Calculating the Effective Capacitance for the RC Interconnect in VDSM Technologies

Soroush Abbaspour and Massoud Pedram
Department of Electrical Engineering-Systems
University of Southern California, Los Angeles, CA 90089
{sabbaspo, pedram}@usc.edu

Abstract

In this paper, we present a new technique for calculating an effective capacitance of an RC interconnect line in very deep submicron design technologies. The calculation scheme guarantees that the effective capacitance model simultaneously matches both the 50% propagation delay and the 0-to-0.8V_{dd} output transition behavior of a standard cell driving an RC interconnect. Experimental results show that the new technique exhibits high accuracy (less than 5% error) and high efficiency (converges in two or at most three iterations). The paper also includes extensions to handle complex cells as drivers of the RC interconnect.

1. Introduction

Circuit delay in VLSI circuits consists of two components: the 50% propagation delay of the driving gates (known as the *gate propagation delays*) and the delays of electrical signals through the wires (known as the *interconnect propagation delays.*) Consider the circuit in Fig.1. The overall delay from input pin "A" of gate "Inv1" to input pin "C" of "Inv 2" can be written as the sum of a gate propagation delay from input pin "A" to the output pin "B" of "Inv1" and an interconnect propagation delay from output pin "B" to the input pin "C" of "Inv2."

$$Delay_{AC} = Delay_{AB} + Delay_{BC} \quad (1)$$

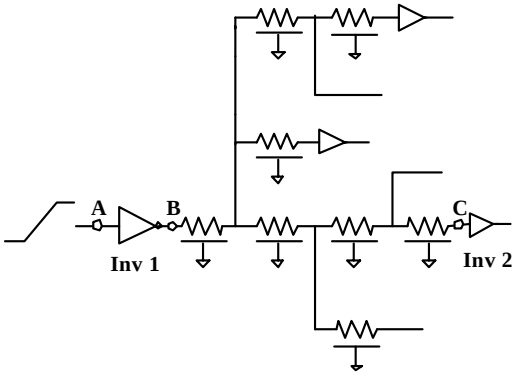

Figure 1: General delay model of interconnects which is driven by CMOS gates and driving another CMOS gates.

This work was supported by the National Science Foundation under grant number 9988441.

The gate propagation delay is in turn divided into two terms: the intrinsic gate delay and the (external) gate load delay. The intrinsic gate delay is due to the native characteristics of the CMOS devices (e.g., transistors) in the gates/cells (more precisely, it is equal to gate propagation delay under zero load condition.) The load delay captures the timing effect of the load on the gate propagation delay.

Fig. 2 shows a gate, which drives a purely capacitive load (C_{load}) where one of its inputs switches (with a signal transition time of T_{in}) and causes the output of the gate/cell to change. The gate propagation delay is a function of the input transition time and the output load. More precisely, the delay increases when C_{load} and/or T_{in} increase, therefore,

$$Gate\ Delay = f(T_{in}, C_{load}) \quad (2)$$

Two approaches for gate propagation delay computation are based on (1) delay tables, and (2) use of a Thevenin equivalent circuit composed of a voltage source and a resistance in series with the gate load. Although the first approach is currently in wide use especially in the ASIC design flow, the second approach promises to be more accurate when the load is not purely capacitive. This is because it directly captures the interaction between the load and the gate/cell structure. The resistance value in the Thevenin model is strongly dependent on the input slew and output load and requires output voltage fitting [2].

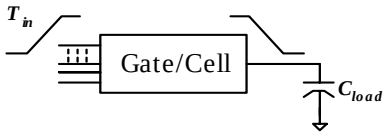

Figure 2: A gate/cell where its output switches when its input switches. The gate load delay is a function of both input slew and output load.

In commercial ASIC cell libraries, for each gate/cell in the library, typically, there is at least one pair of two-dimensional tables, which store the gate propagation delay and output rise/fall time as a function of the output capacitance and input slew. One sample delay table is shown in Fig. 3.

