# Transistor Sizing Issues and Tool For Multi-Threshold CMOS Technology

James Kao, Anantha Chandrakasan, Dimitri Antoniadis
Department of EECS,
Massachusetts Institute of Technology, Cambridge

## ABSTRACT

Multi-threshold CMOS is an increasingly popular circuit approach that enables high performance and low power operation. However, no methodologies have been developed to size the high $V_t$ sleep transistor in an intelligent manner that trades off area and performance. In fact, many attempts at sizing the sleep transistor without close consideration of input vector patterns or internal structures can lead to large overestimates or large underestimates in sleep transistor sizing. This paper describes some of the issues involved in sizing transistors for MTCMOS and also introduces a variable breakpoint switch level simulator that can rapidly calculate delay in MTCMOS circuits as functions of design variables such as $V_{dd}$, $V_t$, and sleep transistor sizing.

## 1. BACKGROUND

Power consumption in conventional CMOS circuits can be attributed to switching power, leakage power, and short circuit power. Switching power is usually the dominant term and is given by the well known formula:

$$P_{switching} = \alpha C_L V_{dd}^2 f_{clk} \quad \textbf{(1)}$$

where $\alpha$ is the activity factor, $C_L$ is the total load capacitance, $V_{dd}$ is the supply voltage, and $f_{clk}$ is the clock frequency.

Clearly, to reduce this energy dissipated to charge and discharge load capacitances, the circuit designer's optimum choice is to scale the supply voltage down. However, in order to maintain performance, the threshold voltage should also be scaled down as well so that the gate drive, $(V_{gs} - V_t)$, remains large enough, since propagation delay in a CMOS gate can be approximated as:

$$T_{pd} \propto \frac{C_L V_{dd}}{(V_{dd} - V_t)^{\alpha}} \quad \textbf{(2)}$$

where $\alpha$ is for modeling short channel effects [1] [2]. By reducing $V_{dd}$, the switching power is reduced quadratically, but a reduction in $V_t$ causes an exponential increase in subthreshold leakage current. As one continues to scale down $V_{dd}$ and $V_t$, the increased leakage power can dominate the dynamic switching power [3].

In many event driven applications, like a processor running an X-server, circuits spend most of their time in an idle state where no computation is being performed, so large subthreshold leakage becomes unacceptable. Multi-threshold CMOS was developed in order to reduce this leakage current during idle modes by providing a high threshold "gating" transistor in series with the low $V_t$ circuit transistors. In active mode, the high $V_t$ transistor is turned on, while in sleep mode it is turned off, providing a small subthreshold leakage current [4]. For a purely combinational circuit, where state does not need to be preserved, only one type of high $V_t$ device is actually required. The NMOS is preferable because it has a lower on resistance and can be sized smaller than a corresponding PMOS sleep transistor.

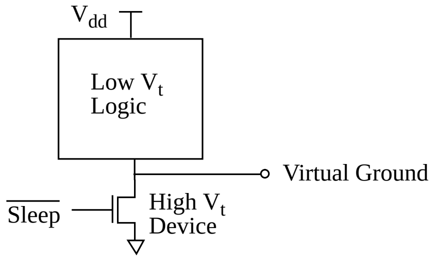


**Figure 1.** MTCMOS circuit structure.

Many other alternatives such as dual gated SOI, substrate biasing, or switched source impedance (closely related to MTCMOS) have recently been proposed to address the conflicting requirement of high performance during active periods and low leakage during idle times [5] [6] [7] [8]. However, MTCMOS has emerged as one of the more practical solutions that can be easily implemented using minor modifications to current designs and technology. The MTCMOS process only requires an extra implant step to produce the high $V_t$ devices, and the circuit implementation can be based on existing CMOS designs. Recently, several large chips have been fabricated and tested including a 1-V DSP chip for mobile phone applications [9].

## 2. ISSUES IN SIZING MTCMOS CIRCUITS

Correct sleep transistor sizing is a key parameter that affects the performance of MTCMOS circuits. If sized too large, then valuable silicon area would be wasted and switching energy overhead would be increased, but on the otherhand if sized too small, then the circuit would be too slow because of the increased resistance to ground. Although there has been much activity and development of MTCMOS circuits recently, little work has been done on methodologies for sizing the high $V_t$ sleep transistors. One possible approach to estimate the transistor size is to sum the widths of internal low $V_t$ transistors, but this can produce unnecessarily large estimates for transistor sizes. Designers may also try to design for peak current spikes [4], but this too gives overly conservative estimates. Ideally, one could simulate circuits for varying sleep transis-

DAC 97, Anaheim, California

tor sizes with SPICE, but this can be very time consuming, especially if one tries to exhaustively test all possible input vectors for a complicated combinational circuit like an adder or multiplier. Clearly, a better, more informative method of sizing the sleep transistor is necessary. The remainder of this paper will attempt to address some of the issues involved in how circuit performance depends on correct sleep transistor sizing, and will also propose a switch based simulation that can rapidly estimate delay in MTCMOS circuits.

