the size of the sleep transistor, the delay of a single gate (τ_d) at the absence of a sleep transistor can be expressed as

$$\tau_d = \frac{C_L V_{dd}}{(V_{dd} - V_{tL})^\alpha} \tag{1}$$

where C_L is the load capacitance at the gate's output, V_{tL} is the LVT=350mV, V_{dd}=1.8V and α is the velocity saturation index which is equal to $\approx$ 1.3 in 0.18μm CMOS technology. In the presence of a sleep transistor, the delay of a single gate τ_d^{sleep} can be expressed as

$$\tau_d{}^{sleep} = \frac{C_L V_{dd}}{(V_{dd} - V_X - V_{tL})^\alpha} \tag{2}$$

where V_X is the potential of the virtual ground. Assuming the circuit could tolerate a 5% degradation in performance due to the presence of the sleep transistor, therefore

$$\frac{\tau_d}{\tau_d^{sleep}} = 95\% \tag{3}$$

Substituting for τ_d and τ_d^{sleep}, and assuming $\alpha = 1$ for simplicity, we get

$$1 - \frac{V_X}{(V_{dd} - V_{tL})} = 95\% \tag{4}$$

Therefore V_X can be formulated as

$$V_X = 0.05(V_{dd} - V_{tL}) \tag{5}$$

The current flowing through the "linearly-operating" sleep transistor is expressed as:

$$\begin{aligned} I_{sleep} &= \mu_n C_{ox}(W/L)_{sleep}[(V_{dd} - V_{tH})V_X - {V_X}^2/2] \\ &\approx 0.05\mu_n C_{ox}(W/L)_{sleep}(V_{dd} - V_{tL})(V_{dd} - V_{tH}) \end{aligned} \tag{6}$$

where μ_n is the N-mobility, C_{ox} is the oxide capacitance and V_{tH} is the HVT=500mV. The size of the sleep transistor can be therefore expressed as

$$(W/L)_{sleep} = \frac{I_{sleep}}{0.05\mu_n C_{ox}(V_{dd} - V_{tL})(V_{dd} - V_{tH})} \tag{7}$$

I_{sleep} and consequently $(W/L)_{sleep}$ are chosen to exhibit low power dissipation. I_{sleep} is chosen to be 250μA, leading to a $(W/L)_{sleep}$ $\approx$ 6 for 0.18μm CMOS technology. This constant size $(W/L)_{sleep}$ = 6 will be used for both proposed methodologies i.e Bin-Packing (BP) and Set-Partitioning (SP) techniques. Agreeable delay, power and leakage values to analytical calculations were verified for the LVT HSPICE models, to ensure correct functionality. Leakage current increases by an order of magnitude for every 85mV reduction in V_{th}.

4. PROPOSED CLUSTERING TECHNIQUE

To illustrate our techniques, six benchmarks are used as test vehicles; a 4-bit Carry Look Ahead (CLA) adder, a 32-bit priority checker, a 6-bit array multiplier design, a 4-bit ALU/Function Generator (74181 ISCAS-85 benchmark), a 32-Single Error Correcting circuit (C499 ISCAS-85 benchmark) and finally a 27-bit Channel Interrupt Controller (CIC) (C432 ISCAS-85 benchmark). These benchmarks have been chosen to offer a variety of circuits with different structures employing various gates, with different fanouts. The 4-bit CLA adder will be first used to demonstrate our techniques, then the results to the other benchmarks will be provided later on.

Figure 2 shows a schematic diagram of the CLA adder, which consists of 28 gates (G_1-G_{28}). All gates are implemented in 0.18μm CMOS technology. In the next section a preprocessing stage of gate currents is described. This stage will be utilized in solving the BP problem.

