TILOS: A POSYNOMIAL PROGRAMMING APPROACH TO TRANSISTOR SIZING

J. P. Fishburn[1] and A. E. Dunlop[2]
AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Abstract
A new transistor sizing algorithm, which couples synchronous timing analysis with convex optimization techniques, is presented. Let A be the sum of transistor sizes, T the longest delay through the circuit, and K a positive constant. Using a distributed RC model, each of the following three programs is shown to be convex: 1) Minimize A subject to $T < K$. 2) Minimize T subject to $A < K$. 3) Minimize AT^K. The convex equations describing T are a particular class of functions called posynomials. Convex programs have many pleasant properties, and chief among these is the fact that *any point found to be locally optimal is certain to be globally optimal.* TILOS (TImed LOgic Synthesizer) is a program that sizes transistors in CMOS circuits. Preliminary results of TILOS's transistor sizing algorithm are presented.

1. Introduction

Given a synchronous MOS circuit of the form shown in Figure 1 with N transistors of sizes (channel widths) x_1, x_2, ..., x_N, the following question is considered: How can the circuit's performance be improved by adjusting the x_i? Two figures of merit are of special interest. T is defined to be the minimum clock period at which the circuit will operate. The other quantity, A, is simply the sum of transistor sizes. A is positively correlated with a number of other attributes of the circuit that should be minimized or constrained: These include silicon area, capacitance-discharge power, short-circuit power, and probability of a device failure within a chip.

TILOS (pronounced tee-los) is a program that requires a transistor connectivity file and I/O-delay file, and adjusts transistor sizes and connectivity within logical gates to meet the user's requirements for A and/or T. TILOS's output is a transistor connectivity file that can be passed to a layout program such as SC2 [5].

TILOS contains a static timing analyzer which recognizes latches and thus is capable of extracting all relevant timing paths from a circuit of the form shown

Authors are currently with [1]Agere Systems and [2]Crossbow Consulting.

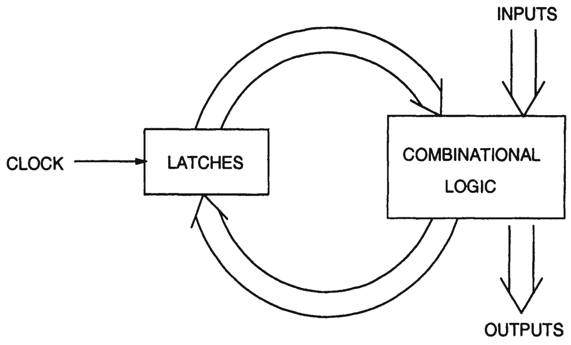

Figure 1. Memory/combinational-logic model of digital MOS circuits. A path begins at the output of a latch or an input, and ends at the input of a latch or an output.

in Figure 1. This recognition process is similar to that used in other static timing analyzers [1] [10] [6], and will not be further described here.

Several authors ([13] [3] [4] [7] [9]) have reported on optimization techniques for transistor sizing. Contributions of this work over previous approaches include: 1) Using a distributed RC model of delay, T is proved to be a convex function of the transistor sizes. T remains convex if wire widths are also considered to be variables. 2) TILOS couples static timing analysis with transistor sizing, relieving the user of the need to specify which paths are to be optimized. Rather, the user specifies desired behavior of I/O signals, including clocks, and TILOS determines what paths are in need of improvement. 3) Transistors in latches as well as combinational gates are sized. 4) TILOS sorts series-connected subnetworks in a complex gate so that the subnetworks with earlier-arriving inputs are closer to the power supply. This heuristic allows transistor sizing a chance to operate: A transistor with an earlier-arriving input can be made larger and still have time to turn on, despite the increased gate capacitance, before other inputs arrive, providing a lower-resistance path to power. In addition, the increased source & drain capacitance of the larger transistor helps, rather than hinders, the output transition.

