May 2015

Gate-level timing analysis and waveform evaluation

Chaobo Li
Syracuse University

Abstract

Static timing analysis (STA) is an integral part of modern VLSI chip design. Table lookup based methods are widely used in current industry due to its fast runtime and mature algorithms. Conventional STA algorithms based on table-lookup methods are developed under many assumptions in timing analysis; however, most of those assumptions, such as that input signals and output signals can be accurately modeled as ramp waveforms, are no longer satisfactory to meet the increasing demand of accuracy for new technologies. In this dissertation, we discuss several crucial issues that conventional STA has not taken into consideration, and propose new methods to handle these issues and show that new methods produce accurate results.

In logic circuits, gates may have multiple inputs and signals can arrive at these inputs at different times and with different waveforms. Different arrival times and waveforms of signals can cause very different responses. However, multiple-input transition effects are totally overlooked by current STA tools. Using a conventional single-input transition model when multiple-input transition happens can cause significant estimation errors in timing analysis. Previous works on this issue focus on developing a complicated gate model to simulate the behavior of logic gates. These methods have high computational cost and have to make significant changes to the prevailing STA tools, and are thus not feasible in practice. This dissertation proposes a simplified gate model, uses transistor connection structures to capture the behavior of multiple-input transitions and requires no change to the current STA tools.

Another issue with table lookup based methods is that the load of each gate in technology libraries is modeled as a single lumped capacitor. But in the real circuit, the

gate connects to its subsequent gates via metal wires. As the feature size of integrated circuit scales down, the interconnection cannot be seen as a simple capacitor since the resistive shielding effect will largely affect the "equivalent" capacitance seen from the gate. As the interconnection has numerous structures, tabulating the timing data for various interconnection structures is not feasible. In this dissertation, by using the concept of equivalent admittance, we reduce an arbitrary interconnection structure into an equivalent π-model RC circuit. Many previous works have mapped the π-model to an effective capacitor, which makes the table lookup based methods useful again. However, a capacitor cannot be equivalent to a π-model circuit, and will thus result in significant inaccuracy in waveform evaluation. In order to obtain an accurate waveform at gate output, a piecewise waveform evaluation method is proposed in this dissertation. Each part of the piecewise waveform is evaluated according to the gate characteristic and load structures.

Another contribution of this dissertation research is a proposed equivalent waveform search method. The signal waveforms can be very complicated in the real circuits because of noises, race hazards, etc. The conventional STA only uses one attribute (i.e., transition time) to describe the waveform shape which can cause significant estimation errors. Our approach is to develop heuristic search functions to find equivalent ramps to approximate input waveforms. Here the transition time of a final ramp can be completely different from that of the original waveform, but we can get higher accuracy on output arrival time and transition time.

All of the methods mentioned in this dissertation require no changes to the prevailing STA tools, and have been verified across different process technologies.

GATE-LEVEL TIMING ANALYSIS AND WAVEFORM EVALUATION

by

Chaobo Li

B.S., Zhejiang University, 2008

Dissertation

Submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical and Computer Engineering.

Syracuse University

May 2015

Table of Contents

List of Figures

Chapter 1

Introduction

Timing analysis is an integral part of modern Very-Large-Scale-Integration (VLSI) chip design. Timing analysis, such as functional verification and delay estimation, is carried out multiple times in the chip design flow. Usually, timing analysis can be categorized into dynamic or static. Dynamic timing analysis requires a set of input vectors and is mainly used to verify design's functionality whereas Static Timing Analysis (STA) checks if the design meets the timing constraints.

SPICE is the de facto circuit-level dynamic timing analysis tool. SPICE simulation is essential for full custom designs to verify electrical properties of the designs and it is widely used for industrial and research purposes due to its high accuracy. Furthermore, the tabulated timing data (i.e., delay information) of logic gates in technology libraries are obtained through extensive SPICE simulation. In this dissertation research, we also use SPICE simulation to verify the accuracy of our methods.

STA has been widely used in VLSI chip design since 1990s [1]. Not only is STA the base of timing analysis tools but also the foundation of other numerous timing optimization tools. In STA, gate delays and net delays are considered in target paths and timing constraints of each path are verified to check whether the design requirements are met. Timing data of each gate and even interconnection are available in the corresponding technology libraries, and thus simulations are not needed during timing analysis. Compared with dynamic timing analysis, STA does not rely on input vectors

and its simulation time is linear with respect to circuit size. Therefore, it is nearly the only feasible way to run full chip simulation due to its acceptable accuracy and low computation cost.

The timing analysis and optimization algorithms have become fairly mature in recent years. However, with the improvement of process technology and the continuous scaling down of feature sizes of integrated circuits, several issues that conventional STA ignored arise, and these issues can cause serious inaccuracy problems in timing analysis. In this dissertation research, we focus on the issues overlooked by conventional STA, yet cannot be ignored in highly accurate timing analysis.

1.1 Multiple-Input Transition

The prevailing STA tools use timing lookup tables (LUTs) for timing analysis. However, the standard LUTs merely provide Single-Input Transition (SIT) information without process technology details. Information for Multiple-Input Transition (MIT), which is very useful to the designers in many cases, is not available in the current LUTs. Conventional methods to tackle MIT cases are very conservative, which usually choose the worst case in SIT for MIT use. Our experiments show that these conservative methods are rather inaccurate. Based on the available timing information in current technology libraries, we propose a simplified transistor-level gate model, and both delays and transition times at gate outputs can be obtained under MIT cases. The proposed model is validated by comparing with HSPICE simulations over a number of process technologies.

1.2 Interconnection Issues

The conventional STA tools assume that the output load of any gate in the circuit is a single lumped capacitor. However, the load actually is a branch of wires connected to subsequent gates, and considering it as a single pure capacitor cannot properly capture its electric characteristics. Due to the resistance shielding effect, the real output waveform (including information such as delay, transition time, etc.) of the gate can be completely different from the ramp approximation in the LUTs. In order to ensure the accuracy of timing analysis, the resistive effect of the wire cannot be ignored any more. We propose a method that can obtain the output waveform with real interconnection as load. Furthermore, with the shrink of device sizes, the interconnection delay gradually becomes the dominant part of the overall path delay. Besides, the shape approximation of waveform at driving point of interconnection has a significant influence on signal propagation along the subsequent interconnection. A ramp approximation at driving point of an interconnection can cause as high as 25% estimation error according to our experiments. We propose a combination of several waveforms rather than a ramp to approximate the real waveform and to obtain more accurate delay results when analyzing interconnections delay.

1.3 Input Waveform Shape

Similar to output interconnection issues, the shape of the input waveform has a significant influence on gate delay estimation. The LUT based STA only uses transition time as the attribute to describe the input waveform. However, with the down scaling of device dimensions, the waveform is very sensitive to noise and electrical effects. The shape of waveforms can become very complicated and transition time alone is not

enough to capture the characteristic of waveforms. Furthermore, the conventional STA models the waveform as a perfect rising/falling waveform and use 50% point on input to 50% point on output as delay definition. However, the variation of waveform causes a problem when using conventional delay definition: as the waveform is not perfect rising/falling one, and signal voltage may pass 50% point multiple times. We propose a method that is able to map any input waveform to an equivalent ramp, and thus the inaccuracy issues caused by the shape of waveform can be handled. Our experiment results show that our approach achieves much higher accuracy when compared with the conventional STA approach.

1.4 Low Power Design

Power dissipation has become a major concern in modern CMOS circuit design. Generally speaking, circuit power dissipation comes from three sources: load charging/discharging, short-circuit current flow, and leakage. Leakage power dissipation is due to leakage current when device is not switching whereas the other dissipations happen during the device switching. Short circuit power dissipation comes from current flowing through temporary paths between voltage source and the ground when a gate output is switching and both PMOS and NMOS blocks are conducted. Dynamic power dissipation results from charging/discharging of the load capacitor during state change of device output, and is the majority of total circuit power dissipation, especially in arithmetic functional units (such as array multipliers, dividers, etc.). The proposed lower power design in chapter 5 focuses on dynamic power reduction.

In logic circuits, because of glitches and hazards, gates can switch multiple times before the circuit reaches stable states. Gates' extra switching, each referred to as a

spurious transition, wastes a lot of energy. Glitches and hazards in a circuit result from the different arrival times of signals. Therefore, we propose a buffer insertion technique to synchronize signal arrival times so as to reduce spurious transitions. Our simulation results show that new designs can significantly reduce over 50% of the spurious transitions.

1.5 Dissertation Outline

The rest of this dissertation is structured as follows: Chapter 2 describes multiple-input transition delay and transition time calculation using single-input transition data. Waveform evaluations at gate outputs are addressed in Chapter 3. Chapter 4 describes the method of how to obtain an equivalent waveform for arbitrary input waveform for delay calculation. Chapter 5 presents our low power design technique. Finally, the dissertation is concluded in Chapter 6.

Chapter 2

Multiple-Input Switching Modeling Using Single Input Switching Data from Cell Lookup Tables

Common standard cell lookup tables (LUTs) only include timing information for single-input switching (SIS), without transistor and process technology details. In some cases, for various purposes, the information for multiple-input switching (MIS) will be useful to designers, but is not provided in the tables. In this chapter, a simplified transistor-level gate model is proposed to analyze gate propagation. Based upon the model and the SIS timing information from lookup tables, either delays (or arrival times) and transition times at gate outputs can be obtained under MIS cases. The proposed model is validated by comparing with HSPICE simulations over a number of process technologies, including several in 22nm and 16nm. Both delays and transition times are well within 8% of HSPICE simulations for all cases and process technologies that we tried.

2.1 Introduction

Static timing analysis (STA) is largely used in CMOS circuit design attributable to its fast runtime and acceptable accuracy in the past process technologies [1]. However, it completely ignores Multiple-Input Switching (MIS) and merely provides cell lookup tables (LUTs) for Single-Input Switching (SIS) [1]. Therefore, the needed timing information for MIS are not available in common standard LUTs, and hence the simulation inaccuracy. Especially this inaccuracy issue has become more acute as process technology scales down.

Specifically, STA tools simply supplies SIS information in MIS cases, which can cause significant timing estimation errors. As Figure 1 indicated, taking NAND2 gate for example: when two falling signals simultaneously arrived at the input terminals, voltage source would charge the load through PMOS transistors, which could be twice faster than charging through one single transistor in SIS. Therefore applying SIS model to this case would largely overestimate the gate delay. Similarly, different relative signal arrival times cause noticeable different delays in [6], and in [8] as high as 100% prediction error can be produced in delay estimates without taking MIS into consideration.

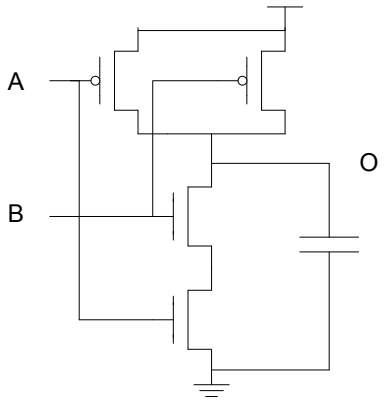

Figure 1. A typical CMOS NAND2 gate.

Two models have been developed to analyze gate timing behavior including MIS in CMOS: Voltage-Response Model (VRM) and Current-Source Model (CSM) [18].

Being a gate-level model, VRM simplifies the output signal as the function of input slew and load capacitance; and then uses them as indices in the common standard 2-dimentional LUTs to obtain output signal's arrival time and rising/falling transition time. Such empirical method of the delay based table [2] could be relatively simple and save both simulation runtime and storage cost. However, the accuracy is greatly compromised unless the LUTs are expanded [2-13] by including more indices to capture every possible event. For example, [2] uses an extra table of relative arrival time; [3] creates a 4-dimensional LUT by modeling the output signal as a function of input transition time, relative arrival time and load capacitor; [6] enlarges the LUTs with more sample points to get more detailed timing information

Especially in the case of MIS [2-6], VRM is incapable of solving the estimation error because it represents each gate as a black box and fails to model the electrical effect [18]. This issue stands even if VRM considers the signals correlation by modifying or expanding the LUTs.

Then, CSM, at both gate level and transistor level, have been academically proposed to model gates as current sources and equivalent capacitors. But the gate-level CSM is just like VRM having the black box issue.

Though the transistor-level CSM considers the gate internal structure and the relationships among physical variables to cover MIS, yet still it brings in additional process variables and that produces complicated equations and enormous detailed background data including interconnection effects and signal waveform [7-19].

Moreover, both types of CSMs require significant modifications or extensions to the LUTs to involve sufficient physical process variables to handle MIS cases [7-19]. Nevertheless, similar to VRM, such high-dimensional LUTs exponentially increase simulation complexity in terms of both runtime and storage space and would make the model incompatible to the current prevailing STA tools.

For instance, [7] modifies the common standard LUTs to retrieve charge characteristic and current mapping based on voltage indices; [9, 10] also modifies the LUTs too to set indices as input voltage and output voltage and change the table data to the current value of the gate; based on [9-10], [11-13] introduce an additional LUT to store the capacitor values rather than the current value. Note that even though [11] tries to take gate internal node into consideration, it only presented a two-input gate model without covering all the MIS events.

Inspired by these works, we model gates at transistor level and use transistors' connection structure to analyze CMOS gate propagation without introducing additional process variables. So we make neither extension nor modification to the LUTs to keep the simulation complexity low. In other word, our model directly uses the existing LUTs to obtain all the needed MIS timing information. As a result, we can obtain the delays (arrival times) and transition times at gate outputs only based on the STA's SIS information while increase the simulation accuracy. This also allows a seamless integration with STA tools to remarkably widen the usage of our model in practice.

Particularly, the previous models, regardless of being VRM or CSM, are all only validated on 90nm or higher that had become increasingly unpopular in industries. But our model is validated by comparing with HSPICE simulations over a number of process

technologies including the dominant 45nm or lower, among which 16nm and 22nm we believe have never been experimented before.

The remainder of this chapter is organized as follows. We explain our model by starting with the single transistor current model in Section 2, and extend it to its variances including the series-connected transistors, parallel-connected transistors as well as the hybrid-connected transistors in Section 3, all of which combined show how our model covers all the MIS cases; and the modeling validation results by experimenting over the process technologies are presented in Section 4; at last, we conclude the chapter in Section 5.

2.2 Transistor Current Model

CMOS gates keep their states by storing the charge in capacitors, so the gate state switching could be considered as charging and discharging load capacitors through gate's transistors. Therefore, we model the output signal based on the characteristics of CMOS transistors and load capacitance.

Then we will explain how to model the transistor current in three basic regions: Cutoff, Saturation and Linear, supposing that a rising input signal arrives at an NMOS transistor and gradually turns it on (same case with PMOS).

At the very beginning, V_{gs} (gate-source voltage) is less than V_{th} (threshold voltage) so the transistor operates in the Cutoff region. Because the leakage currents are too small to be counted in, compared with the currents after the transistor is turned on, we take drain-source current I_{ds}=0.

Then V_{gs} is higher than V_{th} but still lower than V_{ds} (drain-source voltage) and thus the transistor operates in the Saturation region. In this region Ids increases almost linearly with respect to V_{gs}. At t_{ip}, when V_{gs} reaches V_{dd} (supply voltage) or when $V_{gs}=V_{ds}+V_{th}$, the transistor is fully turned on and I_{ds} reaches its maximum value I_m. So during this region I_{ds} can be deduced in Equation 1.

$$I_{ds} = \frac{V_{gs}(t)-V_{th}}{V_{dd}-V_{th}} * I_m \quad (1)$$

Here the I_m is the maximum value and it can be calculated by the following equation. We define I_{max} as the maximum I_{ds} when an ideal step input signal is applied to the gate terminal.

$$I_m = \frac{V_{ds}(t_{ip})}{V_{dd}} * I_{max}$$

In the linear region, the transistor channel is rather stable, and thus the transistor can be considered as a resistor between the drain and source terminal. "*R*" in (2) is the equivalent resistant and "*C*" is the load capacitance.

$$I_{ds} = I_m * e^{-\frac{t-t_0}{RC}} \quad (2)$$

Since all the charger of the load is discharged through the transistor, integration of I_{ds} will give the amount of charge discharged from the load; therefore we have:

$$\int I_{ds}(t)dt = \Delta V_{out}(t) * C_{load} \quad (3)$$

Based on the above transistor current model, we extend it into several variances by considering varied transistor connection structures to retrieve MIS information (delay and transition time) from the SIS timing information in the common LUTs.

2.3 Calculation Method of Multiple-input Switching Timing Information

Our MIS timing information calculation method has the following steps: Firstly, SIS currents are retrieved based on the introduced transistor current model and the SIS timing information from LUTs; Secondly, we use transistor connection structure and SIS current waveforms to recover the MIS current; At last, based upon the relation between current and voltage, gate delay and signal transition time in MIS cases are obtained.

2.3.1 Generating Single-input Switching Current from LUTs

Based on the SIS timing information (including input signal transition time, gate delay and output transition time) from the common standard LUTs, we are able to retrieve signal ramps under SIS cases. Then the SIS current waveform can be retrieved based on our transistor current model.

2.3.2 Current in Series-connected Transistors

For this structure, the charging/discharging current flows along the only transistor path to the source. The MIS current is constrained on the transistor with the narrowest channel (which is turned on most slowly). Take a two-input NAND gate in Figure 1 as an example. Given two SIS cases from different input, we can get input and output ramps

shown at left in Figure 2. According to the transistor current model in Section 2, we are able to model the current of SIS case 1 and case 2.

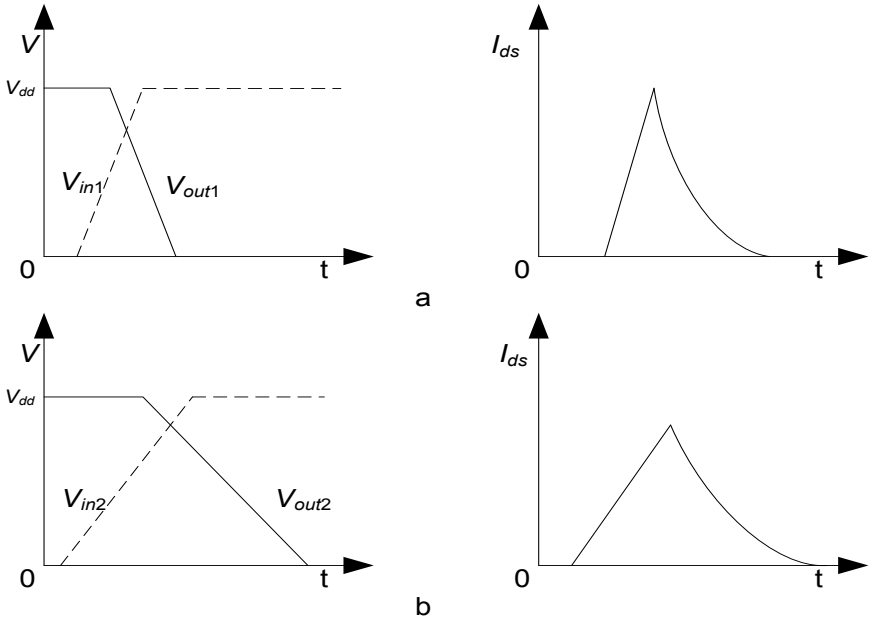

Figure 2. Two SIS currents for NAND gate.

Then we combine SIS currents to get the MIS current. We first draw two SIS currents in Figure 3 for better visualization.

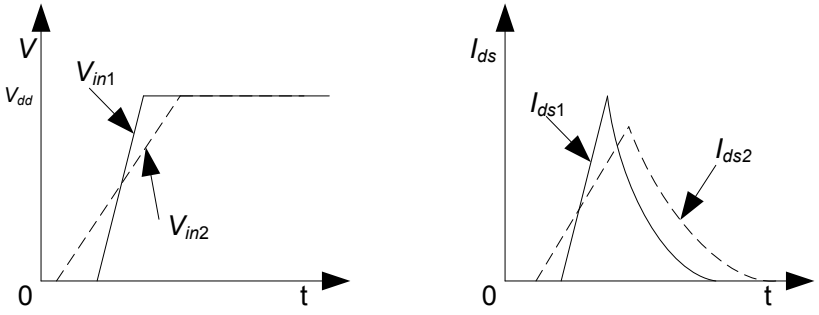

Figure 3. MIS case current.

