Robust Elmore Delay Models Suitable for Full Chip Timing Verification of a 600MHz CMOS Microprocessor

Nevine Nassif
Digital Equipment Corportation, Hudson MA
nevine.nassif@digital.com

Madhav P. Desai
Indian Institute of Technology, Mumbai, India
madhav@ee.iitb.ernet.in

Dale H. Hall
Digital Equipment Corportation, Hudson MA
dale.hall@digital.com

Abstract

In this paper we introduce a method for computing the Elmore delay of MOS circuits which relies on a model of the capacitance of MOS devices and a model of the Elmore delay of individual MOS devices. The resistance of a device is not explicitly modelled. The Elmore models are used to compute the Elmore delay and the 50% point delay of CMOS circuits in a static timing verifier. Elmore delays computed with these models fall within 10% of SPICE and can be computed thousands of times faster than if computed using SPICE. These models were used to verify critical paths during the design of a 600MHz microprocessor.

1 Introduction

The static timing verifier developed and used to verify the 21264 Alpha microprocessor [3] uses a family of delay models to compute the delay through MOS circuits. Simple complementary structures and some latches are modeled with precharacterized delays. The Elmore delay is one of the simplest delay models used in the static timing verifier [4] and we use it to compute the delays across complex complementary structures and arbitrary non-complementary structures. We need to use the Elmore delay because many of the structures used in the Alpha microprocessor design are arbitrary complementary and noncomplementary structures and it is impractical to characterize the delay for each one of these. The static timing verifier uses exhaustive path tracing techniques to verify every path in the microprocessor design and to identify critical paths (slow paths which violate setup constraints) and race paths (fast paths which violate hold constraints). Static timing verification is an important and necessary part of the design of the 21264 Alpha CPU. Because of the size (more than 15M transistors) and speed (600MHz) of this CPU, it is important for us to be able to filter out as many non-violating paths as possible before having to verify them or analyze them with more time consuming algorithms [1]. It was, therefore, necessary for us to radically improve the accuracy of our simple delay models before resorting to more complicated models or simulation techniques. In addition, because we use the Elmore delay model to compute the maximum delay through a MOS circuit, we had to ensure that the delays were always pessimistic. In other words, the delays computed using the Elmore delay model have to always be greater than those computed using a circuit simulator such as SPICE[7].

The Elmore delay which was defined [2] as

$$\tau = \int_0^\infty t\frac{dv}{dt}dt \tag{1}$$

has been used extensively as a measure of delay for RC interconnect and for MOS circuits. Given an RC tree, the Elmore delay at an output node i can be computed using the following equation [9]:

$$T_{D_i} = \sum_k R_{ki}C_k \tag{2}$$

where R_{ki} is defined as the resistance which the two paths from the input to node i and the input to node k have in common. Equation 2 is also used to compute the Elmore delay of MOS circuits [8]. The effective resistance of each device and the capacitive load seen by each device are used to approximate a MOS circuit by an RC circuit and the Elmore delay is computed. More efficient and general algorithms for computing the Elmore delay have also been developed [5][6].

The resistance and capacitance models which are used to approximate the MOS circuit are a great source of error in computing the Elmore delay. We decided to focus on these models in order to improve the accuracy of the Elmore delay in our timing verifier. Our goal was to use the Elmore delay model to compute 50% point delays which are always greater than SPICE and within 20%.

We began our modelling effort by attempting to define an effective resistance. Our objective was to write the resistance as a function of device size, the slope of the input signal on the gate, and the capacitive load. We wanted to model the resistance using a simple equation in order to make the computation of the resistance as fast as possible. In other words, we wanted to stay away from table lookup algorithms or complicated models.

In our quest for a simple model for resistance we moved away from Equation 2 and instead focused on the voltage waveform whose time constant is the Elmore delay. We realized that we should not even bother modelling the effective resistance and instead model the Elmore delay of a device as a function of device size, slope of the input signal on the

35th Design Automation Conference ®

1-58113-049-x-98/0006/$3.50

DAC98 - 06/98 San Francisco, CA USA

gate, and capacitive load. Since we were interested in using the Elmore delay only as an estimate for the maximum delay of our circuit, we also wanted to ensure that the Elmore delay calculated using the Elmore models was always greater than or equal to the Elmore delay computed using SPICE. The equations we derive are in fact very simple and accurate to within 5% of SPICE. As a result, we can compute the Elmore delay to within 10% of SPICE for most of our circuits.