2. Three Formulations of the Transistor Sizing Problem

Let K be a positive constant, and let A and T be as defined above. Consider the following three optimization programs:

1 Minimize A subject to the constraint $T < K$.

2 Minimize T subject to the constraint $A < K$.

3 Minimize AT^K.

The first formulation can be used when the circuit must fit inside a system with a given clock period K. The second formulation might be used when the circuit is to be made as fast as possible, subject to constraints on silicon area, power, or yield. The third formulation represents a wholistic approach in which both A and T are important, but have relative weights assigned to them.

3. MOSFET Model

The MOSFET model that TILOS uses is shown in Figure 2. The gate, source, and drain capacitances are all proportional to the transistor size X, and the source-to-drain resistance is inversely proportional to X.

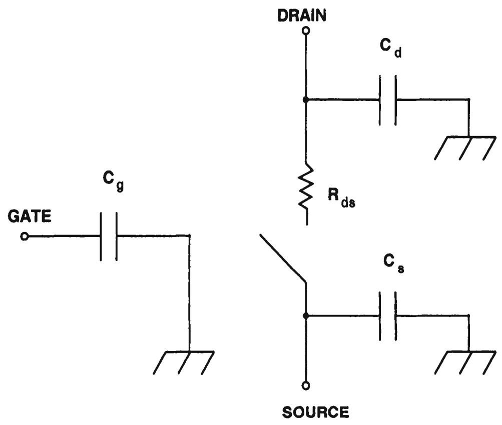

Figure 2. The source, drain, and gate capacitances are proportional to transistor size. The effective resistance is inversely proportional to transistor size.

4. RC Delay Model

Figure 3 illustrates the modeling of gate delay by a distributed RC network [11] [8]. In the RC network shown, an upper bound for the discharge time is

$$(R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3. \tag{1}$$

This represents a much tighter bound than a lumped R and C model.

With the MOSFET and RC-delay models, the delay through any series configuration of transistors can be expressed in terms of the transistor sizes and routing parasitics. The important point here is the form of (1) when expressed as a function of the transistor sizes x_i: Each R_i is proportional to $1/x_i$, and each C_i is some constant (for wire capacitance) plus one term for each transistor whose gate, drain or source is connected to the node. This term is proportional to the

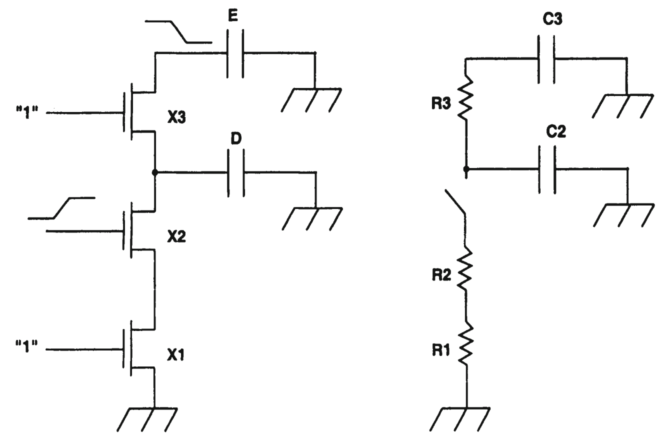

Figure 3. Delay through pulldown network of NAND gate is modeled with RC network.

transistor size. Thus (1) can be rewritten as:

$$(A/x_1 + A/x_2)(Bx_2 + Cx_3 + D) + (A/x_1 + A/x_2 + A/x_3)(Bx_3 + E) \qquad (2)$$

where A, B, and C are constant coefficients for resistance, drain and source capacitance, respectively, and D and E are wire capacitances. It is interesting to note that if wire widths, as well as transistor widths, are treated as variables, then the expression for delay remains in the same form as (2).

5. Delay Through Complex Gate

For each transistor in a pulldown or pullup network of a complex gate, the greatest resistance path from the drain to the gate output is computed, as well as the greatest resistance path from the source to a supply rail. Thus for each transistor, the network is transformed into an equivalent series configuration, and the calculation of the previous section can be applied.