In series-connected structure, the load would not discharge until both transistors leave the cutoff region (i.e., at point *A* as shown in Figure 4). And the value of current for the transistor B in SIS cannot be reached unless the transistor A is fully turned on (i.e., at t_1). As the current increase linearly in saturation region (i.e., from *A* to $t_{1)}$, we obtain MIS current "*AO*" as Figure 4 shows. From t_1 to t_2 the transistor B is still operating in the saturation region while the transistor A is fully turned on. So the MIS load current during this region changes following the transistor B's increasing slope. Thus, we get the load current curve from point *O* to point *B'*.

From (3), we know $Q_3/Q_2 = \Delta U_3/\Delta U_2$, and since $Q_3(t_2)<Q_2(t_2)$ (we can tell from the area); thus, at t_2, we can get$\Delta U_3<\Delta U_2$. In case 2, at t_2, transistor B is fully turned on, so V_{gs}-$(V_{dd}$-$\Delta U_2)$= Vth. In MIS case 3, at t_2, V_{gs}-V_{ds}=V_{gs}–$(V_{dd}$-$\Delta U_3)$<V_{th}, which means the transistor B is still not fully turned on at t_2. Since the discharging current would continue the slop after t_2, and suppose at t_3 transistor B is fully turn on, we can get an equation:

$$V_{gs2}(t3) - \left(V_{dd} - \frac{Q_3(t_3)}{Q_2(t_2)} * \Delta U_2(t_2)\right) = V_{th} \qquad (4)$$

The term $Q_3(t_3)/Q_2(t_2)$ = area(*"AC't*$_3$*"*)/area(*"BB't*$_2$*"*). By solving this equation, we can get t_3.

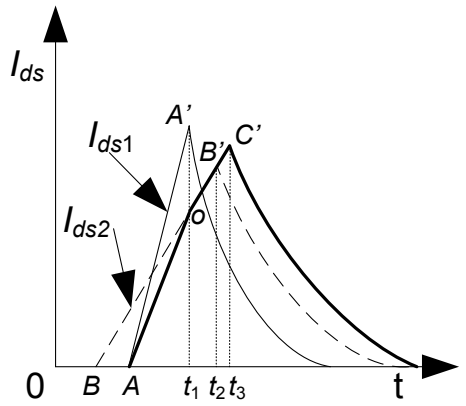

Figure 4. Combine SIS currents into the MIS current.

After t_3 when both transistors go into the Linear, we can have the MIS load current based on (2) following the transistor current model. According to the MIS current value at *t*3, we get the current waveform from SIS case to complete the remaining part of MIS, since both SIS and MIS cases have the same "*R*" and "*C*" now.

2.3.3 Current in Parallel-connected Transistors

In circuits, transistors are connected in parallel, such as NMOS part of the NOR gate in Figure 5. When two NMOS transistors start to conduct simultaneously, the load is able to discharge through these two transistors.

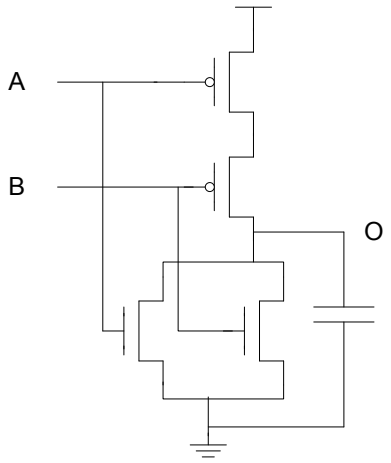

Figure 5. A two-input NOR gate.

Different from series-connected structure, once a transistor starts to conduct, the discharging begins, and the discharging current I_L is the sum of each transistor's I_{ds}.

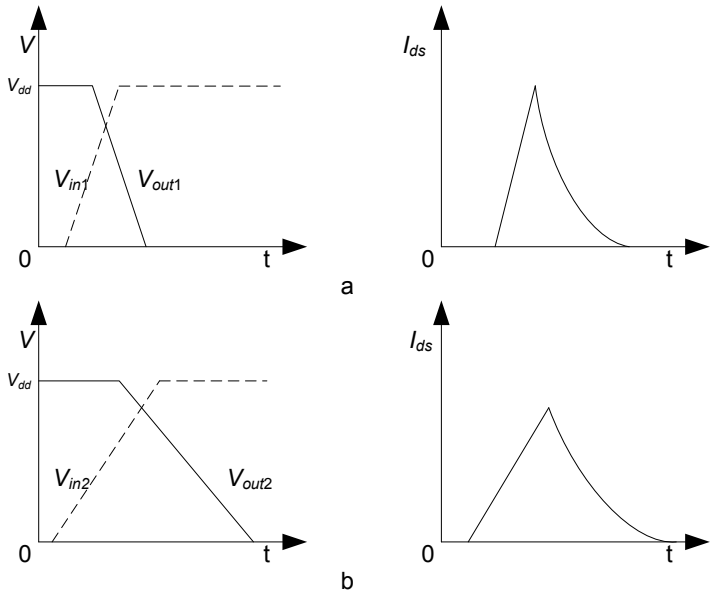

Figure 6. Two SIS cases' currents for NOR Gate.

Similarly, we use two SIS currents to recover the MIS current.

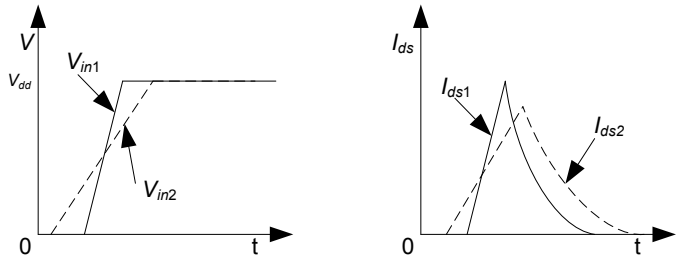

Figure 7. MIS case current for NOR Gate.

When both transistors are cutoff, there is no load current I_L. And when any transistor leaves its cutoff region, the load begins to discharge, and I_L is equal to sum of the two SIS drain-source current. Assume the current through transistor x in SIS case is $I_{ds\text{-}sx}$. Therefore, $I_L = I_{ds\text{-}sb}$ from t_1 to t_2 in this example (shown in Figure 8). After t_2, the other transistor A starts to conduct, and thus the current $I_L = I_{ds\text{-}sa} + I_{ds\text{-}sb}$. I_L continues the slope till one transistor is fully turned on (say, at t_3).

Since the total charge Q_L equals to sum of charge Q_a discharged through transistor A and charge Q_b discharged through transistor B. Thus, according to Equation 3, we can have:

$$\Delta U_L(t_3) = \Delta U_{sa}(t_3) + \Delta U_{sb}(t_3) \tag{5}$$

ΔU_{sx} is the transistor x's drain-source voltage drop, which can be calculated by Equation 3. And because transistor A is fully turned on at t_3, the drain-source voltage U_L is V_{th} less than the transistor A's input voltage $V_{g\text{-}a}$, which means:

$$V_{dd} - \Delta U_L(t_3) = V_{g-a}(t_3) - V_{th} \tag{6}$$

By solving this equation, we are able to get the time t_3.

After t_3, transistor A behaviors like a resistor and transistor B is still gradually turned on till t_5. Similar as the above, we calculate t_5 using the following equations:

$$\Delta U_L(t_5) = \Delta U_{sa}(t_5) + \Delta U_{sb}(t_5) \quad (7)$$

$$V_{dd} - \Delta U_L(t_5) = V_{g-b}(t_5) - V_{th} \quad (8)$$

And after t_5, both of transistors can be considered as resistors. Similar to series-connected cases, we can use SIS current curves of the linear region to model the rest I_L. However, different from series-connected cases, the resistant in MIS between load and ground is $(R_a * R_b)/(R_a + R_b)$. If the transition time after t_5 for SIS case 1 is T_{sa}, and for SIS case 2 is T_{sb}, then the transition time of U_L after t_5 could be $(T_{sa} * T_{sb})/(T_{sa} + T_{sb})$.

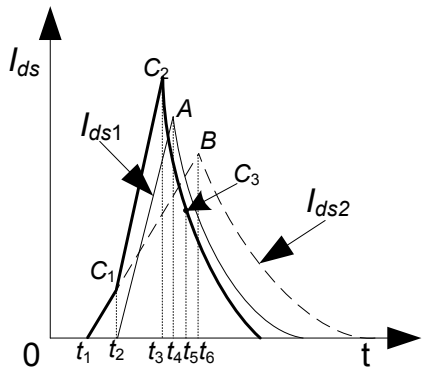

Figure 8. MIS case current combination.

2.3.4 Current in Hybrid-connected Transistors and Delay and Transition Time Calculation

CMOS gates may have complex transistor connection structures including both series-connected and parallel-connected structures. In such a hybrid-connected case, we could acquire MIS current using transistor partition. Take AOI21 gate in Figure 9 for example. The MIS current Ids of transistors A and B is obtained based on the series-connected transistors model. Then take these two transistors as one "transistor", it actually connects with the transistor C in parallel thus we can get the total MIS current between them using the parallel-connected model.

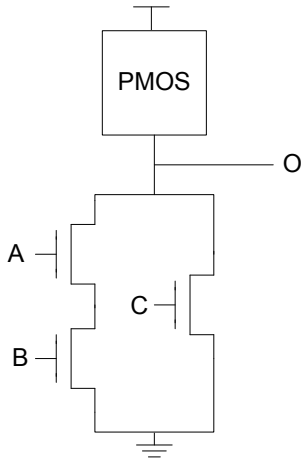

Figure 9. Two series-connected and one parallel-connected.

Since we already retrieved MIS current, we can use the relation between charge and the voltage drop on the load to calculate the time when the voltage drop reaches $0.2V_{dd}$, $0.5V_{dd}$ and $0.8V_{dd}$ as in Equation 3.

2.4 Experimental Results

We evaluate the accuracy of our model by comparing with the HSPICE MIS simulation results over several process technologies with a set of library cells. Here the 45nm, 22nm and 16nm PTM technology from [18] are tested, including their respective HP (high performance) and LP (low power) models at the suggested voltages. Besides, the lower the load capacitance, the more likely output signals are to be affected. Nevertheless, satisfactory results within the 8% of HSPICE simulation results were achieved even when we set the load capacitance at as low as 1.6fF. Moreover, all the SIS LUT data are derived by HSPICE (A2008.03) simulation.

Then, for each gate under several process technologies, the average result for both the delay error and transition time error have been calculated based on every 50 input cases, as the Table 1 and Table 2 respectively shows.

Table 1. Average Delay Error.

Delay Error	NAND2	NOR2	NAND3	NOR3	AOI21
45nm HP	1.42%	4.32%	3.46%	0.66%	0.34%
45nm LP	2.00%	1.35%	3.48%	0.96%	0.56%
22nm HP	0.85%	4.96%	1.88%	4.48%	8.46%
22nm LP	0.71%	8.75%	1.48%	3.50%	8.99%
16nm HP	1.13%	8.10%	1.34%	8.17%	0.16%
16nm LP	0.53%	8.88%	0.81%	9.94%	2.65%

Table 2. Average Transition Time Error.

Slew Error	NAND2	NOR2	NAND3	NOR3	AOI21
45nm HP	0.66%	7.39%	0.55%	7.87%	1.10%
45nm LP	0.41%	6.50%	0.29%	1.95%	9.67%
22nm HP	0.15%	4.96%	0.64%	4.53%	2.27%
22nm LP	0.08%	9.81%	0.07%	8.64%	4.87%
16nm HP	0.23%	8.86%	0.11%	7.19%	9.34%
16nm LP	0.04%	6.31%	0.07%	5.28%	9.89%

As seen clearly from both tables, almost all the results are within 8% except very few, and a number of them could be as low as within 1%. This is so even for 22nm and 16nm which probably have been first simulated by us.

2.5 Summary

In conclusion, our model successfully acquired the needed MIS information without any modifications or extensions to the LUTs, and achieved higher accuracy without increasing simulation complexity. Therefore our model can be readily integrated with the prevailing STA tools and widely used to accommodate the increasingly complex needs in the frontier circuit designs. Particularly our model could be further developed for the statistical STA and this surely could be part of our future works.

Of course our model has its limits. It focused on the circuit purely consisted of transistors, omitting the situation with non-transistors. Though this would be entirely acceptable since the CMOS (transistor) is the most popular/applicable.

Chapter 3

Timing Analysis on RC Interconnection for Lookup Table based Design

This chapter presents a lookup table (LUT) based scheme for timing analysis of circuits. As in many realistic cases, we assume that only LUTs from cell library are available to designers, without detailed technology data. Thus, it is not possible to run SPICE simulation. In this chapter, we first show how to obtain the output waveform for a cell using the information from LUTs. We then show how to propagate the waveforms to the subsequent gates through RC interconnection trees. Since LUTs only provide timing information for a single lumped load (capacitor), we need to obtain an equivalent RC structure by transforming the original RC tree into a reduced form. This is done by obtaining the transfer function from the gate output to each sink node of the interconnect tree. In this way, LUT based methods can be extended to simulate the entire circuit, including gates and interconnection. Experimental results shows good accuracy (within 5%) when compared to SPICE simulation results. This is sufficient for an early-stage timing analysis before the final full-chip analysis.

3.1 Introduction

With the improvement of process technology, the feature size of integrated circuit continue to decrease and the interconnect delay gradually becomes the dominant part of the overall path delay. In order to ensure the accuracy of timing analysis, interconnect delay has become a crucial part in current and future timing analysis.

Timing analysis methods based on lookup tables (LUTs) are largely used in CMOS circuit design due to its fast runtime and acceptable accuracy in the past process technologies [1]. Conventional LUTs use input transition time and load capacitance as parameters to obtain gate delay and output transition time, meaning it merely provides timing information with load of single capacitor. However, the real load of a gate in circuits is a RC interconnect tree, while the conventional LUTs merely provide timing information for a single lump load (capacitor). Due to the resistance shielding effect, the real output waveform (delay, transition time, etc.) of the gate is very different from the ramp approximation in the LUTs.

Furthermore, the waveform approximation at gate output has a great influence on signal propagation along the subsequent interconnection. However, creating higher dimensional LUTs to cover all the interconnection cases is not feasible. Therefore, a lot of methods are proposed to reduce interconnects into a simpler structure. Although these methods predict the gate delay with reasonable accuracy, they are barely able to predict the signal transition time, not to mention the waveform, such as the method in [21]. To obtain a more accurate approximation of the driving point waveform, a piecewise linear waveform model is proposed in this chapter. Based only on the standard LUTs, a

complete gate output waveform can be obtained with reasonable accuracy for any RC interconnect load.

The remainder of this chapter is organized as follows. Section 2 discusses the background of LUT methods, resistance shielding effect and effective capacitance model. Section 3 proposes our method to approximate the driving point waveform and the signal propagation on interconnects. Section 4 describes our signal propagation method along the interconnection. The modeling validation results by experimenting over a number of process technologies are presented in Section 5. Finally, we conclude the chapter in Section 6.

3.2 Background

LUT methods are extensively used in timing analysis of CMOS circuit design due to their feasibility. Using input signal transition time and output load (capacitance) as parameters, conventional timing LUTs provide designers with gate delay and output transition time. LUT methods assume that the gate load is a single lumped capacitor whose capacitance is the sum of all following gates' input capacitance, and the signal is simplified as a ramp. However, these simplifications can cause much inaccuracy with the increase in the interconnect-delay portion of overall circuit delay.

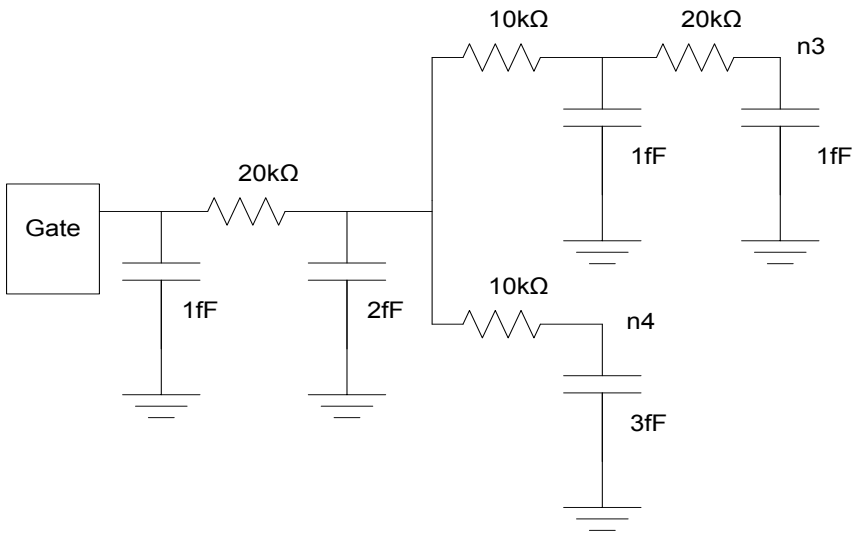

Figure 10. Interconnection example.

It can cause inaccuracy in the timing analysis when assuming the signal as a ramp. Take the circuit in Figure 10 as an example; the ramp approximation is no longer able to model the behavior of the real waveform. Given a 50% point and transition time of driving point waveform, we draw a ramp to approximate the waveform at the gate output. However, as the waveform is propagated through the RC tree, and it results in a 22% less delay and 26% less transition time at 'n3'. At node 'n4', we get similar results as shown in the Figure 11. When the signal is propagated to the following stages, this inaccuracy will accumulate a dramatic error, not to mention that waveform can become more unpredictable as propagated through complicated interconnection.

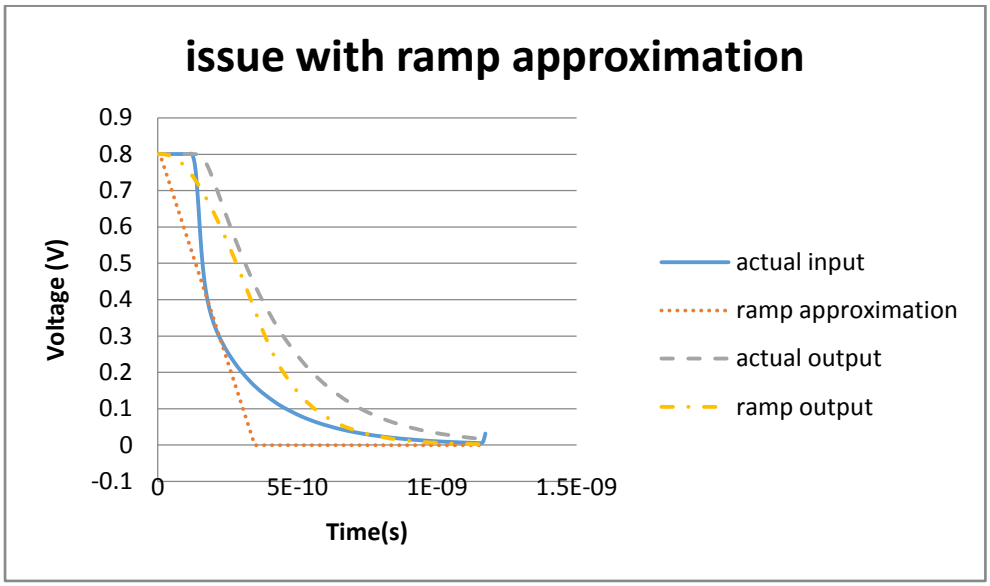

Figure 11. Waveform comparison at node 3 (n3).

The assumption of the load may not be feasible in the current process technologies. When the density of CMOS circuits increases, the gate delay reduces dramatically but the resistance of the interconnection remains relatively constant. The electrical behavior of a gate load can be very complicated. The real load of a gate is usually a metal wire connecting to the following gate input. The wire in the circuit could have many electrical characteristics, such as resistance, capacitance and inductance. Therefore, modeling the load of a gate as a single capacitor can bring in much inaccuracy. Since the conventional LUTs only provide timing information upon the assumption that the load is a single capacitor while the real load is a RC interconnect tree, we need to find the relation between a single capacitor and real load to model the waveform at gate's driving point. Various methods have been proposed to reduce the RC interconnect to a single "effective

capacitor", however, they can only predict the reasonable delay, not the entire waveform. In this chapter, we will introduce a admittance based method to simplify RC interconnects, and model the driving point waveform, only based on the timing information of conventional LUTs and interconnect information.