In Section 2 we review the Elmore delay, describe the characterization of the Elmore delay and introduce the resulting model. In Section 3 we describe how we use the model for the Elmore delay within our timing verifier and how we use it to compute maximum 50% point delays in MOS circuits. In Section 4 we present results comparing the Elmore delay computed with the help of precharacterized delays and the Elmore delay computed using SPICE. We also compare the 50% point delay computed with Elmore delay model to the 50% point delay computed using SPICE.

2 A Model for the Elmore Delay

2.1 The Elmore Delay

The Elmore delay was introduced half a century ago [2] as a measure of delay. The Elmore delay can be used to model the waveforms in an RC network. If the node voltage at a node i in the RC circuit is $v_i(t), 0 \leq t < \infty$, then the Elmore delay model approximates $v_i(t)$ as:

$$v_i^{elmore}(t) = v_i(\infty) + (v_i(0) - v_i(\infty))\, e^{-t/\tau}. \qquad (3)$$

The time constant τ, which is also the Elmore delay, is chosen so that the area under the curve $v_i(\infty) - v_i^{elmore}(t)$ is the same as the area under the curve $v_i(\infty) - v_i(t)$. In other words:

$$\tau = \frac{\int_0^\infty (v_i(\infty) - v_i(t))dt}{v_i(\infty) - v_i(0)} \qquad (4)$$

See Shi [10] for details.

2.2 Characterizing the Elmore Delay

For a falling waveform at the output of a MOS circuit where $v_i(\infty) = 0$ and $v_i(0) = V_{DD}$, the Elmore delay reduces to:

$$\tau = \frac{\int_0^\infty v_i(t)dt}{V_{DD}} \qquad (5)$$

Similarly, for a rising waveform where $v_i(0) = 0$ and $v_i(\infty) = V_{DD}$, the Elmore delay reduces to:

$$\tau = \int_0^\infty dt - \frac{\int_0^\infty v_i(t)dt}{V_{DD}} \qquad (6)$$

From both Equations 5 and 6 we can see that the quantity that must be characterized in order to model the Elmore delay is

$$\int_0^\infty v_i(t)dt \qquad (7)$$

Let us consider the discharge path shown in Figure 1. Let $v_{DS}(t)$ represent the source-drain voltage across an NMOS device for $t > 0$. The Elmore delay at the output node y is computed by "summing up" the Elmore delay from V_{SS} to node x and the Elmore delay from node x to node y:

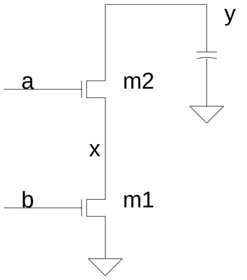

Figure 1: Discharge path through $m1$ and $m2$

$$\begin{aligned}\tau &= \frac{1}{V_{DD}} \int_0^\infty v_y(t)dt \\ &= \frac{1}{V_{DD}} \left[\int_0^\infty v_{DS_{m1}}(t)dt + \int_0^\infty v_{DS_{m2}}(t)dt\right] \\ &= \tau_{m1} + \tau_{m2} \qquad (8)\end{aligned}$$

Therefore if we can model the Elmore delay for a single MOS device, we can use the model to approximate the Elmore delay for a discharge path in a MOS circuit.

In order to get accurate models we used SPICE to simulate several configurations of PMOS and NMOS devices. These configurations which are shown in Figure 2, include 1-high, 2-high and 3-high stacks, where one device is turning on (i.e the gate voltage is transitioning) and the remaining devices are already on. We found it necessary to model each case separately and to differentiate between:

1. NMOS (PMOS) devices whose source was tied to V_{SS} (V_{DD}) and NMOS (PMOS) devices whose source was tied to the drain of another NMOS (PMOS) device;
2. NMOS (PMOS) devices whose drain is tied to the drain of a PMOS (NMOS) device and NMOS (PMOS) devices whose drain is tied to the source of an NMOS (PMOS) device;
3. NMOS (PMOS) devices whose source and drain are tied to NMOS (PMOS) devices.
4. NMOS (PMOS) devices for which the gate voltage is high (low) and NMOS (PMOS) devices for which the gate voltage is rising (falling).

