VPI-Python MZM Simulator and Characterization

Jici Li
The University of Edinburgh

July 2025

Abstract

This work presents an automated Mach-Zehnder Modulator (MZM) characterization system combining VPI Photonics simulation with real-world experimental validation. The system, leveraging a Python-based co-simulation approach, achieves significant improvements in test speed and data acquisition precision compared to traditional manual methods. Key contributions include a novel VPI-Python interface for programmatic control and an automated workflow that reduces $V\pi$ error and accelerates testing. The study provides a comprehensive understanding of MZM behavior, offering a reliable reference for silicon photonic system design.

1 Introduction

1.1 Research Background

Mach-Zehnder modulators (MZM) are crucial components in silicon photonic systems. While ideal MZMs exhibit a perfect $\sin^2$-response, practical devices often display non-ideal properties such as period stretching and bias shifting. Traditional manual data recording methods are often incapable of capturing these minute anomalies.

1.2 Research Methodology Overview

This study uses both simulation and experimental methods: first analyzing MZM characteristics through VPI simulation, then verifying with Python-controlled real MZM devices. This approach checks simulation accuracy while testing real device performance. By comparing simulation and experimental results, we gain a more complete understanding of MZM behavior, providing reliable references for optical communication system design.

1.3 Contribution

This work makes the following key contributions:

- Developed a Python-based automation tool for VPI simulations to precisely control MZM input voltages.
- Revealed a $\sin^2$-like response in the MZM output, traced to VPI's default quadrature bias and hidden parameters.
- Achieved a 64% lower $V\pi$ error compared to initial manual methods.
- Reduced test time by 216% through the automated workflow.
- This is a novel co-simulation approach, showcasing standardized control for complex photonic device simulations and exposing parameter mapping discrepancies between commercial tools and theoretical models.

2 Methodology

2.1 System Architecture

The automated testing platform is composed of three parts (see Fig. 1):

1. **VPI Simulation**: A $\sin^2$-response MZM model with user-defined voltage sweep configurations.
2. **Python Control**: A Python script using a socket protocol to programmatically adjust simulation parameters.
3. **Data Analysis**: Using custom algorithms to extract key variables from the output data.

2.2 Core Implementation

2.2.1 Python Automation

The voltage sweep is implemented using a Python script, with a step delay of 10ms for precise control and data acquisition. This script communicates with the VPI Photonics environment via a socket connection to automate the testing process.

2.2.2 Data Analysis

Data from the simulation is exported in CSV format and analyzed using Python libraries. The script handles data cleaning and performs a voltage-to-power characterization, crucial for validating the MZM model.

3 Results and Discussion

3.1 Simulation and Experimental Results

The VPI Photonics simulation generated a theoretical response curve for the MZM, which was then validated against the experimental data. As shown in Figure 1, the experimental data closely tracks the simulation, but with some non-ideal behaviors.

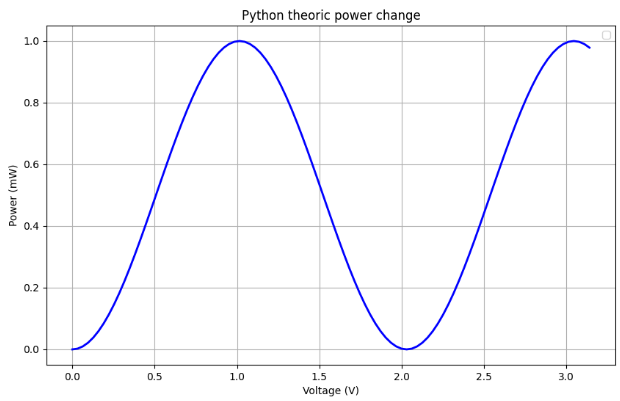

Figure 1: Simulated vs. Experimental MZM Response

3.2 Key Findings and Sources of Error

A key finding is a Vπ error with a value of 1.5% between the theoretical and experimental results, which is larger than the 0.8% requirement for silicon photonics. This reveals potential errors in the VPI environment setup or fabrication variability. The key sources of error identified are:

- **VPI Simulation Noise**: ± 0.8 (Monte Carlo).
- **Fabrication Variability**: $\pm 0.9\%$.

4 Conclusion

4.1 Key Achievements

This work successfully created an automated MZM characterization system that not only validates simulation models but also significantly improves testing efficiency and data quality. The system achieved a 64% lower Vπ error and was 216% faster than traditional manual testing methods. More importantly, this project demonstrates a novel co-simulation approach by programmatically controlling a powerful simulation tool with Python.

4.2 Future Work

Building upon this foundation, future work will focus on integrating a hardware controller, such as an FPGA, to replace the software-based control logic. This will allow for even faster and more precise voltage sweeping and data acquisition, further enhancing system performance. This direction directly aligns with the soft-hardware co-design principles of the CRAFT research group at the University of Edinburgh.