### 2.1 Finite Resistance Approximation For High $V_t$ Sleep Transistor

The effect of an "ON" NMOS sleep transistor in series with a low $V_t$ circuit can be approximated very accurately by replacing the high $V_t$ device with a single linear resistor R. During normal circuit operation, the virtual ground node is close to real ground, so $V_{ds}$ of the sleep transistor is small and the resistive approximation is very accurate.

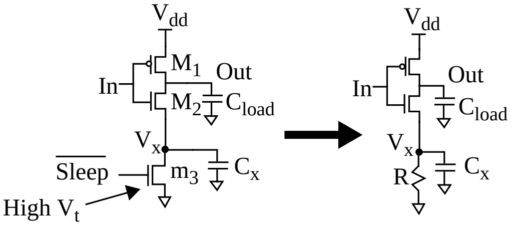


**Figure 2.** Sleep transistor modeled as resistor.

Analysis of the MTCMOS inverter shown in Figure 2, while simplistic, still can give us valuable insight into the relationship between sleep transistor size and circuit performance. First of all, it is important to see that only the output high to low transition is affected by the insertion of an NMOS sleep transistor and that the low to high transition behaves exactly the same as conventional CMOS circuits. When the inverter is discharging, and neglecting the parasitic capacitance $C_x$, any charge flowing out of the source of $M_2$ will flow through the sleep resistor R, inducing a voltage drop $V_x$. This voltage drop has two effects: first it reduces the gate drive from $V_{dd}$ to $V_{dd}$-$V_x$, and second it causes the threshold voltage of the pulldown NMOS to increase due to the body effect. Both changes result in a decrease in the discharging current, which slows the output high to low transition. To maximize performance, the resistor should be made as small as possible and consequently the transistor as large as possible. The size of the sleep transistor is of course limited by area constraints, but increased switching energy overhead and increased leakage current can also be limiting factors. As one continues to scale $V_{dd}$ to lower voltages, the effective resistance of the sleep transistors will increase dramatically, requiring even larger size sleep transistors.

### 2.2 Impact of Virtual Ground Parasitic Capacitance

The parasitic capacitances due to wiring and junction capacitances on the virtual ground actually helps reduce the virtual ground line bounce by serving as a local charge sink or reservoir for current [4]. However, this capacitance would have to be extremely large in order to offset the effects of a poorly sized sleep transistor. The RC network serves as a lowpass filter, where the RC time constant would have to be large enough such that the virtual ground voltage can only rise to a fraction of it's peak DC value (I * R). Since the resistance is typically low, the capacitance required can be on the order of pico farads. For more complicated logic blocks, the current profile may have a very long period (lower frequency content), and thus even larger capacitances are required to ensure a slow enough rise time.

If the time constant is very large, then it will also take longer for the virtual ground node to discharge back to ground after a transition. For example, if the virtual ground is slow to discharge, then later gates (that are not sinking too much current) might be slowed down excessively and could have operated faster had there been a smaller parasitic capacitance on the virtual ground node. Rather than rely on large capacitances to ensure MTCMOS performance, it is much easier to lower the effective resistance with proper transistor sizing instead. Also, since SOI is emerging as a likely candidate for low power circuit design, and SOI has small junction capacitances, one cannot rely on any significant capacitive loading to improve switching performance in MTCMOS (SOI) circuits [10].

### 2.3 Reverse Conduction Paths in MTCMOS

MTCMOS logic blocks can also suffer from reverse conduction, where current flows from the virtual ground through the low $V_t$ NMOS transistor and charges up the output capacitance (or conversely the output capacitance partially discharges as current flows up towards a virtual $V_{dd}$ line in the case for a PMOS sleep transistor). To be more specific, in the NMOS case, the virtual ground node can rise above 0V so that another gate, which is supposed to be low, can experience reverse conduction as the output voltage rises from 0V to $V_x$. This charging current comes from the discharging current of other gates transitioning from high to low, where only a fraction of the discharge current is actually bypassing the sleep transistor. As a result, the MTCMOS circuit is slightly faster because the $V_x$ voltage drop is not quite as large as one would expect if all current flowed through the sleep transistor to ground. Another effect of the reverse conduction, which pins output low voltages to $V_x$, is that a gate charging from low to high would be faster since it is already precharged to $V_x$. The drawback is that the noise margins in the circuits are reduced, and in the worst case the circuit can fail logically.