Given any RC interconnect tree, let $Y(s)$ denote the driving point admittance. We can represent $Y(s)$ by its Taylor series expansion around s=0.

$$Y(s) = \sum_{i=1}^{\infty} y_i s^i \tag{9}$$

For a single capacitor, the admittance is

$$Y(s) = Cs \tag{10}$$

For RC-lump,

$$Y(s) = \frac{Cs}{1+RCs} = \sum_{i=1}^{\infty} (-1)^{i-1} R^{i-1} C^i s^i \tag{11}$$

For RC π-structure,

$$Y(s) = C_1 s + \frac{C_2 s}{1+RC_2 s} = (C_1 + C_2)s + \sum_{i=2}^{\infty} (-1)^{i-1} R^{i-1} C^i s^i (4) \tag{12}$$

In [22], a reduction method has been provided to calculate the first three moments of the driving point admittance from the leaf node. As for the case in Figure 12,

Figure 12. *Y*(*s*) propagation over a resistant.

We can calculate the $Y_i(s)$ from $Y_j(s)$:

$$\begin{cases} y_{i1} = y_{j1} \\ y_{i2} = y_{j2} - R_{ij}\, {y_{i1}}^2 \\ y_{i3} = y_{j3} - 2R_{ij}\, y_{j1} y_{j2} + {R_{ij}}^2 {y_{j1}}^3 \end{cases} \tag{13}$$

As for the case in Figure 13,

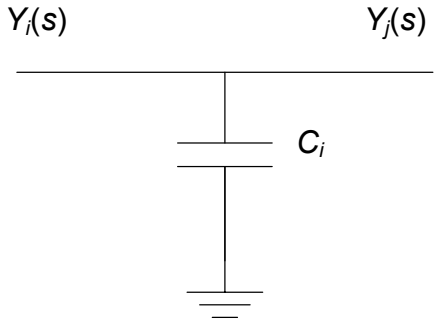

Figure 13. *Y(s)* propagation over a capacitor.

We can calculate the $Y_i(s)$ from $Y_j(s)$:

$$\begin{cases} y_{i1} = y_{j1} + C_i \\ y_{i2} = y_{j2} \\ y_{i3} = y_{j3} \end{cases} \tag{14}$$

As for the case in Figure 14,

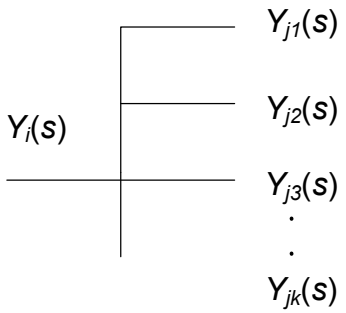

Figure 14. Y(s) propagation at fan-out node.

We can calculate the $Y_i(s)$:

$$\begin{cases} y_{i1} = \sum_{k=1}^{n} y_{jk\,1} \\ y_{i2} = \sum_{k=1}^{n} y_{jk\,2} \\ y_{i3} = \sum_{k=1}^{n} y_{jk\,3} \end{cases} \tag{15}$$

By using the first three moments of the driving point admittance, the RC interconnect tree can be reduced to a π-model. Once we obtain the $Y(s)$, we can calculate the R, C_1, C_2 by Equation 16.

$$\begin{cases} C_1 = \frac{{y_2}^2}{y_3} \\ C_2 = y_1 - \frac{{y_2}^2}{y_3} \\ R = -\frac{{y_3}^2}{{y_2}^3} \end{cases} \tag{16}$$

Figure 15 shows the π-model synthesis for the RC interconnect in Figure 10.

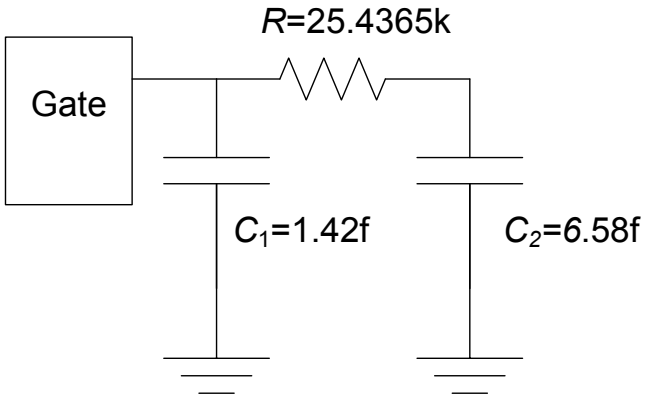

Figure 15. A reduced model for the RC interconnect in Figure 10.

Simulation result in Figure 16 shows that the two waveforms of the driving points at these two circuits are very close to each other. Therefore, using first three moments of the admittance provides good accuracy for the RC interconnect reduction. Furthermore, we can use four or even more moments to increase the accuracy in this admittance calculation.

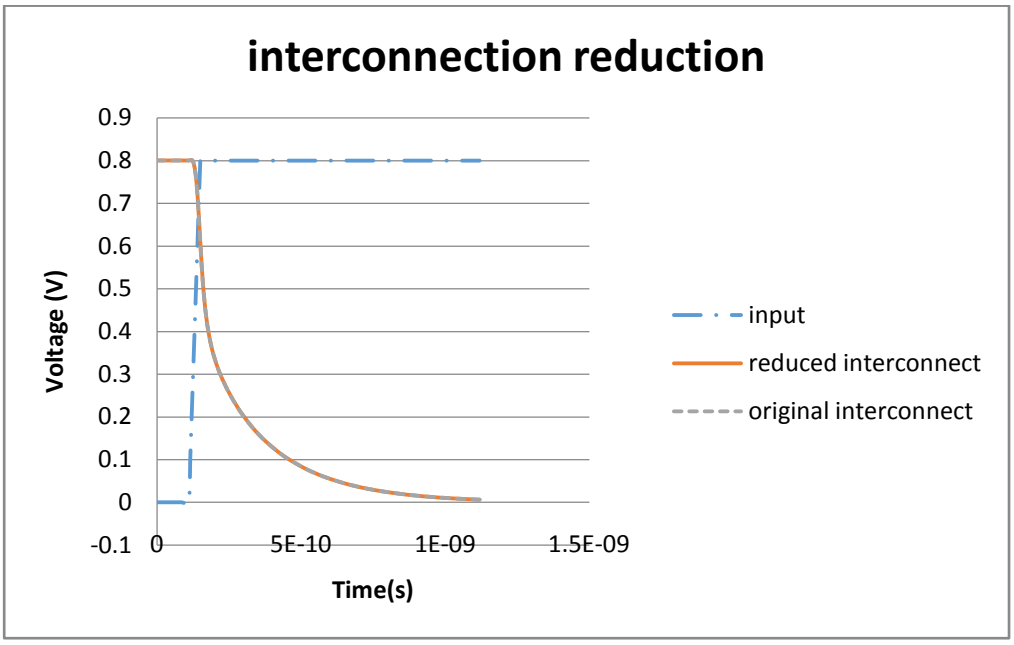

Figure 16. Waveform comparison at the driving point.

3.3 Waveform Approximation

3.3.1 Reduction Algorithm and Transfer Function Propagation

Given RC interconnect *TREE*(*E,V*), we can calculate the driving point admittance from its leaf nodes. *E* is the set of all edges in *TREE*: include resistors and capacitors, and *V* is the node set, including the root node (driving point). Let C_i denote the capacitance between node v_i and ground, and R_{ij} denotes the resistance between node v_i and node v_j. The admittance of each node can be presents as its Taylor series expansion. In our experiment, we use first three moments of admittance' Laplace expression. That means:

$$Y(s) = y_1 s + y_2 s^2 + y_3 s^3 \qquad (17)$$

We use the following two functions to calculate the admittance of each node on the interconnection.

```
//Calculate the admittance of each node
CalculateY(v_i)
{
    if (v_i is a leaf node)
    {
        y_i1 = C_i ;
        y_i2 = 0 ;
        y_i3 = 0 ;
    }
    else
    {
        foreach child node v_j of v_i
        {
            Y_i(s) = Y_i(s) + PropagateY(v_i, v_j);
            y_i1 = y_i1 + C_i;
        }
    }
    return Y_i(s);
}
```

Figure 17. Function to calculate the admittance.

The function CalculateY uses recursive procedure to calculate the input node *vi*'s admittance. If the current node is a leaf node, which means the node connects to the ground via a capacitor, the node admittance can be obtained by Equation 10. For other nodes, its node can be obtained by the sum of the admittances of all its child nodes. The admittance of each child node can be propagated to the current node by using the function PropagateY. In the real use, we pass the driving point to the function for each interconnect tree. Each node will be travelled once.

```
//Calculate the admittance of v_i by the admittance of v_j
PropagateY(v_i, v_j)
{
    Y_j(s) = CalculateY(v_j);
    y_i1 = y_j1;
    y_i1 = y_j2 − R_ij y_i1^2;
    y_i3 = y_j3 − 2R_ij y_j1 y_j2 + R_ij^2 y_j1^3;
    return Y_i(s);
}
```

Figure 18. Function to propagate the admittance.

The function PropagateY implements Equation 13 to propagate admittance from downstream node. It recursively calls the function CalculateY to return the admittance results.

By these two functions, we can calculate the first three moments of each node's admittance along the RC interconnect tree. Therefore, we can use Equation 16 to get equivalent π-model load at any node.

In order to propagate the waveform from the driving point to the leaf node, we need to get the transfer function between the driving point and the leaf node. Since we have retrieved the admittance of any node, we can use the following method to obtain the transfer function between two adjacent nodes.

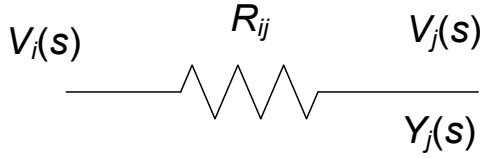

Figure 19. Transfer function calculation.

The current through register R_{ij} is:

$$I_{ij}(s) = V_j(s) * Y_j(s) \quad (18)$$

Since,

$$V_i(s) = V_j(s) + I_{ij}(s) * R_{ij} \quad (19)$$

By using Equation 18 and Equation 19, we can obtain the transfer function between node i and node j:

$$H_{ij}(s) = \frac{V_j(s)}{V_i(s)} = \frac{1}{1+Y_j(s)*R_{ij}} \quad (20)$$

Therefore, the transfer function between any two node on the interconnect tree can be get from Equation 21.

$$H_{mn}(s) = H_{mi}(s) * H_{ix}(s) \dots H_{yj}(s) * H_{jn}(s) \quad (21)$$

By this way, we can propagate the waveform of the driving point to any node on the RC interconnect tree.

3.3.2 Effective Capacitance

This subsection introduces a method to find an effective capacitor that will result in the same delay as a RC interconnect tree. Since a π-model load can be retrieved from the last subsection and this π-model load has very close waveform as the original RC interconnect tree, we adopt an effective capacitor method in [26] to find an effective capacitor.

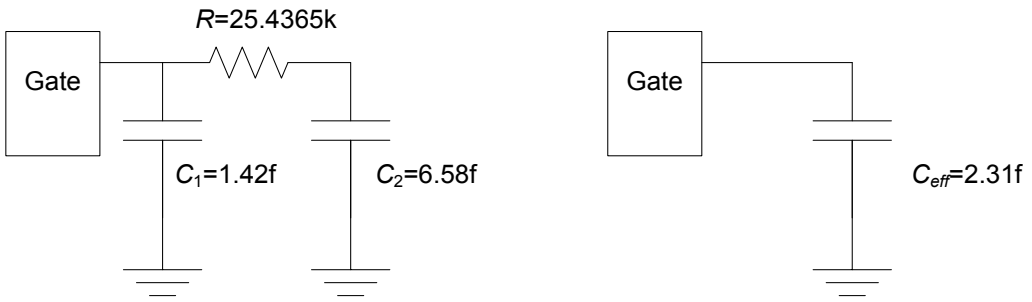

Figure 20. Effective capacitance calculation.

The average currents for the waveform of $V_{out}(t)$ from starting time (t_0) to 50% delay (t_{50}) are equal, because of the same voltage drop and charge change. Therefore, we can get Equation 22.

$$\frac{1}{t_{50}-t_0}\int_{t_0}^{t_{50}} I_{\pi}(t)dt = \frac{1}{t_{50}-t_0}\int_{t_0}^{t_{50}} I_{ceff}(t)dt \tag{22}$$

By using the Equation 22, we adopt the effective capacitance calculation in [7].

$$C_{eff} = C_1 + C_2 * [1 - \frac{RC_2}{t_{50}-0.5t_{20}} + \frac{RC_2}{t_{20}(t_{50}-0.5t_{20})} e^{\frac{-(t_{50}-t_{20})}{RC_2}}(1 - e^{\frac{-t_{20}}{RC_2}})] \tag{23}$$

In this equation, t_{50} can be obtained from the LUT directly. As for t_{20}, we use Equation 24 to calculate t_{20}.

$$t_{20} = t_{50} - \frac{t_r}{2} \tag{24}$$

We use the total capacitance as a initial effective capacitance to obtain delay and transition time, then use these two timing information to calculate the C_{eff}. By doing this iteratively, the C_{eff} can converge to a stable value. In our experiments, this iteration usually takes three or four times.

The iteration procedure to get effective capacitance:

1) Set the load capacitance C_{eff} to the total capacitance
2) Use C_{eff} and input transition time to get delay and output transition time by looking up timing table
3) Use values got from step 2 and Equation 24 to calculate t_{50} and t_{20}
4) Use Equation 23 to calculate the new C_{eff}
5) Go to step 2 if C_{eff} doesn't converge

Figure 21. Iteration procedure to get effective capacitance.

3.3.3 Waveform Approximation at Driving Point

3.3.3.1 Linear approximation

As shown in section 2, we need to approximate the waveform at the gate output. Usually, the timing information provided by the LUTs is the transition time and delay. Therefore, from conventional timing LUTs, we can get the time t_r during which the output voltage changes from 20% to 80% V_{dd}, and 50% delay point t_{50}. The linear approximation equation for this waveform can be presented in Equation 25.

$$V_{out\,0}(t) = \frac{V_{80}-V_{20}}{t_r}\left[t * u(t) - \left(t - \frac{100}{60} * t_r\right) * u(t - \frac{100}{60} * t_r)\right] \qquad (25)$$

In this equation, $u(t)$ is the unit step function and we assume the output voltage starts to change at time t=0. In this chapter, we define:V_0 is the initial voltage, it equals to V_{dd} for a falling waveform, and 0 for a rising waveform. V_{100} is the final voltage, 0 for a falling waveform and V_{dd} for a rising waveform. Therefore, any voltage V_n can be presented as the following equation.

$$V_n = V_0 + \frac{n}{100}(V_{100} - V_0) \qquad (26)$$

Given the 50% delay point t_{50}, we can obtain a universal form in Equation 26.

$$V_{out}(t) = V_{out\,0}(t - (t_{50} - 0.5 * t_r * \frac{100}{60})) \quad (27)$$

Linear approximation is the straightforward way to approximate the output waveform at driving point, yet it is the roughest and can bring in much inaccuracy in timing analysis as the example in Figure 11 shows. In the following, we are proposing a more accurate way.

3.3.3.2 Piecewise linear approximation

As we know, the waveform of the gate output can be very complicated. In different regions, transistors behave very differently and the interconnection structure makes the waveform approximation much harder. In this subsection, we introduce a method that uses piecewise linear to model the output waveform. In our method, the waveform is divided into three segments: from the t_0 to t_{20}, we use a two orders polynomial equation to model the waveform; from t_{20} to t_{50}, the output waveform is assumed to be linear to the 50% point; from t_{50}, the transistor is fully turned on, so it's can be considered as a resistor with a resistance R. We use exponential function to approximate the last part of the waveform.

$$V_{out}(t) = \begin{cases} V_0 + bt + at^2 & 0 \le t \le t_{20} \\ k(t - t_{50}) + V_{50} & t_{20} \le t \le t_{50} \\ \sum k_i e^{p_i(t - t_{50})} & t \ge t_{50} \end{cases} \quad (28)$$

In this equation, t_x is defined as the time when the voltage reaches V_x. When the RC interconnection contains only one capacitor, the 3rd part of the Equation 28 only has one pole, which means only one exponential equation is needed to model the output waveform. Similarly, when connecting to a π-model load, there are two capacitors, so we need two exponential equations to model the 3rd part of the output waveform. We can

increase the number of exponential function in the 3rd part to get higher accuracy. In our experiments, two exponential functions can get an accurate approximation to HSPICE simulation results. We will use the timing information in the LUTs and load resistance and capacitance to calculate the parameters in Equation 28.

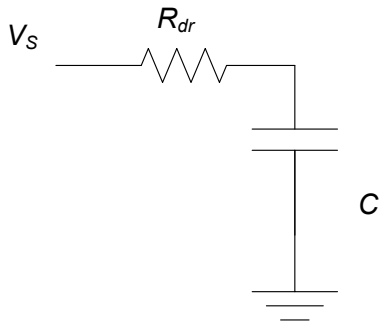

Figure 22. Equivalent circuit when gate is considered as a R_{dr}.

As shown in Figure 22, if the load is a single capacitor, we can get transition time t_r (t_{80}-t_{20}) and 50% delay from LUTs giving input transition time and load capacitance. Using initial voltage values at t_{50}, we can rewrite the 2nd and 3rd parts of Equation 28 as follows.

$$k * (t - t_{50}) + V_{50} \tag{29}$$

$$V_{50} * e^{-\frac{(t-t_{50})}{R_{dr}C}} \tag{30}$$

Because two parts of the waveform are consistent at t_{50}, the derivative of Equation 29 and Equation 30 are equal at t_{50}. Therefore, we can get:

$$k = -\frac{V_{50}}{R_{dr}C} \tag{31}$$

Since

$$k * (t_{20} - t_{50}) + V_{50} = V_{20} \tag{32}$$

$$V_{50} * e^{-\frac{(t_{80}-t_{50})}{R_{dr}C}} = V_{80} \tag{33}$$

Therefore,

$$t_{80} - t_{20} = \left(t_{50} - R_{dr}C \ln\frac{V_{80}}{V_{50}}\right) - \left(\frac{V_{20}-V_{50}}{k} + t_{50}\right) \tag{34}$$

Since t_r, which is (t_{80}-t_{20}), is provided in LUT, we can use Equation 31 and Equation 34 to obtain k and $R_{dr}C$. As for the parameters (a, b) in 1st part of Equation 28, we can use the voltage value and derivative at t_{20} to obtain them.

$$V_0 + bt_{20} + at_{20}{}^2 = V_{20} \tag{35}$$

$$b + 2at_{20} = k \tag{36}$$

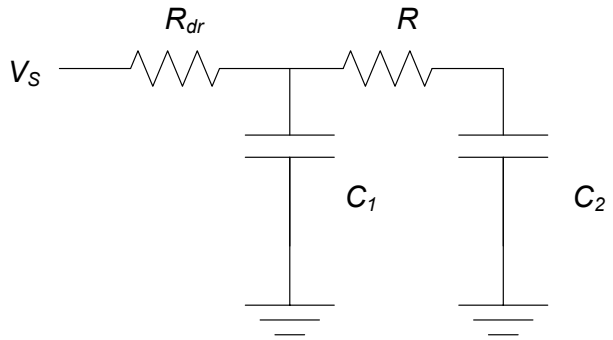

Figure 23. Equivalent circuit when gate is considered as a R_{dr}.

If the load is a π-model RC interconnection, the 3rd part of Equation 28 can be presents as sum of two exponential functions.

$$k_1 e^{p_1(t-t_{50})} + k_2 e^{p_2(t-t_{50})} \tag{37}$$

We use two poles approximation to model this 3rd part of the waveform. The transfer function over R_{dr} in Figure 22 can be presented as:

$$H(s) = \frac{1+RC_2 s}{1+(RC_2+R_{dr}C_1+R_{dr}C_2)s+R_{dr}RC_1C_2s^2} \tag{38}$$

We can obtain p_1 and p_2 by solving the denominator of Equation 38. The variable R_{dr} can be obtained from $R_{dr}C$ in the single capacitor case.

As for k_1 and k_2 in Equation 37, we can use the following two equations to obtain them.

$$k_1 + k_2 = V_{50} \tag{39}$$

$$k_1 p_1 + k_2 p_2 = k \tag{40}$$

For higher order approximations, we can use the similar initial conditions to calculate the k's and p's. Our experiments show that two exponential functions are enough to model the 3rd part of waveform at the driving point. Furthermore, conventional LUTs only provided the transition time from V_{20} to V_{80}, not the time when the waveform reaches V_{20} and V_{80}. If the LUTs can provide the time t_{20} and t_{80}, we believe that the results using our method can be more accuracy.

