Analysis and Optimization of Static Power Considering Transition Dependency of Leakage Current in VLSI Circuits

Afshin Abdollahi
University of Southern California
afshin@usc.edu

Farzan Fallah
Fujitsu Labs of America
farzan@us.fujitsu.com

Massoud Pedram
University of Southern California
pedram@ceng.usc.edu

Abstract- We show that leakage current in VLSI circuits is not only a function of the current state (input combination) of a combinational circuit but also is dependent on the state history (previous input combinations.) As an example application of the transition-dependent leakage model, we extend a known technique for calculating and applying the minimum leakage input vector to a combinational circuit in the standby mode to one which calculates and applies a pair of input vectors to initialize the circuit to the minimum leakage configuration.

1. Introduction

Leakage current in deep submicron MOS transistors is becoming a significant contributor to power dissipation in CMOS circuits as threshold voltages and channel lengths are reduced. In CMOS devices the major leakage mechanism is subthreshold current, which increases due to the short channel effect. Consequently, estimation and control of subthreshold leakage current in CMOS circuits are important issues, especially in low power applications. On the other hand in the thin gate-oxide regime gate tunneling current between the gate and the source-drain extension overlap region, known as edge direct tunneling and between the gate and the channel becomes considerable in the total "off" state leakage current of the transistor [1]. The contribution of gate leakage to the total leakage is expected to increase in future technology generations [2].

The "transistor stacking" technique is proven to be highly effective in lowering the subthreshold leakage when a circuit is in the in the standby-mode of operation [3], [4]. The leakage current flowing through a stack of series connected transistors depends on the number of "off" transistors in the stack. In the two-input NAND gate shown in Fig. 1, turning "off" both M_1 and M_2 raises the intermediate node voltage to a positive value V_m due to a small drain current [3]. The positive potential at the intermediate node has three effects:

1) Gate-to-source voltage of M_1 (V_{gs1}) becomes negative;
2) Negative body-to-source potential (V_{bs1}) of M_1 causes higher body effect;
3) Drain-to-source potential (V_{ds1}) of M_1 decreases, resulting in lower drain-induced barrier lowering (DIBL).

This phenomenon is known as the "stacking effect." A negative V_{gs}, an increase in the body effect (negative V_{bs}), and a reduction in V_{ds} (less DIBL) reduce the subthreshold current exponentially. To analyze the effect of transistor stacking on the gate current, variation of the gate current with source voltage must be studied. With the gate voltage (V_g) at "0", the gate current is dominated by the edge direct tunneling current. At V_g="0", an increase in the source voltage (V_s) increases $|V_{gs}|$. An increase in $|V_{gs}|$ results in an increase in the tunneling through the gate-to-source overlap region. Hence, the total gate current increases. However, with V_g="1", an increase in V_s reduces the gate-to-source tunneling by reducing $|V_{gs}|$. The reduction in V_{gs} also reduces the gate-to-channel tunneling. In addition, at V_g="0" with a high drain voltage (V_d), $|V_{gd}|$ is also high. An increase in $|V_{gd}|$ results in an increase in the tunneling in the gate-drain overlap region, resulting in an increase in the gate current.

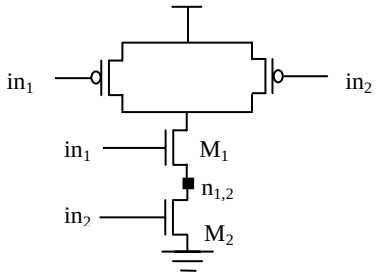


Fig. 1. A two input NAND gate.

Due to the stacking effect, leakage through a logic gate depends on the applied input vector. To evaluate the effect of input vector selection in controlling leakage, the pull-down network of the two-input NAND gate (a two-transistor stack) in Fig. 1 is considered. With both gates at logic "0", the intermediate node voltage is positive (V_m), resulting in a negative value of V_{gs1}. This reduces the subthreshold current in M_1. With all other input vectors, V_{gs} of M_1 and M_2 are either positive or zero. Thus, "00" gives the minimum subthreshold current flowing through a stack of two transistors. It has been shown that subthreshold current flowing through a stack of transistor decreases with an increase in the number of the "off" transistors. Hence, the "input vector control" technique can be effective in reducing the total subthreshold leakage of a circuit, which is in the standby mode [5].

