A Transistor Reordering Algorithm for the Performance Optimization of CMOS Digital Circuits

Bradley S. Carlson[†] and *C.Y. Roger Chen*[‡]

[†]Department of Electrical Engineering
State University of New York at Stony Brook
Stony Brook, NY 11794-2350
Phone: (516) 632-8474
Fax: (516) 632-8494

[‡]Department of Electrical and Computer Engineering
Syracuse University
Syracuse, NY 13244-1240
Phone: (315) 443-4301

Technical Report No. 617
January 31, 1992

Abstract— A model which estimates the relative difference between the best and worst propagation delays of a CMOS complex gate with respect to the order of its transistors, and an algorithm which performs transistor reordering based on this model to significantly reduce propagation delays are presented. Since the algorithm presented in this paper uses a general circuit model based on the results of SPICE simulations, it can be used for circuits of arbitrary functionality and size. Furthermore, it can be used in any semi-custom design environment (e.g., gate array, standard cell), since it is not dependent on a cell library. Although the model is process dependent, it can be parameterized for a new fabrication process automatically. Experimental results for the circuits tested thus far show that the improvement in propagation delay can be as much as 22 percent.

I INTRODUCTION

Great improvements have been made in the Computer-Aided Design (CAD) of VLSI circuits; however, very few circuit designers are able to achieve the circuit performance they desire without manually redesigning the critical delay paths of the circuit. The requirement for manual intervention in the design process defeats the purpose of a semi-custom or module generator based design environment. In an automated design environment the objective is to decrease the need for human intervention as much as possible, while maintaining a high standard of performance. In order to achieve the desired performance automatically it is necessary to design CAD tools which are capable of performing detailed analysis of circuit behavior, and perform the redesign of critical paths automatically. Many techniques have been developed which increase the sizes of transistors along the critical path to decrease propagation delays [1, 2, 3, 4, 5, 6, 7, 8]; however, this technique is often costly in layout area. Furthermore, increasing transistor sizes to decrease propagation delays must be performed iteratively, since one does not want to oversize devices because of the penalty being paid for an increase in layout area. Since transistor sizing is an iterative technique, it can require large amounts of computation time.

The technique proposed in this paper for reducing the propagation delays in VLSI circuits is based on adjusting the order of the transistors in a MOS logic gate according to the input arrival patterns of its input signals. It has been shown in [9] that the order of transistors in a MOS logic gate can have a significant effect on the propagation delay of the gate. Therefore, a naive placement of devices in a MOS logic gate could lead to poor performance or unnecessary transistor sizing. For example, consider the two CMOS NAND gates shown in Fig. 1. It is assumed that the signal I is the *critical input* of the gate; that is, I is the signal which causes the output of the gate to switch. The circuits shown in Figs. 1(a) and (b) represent two different transistor reorderings of the same circuit; that is, the logical behavior of the two circuits are identical, but they have different transistor orders. The ratio of the output high to output low propagation delay of the circuit shown in Fig. 1(a) to that shown in Fig. 1(b) is 1.23. This ratio is called the *delay ratio* and is an indication of the potential improvement in propagation delay due to transistor reordering. The potential improvement in circuit speed using this approach is bounded; however, the improvement is made with nearly zero penalty (i.e., very little, if any, increase in layout area). This technique can be used in conjunction with transistor sizing. Since selecting a proper transistor order reduces the propagation delay, using this technique in conjunction with transistor sizing can improve circuit performance with less increase in layout area than would otherwise be the case.

The problem of assigning input signals to input pins addressed in [10] is not identical to the

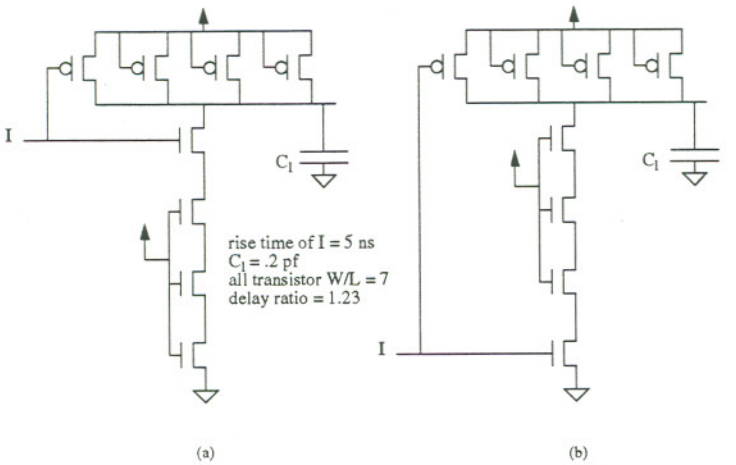

Fig. 1. Two logically equivalent circuits with different propagation delays.

transistor reordering problem which is addressed in this paper. Since the transistors in a CMOS complex gate can have different sizes, the characteristics of a signal (e.g., transition time and delay) driving a gate may be different depending on the pin to which the signal is assigned (i.e., the delay and transition time of a signal are a function of the size of the device which the signal is driving). The pin assignment is being determined based on the arrival patterns of the inputs, and the arrival patterns are a function of the assignment. It is not feasible that a good assignment of signals to pins can be determined in reasonable time because of this interdependence. In contrast to this the algorithm presented in this paper can determine the best transistor order for a logic gate in constant time, since a model is used which is based on the results of SPICE simulations. In the transistor reordering problem the signal to pin assignment is fixed, and the actual positions of the transistors in the gate are altered. The effect of optimizing the transistor order on the propagation delay of the gate (i.e., the local effect) is the same as the effect of optimizing the signal to pin assignment; however, since the signals are driving the same load after the transistor reordering, this method has little effect on the global circuit. Since changing the assignment of signals to pins can change the load capacitance of a fan-in signal of the gate, altering the signal to pin assignment has a global effect on the circuit. If the signal to pin assignment is altered, it is possible that a