6. Delay Through a Single Circuit Path

Every circuit path has two path delays: one for the case where the input to the path is rising, the other for the falling input. Since a path delay is simply a sum of gate delays as in (2), the general form of a path delay is as follows:

$$\sum_{i,j=1}^{N} a_{ij}\frac{x_i}{x_j} + \sum_{i=1}^{N} \frac{b_i}{x_i}, \qquad (3)$$

where the a_{ij} and b_i are nonnegative constants that are mostly zero. The function which (3) describes is *convex*, which means that any straight line segment in

N+1-dimensional space whose endpoints lie in the graph of the function is itself entirely on or above the graph.

7. Delay Through Entire Circuit Is a Convex Function of the x_i

The delay T through a single combinational block is defined as the maximum, over all possible paths through the block, of expressions of the form (3). Since convexity is preserved under sums and maxima, T is thus a convex function of the variables x_i. As a consequence, all three formulations considered in section 2 are the minimization of a convex function subject to an upper bound constraint on another convex function. This implies that any point found to locally minimize the objective function also globally minimizes it. The field of *Convex Programming* [12] has been intensively explored over the last several decades, and many techniques are available. In addition, (3) belongs to a special class of functions called *posynomials*, and the more specialized techniques of *Geometric Programming* [2] can be used. Techniques such as simulated annealing or multiple random starts are unnecessary.

8. Tailoring Convex Programming Techniques to Transistor Sizing

Since we are interested in applying the sizing algorithm to circuits as large as an entire VLSI chip or perhaps to an entire system, the algorithm must be as efficient as possible, perhaps at the expense of absolute accuracy in finding the optimum. In our experience, the algorithm described below provides an efficient method for converging to the optimum point.

TILOS proceeds as follows: Starting with minimum sizes for all transistors a static timing analysis is performed on the circuit, which assigns two numbers to each electrical node: t_l (latest time to go low), and t_h (latest time to go high). From each path output that fails to meet its timing goals for t_l or t_h, TILOS walks backward along the failing path. Whenever a node X is visited, TILOS examines in turn each NFET (if X's t_l is failing) or PFET (if X's t_h is failing) which could have an affect on the path. In general, this includes both the *critical transistor* (the transistor whose input is on the critical path), and *supporting transistors* (transistors along the highest-resistance-to-power-supply path from the source of the critical transistor). TILOS calculates the *sensitivity* of each such transistor i, which is the time savings accruing per increment of x_i. The size of the transistor with the largest sensitivity is increased, and the process is repeated.

Figure 4 illustrates a series configuration in which the critical path extends back along the gate of the top (critical) transistor. The sensitivity calculation for this transistor is derived as follows: Fix all transistor sizes except x, the size of

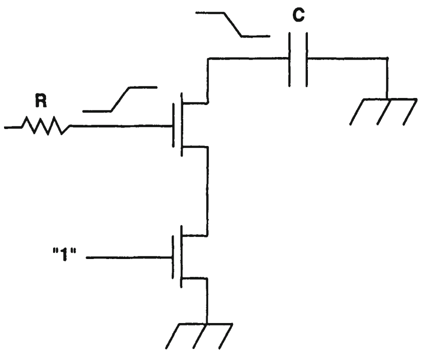

Figure 4. The critical path extends back along the gate of the top transistor.

the critical transistor. R is the total resistance of an RC chain driving the gate. C is the total capacitance of an RC chain being driven by the configuration. Then the total delay D(x) of the critical path is

$$D(x) = K + RC_u x + \frac{R_u C}{x},$$

where R_u and C_u are resistance and capacitance of a unit-sized FET. K is a constant that depends on the resistance of the bottom transistor, capacitance RC chain, and resistance in the driven RC chain. The sensitivity $D'(x)$ is then

$$D'(x) = RC_u - \frac{R_u C}{x^2}.$$

The sensitivity calculation for supporting transistors is done in a similar way. When $D'(x)$ is set equal to zero, the resulting value of x, which minimizes delay, is equal to a constant times

$$\sqrt{C/R}. \tag{4}$$

An interesting consequence of (4) occurs in the special case when all inputs are equally critical. Since the quantity C includes capacitance of all sources and drains of FETS higher in the series configuration, the transistor sizes that produce minimal delay are smaller near the output, and larger near the power supply. This is similar to the "pyramid shaped" nand gates of Shoji [14].