3.3 Propagation along Interconnection

Given any RC interconnection circuit as in Figure 24, the admittance of each node can be represented by its Taylor series expansion as shown in the Equation 9.If we use only the first 3 orders of the Equation 9, the admittance of each node can be represented as followed.

$$Y = y_1 s + y_2 s^2 + y_3 s^3$$

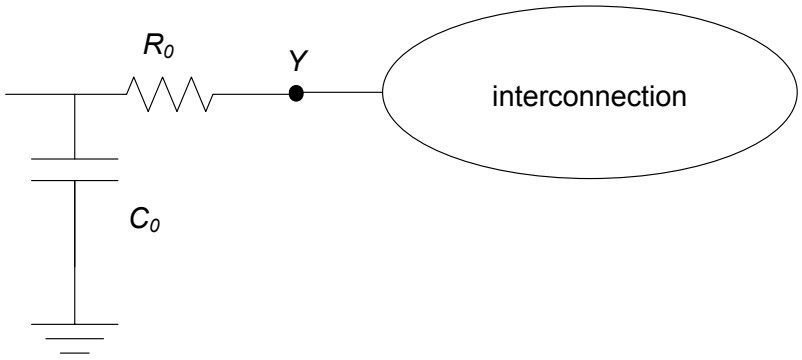

Figure 24. Signal propagates over interconnection.

Given this reduced expression of admittance, we are able to represent any RC circuit as a π-model connection or its simplified form. The values of the resistor and capacitors can be calculated by using Equation 16. The circuit in Figure 24 can be represented as the one in Figure 25.

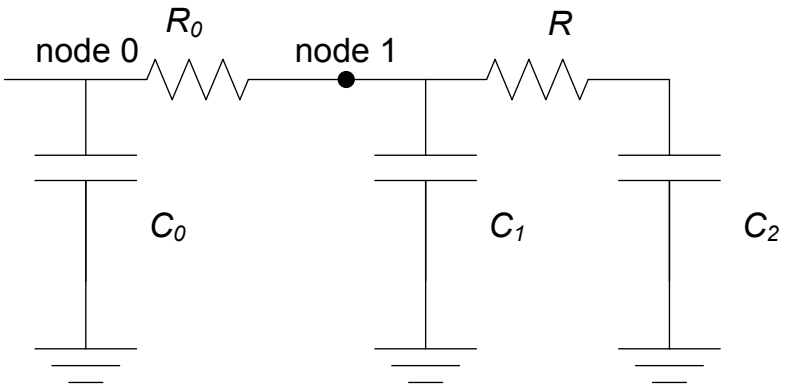

Figure 25. Reduced RC circuit.

In this reduced circuit, signal at node 1 can be easily calculated once knowing the waveform at input node 0.

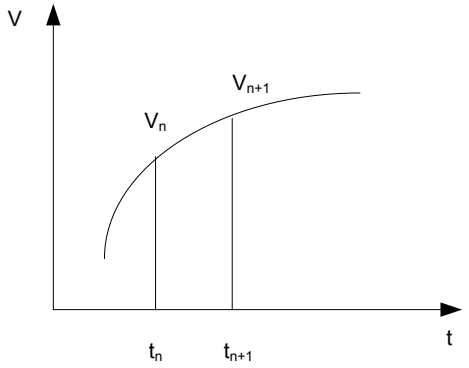

Figure 26. Waveform discretization.

Given a waveform as shown in Figure 26 at node 0, we can get following Equations.

$$V_{n+1} = V_n + \Delta t * \frac{dV_{n+1}}{dt} \tag{41}$$

$$\frac{dV_{n+1}}{dt} = \frac{1}{C} * \frac{dQ_{n+1}}{dt} = \frac{1}{C} I_{n+1} \tag{42}$$

$$V_{n+1} = V_n + \frac{\Delta t}{C} * I_{n+1} \tag{43}$$

$$I_{n+1} = \frac{C}{\Delta t} * V_{n+1} + \frac{C}{\Delta t} * V_n \tag{44}$$

$$I_{n+1} = G * V_{n+1} + I_n \tag{45}$$

Therefore, for each capacitor in the circuit, the capacitor can be replaced as a current source and a conductance as shown in Figure 27.

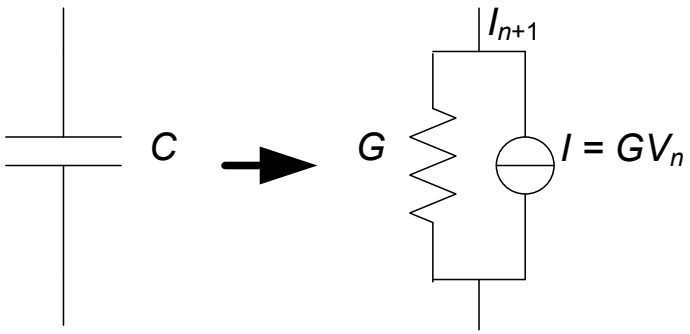

Figure 27. Replace capacitor with a current source.

G represents the current (n+1) part current, and current source I represents the part of C's current dependent on past voltage V_n. Besides, C_0 in the circuit can be omitted, as the waveform at node 0 is fixed and C_0 has no effect on the signal propagation.

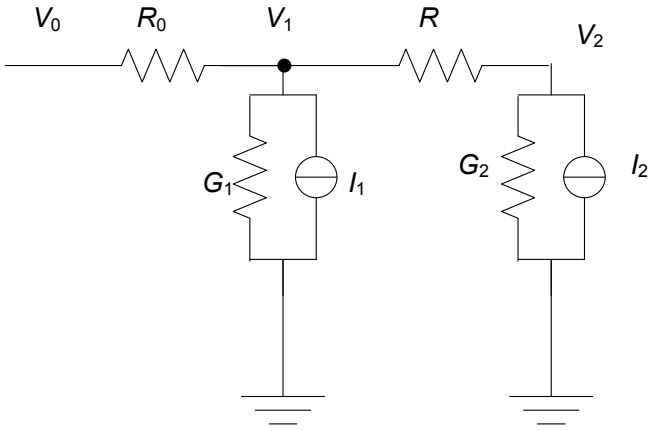

Figure 28. Modified RC circuit.

Using KCL at node 1, we can get

$$I_1 = \frac{V_1 - V_0}{R_0} + \frac{V_1 - V_2}{R} + G_1 * V_1 \tag{46}$$

At node 2, we can get:

$$I_2 = \frac{V_2 - V_1}{R} + G_2 * V_2 \tag{47}$$

Put Equation 46 and 47 together, we can get two linear equations with two unknown variables.

$$\begin{cases} I_1 = \left(\frac{1}{R_0} + \frac{1}{R} + G_1\right) * V_1 + \left(-\frac{1}{R}\right) * V_2 + \left(-\frac{V_0}{R_0}\right) \\ I_2 = \left(-\frac{1}{R}\right) * V_1 + \left(\frac{1}{R} + G_2\right) * V_2 \end{cases} \tag{48}$$

We re-write the equation 48 as follows.

$$\begin{cases} I_1 = G_{11} * V_1 + G_{12} * V_2 + A \\ I_2 = G_{21} * V_1 + G_{22} * V_2 \end{cases} \tag{49}$$

Since the V_0's waveform is already known, we can solve two equations,

$$V_1 = \frac{(I_1 - A) - G_{12} * V_2}{G_{11}} \tag{50}$$

$$V_2 = \frac{I_2 - \frac{G_{21}}{G_{11}} * (I_1 - A)}{G_{22} - \frac{G_{12} * G_{21}}{G_{11}}} \tag{51}$$

By setting a proper Δt, we can calculate the voltage value of node 1 at any time. Therefore, we can get the whole waveform at node 1, which means signal at node 0 is propagated to the node 1. However, the waveform at node 2 is meaningless in this method since node 2 is a pseudo node created for helping calculation. By doing this iteratively, we can propagate signal at driving point to any node at the RC interconnection.

3.4 Results

All of the transistors in this chapter were using 22nm Predictive Technology Model (PTM) provided by Arizona State University [20]. The LUTs of conventional format

were created by doing HSPICE simulation. The 50% delay and transition time (20% to 80% V_{dd}) were measured based on different input transition time and load capacitance.

Given a circuit in Figure 29, we input a rising ramp with a transition time of 40ps.

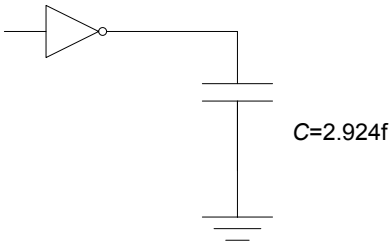

Figure 29. Inverter driving a single capacitor.

Figure 30 shows the comparison between the waveform from HSPICE simulation and the waveform of our model. Two waveforms are very close to each other. From initial conditions used in our model, the 50% point and transition time are equal for these two waveforms.

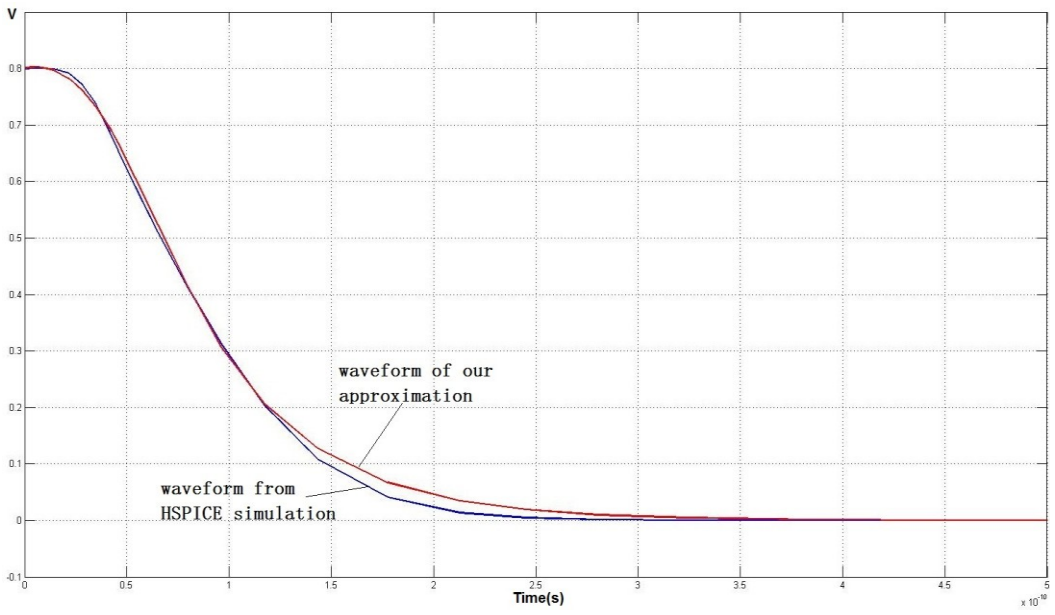

Figure 30. Comparison between our approximation and HSPICE waveform for a single capacitor.

To verify our π-model approximation, we used following circuit shown in Figure 31.

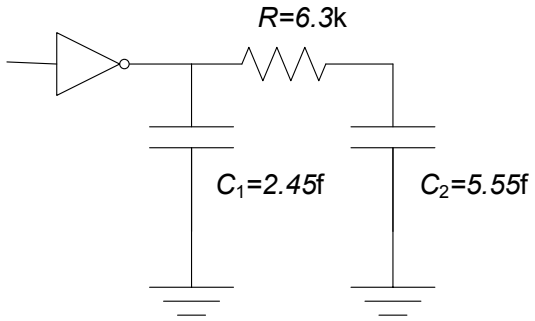

Figure 31. Inverter driving a π-model load.

Figure 32 shows the comparison between the waveform from HSPICE simulation and the waveform of our approximation. As the above result, the 50% delay point of our approximation is the same as the HSPICE simulation, so is the transition time.

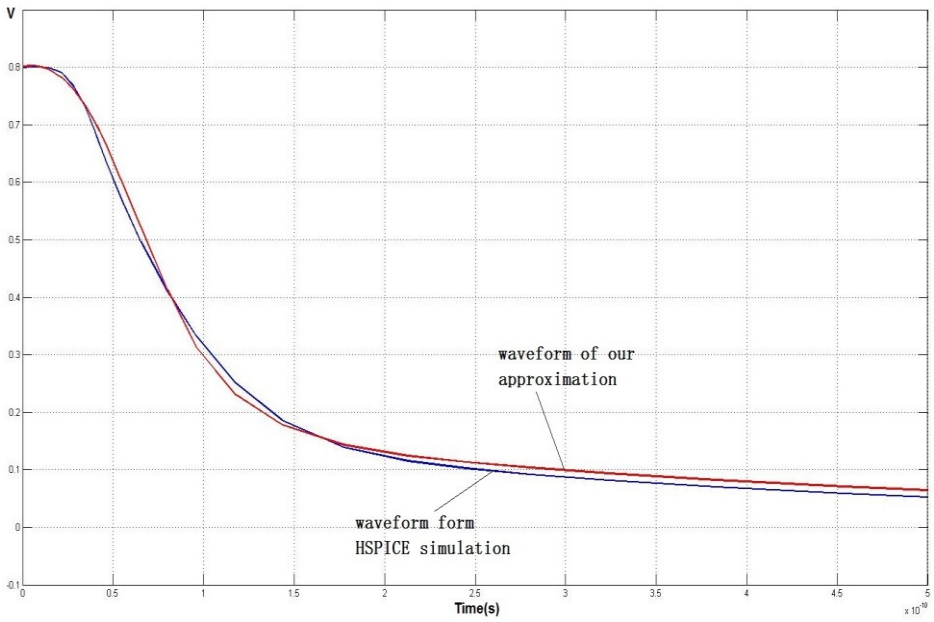

Figure 32. Comparison between our approximation and HSPICE waveform for π-model.

At same time, we implement the method in [26], and show the result in Figure 33.

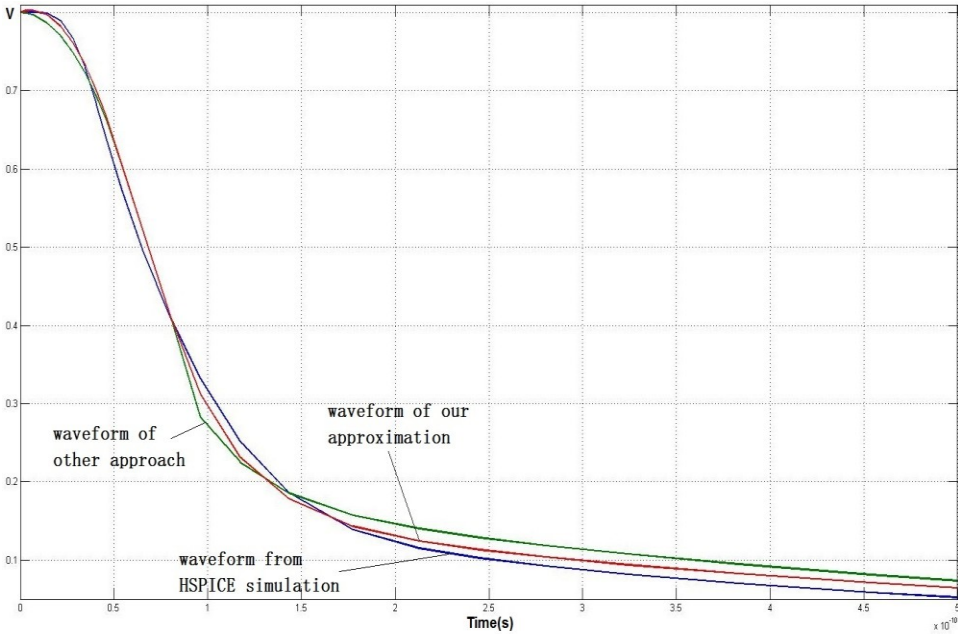

Figure 33. Comparison between different approximations.

As shown in Figure 33, our approximation (red line) is closer to the HPSICE results (blue line). The green line uses $c+at^2$ other than our $c+bt+at^2$ to approximate the waveform before t_{20}, and uses current equations other than voltage derivative equations to model the 3rd part of the waveform. In our experiments, using more initial conditions and more accurate model gets better results.

Table 3. Delay and transition time prediction for various library cells.

Cell	$t_{20}(ps)$		$t_{50}(ps)$		$t_{80}(ps)$	
	Approx.	HSPICE	Approx.	HSPICE	Approx.	HSPICE
Inverter	47.8	45.4	76.5	76.5	131	118
Nand2	47.3	44.9	75.5	75.5	130	116
Nor2	49.2	46.6	78.6	78.6	135	121

3.5 Summary

In this chapter, we proposed a complete method to approximate the waveform at the gate output. Merely based conventional LUTs, a method based on modified “effective capacitance” has been used to get one order approximation. Then, by using the delay and transition time got from LUTs, our method can obtain the waveform for any RC interconnection. The accuracy of our results can definitely improve the accuracy for the following timing analysis of signal propagation on RC interconnection.

Chapter 4

Equivalent Waveform Propagation for Static Timing Analysis

This chapter proposes a method to map diverse input waveforms to effective saturated ramps, which can be used to in the static timing analysis (STA) of VLSI circuits. Conventional STA of VLSI circuits rely on timing lookup table (LUT) which is obtained during library characterization. In LUT creation, each logic gate with various output loads is driven by a range of input ramps, and transition time is the only attribute to describe the input and output waveform. However, transition time is not enough to model the waveform for various waveforms with the same transition time. We propose a new method which can be used to calculate effective ramp for any given input waveform, then use effective ramp attributes (transition time and arrival time) to calculate gate delay and output transition time. This method uses the current STA LUTs, requires no change to library characterization procedure, and only needs a little additional computation; thus, it can be easily implemented in conventional STA tools.

4.1 Introduction

Table Lookup methods are common approaches in static timing analysis (STA) of VLSI circuits [1]. Nowadays, STA is largely used in VLSI industry to verify timing constraints. Furthermore, it is the only feasible way to perform full-chip timing analysis

[31] due to its low computation cost. However, with the down scaling of feature size of integrated circuit, the waveform is easily affected by circuit electrical effects, such as resistive shielding, crosstalk noise, wire inductance, etc. The assumption that one transition time is enough to describe the waveform shape in conventional LUT methods will introduce unacceptable estimation errors.

Conventional LUTs are two-dimensional tables, and each logic gate in the library usually has two tables: one storing the gate delays and the other for transition times of output waveforms. Due to the simulation cost when creating LUTs, only transition time of input waveform and load capacitance are used as parameters to look up delay and output transition time in LUTs. Usually, the delay is defined as the time from 50% point of an input waveform to 50% point of an output waveform, while the transition time of one waveform is defined as the time from 20% voltage point to 80% voltage point.

We have observed that a single transition time is not enough to capture the influence of waveform shape on the gate delay and output waveform. In real circuits, signal waveforms can have various shapes, and the waveform propagated in the circuits is not always close to a ramp due to the circuit electrical effects. Various waveforms may have same transition time, but obviously have different gate response.

Furthermore, using 50% point of the voltage waveform may not always be a good choice for gate delay calculation. As the signal can be affected by the noise, or race and hazard may happen in propagation, signal is not always a perfect rising/falling waveform, and signal voltage may pass 50% point multiple times. Arbitrarily choosing one 50% point may be a straightforward way, but definitely not an optimal solution.

In this chapter, we propose a systematic method that is capable of mapping any input waveform to an effective ramp. We can then use this ramp to obtain an output waveform that is closer to actual one. This method adds a little additional computation before look up library step in STA, and makes no change to the current LUT format.

The remainder of this chapter is organized as follows. Section 2 discussed how the electrical effects affect the STA accuracy and previous work. Our solution for ramp approximation is described in Section 3. The modeling validation results by experimenting over a number of process technologies are presented in Section 4. Finally, we conclude our work in Section 5.

4.2 Electrical Effects and its Influence to Waveform

4.2.1 Resistive Shielding Effect

Signal on a circuit is transferred from one node to another through logic gates and interconnections (wires). Due to the resistance along the wire, the capacitance at driven point will seem quite different from a single lumped load capacitance. Let us take the two circuits in Figure 34 as examples. Although total load capacitance values for the two circuits are the same (i.e., both are 4fF), the slopes of two output waveforms are quite different.

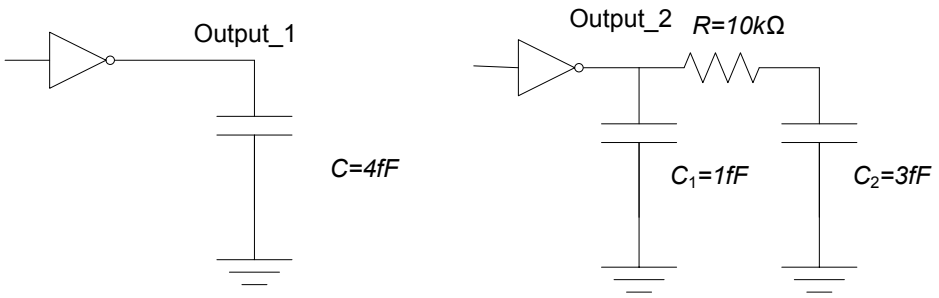

Figure 34. Resistive shielding effect.