non-critical logic path will become critical; however, it is not possible for a non-critical logic path to become critical after transistor reordering, since the loads on the non-critical paths have not been changed. A logic gate can be contained in many logic paths, one of which has greater delay than the others. If the signal to pin assignment of a gate is changed to decrease the delay of the critical path, then it is possible (due to a change in load capacitance) that a non-critical path becomes critical. However, if the transistors of the gate are reordered, the net effect is a decrease in the propagation delays of all the logic paths containing the gate. Since transistor reordering does not have a global effect on the circuit, it is only necessary to recompute the delay of the gate whose transistors have been reordered. Since practical CMOS complex gates do not have more than 20 to 30 transistors, it is possible to perform the simulation using SPICE. In the case of altering the signal to pin assignment, it is necessary to recompute the delays of all the gates whose outputs are inputs of the gate whose signal to pin assignment has been altered. Since this may involve a very large number of gates, simulation will require a very long time if performed by SPICE, or may not be accurate if performed by a switch level simulator.

In [10] the authors address the issue of assigning input signals to input pins in a standard cell layout environment. The objectives of the algorithm presented in [10] are identical to ours (i.e., decrease the propagation delay of logic gates); however, their algorithm has the potential to increase overall circuit delay if the transistors of a gate have different sizes. Furthermore, their technique suffers from limitations that are overcome by our circuit model. Necessary for their algorithm is that the standard cell library be parameterized with information which is inherent in our circuit model. Specifically, for a given standard cell they require that the intrinsic delay of a gate with respect to each input signal and parasitic resistance and capacitance be stored as cell parameters. The intrinsic delay of a gate with respect to some input signal i is the delay of the gate assuming i is the critical input signal. Our algorithm does not require cell parameterization. Moreover, our algorithm does not require that the CMOS complex gates be predesigned cells, and therefore it can be used in any semi-custom design environment including gate arrays. The authors of [10] claim that their algorithm can be used to enhance the results of transistor sizing algorithms; however, since their algorithm is dependent on cell parameterization, it cannot be used on cells for which the transistor sizes differ from the cells contained in the cell library. Since our algorithm uses a general model to determine which transistor order is better, it can operate on circuits of arbitrary functionality and transistor size.

This paper is organized as follows. The problem is formulated in the next section, and the circuit model is presented in Section III. The algorithm is described in Section IV, and experimental

results are presented in Section V. This paper is concluded in Section VI.

II Problem Formulation

The input signals of a given CMOS complex gate are derived from primary inputs and register outputs via logic networks. These logic networks implement different functions, and therefore the signals they produce have different timing characteristics (e.g., delay and transition time). It has been shown in [9] that these characteristics can have a significant effect on the timing behavior of the gate which they are driving. The effect of the timing characteristics of the input signals on a gate is dependent on other parameters such as the load capacitance of the gate, the sizes of the transistors in the gate and the length of the path through which the load capacitance of the gate is charged and discharged [9]. The effect of the input signals on the timing behavior of the gate can be changed by repositioning the transistors with respect to the output and power rails of the circuit. Of course the repositioning of the transistors must preserve the logical behavior of the circuit. The repositioning of transistors is called *transistor reordering*, since the most obvious circuit to which transistor repositioning may be applied is a series chain of transistors, and the result is a reordering of the transistors. Consider the digital circuit shown in Fig. 2. The problem is to determine the transistor order for transistors w, x, y and z, such that the output high to output low propagation delay of the 4-input NAND gate is minimum. SPICE simulations reveal that the output high to output low delay of the NAND gate as shown is .983 nanoseconds, and that by interchanging transistors w and z the high to low propagation delay is reduced to .808 nanoseconds (a reduction of 17.8 percent). The delay of the critical path of the logic network shown in Fig. 2 is 7.983 nanoseconds before reordering the transistors of the NAND gate, and 7.808 nanoseconds after reordering the transistors of the NAND gate. The effect of reordering the transistors of the NAND gate on the path delay is only a 2.2 percent improvement; however, choosing the proper transistor order for each of the gates inside the combinational logic block A of Fig. 2 can further decrease the path delay. As will be shown in the sequel the effect of reducing the propagation delays of individual gates on the path delay is significant.

The following assumptions are made about the circuit.

1. The digital system is synchronous, and the propagation delays are measured between register boundaries.
2. The input description is at the transistor level, and consists of multiple input, single output static CMOS complex gates, transmission gates and latches.

Our algorithm uses a simulator to determine gate delays and signal transition times. It is possible

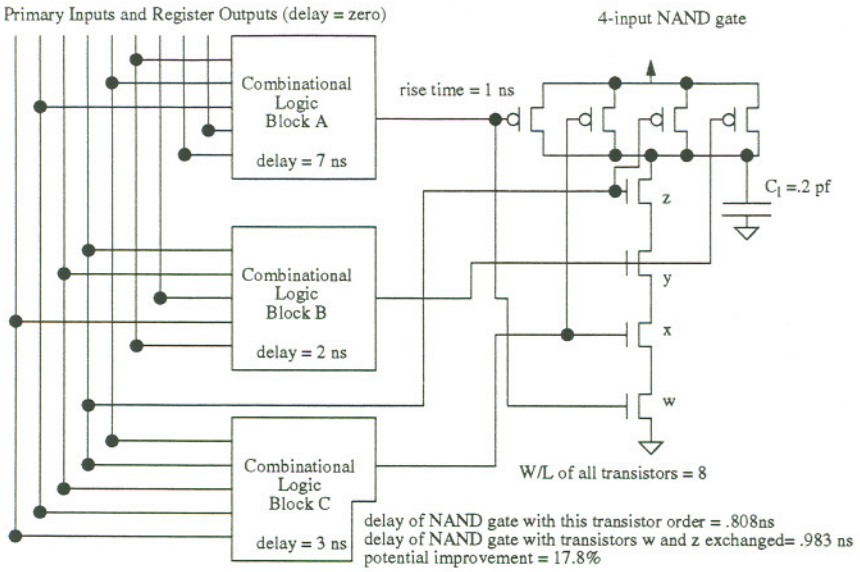

Fig. 2. An example for which transistor reordering produces a significant reduction in propagation delay.