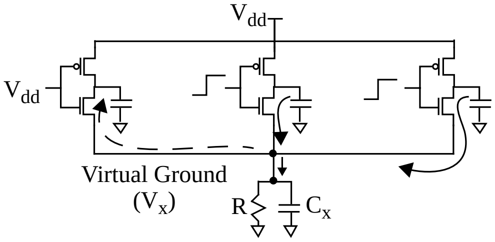


**Figure 3.** Reverse conduction paths.

### 2.4 Input Vector Dependency

For more complex MTCMOS circuits, the input vector plays a very important role in determining worst case circuit performance. For example, the worst case pattern for a base CMOS design will not typically translate to the worst case pattern for an MTCMOS implementation because the MTCMOS circuit will be slowed down due to virtual ground bounce. Thus MTCMOS circuits will be more susceptible to input vectors that will cause large currents to flow through the sleep transistors, whereas ordinary CMOS circuits will not be affected. When analyzing MTCMOS circuits, one cannot simply examine a “critical path” in the circuit, but must also consider all other accompanying gates that are switching. Because the worst case delay is strongly affected by different input vectors and glitching behavior, it is very difficult to correctly size the sleep transistor. In fact, even among different sleep transistor sizing choices in MTCMOS circuits, the worst case input patterns may vary. Section 4 describes in more detail how choice of input vector can affect the sizing requirements of an 8x8 multiplier.

## 3. INVERTER TREE EXAMPLE

The following figure is a typical inverter tree structure implemented in an MTCMOS technology where an NMOS sleep transistor lies between virtual ground and ground. This circuit structure very clearly demonstrates how several gates can switch simultaneously and create large time varying voltage drops across the sleep transistor that slow down the circuits at different rates during signal propagation.

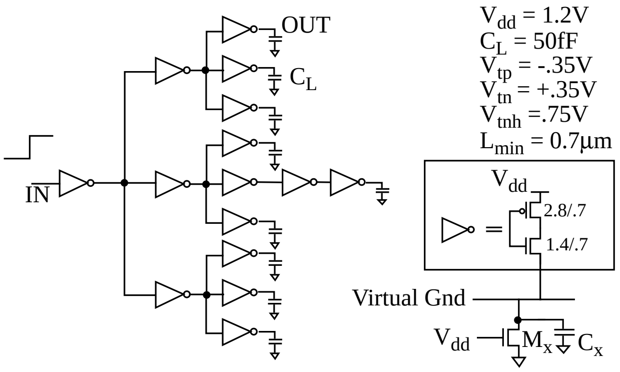


**Figure 4.** MTCMOS inverter tree.

In this example, the input 0->1transition is especially slow because in the third stage, all nine inverters are discharging, which causes the virtual ground line to bounce. Figure 5 shows the virtual ground transient and reveals an initial “bump” when the first inverter is discharging and a larger “bump” when the third stage is reached. The figure also shows how the output waveform slows down when the sleep transistor width is too small.

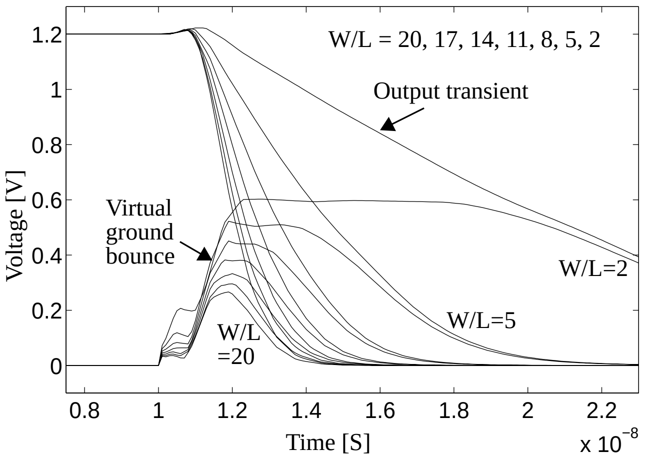


**Figure 5.** Inverter tree SPICE simulation for various W/L.

## 4. MULTIPLIER EXAMPLE

A larger MTCMOS circuit like an 8x8 bit carry save multiplier demonstrates the impact of input vector on circuit performance. Because of size limitations, Figure 6 shows only a 4x4 version with a worst case delay path highlighted.

