Delay Models for the Sea-of-Wires Array Synthesis System

Ing-Yi Chen
Dept. of Electronic Engineering
Chung Yuan Christian University
Chungli, Taiwan, R.O.C.

Geng-Lin Chen
Digital Signal Processing Dept.
Computer & Communication Lab.
ITRI, Hsinchu, Taiwan, R.O.C.

Sy-Yen Kuo
Dept. of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R.O.C.

Abstract

This paper presents two simple, accurate and efficient delay models, the static delay model and the dynamic delay model, to support performance optimization of VLSI Sea-of-Wires Arrays (SWA). The SWA delay model treats each distributed gate as an attribute-based primitive gate with different internal and external connection wires. Instead of solving differential equations, the SWA model determines delays by lookup from a multi-dimensional table. Only a few microseconds of execution time are needed per gate. The propagation delay along a circuit path is the sum of the delay segments of distributed gates in the path. The critical path of an SWA design can be identified with an $O(n)$ timing analysis algorithm. For most AHPL Benchmarks, the table-lookup method achieves 5 orders of magnitude speedup over SPICE for the same circuits with error margin less than 7%.

1 Introduction

For the delay estimation of the critical path, one traditional approach is by the circuit simulator, e.g. SPICE. A circuit is modeled by a collection of differential equations, and equations are solved to predict the behavior of the circuit. The advantage of this approach is accuracy: the ability to simulate exactly the real-world behavior of devices. Unfortunately, the accuracy of the circuit models comes at a high price in execution time. Switch-level simulators typically require several seconds of CPU time per transistor. As a result, the programs are impractical for the state-of-the-art VLSI circuits, which can take weeks of execution time.

A simple and accurate delay model is indispensable to an effective timing analysis of digital circuits. Numerous efforts have been reported on gate delay models, such as linear RC models [1], analytic delay formula methods [2], the table lookup method [3]. The simulators based on the linear RC models can improve the computation speed. However, for those nonlinear MOS transistors, they can lose reliable accuracy. For analytic delay formula methods [2, 4, 5], most of them are concerned with the drain current as a function of applied terminal voltages, gate capacitances (Cgs,Cgb,Cgd), substrate capacitances (Cbd,Cbs), and Miller capacitances effect (Cgd). These considerations are very complex and it is difficult to derive the accurate delay formula for all basic circuit structures. Some models assume the input is a step function to simplify the delay calculation. However, this limits the reality of various input waveforms.

In this paper, a tabular approach is proposed to accommodate the extreme and nonlinear variation of delay with fan-in and fan-out. A gate-attribute is defined to characterize the impact of input waveform on the delay of a gate. SPICE simulation is used to compile delay and transition descriptor tables. Instead of solving differential equations, these tables are then used to replace SPICE simulation and determine delays by lookup from a multi-dimensional table. The rest of paper is organized as following. Section 2 will present an overview of the SWA structure to serve as a general background to the distributed gate design style. Two delay models detailed in Section 3 and 4 are the core of the table lookup approach. The effectiveness of those models is assessed by evaluations of AHPL Benchmarks with respect to resource utilization. Section 5 compares the experimental results of SWA realizations with those from SPICE. Finally, Section 6 draws conclusions.

2 Sea-of-Wires Arrays

SWA is a structured arrays design method based on the distributed gates approach. The SWA layout is a form of layout matrix resembling a Programmable

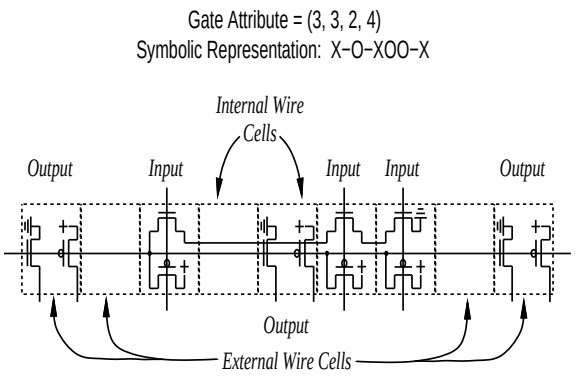

Figure 1: An Example Distributed Gate with an Attribute of (3,3,2,4).

