Improving Transistor Sizing for Asynchronous Circuits

Florian Meinel[H,I], Norman Kluge[H,I], and Ralf Wollowski[H]

[H]Hasso-Plattner-Institut, Potsdam, Germany [I]IHP, Frankfurt (Oder), Germany

1 Introduction

The sizes of a logic circuit's transistors influence its operation speed and energy consumption. Since logic signals propagate by gate pins of transistors being charged and discharged, the capacitances of those pins and the resistances of the conducting channels matter. By altering the transistor area, both of these values are changed. Increasing the size of a transistor leads to smaller resistances, allowing it to charge its load faster (See [1] for details). Yet it also leads to larger capacitances, making it pose a larger load for its predecessor. Thus, the sizing task is to balance the relationships between the transistor sizes, optimizing for operation speed or energy consumption. There are already well-established methods of selecting transistor sizes in acyclic paths. This is sufficient for synchronous circuits because there all cycles are gated by clocked latches. Making the slowest acyclic path between two latches faster allows for a higher clock frequency. However, asynchronous circuits have direct feed-back loops in their control paths. This renders the conventional methods unusable. Instead, heuristic methods have to be applied. In [2], Ebergen et. al. proposed a method for transistor sizing in asynchronous circuits. However, it is limited by particular assumptions, which the new method presented in this paper does not make:

- it does not deal with constraints on the circuit's input pins,
- can not handle cells with multiple CMOS stages (such as AND, which consists a NAND and an INV stage),
- is not designed for use with discrete cell sizes of standard cell libraries,
- allows only solutions where all gates have an equal delay.

This paper presents a method for transistor sizing for asynchronous circuits based on a standard optimization algorithm, Simulated Annealing [3].

2 Method Walkthrough

Standard cell libraries come with precise timing information for their individual cells, created from experiments or SPICE simulations [4]. Using information from such data sheets (often provided in the Liberty file format [5]), heuristic models of the overall operation speed of a circuit can be constructed. The actual run time of a circuit generally varies with its input data, with some cells switching more often than others. However, a simplified metric $\mathscr{D} = \sum d_i$ that simply sums the switching delays of all cells can be useful. Optimizing this metric poses a useful heuristic for optimizing the real operation speed. This is what our proposed method does.

Algorithm Figure 1 shows the rough process of the algorithm. It starts out with an initial configuration where all cells in the circuit have the default size from the standard cell library. It then calculates a cost function c for this configuration, for example, $c = \mathscr{D}$. The individual cell delays are looked up in the cell library data sheet. They depend on (1) the selected size of the cell itself and (2) its load capacitance. The load capacitance is the sum of its successor's input capacitances, as well as static loads such as circuit output pins.

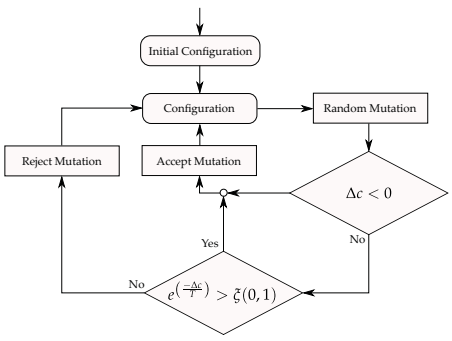

Fig. 1: The Simulated-Annealing-based algorithm accepts random mutations with a probability depending on a cost function c

In the next step, a random mutation is applied to the configuration. For example, a randomly chosen cell could randomly be scaled either up or down to the next available size from the cell library. The cost function is then calculated again, and compared with the value before the change. If the cost gets lower, the change gets accepted. The mutated configuration is now the starting point for the next iteration.

However, if the cost gets higher in a mutation, there is still a probability of the change being accepted, depending on the amount of cost increase. This prevents the optimization to get stuck in local minima. A variable T (temperature) is defined to steadily decrease over the course of the algorithms iterations (multiplied by α each time, with $0 < \alpha < 1$), influencing this probability. It is designed so that in the beginning almost any change is accepted, whereas towards the end the optimizer gets more strict and turns into a greedy algorithm.
The final configuration of the circuit has a minimized $\mathscr{D}$ given the standard cell library and some constraints: (1) output pins of the circuits are assumed to have a static load capacitance, representing the inputs of succeeding components, or wire loads. (2) The cells that are connected to an input pin of the circuit cannot be resized arbitrarily. This is to avoid passing all driving responsibility to the circuit's predecessors.

Optimizing Energy Consumption Another benefit of the method is the adjustability of the cost function. For example, if the cell library data sheet also supplies data on the cells' power consumption, a second metric $\mathscr{E} = \sum e$ can be introduced, summing the energy consumed by all cells if they each switch once. These values also depend on the cell sizes and their individual delays. The adjusted cost function $c = w_E \cdot \mathscr{E} + (1 - w_E) \cdot \mathscr{D}$ can be weighted with $0 \leq w_E \leq 1$ to emphasize energy consumption or speed in the optimization.

Algorithm Parameters Appropriate values for the number of iterations i, the initial temperature T_0 and the decay factor α need to be selected. More iterations generally yield better results, but in our experiments, after increasing them above 1000 per circuit cell, no significant further improvement was achieved. Given i, the other parameters are calculated from the desired probabilities. In the beginning, almost all changes should be accepted ($p_0 = 95\,\%$). Then, towards the end the probability for accepting negative changes should become very small (e.g. $p_g = 5\,\%$ after 70 % of the iterations). To calculate the parameters that achieve this, a calibration step determines the average cost change Δc_e. The values are then $T_0 = \frac{-\Delta c_e}{\ln p_0}$ and $\alpha = \left(\frac{-\Delta c_e}{T_0 \cdot \ln p_g}\right)^{\frac{1}{70\,\% \cdot i}}$.