The input vectors have an impact on the gate leakage which is different from that on the subthreshold leakage. With "10" as the input vector, $|V_{gs1}|=V_{th\_b}$ (the threshold voltage considering body effect), whereas with "00" as the input $|V_{g1}|=V_m(<V_{th})$. Hence, the gate-to-source overlap and gate-to-channel currents of M_1 with "10" (in_1=1 and in_2=0) are higher than the corresponding currents with "00". However, this increase is much less than the decrease in gate-to-drain tunneling in M_2 with "10" ($|V_{gd2}|=V_{dd}-V_{th}$) from the gate-to-drain tunneling in M_1 with "00" ($|V_{gd1}|=V_{dd}$). This is due to the fact that the rate of change of tunneling current density increases rapidly with an increase in V_{ox} (potential drop across oxide) [6]. Therefore, the total gate current in a stack with input "10" is less than that with input "00" because a decrease in V_{ox} reduces the tunneling current.

Hence, the determination of the minimum leakage input combination depends strongly on the ratio of the gate leakage to the subthreshold leakage. For larger devices with thicker gate oxides where subthreshold current dominates the gate leakage, the input vector with all zeros gives the minimum subthreshold current and, therefore, the minimum overall leakage. However, with a decrease in oxide thickness with device scaling, gate leakage may become a considerable component of the total leakage. Hence, turning on the top transistor may give the minimum leakage condition in future scaled devices. In section 2 transition dependency of leakage current is described. In section 3 we use the transition dependency of leakage current to further reduce the leakage by improving a previous technique. In sections 4 and 5 simulation results and conclusions are described.

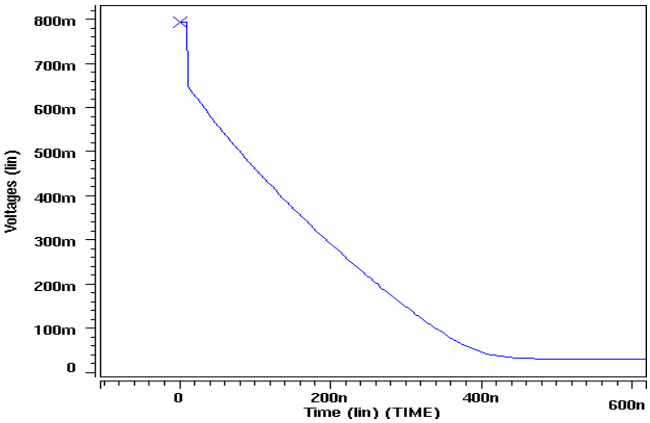


Fig.2.(a) Voltage of the internal node between series-connected NMOS transistors in the NAND gate of Fig. 1. after an input transition from "10" to "00" .

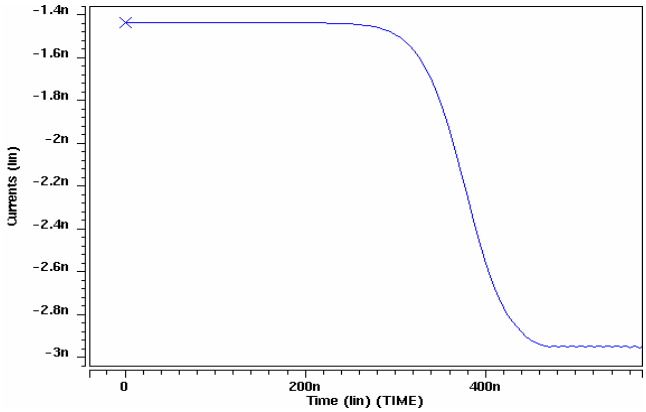


Fig.2.(b) Current of the voltage supply line of the NAND gate of Fig. 1. after a "10" to "00" input transition.

2. Transition dependency of leakage

In CMOS VLSI circuits, leakage current is generally known to be dependent only on the current state (input combination) of the circuit and is independent of the state history or state transitions. Currently, all leakage estimation and reduction techniques are based on the assumption that leakage is a memoryless function of the state of the circuit. In this paper, we will show that in some cases, leakage current is actually a function of not only the current state but also the previous state. The basic reason for this behavior is that although the outputs of logic gates reach their steady state voltage levels in a very short time (equal to the rise or fall time of the gates), for some input transitions, it can take a significantly longer time for some internal nodes of the logic gates to reach their steady state voltage levels. Notice that, in general, the leakage of a logic gate is a function of both the input voltage levels, and the voltage levels of the internal nodes of the gate. We will show that the dependence of leakage on internal node voltages results in a transient behavior for leakage current. The length of the transient time period tends to be noticeably long. Hence, in most cases the leakage value will not reach its steady state value before the state of the gate changes due to an input transition. This transient behavior is due to the process of slowly discharging the capacitive charge of internal nodes through OFF transistors in the path from the node to ground.