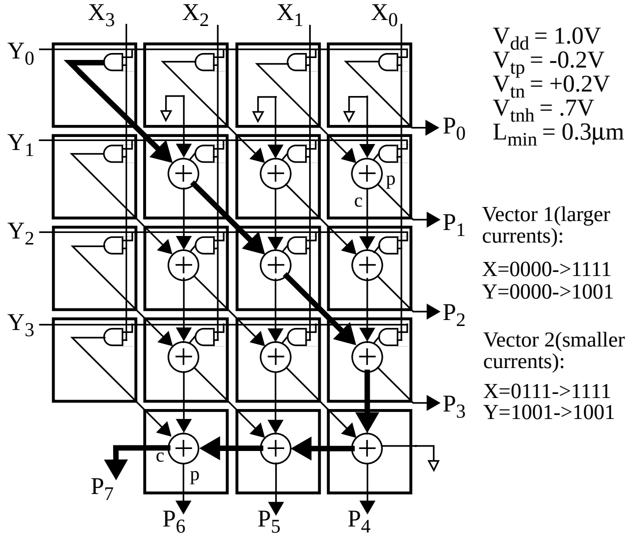


**Figure 6.** Carry save adder diagram (4x4bit version)[11].

Because of the regularity of this implementation, it is easy to see that one critical path (many others exist) lies along the diagonal and bottom row. However, two distinct input vectors that give the same delay in a CMOS implementation can give very different results in an MTCMOS circuit. The transition from (x:00,y:00) -> (x:FF,y:81) for example causes many more internal transitions in adjacent cells and thus is more susceptible to ground bounce than the (x:7F, y:81) -> (x:FF, y:81) transition. The second input causes a rippling effect through the multiplier, where only a few blocks are discharging current at the same time. Figure 7 shows how delay varies with the W/L ratio of the sleep transistor for these two cases.

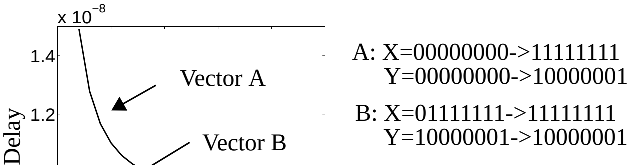


**Figure 7.** 8 bit multiplier delay vs. W/L for different input vectors (SPICE).

| Initial<br>X Y | Final<br>X Y | Delay<br>CMOS | % Degr<br>W/L=60 | % Degr<br>W/L=170 |
|---|---|---|---|---|
| 0x 00 00 | 0x FF 81 | 8.96ns | 18.1% | 5% |
| 0x 7F 81 | 0x FF 81 | 8.93ns | 4.8% | 1.7% |

**Table 1.** CMOS delay, and % degradation for various W/L.

Table 1 summarizes some key values from the plot. For example if one wished to size the sleep transistor to provide less than 5% speed penalty for vector A, then one must size the sleep transistor greater than W/L=170. On the otherhand, if one were to examine the vector B, the same analysis could lead one to erroneously size the sleep transistor to be only W/L=60, which would actually correspond to an 18% degradation in speed for the previous case. Since input vector strongly influences delays in MTCMOS, it is very important to determine the worst case input vector for properly sizing sleep transistors.

An alternative to sizing for the worst case input vector is to try to size for the worst case peak current and to ensure that the virtual ground does not cross a threshold. However, this tends to be an extremely conservative approximation since current levels will usually not peak throughout the entire logic computation period. Instead, in the context of MTCMOS, gates will slow down during large current spikes but speed up again when fewer gates are transitioning. To emphasize this point, the maximum current for the (00 00)->(FF,81) transition was simulated to be 1.174mA (not necessarily the actual peak current experienced by the circuit). If the virtual ground bounced were fixed, then a 50mV offset would result in a 5% degradation. Assuming the fixed current of 1.174mA, then one would have to size the sleep transistor with W/L greater than 500, which is almost three times larger than necessary.