For intermediate values of input transition time and output load, piecewise linear approximation is used. The table is equivalent to empirical "k-factor' formulas for delay and output transition time. However, in VDSM technologies, we cannot neglect the effect of interconnect resistances and as shown in Fig. 1, the load is not purely capacitive. Using the sum of all load capacitances

as the capacitive load is the simplest pessimistic approximation [7]. A better approximation for the n'th order load (i.e., with n distributed capacitances) seen by the gate/cell is a second order RC-π model [3, 5].

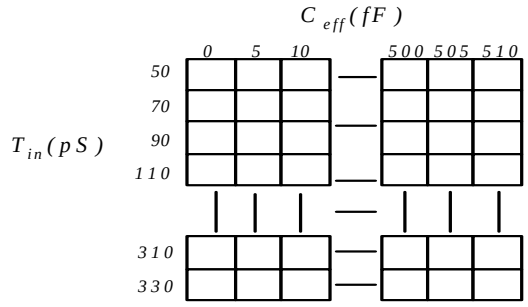

Figure 3: A two-dimensional delay table, one dimension is effective load and the other one is input transition time.

Equating the first, second and third moments of the admittance of the real load with the first, second and third moments of the RC-π load, we can find C_1, R_π, and C_2 in Fig. 4.

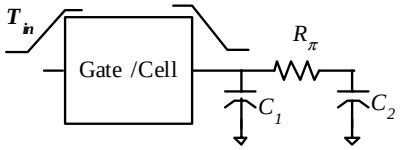

Figure 4: A gate/cell, which drives a RC-π calculated load.

$$Y_{in}(s) = A_1 s + A_2 s^2 + A_3 s^3 + ... \quad (3)$$

$$\hat{Y}_{in}(s) = (C_1 + C_2)s - R_\pi C_2^2 s^2 + R_\pi^2 C_2^3 s^3 + ... \quad (4)$$

Eq. (3) is the Taylor expansion of the real load around *s=0* and Eq. (4) is the Taylor expansion of the approximated load around *s=0*. Then,

$$C_1 = A_1 - \frac{A_2^2}{A_3} \qquad R_\pi = -\frac{A_3^2}{A_2^3} \qquad C_2 = \frac{A_2^2}{A_3} \quad (5)$$

It follows that for accurate delay calculation; we need to have a four-dimensional delay table, the four dimensions being input transition times, C_1, R_π, and C_2. However, this is very costly in terms of storage and computational requirements. Therefore, the "effective capacitance" approach was proposed in [4,8]. Effective capacitance approach uses the two-dimensional delay table; it approximates the *RC-π* model with an equivalent capacitance, which approximates the either 50% propagation delay or the transition time.

Given the problem described above, we propose an algorithm, which calculates an effective capacitance that results in a very good match both in terms of the output transition time and the 50% propagation delay, simultaneously. This paper is organized as follows: Section 2 reviews the previous work to compute effective capacitance. Section 3 describes our algorithm for cases of a simple driver. Section 4 extends the effective capacitance computation algorithm for complex gates. We provide the results in Section 5. The conclusions are presented in Section 6.

2. Prior Work

Many research results have been reported for calculating the interconnect propagation delay. These papers are simulation-based [6,7,9] or rely on analytical derivations [1,11]. Similarly and more recently, a number of research results have been published that focus on the loading effect of the RC wires on the gate propagation delay [8,9,12,13].

The effective capacitance is a function of two factors: (1) the output voltage waveform of the driving cell and (2) the load or more appropriately, the driving point admittance of interconnects. If two cells produce the same output waveform when the same load is applied, the two cells are equivalent in terms of calculating C_{eff} [12].

