Optimizing Digital Circuits Considering Internal Voltage Drops

Ishwar Suriyaprakash, Homestead High School
Research Advisor: Prof. Vidya Chhabria, Arizona State University

Motivation

Compact electronic Integrated Circuits (ICs or chips) underpin significant societal advancements through products such as life-saving implantable medical devices, cell phones that allow ubiquitous communication, and data centers that power global e-commerce. This revolution has been enabled by manufacturing technology that allows relentless miniaturization of these chips while significantly increasing their functionality and performance. However, this trend has led to undesirable consequences in chips, such as increased power densities (i.e., the heat/power dissipated per square inch) [1] and resulting performance issues and erroneous operation.

The chip design process involves complex optimization steps for achieving the desired functionality and performance while reducing its power impact. Because chips today contain billions of interconnected transistor switches, this is impossible to perform manually, so Electronic Design Automation (EDA) software tools are used to design chips efficiently. However, current EDA tools have drawbacks. High computation times can result in an undesirable increase in design turnaround time, delaying product introduction to the market. Heuristic approaches used in these tools can result in suboptimal circuits, and oversight of certain physical phenomena can result in circuits being error-prone on silicon. This work focuses on algorithms and machine learning methods to address problems in chip design that deal with the effects of the power delivery network on circuit performance.

Background

Modern chips implement complex functions as finite-state machines using a synchronous sequential circuit. As shown in Figure 1, a sequential circuit consists of a combinational circuit and memory elements. The combinational circuit consists of a directed acyclic graph of interconnected cells, as shown in the shaded area of Figure 1. Each cell implements a simpler Boolean function using transistors as switches. The memory elements are implemented using flip-flops (e.g., FF1 in Figure 1), which are timed by a special square-wave signal called the clock signal. At a specified edge of the clock signal (say, the rising edge), the flip-flops transfer the Boolean values at their inputs to their outputs. These signals then flow through the combinational circuit. The period of the clock signal is set using an estimate of the maximum time taken, called the circuit propagation delay, for changes at all the inputs of the combinational circuit to flow to its outputs.

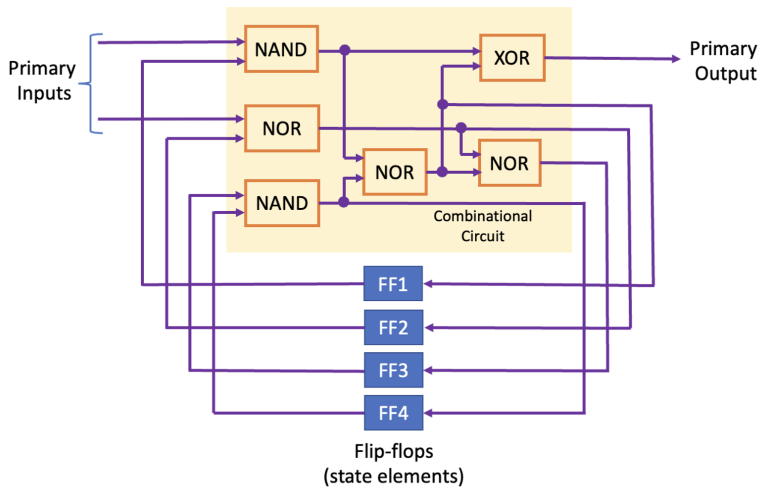

Figure 1. Synchronous sequential circuit.

During the IC design process, the delay of a circuit is calculated using a Static Timing Analysis (STA) tool. Given the pre-computed worst-case input-to-output delays of the component cells of a combinational circuit, STA computes the worst-case circuit propagation delay through this circuit. Cells and flip-flops that form part of this integrated circuit receive power from an external power supply through a tree of interconnects called the Power Delivery Network (PDN) (not shown in Figure 1). Due to the resistance of the wires in the PDN, the voltage at each cell's power input can be lower than that of the power supply, which can result in an increase in the cell's delay. In current practice, STA does not effectively consider these voltage-drop-induced cell-delay increases at each cell. Either the STA is performed ignoring these cell-delay increases altogether, resulting in underestimating the circuit delay (optimistic), or STA is performed assuming the worst-case voltage drop at every cell resulting in overestimating the circuit delay (pessimistic). Designing using the optimistic case can result in erroneous circuit operation, while using the pessimistic case can result in suboptimal circuit operation below the achievable speed.

Objectives

Objective (a): Determining the impact of PDN-induced voltage drops: My first objective was to enhance the IC design methodology by determining a circuit's delay in the presence of PDN-induced internal voltage drops by developing a voltage-aware static timing analysis (VA-STA) tool. Having completed this goal, I have the following two objectives leveraging this VA-STA methodology.

Objective (b): PDN optimization: If the early stages of the IC design process produce a circuit with excessive timing slack because of assuming worst-case internal voltage drops, the goal is to exploit choices in the power delivery network to reduce the overall design turnaround time of such a circuit.

Objective (c): Circuit optimization: On the other hand, if a circuit had been designed with insufficient timing slack because of ignoring internal voltage drops, the goal is to reduce the delay of such a circuit by replacing cells on timing critical paths with functionally equivalent cells of lower delay.