to use any simulator which determines delays and transition times; however, it is assumed that the simulator is sufficiently accurate. The SPICE simulator is used for all the examples presented in this paper. The objective of this work is to determine a model which successfully identifies in a very short time the best transistor order for a given CMOS complex gate, and to provide an algorithm for performing the reordering. Ideally one would like to perform SPICE simulations to determine the best transistor order; however, this is not a feasible solution due to the large number of reorderings and the excessive computing time that is required by SPICE. Therefore, a model is presented which estimates the best transistor order for a CMOS complex gate. The model is based on SPICE simulations, and measures the relative difference between the propagation delays of the circuit for different transistor orders.

A *logic path* in a circuit is a set of CMOS complex gates $\{G_1, G_2, \ldots, G_n\}$, such that the inputs to G_1 are either register outputs or primary inputs, the output of G_n is either a register input or primary output, and the output of G_i is an input to G_{i+1} for $1 \leq i < n$. The *delay* of a logic path is the sum of the propagation delays of the gates contained in the path. The *critical logic path* is the logic path in a circuit which has the greatest delay. The algorithm presented in this paper reduces the propagation delay of CMOS complex gates. Since the objective is to minimize

the delay of logic paths, it must be shown how the delay reduction of individual gates affects the total path delay.

Let δ_i be the propagation delay of gate i in the logic path P. The delay of P, denoted by $\Delta(P)$, is

$$\Delta(P) = \sum_{\forall i \in P} \delta_i.$$

Let δ_i^r denote the delay of gate i after reordering, and $\rho_i = 1 - \frac{\delta_i^r}{\delta_i}$ be the improvement in propagation delay due to the reordering.

Lemma 1 *Let $\sigma = \rho_i$, such that $\rho_i \leq \rho_j, \forall j \in P$, $\omega = \rho_k$, such that $\rho_k \geq \rho_j, \forall j \in P$, and R_P be the delay reduction in P due to the reordering of transistors in the gates of P. The following relation gives an upper and lower bound on the delay reduction of a logic path P.*

$$\sigma \leq R_P \leq \omega$$

Proof: The delay of P prior to transistor reordering is

$$\Delta(P) = \sum_{\forall i \in P} \delta_i,$$

and after transistor reordering is

$$\Delta^r(P) = \sum_{\forall i \in P} \delta_i^r = \sum_{\forall i \in P} (1 - \rho_i)\delta_i.$$

Since σ is the smallest $\rho_k, \forall k \in P$,

$$\sum_{\forall i \in P} (1 - \rho_i)\delta_i \leq (1 - \sigma) \sum_{\forall i \in P} \delta_i,$$

and

$$\frac{\sum_{\forall i \in P}(1 - \rho_i)\delta_i}{\sum_{\forall i \in P} \delta_i} \leq 1 - \sigma.$$

Since

$$R_P = 1 - \frac{\Delta^r(P)}{\Delta(P)},$$

$$\frac{\sum_{\forall i \in P}(1 - \rho_i)\delta_i}{\sum_{\forall i \in P} \delta_i} = 1 - R_P \leq 1 - \sigma.$$

Hence,

$$R_P \geq \sigma.$$

The proof for the upper bound is similar. □

The previous lemma shows that reducing the propagation delays of individual gates may have a significant global impact on the delay of the circuit.

III Circuit Model

The circuit model used to determine the best transistor order is based on data obtained from SPICE [11] simulations, and derived using curve fitting techniques. SPICE simulations have been performed on a CMOS NAND gate for a range of load capacitances, critical input signal transition times, stack heights and transistor sizes. Given a load capacitance, critical input signal transition time, stack height and transistor sizes, the NAND gate is simulated for two reorderings. The two reorderings are the cases where the transistor driven by the critical input signal is connected to the output and ground nodes of the circuit. It has been verified by SPICE simulation that it is necessary to consider only these two reorderings, since the propagation delays of all other reorderings fall in an interval bounded by these two. Given a circuit, the ratio of the propagation delays of the two reorderings is recorded. This ratio is called the *delay ratio.* Shown in Figs. 3(a), (b), (c) and (d) are the delay ratio versus load capacitance, critical input signal transition time, stack height and transistor size, respectively. In each of the figures only one of the parameters is varying. Fig. 3(b) shows that the delay ratio appears to be a linear function of the critical input signal transition time. An examination of each of the curves in Fig. 3 yields the following results. Let the function f denote the delay ratio, the variable x denote the specified parameter in the table below and

$$\Theta(g(n)) = \{f(n) | \exists c, d, n_0 \cdot \ni \cdot \forall n \geq n_0, cg(n) \leq f(n) \leq dg(n)\}[12].$$

parameter	behavior
load capacitance	$f(x) \in \Theta(\frac{1}{x})$
critical input transition time	$f(x) \in \Theta(x)$
stack height	$f(x) \in \Theta(x)$
transistor size	$f(x) \in \Theta(x)$

Given a CMOS complex gate it is necessary to consider only the reorderings where the critical transistor is nearest the output node and power node of the circuit. Since it is assumed that all non-critical inputs and the internal source/drain nodes of a gate have reached steady state, the gate may be modeled by a two input gate with two transistors in series. Therefore, in the model for the delay ratio there are parameters for the size of the critical transistor and the non-critical transistor. The non-critical transistor size parameter is an approximation of all the non-critical transistors in the gate. Since the effective channel resistance of a MOSFET is proportional to the ratio of the length to the width of the channel, the width to length ratios of the non-critical transistors of the circuit are combined according to the rules for computing equivalent resistances. To compute the size approximation of transistors connected in parallel (respectively, series) it is necessary to sum the width to length (respectively, length to width) ratios of the transistor channels. Consequently,