To make this discussion more clear, consider the pull-down network of the two-input NAND gate (a two-transistor stack) in Fig. 1. As described in the previous section with "10" as the gate input, the voltage of internal node, $n_{1,2}$, is V_{dd}-V_{th}, whereas with "00" as the input, the voltage of $n_{1,2}$ is $V_m << V_{dd}$-V_{th}. Existing leakage estimation techniques measure the leakage current of this NAND2 gate for the "00" input combination assuming that the voltage of the internal node is at its steady state level (V_m). Now consider the input transition from "10" to '00' at the input of the gate. After the input transition, the voltage of intermediate node drops slightly from V_{dd}-V_{th} (to V_n) due to capacitive coupling between the gate and source of transistor M_1. Next voltage level of $n_{1,2}$ slowly decreases toward zero mostly because of the subthreshold leakage current of M_2, but also because of the gate leakage currents from $n_{1,2}$ to gates of transistors M_1 and M_2. As a result the voltage of $n_{1,2}$ slowly drops to its steady state level V_m as shown in Fig. 2.(a). During this period, the current of the supply voltage is determined by the subthreshold current of transistor M_1 and the gate leakage current from the drain to source of transistor M_1 (the gate leakage currents of PMOS transistors are negligible.) Fig 2.(b). shows the current of voltage supply during the transition period.

During most of this transition, the drain to source voltage of transistor M_1 is much smaller than its steady state value. Therefore, the subthreshold current of M_1 (which accounts for the bulk of the leakage current from the supply line) is much less than its steady state value. Similarly, during the transition period, the source to gate voltage of M_1 ($|V_{gs1}|=V_n$) and the drain to gate voltage of M_2 ($|V_{gd1}|=V_n$) are higher than their steady state values ($|V_{gd1}|=|V_{gs1}|=V_m<V_n$). Therefore, the gate leakage current of both transistors increases. However, this gate leakage current is not drawn from the voltage supply line. Instead, it is coming from the charge, which is stored on the parasitic diffusion capacitance of $n_{1,2}$.

In summary during the transition period, the current of the voltage supply line is mostly due the subthreshold current of transistor M_1 which is significantly reduced because of the lower drain to source voltage of M1. For a two-input NAND gate the only input transition that results in transient leakage behavior is the transition from "10" to "00'. Based on the arguments in this and previous sections, applying the input pair "10" and "00" consequently, results in significantly more leakage reduction than applying any other pair of input vectors.

Generally in the pull-down network of a CMOS gate, under a two-input vector combination, an input transition from "1" to "0" applied to the gate terminals(s) of input transistor(s) that are closest to the output terminal of the gate, will minimize the leakage current if there is no conducting path to ground from the source terminal(s) of these transistor(s) under the second input vector . (A conducting path is a path of NMOS transistors with their gate voltages at "1".) In addition to the above case, depending on the structure of the gate, there may be more input transitions that may result in a situation for which the leakage current is dependent on previous input combinations. In this paper we only use one case described above for leakage reduction, since the above case is very likely to occur in our leakage reduction technique described in section 3.1.

From the above discussion, one concludes that leakage current estimation and reduction techniques need to be revised to comply with state transition dependence of leakage current. In [7], the authors introduced SAT-based methods for leakage reduction including input vector control and gate modification techniques. In this paper we improve the gate modification technique by considering the transition dependency of leakage current.

3. Leakage Minimization by Input Vector Control

By applying a minimum leakage vector (MLV) to a circuit, it is possible to decrease the leakage current of the circuit when it is in the standby mode. Note that applying MLV for leakage reduction is independent of the source of leakage, which may include the sub-threshold and the gate tunneling leakage currents. We assume that the environment in which the circuit is placed e.g., with the aid of a power management unit, generates a *SLEEP* signal for the circuit.

This signal is then used to initiate the application of the MLV to the circuit inputs. To use this method for leakage reduction, it is necessary to find an input vector that causes the minimum leakage current in a VLSI circuit.

3.1 Finding the Minimum Leakage Vector

A *trivial lower (upper) bound* on the leakage current is the sum of the minimum (maximum) leakage currents of all logic gates in the circuit. However, this may not correspond to any feasible solution because the input combination that produces the minimum (maximum) leakage in some gate, $gate_i$, may conflict with the one that produces the minimum leakage for another gate, $gate_j$. The following algorithm is used for finding an MLV for a given combinational logic circuit. Given a combinational logic circuit description, first a Boolean *Leakage Computing Network* (LCN), which computes the total leakage of the circuit is constructed. The LCN computes the leakage of the circuit as a function of the primary inputs and internal signals of the original circuit. Next from the LCN description, a set of Boolean clauses that capture the leakage current of the original circuit is generated. A SAT solver is employed to find an input vector that results in a leakage less than a given number *C*. Next, a linear search is performed on the value of *C* to find the MLV. Finally, the original circuit is modified by adding a number of multiplexers to shift in the MLV when the circuit enters the idle mode. Notice that the LCN is only used as a computational tool and the only actual hardware is the original circuit and the final circuit (which is augmented by the multiplexers and the MLV vector).