The publicly available OpenROAD EDA tool flow [2], an open-source automated flow that aims to eliminate the need for human intervention in chip design, is used for experiments in this project. Algorithms that are developed are used in conjunction with the OpenROAD tools.

Research Status

Objective (a)

For this objective, I developed a traditional STA tool and subsequently enhanced it to create a voltage-aware STA (VA-STA) tool in Python and integrated both into the open-source OpenROAD design flow as described below.

STA development: In STA, the worst circuit delay is found by first modeling the combinational circuit as a directed acyclic graph (DAG) with component cells as vertices. The STA tool then performs topological sorting of the vertices in the graph, and subsequently computes the input-to-output delays of each cell instance through a forward traversal of this sorted graph. During the forward traversal, each cell instance is annotated with delays calculated using data from cell characterization files for each cell type in the standard cell library. Each pin-to-pin delay and output slew for a cell instance is computed by first selecting the appropriate timing characterization table for that cell type in the standard cell library, and then interpolating to determine the actual delay and output slope corresponding to the input slope and output load for that cell instance. The worst-case timing critical path consists of the sequence of connected cells in the circuit with the largest cumulative delay. During a subsequent reverse traversal of the graph, the timing slacks, or margins, available at each cell output for a specified clock period are calculated.

VA-STA development: I then enhanced my STA tool to create the VA-STA tool. This tool incorporates the voltage drop at each cell instance from Prof. Chhabria's PDNSim tool [3] and calculates the corresponding cell delay increase for that modified cell voltage by interpolating between delay tables for that cell at different supply voltage set points.

Test cases: For use as test cases for my experiments with the OpenROAD flow, I included the ISCAS85 benchmark circuits [4] into the OpenROAD infrastructure after modifying them to sequential circuits by including flip-flops at inputs and outputs as required by OpenROAD tools.

Experiments: The OpenROAD design process of synthesis, floorplanning, placement, and routing was completed for 11 ISCAS85 test case circuits, and STA and VA-STA were performed on the resulting netlists. Results show that the mean worst-case circuit delay increase was 6.14% with VA-STA over baseline STA, and the worst-case timing critical path changed for 4 out of 11 test cases with VA-STA compared to STA. On the largest 4 test cases, the mean worst-case circuit delay increase was 10.7% with VA-STA over baseline STA, and the worst-case timing critical path changed for 2 out of these 4 test cases.

Objective (b) - Ongoing work

If the early stages of the design of a circuit had assumed worst-case voltage drops at each cell instance, the actual voltage-aware delay could be smaller than the estimated delay. The goal is to leverage this extra delay margin and choices in the power delivery network to reduce routing congestion and ensure faster design turnaround time.

The PDN configuration from the power supply decides the number of power delivery paths to each cell instance. These paths dictate the voltage drop at each cell instance and the cell's delay. The goal is to identify an optimal PDN configuration, which allows for efficient routing of all the wires to obtain an IC with a much shorter turnaround time while still satisfying the delay constraint. To determine the least number of PDN wires needed with which the circuit will still satisfy its delay constraint, I developed the following initial approach. First, the power consumption of each cell after its placement is calculated using a TCL program I developed. This information, along with each configuration of PDN wires, is then used to generate the internal voltage drops with Dr. Chhabria's solver, which is then fed to my VA-STA tool to compute voltage-aware circuit delays.

Since the number of PDN configurations can be numerous for large circuits, this iterative approach can be computationally expensive. So, I am developing a convolutional neural network (CNN) that takes heatmaps for power, timing slack, and cell proximities to power supply sources as inputs and outputs the optimal PDN configuration. I developed a Python automation to generate the training data for the CNN. A CNN was chosen as the ML architecture for its localized learning abilities.

Objective (c) - Ongoing work

In a circuit that has been designed ignoring voltage drops due to PDN, the actual voltage-aware delays of each cell instance can be larger than those used for computing the circuit's worst-case delay using traditional STA. To correct such a circuit and avoid erroneous operation, I developed an iterative approach that identifies a cell instance with the minimum slack from the current most timing-critical path in the circuit and replaces it with a cell that performs the same function but has a greater drive strength. A cell with a greater drive strength has a smaller delay, which helps reduce the delay of the circuit. Experiments with this technique are currently in progress.

The iterative replacement strategy discussed above involves replacing only one cell at a time, and may not result in an optimal solution. An alternative approach of replacing different combinations of cells at a time can become computationally expensive due to exponentially many combinations, rendering the approach infeasible. To address this problem, I am developing a deep learning approach that determines the optimal replacement strategy by learning from a limited set of replacements.

References

[1] International Business Strategies, "FinFET and FDSOI: Market and Cost Analysis," 2018.

[2] T. Ajayi, V. A. Chhabria, M. Fogaça, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Lee, U. Mallappa, M. Neseem, G. Pradipta, S. Reda, M. Saligane, S. S. Sapatnekar, C. Sechen, M. Shalan, W. Swartz, L. Wang, Z. Wang, M. Woo, and B. Xu, "Toward an Open- Source Digital Flow: First Learnings from the OpenROAD Project," *Proceedings of Design Automation Conference (DAC)*, 2019.

[3] PDNSim tool, https://openroad.readthedocs.io/en/latest/main/src/psm/README.html

[4] ISCAS85 benchmark circuits, https://github.com/jpsety/verilog_benchmark_circuits