Logic Array (PLA) structure. Instead of thinking of an integrated circuit as initially covered with a *sea-of-cells* that can be interconnected with wires by designer or router, the *SWA* method views the IC as a set of *horizontal and vertical wires* into which the designer can place building blocks called *SWA cells*. This predefined grid of vertical and horizontal wires is, therefore, called a *Sea-of-Wires* [6]. The number of wires (grids) in each row and column varies and depends on technology and type of circuitry used.

Primitive gates and *wires* are considered as the basic constructs of a distributed gate in the delay model. A distributed gate *attribute* is characterized by a description containing four slots (n, m, L_i, L_e), where n is the number of gate inputs, m is the number of gate outputs, L_i is the number of internal wire cells in the gate, and L_e is the number of external wire cells out of the gate. As an example, the distributed gate illustrated in Figure 1 has an attribute of (3,3,2,4). The delay properties of the basic components are measured by running SPICE and distilling the results down to a multi-dimensional table. When analyzing a circuit, distributed gates are first broken down and mapped into the basic components according to the gate attribute and then the delay values for these components are looked up from the table. Neglecting the effect of input rise time, the proposed delay estimate of a gate is computed quickly by the empirical linear approximation given in Equation 1.

$$Delay(n, m, L_i, L_e) = Gate(n, m, 0, 0) + Wire(n, m, L_i, 0) + Wire(n, m, 0, L_e) \quad (1)$$

The following sections will justify this expression and develop the mechanism for computing and retrieving each term. We shall conclude that this simple equation is justified because delay is very nearly linearly related to capacitance values.

3 The Static Delay Model

The first term of Equation 1 is primitive gate delay $Gate(n, m, 0, 0)$ representing the intrinsic delay of the gate with n input cells and the loading delay resulting from capacitances on the m output cells. The input excitation is made active only on the input having the longest path length to the farthest output. Both of rising and falling delays are measured for NAND gates with input and output numbers varying from 1 to 9 respectively. All SPICE simulations and delay tabulations are based on the processing conditions of 2μ technology. The tabulation would be redone as more refined technologies are adopted.

3.1 Primitive Delay Modeler

The delay of a single isolated primitive gate with n inputs and m loads is represented in the table as $SPICE(n, m)$. However, the delay of a gate within a network is more complex to estimate than the delay of a single isolated gate. Equation 2 provides an approximate delay value for a primitive gate within a path.

$$Gate(n, m, 0, 0) = SPICE(n, 0) + [(SPICE(n, m) - SPICE(n, 0)] * (1 - \tfrac{1}{m}) \quad (2)$$

$SPICE(n, 0)$ is the approximate intrinsic delay assignable to the n inputs without output load as determined by SPICE simulation. Similarly, $SPICE(n, m)$ is the SPICE measurement of the n input gate but with m output loads. The difference between those two terms is the delay assignable to the m output loads. In a gate network, the loading capacitance seen from the stage ν of a path of interest will become the driving capacitance when the delay computation focuses on the gate of stage $\nu + 1$. The delay associated with a single capacitance will be counted twice as a load in one stage and as a drive in the next stage if a gate delay is simply equal to $SPICE(n, m)$. The 1/m loading delay of each gate along the path is therefore removed from the gate delay estimation of Equation 2.

3.2 Wire Delay Modeler

The second term of Equation 1 is the internal wire delay $Wire(n, m, L_i, 0)$ parameterized with the wire length L_i measured for a gate with n input and m

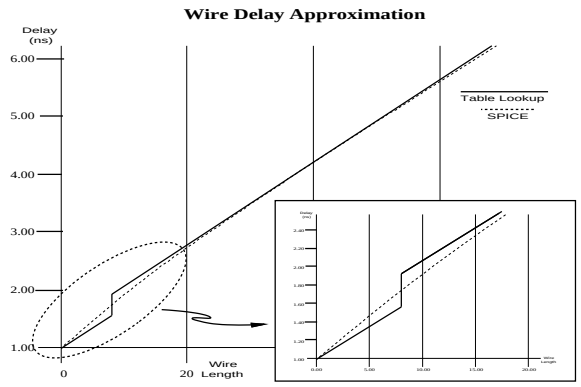

Figure 2: An Example of Internal Wire Delay Approximation.

