Work-in-Progress: DeVos: A Learning-based Delay Model of Voltage-Scaled Circuits

Dongning Ma
Villanova University
dma2@villanova.edu

Siyu Shen
Boston University
ssy00603@bu.edu

Xun Jiao
Villanova University
dma2@villanova.edu

ABSTRACT

Dynamic voltage and frequency scaling (DVFS) is a typical method to reduce energy consumption of circuits. However, it may cause timing errors if the frequency is not set properly under scaled voltages. To alleviate this issue, this paper proposes **DeVos**, a supervised learning model that can predict dynamic delay of voltage-scaled circuits based on their input workload. We measure the dynamic delay using switching activity generated through gate-level simulation of a post place-and-route design in TSMC 45nm process. We then look for features in the input data that influence dynamic path sensitization. Using these features we apply random forest to construct a predictive model trained and tested using random data. Across a wide range of voltage levels, **DeVos** achieves on average a mean absolute percentage error (MAPE) of less than 5%. To our best knowledge, **DeVos** is the first dynamic delay model of voltage-scaled circuits and can be used to enable accurate dynamic voltage and frequency scaling.

CCS CONCEPTS

• **Hardware → Electronic design automation**; **Robustness**;

KEYWORDS

Voltage Scaling, Circuit Delay

ACM Reference format:
Dongning Ma, Siyu Shen, and Xun Jiao. 2019. Work-in-Progress: DeVos: A Learning-based Delay Model of Voltage-Scaled Circuits. In *Proceedings of International Conference on Hardware/Software Codesign and System Synthesis, New York, NY, USA, October 13–18, 2019 (CODES/ISSS '19),* 2 pages.
https://doi.org/10.1145/3349567.3351725

1 INTRODUCTION

Dynamic voltage and frequency scaling (DVFS) is a typical method to reduce energy consumption of circuits. However, voltage scaling can change the path delay in circuits and hence causing timing errors. Therefore, it is critical to set the proper frequency to protect circuits from timing errors. This requires a knowledge of the dynamic circuit delay under different voltage levels.

Recently, many models are proposed to characterize or predict the delay of functional units (FUs) or instructions [2, 3, 5, 7]. For example, authors in [2, 5, 7] quantify the delay of instructions based on a dynamic timing analysis using gate-level simulation with representative datasets. However, these models assume a worst case path sensitization scenario in predicting the delay, which overlooks the effect of input data on the path sensitization behavior, leading to a less efficient or pessimistic modeling. Jiao et.al classify the delay of functional units into five classes based on different datasets [3] but it cannot predict the delay of circuits under voltage scaling.

To address these problems, we propose a delay model of voltage-scaled circuits called **DeVos**. The key of **DeVos** is using a machine learning-based approach to learn the relationship between dynamic delay and the joint effects of voltage scaling and input data.

Our contributions are as follows:

- We analyze the delay distribution of circuits under a wide range of voltage levels.
- We propose **DeVos**, the first dynamic delay model of voltage-scaled circuits, which can be used to enable accurate dynamic voltage and frequency scaling.

2 PROPOSED APPROACH

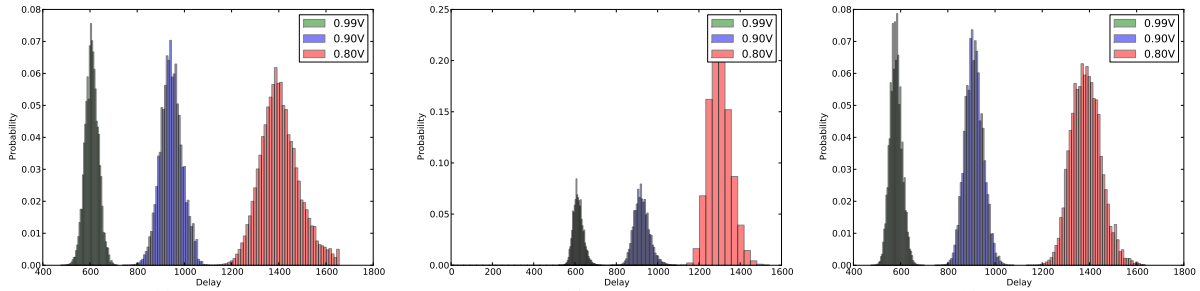
We present the **DeVos** overview in Fig. 3. We start from the RTL-level description implemented in synthesizable VHDL. We perform logic synthesis using Synopsys Design Compiler, place-and-route using Synopsys IC Compiler to generate post-layout netlist in TSMC 45nm technology. Then, we use voltage scaling features of Synopsys PrimeTime to perform static timing analysis and generate corresponding standard delay format (SDF) file for each voltage level. Next, we perform gate-level simulation (GLS) using post-layout netlist, SDF, and input data to generate switching activity file, i.e., value change dump (VCD) file containing the toggling events in the circuit. Using the VCD file, we perform a dynamic timing analysis to extract the dynamic delay of circuits at each clock cycle.

Dynamic Delay Extraction The VCD file records the toggled endpoints at each cycle since we are only interested in the circuit delay. We run the simulation at a relatively slow clock period to make sure there are no timing errors. To measure the dynamic delay at each cycle, we use the last toggle event time of all endpoints to subtract the last positive clock edge arrival time. For example, at cycle N the positive clock edge occurs at time t, and the very last toggled event occurs at time $t\prime$, then the dynamic delay at this cycle is $t\prime - t$.

