eSim Semester Long Internship Autumn 2025

On

A Study on the Integration, Simulation, and Performance Analysis of IHP OpenPDK-Based CMOS Circuits in eSim

Submitted by

Keerthana R

Sri Eshwar College of Engineering
Coimbatore

Under the guidance of

Prof. Prabhu Ramachandran

Aerospace Engineering Department
IIT Bombay

January 29, 2026

Acknowledgment

I would like to express my deepest gratitude to Prof. Prabhu Ramachandran for providing me the opportunity to be a part of the FOSSEE internship program and for supporting open-source engineering tool development at IIT Bombay. His guidance and vision have encouraged students and researchers to actively contribute to the open-source ecosystem.

I would also like to acknowledge Prof. Kannan M. Moudgalya for his foundational role in establishing and nurturing the FOSSEE initiative. His contributions toward opensource education and the creation of the FOSSEE fellowship framework have directly shaped the platform through which this internship was undertaken.

My sincere appreciation extends to my mentor, Sumanto Kar, for his continual support, technical guidance, and encouragement throughout the duration of this project. His insights and feedback played a key role in refining ideas, overcoming challenges, and ensuring timely completion of the tasks assigned to me.

I would also like to thank my internal mentors, Mr. Varad Patil, Ms. Shanthi Priya K, and Mr. Aditya M, for their valuable guidance, coordination, and technical inputs during the internship. Their mentorship contributed significantly to the clarity, progress, and successful execution of the work.

This internship has been an enriching learning experience, allowing me to work closely with open-source EDA tools, develop IC subcircuits in eSim, and gain exposure to realworld circuit modeling and simulation workflows. The knowledge acquired during this period will undoubtedly support my future academic and professional pursuits.

I would also like to thank the entire FOSSEE team for their coordination, assistance, and timely interactions at various stages of this work. Their collective efforts ensured smooth workflow, resource accessibility, and effective project execution.

Contents

Chapter 1

Introduction

1.1 Background of Electronic Design Automation (EDA)

Electronic Design Automation (EDA) tools automate and streamline the design, simulation, and verification of electronic circuits. Traditional EDA tools are proprietary and often expensive, limiting access for students and researchers. Open-source EDA tools like eSim provide a cost-effective platform for learning, experimentation, and research in analog and digital circuit design.

1.2 Importance of Open-Source EDA Tools

Open-source EDA platforms offer transparency, flexibility, and reproducibility. They allow students to:

- Access advanced design features without licensing costs
- Understand internal modeling through SPICE-level simulation
- Develop skills relevant to industry-standard workflows

1.3 Motivation for Using eSim and OpenPDKs

eSim integrates KiCad for schematic capture and NgSpice for simulation. Combining eSim with the open-source IHP OpenPDK enables simulation of CMOS circuits with realistic device parameters. This integration bridges academic learning and practical semiconductor design workflows.

1.4 Scope

- Configure IHP OpenPDK with eSim
- Design basic CMOS circuits
- Perform DC and transient simulation analyses
- Document workflow and challenges
- Validate circuit performance

Chapter 2

Overview OF eSim

2.1 Introduction to eSim

eSim is an open-source Electronic Design Automation tool developed under the Free and Open Source Software for Education (FOSSEE) initiative. It is designed to support analog, digital, and mixed-signal circuit simulation. eSim combines multiple open-source tools into a unified graphical environment.

2.2 Architecture of eSim

The architecture of eSim consists of a schematic editor, simulation engine, and waveform viewer. KiCad is used for schematic capture, while NgSpice handles simulation tasks. This modular architecture allows flexibility and extensibility.

2.3 Features of eSim

Key features of eSim include:

- User-friendly schematic editor
- Support for SPICE simulations
- Custom library and model integration
- Cross-platform compatibility
- Educational and research suitability

2.4 Tools Integrated within eSim

eSim integrates KiCad for schematic design, NgSpice for simulation, and additional utilities for waveform visualization and analysis. This integration simplifies the circuit design workflow for users.

Chapter 3

Overview OF IHP OpenPDK

3.1 Process Design Kits (PDKs)

A Process Design Kit (PDK) is a comprehensive set of data files and documentation provided by semiconductor foundries to enable circuit designers to accurately model and simulate integrated circuits for a specific fabrication process. PDKs form the foundation of any integrated circuit design flow, as they define the electrical behavior of devices under different operating conditions.

Typically, a PDK includes:

- Transistor SPICE models describing currentvoltage characteristics
- Parasitic capacitance and resistance information
- Process-specific parameters such as threshold voltage, channel length, and oxide thickness
- Design rules and technology constraints

Without a PDK, circuit simulations rely on ideal or simplified models, which do not accurately reflect real silicon behavior. Therefore, PDKs are essential for achieving reliable, predictive circuit simulations before fabrication.

3.2 Need for Open-Source PDKs

Most commercially available PDKs are proprietary and distributed only under strict non-disclosure agreements. This significantly limits their availability to academic institutions and independent researchers. As a result, students often lack exposure to realistic transistor-level design workflows.

