Transistor Sizing and Gate Sizing Using Geometric Programming Considering Delay Minimization

Gracieli Posser, Guilherme Flach, Gustavo Wilke, Ricardo Reis
{gposser,gaflach,wilke,reis}@inf.ufrgs.br

Universidade Federal do Rio Grande do Sul (UFRGS)
Instituto de Informática – PPGC/PGMicro
Av. Bento Gonçalves 9500 Porto Alegre, RS - Brazil

Abstract

It is presented a comparison between gate sizing and transistor sizing to analyze the trade-off between execution time and minimum delay achieved. The transistor and gate sizing tools are based on Geometric Programming (GP) and delay is calculated using the Elmore delay model. Tests were made mapping ISCAS'85 benchmark circuits for 45nm technology considering delay minimization. First, circuits were mapped to a typical standard cell library. Then, the gate sizing and transistor sizing were performed. Gate sizing reduced the delay by 21%, in average, for a same area and power values of the sizing provided by standard-cells library. After transistor sizing reduced delay by 40.4% and power consumption by 2.9%, in average, compared with gate sizing. However, transistor sizing requires a bigger computing time, using a number of variables twice higher than with gate sizing.

1. Introduction

Circuit delay may be reduced by properly set transistor sizes. A larger transistor has an increased ability to carry (discharge) a load, decreasing the time required to change a signal from 0 (1) to 1 (0). However, a larger transistor also imposes a larger load to be loaded (discharged) by its driver. Therefore, choosing the right transistor size is not a trivial problem. The optimal size should be the one where the gate is capable of supplying the load attached to it without producing a high load to its driver.

One efficient way to model transistor (gate) sizing problem is via Geometric Programing (GP) [1]. GP is a mathematical optimization method able to find a global optimum if one exists, in a reasonable time.

For GP modeling, delay and area equations should be described using posynomial functions. Delay can be easily casted to a posynomial function using Elmore delay model [2] while transistor area is a linear function of the transistor width, which, by its turns, is a posynomial function.

In transistor sizing to each transistor of a logic cell can be set a different scale factor W that represents the transistor width. Gate sizing is a special case of transistor sizing where transistors of a same gate are tied to a same scale factor X considering different widths for PMOS (Wp) and NMOS (Wn) transistors. This is more restrictive than transistor sizing, but the reduced number of unknowns leads to a lower execution time.

In this work, both transistor and gate sizing are performed. The contributions are:

• a transistor and gate sizing tool based in [3] and [4] works that uses Geometric Programming;

• a transistor and gate sizing tool configurable to several manufacturing technologies;

• a comparison between gate sizing and transistor sizing taking account the tradeoff between number of variables (execution time) and circuit total delay reduction.

Section II presents some transistor sizing and gate sizing related works. In section III, we show the problem formulation. The tool development is presented in Section IV. Section V shows the comparison results, where we size the benchmark circuits mapped to a commercial cell library using gate sizing and transistor sizing methods, where the objective is to minimize delay. Conclusions are drawn in section VI.

2. Related Works

TILOS [5] was the first algorithm to attempt at transistor sizing, using Elmore delay model [2]. It identifies a critical delay path and uses a heuristic method to reduce the delay along this path.

[6] is a classical work in transistor sizing, where the objective is to minimize area considering a delay specification. Delay is modeled as posynomial functions of transistor size, creating a geometric programming problem that is transformed into a convex program, guarantying to find the exact solution. [3] presents a method for transistor sizing, where the sizing problem is modeled and solved by Geometric Programming. Gates are modeled using the Switch-Level RC Gate Model. In this model, a gate is viewed as a set of RC trees, one for each possible input vector. Gate delay is the maximum delay generated by its compound RC trees.

The most widely way to solve the gate sizing problem is the logical effort method [7], which provides fast heuristics or design guidelines for solving the gate-sizing problem approximately. Linear Programming is used in [8] and [9]. In [10] and [11] is used Non-Linear Programming.

