ISCAS 2000 - IEEE International Symposium on Circuits and Systems, May 28-31, 2000, Geneva, Switzerland

AUTOMATED TRANSISTOR SIZING ALGORITHM FOR MINIMIZING SPURIOUS SWITCHING ACTIVITIES IN CMOS CIRCUITS

Artur Wróblewski, Christian V. Schimpfle and Josef A. Nossek

Munich University of Technology
Arcisstr. 21, 80333 Munich, Germany
e–mail: arwr@nws.e-technik.tu-muenchen.de

ABSTRACT

In this paper a new approach for minimizing glitches in the combinational parts of static CMOS circuits is presented. Delay balancing is applied in order to guarantee synchronously arriving signal slopes at the inputs of the logic gates. Thus, glitching can be avoided. The delay of a logic gate depends directly on the transistor sizes, i.e. the channel-widths and -lengths (W and L). Specific variation of the transistor sizes allows to equalize different path delays without influencing the total propagation delay of the circuit. Besides the delay, the total capacitance and the short-circuit power consumption of a circuit also depend on the transistor sizes. In order to take this fact into account when sizing transistors for delay balancing, the method is formulated as a multiobjective optimization problem, where the path delay differences and the power consumption are the design objectives. A program **GliMATS** for automated circuit optimization has been implemented. Experimental results show that significant power savings can be achieved with this method.

1. INTRODUCTION

Optimal sizing of MOS-transistors is a widely investigated method for the design of CMOS-circuits with restricted area, propagation delay or power consumption. A large number of the previously published approaches aim at area and power optimization under given delay constraints [1, 2, 4]. Optimal area utilization is still important but less critical since deep submicron techniques offer increasingly high integration densities. The power consumption models often include only the power consumed for charging transistor gate- and drain/source-capacitances. The power models in [1] include also the dissipation caused by short-circuit currents that occur during transition when both P- and N-transistors of a CMOS-stage are conducting. Delay balancing for reducing the glitching activities in combinational Wallace-trees and array multipliers has been introduced in [5].

Based on the approach presented in [6] a similiar concept has been developed. It includes automated solving of the delay balancing problem within a transistor sizing algorithm. Unlike for most methods that focus on maximizing the speed of a circuit, the method here allows also the transistor lengths to be variable. Reducing the speed for delay balancing is only allowed for parts of the circuit that are not in the critical path. In [7] a method is presented, where all transistor widths outside the critical path are reduced in order to reduce the total capacitance of the circuit. However, delay balancing may not be possible if only the widths are variable because the limit here is the minimum feature size. Further speed reduction can then be achieved by increasing the transistor length. In order to keep track of the contrary design objectives like enlarging transistor sizes for delay balancing, and at the same time reducing the total power consumption caused by charging capacitances, the method is formulated as a multiobjective optimization problem.

2. DELAY AND POWER MODELS

The delay and power models used for the transistor sizing method presented here are defined at gate level. When modeling a circuit at gate level (*macromodeling*), the relatively large number of local parameters that describe every single transistor is reduced to a set of scale factors for each gate. This affords acceptable computation time for optimization of larger circuits. In the considered case the number of variables is reduced to one specific W and one specific L for each gate. If W and/or L are varied, all transistor widths and/or lengths within the gate are scaled by the same factor simultaneously.

2.1. Delay Model

The macromodel delay has to be described for each type of logic gate separately. In the following, the delay model is derived exemplarily for a 2-input CMOS AND gate. The generalization to any other logic gate type is straightforward. The delay of a gate at position m can be split up into two parts [3, 4]: The step response delay $\tau_{s,m}$, which is independent from the input signal form, and $\tau_{in,m}$, which is the contribution caused by the finite input signal rise and fall times. The total delay τ_m is then approximated by