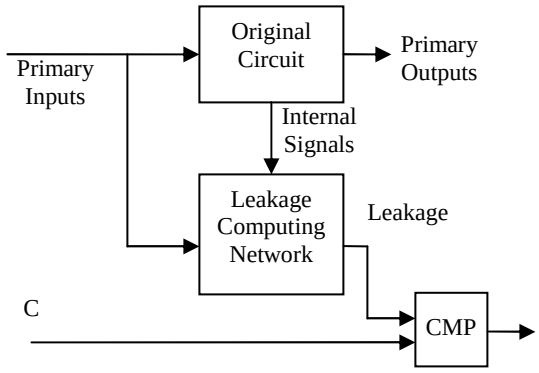


Fig. 3. Block diagram of the circuit structure that is input to the SAT.

The following algorithm performs a linear search on the values between *LB* and *UB* to find the minimum leakage current.

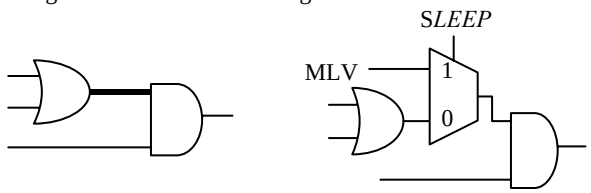
Algorithm LIN_SEARCH_FOR_MLV:

```
1.Find the trivial bounds on leakage
   current, LB and UB
2.C = UB, mlv = {}
3.Write Boolean clauses to model the
   circuit leakage and the condition that
   total_leakage ≤ C
4.Solve the resulting SAT problem
5.If there is no solution, stop; C + 1 is
   the  minimum leakage and mlv is the
   solution
6.mlv = the vector found by the SAT solver
7.C = C -1
8.If C < LB, stop; C + 1 is the minimum
   leakage and mlv is the solution
9.Go to step 3
```

The search starts from *UB* and proceeds toward *LB*. During the search all problems are feasible except the last one. Note that the constraints corresponding to $total\_leakage \leq C - 1$ are tighter than the ones corresponding to $total\_leakage \leq C$. Thus, every solution of iteration $i+1$ is a solution of iteration i. In every iteration, the SAT solver produces many conflict clauses during the search for the answer.[1] We use this fact to speedup the search by using the conflict clauses that are generated during the i^{th} iteration and adding new clauses to them to model the $(i+1)^{th}$ iteration. This is instrumental in substantially decreasing the computation time.

3.2 Leakage Reduction by Adding Control Points

In the previous section, we described how to reduce the leakage current by using an input vector control mechanism. However, in circuits with large logic depth, an externally applied input vector may effectively control only the gates that are close to primary inputs. If we find a way to directly control at least some of the internal nodes of a circuit, further reduction of the leakage of the circuit is possible. An easy way to control the value of an internal signal (line) of a circuit is to cut the internal line and insert a 2-to-1 multiplexer that is controlled by the SLEEP signal. The two inputs of the multiplexer include the incoming signal and a ZERO or ONE value decided by the leakage current minimization algorithm.



a) A line in a circuit b) The modified circuit

Fig.4. Replacing a line by a multiplexer.

The leakage cost of multiplexers serves as a disincentive to employ a large number of these multiplexers in the circuit. Therefore, an alternative method is described to control the outputs of internal gates in a circuit. Since the new method does not add any gate to the circuit, there is no extra leakage associated with adding a control point to the circuit.

We use three variables *X, Y* and *Z* (only two variables have been used in [7]) for each gate in the circuit. The value of *X* determines whether or not a gate in the circuit undergoes some change. The values of *Y* and *Z* determine the way that the gate is changed. Consider a fully-complementary CMOS gate, *out = g(in)*. Based on the values of *X, Y* and *Z*, which are in turn computed by our leakage minimization algorithm; This gate is changed as follows:

```
If (X==1) out = g(in)
else if (Y == 1)
     out = OR(SLEEP, g(in))
else out = AND(NOT(SLEEP),g(in))
```

As described above, modifying this gate enables controlling the output of the gate independent of its inputs in the standby mode. In other words, if we must have a "1" at the output of the gate when in

[1] Conflict arises when during the search one or more clauses become unsatisfiable in the current search sub-space. The SAT algorithm backtracks from this point and also learns form the conflict by adding one or more conflict clauses to its database. Adding such conflict clauses prevents the algorithm from encountering the same conflict. In other words, clauses prune the search space efficiently [10].

the standby mode, we replace the gate with *OR(SLEEP, g(in))*. Similarly, if we ought to have a "0", we replace it with *AND(NOT(SLEEP), g(in))*.