4.1 Preprocessing of Gate Currents

The main objective of the preprocessing stage is to group gates into subclusters such that the combination would not exceed the max current of any gate within the cluster. Randomly chosen input vectors are applied and the highest discharging current at the output of every gate is monitored (worst case). The discharge current is only monitored because this is the current that flows through the sleep transistor and eventually $ground$. A load of 6fF is applied to the outputs of each circuit. The probability that discharging takes place (switching activity) at the output of each gate is

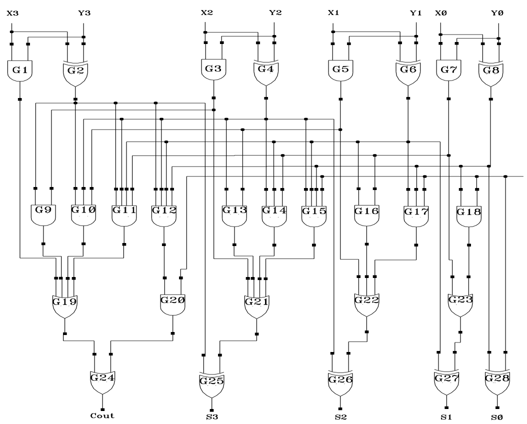


Figure 2: 4-bit Carry Look-Ahead Adder

calculated and multiplied by the corresponding discharge peak current. This current is composed of the discharge and the short-circuit currents that take place during switching. Sleep transistors should be sized to also accommodate the short-circuit currents, otherwise speed will degrade. The peak current value and time at which the switching occurs as well as its duration are monitored. The time the switching takes place depends on the gate's propagation delay and input pattern, while the current duration depends on the slope of the input signal as well as the fanout of the gate. The larger the input slope and/or gate fanout, the longer the switching duration. The discharge current of each gate takes a triangular shape, whose peak occurs at a time equal to the gate delay, and spans a time, mainly function in the fanout of the gate.

To facilitate vector comparisons and to offer an automated design environment, every discharge current at the output of a gate is represented by a vector. The time axis is divided into time slots each equal to 10psec as shown in Figure 3. A time slot of 10psec is sufficient in 0.18μm CMOS technology to offer relatively good accuracies. Each time slot holds a value that represents the magnitude of the discharge current at that specific time which constitutes an element in the vector. In order to illustrate this idea, Figure 3 shows a 2-input AND gate (G1) with a fanout of 2 driving a 2-input OR gate (G2) with a fanout of 4. The disharge currents of G1 and G2 (I_1 and I_2) are presented as a vector. Each element

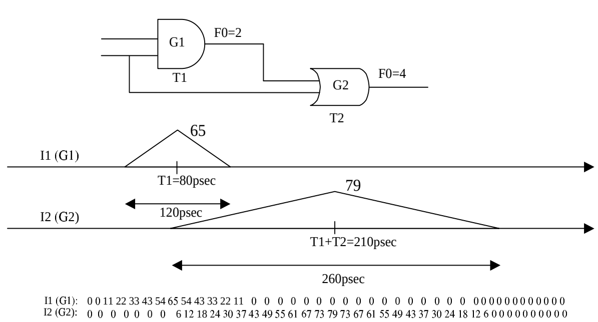


Figure 3: Timing Diagram

in the vector presents the magnitude of current at this 10psec time slot. The peak of the discharge current for G1 occurs at the gate's delay time ($T_1 = 80psec$), while the discharge current I_2 occurs at time ($T_1 + T_2 = 210psec$), because G2 will not discharge till G1 discharges. The peak currents of gates G1 and G2 are $65\mu A$ and $79\mu A$ respectively. The triangular shaped currents are converted into vectors as seen in Figure 3. Since G2 has a large fanout of 4, the duration of the discharge current is long (260 psec), while the duration of the discharge current in G1 is short due to the small fanout of 2 (120 psec). Therefore, for every gate in the circuit, a vector is constructed that carries information about the delay of the gate (when the peak occurs), the fanout of the gate (the duration at which the current lasts) and the magnitude of the current in each time slot. By constructing a vector for each gate, a series of vectors (28 in this case) are produced, that carry information about the whole circuit.