$$\tau_m = \tau_{in,m} + \tau_{s,m} \quad . \tag{1}$$

The optimization method aims at the minimization of glitches, which necessitates equalizing all path delays. However, the step response delay $\tau_{s,m}$ depends on the input transition. For example: The step response delay of a 2-input AND gate in $0.25\mu m$ technology with a certain load is $0.4ns$ for the input transition $11 \rightarrow 00$ and $0.75ns$ for $00 \rightarrow 11$. Therefore, the different paths can exactly be balanced for one special transition only. Experiments have shown that the worst case delay is a good choice and is easy to formulate in the model. Furthermore, numerous simulations based on this model show, that although the paths can-not be exactly balanced for all transitions, glitching can be eliminated. The step response delay $\tau_{s,m}$ is described with Elmore's delay model. It is assumed that there is a fixed ratio ρ between N- and PMOS transistor sizes ($(\frac{W_m}{L_m})_p = \rho(\frac{W_m}{L_m})_n$) to achieve $R_{n,m} = R_{p,m} = R_m$ for the channel resistances of N- and PMOS transistors. The drain/source and gate capacitances can then be written as $C_{d/s_n,m} = C_{d/s,m}$, $C_{d/s_p,m} = \rho C_{d/s,m}$ and $C_{g_n,m} = C_{g,m}$, $C_{g_p,m} = \rho C_{g,m}$. All components can be formulated as functions of the channel length and width: $R_m = r\frac{L_m}{W_m}$, $C_{d/s,m} = c_{d/s}W_m$ and $C_{g,m} = c_g W_m L_m$. The load capacitance depends on the transistor sizes at position $m+1$ and the number of input transistor gates κ_{m+1} at position $m+1$, in such a way that $C_{L,m} = \kappa_{m+1}c_g W_{m+1}L_{m+1}$. For the 2-input AND gate follows

$$\begin{aligned}\tau_{s,m} &= rc_{d/s}(5+3\rho)L_m + 2rc_g(1+\rho)L_m^2 + \\ &\quad rc_g(1+\rho)\frac{L_m}{W_m}W_{m+1}L_{m+1} \qquad (2)\end{aligned}$$

corresponding to Fig. 1.

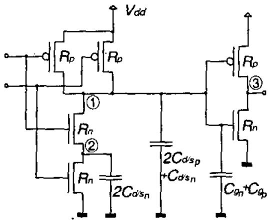

Figure 1: AND gate with parasitics for delay modeling.

According to [4] the input dependent delay $\tau_{m,in}$ is given by:

$$\tau_{m,in} = \frac{1}{6}\tau_{r/f,m-1}(1 + \frac{2V_t}{V_{dd}}) \qquad (3)$$

where $\tau_{r/f,m-1}$ is the rise/fall time of the input signal, V_t is the threshold voltage and V_{dd} the power supply voltage. If a chain of gates is considered, $\tau_{r/f,m-1}$ is the rise/fall time of the output signal of the previous gate at position $m-1$. In good approximation these signal ramps can be considered to be quasi-linear, such that $\tau_{r/f,m-1}$ in (3) can be replaced by its linear approximation $\tau_{eff,m-1}$ which is given by:

$$\tau_{eff,m-1} = \frac{8}{3\ln 3}\tau_{s,m-1}(1 - 0.27\frac{V_t}{V_{dd}}) \quad . \qquad (4)$$

For details see [4]. The input dependent delay can then be written as:

$$\begin{aligned}\tau_{m,in} &= \frac{4}{9\ln 3}(1 + 1.73\frac{V_t}{V_{dd}} - 0.54(\frac{V_t}{V_{dd}})^2)\cdot\tau_{s,m-1} \\ &= K\cdot\tau_{s,m-1} \qquad (5)\end{aligned}$$

where all technology dependent parameters are merged in the constant K. With equations (1), (2) and (5) the total gate delay is given by:

$$\begin{aligned}\tau_m &= K\cdot(rc_{d/s}(5+3\rho)L_{m-1} + 2rc_g(1+\rho)L_{m-1}^2 + \\ &\quad rc_g(1+\rho)\frac{L_{m-1}}{W_{m-1}}W_mL_m) + rc_{d/s}(5+3\rho)L_m + \\ &\quad 2rc_g(1+\rho)L_m^2 + rc_g(1+\rho)\frac{L_m}{W_m}W_{m+1}L_{m+1} \\ &= k_1L_{m-1} + k_2L_{m-1}^2 + k_3\frac{L_{m-1}}{W_{m-1}}L_mW_m + \\ &\quad k_4L_m + k_5L_m^2 + k_6\frac{L_m}{W_m}L_{m+1}W_{m+1}. \qquad (6)\end{aligned}$$

All technology dependent parameters are merged in constants $k_{1\ldots6}$. The total delay of a path number ν is the sum over all gate delays in this path:

$$\tau_\nu = \sum_{m=1}^{n}\tau_m \quad , \qquad (7)$$

where n is the number of gates in the path.