3 Experiments

To test this approach, we implemented it in a tool called *ASGdrivestrenght*[1]. We conducted a number of experiments using simulators to estimate the runtime and energy consumption of circuits before and after the optimization step. The tests were conducted with a 130 nm standard cell library with 80 cell types available in a range of sizes each. We tested six different asynchronous circuits with cell counts ranging from 57 to 362. We assumed different values for the static output pin loads, here included are the results for 0.012 pF (4 times inverter input capacitance). The results shown are for a circuit performing the task of multiplying two 8-bit numbers (362 cells, 21 inputs, 21 outputs). Figure 2 shows the locally estimated metrics $\mathscr{D}$ and $\mathscr{E}$ alongside actual simulator runs with circuit testbench data. This way we can evaluate both the algorithm and the selected heuristics.

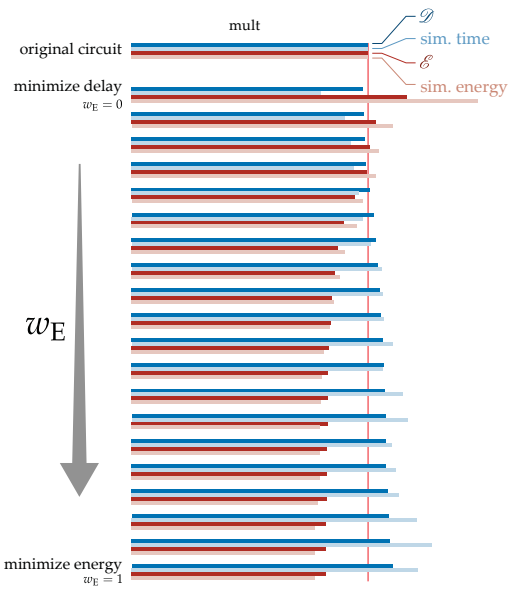

Fig. 2: Simulation results for multiplication circuit before and after optimization, with different paramters

Representative for all conducted tests, Figure 2 shows that all of the metrics can be improved in comparison to the initial circuit (where all cells have the *standard size*, which the cell library calls 1x). It shows that an improvement in $\mathscr{D}$ and $\mathscr{E}$ correlates with improvements in actual run time and energy consumption respectively. When changing the weight of the energy optimization w_E, a gradient becomes visible in the results. The extreme values 0 and 1 create the biggest improvements for their respective variable, yet there are also values for which both operation run time and energy consumption are lower than in the original *at the same time*.

4 Conclusion and Future Work

A new method has been presented for selecting transistor sizes to optimize operation speed and energy consumption of asynchronous circuits. It has been shown that it comprises several benefits over previous methods, specifically the handling of multi-stage CMOS cells and generally the usage of discrete values from a standard cell library, making it ready for CAD tools. Experiments have been carried out that show that the method can reduce both run-time and energy consumption, balancing between the two, but also improving both at the same time.
However, there are multiple points of interest to explore further: One is to refine the algorithm's cost function. Right now, the optimized metrics use only the capacitances of the cells' successors as loads, as well as output pins. In reality, also wires between cells contribute to the capacitances and thus the delays. Hence, it would be interesting to use information from a layouting step. This would make the procedure iterative since the transistor sizes *influence* the layouting step. Currently, the cost function does not consider critical paths. This factor could close the gap between estimated and real delay and energy consumption resp.
Another area of future work is the automated suggestion for a value of w_E. A designer might not want to specify exactly how important the different metrics are. Possibly algorithms can determine how much benefit in one metric would be gained by some sacrifices in the other.
Furthermore, more detailed tests should be conducted, especially comparing the performance of this method to the optimization methods (for combinatorial logic) in the synchronous world.

5 Acknowledgements

We thank Milos Krstic, Steffen Zeidler and Anselm Breitenreiter from IHP GmbH (Frankfurt/Oder) for their helpful advice.

References

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

[2] J. C. Ebergen, J. Gainsley, and P. Cunningham, "Transistor sizing: How to control the speed and energy consumption of a circuit," in *10th International Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC 2004), 19-23 April 2004, Crete, Greece*. IEEE Computer Society, 2004, pp. 51–61. [Online]. Available: http://dx.doi.org/10.1109/ASYNC.2004.1299287

[3] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983. [Online]. Available: http://www.jstor.org/stable/1690046

[4] L. W. Nagel and D. Pederson, "Spice (simulation program with integrated circuit emphasis)," EECS Department, University of California, Berkeley, Tech. Rep. UCB/ERL M382, Apr 1973. [Online]. Available: http://www2.eecs.berkeley.edu/Pubs/TechRpts/1973/22871.html

[5] IEEE Industry Standards and Technology Organization, "Liberty Technical Advisory Board," http://ieee-isto.org/member_programs/liberty-technical-advisory-board/, accessed: 2017-04-24.

Florian Meinel
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2–3
D-14482 Potsdam, Germany
IHP
Im Technologiepark 25
D-15236 Frankfurt (Oder), Germany
florian.meinel@hpi-alumni.de

Norman Kluge
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2–3
D-14482 Potsdam, Germany
IHP
Im Technologiepark 25
D-15236 Frankfurt (Oder), Germany
norman.kluge@hpi.de

Ralf Wollowski
Hasso-Plattner-Institut
Prof.-Dr.-Helmert-Str. 2–3
D-14482 Potsdam, Germany
ralf.wollowski@hpi.de

Digital Engineering · Universität Potsdam