Consider the circuit in Fig. 4. If R_π goes to infinity, then the gate/cell will see only C_1 as its load. On the other hand if R_π goes to *0*, the gate/cell will see $C_{tot}=C_1+C_2$. Then the effective capacitance that the driver see, can be written as:

$$C_{eff} = C_1 + kC_2 \quad \text{where} \quad 0 \le k \le 1 \quad (6)$$

Using a table of circuit simulation results and a pair of two-dimensional delay tables (cf. Fig. 3), Macys *et al.* [12] obtain the effective capacitance. In their work, the effective capacitance is a function of the total capacitance in the RC-π model (C_{tot}), the gate output slew rate, and the Elmore delay [1] of the load. The authors approximate the RC-π load with an effective capacitance such that the output voltage waveforms of the driving cell passes through some critical voltages (e.g., 0 and $0.75V_{dd}$) at the same instances in time. They also normalize the four model parameters (output slew time and three π model parameters) to two parameters and use the table of circuit simulation results to find the effective capacitance by performing some iteration. However, their approach is not based on any analytical derivation. Furthermore, it requires a rather large number of iterations to achieve convergence. Finally, it is either inaccurate or very costly for complex gates.

In one extension to handle complex gate, Macys's approach can be modified where a complex gate is first converted into an equivalent inverter with an equivalent input slew time and then the delay tables are used to calculate the gate delay. This is obviously very inaccurate. In another extension, Macys's approach can be modified by generating a pair of two-dimensional delay tables for each unique combination of possible input transitions and slew rates. Obviously, this is very expensive for complex gate with say three of four inputs. In contrast, in our approach, we only need to generate four two-dimensional tables independent of the input combination and slew rate.

Using a two-piece output waveform, Qian *et al.* propose an effective capacitance calculation approach that approximates the output waveform for single-stage gates [10]. The authors calculate the effective capacitance by equating the currents at the gate output when using the driving-point admittance as the load and when using a single effective capacitor as the load. Average currents for both loads models are equated until the gate output

voltage reaches the 50% threshold. Qian's effective capacitance approach requires a large number of iterations (e.g., 5 to 10 iterations). It also involves empirical equations that assume fast input transitions and/or require extensive effort to generate characterization tables. Finally, it does not describe any method to handle complex gates.

Kahng and Muddu [11,13] propose a number of effective capacitance algorithms. In their latest approach [13], they state that by using the voltage of output pin of the gate/cell, they can find a non-iterative and fast method for calculating the effective capacitance that accurately matches the output waveforms in a range from $0.3V_{dd}$ to $0.6V_{dd}$. In fact, finding the output transition time (from the complex set of equations that these authors present) can be very costly. Furthermore, the driver resistance in their model is a function of the output load and input transition time and can vary greatly. However, the authors use a single value for the driver resistance corresponding to the case that the driver sees the total capacitances of load. This is a major source of error in their proposed approach.

In this paper, we propose an optimal effective capacitance algorithm for both speed and accuracy; such that this capacitance value approximates both the transition time from 0 to $0.8Vdd$ and 50% propagation delay simultaneously. We also propose a new algorithm to find 50% propagation delay and "0 to $0.8V_{dd}$" output transition time using effective capacitance for complex gates.

3. A New Algorithm for Effective Capacitance Calculation of RC Loads

We now propose a new algorithm that provides the effective capacitance in VDSM technologies, which approximates both the "0 to $0.8V_{dd}$" output transition time and the 50% gate propagation delay simultaneously.

For any combination of input transition time and output capacitive load, the gate can be replaced with a Thevenin equivalent circuit. This consists of an effective driver resistance, ramp input source and an intrinsic delay [2]. Therefore, the circuit in Fig. 4 can be approximated with the circuit in Fig. 5. Then R_d is a linear resistance, which can be written as:

$$R_d = \begin{cases} \frac{r_{dn}}{W_n} & for \quad NMOS \\ \frac{r_{dp}}{W_p} & for \quad PMOS \end{cases} \tag{7}$$

where r_{dp} and r_{dn} are both functions of input transition time and output capacitive load. r_{dp} and r_{dn} for some combinations of C_{eff} and T_R are shown in table 1. To find the effective capacitance, what we need is to approximate the *RC-π* load with an effective capacitive load (Fig. 6) such that the voltage at the output pin of the gates in both circuits in Figs. 5 and 6 follow each other during output transition.