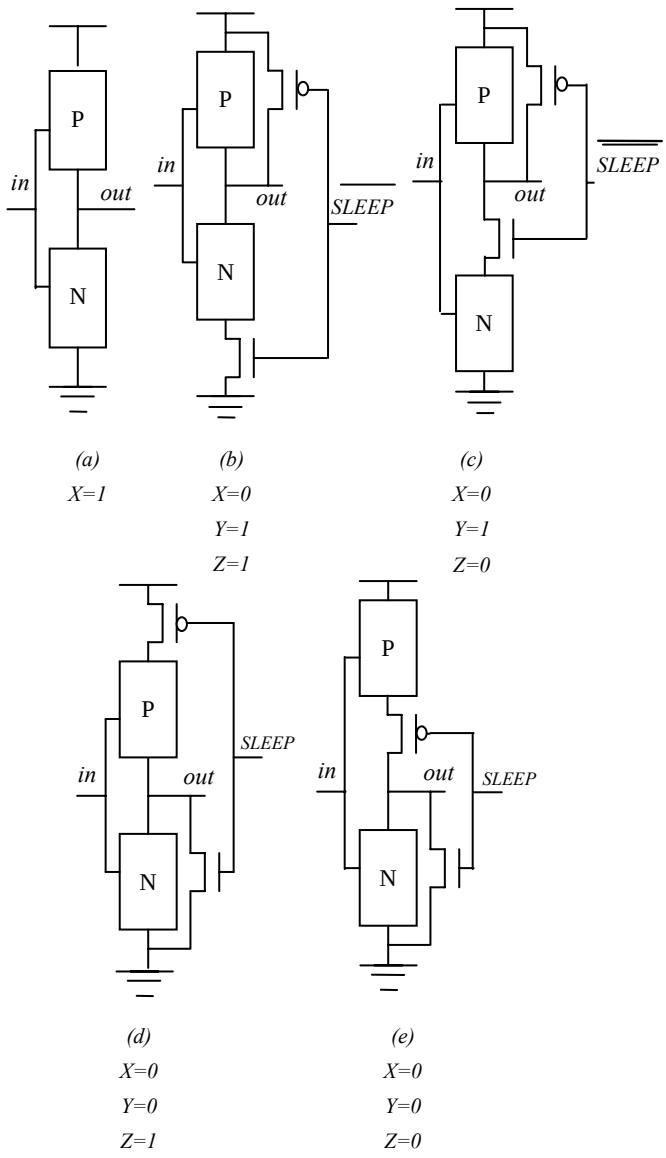


Fig 5. A complementary CMOS gate and various transistor-level leakage-guarding mechanisms

Fig.5. shows a CMOS gate with its PMOS and NMOS sections and four ways to modify the gate. In each case a transistor is added in series with one of the N or P sections of the gate. We consider two choices for inserting an NMOS (PMOS) transistor in series with the N (P) section of the gate.

1. Inserting the NMOS (PMOS) transistor near ground (Vdd) as in Fig. 5(b) and (d),
2. Inserting the transistor near output as in Fig. 5(c) and (e).

Note that in all cases adding the transistor in series with one of the N or P sections is different from adding a single high-Vth sleep transistor to the circuit. This sleep transistor may result in a number of complications, including large delay penalty in the active mode and undesirable leakage peaks while waking up a "sleeping" circuit, or "turning off" the circuit. In contrast, we replace some of the logic gates in the circuit with other "leakage guarded" gates with an extra input *(sleep)* and identical functionality when sleep=0.

The amount of the leakage reduction in each gate as a result of this modification depends on the number of transistors in the original gate [8]. The key advantage of the proposed method is that it enables us to control the values of the internal lines in the circuit; thus, reducing the leakage current of the gates that are driven by these lines. Modifying a gate in this way can result in a delay and an area penalty. For example, in cases (b) and (c), the high-to-low transition becomes slower, whereas in cases (d) and (e) the low-to-high propagation delay is increased.