To optimally size a sleep transistor, one must accurately determine the worst case input vector, which can be a very difficult task. Although one could exhaustively simulate all possible input transitions with SPICE for smaller circuits, it soon becomes impossible with more complicated logic blocks. Furthermore, current tools to extract critical paths may not be adequate since they do not take into account the virtual ground bounce associated with discharge currents.

## 5. MTCMOS DELAY ANALYSIS TOOL

To help analyze worst case input vector patterns, a switch level variable breakpoint simulator was developed to rapidly compute delay as a function of sleep transistor size. The advantage of this simulator is that first order timing information can be gathered very quickly for very large input vector spaces. Rather than using the delay information directly, the tool is more useful for identifying potential vectors that will cause large variations in an MTCMOS circuit and can be used to narrow down the vector space to be analyzed with a more detailed simulator like SPICE.

### 5.1 Simple Model For MTCMOS Propagation Delay

To model the effects of MTCMOS on circuit delay, it is useful to consider the delay of an inverter when N-1 other inverters are simultaneously switching through a shared sleep transistor.

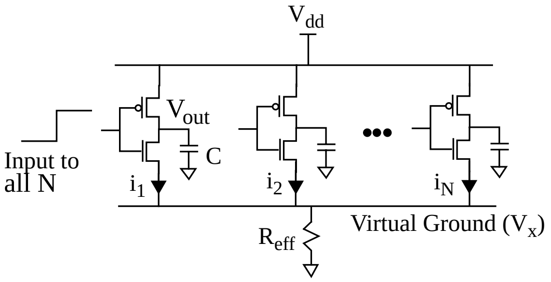


**Figure 8.** Circuit model for MTCMOS delay.

$V_x$ can be assumed to be the equilibrium point where the current $V_x/R_{eff}$ is equivalent to the sum of the saturation currents that are set by the reduced gate drive of each gate. Assuming the discharge current is constant and all gates are switching continuously during the period, the propagation delay for a particular gate (jth) can be modeled as:

$$T_{pdhl} \approx \frac{C_L V_{dd}}{2 I_j} \qquad (3)$$

where $I_j$ needs to be solved for explicitly shown in Eq. 5 below. By summing the total mosfet gain factors for each discharging gate, where $\beta_j = \mu_n * C_{ox} * (W/L)$ and $\beta_{total} = \beta_1 + ... \beta_n$, and equating $V_x$ to the voltage drop across the sleep resistor, we have:

$$V_x = \frac{1}{2}\beta_{total}(V_{dd} - V_x - V_t)^2 R_{eff} \qquad (4)$$

This can easily be solved for $V_x$, which can be used to compute the saturation current from the jth gate.

$$I_j = \frac{1}{2}\beta_j (V_{dd} - V_t - V_x)^2 \qquad (5)$$

### 5.2 Variable Breakpoint Switch Level Simulation Tool

The underlining algorithm behind this tool is to dynamically adjust each gate's propagation delay based on the total number of gates switching, since different amounts of currents will produce different voltage drops across the sleep transistor. If each gate is modeled as an equivalent inverter with an effective load capacitance $C_L$, then the delay model derived in the previous section for N inverters discharging simultaneously can be applied directly to more complex logic circuits [12].

The input and output voltage waveforms for each gate are treated as piecewise linear, and gates are assumed to begin switching exactly when the input voltage exceeds $V_{dd}/2$. In the case of an ordinary CMOS implementation (with sleep resistance equal to 0), the simulation tool simply models each gate as a constant current source that discharges a load capacitance. When a finite sleep resistance is introduced in the circuit, the gates are modeled as time varying (stepwise) current sources discharging their respective load capacitances, which results in a piecewise linear output voltage whose slopes can vary in time. These breakpoints occur whenever a gate in the logic block starts or stops switching because delays must be recomputed when the total current flowing through the sleep transistor changes. With each gate modeled as a first order dynamic system, one only needs to keep track of the current output voltage (state) and input stimulus to predict the delay behavior.