Table 1: r_{dn} and r_{dp} for some combinations of C_{eff} and T_R

T_R	C_{load}	r_{dn}	r_{dp}
200pS	200fF	23.9K	49.2K
200pS	500fF	14.0K	38.9K
400pS	200fF	34.1K	69.2K
300pS	500fF	15.9K	39.1K
100pS	300fF	12.8K	42.6K

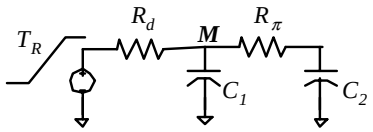

Figure 5: The equivalent circuit of a gate/cell driving an *RC-π* load.

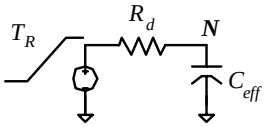

Figure 6: Using C_{eff} instead of load such that the voltage of *M* and *N* in figure 5 and this figure, follow each other during output transition.

The voltages of *M* and *N* in Laplacian domain can be written as:

$$V_M(s) = \frac{sR_\pi C_2 + 1}{s^2 R_d R_\pi C_1 C_2 + s[R_d(C_1 + C_2) + R_\pi C_2] + 1} V_{in}(s) \tag{8}$$

$$V_N(s) = \frac{1}{sR_d C_{eff} + 1} V_{in}(s) \tag{9}$$

where;

$$V_{in}(s) = \frac{1 - e^{-T_R s}}{s^2} \frac{V_{dd}}{T_R} \tag{10}$$

Doing inverse Laplace transforms, we can find $V_M(t)$ and $V_N(t)$ in time domain as follows:

$$V_M(t) = \begin{cases} \frac{V_{dd}}{T_R}\left(t - B + Ae^{-\alpha t} Cosh(\omega t + \phi)\right) & 0 \le t \le T_R \\ \frac{V_{dd}}{T_R}\left(T_R + A' e^{-\alpha t} Cosh(\omega t + \phi')\right) & T_R < t \end{cases} \tag{11}$$

$$V_N(t) = \begin{cases} \frac{V_{dd}}{T_R}\left(t - R_d C_{eff} + R_d C_{eff} e^{-\frac{t}{R_d C_{eff}}}\right) & 0 \le t \le T_R \\ \frac{V_{dd}}{T_R}\left(T_R + R_d C_{eff}(1 - e^{\frac{T_R}{R_d C_{eff}}}) e^{-\frac{t}{R_d C_{eff}}}\right) & T_R < t \end{cases} \tag{12}$$

where;

$$\phi = tgh^{-1}\left(\frac{R_d(C_1 + C_2)^2 + R_\pi C_2(C_2 - C_1)}{(C_1 + C_2)\sqrt{R_d^2(C_1 + C_2)^2 + 2R_d R_\pi C_2(C_2 - C_1) + R_\pi^2 C_2^2}}\right)$$

$$A=\frac{R_d(C_1+C_2)}{Cosh(\phi)},\ B=R_d(C_1+C_2),\ \alpha=\frac{R_d(C_1+C_2)+R_\pi C_2}{2R_dR_\pi C_1C_2} \quad (13)$$

$$\omega=\frac{\sqrt{{R_d}^2(C_1+C_2)^2+{R_\pi}^2{C_2}^2+2R_dR_\pi C_2(C_2-C_1)}}{2R_dR_\pi C_1C_2}$$

An accurate approximation for C_{eff} may be obtained by minimizing the error between voltages of V_M *(t)* and V_N *(t)* from $0.2V_{dd}$ to $0.8V_{dd}$. More precisely, we have:

$$\frac{\partial}{\partial Ceff}\int_{0.2Vdd}^{0.8Vdd}(V_M-V_N)^2\,dV=2\int_{0.2Vdd}^{0.8Vdd}(V_M-V_N)\frac{\partial V_N}{\partial Ceff}\,dV=0 \quad (14)$$