First notice that, in the active mode, the inputs to the extra series- whereas inputs to the extra series-connected P transistors in Fig. 5(d) and (e) are set to fixed "0". Now in order to gain a higher switching speed for the gates in the active mode, it is desirable to insert the series-connected N or P transistors near the ground or V_{dd} terminals (cf. Fig. 5(b) and (d).) However, if we consider the transition dependency of leakage current as explained in section 2, then inserting the series-connected N or P transistors near the output of the gate (cf. Fig. 5(c) and (e)) results in lower leakage during the sleep period. This can be easily seen by treating the N section in Fig. 5 as a single equivalent NMOS transistor and following the discussion presented for the 2-inpout NAND gate of Fig. 1. Therefore, we have

$$delay(a) < delay(b) < delay(c)$$

$$leakage(a) > leakage(b) > leakage(c)$$

Similarly,

$$delay(a) < delay(d) < delay(e)$$

$$leakage(a) > leakage(d) > leakage(e)$$

We take the *pin-dependent* propagation delay of a gate to be the average of input-output gate delays for the rising and falling transitions. Obviously, the delay and area penalties depend on the sizes of the added transistors in each case. We size these transistors so that the increase in the delay and the area of each gate is no more than a given percentage.

In the sequel, we present a method to extend the LCN so that the leakage minimization is performed subject to a delay constraint on all of the primary input to primary output paths in the circuit. The circuit structure in Fig.6. (a) selects the correct value of the leakage for each gate in the circuit whereas the structure in Fig.6. (b) does the same for delay calculation.

Note in this figure $leakage_a$ *and* $delay_a$ denote the leakage current and propagation delay of the gate without modification (i.e., *out=g(in)*). As in static timing analysis, the gate delay values are used to calculate the maximum delay of the circuit for all input-output paths using the circuit shown in Fig.7. The arrival time of each gate is the maximum of the sum of the arrival time of each of its inputs and the pin-dependent delay from that input to the output of the gate.

The maximum delay of the circuit is the maximum of arrival times of its primary outputs. Fig.8 shows the circuit for comparing the maximum delay of the circuit with a given threshold.

The leakage minimization problem can be stated as that of minimizing the value of L_{total} which is a function of input vector and also variables *X*'s, *Y*'s and *Z*'s. The leakage minimization has to be performed under the delay constraint illustrated in Fig.9. Therefore, the minimization algorithm should take into account the values of leakage and delay as depicted in Fig.8.

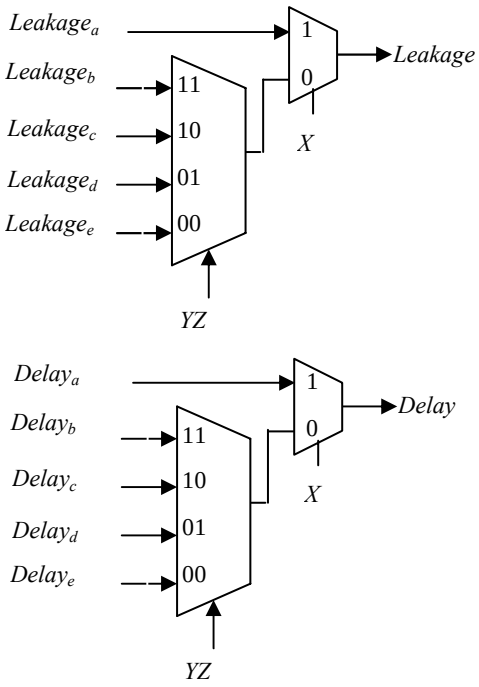


Fig. 6. (a) Leakage and (b) delay values of a modified gate.

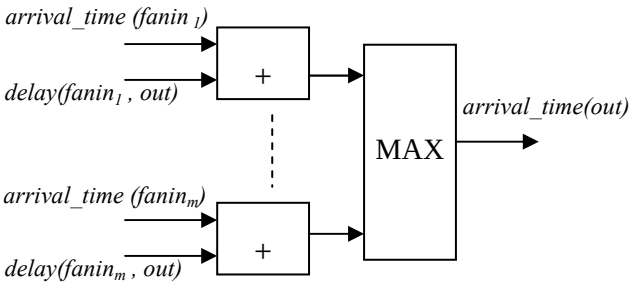


Fig 7. Calculating the output arrival time of a gate.

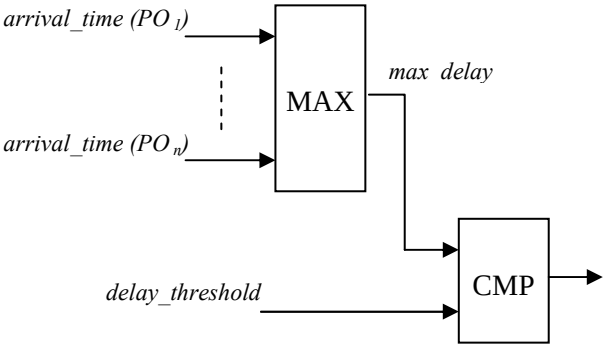


Fig.8. Comparing the maximum delay of the circuit with a delay threshold.