Figure 4 illustrates the used preprocessing heuristic that forms a set of subclusters of gates that when combined would not exceed the maximum current of any gate within the cluster. Table 1 shows

```
PREPROCESSING HEURISTIC
1. Initialize current vectors
2. Set all Gates free; to move to subcluster;
3. For all gates in circuit
        If gate i is not clustered yet
          assign gate i to new cluster k
          update cluster current vector
          calculate max current,start,end time
        For all other gates in circuit
          If (gate j is not clustered yet)
            add current of gate j to cluster k
            If (combination ≤ max current)
              append gate to cluster
              update cluster info
              set gate j locked in cluster k
        End For
   End For
4. Return all clusters formed.
```

Figure 4: Heuristic for Preprocessing

the results of applying the preprocessing heuristic to the 4-bit carry look-ahead adder that was presented in Figure 2. In this example seven subclusters were formed. For example the third column in Table 1 (I_{EQ_3}) represents a subcluster formed by combining Gates G5,G6 and G14 which has a maximum current of 110 of the partially overlapped discharging gates ($I_{G5}^{max} = 83$, $I_{G6}^{max} = 110$ and $I_{G14}^{max} = 30$). $I_{overlap}^3 = max\{I_{G5}^{max}, I_{G6}^{max}, I_{G14}^{max}\}$. The objective is then to group as much current (gates) as possible without exceeding the current limit of the sleep transistor (250μA), while minimizing the number of sleep transistors used as will be shown in Table 2 of Section 4.2. This is analogous to the Bin-Packing problem in operations research.

Table 1: Results: Current Equivalence

I_{EQ_1} $I_{overlap} = 80$	I_{EQ_2} 80	I_{EQ_3} 110	I_{EQ_4} 90	I_{EQ_5} 50	I_{EQ_6} 30	I_{EQ_7} 50
I_1, I_2, I_{10}, I_{11}, I_{12}, I_{19}, I_{20}, I_{21}, I_{22}, I_{24}, I_{25}, I_{26}, I_{27}	I_3, I_4, I_{13}, I_{15}, I_{28}	I_5, I_6, I_{14}	I_7, I_8, I_{16}, I_{23}	I_9	I_{17}	I_{18}

4.2 The Bin-Packing Technique

The Bin-Packing (BP) problem [7] can be described as follows. Given $n\ items$ ($currents$ in this case) and $m\ bins$ ($sleep\ transistors$ in this case), with

I_{EQ_j} = equivalent $current$ of gate j,
I_{max} = $capacity$ of each $sleep\ transistor$ = 250μA

The objective is to assign each I_{EQ} to one bin so that the total current in each bin does not exceed I_{max} and the number of bins used is minimized.

The mathematical formulation of the problem is as follows

$$Minimize\ z = \sum_{i=1}^{m} y_i \tag{8}$$

subject to

$$\begin{aligned} &\sum_{j=1}^{n} I_{EQ_j} x_{ij} \leq I_{max} y_i, \quad i \in \{1, ..., m\}, \\ &\sum_{i=1}^{m} x_{ij} = 1, \end{aligned} \tag{9}$$

where

$$y_i = \begin{cases} 1, & \text{if bin } i \text{ is used} \\ 0, & \text{otherwise} \end{cases} \quad x_{ij} = \begin{cases} 1, & \text{if items } j \in \text{bin } i; \\ 0, & \text{otherwise} \end{cases}$$

This model is a pure Integer Linear Programming problem (ILP). The objective function to be minimized; z, is analogous to the minimum number of sleep transistors used. y_i is analogous to the sleep transistors available. x_{ij} takes a value of "1" if current I_{EQ_j} is assigned to bin i. CPLEX 6.5; a commercial ILP solver, was used to solve this BP problem, to determine which currents should be grouped together, and to which sleep transistor they are assigned. A summary of the current assignments is shown in Table 2.

Table 2: Results: Current Assignments

Sleep Transistor (Cluster)	1	2
Equivalent Currents	$I_{EQ_3}, I_{EQ_4}, I_{EQ_7}$	$I_{EQ_1}, I_{EQ_2}, I_{EQ_5}, I_{EQ_6}$
Assigned Gates	G_5, G_6, G_7, G_8, $G_{14}, G_{16}, G_{18}, G_{23}$	$G_1, G_2, G_3, G_4, G_9, G_{10}, G_{11}, G_{12}, G_{13}, G_{15}, G_{17}, G_{19}, G_{20}, G_{21}, G_{22}, G_{24}, G_{25}, G_{26}, G_{27}, G_{28}$
$\sum$ Currents(μA)	250	240

It is clear from Table 2 that two sleep transistors will be needed to contain all the gates in the circuit ($z = 2$). It should be noted that the total current of any cluster must never exceed the maximum current limit of the sleep transistor, which is 250μA.