which is too complex to solve analytically. From Eq. (11) and (12) it is obvious that V_M *(t)* and V_N *(t)* are both equal at t=0. In Eq. (11), α and ω are the key parameters that determine the behavior of the output voltage. We note that these parameters do not change before and after $t=T_R$. Therefore, if we match V_M *(t)* and V_N *(t)* at some critical time instances before $t=T_R$, (say $t_{0\text{-}0.5Vdd}$) and calculate an effective capacitance on that basis, then this effective capacitance will be a good approximation of α and ω even for t > T_R because V_M *(t)* and V_N *(t)* will continue to be very similar even after $t_{0\text{-}0.5Vdd}$. Our experimental results have shown that the resulting effective capacitance causes V_M *(t)* and V_N *(t)* to closely follow one another at least until $0.8V_{dd}$. Consequently, we end up with an iterative equation as follows:

$$C_{eff}=f(C_{eff},R_d,t)=\frac{-Ae^{-\alpha t}Cosh(\omega t+\phi)+B}{R_d(1-e^{-\frac{t}{R_dC_{eff}}})}$$

$$=(C_1+C_2)\frac{1-e^{-\alpha t}\frac{Cosh(\omega t+\phi)}{Cosh(\phi)}}{(1-e^{-\frac{t}{R_dC_{eff}}})} \quad (15)$$

To solve the above equation, we need to update the characterization table. Therefore, the new characterization table should have r_{dn}/r_{dp}, 50% propagation time, "0-0.5V_{dd}" output transition time, and "0-0.8V_{dd}" output transition time for each combination of T_R and C_{eff}.

For fast convergence of the nonlinear iterative equation, we need a good initial value of effective capacitance. Effective capacitance as a function of R_π for two different values of T_{in} and R_d is shown in Fig. 7.

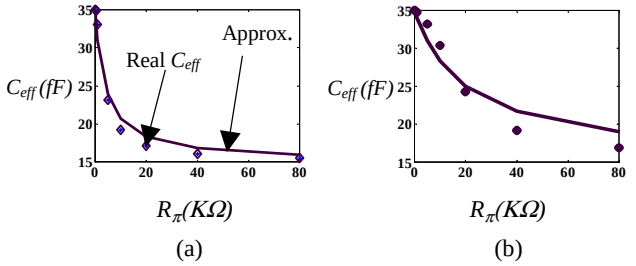

Figure 7: Effective capacitance for two different cases of T_{in} and driver sizes (C_1=15*fF*, C_2=20*fF*)(a) *driver size=500λ,T_R=100pS* (b) *driver size=100λ,T_R= 200pS.*

k in Eq. (6) is a function of driver strength and resistance of line. If the driver is weak (R_d is large), the driver would see the entire capacitive load (C_1+C_2) and if driver is very strong, then it would see only C_1, since the output would change very fast and it only sees the near end capacitance. Also, if R_π goes to infinity, the driver would see only the near end capacitance (C_1) and if R_π goes to zero, then the driver would see all capacitive load (C_{tot}=C_1+C_2). From the figures and the above explanations, we can say that the initial value, approximately, can be:

$$C_{eff}=C_1+\frac{R_d}{R_d+R_\pi}C_2 \quad (16)$$

which is a very good approximation as an initial guess. R_d is the driver resistance when it sees whole capacitive load (C_{eff}=C_1+C_2). Then, using iterative equation, which is shown in Eq. (15), we can find the accurate and fast C_{eff} for both 50% propagation delay and 0-0.8V_{dd} output transition time.

Our algorithm for calculating C_{eff} is as follows. Given the following information for a particular timing path of a cell: the input slew time, T_R, the π-load model parameters,(C_1,R_π,C_2), and the updated characterized cell output slew model, we perform the following steps:

1. Calculate an initial value of C_{eff} from Eq. (16).
2. Obtain $t_{0\text{-}50\%}$ from the characterization table based on values of C_{eff} and T_R.
3. Obtain R_d from the characterization table based on values of C_{eff} and T_R.
4. Compute a new value of C_{eff} from Eq. (15)
5. Find new $t_{0\text{-}50\%}$ based on the new C_{eff} and given T_R.
6. Compare the previous value of $t_{0\text{-}50\%}$ with the new $t_{0\text{-}50\%}$ in step 5.
7. If not within acceptable tolerance, then return to step 3 until $t_{0\text{-}50\%}$ converges.
8. Report $t_{50\%\ propagation\ delay}$ and $t_{0\text{-}80\%}$ from the table.