output cells. Unit wire cell delay of SPICE simulation increases linearly for most gate configurations. But some gates with lower input number n have non-linear curves in the short wire length region. An empirical linear Equation 3 was obtained from extensive experiments to approximate the internal wire delays.

$$Wire(n, m, L_i, 0) = \begin{cases} S_I * L_i & \text{if } L_i < N_{n,m} \\ K_I + S_I * L_i & \text{otherwise} \end{cases} \quad (3)$$

The solid line in Figure 2 graphically represents Equation 3 applied to an instance of a gate with 2 inputs and 4 outputs to approximate the dash line measurements of SPICE. S_I of Equation 3 is the average delay increase assignable to long internal wires. K_I is determined by the maximum difference of SPICE measurements and values of $S_I * L_i$. The intention is to make the table lookup values $K_I + S_I * L_i$ a close upper bound of the corresponding SPICE delays. Experimental results demonstrate that $K_I + S_I * L_i$ is a very good approximation for long wires. However, it yields excessively high values for short lines of gates with low input number n. A critical wire length $N_{n,m}$ is found for every SPICE curve of gate (n, m), where delay at this point is equal to $\frac{1}{2} * K_I + S_I * N_{n,m}$. For those wires shorter than $N_{n,m}$, delays are computed by the term $S_I * L_i$ only. By not adding the constant term to some gates with very short wires, Equation 3 balances delay estimations over the whole region. Equation 3 is particular necessary in a critical path of short lines than the wire length $N_{n,m}$. With similar analysis, the third term of Equation 1, the external wire delay, can be determined by Equation 4.

$$Wire(n, m, 0, L_e) = \begin{cases} S_E * L_e & \text{if } L_e < M_{n,m} \\ K_E + S_E * L_e & \text{otherwise} \end{cases} \quad (4)$$

4 The Dynamic Delay Model

The propagation delay of each path is assumed to be the sum of gate delays calculated by Equation 1. However, the waveform through a chain of gates may have different transition times from the default value (5 ns) in the lookup table. In practice, the effective resistance of a gate in a path depends on its input transition waveform. Experimental results shows a slow output transition waveform or a long delay of a predecessor gate makes its next stage gate have a longer delay than the table lookup estimation by Equation 1. This difference,

$$Diff = SPICE(n, m, L_i, L_e) - Table(n, m, L_i, L_e), \quad (5)$$

is almost linearly proportional to the input transition time. Furthermore, the difference value of Equation 5 also depends on the configuration of the gate. For a given same input transition, the difference increases with the load on the driving transistor of the gate. The experimental data shows that the difference does not increase linearly with respect to the gate delay. The slope ℓ of the difference decreases as the gate configuration becomes complex. Further data analysis shows that the difference is actually linearly sensitive to the cube root of the gate delay.

The difference between the actual delay and the table lookup value due to different transition times from the default in the table must be corrected into the timing model. Equation 6 is proposed as an expression for the difference given in Equation 5. The constant K_t and the values of exponents power a and b are to be determined.

$$\begin{aligned} Diff &= K_t * f(a, b) \\ &= K_t * (Input\ Transition)^a * (Gate\ Delay)^b \end{aligned} \quad (6)$$

Extensive experiments were conducted to determine the value of the constant K_t for different powers a and b of Equation 6. The ratio of the difference to the parameters are calculated in each set of experiments. An individually accurate value of K_t was computed for each pair (a, b) for each input delay/gate delay data point. Based on the hypothesis that a constant K_t could be found, the average K_t for all data points was computed. The values of a and b were selected for which the average deviation from this proposed mean K_t as computed from Equation 7 was minimal.