For verification of the delay model the delay through a chain of two AND gates loaded with an inverter is considered. The circuit is shown in the lower right corner of Fig. 2. The diagram

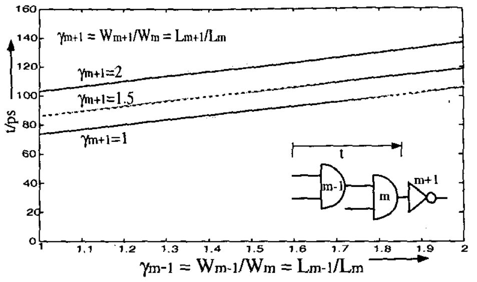

Figure 2: SPICE-simulated (dotted line) and calculated (solid line) delays t for different ratios $\gamma_{m-1} = \frac{W_{m-1}}{W_m} = \frac{L_{m-1}}{L_m}$ and $\gamma_{m+1} = \frac{W_{m+1}}{W_m} = \frac{L_{m+1}}{L_m}$.

in Fig. 2 shows the SPICE-simulated (dotted line) and calculated (solid line) delays from the primary inputs to the output of gate m for different loads (corresponding to W_{m+1} and L_{m+1}) and different W_{m-1} and L_{m-1} (W_m and L_m is held constant). The calculation with the proposed model shows good correspondence with the simulations.

2.2. Power Consumption Model

With the objective function (7) only the delay can be considered in an optimization procedure so far. In order to take into account also the transistor size dependency of the short-circuit currents and the total capacitance of a circuit, an objective function for the power consumption of gate number m can be formulated as follows:

$$P_m = P_{m,cap} + P_{m,sc} \quad , \qquad (8)$$

where $P_{m,cap}$ denotes the power consumed for charging the gate and drain/source capacitances and $P_{m,sc}$ denotes the short-circuit power consumption of gate m. For a 2-input AND, $P_{m,cap}$ can be written as:

$$P_{m,cap} = c_1L_m + c_2W_mL_m + c_3W_{m+1}L_{m+1} \quad , \qquad (9)$$

where $c_1 = fV_{dd}^2(\alpha_1c_{d/s}(1+2\rho)+2\alpha_2c_{d/s})$, $c_2 = fV_{dd}^2\alpha_1c_g(1+\rho)$ and $c_3 = \alpha_3fV_{dd}^2(c_{d/s}(1+\rho) + c_g\kappa_{m+1}(1+\rho))$. Factors α_1, α_2, and α_3, denote the switching activities at nodes 1, 2, and 3 respectively (see the numbers in circles in Fig. 1), which are considered as constants over W and L.

The short-circuit power dissipation of a CMOS inverter according to [8] is given by:

$$P_{m,sc} = \tau_{eff,m-1}\frac{\beta_m}{12} f(V_{dd} - 2V_t)^3. \qquad (10)$$

β_m is the transistor gain factor ($\beta_m \sim \frac{W_m}{L_m}$) and $\tau_{eff,m-1}$ is given in (4). For a $0.25\mu m$ technology and $V_{dd} = 2.5V$ $P_{m,sc} \approx 0.00045 f \frac{W_m}{L_m}\tau_{s,m-1}$. It can be shown experimentally, that neglecting the contribution of $P_{m,sc}$ has no negative influence on the results of the path balancing method even for complex gates. Therefore, it is reasonable to set $P_m = P_{m,cap}$. The total transistor size dependent power consumption in path number ν can be formulated as:

$$P_\nu = \sum_{m=1}^{n} P_m \quad , \qquad (11)$$

for a path with n gates.

3. MULTIOBJECTIVE OPTIMIZATION

In order to find a power optimal solution for W and L the designer is confronted with two conflicting design criteria: path balancing by transistor sizing, achieved by enlarging transistors, and low power consumption for charging capacitances which requires small transistors at the same time. In order to equalize all the path delays with respect to the critical path, every path requires individual optimization. Let τ_{crit} denote the critical path delay of the circuit. For every path ν a solution of

$$\min_{W,L} |\tau_\nu - \tau_{crit}| \qquad (12)$$

must be calculated to achieve path balancing. The path delay τ_ν is defined in (7). The power consumption according to (11) is minimized by

$$\min_{W,L}(P_\nu = \sum_{m=1}^{n} P_m) \quad . \qquad (13)$$

Equations (12) and (13) describe convex optimization problems in W and L. The multiobjective optimization problem is given by:

$$\min_{W,L}(S_\nu = w \cdot (\tau_\nu - \tau_{crit})^2 + (1-w)\cdot P_\nu) \quad . \qquad (14)$$

The weight factor w varies between 0 and 1, $w \in [0, 1]$. In order to avoid the case differentiation at the discontinuity $\tau_\nu = \tau_{crit}$, $|\tau_\nu - \tau_{crit}|$ is replaced by its square.