Feature Extraction We extract the features from the input data and voltage levels. Since the voltage scaling is typically performed at limited discrete levels within a specific range [8], it is possible to train the model using all possible voltage levels. In this paper, we consider 20 voltage levels from 0.80V to 0.99V, with a step size of 0.01V. We extract the input features from input data as well. Because each bit position has an effect on the path sensitization, each feature vector is the vector of bits. Because the dynamic delay is the delay of the longest sensitized path and the path sensitization behavior is determined by both the preceding input (which sets the circuit state) and the current input (which toggles the node value),

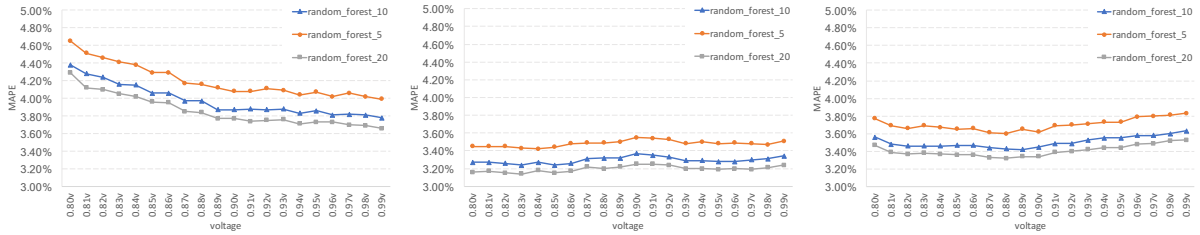
CODES/ISSS '19, October 13–18, 2019, New York, NY, USA

ACM ISBN 978-1-4503-6923-7/19/10…$15.00
https://doi.org/10.1145/3349567.3351725



(a) INT_MUL (b) FP_MUL (c) APP_MUL

Figure 1: Delay distribution of three multipliers



(a) INT_MUL (b) FP_MUL (c) APP_MUL

Figure 2: Prediction accuracy of three multipliers

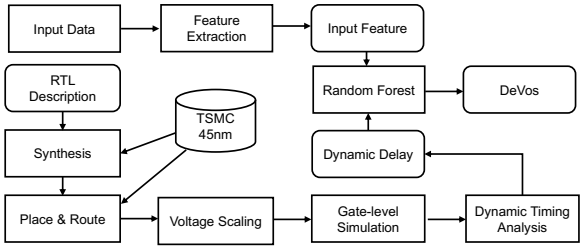


Figure 3: DeVos overview

we set the $\{x[t-1], x[t], V\}$ as our feature, where $x[t]$ and $x[t-1]$ are the input binary vector at cycle t and $t-1$, respectively. We set $D[t]$ as our output label where $D[t]$ is the dynamic delay at cycle t.

Model Training We use random forest tree, an ensemble supervised learning method, as our training method. Each feature element of the feature vector is the value of a single bit position (i.e., 0/1). The machine learning method is adopted from *Scikit-Learn* [4].

3 EXPERIMENTAL RESULTS

We evaluate **DeVos** on three different multipliers: integer multiplier, float point multiplier, and approximate multipliers from lpACLib [6]. We use multipliers because they are the most widely-used arithmetic circuit and contain representative circuit blocks such as adder.

We illustrate the delay distribution of the three multipliers in Fig. 1, which shows Gaussian-like delay distribution. Additionally, a higher voltage indicates shorter delay and a more compact distribution while lower voltages yield longer delay with larger deviation.

We train and test the random forest tree model with 5, 10, and 20 different decision trees to evaluate **DeVos** accuracy on three multipliers under 20 voltage levels. We use MAPE (mean absolute percentage error) [1] as our accuracy metric. Fig. 2 illustrates the prediction results. We can observe that for all 20 voltage levels, the MAPE of **DeVos** is less than 5%. Moreover, the accuracy of random forest with 20 decision trees shows best performance among the three models. This is coherent with the reasoning that random forest performs better with more decision trees. Our future work will investigate the trade-off between accuracy and running time.

4 CONCLUSION AND FUTURE WORK

This paper presents **DeVos**, a dynamic delay model of voltage-scaled circuits. It predicts the dynamic delay of circuits under different input data and voltage levels. Using random forest, it trains the data points extracted from gate-level simulation and can predict the dynamic delay of several multiplier circuits with a less than 5% MAPE. This promising result motivates us to develop learning-based models for more complex circuits.

REFERENCES

[1] J Scott Armstrong et al. Error measures for generalizing about forecasting methods: Empirical comparisons. *International journal of forecasting*, 1992.
[2] Jeremy Constantin et al. Exploiting dynamic timing margins in microprocessors for frequency-over-scaling with instruction-based clock adjustment. In *DATE*, 2015.
[3] Xun Jiao et al. Wild: A workload-based learning model to predict dynamic delay of functional units. In *ICCD*. IEEE, 2016.
[4] Fabian Pedregosa et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 2011.
[5] Sanghamitra Roy et al. Predicting timing violations through instruction-level path sensitization analysis. In *DAC*, 2012.
[6] Muhammad Shafique et al. A low latency generic accuracy configurable adder. In *DAC*, 2015.
[7] Jing Xin et al. Identifying and predicting timing-critical instructions to boost timing speculation. In *MICRO*, 2011.
[8] Bo Zhai et al. Theoretical and practical limits of dynamic voltage scaling. In *DAC*, 2004.