For each simulation we varied the load seen at the output node and the sizes of the devices over the ranges allowed by the design rules imposed on the circuit designers. For each simulation of a MOS device we computed the following two quantities from the current and voltage waveforms:

$$\tau_m V_{DD} = \int_0^\infty v_{DS}(t)dt \qquad (9)$$

and

$$q = \int_0^\infty i_D(t)dt \qquad (10)$$

where $i_D(t)$ represents the drain current for $t > 0$ and q is the charge discharged by the device.

In Figure 3 we show a plot of $\tau_m V_{DD}$ as a function of q for various device widths. The linear relationship between

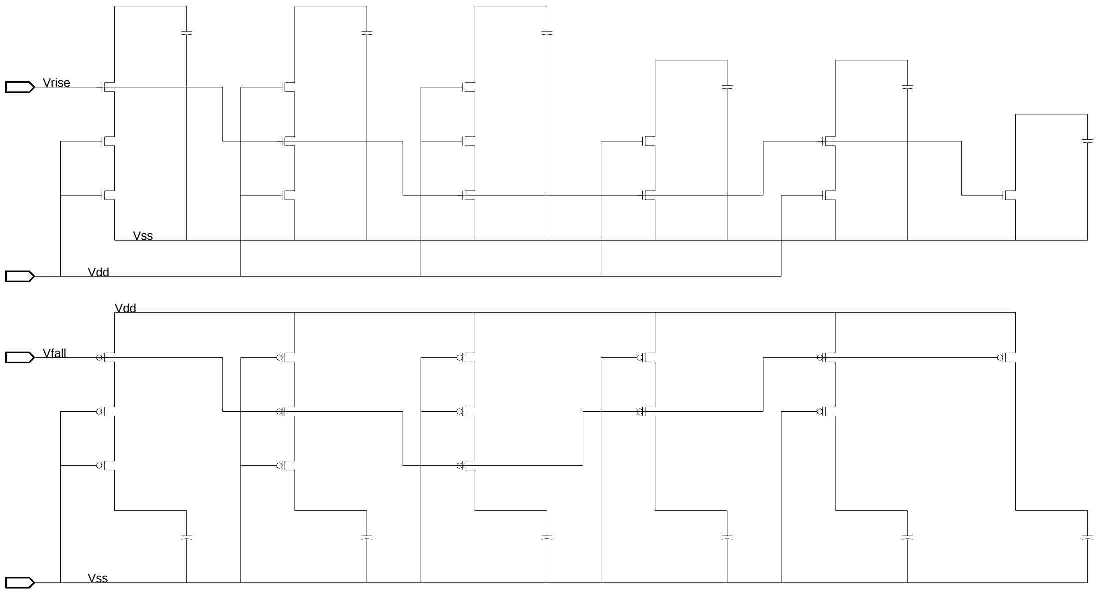

Figure 2: Configurations for which τ_m is modeled.