The sizing process terminates when either the constraints are met, or when the circuit has passed its absolute minimum and is getting slower instead of faster. Since the number of paths through a circuit can be very large in comparison to the size of the circuit itself, the optimization is performed without ever actually keeping track of a sensitivity (Lagrange Multiplier) for each critical path, or even enumerating all paths.

9. Some Preliminary Results

TILOS currently consists of 2705 lines of C, Lex, & Yacc source code. Design started in January, and coding in March of 1985. Recently sorting of series subnetworks, recognition of latches, and sizing of transistors within latches were added. The following table summarizes TILOS's performance on 4 sample circuits. The first 3 are 2, 8, and 32-bit adders, and the fourth is a standard-cell finite-state machine with 2 static flip-flops. The table gives the number of transistors, T (ns) and A (microns) before and after sizing, and the number of seconds of TILOS execution on a 68000-based workstation.

Speedup Due to Transistor Sizing

		Unsized		Sized		
Name	FETs	T ns	A mic.	T ns	A mic.	Ex. sec.
add2	56	14	210	8	278	6
add8	224	42	840	18	986	49
add32	896	154	3360	58	3609	1424
fsm	180	30	675	8	744	97

10. Directions for Future Research

The major source of error in the distributed RC timing model is the lack of consideration for slowly rising inputs. Unfortunately, it has not been possible to prove, as it was for the distributed RC model, that delay through the circuit is a convex function of the transistor sizes when the input waveform shape is taken into account. Although there are several static timing analyzers and a transistor sizer [9] that take into account input waveform shape, we hesitate to do so without a convexity proof in hand. If a more accurate model turns out to be non-convex, there is always the danger that the optimizer might become trapped in a local minimum that is not a global minimum, resulting in a more pessimal solution than the less accurate model.

11. Acknowledgments

The authors would like to thank T. G. Szymanski, D. M. Gay, and E. Rosenberg for many helpful discussions.

References

[1] V. Agrawal. Synchronous path analysis in MOS circuit simulator. In *Design Automation Conf.*, pages 629–635, 1982.

[2] J. G. Ecker. Geometric programming: methods, computations and applications. *SIAM Review*, 22(3):338–362, July 1980.

[3] L. A. Glasser and L. P. J. Hoyte. Delay and power optimization in VLSI circuits. In *Design Automation Conf.*, pages 529–535, 1984.

[4] K. S. Hedlund. Electrical optimization of PLAs. In *Design Automation Conf.*, pages 681–687, 1985.

[5] D. D. Hill. SC2: A hybrid automatic layout system. In *Int. Conf. on Computer Aided Design*, pages 172–174, 1985.

[6] N. Jouppi. Timing analysis for nMOS VLSI. In *Design Automation Conf.*, pages 411–418, 1983.

[7] C. M. Lee and H. Soukup. An algorithm for CMOS timing and area optimization. *IEEE J. of Solid-State Circuits*, 19:781–787, October 1984.

[8] T. M. Lin and C. Mead. Signal delay in general RC networks with application to timing simulation of digital integrated circuits. In *Conf. on Advanced Research in VLSI*, pages 93–99, 1984.

[9] M. Matson. Optimization of digital MOS VLSI circuits. In *Proc. Chapel Hill Conf. on VLSI*, pages 488–491, 1985.

[10] J. Ousterhout. Switch-level delay models for digital MOS VLSI. In *Design Automation Conf.*, pages 542–548, 1984.

[11] P. Penfield and J. Rubinstein. Signal delay in RC tree networks. In *Proc. 2nd Caltech VLSI Conference*, pages 269–283, 1981.

[12] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[13] A. E. Ruehli, P. K. Wolff, and G. Goertzel. Analytical power/timing optimization technique for digital system. In *Design Automation Conf.*, pages 142–146, 1977.

[14] M. Shoji. Electrical design of BELLMAC-32A microprocessor. In *Int. Conf. on Circuits and Systems*, pages 112–115, 1982.