The BP technique was further applied to the other five benchmarks. Keeping the 5% speed degradation as a comparison basis (operational frequency 500MHz) , the BP technique is compared to [3] and [4]. The results are mentioned in Section 5 and summarized in Table 3 (Normalized to [3]).

The BP technique is particularly efficient when it is applied to small circuits that have unbalanced structures. One limitation is that the BP technique does not take the physical locations of the gates on the chip into consideration. For larger circuits this might cause two gates located far apart to be clustered together which will augment the routing complexity of the circuit, as discussed earlier. The Set-Partitioning technique solves this problem, and consequently reduces the routing complexity of the circuit unlike [3] and [4].

4.3 The Set-Partitioning Technique

The Set-Partitioning (SP) problem [7] can be described as follows: Similar to the BP problem, m currents (gates) are arranged into groups such that each element is included only once in a cluster. A cost function; c_j is associated with each group j (S_j). The cost function c_j is evaluated from the physical locations of the gates with respect to each other, which is related to the routing complexity of the circuit.

In order to evaluate the physical locations of the gates, the Cadence Virtuoso Placement and Route tool has been used to produce a compact layout from the schematic entry. Once the compact layout is constructed, the X,Y coordinates for every gate are extracted and the cost functions are evaluated. Figure 5 shows the floor-plan layout for the 4-bit CLA adder. The V_{dd} and gnd rails are shown and a cavity exists where the sleep transistors are located.The cavity of the sleep transistors has been taken into consideration when extracting the X,Y coordinates of every gate.

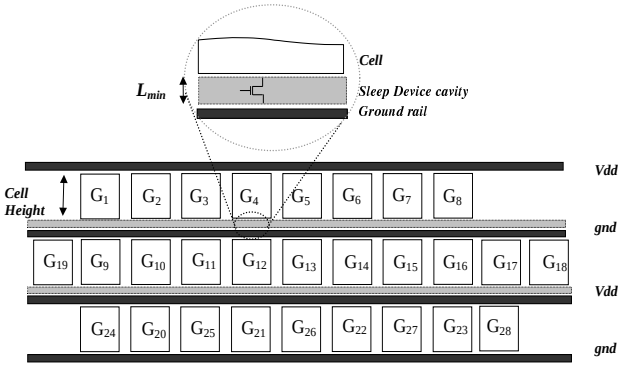


Figure 5: 4-bit CLA Adder Floorplan

In Figure 5 gates G1 to G28 are identified, and the relative distances are computed from the compact layout.

The cost function is formulated as follows:

$$c_j = (w_1 \times c_{j1}) + (w_2 \times c_{j2}) \tag{10}$$

where c_{j1} represents the difference between the maximum cluster capacity and the sum of all currents of gates within a cluster and c_{j2} is a distance function (i.e rectilinear distance between gates within a cluster).

$$c_{j1} = Sleep\_Transistor_{max\_current} - \sum current_i \quad \forall i \tag{11}$$

The weights w_1 and w_2 are the weights associated with the cost of the two constraints i.e distance and capacity of the formed clusters. In this paper we have assigned equal values to the weights w_1 and w_2 respectively. This will allow the set partitioning modeling of the problem to favor clusters with full capacity (more gates within a cluster) and minimum distance as will be explained later on.

$$c_{j2} = \sum d_{uv} \text{ in a group } S_j \tag{12}$$

where d_{uv} is the distance between the centers of gates G_u and G_v. For example, referring to Figure 6, group S_j is composed of gates G_u,G_v and G_w. The value of the partial cost function of group S_j is: $c_{j2} = d_{uv} + d_{vw} + d_{wu}$