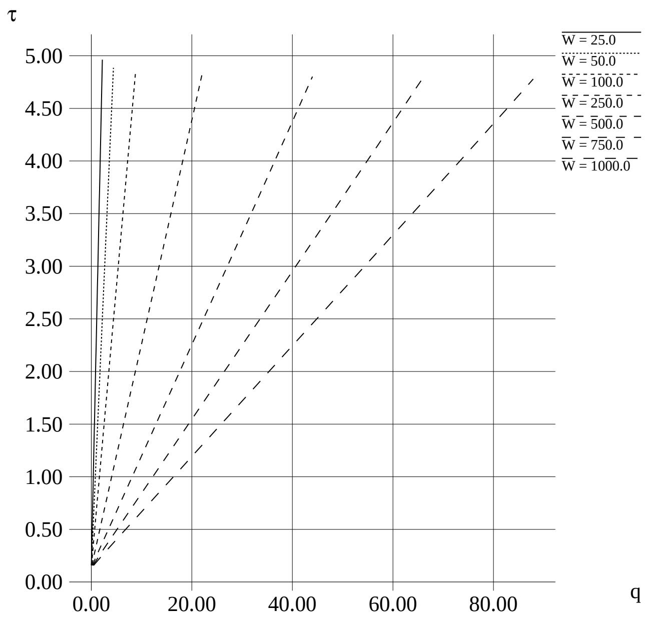

Figure 3: τ_m vs q

$\tau_m V_{DD}$ and q helps to make the modelling of the Elmore delay for MOS devices very simple.

We wanted to write $\tau_m V_{DD}$ as a function of q and the device width W. For each of the configurations shown in Figure 2 we were able to fit our simulation data to equations of the following form:

$$\tau_m V_{DD} = k_2 \frac{q}{W} + k_1 \frac{1}{W} + k_0 \qquad (11)$$

In addition the coefficients k_0, k_1, and k_2 are chosen so that $\tau_m V_{DD}$ computed using Equation 11 is always greater than $\tau_m V_{DD}$ computed from SPICE waveforms. The maximum error in $\tau_m V_{DD}$ when computed using equations of the form of Equation 11 compared to the data simulated and measured using SPICE was less than 5% for more than two thirds of the configurations shown in Figure 2. The maximum error across the samples taken in our simulations for the remaining configurations was less than 10%. It was necessary to model each configuration with a separate equation in order to obtain this accuracy.

3 Computing the Elmore Delay

In order to compute the Elmore delay for a given structure, the timing verifier identifies the discharge path of that particular structure, computes the charge which must be discharged through each device due to its load capacitance, computes the Elmore delay for each device along the path, and sums up the total delay. For symmetric waveforms the Elmore delay is the 50% point delay. Unfortunately, our voltage waveforms, examples of which are shown in Figures 4 and 5 are rarely symmetric. We therefore use a waveform model to map the Elmore delay into a 50% point delay.

Consider the circuit shown in Figure 6. If input b is rising, the discharge path will be from the output node y through node x to V_{SS}. Therefore the Elmore delay τ_y at the output node y will be the sum of the Elmore delay through devices $m1$ and $m2$ as shown by Equation 8. τ_{m1} is a function of W_{m1} and q_{m1}. W_{m1} is the width of $m1$ and q_{m1} is