The upper and lower bounds of the transistor sizes are determined by the minimum feature size of the used technology and the user defined limits for the maximum available area for a single transistor. These additional constraints have to be considered separately.

Assigning a value to w allows a solution to be chosen depending on which of the design objectives is more desired: low power consumption caused by the total capacitive load or balanced path delays. However, experiments have shown that for many circuits the best low power solution is obtained if $|\tau - \tau_{crit}| = 0$, i.e. for optimally balanced paths. This is usually given when $w = 0.5...1$.

4. BALANCING OF THE PATH DELAYS

The transistor sizing algorithm for the reduction of glitching activity is implemented in the program **GliMATS**(**Gli**tch **M**inimization by **A**utomated **T**ransistor **S**izing). It allows to optimize the circuits automatically. Inputs to **GliMATS** are the netlist of the circuit, the user defined value of the weight factor w and a library, including the functions τ_m and P_m for every gate type as shown in (6) and (9) for the 2-input AND gate. As the output **GliMATS** produces netlist of the glitch minimized circuit. It is assumed that the circuit is already optimized to match eventually given timing constraints, so the critical path must not be manipulated in order to retain the required maximum delay. The input netlist to the path balancing program is given from this previous speed optimization.

GliMATS starts at the primary inputs with building a node list which describes the dependencies of all nodes from their predecessors. The delay of every passed node and its predecessors is saved. After all output nodes are reached, the critical path delay is known. Then the algorithm starts at the output nodes and traces back to the primary inputs by choosing at each passed node the predecessor with the largest delay for the next node. The delay and power functions of this path are built according to equations (7) and (11). The actual delay $\tau_{\nu,0}$ of the path is calculated. If $\tau_{\nu,0} \neq \tau_{crit}$, the path is optimized according to (14). For this purpose a MATLAB optimization routine is started. Once a path ν is optimized or if $\tau_{\nu,0} = \tau_{crit}$, all gates in this path are "marked". In all further optimization steps for the yet non-balanced paths, the transistor sizes in "marked" gates are considered to be fixed and are not affected by the sizing algorithm. This complete procedure is repeated until all gates in the circuit are "marked". Finally the netlist of the optimized, path delay balanced circuit is obtained. Note that it may not be possible to balance all the paths. For example, if one input of a gate is a primary input and another input of the same gate is the output of some other gate. Then the signal from the primary input can not be further slowed down. Despite this case a circuit can be very well balanced as shown in Section 5.

5. APPLICATIONS AND EXPERIMENTAL RESULTS

The proposed path balancing method has been tested on some circuit examples, a few selected are shown here. 4 × 4, 8 × 8, and 16 × 16 array multipliers have been simulated with PowerMill before and after transistor optimization for glitch reduction. For simulation, 10000 random input vectors have been applied for each circuit. The increase of chip area has been estimated by measuring the area inrease of a single cell due to transistor sizing and projecting this to the total chip area including the wiring. The results are shown in Table 1. Another logic circuit which is shown in Fig.

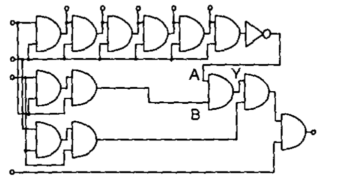

Figure 3: Logic circuit for demonstration of the path balancing method.

3 has been designed to demonstrate the effect of path balancing

by transistor optimization. If zero delay is assumed for all gates in this circuit, the output is always 0 regardless of the inputs. In a simulation with complex transistor models only glitches due to unbalanced paths are visible at the output.

The signal Y of the unbalanced circuit (all transistor sizes minimal) for random input signals is shown on top of Fig. 4. The same

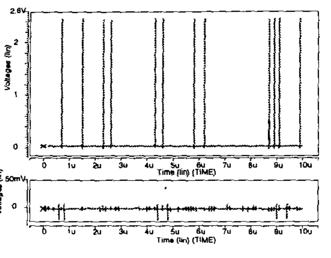

Figure 4: Signal Y of the circuit in Fig. 3 with unbalanced (top) and balanced path delays (bottom).