Gates are grouped, while meeting the constraint that the sum of currents does not exceed I_{max}=250μA. Figure 7 presents a very fast and efficient heuristic to form groups of clusters that will be used by the SP technique. The heuristic forms different types of clusters (i.e clusters consisting of single gates, two gates e.t.c). This will guarantee that the set partitioning technique will find a solution for the problem. The target is to select certain groups (clusters) to

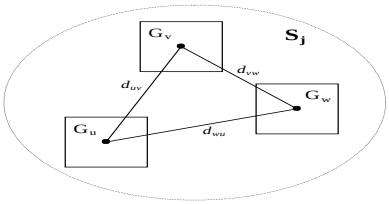


Figure 6: Cost Function Calculation Example

CLUSTERING HEURISTIC

Create_Clusters()
1. Calculate distances between all gates;
2. Initialize maxgates_per_cluster=n;
3. Create clusters with Single gates;
4. **For** cl=2; cl $\leq$ maxgates_per_cluster
   Create_n_Gate_Cluster(cl)
5. For all clusters created **calculate_cost()**

Create_n_Gate_Clusters(cl)
1. **For** cluster of type cl
   create_new_cluster()
   While not done
   Choose Gate with minimum distance
   If sum of currents $\leq$ capacity
   append gate to newly created cluster
   If total gates within cluster $\geq$ limit
   break;
   End
2. return newly created cluster

Figure 7: Heuristic for Grouping Gates

achieve lowest cost value, while maintaining the I_{max} constraint. The groups must also cover all gates with no repetition. The mathematical formulation of the set partitioning problem is as follows

$$Minimize \quad Z = \sum_{j=1}^{n} c_j S_j \tag{13}$$

subject to

$$\sum_{j} S_j = 1 \tag{14}$$

$$S_j = \begin{cases} 1, & \text{if the } j\text{th subset is formed into a group} \\ 0, & \text{otherwise} \end{cases}$$

n is the number of groups generated. The above model is also a 0-1 pure integer LP problem, which is again solved using CPLEX version 6.5. Figure 8 shows the solution of the SP technique by

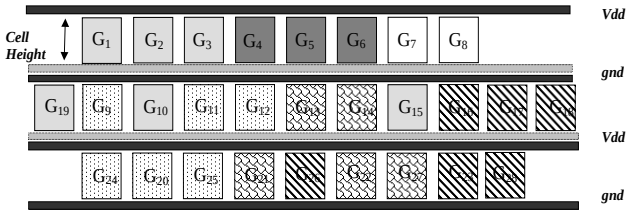


Figure 8: Results:4-bit CLA Adder Floorplan

highlighting the gates that are clustered together. It is evident from Figure 8 that gates that are placed closely were clustered together (i.e gates in two consecutive rows) with a specific sleep transistor therefore minimizing the wire-length. Figure 9 shows the CPU time involved in solving the benchmarks for both the BP and SP problems. It is evident from the figure that solving the SP problem involves more CPU cycles than solving the BP problem. This is due to the fact that the number of variables and constraints in the SP problem are much larger than that of the BP problem. It is important to point-out that as we increase the number of clusters generated for the SP technique the smaller the computation time involved. The BP preprocessing algorithm has a worst case complexity of $O(n^2)$, where n is the number of gates in the circuit. On the other hand, the SP algorithm complexity is $O(nk)$ where n is the number of gates in the circuit and k is the maximum gates to be appended in a cluster. For large circuits it is recommended that heuristic search techniques such as Genetic Algorithms would be used instead of the CPLEX solver.