Due to the resistive shielding effect, the beginning of the output waveform is steeper, and the tail part gentler as shown in Figure 35. Therefore, the waveform becomes more different from ramp-like shape, which clearly shows that a single attribute (transition time) is not able to capture the shape of waveform, thus making ramp approximation less accurate.

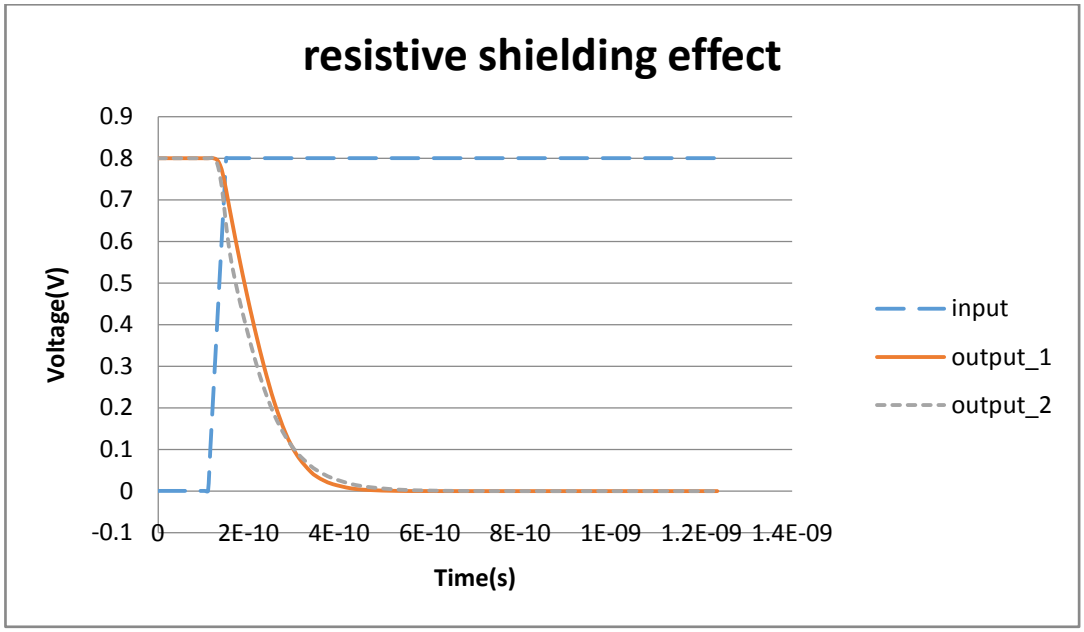

Figure 35. Waveform comparison of resistive shielding effect.

4.2.2 Noise and Race Hazard

The noise and race hazard make the waveform more complicated. The complication of waveform can affect the accuracy of timing estimation in two aspects. Firstly, the same as resistive shielding effect, the waveform is not ramp-like shape any more. Using transition time alone cannot capture the shapes of waveforms. As shown in Figure 36, the waveforms with noise (blue) and the ramp (red) are of same transition time; however the output waveforms of two cases are different. Secondly, the complication of waveform can make delay definition very difficult. As shown in Figure 36, the waveform with noise passes 50% voltage point twice, which means there are two different delays according to the delay definition.

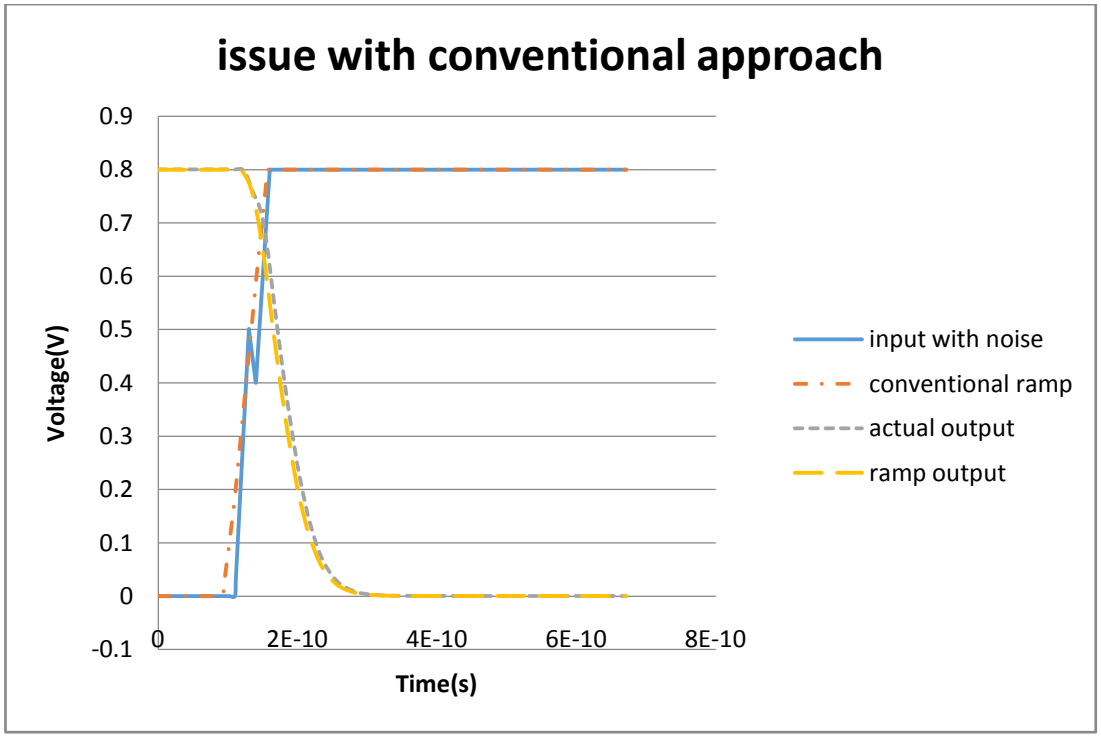

Figure 36. Complicated waveform.

Due to these effects described above, the conventional static timing tools need to handle these increasingly complicated waveforms to retain the required accuracy. Some of the solutions use much advanced models to propagate waveforms through the logic gates, e.g., CSM VIVO models [33, 34, 35]. These solutions model the gate behaviors based on currents, and retain relatively higher accuracy as well as higher computation complexity. Some solutions use more attributes rather than a single transition time to capture the shape of waveforms [32], which requires many changes to the prevailing static timing tools. Our solution attempts to find a close ramp-like waveform to approximate a given arbitrary waveform such that the accuracy of timing analysis can be retained. At the same time, our solution is based on conventional LUT methods: only a little extra computation is needed to allow the prevailing tools to handle complicated waveforms.

4.3 Waveform Approximation

4.3.1 Ideal Solution and Practical Solution

Based on the above discussion, our problem is then formulated as: Given an arbitrary input waveform, how to find a ramp that has an output waveform closest to the actual one? If we denote the actual input waveform as $v_{in}(t)$, and its output waveform as $v_{out}(t)$, what we need is to find a ramp $r(t)$, and its relative output $v'_{out}(t)$ to minimize the Equation 52.

$$F1(t) = \int_{-\infty}^{+\infty} |v_{out}(t) - v'_{out}(t)| dt \qquad (52)$$

Here, we select transition time and arrival time of $r(t)$ in order to get minimum $F1(t)$. However, it might overly simplify the problem. Practically, we have to face several

problems that we cannot avoid. $V_{out}(t)$ is the actual output which is unknown and we need to estimate. Furthermore, standard LUT only provides transition time and arrival time (50% voltage point) of $v'_{out}(t)$, rather than the whole waveform. It seems that this method does not work, but it helps us to find a practical approach. We propose to use $F2(t)$ in Equation 53 instead of $F1(t)$. If we can find a minimum value of $F2(t)$ in Equation 53, we can claim that $F1(t)$ is very close to its minimum value. This is very straightforward, since two output waveforms must be the same when we can find a waveform that is the same as actual input waveform.

$$F2(t) = \int_{-\infty}^{+\infty} |r(t) - v_{in}(t)|\, dt \tag{53}$$

Since $v_{in}(t)$ is the given input waveform and $r(t)$ is the waveform we adjust, solving our problem becomes finding minimum value of object function $F2(t)$, which is feasible.

4.3.2 Integration Region

In equation 53, the integration region can be narrowed. We denote the time when a waveform starts to increase/decrease as T_s, and the time to stop as T_e. Since the time period in which voltage keeps changing is the critical period, we only need to minimize $F2(t)$ in the region [T_s of $v_{in}(t)$, T_e of $v_{in}(t)$].

Furthermore, the output waveform is affected by not only input waveform but also the gate characteristic and load capacitance. In order to account for these issues, we bring in the output waveform shape to optimize our calculation. At very beginning of input waveform, the input voltage is less than the threshold voltage of transistors, and thus the variation of input voltage has little effect to the output waveform. It means that we need to choose the time when the gate starts to switch (output waveform starts to increase/decrease) as the starting time of the integration region. The timing information

of output waveform is transition time and arrival time from LUTs. Therefore, we use $r(t)$ and load capacitance as parameters to get $r(t)$'s relative output waveform, and use the waveform to select a critical region for integration. Usually, the region is [T_s of $v'_{out}(t)$,T_e of $v_{in}(t)$] as shown in the Figure 37.

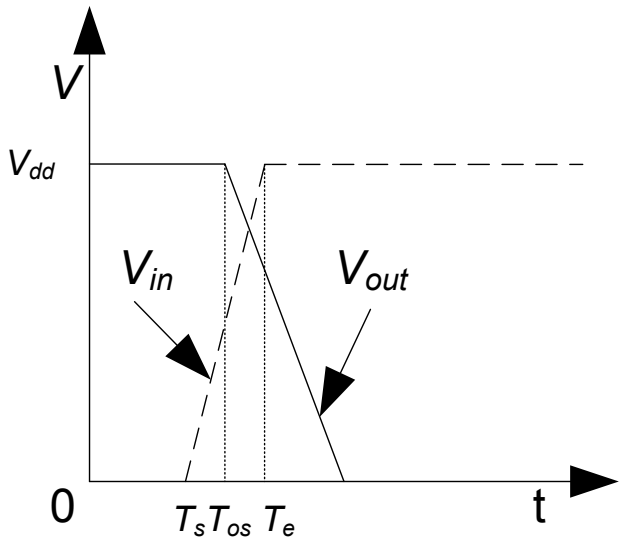

Figure 37. Critical region for integration.

Therefore, the Equation 53 can be re-write as

$$F2(t) = \int_{T_{os}}^{T_e} |r(t) - v_{in}(t)| dt \tag{54}$$

4.3.3 Proposed Heuristic Method

As described above, we use the following heuristic method to find the effective ramp for an arbitrary waveform.

```
//Heuristic method for equivalent ramp calculation
Ramp_Approx(v_in(t))
{
    set initial arrival time as the arrival time of v_in(t);
    set initial transition time as the transition time of v_in(t);
    get relative output waveform v'_out(t);
    update integration region;
    calculate the initial value diff of Equation 3;
    set function accuracy min_step;
    set approaching step as 2*min_step;
    While(step>= min_step)
    {
       new transition time  = transition time + step;
        get new v'_out(t);
        update integration region;
        calculate the value diff of Equation 3;
        if(new diff<= diff )
        {
            step = 2*step;
            transition time = new transition time;
            diff = new diff;
        }
        else
            step = step/2;
    }
    set approaching step as 2*min_step;
    While(step>= min_step)
    {
        new arrival time= arrival time + step;
        get new v'_out(t);
        update integration region;
        calculate the value diff of Equation 3;
        if(new diff<= diff )
        {
            step = 2*step;
            arrival time = new arrival time;
            diff = new diff;
        }
        else
            step = step/2;
    }
    return  arrival time, transition time;
}
```

Figure 38. Pseudo code for equivalent ramp calculation.

As shown in the function, we use the conventional approximation method to initiate our transition time and arrival time, and then use two loops to find new transition time and arrival time to minimize Equation 3. The step in the function is the accuracy that we set. We increase the search step when there is still room for improvement, and decrease it when there is none. In this way, our function can locate the suitable transition time and arrival time fast.

4.4 Results

All of the transistors in logic gates were using Predictive Technology Model (PTM) provided by Arizona State University. The LUTs of conventional format were created by doing HSPICE simulation. The 50% delay and transition time (20% to 80% V_{dd}) are measured based on different input transition time and load capacitance.

Given a circuit in Figure 39, a falling waveform (blue line in Figure 40) is applied to the input. We compare our ramp approximation method with the conventional one.

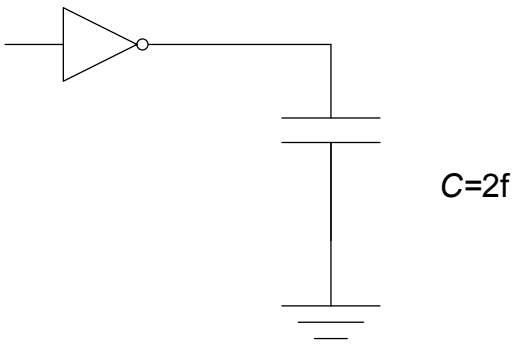

Figure 39. Inverter driving a single capacitor.

Figure 40 shows the comparison between conventional approach and our approach. The ramp (red line) got from conventional method has the same transition time as the original waveform and pass the 50% voltage point. The green line in the Figure 40 is the

ramp obtained from our method. Then we respectively apply these two ramps to the same circuit as in Figure 39. From the Figure, we can tell the output waveform obtained from our approach (green line) is closer to the actual one (blue line).

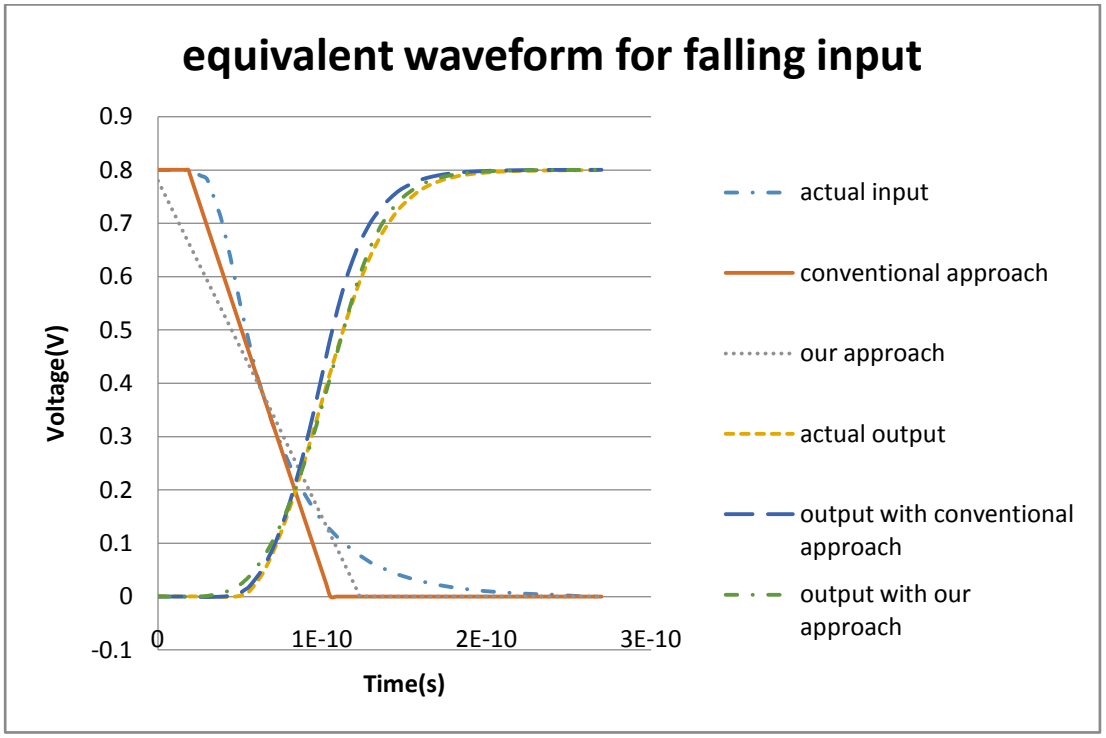

Figure 40. Waveform comparison between our approach and conventional approach (falling input) .

At the same time, we use a rising waveform of higher transition time as the input waveform. The similar result is given in the Figure 41.

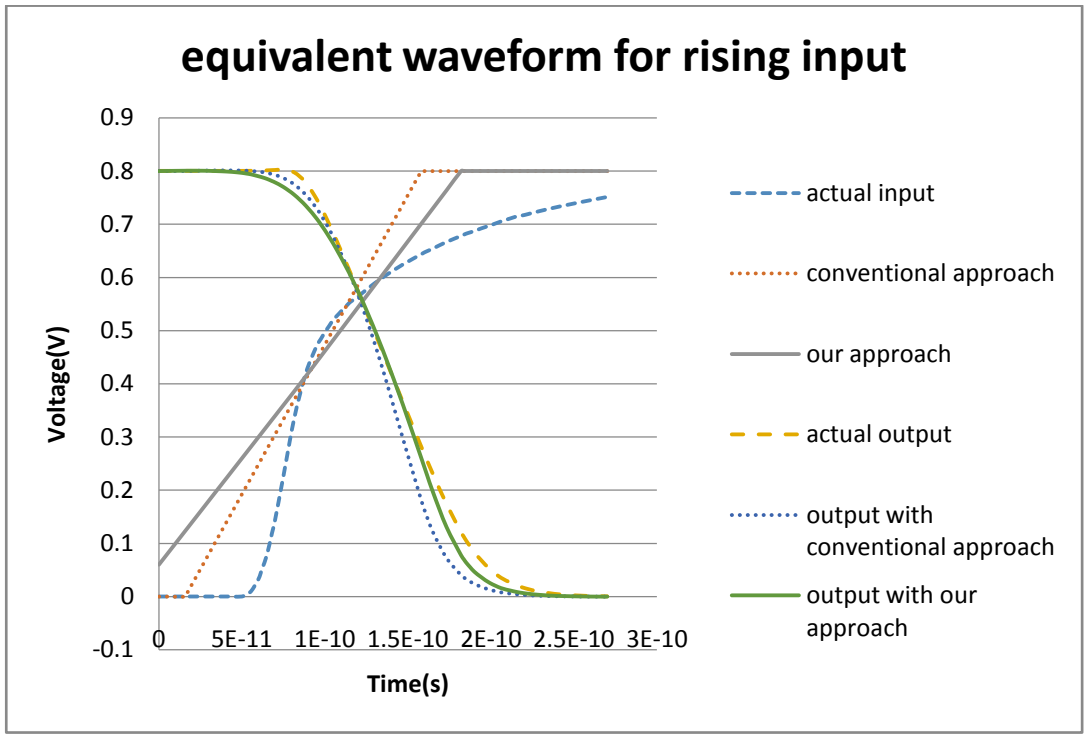

Figure 41. Waveform comparison between our approach and conventional approach (rising input).

To verify our approach over a serial of basic logic gates, we use the following circuit in Figure 42 to compare our results with the conventional ramp approach. In the experiment, we test inverter, nand2 and nor2 gates with a process technology of 22nm high performance model from Arizona State University [20].

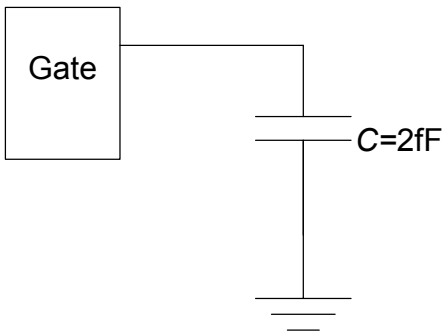

Figure 42. Test the method for various logic gates and process technologies.

In the experiment, we choose two actual waveforms as shown in Figure 40 and 41 to drive basic logic gates: one is a slow rising waveform and the other is a fast falling waveform. Table 4 shows both of the conventional and proposed ramp approximation results for the slow input waveform.

Table 4. Input waveform comparison on arrival and transition time (22nm, rising waveform).