The traditional gate sizing [4] and transistor sizing methodologies [6], [3] use Elmore delay to model delay as posynomial functions allowing the gate sizing to be formulated as a Geometric Program.

In [4] is showed a gate sizing method, where a scale factor X_i is associated to each gate. These variables are the optimization variables of the GP. Circuit area is the sum of the area of each port that makes up the circuit. The RC product gives the path delays and circuit delay is the maximum delay among all circuit paths.

3. Problem Formulation

Equation (1) shows the formulation of the optimization problem for delay minimization, considering that D values are the delays of the circuit paths. X_{min} and X_{max} are the minimum and maximum size of the gate, in a gate sizing optimization or the minimum and maximum allowed transistors width, when transistor sizing is executed. The maximum transistor width was defined by the maximum transistor width found in a cell library with a same manufacturing technology. $C_{in}^{\max}$ is the maximum input capacitance acceptable to the circuit, avoiding a high load to its driver. A_{max} is the maximum circuit area.

$$\begin{aligned} &\text{minimize} && D = max(D_1 ... D_n) \\ &\text{subject to} && X_{\min} \le X_i \le X_{\max} \\ & && C_{in} \le C_{in}^{max} \qquad A \le A_{max} \end{aligned} \tag{1}$$

4. Tool Development

Our transistor and gate sizing tool using GP was developed as follow:

1) The logic gates are modeled using the Switch-Level RC Gate Model [3]. In this model, a gate is viewed as a set of RC trees one for each possible input vectors and the gate delay is the maximum delay generated by its compound RC trees.

2) In the transistor sizing, for each transistor is set a variable that represents the transistor width. For gate sizing, is used a scale factor to each gate that multiplies the widths of the gate transistors. These are the problem optimization variables and they affect the total area, power consumption and circuit speed.

3) Capacitance and resistance values used to calculate delay and power are obtained using SPICE simulations for a PMOS and NMOS transistors Transistor capacitances are proportional to the transistor width and the driving resistance is approximately inversely proportional to the transistor width.

4) Delay is calculated using the Elmore delay model, which produces posynomial functions, enabling the problem solution by Geometric Programming and circuit delay is the maximum delay among all circuit paths.

5) Area is the sum of the width W_i of each transistor *i* that make up the circuit, where *n* is the number of transistors of the circuit:

$$A_{total} = \sum_{i=1}^{n} W_i \tag{2}$$

6) Switching power is calculated considering all capacitances of the circuit: load capacitance (C_{load}) and input capacitances (C_{in}). *Vdd* is the circuit voltage. α is the switching probability, which we consider 20% and *f* is the clock frequency, that we set to 500MHz for our tests.

$$P = (C_{load} + \sum_{i=1}^{n} Cin_i) * Vdd^2 * \alpha * f \tag{3}$$

5. Comparison Results

A set of the ISCAS'85 benchmark circuits were mapped using RTL Compiler tool [12] from Cadence to a 45nm library considering only CMOS cells. These circuits were inserted in our sizing tool where the area, timing and power values are calculated. The area value from circuits using standard cells is used as restriction to size the circuit in our implementation. Using this description, we size the gates that compose the circuit using the formulation (1). The comparison results between our gate sizing tool using Geometric Programming (GP) and the sizing available in a typical standard cell library (SC) are presented in Tab. 1, where delay, area, power values and their reductions are shown in percentage. Circuits sized using our methodology (GP) obtained a reduction, on average, of 21% in delay, keeping the same area and power values of the sizing provided using standard-cells library.

Tab. 2 shows the values of the circuits sized by gate sizing (GS), Tab. 1, and values for transistor sizing (TS). It is considered the number of variables used by the two methods, i.e., the execution time for each

method. The reductions (R) of power, timing, area and number of variables are showed in percentage. Negative values indicate that the gate sizing (GS) yields a better result than transistor sizing (TS), which are some values of area. This is because the area restriction used for GS and TS is the same area of the circuit mapped to standard cells. Using the transistor sizing is possible to reduce the delay in 40.4%, on average, compared to the delay reduction achieved by gate sizing that is 21%, over a standard cell library. Area and power are kept almost the same as in the gate sizing.