Using this algorithm, one can obtain an effective capacitance that accurately approximates the 50% propagation delay as well as output transition time from 0 to 0.8V_{dd}.

4. Extension to Complex Gates

To use the above algorithm for complex CMOS gates, one must accurately calculate the output resistance of a channel-connected gate. Due to the body effect, it is difficult to compute the exact gate output resistance in channel-connected gates. Let r_{dn} denote the driver resistance of a 1μm wide transistor of length L. By using a first order model (usually taught in textbooks on VLSI design), the driver output resistance without consideration of the body effect is calculated by assuming that the "on" resistance of any transistor with a W/L ratio is simply proportional to *L/W* of that transistor. For instance, the output resistance of a 2-input NAND gate for a falling output transition (with a *W/L* ratio for each n-type transistor) is simply $2r_{dn}/W$.

Accounting for the body effect in channel-connected transistors in stacks, Kahng *et al*. [13] propose an approach with which one can approximate the output resistance more precisely.

Performing some simulations, they find a body effect coefficient, which is dependent on the number of transistors in the stack and the parameters of the process technology. For instance, the output resistance of a 2-input NAND gate for a falling output transition (with *W/L* ratios for each n-type transistor) is $2.86r_{dn}/W$ in 0.25µm process technology. In Kahng's approach, the coefficient is constant for a gate and is independent of the input combination (i.e., which input switches). Consider four input transition cases of a *3*-input NAND gate as shown in Fig. 8. According to both the first order model and Kahng's model, all of these cases result in the same gate output resistance, which is obviously quite inaccurate.

We propose an output resistance calculation equation that uses *m+1* coefficients for *m*-series-connected transistors in the *N* or *P* type stack. More precisely,

$$W_{Neff} = K_n \frac{1}{K_{bn(1)}\frac{L_1}{W_1} + K_{bn(2)}\frac{L_2}{W_2} + ... + K_{bn(m)}\frac{L_m}{W_m}}$$
$$W_{Peff} = K_p \frac{1}{K_{bp(1)}\frac{L_1}{W_1} + K_{bp(2)}\frac{L_2}{W_2} + ... + K_{bp(m)}\frac{L_m}{W_m}} \quad (17)$$

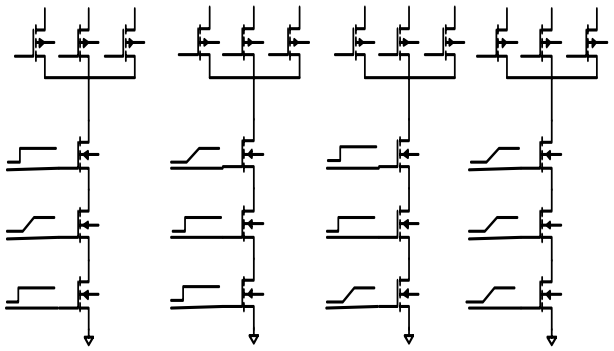

Figure 8: Gate output driver resistance will change according to input pattern.

To find the effective width of channel-connected transistors, we set $K_{bp(i)}$ and $K_{bn(i)}$ in Eq. (17) equal to 1 if the corresponding transistor input does not switch; otherwise, we use tables 2 and 3 to provide the required coefficient values in Eq. (17). K_{bp}'s and K_{bn}'s are obtained by HSPICE simulation for each process technology. For example, for a 0.1µm technology, these values are provided in tables 2 and 3.