Cell	Conventional(ps)		Proposed(ps)	
	Transition time	Arrival time	Transition time.	Arrival time
Inverter	86.9	86.7	118.1	84.9
Nand2	86.9	86.7	118.1	84.9
Nor2	86.9	86.7	118.1	84.9

Table 5 shows the transition time and arrival time of output waveforms when using conventional approach, proposed approach and actual waveform respectively. From the table, results of the proposed waveform are closer to the actual output which means using our approach can get higher estimation accuracy.

Table 5. Output waveform comparison on arrival and transition time (22nm, rising waveform).

Cell	Conventional(ps)		Proposed(ps)		Actual(ps)	
	Transition time	Arrival time	Transition time.	Arrival time	Transition time.	Arrival time
Inverter	50	137	61	142	65	142
Nand2	50	132	60	135	59	137
Nor2	50	137	61	142	65	142

When given a fast input, the results are shown in table 6 and 7. Our approach also gets better results than the conventional one. From the results, we can claim that our approach is suitable for both of rising and falling waveforms.

Table 6. Input waveform comparison on arrival and transition time (22nm,falling waveform).

Cell	Conventional(ps)		Proposed(ps)	
	Transition time	Arrival time	Transition time.	Arrival time
Inverter	52.1	61.8	75.8	60.7
Nand2	52.1	61.8	75.8	60.7
Nor2	52.1	61.8	75.8	60.7

Table 7. Output waveform comparison on arrival and transition time (22nm,falling waveform).

Cell	Conventional(ps)		Proposed(ps)		Actual(ps)	
	Transition time	Arrival time	Transition time.	Arrival time	Transition time.	Arrival time
Inverter	41.7	98.8	48.8	104	50.5	103
Nand2	41.6	99.8	49.4	105	50.5	104
Nor2	51	142	61	148	67	148

4.5 Summary

In this chapter, we proposed a method to find an effective ramp for given arbitrary waveform. The simulation results showed that our method obtained much higher accuracy than the conventional approach in timing analysis. Merely based on conventional LUTs, our method required no changes to the prevailing static timing model, which means our method is completely compatible with current STA tools. Furthermore, our approach is useful in reducing the effects from cross talk noise and interconnection electrical effects.

Chapter 5

Glitch Elimination Method for Low Power Function Unit Design

In logic circuits, because of glitches and hazards, gates can switch multiple times before the circuit reaches stable state. Gates' extra switching increases circuit dynamic power dissipation, especially in calculation circuits. In this chapter, a buffer insertion method is proposed to synchronize signal arrival time so as to reduce glitches. We implement this method on several calculation circuits. Based on HPSICE simulation results, spurious switching activities are eliminated and new designs consume 20%~80% less power as compared with conventional circuits. In addition, critical paths are not modified in order to retain circuit's timing performance.

5.1 Introduction

Because of the prevalence of portable electronic devices and limitation of battery technology, power dissipation has become one major concern in modern CMOS circuit design. Reducing devices' dynamic power dissipation could extend the devices' operation time, for that the dynamic power is dominant component in overall power dissipation of CMOS circuits, especially in function units for calculation operations.

Dynamic power dissipation results from signal transitions, as represented in Equation 55: VDD is the supply voltage, C is the effective load capacitance, f is clock frequency and N is the signal transition within a clock cycle.

$$P_{dynamic} = \frac{1}{2} V_{DD}^2 * C * f * N \quad (55)$$

Based on the dissipation sources, some approaches lower the supply voltage and operating frequency to reduce the dynamic power dissipation. [38] described a dynamic voltage scaling (DVC) technology to lower the supply voltage of logic blocks on non-critical path dynamically to minimize the power consumption. In [40], the authors developed a low power processor by using dynamic voltage and frequency scaling (DVFS) and "adaptively assigned breaking-off condition" algorithms. In [41], authors provide a configurable multiplier, which can work in low power mode by lowering the supply voltage and frequency. However, these methods require hardware support for DVFS and efficiency of power reduction depends on the hardware support. Furthermore, lowering supply voltage or frequency could compromise circuit performance.

Another direction to reduce the dynamic power is to decrease signal transitions. Signal transitions fall into two categories: functional transitions and spurious transitions

(glitches). The first one is logic transition desired for gates' functioning while the glitches are unnecessary transitions before signals reach stable. These glitches account for 30~70% of overall power dissipation, which should be eliminated to save energy. To reduce the logic transitions, low power multipliers of new architectures are provided in [42-44], and both timing performance and power saving are considered. However, these methods are merely based on specific type of multipliers and the circuit logic is largely modified. Thus the methods can hardly be adopted in common circuits. Meanwhile, various approaches are proposed in [45-50] to reduce glitch power. [45] uses filtering hazard pulses to reduce circuit glitches. This technique increases the gate size, which means more power consumption, thus reduces the power saving. Besides, real delay highly depends on signal arrival time and waveform, changing gate size to eliminate the hazard is very limited. In [46], author uses a simple delay circuit to generate an enable signal to synchronize the operation of each logic stage. Though glitches are eliminated, circuit timing performance is largely reduced. In [47], author insert a compensation circuit with certain delay to balance the input signal arrival time during physical design stage based on IR drop numbers. This method is very straightforward, but circuit inner glitches could not be eliminated. In [48], author uses the wire routing to modify library cell designs to get glitch free designs. It cannot be applied in the circuit level. In [49], a gate freezing method to minimize the glitch power is provided. However, it requires designing new F-gate rather than using original library cells, this method is not flexible for re-spin designs [47], same problem also happens in [50]. Although the method in [50] could solve the optimal problem by using linear programming technology, the delay and dynamic power model are too simple to be applied to real circuits. Furthermore, the method cannot avoid

inserting buffer along critical paths. In [51], author use gate freezing, gate sizing and buffer insertion to reduce the dynamic power, but critical paths would also be modified in the method, not to mention that heuristic method is timing consuming and has no promise for the global optimal solution.

In this chapter, glitch source and elimination method are first discussed, then our method is described in details. Our work is dedicated calculation function units designs based on full adders. Based on Static Timing Analysis (STA) method, glitch power is estimated. And we choose certain circuit positions to insert buffers to eliminate glitches, thus reducing dynamic power. Our method neither changes the circuit logic nor alters the circuit timing performance. The complexity of our method is linear with gate number in circuit.

5.2 Glitch Elimination Method

5.2.1 Glitches Source

In circuit, glitches are unnecessary signal transitions, waste energy and increase dynamic power dissipation. Signals' different arrival time generates glitches, and signal propagation would make spurious transitions worse.

When input signals have different path delay (DPD) and the DPD is larger than gate's inertial delay, glitches are generated at gate's output as shown in Equation 56.

$$DPD > D_{inertial} \tag{56}$$

Take the circuit in Figure 43 for example: The inverter has 1 unit delay, AND gate has 2 units delay and 3 signals arrive at primary inputs at t=0. Because the inverter is on the path to the upper input of "and1" gate, two input signals arrive at the "and1" gate through paths with different path delay. Based on STA, the signal to the lower input makes output signal of "and1" to transit from '0' to '1' at t=2. Then, the signal to the upper input causes the output signal to transit back to '0' at t=3. Due to the different path delay from primary inputs to the output of "and1" gate, a glitch is generated.

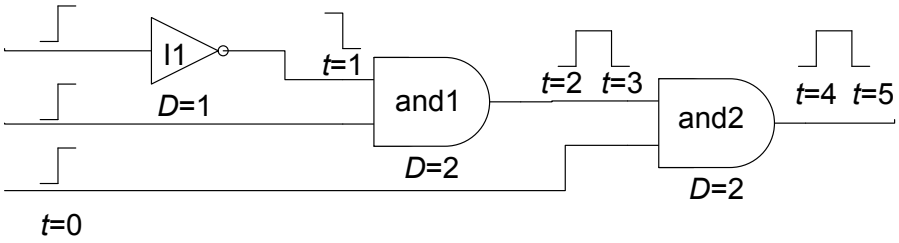

Figure 43. Different signal arrival times generate glitches; signal propagation makes glitches worse.

Furthermore, if the glitch width is larger than gate's inertial delay, it would be propagated to the following circuit stage, which causes more unnecessary transitions. As in Figure 43, the glitch generated by "and1" gate, causes a new glitch at the output of following "and2" gate.

In conclusion, there are two glitch sources:

If a gate G has inertial delay $d(G)$, with n fan-ins n_1, n_2, … , n_n the DPD_{ij} is the different path delay of *ith* and *jth* fan-in nodes. When $DPD_{ij} > d(G)$, glitches are generated at the fan-out nodes of gate G.

As defined in [52], a glitch window gw_i at node i is the width of a timing window, that is, a time interval bounded by the earliest and the latest signal arrival times. When signal glitch width of fan-in nodes is $gw(n_1)$, $gw(n_2)$, … $gw(n_n)$ respectively and $gw(n_i)>d(G)$, the glitches on the *ith* signal would be propagated through gate *G*.

5.2.2 Transition Activity Calculation

Since glitches come from DPD and signal propagation, we can use static timing analysis (STA) to estimate the path delay, and then get transition times of circuit nodes. Specifically, in the full-adder-based (FA-based) calculation circuits, DPD of inputs signals to each FA could cause many sum and carry-out signal glitches. In the example shown in Figure 43, primary signals have seven different paths to the net 'sum_{11}'. Suppose that each FA has the same fixed delay *d* to its two output nodes, the seven paths to node sum_{11} could have two different delays:

$$d1 = d2 = d6 = d4 = d5 = d7 = 2 * d;$$

$$d3 = d;$$

Because of DPD, the signal at node sum_{11} could transit twice at time *d* and 2*d*, while required functional transition is merely once. Transition happens under the simplest delay model: each FA has the same fixed delay to its two outputs, but the real timing model is more complicated, which means transition would be more than twice. In the following sections, we will use fixed delay for transition activity calculation, but our method could also be used for any static timing models.

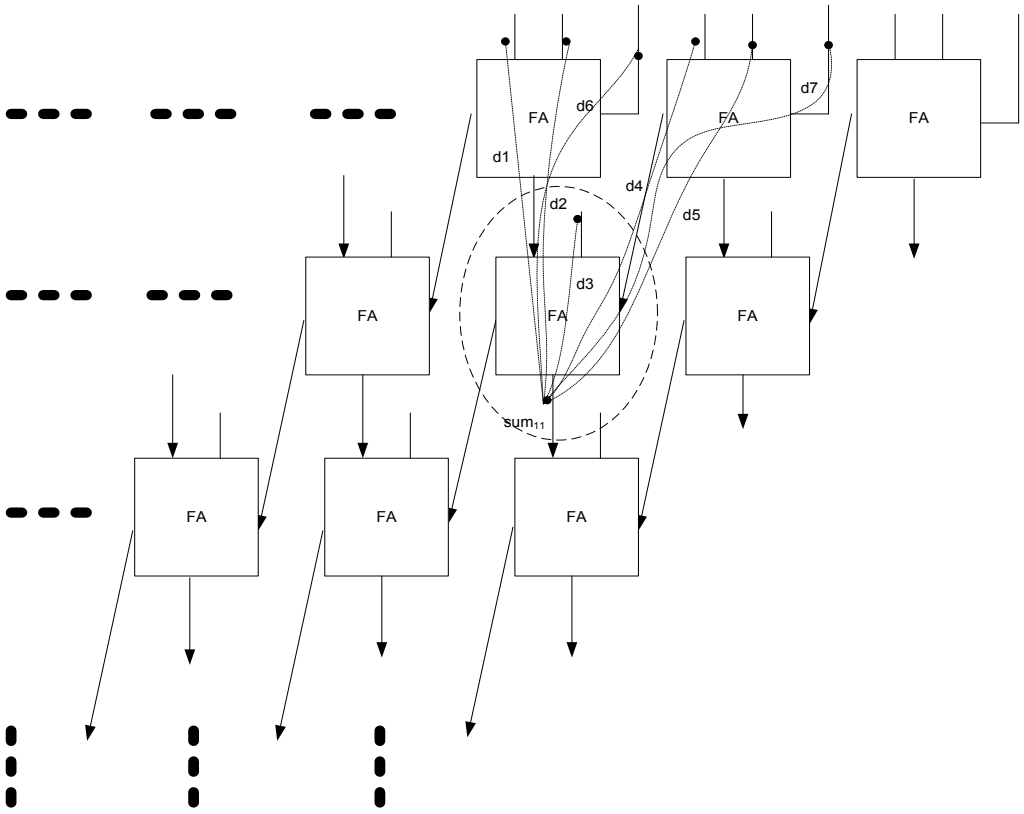

Figure 44. Array multiplier.

In the calculation circuit, suppose a node i has n different paths p_1, p_2, … , p_n to circuit primary inputs, the transition time $TR(i)$ of node i could be represented as:

$$\text{number of different path delay}\{d(p_1), d(p_2), \dots, d(p_n)\}$$

or

$$\text{number of signal arrival times at node } i$$

In above array multiplier example, though node sum_{11} has seven paths to the primary inputs, number of different path delays is two, therefore the transition time of node sum_{11} is two.

Via STA, we could get signal arrival times of each node, and transition times by counting the number of arrival times. In Figure 45, three input nodes have arrival times: {0}, {2*d*, 3*d*}, {3*d*, 4*d*} respectively, and the FA has a fixed delay *d*, so we could calculate the output nodes arrival time as follows :{*d*, 3*d*, 4*d*, 5*d*}.

$$d = 0 + d; \qquad 3d = 2d + d;$$

$$4d = 3d + d; \quad 5d = 4d + d$$

Therefore, the output node of the FA could have transition at time: *d*, 3*d*, 4*d* and 5*d*. In this propagation way, we could get signal transition activity of all nodes in the given FA-based circuit.

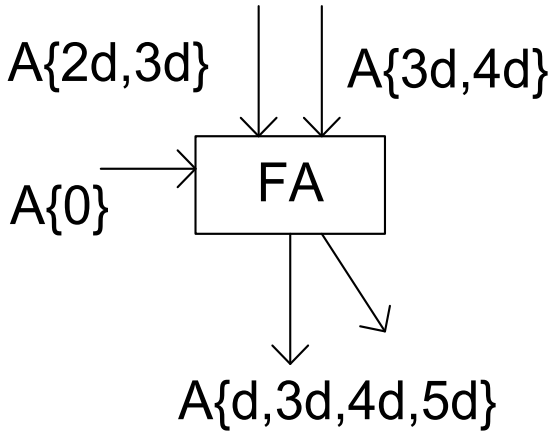

Figure 45. Arrival time propagation example.

Given a FA-based calculation circuit *FU*(*G*,*V*), where *G* is the set of all gates in *FU*, and *V* is the node set, including primary input, primary output and interconnection. Let S_i be the *ith* node arrival time set, the transition activity calculation of each node is described as follows:

```
//Function for counting the circuit transition
Transition_Activity_Calculation()
{

    foreachprimary_input_node v_i
        S_i = {0};
    end

    sort gate in order of logic depth from primary inputs;

    store gates in a array GA;

    foreach gate g_i in the GA

        set_temp = ∅;

        foreach node v_j in g_i's input nodes
            set_temp = set_temp ∪ S_j;
        end

        foreach node v_k in g_i's output nodes
            S_k = {d_l + d(g_i) | d_l ∈ set_temp };
        end

    end
}
```

Figure 46. Pseudo code for circuit activity counting.

After getting transition activity of each node, we can use the following function to calculate the max transition time for a given circuit *G*:

```
//Function for counting maximum transition of the circuit
Max_Transition_Time()
{

    num = 0;       // store the total transition

   foreach v_i in G
      num+= numberof S_i elements;
   end

}
```

Figure 47. Pseudo code for counting maximum transition of circuits.

The following table shows maximum transition time and functional transition of several FA-based circuits.

Table 8. Transition counting for FA-based circuit.

circuit	max transition	functional transition
8-bit RCA	72	16
8-bit array multiplier	780	144
6-bit low-power multiplier	220	60
32-bit low-power multiplier	33,696	1,984
64-bit low-power multiplier	265,860	8,064
16-bit non-restore divider	4,160	128

5.2.3 Elimination Method

As mentioned above, DPD generates glitches and glitch will be propagated to the following stages. Therefore, to eliminate the glitches, our goal is to avoid glitch generation and propagation, thereby reducing dynamic power dissipation.

As shown in Equation 56, since glitches generation is due to the DPD, both inside and outside full adder, we insert buffers to introduce certain delay to synchronize signals' arrival time to make sure that DPD is less than the gate's inertial delay.

Via transition activity calculation method in former subsection, we calculate the possible arrival times set S_i of each node, and we define the largest element in set S_i of primary outputs as required arrival time. Obviously, the largest required arrival time is the delay of circuit critical path. From the primary outputs to primary inputs, we use the following method to calculate the required arrival time for each node. Let *rt* be the required arrival time and RT_x is the set of *rt*s at node *x*.

```
//Function for arrival time calculation
Required_Arrival_Time_Calculation ()
{

    foreach node v_i in primary outputs
        RT_i = RT_i.add(max d in S_i);
    end

    sort gate in order of logic depth from primary outputs;

    store gates in a array GA;

    foreach gate g_i in GA

        foreach node v_j in g_i's input nodes
            RT_j.add(max rt in RT_i − d(g_i));
        end

    end
}
```

Figure 48. Pseudo code for calculating required arrival time.

Using Required_Arrival_Time_Calculation(), we can calculate each node's required arrival time. Specially, each RT_i merely contains one required arrival time except nodes along broadcast lines because they connect to gates with different logic depth. In the following paragraphs, we are going to describe the method how to insert buffer along broadcast lines to realize the required arrival times.

```
//Function for buffer insertion procedure
Buffer_Insertion ()
{
    foreach RT_i that number of elements > 1

        case 1 (all rt's in RTi are identical)
        {
            insert a buffer with delay d;
            ( d = rt_i )
        }

        case 2 (all rt's in RT_i are different)
        {
            insert a buffer array with delays d_i;
            d_1 = rt_1;
            d_2 = rt_2 − rt_1;
            d_3 = rt_3 − rt_2;
               …

            ( rt_1 < rt_2 < rt_3 < ⋯ )
        }

        case 3 (node i is not primary input )
        {
            insert a stack buffer with 'en' at max rt;
        }
    end

}
```

Figure 49. Pseudo code for procedure of buffer insertion.

As for calculation circuit, there are many nodes connecting FA units of different logic depth. Our method to eliminate glitches is to insert buffers at signal broadcast lines to avoid glitches generation and propagation. The reasons why we only insert buffers along the broadcast lines are as followed:

Trade-off concern: Inserting buffers reduces the glitches, which means reduce dynamic power dissipation. However, introduce of additional circuit also increase the power dissipation, both dynamic and leakage. In order to reduce overall power dissipation, we choose to insert buffers merely along the broadcast net. Furthermore, more inserting means larger circuit area penalty and we need to add limited circuit in order not to increase the circuit area much.

Broadcast nets cost more dynamic power dissipation. The Load capacitance of broadcast nets is usually high due to their very high fan-out. According to the Equation 55, larger the load capacitance is, higher dynamic power each transition can consume. Therefore, avoid glitches along the broadcasting lines could largely reduce the dynamic power dissipation

Broadcast nets connect to more gates: The glitches would be propagated to more gates because of the high fan-out property to broadcast nets. These propagated glitches would cause more glitches in the following nets, which cause more power dissipation. A buffer of scheduled working could block the glitches' propagation, which means less dynamic power dissipation in the following nets.

Signal quality. Usually, the load capacitance for the broadcast signal is high to drive. Inserting a proper buffer could increase the broadcast signal's drive ability, thus make sure the quality of the signal waveform. Inserting a proper buffer could improve the signal quality without introducing delay penalty.

In one word, DPD due to the signal broadcasting is the main source of glitches, and only inserting buffers along the broadcast lines could largely reduce overall dynamic

power with little area penalty. The following table shows the theoretical transition times after buffer insertion along the broadcast lines.

Table 9. Transition counting for FA-based circuit after buffer insertion.

Circuit with buffer insertion	max transition	Reduced transition
8-bit array multiplier	144	81.54%
6-bit low-power multiplier	60	72.73%
32-bit low-power multiplier	1984	94.11%
64-bit low-power multiplier	8064	96.97%
16-bit non-restore divider	288	96.92%

5.2.4 Full Adder Internal Circuit

Besides, FA itself is a digital circuit and DPD of signals inside FA design could cause glitches too. Usually, the sum operation is realized by $A \oplus B \oplus C$, as shown in Figure 50. If two XOR gates have identical gate delay and three input signals arrival the FA simultaneously, DPD of three signals would cause glitches at the *sum* signal.