Tab. 1 – Comparison results between standard cells (SC) sizing and sizing using Geometric Programming (GP) proposed in this work to 45nm minimizing delay subject to area

	Power (µW)			**Timing (ρs)**			**Area (µm^2)**		
	SC sizing	**GP sizing**	**R (%)**	**SC sizing**	**GP sizing**	**R (%)**	**SC sizing**	**GP sizing**	**R (%)**
C432	22.2	22.4	-0.9	718	666	7.3	210.4	210.4	0.0
C499	58.3	58.4	-0.2	750	651	13.1	536.4	536.4	0.0
C1908	33.6	33.7	-0.3	472	425	10.0	304.3	304.3	0.0
C880	31.4	31.1	1.1	451	330	26.8	281.0	277.4	1.3
apex1	239.8	239.5	0.1	673	504	25.2	2304	2296	0.4
apex2	527.1	523.6	0.7	863	650	24.7	5180	5145	0.7
apex3	254.3	251.9	0.9	687	507	26.3	2441	2413	1.2
apex5	264.6	258.3	2.4	662	431	34.9	2512	2446	2.6
Avg.	178.9	177.3	**0.5**	660	521	**21.0**	1721	1704	**0.8**

Total number of variables used to solve the Gate Sizing (GS) and Transistor Sizing (TS) problem and the difference (Diff.), in percentage, between these numbers is presented in Tab. 3. Tab. 3 also shows for each circuit the number of gates, the number of transistors and the number of auxiliary variables. Auxiliary variables are used to translate the generalized geometric problem to the standard form. The number of auxiliary variables used in the design of both, gate sizing and transistor sizing, is the same considering the same circuit. Thus, the total number of variables for the gate sizing (GS) is the sum of the number of gates and the number of auxiliary variables. The total number of variables for the transistor sizing (TS) is given by the sum of the number of transistors in the circuit and the number of auxiliary variables.

Tab. 2 – Comparison results between gate sizing (GS) and transistor sizing (TS) proposed in this work to 45nm minimizing delay subject to area

	Power (µW)			**Timing (ρs)**			**Area (µm^2)**		
	GS	**TS**	**R (%)**	**GS**	**TS**	**R (%)**	**GS**	**TS**	**R (%)**
C432	22.4	21.8	**2.7**	666	401	**39.9**	210.4	210.4	**0.0**
C499	58.4	56.2	**3.8**	651	422	**35.3**	536.4	536.4	**0.0**
C1908	33.7	32.3	**4.7**	425	253	**40.4**	304.3	304.3	**0.0**
C880	31.1	30.2	**3.9**	330	188	**43.0**	277.4	277.4	**-1.3**
apex1	239.5	231.3	**4.9**	504	294	**41.6**	2296	2296	**-0.4**
apex3	251.9	245.1	**4.8**	507	294	**42.0**	2413	2441	**-1.2**
apex5	258.3	255.7	**1.0**	431	256	**40.6**	2446	2512	**-2.7**
Avg.	127.9	124.7	**2.9**	502	301	**40.4**	1212	1704	**-0.8**

Tab. 3 – Total number of variables used to solve the gate sizing (GS) and transistor sizing (TS) problem and the difference (diff.) Between these numbers

	# Gates	**# Transistors**	**# Aux. Variables**	**Total # of Variables**		
				GS	**TS**	**Diff. (%)**
C432	184	666	344	528	1010	91.3
C499	403	1608	755	1158	2363	104.0
C1908	259	1008	455	714	1463	104.9
C880	232	900	399	631	1299	105.9
apex1	1728	6842	3351	5079	10193	100.7
apex3	1939	7476	3771	5710	11247	97.0
apex5	1942	8244	3663	5605	11907	112.4
Avg.	**955**	**3821**	**1820**	**2775**	**5640**	**102.3**

The only drawback in Transistor Sizing is the number of variables, which is more than double than in gate sizing, 102.3%, as shown in Table III. This produces an execution time considerably higher, because geometric program solvers scale cubically [13].