5. Experimental Results

We performed simulations for a number of different circuits in 0.1µm CMOS technology and report results. For our experiments we considered two inverters connected in series. We fixed the size of the first inverter at 60/30 µm and applied an input transition time of 300*pS* at the its input. We put an RC-π load for the second inverter. We provide the results for comparison between HSPICE results and results of our algorithm for the output voltage of the second inverter in table 4. We also simulate a 3-input NAND gate to validate our algorithm proposed for complex gates. For each input of the 3-input NAND gate, we provided the same circuit as we mentioned for Inverter gate. For Table 5, we only applied the switching input to the top transistor in stack. Table 6 shows the results when we apply a transitioning input waveform with the same slew times to all three inputs of the 3-input NAND gate. All transistors of the same type have the same width as indicated in the first column of the tables. Results show that our approach is both accurate and fast for VDSM technologies within 5% for both 50% propagation delay and the "0- $0.8V_{dd}$" output transition time with at most 3 iterations.

	K_n	$K_{bn(1)}$	$K_{bn(2)}$	$K_{bn(3)}$	$K_{bn(4)}$
1 Tran. In Stack	2.00	1.00			
2 Tran. In Stack	2.18	1.00	1.23		
3 Tran. In Stack	2.24	1.00	1.28	1.29	
4 Tran. In Stack	2.31	1.00	1.3	1.32	1.36

Table 2: Body effect coefficients for 0.1µm technology for NMOS transistors

	K_p	$K_{bp(1)}$	$K_{bp(2)}$	$K_{bp(3)}$	$K_{bp(4)}$
1 Tran. In Stack	2.00	1.00			
2 Tran. In Stack	2.26	1.00	1.35		
3 Tran. In Stack	2.36	1.00	1.37	1.39	
4 Tran. In Stack	2.45	1.00	1.39	1.41	1.48

Table 3: Body effect coefficients for 0.1µm technology for PMOS transistors

6. Conclusion

An increase in interconnect resistance in VDSM technologies has two important results: 1) the interconnect RC-delay resistance portion of a timing path increases and 2) the interconnect resistance reduces the cell delay via shielding the far end capacitances. Gate load delay calculation requires accuracy, and using delay tables is essential for accurate delay calculation for given capacitive load and input transition time. To use the delay tables, what we need is to approximate the load with an effective capacitance that is equivalent to the real load in terms of its 50% propagation delay and output transition time. In this paper we presented a new efficient algorithm which ends up in an effective capacitance for both 50% propagation delay and "0- $0.8V_{dd}$" output transition time, simultaneously. In addition, we proposed a new algorithm for complex gates, which calculates the gates delay for different input patterns.

7. References

[1] W. C. Elmore, "The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers," *Journal of Applied Physics*, 19, Jan. 1948, pp. 55-63.

[2] J. K. Ousterhout, "A Switch-level Timing Verifier for Digital MOS VLSI," *IEEE Trans. on Computer Aided Design of VLSI Circuits and Systems*, vol. 4 (1985), pp. 336-349.

[3] P. R. O'Brien and T. L. Savarino, "Modeling the Driving-Point Characteristic of Resistive Interconnect for Accurate Delay Estimation," *Proc. of IEEE Int'l Conf. on Computer Aided Design*, 1989, pp. 512-515.

[4] S. P. McCormick, Modeling and Simulation of VLSI Interconnections with Moments, PhD Thesis, MIT, June 1989.

[5] P. R. O'Brien and T. L. Savarino, "Efficient On-Chip Delay Estimation for Leaky Models of Multiple-Source Nets," *Proc. of IEEE Custom Integrated Circuits Conf.*, 1990, pp. 9.6.1-9.6.4.

[6] L. T. Pillage and R. A. Rohrer, "Asymptotic Waveform Evaluation for Timing Analysis," *IEEE Trans. on Computer Aided Design of VLSI Circuits and Systems*, vol. 9 (1990), pp. 352-366.

[7] C. L. Ratzlaff, N. Gopal, and L. T. Pillage, "RICE: Rapid Interconnect Circuit Evaluator," *Proc. of 28th ACM/IEEE Design Automation Conf.*, June 1991, pp. 555-560.

