ASPDAC Conference 2003

Effective Capacitance for the RC Interconnect in VDSM Technologies

Soroush Abbaspour and Massoud Pedram

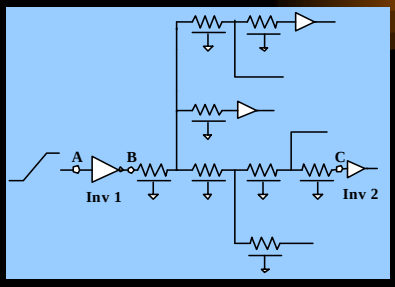
Department of Electrical Engineering-Systems

University of Southern California

Outline

- **Background**
- **Prior Work**
- **A New Algorithm for Calculating the Effective Capacitance**
- **Experimental Results**
- **Conclusion**

Circuit Delay

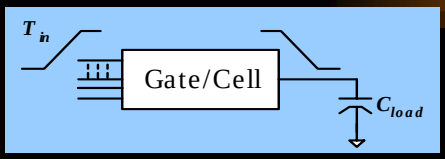


$$Delay_{AC} = Delay_{AB} + Delay_{BC}$$

The circuit delay in VLSI circuits consists of two components:

1. **the 50% propagation delay of the driving gates (known as the *gate propagation delay*)**
2. **the delay of electrical signals through the wires (known as the *interconnect propagation delay*)**

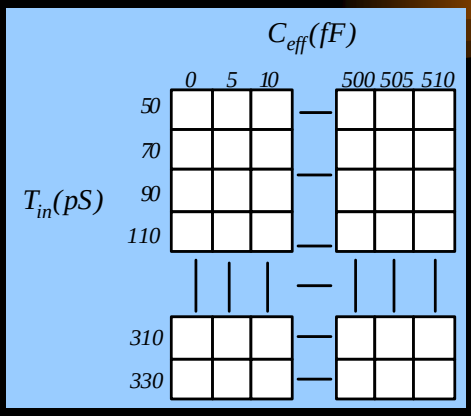
Gate Delay



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

The gate load delay is a function of both input slew and the output load

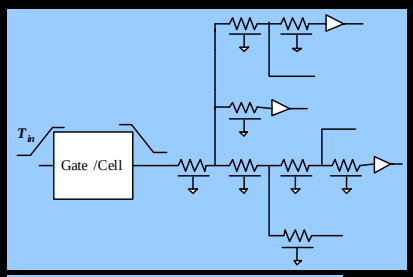
Library-based Delay Model

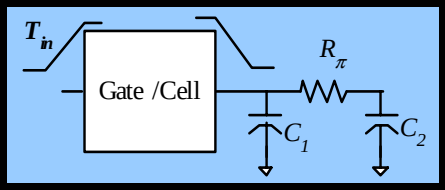


A pair of two-dimensional delay tables, one for providing the gate delay, the other for the output rise/fall time as a function of the effective load and the input transition time

Second RC-p Model for Load

Using Taylor Expansion around $s = 0$



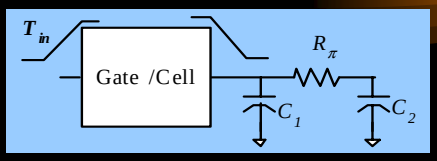


$$Y_{in}(s) = A_1 s + A_2 s^2 + A_3 s^3 + ...$$

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

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

Second RC-p Model (Cont'd)

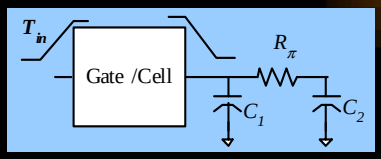


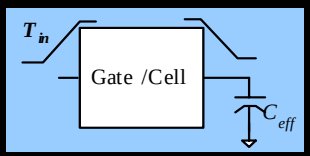
$$Gate \quad Delay = f(T_{in}, C_1, R_\pi, C_2)$$

Therefore, it is required to create a four-dimensional table to achieve high accuracy

This is however costly in terms of memory space and computational requirements

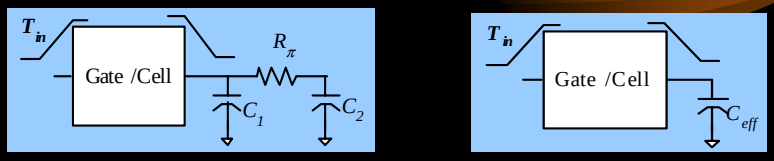
Effective Capacitance Approach





The "Effective Capacitance" approach attempts to find a single capacitance value that can be replaced instead of the RC-π load such that both circuits behave similar during transition

Effective Capacitance (Cont'd)