6. Conclusions and Future Works

Gate sizing and transistor sizing using GP achieves better results compared with a circuit using commercial standard cell sizes selected by RTL Compiler from Cadence.

Tests were made considering 45nm technology, where gate sizing reduced the delay in 21%, on average, for same area and power values of the sizing provided using standard-cells library. Transistor sizing reduced 40.4% in delay, on average, compared with the results from gate sizing. The only drawback using transistor sizing is the number of variables, which is more than double compared to the gate sizing.

Using an automatic cell generation tool we can generate cells with the desired size and take advantage of the better results in timing, area and power, which is critical in recent technologies. As a future work, we intend to use transistor sizing only for the cells of critical paths and use gate sizing for the other cells.

7. Acknowledgment

This work is partially supported by Brazilian National Council for Scientific and Technological Development (CNPq–Brazil) and Coordination for the Improvement of Higher Education Personnel (CAPES).

8. References

[1] R. J. Duffin, E. L. Peterson, and C. Zener, “Geometric programming- theory and application,” *John Wiley & Sons*, 1967.

[2] W. Elmore, “The transient analysis of damped linear networks with particular regard to wideband amplifiers,” *J. Applied Physics*, vol. 19, 1948.

[3] S.Boyd, S. J. Kim, D. Patil, and M. Horowitz, “Digital circuit optimization via geometric programming,” *Operations Research*, vol. 53, no. 6, pp. 899–932, Nov.-Dec. 2005.

[4] S. Boyd, S.-J. Kim, L. Vandenberghe, and A. Hassibi, “A tutorial on geometric programming,” *Springer Science+Business Media*, pp. 67– 127, 2007.

[5] J. P. Fishburn and A. E. Dunlop, “Tilos: A posynomial programming approach to transistor sizing,” in *Int. Conference on Computer Aided Design*, Las Vegas, Nevada - USA, 1985, pp. pp. 326–328.

[6] S. Sapatnekar, V. Rao, P. Vaidya, and S.-M. Kang, “An exact solution to the transistor sizing problem for cmos circuits using convex opti- mization,” *IEEE Transactions on Computer Aided Design of Integrated circuits and Systems*, vol. 12, no. 11, pp. 1621–1634, 1993.

[7] I. Sutherland, B. Sproull, and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999.

[8] M. Berkelaar and J. Jess, “Gate sizing in mos digital circuits with linear programming,” in *EDAC‘90: Conference on European Design Automation*, Glasgow, Scotland, 1990, pp. 217–221.

[9] K. Bhattacharya and N. Ranganathan, “A linear programming formulation for security-aware gate sizing,” in *18th ACM Great Lakes symposium on VLSI, Orlando*, Florida - USA, 2008, pp. 273–278.

[10] S. S. Sapatnekar and W. Chuang, “Power-delay optimization in gate sizing,” *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 5, no. 1, pp. 98–114, 2000.

[11] V. Mahalingam and N. Ranganathan, “A nonlinear programming based power optimization methodology for gate sizing and voltage selection,” in *ISVLSI 2005: IEEE Computer Society Annual Symposium on VLSI*, Tampa, Florida - USA, 2005, pp. 180–185.

[12] Cadence, “RTL compiler,” 2009, available in: http://www.cadence.com.

[13] M. K. K. Kasamsetty and S. S. Sapatnekar, “A new class of convex functions for delay modeling and their application to the transistor sizing problem,” IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, vol. 19, no. 7, pp. 779–788, 2000.