$$Average\ Deviation = \sqrt{\frac{\sum (\frac{Diff}{f(a,b)} - \frac{\sum \frac{Diff}{f(a,b)}}{n})^2}{n}}$$

$$= \sqrt{\frac{\sum (K_i - K_M)^2}{n}} \qquad (7)$$

Input transition time varies in a 20 ns interval from 5 ns to 165 ns for the high to low input excitation and in a 5 ns interval from 5 ns to 45 ns for the high to low excitation for each set of 17 different gate configurations. K_t is determined by taking the average ratio of all the experiments except the inverters with an attribute of (1,1,0,1). The set of experiments with $a = 0.77$ and $b = 0.25$, and with input excitation high to low yielded an average K_t of 0.839 with an average deviation of 7%. In another set of experiments with $a = 0.66$ and $b = 0.33$, and with low to high input transition yielded a different constant K_t of 0.206 with 16% average deviation. In summary, gate delays along a path can be computed from Equation 8 for various input delays, where $Delay(n, m, L_i, L_e)$ is the uncorrected gate delay estimated from Equation 1.

$$Delay = \begin{cases} Delay(n, m, L_i, L_e) \\ \qquad \text{if } Input\ Delay < 5 \text{ ns} \\ Delay(n, m, L_i, L_e) + \\ \quad K_t * (Input\ Delay)^a * (Gate\ Delay)^b \\ \qquad \text{otherwise} \\ K_t = 0.839, a = 0.77, b = 0.25 \\ \qquad \text{for input } 1 \rightarrow 0 \\ K_t = 0.206, a = 0.66, b = 0.33 \\ \qquad \text{for input } 0 \rightarrow 1 \end{cases} \qquad (8)$$

5 Evaluation

A pilot version of the timing analysis phase of the SWA synthesis system has been implemented and successfully applied to the AHPL Benchmarks over a range of network size and complexity. The input of the timing analyzer is the *routing-completed SLF representation* [6]. The critical path of each design is identified with path segments containing gate configurations and their corresponding delays calculated by the table lookup method. Those critical paths are then simulated with SPICE for the purpose of comparison. The intention is to use the SPICE measurements to evaluate the accuracy and efficiency of the table lookup timing analysis approach to assure the benefit of the input transition correction term. The absolute performance of the designs depends on details of the SWA cell design and the actual processing technology. At this writing, a set of tables have been constructed accurately representing one 2.0 μ CMOS process.

For purposes of this paper, we focus on tracking the overall error from gate to gate through the worst case path of typical example realizations. We have synthesized in SWA format 6 separate digital systems and analyzed the critical path of each one. The critical path complexity varies from 7 segments of a simple shift register circuit HWTEST to 33 segments of a 4-bit processor PROCER. Table 1 shows the results of the static model delay calculations *without* adding the correction term. The first column indicates the number of gates in series in the critical path for each system. Column 2 is the path delay resulting from table calculation. Column 3 contains the path delay determined by SPICE simulation of the entire path. Column 4 is the calculated path delay over the SPICE path delay. The last column is the total percent error in path delay given by

Circuit	Gates in Path	Table (ns)	SPICE (ns)	Path *Error*
SHIFTR	7	5.98	5.62	6.40 %
HWTEST	10	11.11	11.08	0.27 %
SEQSEL	10	10.36	10.66	2.81 %
SERCOM	15	30.02	30.98	3.10 %
STACK4	8	20.24	21.18	4.44 %
PROCER	33	265.50	333.83	20.47 %

Table 1: Performance of the AHPL Benchmarks without the Correction Term.

$$PathError = \frac{|Table - SPICE|}{SPICE} \times 100\% \qquad (9)$$

Most benchmarks have highly accurate delay estimation as compared to the corresponding SPICE measurement except for the processor. The discrepancy of the processor results between table lookup value and SPICE measurement is the result of many *complex* gates along the critical path which cause inaccurate estimates of successor gate delays.

The input transition waveform correction term of the dynamic delay model given by Equation 8 makes an improvement in accuracy. Multiple paths in a layout parameter lookup could be accomplished at the outset and Equation 8 stored, as functions of only input gate attributes, for every gate in the network. A path delay calculation proceeds from input to output with gate attributes for predecessor gates pass to successor gates as input transition times. Gates will be proposed in a search for the critical path. Parameters of the gate attribute will be looked up, equations formed, used, and then likely discarded as the search process proceeds. Transition times must still be saved

Benchmark	Table (ns)	SPICE (ns)	Ratio Tab/SPICE	*Path Error*
SHIFTR	5.98	5.62	106.40 %	6.40 %
HWTEST	11.11	11.08	100.27 %	0.27 %
SEQSEL	10.36	10.66	97.19 %	2.81 %
SERCOM	30.02	30.98	96.90 %	3.10 %
STACK4	21.59	21.18	101.94 %	1.94 %
PROCER	345.59	333.83	103.52 %	3.52 %

Table 2: Performance Evaluation with the Correction Term.