By running LIN_SEARCH_FOR_MLV on the modified LCN with the aforementioned Delay Computing Network (DLN) and variables (*X*'s, *Y*'s and *Z*'s), we can obtain the following:

ML, Gates that are structurally modified, *Y* and *Z* value for each modified gate, which identifies the method for modifying the gate.

Our minimization algorithm finds the optimum subset of gates, which are modified. The minimization algorithm considers the advantages of modifying the gates in the circuit (which are controlling internal signal as well as reducing the gate leakage due to the stack effect) and weighs them against the disadvantage of additional delay overhead due to the added transistors.

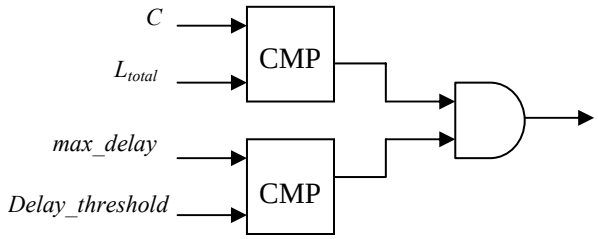


Fig. 9. Considering the delay constraint in leakage minimization.

4. Experimental Results

We applied the proposed technique to reduce the leakage currents of the circuits in the MCNC91 benchmark. Each of the circuits was optimized by the SIS *script.rugged* and mapped to a technology library using the SIS mapper. We used a library built in predictive 65nm CMOS technology [9] with a supply voltage level of 1V. We used HSPICE simulation to report the leakage current (including subthreshold and gate leakages) of the gates in the ASIC library for all possible combinations of their inputs considering the transition dependency of leakage. We used the technique in section 3.1. (adding control points) to find the MLV as well as the optimum subset of gates for modification and the way of modification by employing an efficient SAT solver[10].

Fig 10. shows the distribution of power savings in standby mode for the MCNC91 suite, which was achieved by using the control point addition mechanism of Section 3.1. under different delay constraints. When we do not allow any speed degradation, only a small number of gates are changed. As a result, the average energy saving is less than 25%. Increasing the limit on speed degradation helps improve the results as is evident from the figure. For example, with a 15% delay penalty, the average energy savings is 55%. The area overhead is proportional to the number of added transistors and is 15% at most.

Switching the inputs and internal control points of a circuit to its MLV and vise versa consumes some dynamic power. The amount of power saved using our runtime leakage control mechanisms depends on the duration of the standby mode for the circuit. For short standby periods, it is not worthwhile to switch between the current input and the MLV. For long standby periods, the energy savings can become quite significant. To make this statement more precise, we calculated the minimum duration of the idle time above by which power savings becomes possible when "shifting" the MLV in Fig.11 shows the distribution of this minimum time (in terms of the number of clock cycles in 1GHz) for MCNC91 benchmark circuits.

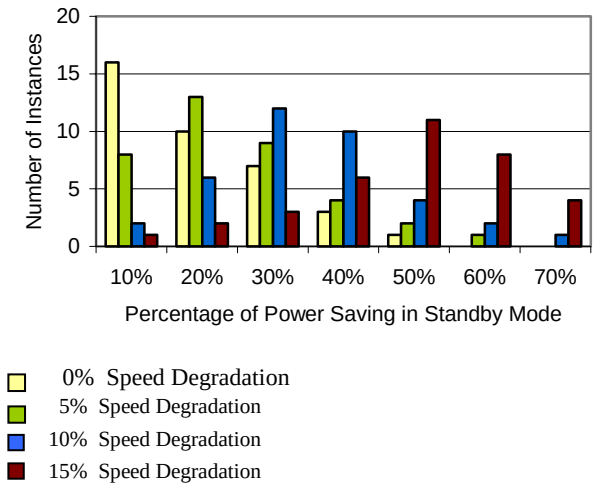


Fig. 10. Distribution of power savings in the standby mode achieved by using the control point addition mechanism of Section 3.1 (modifying gates) under different delay constraints.

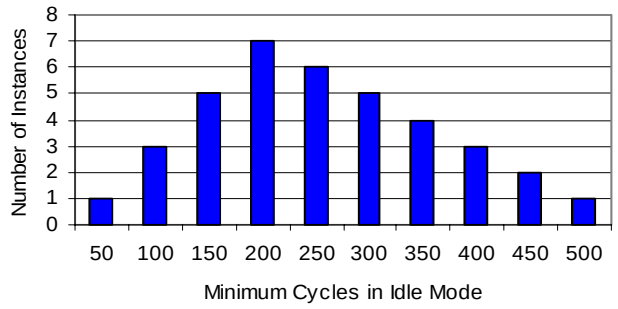


Figure 11. Minimum number of clock cycles that the circuit should stay in the standby mode for the dynamic leakage control to become effective.