In order to process these breakpoints, the simulator computes an associated "best guess" for time to reach the switching threshold and time to finish switching for each gate. The simulator time steps to the nearest breakpoint, determines if any new elements are switching and then recomputes the "best guess" for these breakpoints by taking into account slower or faster gate transitions. The breakpoint times for individual gates are not fixed because if another gate switches first, then the speed of the subsequent gate will change, requiring a new delay calculation. For a simulation time of $T_{sim}$, current drive of $I_j$, and load capacitance $C_L$, a discharging gate who's output voltage is currently > $V_{dd}/2$ would have it's expected "switching threshold" breakpoint calculated as:

$$T_{dnext} = T_{sim} + \frac{C_L}{I_j}\left(V_{out}(T_{sim}) - \frac{V_{dd}}{2}\right) \quad V_{out} > V_{dd}/2 \qquad (6)$$

Conversely, the simulation time breakpoint corresponding to when the gate finishes transitioning is represented by:

$$T_{end} = T_{sim} + \frac{C_L}{I_j} V_{out}(T_{sim}) \qquad V_{out} > 0 \qquad \textbf{(7)}$$

Figure 9 shows the output waveforms as functions of time for three different gates in a larger MTCMOS circuit. One breakpoint is labeled as $t_i$, corresponding to the switching threshold of gate 2, and another is shown as $t_{i+1}$, corresponding to the time gate 1 finishes discharging. The other six breakpoints are not labeled.

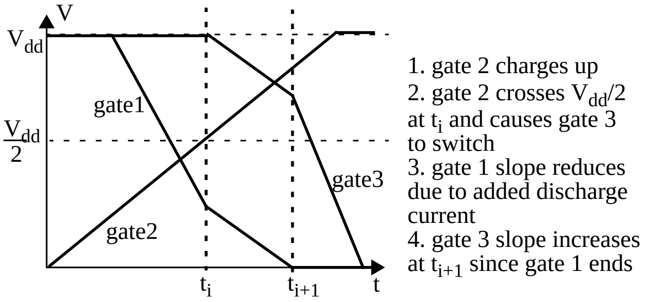

V
$V_{dd}$
$\frac{V_{dd}}{2}$
gate1
gate2
gate3
$t_i$
$t_{i+1}$
t
1. gate 2 charges up
2. gate 2 crosses $V_{dd}/2$ at $t_i$ and causes gate 3 to switch
3. gate 1 slope reduces due to added discharge current
4. gate 3 slope increases at $t_{i+1}$ since gate 1 ends

**Figure 9.** Typical output waveform transitions in variable breakpoint simulator.

Immediately before time $t_i$, gate 1 is discharging at a constant slope and gate 2 is transitioning from low to high. However, at the breakpoint $t_i$, gate 2 passes the threshold voltage and causes gate 3 to begin discharging. This increased current causes the virtual ground to bounce, and consequently both gate 1 and gate 3 slow down. At this point subsequent breakpoints will have to be updated to reflect slower circuits, so that the next breakpoint, $t_{i+1}$, is actually later in time than what was predicted earlier. When gate 1 finishes switching, gate 3 will speed up because less current needs to be sunk through the sleep transistor. Again, the breakpoints are recomputed at this point to reflect different operating conditions. The variable breakpoint simulator thus only needs to simulate the circuit at breakpoints which are variable in time and computed from the current operating conditions.

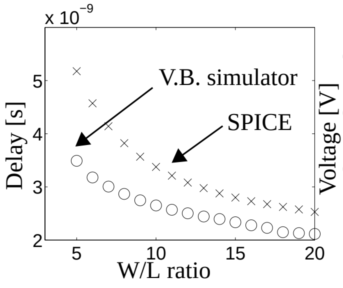
### 5.3 Limitations of Switch Level Simulator

The delay model used in the variable breakpoint switch level simulator has several limitations. First of all, the assumption that the output capacitance is discharged by a current source equal to the saturation current $I_j$ is simply false, since the transistor does spend time in the triode, or linear region of operation. Second, we neglect the effect of parasitic capacitances on the virtual ground line, but this effect becomes important only for large resistances or large capacitances. Also, the effect of the input slope on output delay time [1] [13] is ignored, and only a very simplistic first order MOSFET model (neglecting body effect, channel length modulation, velocity saturation) is used. Another important limitation is that complicated gates are modeled as a simple inverter, which can also lead to timing inaccuracies. By addressing these issues in future work, the simulator accuracy can be improved significantly. However, since the simulator is most useful for qualitative analysis in determining potential vectors that are sensitive to MTCMOS, complete timing accuracy is not mandatory.

## 6. SIMULATION RESULTS FOR VARIABLE BREAKPOINT SWITCH LEVEL SIMULATOR

### 6.1 Inverter Tree Application

The variable breakpoint switch level simulator gives reasonable results when applied to the clock distribution inverter network shown in Figure 4 with a low to high input transition. Figure 10 compares delay measurements computed from SPICE with measurements obtained from the switch level simulator.