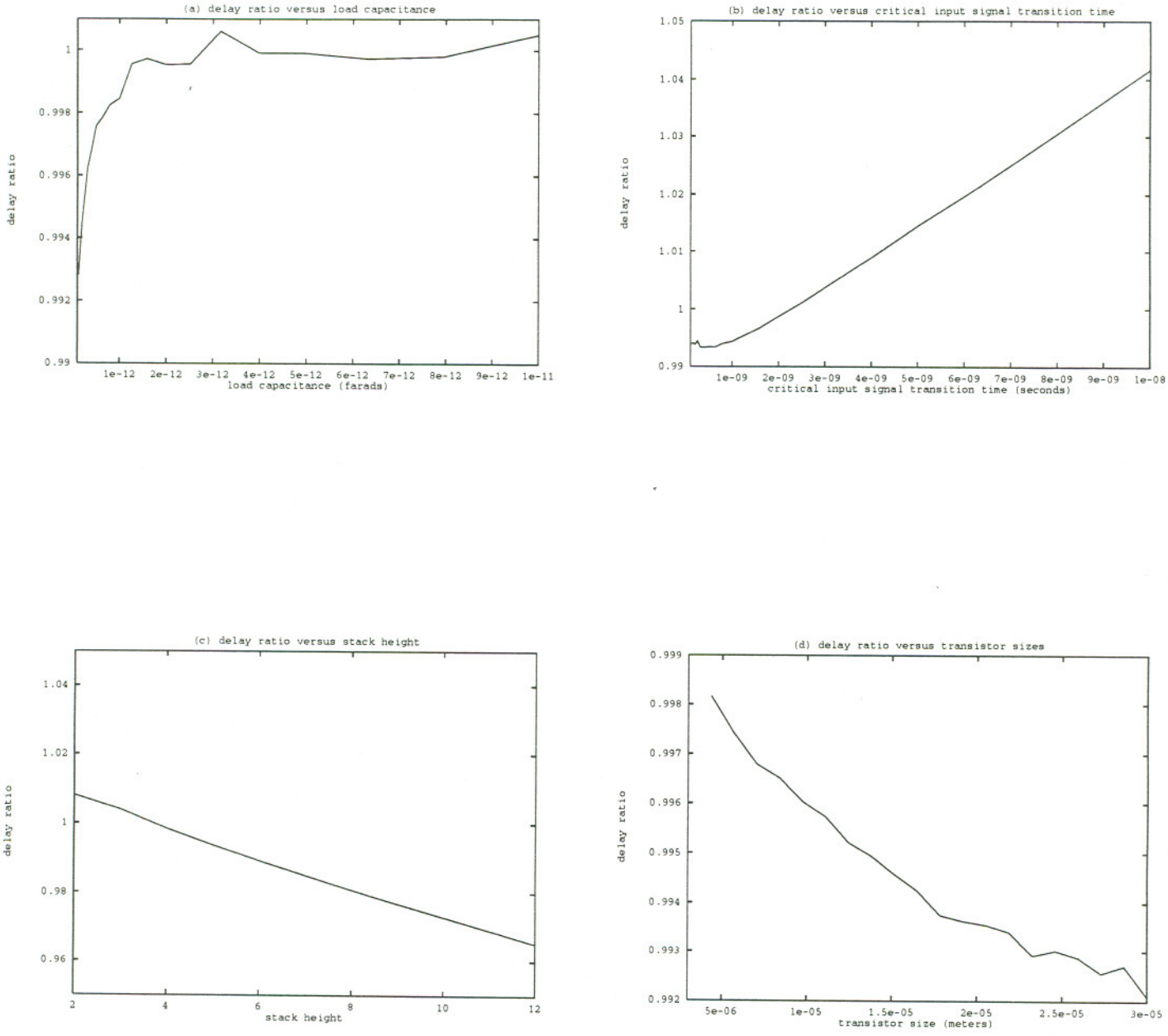

Fig. 3. Data obtained from the SPICE simulations of a CMOS NAND gate, and used to parameterize the transistor reordering model.

an increase in the stack height of a gate may be approximated as an increase in the length of the non-critical transistor.

Further examination of the data obtained from the SPICE simulations reveals that the effects of these parameters on the delay ratio are not mutually independent. For example, shown in Figs. 4(a), (b) and (c) are plots which show how the effect of load capacitance on the delay ratio varies for different critical input signal transition times, stack heights and transistor sizes, respectively. From Fig. 4(a) it appears that the critical input signal transition time has an effect on the effect of load capacitance on the delay ratio when the load is small; however, it has little effect for large loads. From Fig. 4(b) it appears that the effect of load capacitance on the delay ratio is also dependent on the stack height (or length of the non-critical transistor). Similar observations are made for critical input signal transition time, stack height and transistor size. The following general equation is used for the delay ratio.

$$c_1\frac{wyz}{x} + c_2\frac{wy}{x} + c_2\frac{wz}{x} + c_4wyz + c_5\frac{yz}{x} + c_6\frac{w}{x} + c_7wy + c_8wz + c_9\frac{y}{x} + c_{10}\frac{z}{x}$$
$$+ c_{11}yz + c_{12}w + c_{13}\frac{1}{x} + c_{14}y + c_{15}z + c_{16}, \qquad (1)$$

where the c_i are constants to be determined by curve fitting and w, x, y and z represent non-critical transistor size, load capacitance, critical input signal transition time and critical transistor size, respectively. The data used in the curve fitting was obtained from SPICE simulations using the device models and parameters shown in Fig. 5. The transistor parameters shown in Fig. 5 are from the MOSIS[1] $2\mu m$ CMOS p-well fabrication process. The constants in Eq. 1 have been evaluated using a least square approach and are given below.

c_1 = -9.40873e-09	c_9 = 5.44056e-06
c_2 = 5.69209e-06	c_{10} = 6.2696e-15
c_3 = -2.74994e-17	c_{11} = 1.28456e+06
c_4 = 21237.9	c_{12} = -0.0194441
c_5 = -2.92745e-06	c_{13} = -1.88367e-14
c_6 = -4.62322e-16	c_{14} = -5.02042e+06
c_7 = 6.9117e+06	c_{15} = -0.00336689
c_8 = 3.89665e-05	c_{16} = 1.01388

For these particular constants and the data used the mean error is 2.9% and the standard deviation is .0529. Approximately 4,000 data points were used in the curve fitting exercise, and it required approximately 15 minutes of computer time on a SUN SPARCstation 1.