Benchmark	Table (sec.)	SPICE (sec.)	*Efficiency* SPICE/Table
SHIFTR	0.00038	60	2.5*10e4
HWTEST	0.00027	135	5.0*10e5
SEQSEL	0.00058	180	3.1*10e5
SERCOM	0.00042	385	9.2*10e5
STACK4	0.00042	235	5.6*10e5
PROCER	0.00049	11640	2.9*10e7

Table 3: Execution Time Comparisons.

to successor gates in the search. Clearly the process is constrained to proceed from input to output.

Table 2 summarizes the results of the delay calculations *with* the correction term for all 6 layouts. The cumulative path delay error of 3.52% in the PROCER case is typical because individual gate delay errors may be expected to both positive and negative and at least partially cancel. The dynamic model based delay errors are all within 7% of the SPICE's results, while requiring a 5 orders magnitude less execution time on a SPARC station 10 than SPICE as shown in Table 3. The errors of 5% or 6% which may appear in the unlikely absence of cancellation are tolerable for path delay calculation. The decrease in execution time by a factor of 10^5 should make this approach to delay calculation an attractive alternative to SPICE in all analysis except perhaps a final check before a layout is reduced to silicon.

6 Conclusions

Simple, *fast* and *accurate* are the merits of the timing analysis phase of the SWA synthesis system. Two features contribute to the success of the timing analyzer. The first feature, the *table lookup approach* based on the static delay modeler, enables the system to handle a variety of different circuit constructs in a uniform fashion. Over a range of AHPL Benchmark complexity, we conclude that the table lookup is faster by a factor of 10^5 for small circuits. For larger designs, the advantage approaches 10^6. The second feature is the simple and adequately accurate *correction term*, which tunes table lookup results in atypical path configurations without any additional table. The discrepancy of the results between table lookup value and SPICE measurement can be further reduced to 6% when the dynamic delay modeler is adopted to include the effect of the various input waveforms. Those issues makes the SWA synthesis system a complete and efficient design automation system. The ultimate goal of extended SWA synthesis research and development efforts is to achieve an SWA chip layout that compares favorably with manual design of the same architecture.

Acknowledgement

This research was partially funded by the National Science Council, Taiwan, R.O.C., under Grant NSC-83-0404-E-033-005 and NSC-84-0115-C033-01-054E.

References

[1] An-Chang Deng and Yan-Chyuan Shiau, "Generic Linear RC Delay Modeling for Digital CMOS Circuits," *IEEE Trans. on CAD.*, vol. 9, No. 4, pp.367-372, Apr. 1990.

[2] D. Deschacht, M. Robert, and Auvergne, "Explicit Formulation of Delays in CMOS Data Paths," *IEEE Journal of Solid-State Circuits*, vol. 23, No. 5, pp.1257-1264, Oct. 1988.

[3] An-Chang Deng, "Piecewise-Linear Timing Delay Modeling for Digital CMOS Circuits," *IEEE Trans. on CAS.*, vol. 35, No. 10, pp.1330-1334, Oct. 1988.

[4] Matthias Passlack, Manfred Uhle, and Horst Elschner, "Analysis of Propagation Delays in High-Speed VLSI Circuit Using a Distributed Line Model," *IEEE Trans. on CAD.*, vol. 9, No. 8, pp.821-826, Aug. 1990.

[5] Takayasu Sakurai, A. Richard Newton, "Alpha-Power Law MOSFET Model and its Application to CMOS Inverter Delay and Other Formulas," *IEEE Journal of Solid-State Circuits*, vol. 25, No. 2, pp.584-593, Apr. 1990.

[6] I. Chen, G. Chen, F. J. Hill and S. Kuo, "The Sea-of-Wires Array Synthesis System," *30th Design Automation Conference*, pp. 188-193, June 1993.