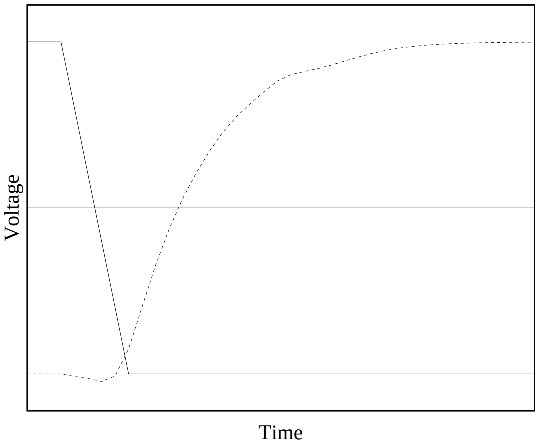

Figure 4: Example of a rising waveform

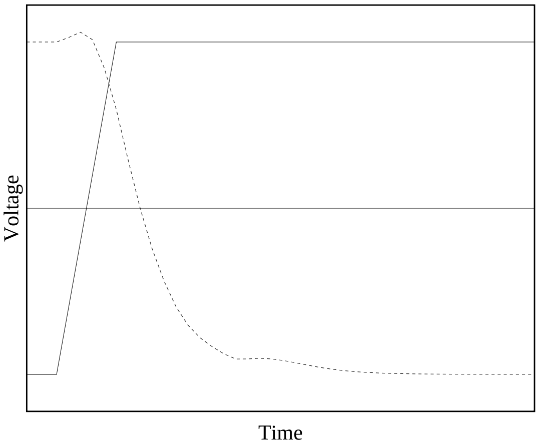

Figure 5: Example of a falling waveform

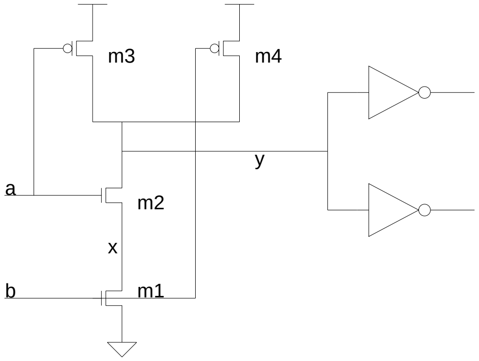

Figure 6: Example

Input	Rising	Falling
a1	6.1 %	3.9 %
b2	5.1 %	3.0 %
d3	3.4 %	5.6 %
c4	1.5 %	2.7 %
a5	4.0 %	3.0 %
d6	2.0 %	3.0 %

Table 1: % Difference in the Elmore delay beween model and SPICE

the charge that must be discharged through $m1$. Similarly, τ_{m2} is a function of W_{m2} and q_{m2}.

In order to use the models derived for the Elmore delay effectively we also had to model the charge seen by the drain of each device as accurately as possible. Here again we chose models for the charge such that the models yield values that are greater than the data obtained from SPICE simulations. Once we have computed the charge, we use the Elmore model for an NMOS device which is turning on and whose source is connected to V_{SS} in order to compute τ_{m1} and the Elmore model for an NMOS which is on and whose drain is connected to the output node to compute τ_{m2}.

After computing the Elmore delay we use a set of precharacterized waveform equations to map the Elmore delay into the 50% point delay. These equations are also based on a set of pre-simulated configurations and are a function of the Elmore delay and the slope of the input waveform.

On an AlphaServer 8400 Model 5/300 (300 MHz CPU) we can compute more than 20,000 delays/sec for circuits of the complexity shown in Figure 7. This includes identifying the discharge path, computing the charge at every node, computing the Elmore delay, and computing the 50% point delay. On the other hand we can only compute about 4 delays/sec with SPICE for these circuits after the input stimulus has been manually set up.

4 Results

For the example circuits shown in Figure 7, we show in Table 1 the % difference between the Elmore delays computed using the models of the Elmore delay that we described in Section 2 and the Elmore delay computed using SPICE. For all the examples shown the difference between SPICE and the precharacterized Elmore model is less than 10%. In 75% of our test cases the Elmore delay computed using the precharacterized model was greater than the Elmore delay computed using SPICE by less than 6%. For the remaining cases the difference was less than 10%.

In Tables 2, 3, and 4 we show the % difference between the 50% point delays computed for the same circuits using the Elmore model and SPICE. In Table 2 the input waveform has a typical transition time, in Table 3, the input waveform has a fast transition time, and in Table 4, the input waveform has a slow transition time. In 45% of our test cases the 50% point delay computed using our delay model is within 5% of SPICE, in 25% of our test cases it is between 5% and 10% of SPICE, and in 23% of our test cases it is between 10% and 20% of SPICE. For the remaining 7% of our test cases the 50% point delay was between 20% and 30% of SPICE.

One weakness still remains in this method in going from the Elmore delay to the 50% point delay. Our goal was to use the Elmore delay model to compute 50% point delays which are always greater than SPICE and within 20%. We