input signals yield signal Y shown on the bottom of Fig. 4 for the circuit after optimization: glitches are completely eliminated. Note that the voltage axis are scaled differently. A comparison of the power consumption is shown in the first row of Table 1, where the circuit is simply denoted as "Logic". Fig. 5 shows the slopes of the signals A and B before (top) and after (bottom) path balancing. This figure demonstrates how signal B is delayed after transistor

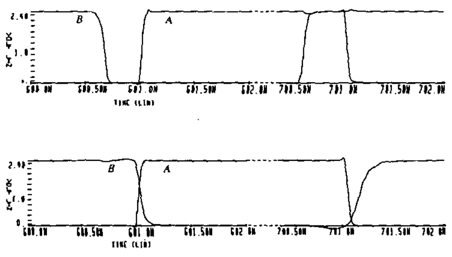

Figure 5: Example for signals A and B of the circuit in Fig. 3 with unbalanced (top) and balanced path delays (bottom).

sizing in order to "wait" on signal A. The simulation results for the multipliers are also shown in Table 1.

Circuit	not balanced	balanced	power savings	area increase
Logic	0.187	0.172	8%	5%
4×4 Mult.	0.914	0.620	32.0%	15%
8×8 Mult.	4.12	2.37	42.4%	19%
16×16 Mult.	19.29	10.49	45.6%	31%

Table 1: Comparison of the power consumption in mW for circuits without and with path balancing by transistor sizing ($0.25\mu m, V_{dd} = 2.5V$, PowerMill simulations with 10000 random input vectors).

Note that the percentage of power reduction due to the glitch elimination increases for larger arrays because of the snowball effect that glitches stimulate in these circuits. The CPU-time for the complete optimization of a 16×16 multiplier is about 7 minutes on a Ultra Sparc 1 workstation.

The results show significant power savings after **GliMATS** has been applied. However, one must be aware that enlarging of the transistor lengths to increase the delay results in slower signal slopes which may lead to larger short circuit power consumption (this is considered in the results of Table 1) and to an increase of the required chip area, as shown in Table 1.

6. CONCLUSION

In this work a method for transistor size optimization to achieve equal path delays in CMOS circuits has been presented. Delay and power consumption of a path can be modeled as functions of the transistor sizes W and L at gate level. With multiobjective optimization the path delay differences and the power consumed for charging capacitances can be minimized simultaneously. The solutions for W and L are restricted by upper and lower bounds, given by the minimum feature size and area limitations. A tool - **GliMATS** - has been implemented that automatically reads the netlist of a circuit, builds the delay and power functions, starts multiobjective optimization and writes the netlist of the optimized, delay balanced circuit with the new values of W and L for each gate. With this method glitching in a circuit can be reduced drastically. Experimental results show significant power savings after optimization.

7. REFERENCES

[1] M. Borah, R. M. Owens, and M. J. Irwin. Transistor Sizing for Low Power CMOS Circuits. *IEEE Trans. on Computer-Aided Design*, 15(6):665–671, June 1996.

[2] J. P. Fishburn and A. E. Dunlop. TILOS: A Posynomial Programming Approach to Transistor Sizing. *Proc. ICCAD*, pages 326–328, 1985.

[3] N. Hedenstierna and K. O. Jeppson. CMOS Circuit Speed and Buffer Optimization. *IEEE Transactions on Computer Aided Design*, CAD-6(2):270–281, March 1987.

[4] B. Hoppe, G. Neuendorf, D. Schmitt-Landsiedel, and W. Specks. Optimization of High-Speed CMOS Logic Circuits with Analytical Models for Signal Delay, Chip Area, and Dynamic Power Dissipation. *IEEE Trans. on Computer-Aided Design*, 9(3):236–247, March 1990.

[5] T. Sakuta, W. Lee, and P. T. Balsara. Delay Balanced Multipliers for Low Power/Low Voltage DSP Core. *IEEE Symp. on Low Power Electronics*, pages 36–37, October 1995.

[6] C.V. Schimpfle, A. Wróblewski, and J. A. Nossek. Transistor Sizing for Switching Activity Reduction in Digital Circuits. *European Conf. on Circuit Theory and Design, ECCTD'99, Stresa, Italy*, 1999.

[7] S. Trimberger. Automated Performance Optimization of Custom Integrated Circuits. *Proc. Int. Symp. on Circuits and Systems*, pages 194–197, 1983.

[8] H. J. M. Veendrick. Short-Circuit Dissipation of Static CMOS Circuitry and Its Impact on the Design of Buffer Circuits. *IEEE Journal of Solid State Circuits*, SC-19(4):194–197, August 1984.