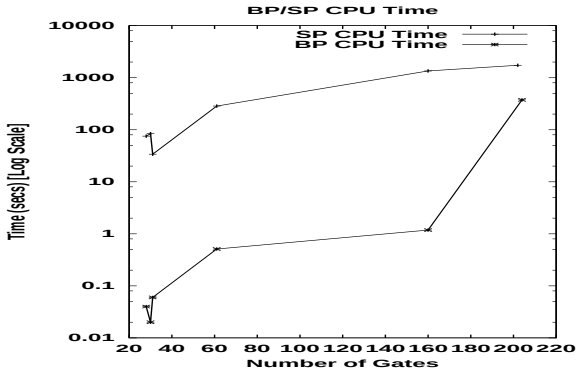


Figure 9: Computation Time for BP and SP

5. RESULTS AND DISCUSSION

Table 3 compares the SP and BP techniques to the literature, while keeping the 5% degradation in speed as a comparison basis. All results are normalized to [3]. The BP & SP techniques employ sleep transistors of equal size calculated as $(\frac{W}{L})_{sleep} = 6$) to achieve only 5% degradation in circuit speed (frequency of operation is set at 500MHz). L_{min} for 0.18μm CMOS technology = 180nm leading to W_{sleep} = 1.1 μm. LVT=350mV and HVT=500mV.

From Table 3,[4] employs a smaller sized single sleep transistor containing the whole circuit compared to [3]. Consequently, a slight reduction in dynamic power is observed due to the reduction of the drain capacitance linked to the sleep transistor. [4] achieves an average of 50% reduction in leakage power compared to [3].

The highest leakage reduction occurs in the 27-bit CIC benchmark. This is due to the large reduction in sleep transistor area (3247 to 153). On the other hand, the BP technique produces large reductions in the sleep transistors total area. Although the number of sleep transistors is higher than [3] and [4], the size of every sleep transistor is much smaller achieving an overall reduction in sleep transistor area. Therefore, the BP technique offers significant dynamic power savings compared to [3] and [4] as shown in Table 3. On average the BP technique achieves 17% reduction over [3] and 14% dynamic power reduction over [4]. The main saving however is associated with the leakage power, due to the reduction of the sleep transistor size, which is directly proportional to the leakage power dissipation. On average the BP technique achieves 95% and 86% leakage power reduction compared to [3] and [4].

The SP technique is then compared to the BP technique, [3] and [4], while still keeping the 5% speed degradation as a compari-

Table 3: Algorithm Comparison

REF	Benchmark	4-bit CLA Adder	32-bit Parity Checker	6-bit Multiplier	4-bit 74181 ALU	32-bit Single Error Correcting C499	27-channel interrupt controller circuit C432
	No. of gates	28	31	30	61	202	160
[3]	Delay	1	1	1	1	1	1
	$P_{dynamic}$	1	1	1	1	1	1
	$P_{leakage}$	1	1	1	1	1	1
	# Sleep Trans	1	1	1	1	1	1
	ST_Area [$W_{sleep}(\mu$m)]	50	42	65	97	176	3247
[4]	Delay	1	1	1	1	1	1
	$P_{dynamic}$	0.98	0.97	0.89	0.97	0.99	0.98
	$P_{leakage}$	0.58	0.51	0.23	0.41	0.46	0.05
	# Sleep Trans	11→1	16→1	5→1	37→1	32→1	52→1
	ST_Area [$W_{sleep}(\mu$m)]	29.3	21.6	15	39.5	81	153
BP	Delay	1	1	1	1	1	1
	$P_{dynamic}$	0.86	0.82	0.69	0.83	0.80	0.98
	$P_{leakage}$	0.044	0.077	0.051	0.068	0.05	0.0054
	$P_{dynamic}$ savings to [3]	14%	18.4%	31.4%	17%	20%	2%
	$P_{dynamic}$ savings to [4]	12.2%	15.9%	23%	14.4%	19.2%	0%
	$P_{leakage}$ savings to [3]	95.6%	92.3%	94.9%	93.2%	95%	99.5%
	$P_{leakage}$ savings to [4]	92.5%	84.8%	77.8%	83.3%	89.1%	88.5%
	# Sleep Trans	2	3	3	6	8	16
	ST_Area [$W_{sleep}(\mu$m)]	2.2	3.3	3.3	6.6	8.8	17.6
SP	Delay	1	1	1	1	1	1
	$P_{dynamic}$	0.93	0.91	0.81	0.89	0.91	0.98
	$P_{leakage}$	0.13	0.15	0.15	0.14	0.13	0.011
	$P_{dynamic}$ savings to [3]	7%	9%	19%	11%	9%	2%
	$P_{dynamic}$ savings to [4]	5.1%	6.2%	9%	8.2%	8.1%	0%
	$P_{leakage}$ savings to [3]	87%	85%	85%	86%	87%	98.9%
	$P_{leakage}$ savings to [4]	77.7%	70.4%	34.8%	65.6%	71.1%	76.6%
	# Sleep Trans	6	6	9	12	22	33
	ST_Area [$W_{sleep}(\mu$m)]	6.6	6.6	9.9	13.2	24.2	36.3