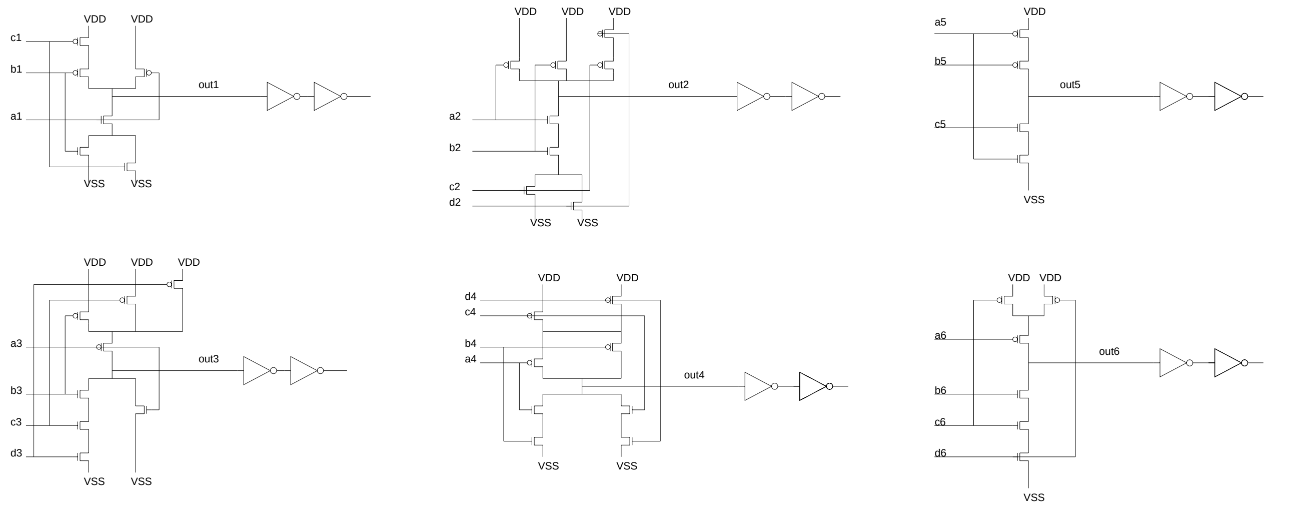

Figure 7: Sample circuits

Input	Rising	Falling
a1	3.9 %	1.2 %
b2	1.3 %	11.2 %
d3	0.8 %	23.1 %
c4	0.4 %	6.2 %
a5	7.9 %	1.0 %
d6	2.4 %	18.6 %

Table 2: % Difference in 50 % point delay for typical input slope

Input	Rising	Falling
a1	5.5 %	4.1 %
b2	0.7 %	13.1 %
d3	3.6 %	22.7 %
c4	3.8 %	4.5 %
a5	9.0 %	11.6 %
d6	2.7 %	7.0 %

Table 3: % Difference in 50% point delay for fast input slope

Input	Rising	Falling
a1	0.3 %	1.0 %
b2	0.3 %	11.8 %
d3	5.6 %	23.6 %
c4	4.4 %	2.4 %
a5	6.3 %	0.9 %
d6	0.0 %	30.0 %

Table 4: % Difference in 50% point delay for slow input slope

have fallen short of this goal for about 7% of our circuits where the error is greater than 20%. However in all cases we compute a delay which is greater than the delay computed with SPICE.

In Figure 8 a section from a critical path which was detected by the timing verifier is shown. The path contains circuits whose delays are precharacterized and circuits for which the Elmore model is used to compute the delay. For the section of the path through the latch between CLK and I_H and I_L and through the simple complementary logic between Y_L and OUT the delay is computed in the timing verfier using precharacterized delays. For the section of the path between I_H/I_L and Y_H/Y_L the Elmore model is used. The 50% point delay from CLK to OUT rising calculated using the various delay models in the timing verifier is 19% greater than the delay calculated using SPICE. The 50% point delay from CLK to OUT falling computed in the timing verifier is 11% greater than the delay computed using SPICE.

5 Conclusions

We have succeeded in modelling the Elmore delay of a device accurately and ensuring that the Elmore delay is always pessimistic. For almost all of our circuits the Elmore delay computed using the model described in this paper is greater than the Elmore delay computed using SPICE by less than 10%. We have successfully incorporated the model into our static timing verifier and used it to compute the 50% point delays of all circuits whose delays could not easily be precharacterized. The speed and accuracy of this model makes full chip timing verification of 600MHz, 15 million transistor microprocessor design feasible since the majority of non-critical paths can be filtered out by the timing verifier and do not need to be verified using time consuming circuit simulations.