A very fine quality of the delay ratio model is that an error caused by its inaccuracy is unlikely to affect the final result. Since the model is used only to determine which of the two

[1]MOSIS is the Metal-Oxide-Semiconductor Implementation Service located at the University of Southern California

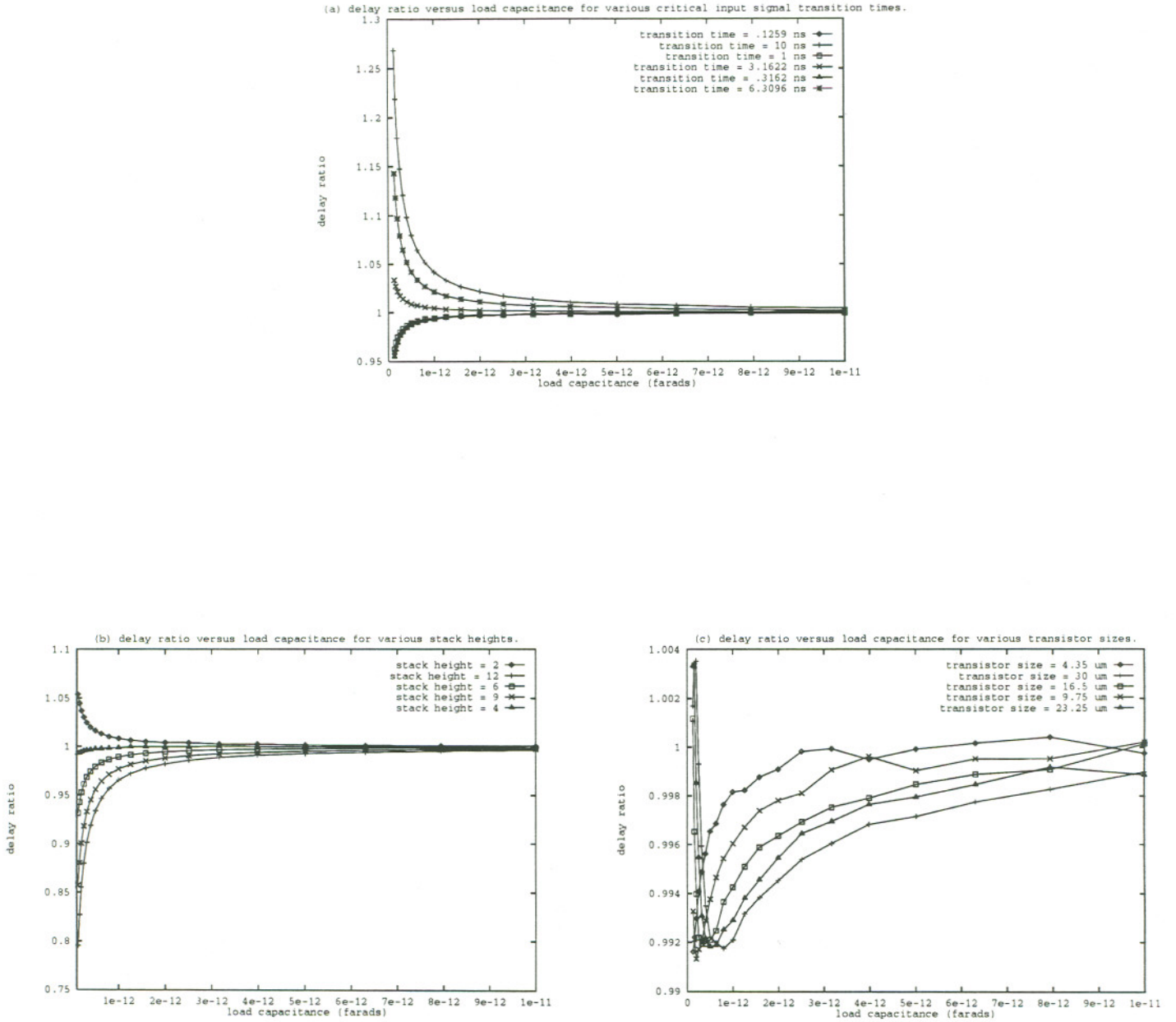

Fig. 4. The second order effect of critical input signal transition time, stack height and transistor size on the delay ratio as a function of load capacitance.

```
.MODEL CMOSN NMOS LEVEL=2.00000 LD=0.280000U TOX=500.000E-10
+NSUB=1.000000E+16 VTO=0.827125 KP=3.286649E-05 GAMMA=1.35960
+PHI=0.600000 UO=200.000 UEXP=1.001000E-03 UCRIT=999000.
+DELTA=1.24050 VMAX=100000. XJ=0.400000U LAMBDA=1.604983E-02
+NFS=1.234795E+12 NEFF=1.001000E-02 NSS=0.000000E+00 TPG=1.00000
+RSH=25 CGSO=5.2E-10 CGDO=5.2E-10 CJ=3.2E-4 MJ=0.5 CJSW=9E-10 MJSW=0.33
.MODEL CMOSP PMOS LEVEL=2.00000 LD=0.280000U TOX=500.000E-10
+NSUB=1.121088E+14 VTO=-0.894654 KP=1.526452E-05 GAMMA=0.879003
+PHI=0.600000 UO=100.000 UEXP=0.153441 UCRIT=16376.5
+DELTA=1.93831 VMAX=100000. XJ=0.400000U LAMBDA=4.708659E-02
+NFS=8.788617E+11 NEFF=1.001000E-02 NSS=0.000000E+00 TPG=-1.00000
+RSH=95 CGSO=4E-10 CGDO=4E-10 CJ=2E-4 MJ=0.5 CJSW=4.5E-10 MJSW=0.33
```

Fig. 5. MOSFET model and parameters used in SPICE simulations.

transistor orders is better, it is only necessary that the delay ratio be modeled accurately when it is close to one. Moreover, if a wrong decision is made, then it can be concluded that the delay ratio is very close to one and the effect of the wrong decision on the total path delay is likely to be insignificant (i.e., the propagation delays of the two reorderings are similar).