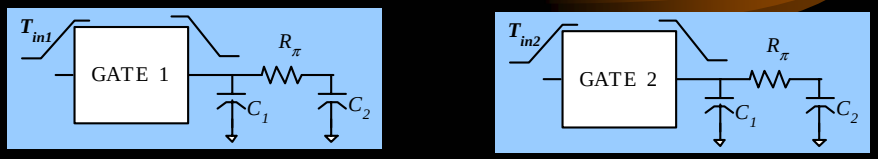
$$C_{eff} = C_1 + kC_2 \qquad 0<k<1$$

Because of the shielding effect of the interconnect resistance , the driver will only "see" a portion of the far-end capacitance C_2

$$R_\pi \rightarrow 0 \Rightarrow k = 1$$

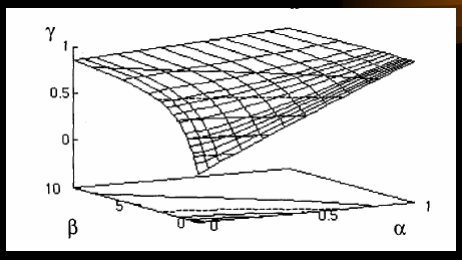
$$R_\pi \rightarrow \infty \Rightarrow k = 0$$

Prior Work - Macys's Approach



Assumption: If two circuits have the same loads and output transition times, then their effective capacitance are the same. In other words, the effective capacitance is only a function of the output transition time and the load

Macys's Approach (Cont'd)



Normalized Effective Capacitance Function

$$\alpha = \frac{C_1}{C_1 + C_2} \qquad \gamma = \frac{C_{eff}}{C_1 + C_2} \qquad 0 \le \alpha \le \gamma \le 1$$

$$\beta = \frac{T_{out}}{R_\pi C_2}$$

Macys's Approach (Cont'd)

1. Compute α from C_1 and C_2
2. Choose an initial value for C_{eff}
3. Compute t_{output} for the given C_{eff} and T_{in}
4. Compute β
5. Compute γ from α and β
6. Find new C_{eff}
7. Go to step 3 until C_{eff} converges

Prior Work - Qian's Approach

Calculate the effective capacitance by equating the currents at the gate output by using:

(a) the driving-point admittance as the load

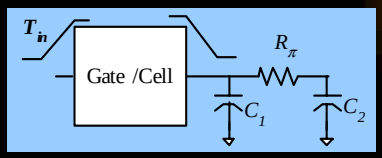
(b) using a single effective capacitance as the load

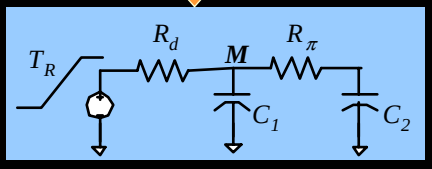
Average currents for both loads models are equated until the gate output voltage reaches the 50% threshold

Outline

- Background
- Prior Work
- A New Algorithm for Calculating the Effective Capacitance
- Experimental Results
- Conclusion

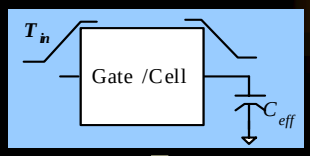
A New Effective Capacitance Algorithm

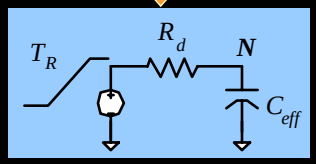




$$V_M(t) = \begin{cases} \frac{V_{dd}}{T_R}\left(t - B + Ae^{-\alpha t}Cosh(\omega t + \phi)\right) & 0 \le t \le T_R \\ \frac{V_{dd}}{T_R}\left(T_R + A'e^{-\alpha t}Cosh(\omega t + \phi')\right) & T_R < t \end{cases}$$

New Algorithm (Cont'd)





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

Eff_Cap Equation

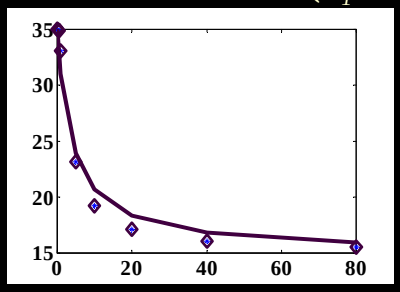
$$C_{eff} = (C_1 + C_2)\frac{1 - e^{-\alpha t}\dfrac{Cosh(\omega t + \phi)}{Cosh(\phi)}}{(1 - e^{-\frac{t}{R_d C_{eff}}})}$$

This is an Non-Linear Iterative Equation

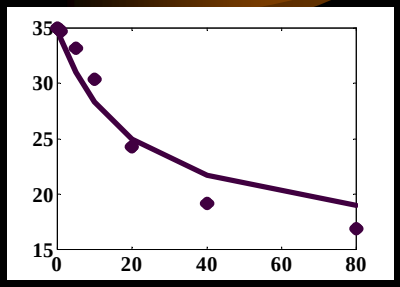
A good initial value for C_{eff} can speed up the procedure to find the answer

Initial Guess

(C_1=15*fF*, C_2=20*fF*)



C_{eff} (*fF*)



R_π (KΩ)

R_π (KΩ)