[8] C. Ratzlaff, S. Pullela, and L. Pillage, "Modeling the RC Interconnect effects in a Hierarchical Timing Analyzer," *Proc. of IEEE Custom Integrated Circuits Conference*, May 1992, pp. 15.6.1-15.6.4.

[9] M. Sriram and S. M. Kang, "Fast Approximation of the Transient Response of Lossy Transmission Line Trees," *Proc. of 30th ACM/IEEE Design Automation Conf.*, June 1993, pp. 691-696.

[10] J. Qian, S. Pullela, and L. Pillage, "Modeling the "Effective Capacitance" for the RC Interconnect of CMOS gates," *IEEE Trans. on Computer Aided Design of VLSI Circuits and Systems*, vol. 13 (1994), pp. 1526-1535.

[11] A. B. Kahng and S. Muddu,' "Accurate Analytical Delay Models for VLSI Interconnects," *Proc. of IEEE International Symposium on Circuits and Systems*, May 1996.

[12] R. Macys, S. McCormick, "A New Algorithm for Computing the "Effective Capacitance" in Deep Sub-micron Circuits," *Proc. of IEEE Custom Integrated Circuits Conference*, June 1998, pp. 313-316.

[13] A. B. Kahng and S. Muddu,' "Improved Effective Capacitance Computations for Use in Logic and Layout Optimization," *Proc. of the 12th International Conference on VLSI Design*, 1999, Pages: 578 -582.

Table 4: Comparison between our approach with HSPICE for 50% propagation delay and 0-0.8Vdd for a simple driver in 0.1 µm technology

Inverter Size (Wp/Wn) µm	C_1(pF)/R_π(Ω)/C_2(pF)	HSPICE 50% delay (pS)	Estimated 50% delay (pS)	Error	HSPICE 80% delay (pS)	Estimated 80% delay (pS)	Error	Number of Iterations
10/5	0.05/410/0.15	66.1	69.0	4.5%	142.3	140.6	1.2%	3
40/20	0.1/290/0.25	39.3	41.0	4.3%	95.3	97.4	2.2%	3
40/20	0.5/810/0.7	74.0	76.4	3.2%	136.2	134.5	1.3%	2
30/15	0.4/1000/0.8	76.3	79.4	4.1 %	142.5	138.5	2.8%	1
100/50	0.9/300/1.4	62.7	65.1	3.8%	121.0	123.1	1.7%	2
Avg. Error	-------	-------	-------	4.0%	-------	-------	1.8%	-------

Table 5: Comparison between our approach with HSPICE for 50% propagation delay and 0-0.8Vdd for 3 input NAND gates in 0.1 µm technology (when we apply a switching input to the topmost transistor in the stack)

3-input NAND (Wp/Wn) µm	C_1(pF)/R_π(Ω)/C_2(pF)	HSPICE 50% delay (pS)	Estimated 50% delay (pS)	Error	HSPICE 80% delay (pS)	Estimated 80% delay (ps)	Error	Number of Iterations
20/60	0.4/1000/0.8	34.7p	36.4p	4.9%	42.1p	43.2p	2.6%	2
40/120	0.5/510/1.2	26.4p	27.1p	2.7%	78.1p	79.5p	1.8%	2
Avg. Error	-------	-------	-------	3.6%	-------	-------	2.2%	-------

Table 6: Comparison of our approach with HSPICE for 50% propagation delay and 0-0.8Vdd for 3 input NAND gate in 0.1 µm technology (when we simultaneously apply the same switching input to all inputs)

3-input NAND (Wp/Wn) µm	C_1(pF)/R_π(Ω)/C_2(pF)	HSPICE 50% delay (pS)	Estimated 50% delay (pS)	Error	HSPICE 80% delay (pS)	Estimated 80% delay (pS)	Error	Number of Iterations
20/60	0.4/1000/0.8	64.7p	67p	3.6%	64.4p	64.8p	0.6%	3
40/120	0.5/510/1.2	54.1p	55.5p	2.6%	83.5p	84.5p	1.2%	2
Avg. Error	-------	-------	-------	3.1%	-------	-------	0.9%	-------