son basis. The SP technique produces large reductions in the sleep transistors total area compared to [3] and [4], but higher than BP because an additional constraint to the objective function is added (i.e routing cost) and no preprocessing is incorporated as explained in Section 4.1. The SP technique reduces the dynamic power on average by 16% and 6% compared to [3] and [4] respectively. This is attributed to the reduction of capacitance due to the down-sizing of the sleep transistors.

Furthermore, the SP technique achieves 88% and 66% leakage reduction compared to [3] and [4]. The main advantage of the SP technique is taking into consideration the location of the blocks in order to reduce the overall interconnects, providing more optimization to the area. The advantages of the SP technique will be even more evident in the DSM regime when interconnects dominate circuit performance and dynamic power. More-over, equally sized sleep devices as for BP and SP facilitate design for other circuits and provides more regular layouts. The area of the sleep transistor (ST) is equal to $W_{sleep} \times L_{sleep}$. Keeping the length of the sleep transistor (L_{sleep}) constant in the 4 techniques mentioned in Table 3, the sleep transistor width (W_{sleep}) can now be used as the sleep transistor area representative. The reduction in dynamic power is dependant on the number and size of sleep transistors and how big the circuit is (ratio of ST capacitance to overall circuit capacitance), while leakage power is only dependant on the number and size of the sleep transistors. Therefore, it can be noticed from Table 3, that the savings in leakage power is directly proportional to the reduction in total sleep transistor area. Finally, the proposed technique offers minimal area overhead, with no perturbation to the layout. This is attributed to the very narrow cavity (Figure 5) that holds the sleep transistors, which is located at a fixed location parallel to either the supply or ground rails. This further guarantees that the sleep transistor will not change the overall floorplan of the circuit.

6. CONCLUSIONS AND FUTURE WORK

Two techniques are applied to efficiently cluster gates in MTCMOS circuits. The first gives the minimum number of sleep transistors to be employed, while the second takes the circuit's routing complexity into consideration. On average the BP technique reduces dynamic and leakage power by 15% and 90% respectively. The SP technique also reduces dynamic and leakage power on average by 11% and 77% respectively. Future work involves improving the computation time involved to solve the SP and BP problems by using heuristic search techniques in the form of Genetic Algorithms that are suitable for multi-objective optimization problems.

7. REFERENCES

[1] L.Wei et al., "Design and optimization of dual-threshold circuits for low-voltage low-power applications," *IEEE Trans. on VLSI Systems*, pp. 16–24, 1999.

[2] S.Sirichotiyakul et al., "Stand-by Power Minimization through Simulataneous Threshold Voltage Selection and Circuit Sizing," Proc. of the 36th DAC, pp. 436–441, 1999.

[3] S.Mutah et al., "1-V Power Supply High-Speed Digital Circuit Technology with Multi-Threshold Voltage CMOS," *IEEE JSSC*, pp. 847–853, 1995.

[4] J.Kao et al., "MTCMOS Hierarchical Sizing Based on Mutual Exclusive Discharge Patterns," Proc. of the 35th DAC, pp. 495–500, 1998.

[5] J.Kao et al., "Transistor Sizing Issues And Tool For Multi-threshold CMOS Technology," Proc. of the 34th DAC, pp. 409–414, 1997.

[6] M.Bohr and Y.Elmansy, "Technology for Advanced High-Performance Microprocessors," *IEEE Trans. on Electron Devices*, vol. 45, pp. 620–625, 1998.

[7] R.Rardin, Optimization in Operations Research, Prentice Hall, 1998.