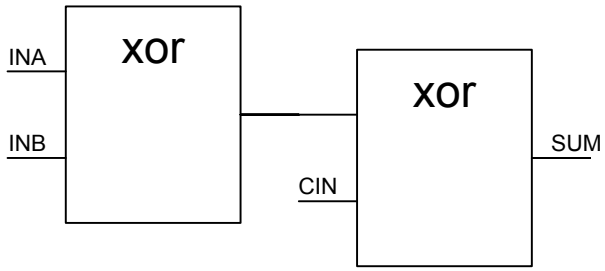

Figure 50. ***Sum=A+B+C.***

Therefore, we need to balance the path delays, both outside and inside FA circuit to decrease the DPD. When the DPD of input signals is less than gate's inertial delay, the glitch could be eliminated as in Equation 56.

For the part of full adder design given in the Figure 50, we could insert a buffer between 'CIN' and XOR gate. The buffer is chosen based on timing lookup tables so that the $|D1\text{-}D2|$ is less than the XOR gate's inertial delay. Meanwhile, $D2$ is less than $D1$, which makes sure that the buffer insertion would not change circuit's critical path, from 'INA' to 'SUM'. In the latter divider example, we also use this way to decrease the DPD of its basic unit, controlled add/subtract (CAS).

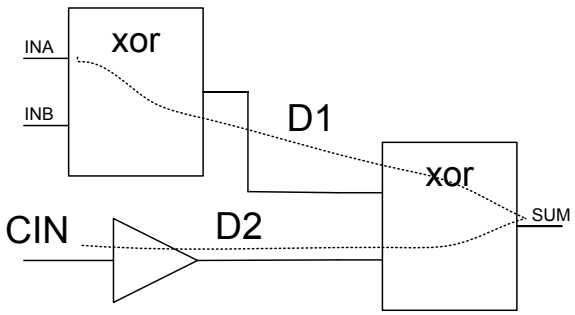

Figure 51. Sum circuit with buffer insertion.

The other way to eliminate the glitches inside the full adder is to choose another design. The XOR gate in the full adder makes DPD of signals higher than its inertial delay that we have to insert a buffer to decrease the DPD. Or we can use a full adder design without XOR gate such as the one in Figure 52. In this design, DPDs of the three input signals to the sum or carry-out are so small that no glitch would be generated by the full adder's circuit.

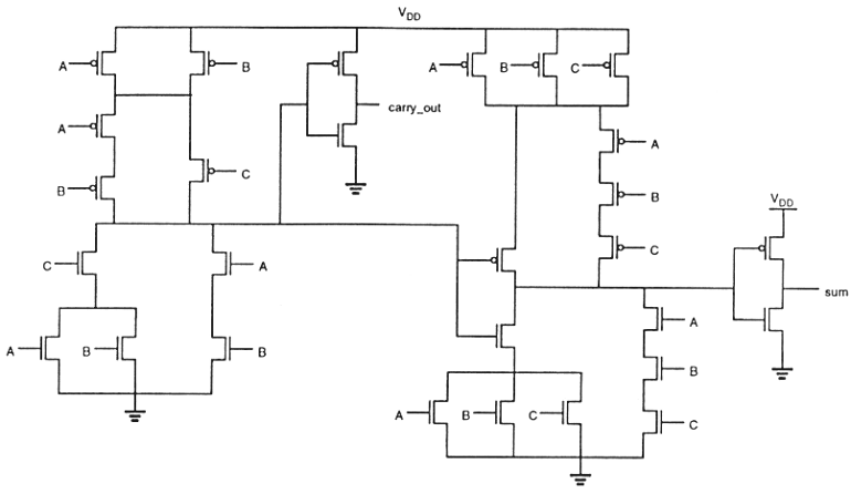

Figure 52. Full adder without XOR gate.

In the following section, we would use real circuit examples to demonstrate how to verify our method.

5.3 Circuit Implementation

5.3.1 Buffer Design and Timing Table

In our implementation, we use serial buffer types to provide a range of delay. The Figure 53 shows the buffer designs we use in our experiment. Then we create timing tables for each buffer type via the HSPICE simulation. Given a node's load capacitance and required delay needed to insert, we could select one buffer or combination of several buffers to get required delay.

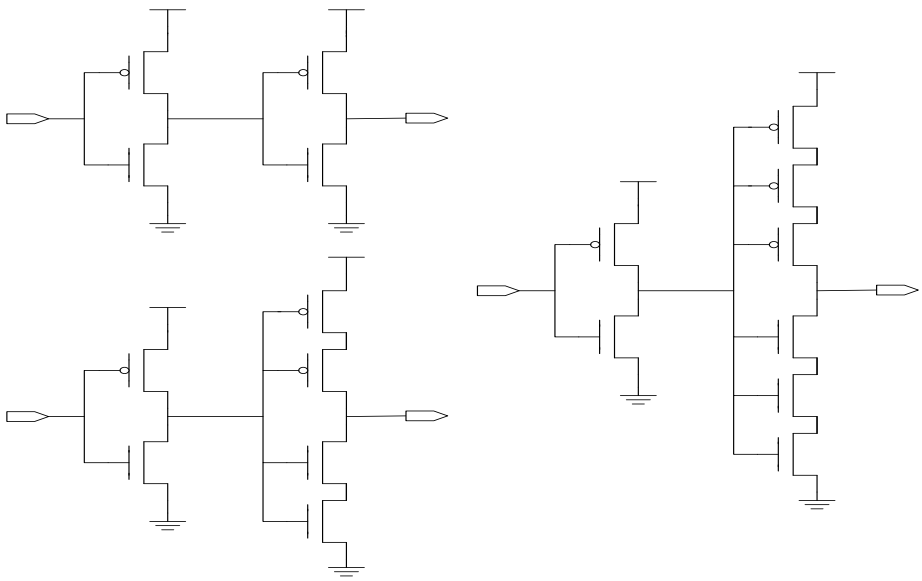

Figure 53. Buffer designs.

For propagation schedule purpose at certain cases (i.e. divider), one buffer with enable signal is provide for signal schedule. We could use system clock signal as the enable signal. Since the primary input data are from registers or latches which need a clock edge to propagate the stored data to broadcast lines, we use this enable signal to schedule the data signals at required time, thus glitches would not be propagated to following gates.

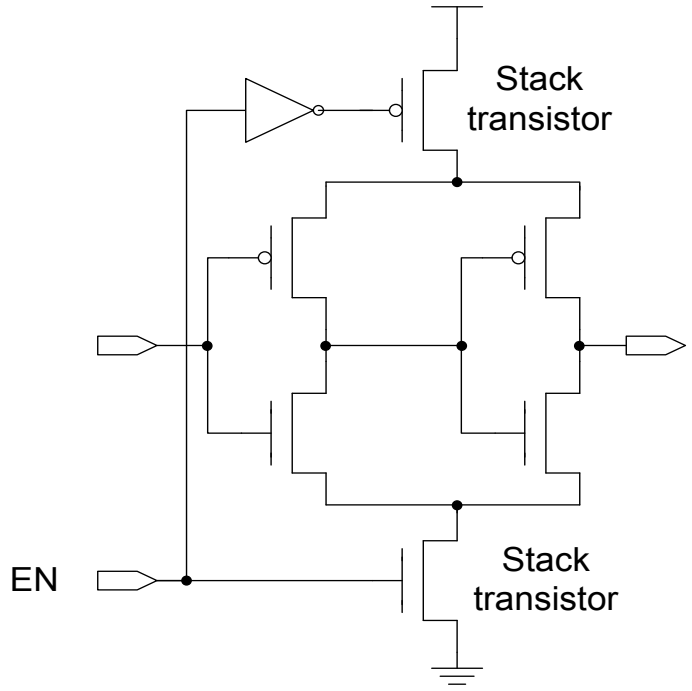

Figure 54. Stack buffer schematic.

In the following, we would demonstrate our buffer-insertion method via two types of calculation circuits: multiplier and divider. The signal arrival time of each node in the

circuit are calculated through STA. Since FA units are identical and have similar load capacitance, we could consider each FA has the same delay in our experiment.

5.3.2 Multiplier Glitch Reduction

The 6-bit robust low-power multiplier schematic is shown in Figure 55. For this circuit, merely primary signals are broadcasted to the circuit. After STA, we can get required arrival time for each broadcast line. Since broadcast lines only appear at primary input signals, case 1 and case 2 in Buffer_Insertion() function would be used.

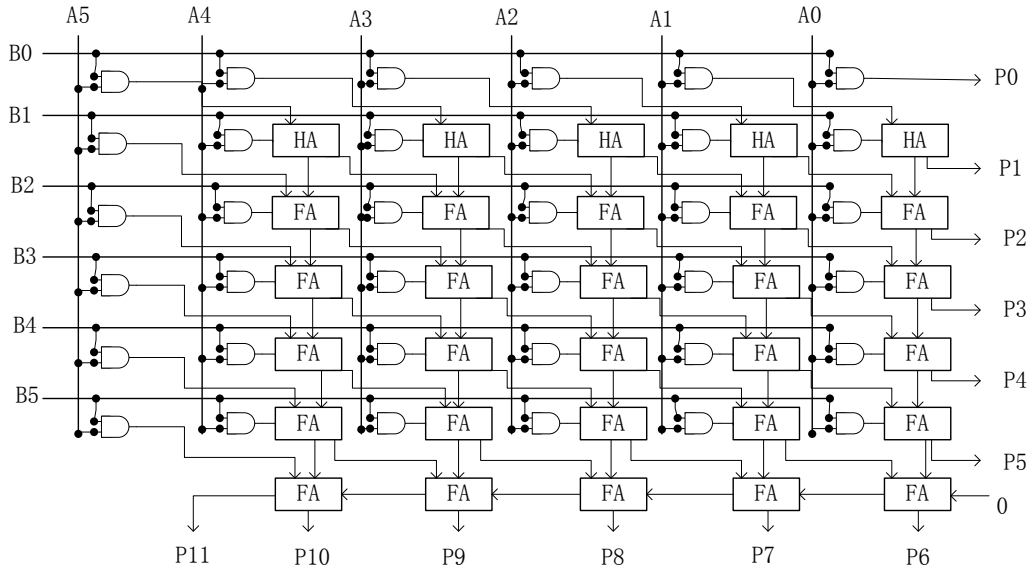

Figure 55. Robust low-power multiplier.

Figure 56 shows the schematic with buffers inserted. By using the method we introduced in last section, buffers are inserted along broadcast lines. Signal *A* s are

broadcasted to the adders of different logic depth, so it's case 2 in the insertion method. As for signal Bs, case 1 is implemented. As for half adders of first row, signal A s arrive at them through one AND gate, which means the path delay to half adders are the same, thus there is no need to insert buffer before adders of first row. As for the second row, the output signals of half adder have delay of one half adder and one AND gate, while the primary signals $B1$ through broadcast line only have delay of one AND gate. The DPD of these two signals would generate glitches at full adders of the second row.

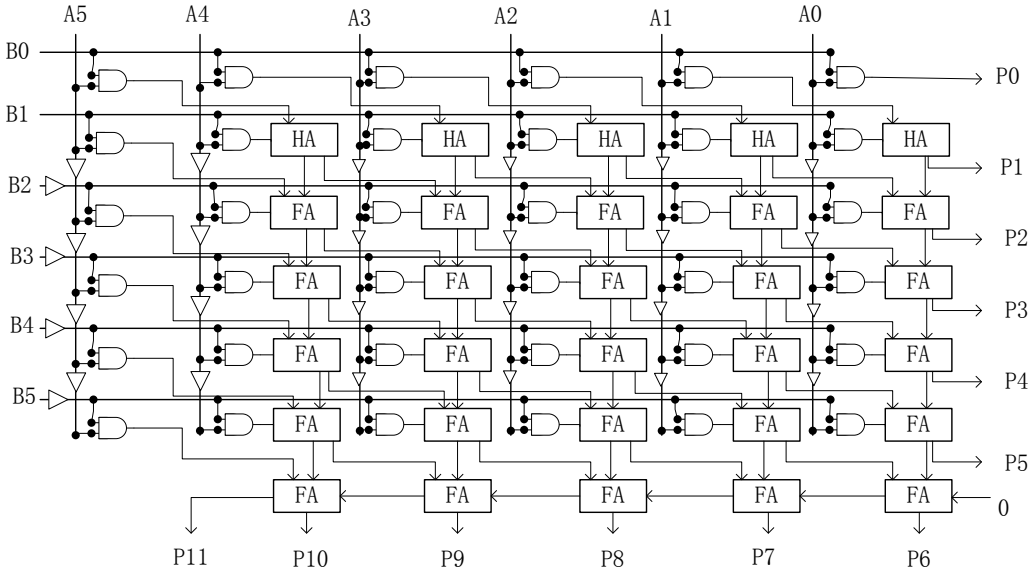

Figure 56. Robust low-power multiplier with buffers.

The following waveform is HSPICE simulation result for FA's *SUM* bit of row 5 and column 5. As shown, the blue line is the waveform before buffer insertion while the red line shows that two glitches are almost eliminated by inserting buffer along the broadcast lines.

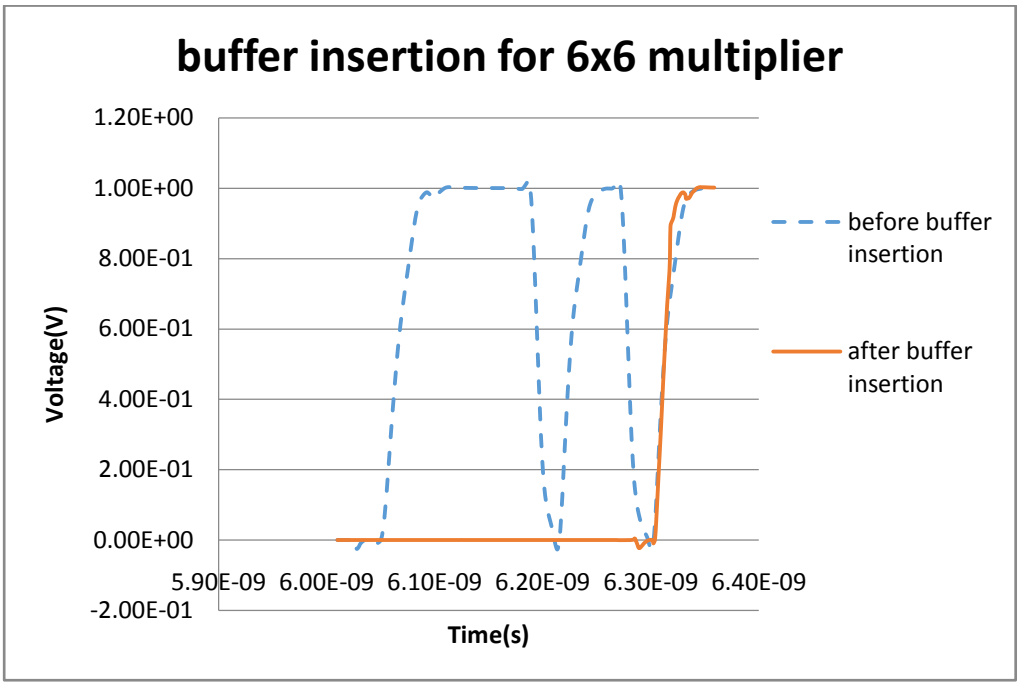

Figure 57. waveform comparison example of case 1 and 2.

5.3.3 Non-restore Divider Glitch Reduction

The non-restore divider circuits shown in Figure 58 is consist of controlled add/subtract unit (CAS). Because of the XOR gate, the inputs signals to the CAS would have different path delay, which means there would be glitches at CAS outputs. Furthermore, 'quotient' signals are broadcasted to following stage to choose the operation (add or subtract). The glitches on the 'quotient' signals would cause lots of glitches in the following circuit. It's the case 3 in the Buffer_Insertion() method. Therefore, we need to stop the glitch propagation along 'quotient' signal lines.

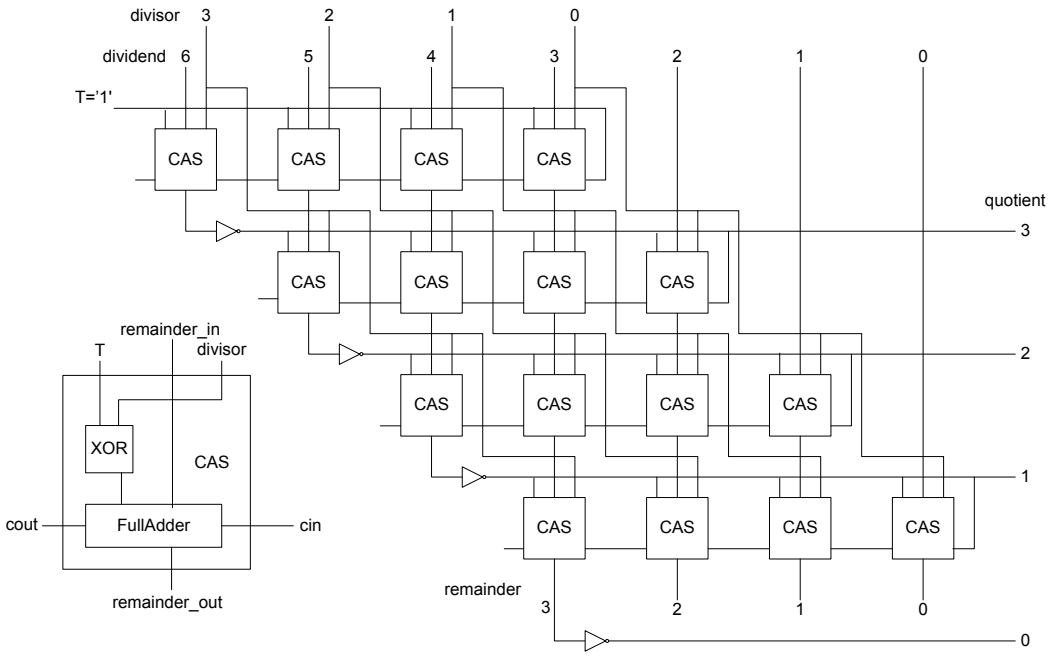

Figure 58. Non-restore divider.

The first thing we need to do is to insert a buffer at the 'remainder-in' line in the CAS to eliminate the glitches generated by the CAS design, as shown in Figure 59. According to the input capacitance of FA in the CAS, we choose a buffer with the identical delay as XOR gate. We don't insert buffers along the *cin* signal, because the signal is along the circuit critical path. Inserting buffer along the critical path would largely lower the circuit timing performance.

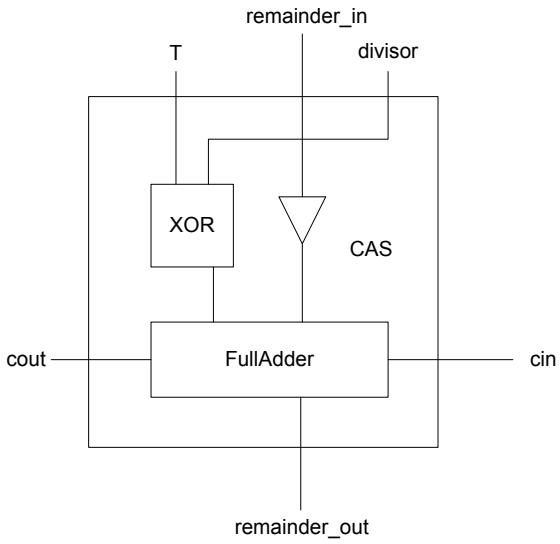

Figure 59. Modified CAS unit.

The second step is to insert buffers along the broadcast lines to stop the glitches propagation as shown in Figure 60. And the Figure 61 shows the scheduled waveform along the broadcasting. The blue line is the broadcast signal of row 5 before buffer insertion while the red line is the waveform after insertion. The dynamic power caused by the red waveform is largely less than the blue one. And the glitches are eliminated by our stack buffer.