It is possible to obtain a more accurate model by making the general form (Eq. 1) a polynomial of higher degree. However, for transistor reordering purposes it is only necessary that the model predict which transistor order is better. Therefore, Eq. 1 is sufficient for our purposes.

IV Transistor Reordering Algorithm

The transistor reordering algorithm performs a breadth-first search through the logic network starting at the primary inputs and register outputs, and ending at the primary outputs and register inputs. A depth-first search strategy is employed to determine the transistor order of each logic gate. The best transistor order can be determined in constant time using the model described in the previous section; however, in the worst case reordering the transistors to realize the best transistor order requires computing time which is linear in the number of transistors. In its current form this algorithm is restricted to series/parallel circuits; however, it can be extended to operate on non-series/parallel circuits using the methods described in [9].

A Algorithm Description

Given a logic gate G let the *fan-in set* of G, $FI_G = \{G_1, G_2, \ldots, G_n\}$, be the set of logic gates whose outputs are inputs of G, and let the *fan-out set* of G, $FO_G = \{G_1, G_2, \ldots, G_n\}$, be the

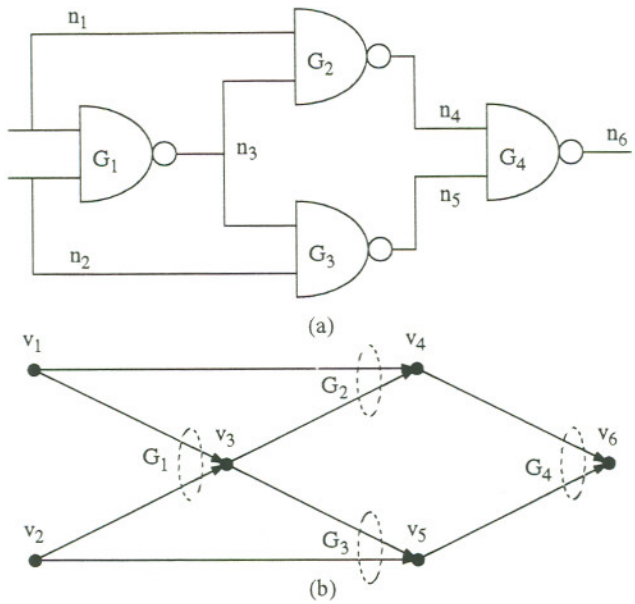

Fig. 6. A logic network and its graph representation.

set of logic gates such that the output of G is an input of $G_i (1 \leq i \leq n)$. A logic gate G is said to be a *primary logic gate* if $FI_G = \emptyset$; that is, all the inputs of G are either primary inputs or register outputs. A logic network N consists of a set of logic gates, and is represented by a directed graph $\Gamma(V, A)$, where V is the set of vertices and A is the set of arcs of Γ. The input and output signals of gates in N correspond to vertices in Γ, and there is an arc from vertex v_i to vertex v_j in Γ if and only if $\exists G_i \in N$, such that v_i corresponds to an input signal of G_i and v_j corresponds to an output signal of G_i. The set of incoming edges of a vertex in Γ represents a gate in the logic network N. For example, the logic network shown in Fig. 6(a) is represented by the directed graph shown in Fig. 6(b), where vertex v_i in the graph represents node n_i in the network and gate G_i is represented by the corresponding set of arcs in Γ. The directed graph Γ characterizes the flow of logical signals in the network. Complex gates as well as primitive gates can be represented by Γ. Since it is assumed that there is no feedback in N, Γ is acyclic. Associated with each vertex in Γ is the worst case arrival time and transition times of the corresponding signal in N, and associated with each arc is the worst case delay of its corresponding gate. The arrival time and transition times associated with a vertex $v \in V$ and delay time associated with an arc $a \in A$ are determined by simulation. Assuming the problem formulation presented in [10], associated with each arc $a \in A$ in the graph

```
Optimize_Order (N, Γ){
    place each primary gate of N in the queue;
    while (the queue is not empty){
        choose the first element in the queue, say G, such that FI_G = ∅;
        compute the delay ratio;
        if (delay ratio < 1)
            place the critical transistor near the output node of G;
        else
            place the critical transistor near the power rail of G;
        estimate the delay of G and the transition times of its output using SPICE;
        update Γ;
        for (each gate G_i ∈ FO_G){
            if (G_i is not in the queue)
                place G_i in the queue;
        }
        for (each G_i ·∋· G ∈ FI_{G_i})
            remove G from FI_{G_i};
        remove G from the queue;
    }
}
```

Fig. 7. The optimization algorithm.

representation Γ of the network N is the intrinsic delay of the gate with respect to the input signal corresponding to the vertex $v \in V$, where a is an outgoing arc of v. The authors of [10] assume that the intrinsic delay of a gate with respect to an input is independent of the characteristics (e.g., transition time) of the input signal. In our problem formulation the set of arcs in Γ corresponding to a gate in N are all associated with the same delay, and when the transistors are reordered the delay of the gate is recomputed. Since no assumptions are made about circuit delay, the accuracy of the worst case propagation delay estimates used in our algorithm are dependent solely on the accuracy of the simulator used.