(a) *driver size=500λ,T_R=100pS*

(b) *driver size=100λ,T_R= 200pS*

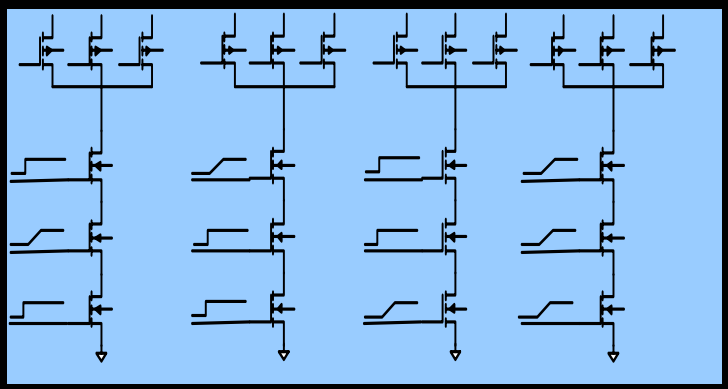
$$C_{eff} = C_1 + \frac{R_d}{R_d + R_\pi} C_2$$

Iterative Procedure to Calculate C_{eff}

1. Start with the initial guess for C_{eff}
2. Obtain $t_{0\text{-}50\%}$ based on values of C_{eff} and T_R
3. Obtain R_d based on values of C_{eff} and T_R
4. Compute a new value of C_{eff} from the Eff_Cap equation
5. Find new $t_{0\text{-}50\%}$ based on the new C_{eff} and given TR
6. Compare the values of $t_{0\text{-}50\%}$ from step 5
7. If not within acceptable tolerance, then return to step 3 until $t_{0\text{-}50\%}$ converges
8. Report $t_{50\%\ \text{propagation delay}}$ and $t_{0\text{-}80\%}$ from the table

Extension to Complex Gates

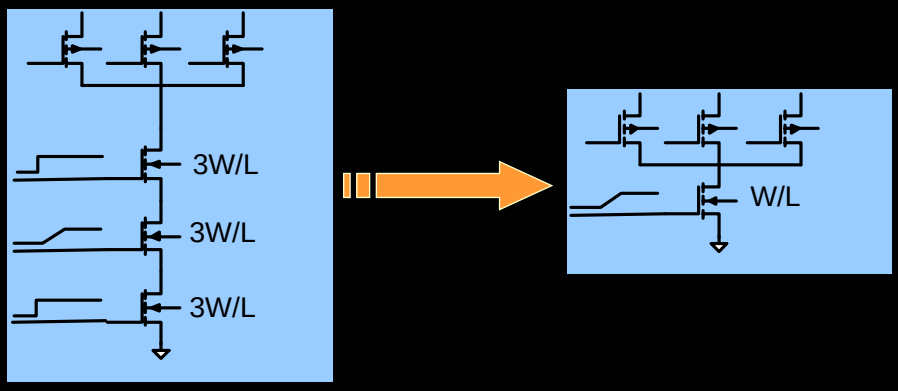
To extend the previous algorithm to complex gates, we only need to compute the value of R_d



The gate output driver resistance changes as a function of the applied input waveforms

Complex Gates (Cont'd)

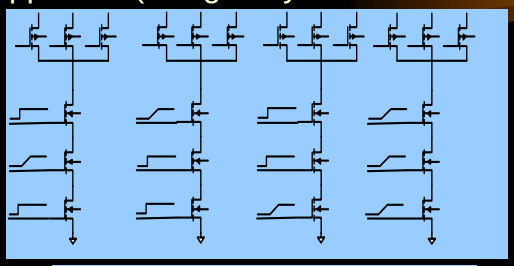
First Order (simple) approximation:



Due to the body effect, this value is over-estimated

Extension to Complex Gates,Cont'd

Our approach (using body effect coefficients):



$$W_{Neff} = K_n \frac{1}{K_{bn(1)}\frac{L_1}{W_1} + K_{bn(2)}\frac{L_2}{W_2} + \ldots + K_{bn(m)}\frac{L_m}{W_m}}$$

$$W_{Peff} = K_p \frac{1}{K_{bp(1)}\frac{L_1}{W_1} + K_{bp(2)}\frac{L_2}{W_2} + \ldots + K_{bp(m)}\frac{L_m}{W_m}}$$

K_i is set to 1 if the corresponding input is not switching; Read K_i's from a lookup table if the correspondent input is switching

Experimental Results

Inverter Size (Wp/Wn) mm	C_1(pF)/R_p(W)/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 %	-------

Experimental Results

3-input NAND (Wp/Wn)	C_1(pF)/R_p(W)/C_2(pF)	HSPICE 50% delay	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%	-------

The topmost transistor in the stack is switching

3-input NAND (Wp/Wn)	C_1(pF)/R_p(W)/C_2(pF)	HSPICE 50% delay	Estimated 50% delay (pS)	Error	HSPICE 80% delay (pS)	Estimated 80% delay	Error	Number of Iterat.
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%	-------

All three transistors in the stack are switching