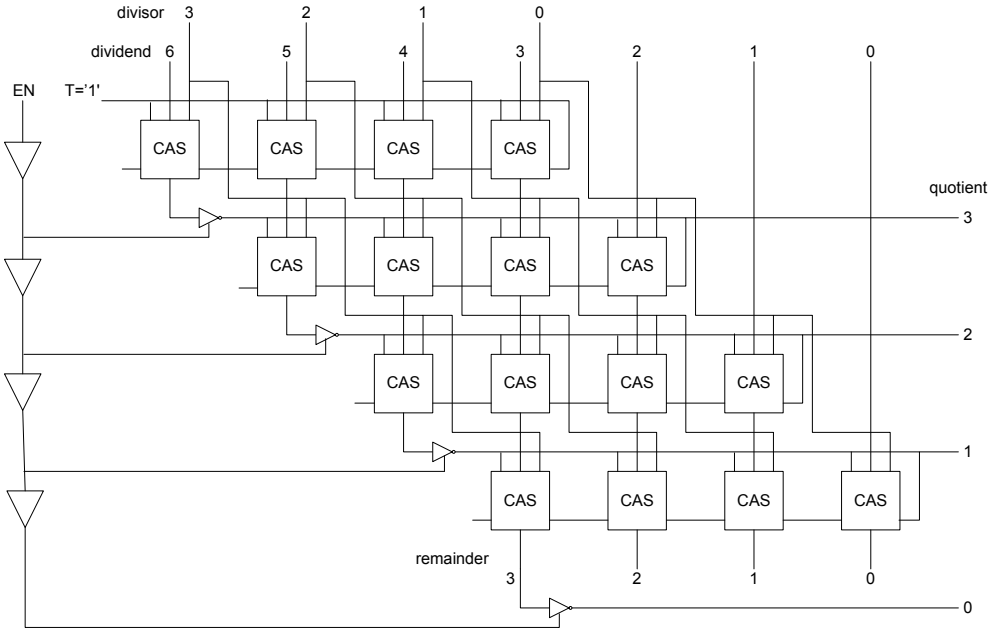

Figure 60. Non-restore divider with stack buffers.

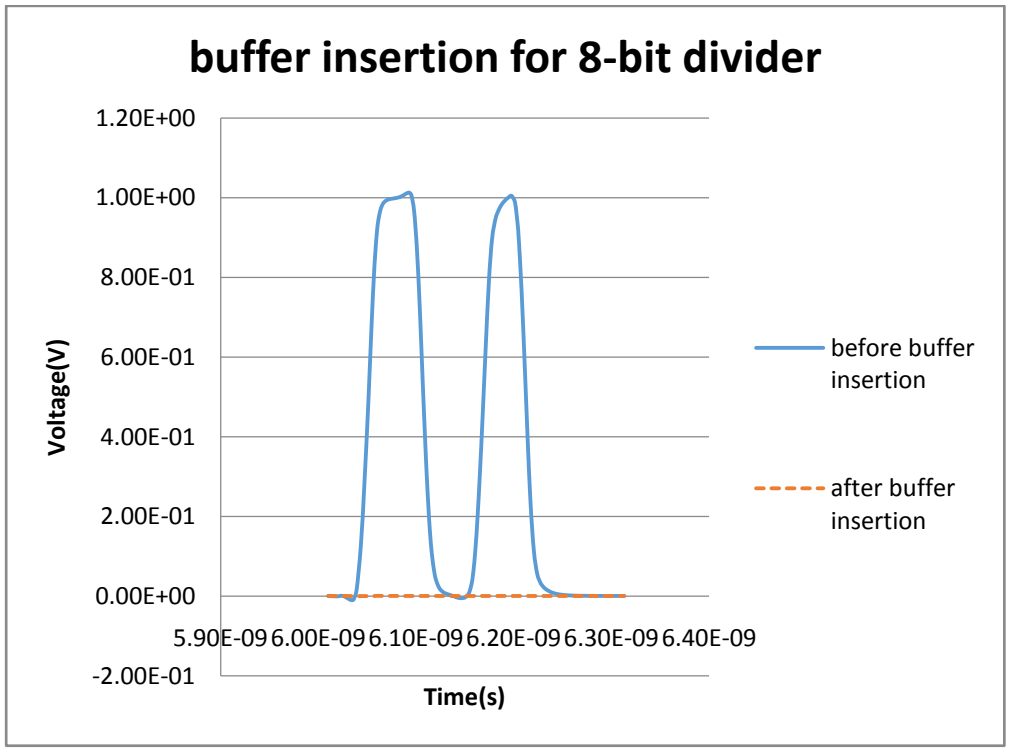

Figure 61. Waveform comparison example of case 3.

5.4 Simulation Results

We evaluate the efficiency of our method using the HSPICE (A2008.03) simulation results over 45nm PTM process technology [20]. Here, 6-bitrobust low-power multiplier (Num.1 in the chart), 32-bit robust low power multiplier (Num.2 in the chart) and non-restore 16/8 divider ((Num.3 in the chart)) are tested. Moreover, all the buffer timing data are derived by HSPICE simulation. In the 32 bit multiplier, we achieve 50% reduction in number of transition, and in the 16/8 divider, we achieve a 72% reduction, in number of transitions. Since as the number of bits increases, the number of glitches can grow dramatically. Our realistic projection of the reduction in number of transitions in 64-bit and 128-bit multiplier is higher than 70% and 90%, respectively. In all cases, the area overhead is about 5%, while the timing performance of all circuits remains the same as the ones before buffer insertion.

5.5 Summary

In this chapter, we proposed a transition estimation method and buffer insertion method to reduce glitches, thus circuit energy consumption. The Simulation results showed that our method reduced spurious transitions via FA-based calculation circuits. We created timing tables of several buffers for accurate delay insertion. The results turned out that as much as 60% energy can be reduced as compared to the original circuits with little extra area.

Chapter 6

Conclusion and Future works

In this dissertation research, gate modeling and gate-level timing analysis for CMOS circuits have been performed. We address the issues of conventional STA that will cause inaccuracy in modern timing analysis: multiple-input transition, complex load (interconnects), and complicated waveform. All these issues were overlooked in conventional STA but cannot be neglect in modern chip design. In order to accurately estimate timing behavior of circuits, we propose a transistor-level gate model to extract MIT timing information from SIT data. A waveform evaluation method is provided to handle the real load issue. Merely based on the timing data with the load of a single capacitor, the gate behavior with any RC circuit load can be simulated, and more accurate waveform can be obtained at gate output, which makes subsequent signal propagation along the interconnections more accurate. Then an equivalent waveform approach is proposed to handle the increasingly complicated waveform. With all these methods, conventional STA can be used in modern timing analysis with high accuracy. At last, due to the low power issue, we propose a buffer insertion method for dynamic power reduction for arithmetic functional units.

6.1 Summary and Conclusion

Single-input transition is widely used in conventional STA tools. Ignoring multiple-input transition could simplify the simulation but bring in timing inaccuracy. As discussed in this dissertation, this overlook can lead to very significant estimation errors. Therefore, a simplified transistor-level gate model is proposed. Using the transistor connection structure, MIT time information can be obtained from SIT data. No additional gate simulation is needed. Extensive experiments are performed over a wide range of logic gates with different process technologies. Our results show that the proposed method has much higher accuracy than the SIT methods. This method requires no changes to the current library format, and thus is compatible with the current STA tools.

The timing LUTs only provide the delay information with a load as a single capacitor. However, the actual load can be a complex RC circuit. Ignoring the resistance of the load can bring in significantly estimation errors. Therefore, we propose a method which uses equivalent admittance to reduce the complex RC circuit into a π-model. Then an "effective capacitor" technology is used to calculate the actual delay and transition for the real circuit. This method also requires no changes to the current library format, while increasing the accuracy of STA.

Different waveforms have different gate and net propagation responses. However, conventional library only uses transition time to capture the characteristic of any waveform, which can result in significant estimation errors. Given the transition time, we propose a waveform evaluation method to obtain the shape of waveform, and use

piecewise linear expressions to model waveforms. Using the obtained waveform, rather than conventional ramp approximation, can greatly increase the accuracy of timing analysis along the interconnection.

Due to the noise and hazards, the real waveform arriving at gate input can be very complicated. A single attribute (transition time) is not enough to describe it. Conventional STA uses ramps to approximate the real waveforms which can cause estimation errors. In order to increase the estimation accuracy, an equivalent waveform approach is proposed in this dissertation. Given any waveform, we use a heuristic method to search the closest waveform in a critical region. Then this equivalent waveform is used for gate timing analysis. Delay and transition time estimation results show that our approach has higher accuracy than the conventional one.

As for the low power design, different arrival times of signals results in spurious transitions, which can waste energy during circuit functioning. We estimate the signal arrival along paths in the circuit and provide a buffer insertion method to reduce the spurious transitions, thus reducing power dissipation. We implement our method over a series of arithmetic function units, and obtain significant energy reduction.

6.2 Future Works

STA is not only a most widely used timing analysis approach in chip design but also the foundation of numerous timing optimization tools. With the improvement of process technology, STA confronts a lot of challenges. At the same time, because of the prevalence of portable devices and limitation of battery technology, power dissipation has

become a crucial concern in VLSI circuit design. According to the challenges we are facing, several suggestions of the future works are described in the following:

- **Model the gates with new process technologies.**

New process technologies come up every year and no one can ensure that STA algorithms based on the old technologies work on the new ones. Conventional algorithms may need to be changed to handle the new technologies. This is a non-stop procedure so long as technology continues to improve.

- **Develop a method that can handle more complicated load**

The simplest load is a pure capacitor. And in this dissertation, we propose a method that can handle a load as complicated as RC trees. However, the real interconnection can be much more complicated, such as with inductance. And the structure of interconnection could be more than a tree structure. Thus all these cases need to be handled for higher estimation accuracy.

- **Use other waveforms rather than a ramp to approximate input waveform**

In our approach, we use equivalent ramp to approximate input waveform. That is because the timing LUT only provides delay information for ramp input. However, other kinds of waveforms may be a better approximation than a ramp. The challenge here is that LUTs provide very limited information (only transition time) for a more complicated waveform.

- Present a method for power consumption estimation of CMOS circuit.

Power estimation for a circuit is crucial in the VLSI design. Since dynamic power consumption can be estimated through circuit activities, there must be a relation between power consumption and circuit timing simulation which is used to estimate the circuit activities. Dynamic timing analysis, such as SPICE, is a most accurate way but not feasible in practice. As for STA, signal waveforms are ignored which may lower the estimation accuracy. Thus, accurate and sufficient power estimation method is required in the VLSI design.

References

[1] Blaauw, David, et al. "Statistical timing analysis: From basic principles to state of the art." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 27.4 (2008): 589-607.

[2] Chen, Liang-Chi, Sandeep K. Gupta, and Melvin A. Breuer. "A new gate delay model for simultaneous switching and its applications." Proceedings of the 38th annual Design Automation Conference. ACM, 2001.

[3] Tsai, Shihheng, and Chung-Yang Huang. "A false-path aware formal static timing analyzer considering simultaneous input transitions." Design Automation Conference, 2009. DAC'09. 46th ACM/IEEE. IEEE, 2009.

[4] Li, Zhentao, and Shuming Chen. "Transistor level timing analysis considering multiple inputs simultaneous switching." Computer-Aided Design and Computer Graphics, 2007 10th IEEE International Conference on. IEEE, 2007.

[5] Ohkubo, Naoaki, and Kimiyoshi Usami. "Delay modeling and static timing analysis for MTCMOS circuits." Proceedings of the 2006 Asia and South Pacific Design Automation Conference. IEEE Press, 2006.

[6] Sridharan, Jayashree, and Tom Chen. "Gate delay modeling with multiple input switching for static (statistical) timing analysis." VLSI Design, 2006. Held jointly with 5th International Conference on Embedded Systems and Design., 19th International Conference on. IEEE, 2006.

[7] Amin, Chirayu, et al. "A multi-port current source model for multiple-input switching effects in CMOS library cells." Proceedings of the 43rd annual Design Automation Conference. ACM, 2006.

[8] Keller, Igor, King Ho Tam, and Vinod Kariat. "Challenges in gate level modeling for delay and SI at 65nm and below." Proceedings of the 45th annual Design Automation Conference. ACM, 2008.

[9] Liu, Bao, and Andrew B. Kahng. "Statistical gate level simulation via voltage controlled current source models." Behavioral Modeling and Simulation Workshop, Proceedings of the 2006 IEEE International. IEEE, 2006.

[10] Liu, Bao. "Gate level statistical simulation based on parameterized models for process and signal variations." Quality Electronic Design, 2007. ISQED'07. 8th International Symposium on. IEEE, 2007.

[11] Amelifard, Behnam, et al. "A current source model for CMOS logic cells considering multiple input switching and stack effect." Proceedings of the conference on Design, automation and test in Europe. ACM, 2008.

[12] Goel, Amit, and Sarma Vrudhula. "Current source based standard cell model for accurate signal integrity and timing analysis." Proceedings of the conference on Design, automation and test in Europe. ACM, 2008.

[13] Goel, Amit, and Sarma Vrudhula. "Statistical waveform and current source based standard cell models for accurate timing analysis." Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE. IEEE, 2008.

[14] Devgan, Anirudh. "Accurate device modeling techniques for efficient timing simulation of integrated circuits." Computer Design: VLSI in Computers and

Processors, 1995. ICCD'95. Proceedings., 1995 IEEE International Conference on. IEEE, 1995.

[15] Raja, S., et al. "Transistor level gate modeling for accurate and fast timing, noise, and power analysis." Proceedings of the 45th annual Design Automation Conference. ACM, 2008.

[16] Tang, Qin, et al. "Transistor-level gate model based statistical timing analysis considering correlations." Proceedings of the Conference on Design, Automation and Test in Europe. EDA Consortium, 2012.

[17] Fatemi, Hanif, Shahin Nazarian, and Massoud Pedram. "Statistical logic cell delay analysis using a current-based model." Proceedings of the 43rd annual Design Automation Conference. ACM, 2006.

[18] Tang, Qin, et al. "Statistical delay calculation with multiple input simultaneous switching." IC Design & Technology (ICICDT), 2011 IEEE International Conference on. IEEE, 2011.

[19] Tang, Qin, et al. "Statistical Transistor-Level Timing Analysis Using a Direct Random Differential Equation Solver." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 33.2 (2014): 210-223.

[20] Arizona State University, “Predictive Technology Model (PTM)”, http://ptm.asu.edu/

[21] Kahng, Andrew B., and Sudhakar Muddu. "Efficient gate delay modeling for large interconnect loads." Multi-Chip Module Conference, 1996. MCMC-96, Proceedings., 1996 IEEE. IEEE, 1996.

[22] O'Brien, Peter R., and Thomas L. Savarino. "Modeling the driving-point characteristic of resistive interconnect for accurate delay estimation." The Best of ICCAD. Springer US, 2003. 393-402.

[23] Gopal, Nanda, Dean P. Neikirk, and Lawrence T. Pillage. "Evaluating RC-interconnect using moment-matching approximations." Computer-Aided Design, 1991. ICCAD-91. Digest of Technical Papers., 1991 IEEE International Conference on. IEEE, 1991.

[24] Devgan, Anirudh, and Peter R. O'Brien. "Realizable reduction for RC interconnect circuits." Computer-Aided Design, 1999. Digest of Technical Papers. 1999 IEEE/ACM International Conference on. IEEE, 1999.

[25] Xu, Jingye, and Masud H. Chowdhury. "Fast waveform estimation (FWE) for timing analysis." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 19.5 (2011): 846-856.

[26] Qian, Jessica, Satyamurthy Pullela, and Lawrence Pillage. "Modeling the "effective capacitance" for the RC interconnect of CMOS gates." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 13.12 (1994): 1526-1535.

[27] Huang, Zhang-cai, Atsushi Kurokawa, and Yasuaki Inoue. "Effective capacitance for gate delay with RC loads." Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on. IEEE, 2005.

[28] Fang, Shuai, et al. "Calculating the effective capacitance for interconnect loads based on Thevenin model." Communications, Circuits and Systems Proceedings, 2006 International Conference on. Vol. 4. IEEE, 2006.

[29] Jiang, Minglu, et al. "A non-iterative effective capacitance model for CMOS gate delay computing." Communications, Circuits and Systems (ICCCAS), 2010 International Conference on. IEEE, 2010.

[30] Kahng, Andrew B., Kei Masuko, and Sudhakar Muddu. "Analytical delay models for VLSI interconnects under ramp input." Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design. IEEE Computer Society, 1997.

[31] Ling, David D., et al. "A moment-based effective characterization waveform for static timing analysis." Proceedings of the 46th Annual Design Automation Conference. ACM, 2009.

[32] Hashimoto, Masanori, Yuji Yamada, and Hidetoshi Onodera. "Equivalent waveform propagation for static timing analysis." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 23.4 (2004): 498-508.

[33] Keller, Igor, King Ho Tam, and Vinod Kariat. "Challenges in gate level modeling for delay and SI at 65nm and below." Proceedings of the 45th annual Design Automation Conference. ACM, 2008.

[34] Raja, S., et al. "Transistor level gate modeling for accurate and fast timing, noise, and power analysis." Proceedings of the 45th annual Design Automation Conference. ACM, 2008.

[35] Menezes, Noel, Chandramouli Kashyap, and Chirayu Amin. "A “true” electrical cell model for timing, noise, and power grid verification." Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE. IEEE, 2008.

[36] Kawano, Takao, et al. "Adjacent-State monitoring based fine-grained power-gating scheme for a low-power asynchronous pipelined system." Circuits and Systems (ISCAS), 2011 IEEE International Symposium on. IEEE, 2011.

[37] Morgenshtein, Arkadiy. "Short-Circuit Power Reduction by Using High-Threshold Transistors." Journal of Low Power Electronics and Applications 2.1 (2012): 69-78.

[38] Sreenivaas, V. L., et al. "A novel dynamic voltage scaling technique for low-power FPGA Systems." Signal Processing and Communications (SPCOM), 2010 International Conference on. IEEE, 2010.

[39] Lin, Tong, et al. "Fine-grained power gating for leakage and short-circuit power reduction by using asynchronous-logic." Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on. IEEE, 2009.

[40] Enomoto, Tadayoshi, and Nobuaki Kobayashi. "A low power multimedia processor implementing dynamic voltage and frequency scaling technique." Design Automation Conference (ASP-DAC), 2013 18th Asia and South Pacific. IEEE, 2013.

[41] Zhang, Xiaoxiao, Amine Bermak, and Farid Boussaid. "Dynamic voltage and frequency scaling for low-power multi-precision reconfigurable multiplier." Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on. IEEE, 2010.

[42] Kuo, Ko-Chi, and Chi-Wen Chou. "Low power and high speed multiplier design with row bypassing and parallel architecture." Microelectronics Journal 41.10 (2010): 639-650.

[43] Yan, Jin-Tai, and Zhi-Wei Chen. "Low-power multiplier design with row and column bypassing." SOC Conference, 2009. SOCC 2009. IEEE International. IEEE, 2009.

[44] Prabhu, A. S., and V. Elakya. "Design of modified low power booth multiplier." Computing, Communication and Applications (ICCCA), 2012 International Conference on. IEEE, 2012.

[45] Agrawal, Vishwani D. "Low-power design by hazard filtering." VLSI Design, 1997. Proceedings., Tenth International Conference on. IEEE, 1997.

[46] Sobelman, Gerald E., and Donovan L. Raatz. "Low-power multiplier design using delayed evaluation." Circuits and Systems, 1995. ISCAS'95., 1995 IEEE International Symposium on. Vol. 3. IEEE, 1995.

[47] Kumar, V. B. V. P., et al. "A technique to eliminate glitch power consumption at physical design stage in CMOS circuits." Information and Communication Technologies (WICT), 2011 World Congress on. IEEE, 2011.

[48] Uppalapati, Siri, Michael L. Bushnell, and Vishwani D. Agrawal. "Glitch-free design of low power ASICs using customizedresistive feedthrough cells." Proc. VLSI Design And Test Symp. 2005.

[49] Benini, Luca, et al. "Glitch power minimization by selective gate freezing." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 8.3 (2000): 287-298.

[50] Raja, Tezaswi, Vishwani D. Agrawal, and Michael L. Bushnell. "Minimum dynamic power CMOS circuit design by a reduced constraint set linear program." VLSI Design, 2003. Proceedings. 16th International Conference on. IEEE, 2003.

[51] Lee, Hyungwoo, Hakgun Shin, and Juho Kim. "Glitch elimination by gate freezing, gate sizing and buffer insertion for low power optimization circuit." Industrial Electronics Society, 2004. IECON 2004. 30th Annual Conference of IEEE. Vol. 3. IEEE, 2004.

[52] Wang, Lei, et al. "A gate sizing method for glitch power reduction." SOC Conference (SOCC), 2011 IEEE International. IEEE, 2011.

VITA

NAME OF AUTHOR: Chaobo Li

PLACE OF BIRTH: Zhejiang, China

DATE OF BIRTH: Dec.05, 1985

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

Zhejiang University, Hangzhou, Zhejiang, China

DEGREES AWARDED:

Bachelor of Engineering

PROFESSIONAL EXPERIENCE:

Teaching Assistant, Department of Electrical Engineering, Syracuse University