The breadth-first search algorithm for processing the graph representation Γ of a logic network is shown in Fig. 7. The transistor order for a gate G may be determined after the transistor order for each gate in FI_G has been determined. Therefore, a breadth-first search is performed on Γ beginning with the vertices which correspond to primary logic gates (i.e., vertices with no incoming arcs). A gate G is chosen such that the transistor orders for all the gates in FI_G have already been determined. The delay ratio of the gate is computed using the model presented in the previous section, and the transistor reordering is performed. The transistor driven by the critical

input signal is then placed as close as possible to either the output node or power rail of the circuit depending on the value of the delay ratio. This is accomplished using a tree representation of the circuit so that logically equivalent permutations of transistors can be easily identified [9]. It may be the case that there is not a logically equivalent circuit, such that the critical transistor is connected to the output node or the power rail. In this case the critical transistor is placed as near as possible to the appropriate node. The delay of the gate and the transition times of its output must be recomputed after its transistors have been reordered, and they are determined by simulation. Any simulator can be used for this purpose; however, the quality of the results will depend on the accuracy of the simulator. Since the load capacitance and the transition times of the input signals of the gate are known, it is only necessary to simulate the gate whose transistors have been reordered. Therefore, the SPICE [11] simulator is used for all the examples presented in this paper. Since the complex logic gates are simulated individually, the computation time required by SPICE is not excessive.

V Experimental Results

The algorithm has been implemented in C on a SUN SPARCstation 2. The execution times for the circuits tested are dominated by the SPICE simulations, and are tabulated with the results. The results of the circuits tested are shown in Table I. The improvement in propagation delay due to transistor reordering is computed by comparing the worst case propagation delays of the best and worst transistor orders. Given in Table I is the overall improvement in the propagation delay of the critical path, the improvement in propagation delay of the gate on the critical path which has the least improvement of all gates on the critical path, and the improvement of the gate which has the greatest improvement. The improvement in delays of the gates with the least and greatest improvement of all gates on the critical path correspond to the lower and upper bounds on the improvement of the path delay, respectively.

The logic diagram of the 8-bit barrel shifter is shown in Fig. 8. The barrel shifter is constructed using 2-input multiplexers, and the circuits used to implement the 2-input multiplexers are shown in Fig. 9. All the logic paths of this particular circuit have nearly the same delay, since there are three 2-input multiplexers in each path. The critical input signals of each gate are the data inputs, since it is necessary for these signals to propagate through the multiplexers. The first gate in the critical path is the gate for which transistor reordering results in the least improvement, and the last gate is the gate for which transistor reordering results in the greatest improvement in delay. The block carry lookahead unit is implemented using 2-input NAND gates, and the rest of

Table I. Experimental results.

circuit	number of transistors	number of gates on critical path	of all gates on the critical path	
			least improvement (lower bound)	greatest improvement (upper bound)
full adder	32	1	8.3%	8.3%
4-bit block carry lookahead unit	34	6	11.8%	22.2%
9-bit parity generator	96	8	8.2%	19.5%
8-bit barrel shifter	240	3	20.6%	47.7%
5x5 array multiplier	998	4	4.7%	11.6%

Table I(Continued)

circuit	critical path delay (ns)		improvement in critical path delay	approximate CPU time (minutes)
	top-critical ordering	bottom-critical ordering		
full adder	13.2	12.1	8.3%	7
4-bit block carry lookahead unit	13.7	11.8	13.4%	5
9-bit parity generator	68.1	60.7	10.9%	6
8-bit barrel shifter	26.5	20.6	22.3%	10
5x5 array multiplier	98.2	88.8	9.6%	12

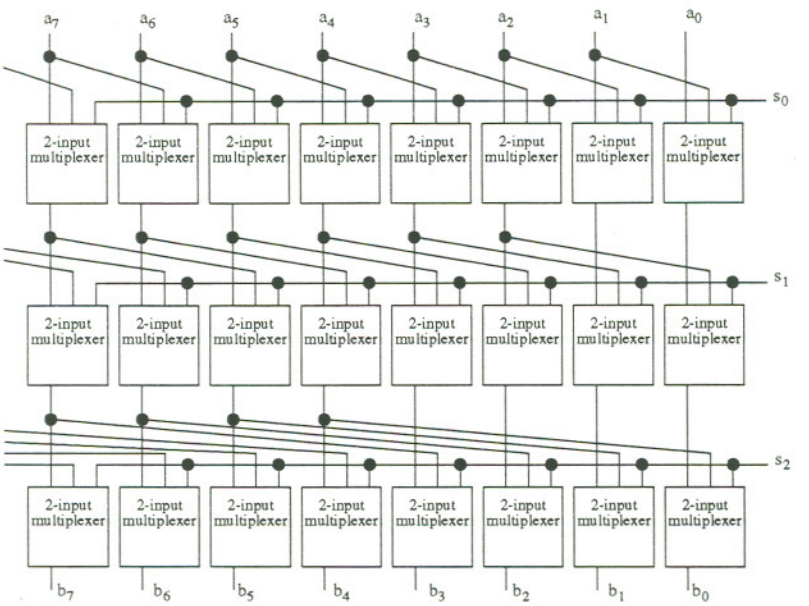

Fig. 8. An 8-bit barrel shifter used to test the transistor reordering algorithm.

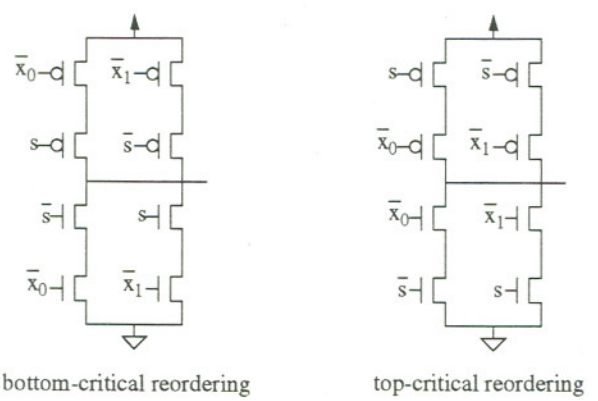

Fig. 9. The implementation of the 2-input multiplexer used in the design of the barrel shifter shown in Fig. 8.

the circuits in Table I are implemented with complex gates.