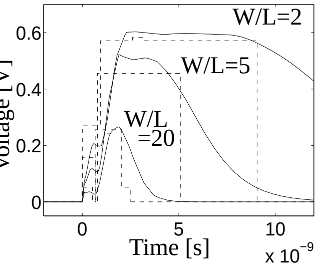


**Figure 10.** Delay comparison as function of W/L.

**Figure 11.** Ground bounce transient comparison.

The variable breakpoint simulator captures the basic effect of sleep transistor sizing on propagation delay, and even though it is based on a first order delay model, still manages to track the switching variations of this MTCMOS circuit. Figure 11 shows the virtual ground variation in the inverter tree during the transition as computed from SPICE as well as the simulator. Since the simulator models discharging gates as constant current sources and neglects the effects of capacitance in parallel with the sleep transistor, the ground bounce should be a stepwise function. For the very high resistance case (unrealistic/ undesirable in actual circuits), the virtual ground is very slow in discharging due to a larger RC time constant.

### 6.2 Results From Adder Simulation

A 3 bit ripple carry adder was exhaustively simulated both with SPICE and with the variable breakpoint switch level simulator. The adder is a standard “mirror adder” implemented with 3x28 transistors, and the circuit was simulated with the initial carry bit grounded, but using every possible pair of 6 bit input vectors. This resulted in $2^6 * 2^6 = 4096$ possible vectors.

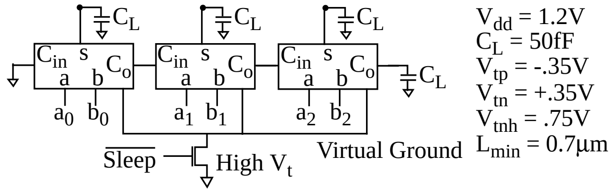


**Figure 12.** 3 bit MTCMOS ripple adder.

Even for such a small circuit, SPICE required 4.78 hours of CPU time on a Sun Sparc 5 to simulate all 4096 input vectors. On the otherhand, the variable breakpoint switching simulator required only 13.5 seconds of CPU time, and the code has not yet been optimized for speed. Figure 13 shows a comparison between the propagation delay on the 3 bit ripple carry adder as a function of W/L between SPICE and the variable breakpoint switch level simulator.

Two different vector pairs are simulated: (x:000 y:001)->(x:110 y:101) is the worst case delay vector pattern, while (x:110 y:001) -> (x:110 y:101) is a vector less susceptible to MTCMOS delay. From the plot, we can see that the simulator gives extremely good results for delay. Since the experimental circuit is small, the variation in delay as a function of different input vectors is not as pronounced as in the 8 bit multiplier.

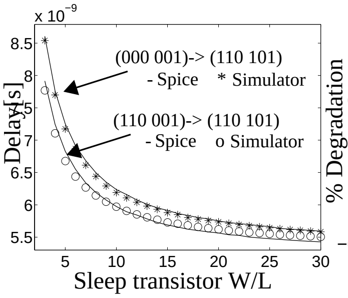


**Figure 13.** Delay comparison of 3 bit adder for two different input vectors.

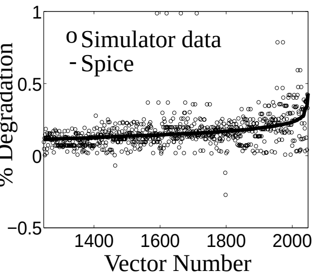


**Figure 14.** % degradation due to MTCMOS for 800 vectors.

Figure 14 shows how different input vectors are susceptible to delays in MTCMOS. The solid line shows the percent degradation due to MTCMOS (W/L=10) measured in SPICE for 800 vector transitions (ordered from worst degradation to best) that involve a transition on the $S_2$ bit. The data points shown correspond to the same calculation computed with the variable breakpoint simulator. Although the simulator shows a significant spread about the SPICE prediction, the general trend is correct.

### 6.3 Simulator Accuracy

The accuracy of the simulator needs to be improved, but the results so far have shown that the initial simulator does follow the trends in MTCMOS delay as a function of sleep transistor sizing. The adder delay measurement was much more accurate than the inverter tree simulation, and a likely explanation for this is that load capacitances and gate drives are matched more closely to SPICE in the adder experiment. Figure 14 does show that for many input vectors, the simulator results deviate significantly from SPICE predictions. One possibility is that the variable breakpoint simulator is too sensitive to circuit glitches, and work is currently being done to improve this. Other mismatches between SPICE and the simulator can be attributed to a very simplistic delay model that does not take into account the second order effects described in section 5.1. By improving the simulator to better model glitches in MTCMOS and taking into account effects like velocity saturation, body effect, reverse conduction paths, parasitic capacitances, and better compound gate models, we can significantly improve the accuracy of the variable breakpoint switch level simulator.