6 Acknowledgements

The authors would like to thank Jim Farrell for his help.

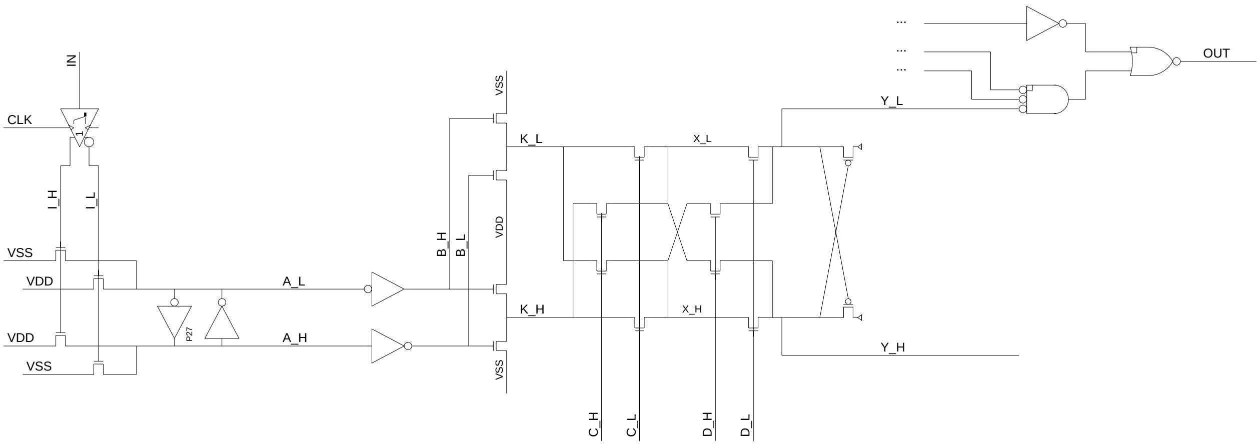

Figure 8: Critical path in 21264 design

References

[1] M.P. Desai, Y.T. Yen, A Systematic Technique For Verifying Critical Path Delays in a 300MHz Alpha CPU Design Using Circuit Simulation, *Proceedings of the 33rd Design Automation Conference*, pp. 125–130, June 1996.

[2] W.C. Elmore, The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers, *Journal of Applied Physics*, Vol. 19, pp. 55–63, January 1948.

[3] B. Gieseke, et. al., A 600MHz Superscalar RISC Microprocessor with Out-of-Order Execution, *ISSCC Digest of Technical Papers*, pp. 176–177, February 1997.

[4] W.J. Grundmann, D. Dobberpuhl, R.L. Allmon, N.L. Rethman, Designing High Performance CMOS Microprocessors Using Full Custom Techniques, *Proceedings of the 34th Design Automation Conference*, pp. 722–727, June 1997.

[5] T.M. Lin, C.A. Mead, Signal Delay in General RC Networks, *IEEE Transactions on Computer-Aided Design*, Vol. CAD-3, No. 4, pp.331–349, October 1984.

[6] D. Martin, N.C. Rumin, Delay Prediction from Resistance-Capacitance Models of General MOS Circuits, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 12, No. 7, pp. 997–1003, July 1993.

[7] L.W. Nagel, SPICE2: A Computer Program to Simulate Semiconductor Circuits, *Rep. No. ERL-M520, Electronics Research Laboratory, University of California, Berkeley*, May 1975.

[8] J.K. Ousterhout, A Switch-Level Timing Verifier for Digital MOS VLSI, *IEEE Transactions on Computer-Aided Design*, Vol. CAD-4, No. 3, pp. 336–349, July 1985.

[9] J. Rubinstein, P. Penfield, Jr., M.A. Horowitz, Signal Delay in RC Tree Networks, *IEEE Transactions on Computer-Aided Design*, Vol. CAD-2, No. 3, pp. 202–211, July 1983.

[10] C.-J. Shi, Analysis, Sensitivity, and Macromodelling of the Elmore Delay in Linear Networks for Performance-Driven VLSI design, *Int. J. Electronics*, Vol. 75, No. 3, pp. 467–484, 1993.