The results shown in Table I indicate that a substantial improvement in the propagation delay of digital VLSI circuits can be obtained by reordering the transistors in CMOS complex gates using the technique described in this paper. It is not necessary that the circuits be implemented using complex gates in order to achieve improvement in propagation delays by transistor reordering. The improvement obtained in the block carry lookahead unit indicates that significant improvement can be obtained in circuits which are implemented with primitive gates (e.g., NAND and NOR gates).

VI Conclusion and Future Work

The technique presented in this paper for optimizing the performance of MOS digital circuits with respect to the arrival patterns of the gate inputs is superior to the technique presented in [10] for the following reasons.

1. This technique can be used in any semi-custom design environment, and the technique presented in [10] is limited to a standard cell environment.
2. Transistor reordering does not have an adverse effect on the non-critical delay path of the circuit, unlike changing the signal to pin assignment.
3. Transistor reordering affects only the delay of the gate whose transistors have been reordered. Reevaluation of the circuit performance can be much more precise, since a small number of elements meed to be included in the simulation.
4. The model used in this algorithm can be easily parameterized for a new fabrication process, and the model used in [10] is dependent on the cell library.

Although the potential improvement in circuit speed by reordering transistors is bounded, significant reduction in propagation delays can be achieved. Furthermore, it is reported in [13] that experimental results indicate that transistor reordering can enhance the effects of transistor sizing. That is, the effect on the propagation delay of sizing the transistors of a logic gate depends significantly on the position of the critical transistor in the gate. Therefore, it is possible to achieve greater delay reduction with smaller increase in transistor size by performing transistor reordering in conjunction with transistor sizing. Although the layout system presented in [13] does perform transistor reordering in conjunction with transistor sizing, it assumes the best transistor order is placing the critical transistor nearest to the output node of the gate. It is clear from the results presented in this paper and from the results presented in [9] that it is not always the case that placing the critical transistor nearest to the output node in the gate results in the least propagation delay.

The algorithm presented in this paper can be used in conjunction with transistor sizing algorithms, since the model is developed for arbitrary CMOS complex gates with arbitrary transistor sizes.

The algorithm presented in this paper is currently limited to static CMOS circuits; however, it is applicable to MOS circuit structures other than static CMOS (e.g., NMOS, dynamic CMOS) as long as the delay ratio model is determined appropriately. That is, the simulation data must be obtained for the appropriate circuits, and the curve fitting performed. The current implementation of the algorithm is restricted to series/parallel circuits; however, with the techniques presented in [9] for representing non-series/parallel circuits by trees this algorithm can easily be extended to operate on non-series/parallel circuits.

Acknowledgements

The authors would like to thank Mr. Juihsiang Liu for developing the program to do the curve fitting.

References

[1] M.C. Chang and C.F. Chen. "PROMPT3 - A Cell-Based Transistor Sizing Program Using Heuristic and Simulated Annealing Algorithms". In *Proceedings of the Custom Integrated Circuits Conference*, pages 17.2.1 – 17.2.4, 1989.

[2] H.Y. Chen and S.M. Kang. "iCOACH: A Circuit Optimization Aid for CMOS High-Performance Circuits". In *Proceedings of the International Conference on Computer-Aided Design*, pages 372–375, 1988.

[3] J.P. Fishburn and A.E. Dunlop. "TILOS: A Posynomial Programming Approach to Transistor Sizing". In *Proceedings of the International Conference on Computer-Aided Design*, pages 326–328, 1985.

[4] K.S. Hedlund. "Aesop: A Tool for Automated Transistor Sizing". In *Proceedings of the Design Automation Conference*, pages 114–120, 1987.

[5] W.H. Kao, N. Fathi, and C.H. Lee. "Algorithms for Automatic Transistor Sizing in CMOS Digital Circuits". In *Proceedings of the Design Automation Conference*, pages 781–784, 1985.

[6] D.P. Marple and A.E. Gamal. "Optimal Selection of Transistor Sizes in Digital VLSI Circuits". In *Stanford Conference on Advanced Research in VLSI*, pages 151–172, 1987.

[7] J.D. Pincus and A.M. Despain. "Delay Reduction Using Simulated Annealing". In *Proceedings of the Design Automation Conference*, pages 690–695, 1986.

[8] J.M. Shyu, A. Sangiovanni-Vincentelli, J.P. Fishburn, and A.E. Dunlop. "Optimization-Based Transistor Sizing". *IEEE Journal of Solid-State Circuits*, 23(2):400–409, April 1988.

[9] B.S. Carlson. *"Transistor Chaining and Transistor Reordering in the Design of CMOS Complex Gates"*. PhD thesis, Syracuse University, 1991.

[10] M. Marek-Sadowska and S.P. Lin. "Pin Assignment for Improved Performance in Standard Cell Design". In *Proceedings of the International Conference on Computer Design*, pages 339–342, 1990.

[11] L.W. Nagel. *"SPICE2: A Computer Program to Simulate Semiconductor Circuits"*. PhD thesis, University of California, Berkeley, 1975.

[12] G. Brassard and P. Bratley. *Algorithmics: Theory and Practice*. Prentice Hall, 1988.

[13] A.E. Dunlop, J.P. Fishburn, D.D. Hill, and D.D. Shugard. "Experiments Using Automatic Physical Design Techniques for Optimizing Circuit Performance". In *Proceedings of the International Symposium on Circuits and Systems*, pages 847–851, 1990.