## 7. CONCLUSION

Multi-threshold CMOS is becoming a very popular circuit technique for low power, high performance applications. Recently there has been a great number of MTCMOS implementations, but as this technology becomes more widestream, it will be important to develop some important sizing methodologies for the high $V_t$ sleep transistor. This paper described some of the issues presented in sizing MTCMOS circuits, and then proceeded to develop a simple MTCMOS delay model that was applied to a variable breakpoint switch level simulator that could very quickly simulate large numbers of input vectors. The key for this tool was to provide the circuit designer with initial delay information as a function of input vector, $V_{dd}$, $V_t$, and sleep transistor sizing, so that the he/she may recognize input vector patterns that may be especially susceptible in MTCMOS circuits. After the design and simulation space is narrowed sufficiently, the designer could then use a more detailed simulator like SPICE to verify circuit details.

## 8. ACKNOWLEDGEMENTS

This work was funded by DARPA contract #DABT63-95-C-0088.

## 9. REFERENCES

[1] T. Sakurai, R. Newton, “Alpha-Power Law MOSFET Model and its Applications to CMOS Inverter Delay and Other Formulas,” IEEE JSSC, vol. 25, no. 2, pp. 584-594, April 1990.

[2] T. Sakurai, R. Newton, “A Simple MOSFET Model for Circuit Analysis,” IEEE Transactions on Electron Devices, vol. 38, no. 4, pp. 887-894, April 1991.

[3] A. Chandrakasan, I. Yang, C. Vieri, D. Antoniadis, “Design Considerations and Tools for Low-voltage Digital System Design,” 334d Design Automation Conference, pp. 113-118, June 1996.

[4] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, J. Yamada, “1-V Power Supply High-Speed Digital Circuit Technology with Multithreshold-Voltage CMOS,” IEEE JSSC, vol. 30, no. 8, pp. 847-854, August 1995.

[5] T. Kawahara, M. Horiguchi, Y. Kawajiri, G. Kitsukawa, T. Kure, “Subthreshold Current Reduction for Decoded-Driver by Self-Reverse Biasing,” IEEE JSSC, vol. 28, no. 11, pp. 1136-1144, Nov. 1993.

[6] I. Yang, C. Vieri, A. P. Chandrakasan, and D. Antoniadis, “Back Gated CMOS on SOIAS for Dynamic Threshold Control,” IEEE 1995 International Electron Devices Meeting (IEDM), pp. 877-880, December 1995.

[7] T. Kuroda, T. Fujita, et al, “A 0.9V, 150MHz, 10mW, 4mm$^2$, 2-DCT Core Processor with Variable VT Scheme,” IEEE JSSC, vol. 31, no. 11, pp. 1770-1778, Nov 1996.

[8] K. Seta, H. Hara, T. Kuroda, M. Kakumu, T. Sakurai, “50% Active-Power Saving Without Speed Degradation Using Standby Power Reduction (SPR) Circuit,” IEEE ISSCC, pp 84-85, 1995.

[9] S. Mutoh, S. Shigematsu, Y. Matsuya, H. Fukada, J. Yamada, “1V Multi-Threshold CMOS DSP with an Efficient Power Management Technique for Mobile Phone Application”, IEEE ISSCC, pp. 168-169, 1996. 1995, pp. 318-319, 1995.

[10] . Douseki, S. Shigematsu, Y. Tanabe, M. Harada, H. Inokawa, T. Tsuchiya, “A 0.5V SIMOX-MTCMOS Circuit with 200ps Logic Gate”, IEEE ISSCC, pp. 84-85, Feb. 1996.

[11] N. Weste, K. Eshraghian, “Principles of CMOS VLSI Design,” Addison-Wesley, Reading MA., p. 548, 1993.

[12] T. Sakurai, R. Newton, “Delay Analysis of Series-Connected MOSFET Circuit,” IEEE Journal of Solid State Circuits, Vol. 26, No.2, Feb 1991.

[13] S. Dutta, S. Shetti, S. Lusky, “A Comprehensive Delay Model for CMOS Inverters,” IEEE Journal of Solid State Circuits, Vol. 30, No. 8, August 1995.