We also measured the dynamic power penalty due to the overhead of additional transistors to the circuit, which increases the switching power because of added capacitance. Fig. 12 shows the dynamic power penalty for "adding control point" mechanism under different delay constraints. As can be seen, when we do not allow any speed degradation, only a small number of gates are changed so the additional capacitance overhead is small and the dynamic power penalty is on average 5%. When more speed degradation is allowed, dynamic power penalty is increased because more control transistors are employed. The dynamic power penalty is tolerable if the leakage saving in the idle mode is significant enough which would be the case if the aggregate idle times are sufficiently larger than the aggregate active times.

5. Conclusions

In this paper we showed that leakage current not only is a function of the current state (input combination) but also may be dependent on the previous states. Hence current leakage estimation and reduction techniques should be revised to comply with transition dependency of leakage current. We also exploited this fact for further leakage reduction by improving a previous leakage reduction technique (adding control points [7]) based on the transition dependency of leakage current. Simulation results show up to 70% leakage reduction for the benchmark circuits

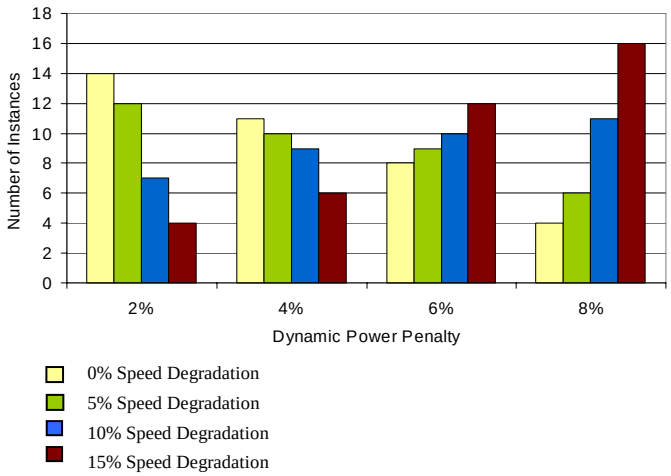


Figure 12. Dynamic power penalty for the method of section 3.1 under different delay constraints

References

[1] N. Yang, W. K. Henson, and J. Wortman, "A comparative study of gate direct tunneling and drain leakage currents in N-MOSFETs with sub-2100-nm gate oxides," *IEEE Trans. Electron Devices*, vol. 47, Aug. 2000, pp. 1636–1644.

[2] International Technology Roadmap for Semiconductors, Available: http://public.itrs.net/Files/2001ITRS/Home.htm

[3] V. De et al., "Techniques for leakage power reduction," in *Design of High-Performance Microprocessor Circuits*, A. Chandrakasan, W.J. Bowhill, and F. Fox, Eds. Piscataway, NJ: IEEE Press, 2001, ch. 3, pp. 52–55.

[4] Y. Ye, S. Borkar, and V. De, "New technique for standby leakage reduction in high-performance circuits," in *Symp. VLSI Circuits Dig. Tech. Papers*, 1998, pp. 40–41.

[5] Z. Chen, L. Wei, A. Keshavarzi, and K. Roy, "IDDQ testing for deep submicron ICs: challenges and solutions," *IEEE Design Test. Computers*, Mar.–Apr. 2002, pp. 24–33.

[6] S. Mukhopadhyay, C. Neau, R. Cakiki, A. Agrawal, C. Kim, K. Roy, "Gate Leakage Reduction for Scaled Devices Using Transistor Stacking, " *IEEE Transactions on VLSI Systems,* Vol. 11, No. 4, August 2003, pp. ???.

[7] A. Abdollahi, F. Fallah, and M. Pedram, "Leakage current reduction in CMOS VLSI circuits by input vector control." *IEEE Transactions on VLSI Systems*, Vol. 12, No. 2, Feb. 2004, pp.140-154.

[8] Assaderaghi, F., Sinitsky, D., Parke, S.A., Bokor, J., Ko, P.K. and Hu, C., "Dynamic threshold-voltage MOSFET (DTMOS) for ultra-low voltage VLSI," *IEEE Transactions on Electron Devices,* Vol. 44, No. 3, Mar. 1997, pp. 414 –422.

[9] Berkeley Predictive Technology Model (BPTM), Available:http://www-device.eecs.berkeley.edu/~ptm/ introduction .html

[10] L. Zhang, C. Madigan, M. Moskewicz, S. Malik, , "Efficient Conflict Driven Learning in a Boolean Satisfiability Solver," *Proc. of Int'l Conf. on Computer Aided Design*, Nov. 2001, pp. 279-285.