Open-source PDKs address this limitation by:

- Providing unrestricted access to device models
- Enabling reproducible and transparent research
- Allowing students to explore industry-relevant design methodologies
- Supporting learning without financial or licensing barriers

The availability of open-source PDKs such as IHP OpenPDK plays a crucial role in democratizing semiconductor education and fostering innovation in open hardware development.

3.3 Introduction to IHP OpenPDK

IHP OpenPDK is an open-source Process Design Kit developed and released by IHP Microelectronics. It is intended primarily for academic, research, and educational use.

The PDK includes detailed SPICE-level models for NMOS and PMOS transistors that accurately represent the electrical behavior of CMOS devices.

The IHP OpenPDK is compatible with open-source simulators such as NgSpice, making it well suited for integration with tools like eSim. By using IHP OpenPDK, designers can simulate CMOS circuits with realistic device parameters, bridging the gap between theoretical design and practical semiconductor behavior.

3.4 Components of IHP OpenPDK

The IHP OpenPDK consists of several essential components that support circuit simulation:

- NMOS and PMOS SPICE model files: These define the electrical characteristics of transistors, including current flow, threshold voltage, and channel effects.
- Device parameter definitions: These parameters specify physical and electrical properties such as channel length, oxide thickness, and mobility.
- Supporting model libraries: Additional libraries provide dependencies required for accurate simulation and convergence.

Together, these components enable realistic transistor-level simulation of CMOS circuits within SPICE-based environments.

3.5 Advantages of IHP OpenPDK

The key advantages of IHP OpenPDK include:

- Free and open-source availability
- Realistic semiconductor device modeling
- Compatibility with open-source EDA tools
- Suitability for academic instruction and research
- Support for hands-on CMOS design experimentation

These advantages make IHP OpenPDK an ideal choice for educational projects focused on VLSI design and simulation.

Chapter 4

Objectives

4.1 Primary Objectives

The primary objective of this internship is to successfully integrate IHP OpenPDK with eSim in order to enable accurate CMOS circuit simulation using open-source tools. This integration aims to demonstrate the feasibility of performing PDK-based circuit design without relying on proprietary software.

4.2 Technical Objectives

The technical objectives of this internship include:

- Configuring IHP NMOS and PMOS SPICE models within the eSim simulation environment
- Designing a CMOS inverter using realistic transistor models
- Performing transient and DC sweep analyses using NgSpice
- Verifying the functional correctness and switching behavior of the inverter

These objectives focus on building practical skills in transistor-level circuit simulation.

4.3 Learning Objectives

The learning objectives of the internship are:

- To understand PDK-based CMOS design workflows

- To gain hands-on experience with SPICE simulation techniques
- To develop familiarity with open-source EDA tools
- To learn best practices in documentation and open-source contributions

These learning outcomes contribute to both technical and professional development.

Chapter 5

Tools and Technologies Used

5.1 Software Tools

The primary software tool used in this internship is eSim, an open-source EDA platform developed by FOSSEE, IIT Bombay. eSim provides a unified environment for schematic capture, simulation, and analysis. Its integration with KiCad and NgSpice allows users to design and simulate circuits efficiently.

In addition to eSim, KiCad was used internally for schematic capture. KiCad is an open-source electronics design automation suite that supports hierarchical schematics and component libraries.

5.2 Simulation Tools

NgSpice is the core simulation engine used in this work. It is an open-source mixed-signal circuit simulator derived from SPICE. NgSpice supports DC, AC, transient, and parametric analyses, making it suitable for evaluating digital and analog circuits.

NgSpice was chosen due to its compatibility with IHP OpenPDK transistor models and its ability to simulate large-scale circuits with high accuracy.

5.3 Version Control and Documentation Tools

Git and GitHub were used for version control and documentation management. Git allows tracking of changes and maintaining clean commit histories. GitHub was used to submit documentation contributions to the FOSSEE eSim repository through pull requests, following open-source contribution guidelines.

5.4 Operating System and System Requirements

The simulations were performed on a Linux-based operating system. Linux is preferred for open-source EDA tools due to better compatibility, stability, and ease of package management.

Chapter 6

Installation and Setup

6.1 Installation of eSim

The eSim tool was installed by following the official installation guidelines provided by FOSSEE. After installation, example circuits were simulated to verify proper functioning of the schematic editor and simulation engine.

The successful execution of sample simulations ensured that KiCad and NgSpice were correctly integrated within eSim.

6.2 Installation of IHP OpenPDK

The IHP OpenPDK repository was cloned from GitHub. The repository contains SPICE models, technology files, and supporting libraries. Special attention was given to locating NMOS and PMOS SPICE model files required for CMOS circuit simulation.

Build Tools Installation

```
sudo apt-get install -y build-essential
sudo apt-get install -y qtbase5-dev qttools5-dev
sudo apt-get install -y clang cmake libtool autoconf
sudo apt-get install -y python3 python3-dev python3-pip python3-virtualenv python3-
sudo apt-get install -y ruby ruby-dev
```

Useful Tools Installation

```
sudo apt-get install -y btop tree xterm graphviz git
```

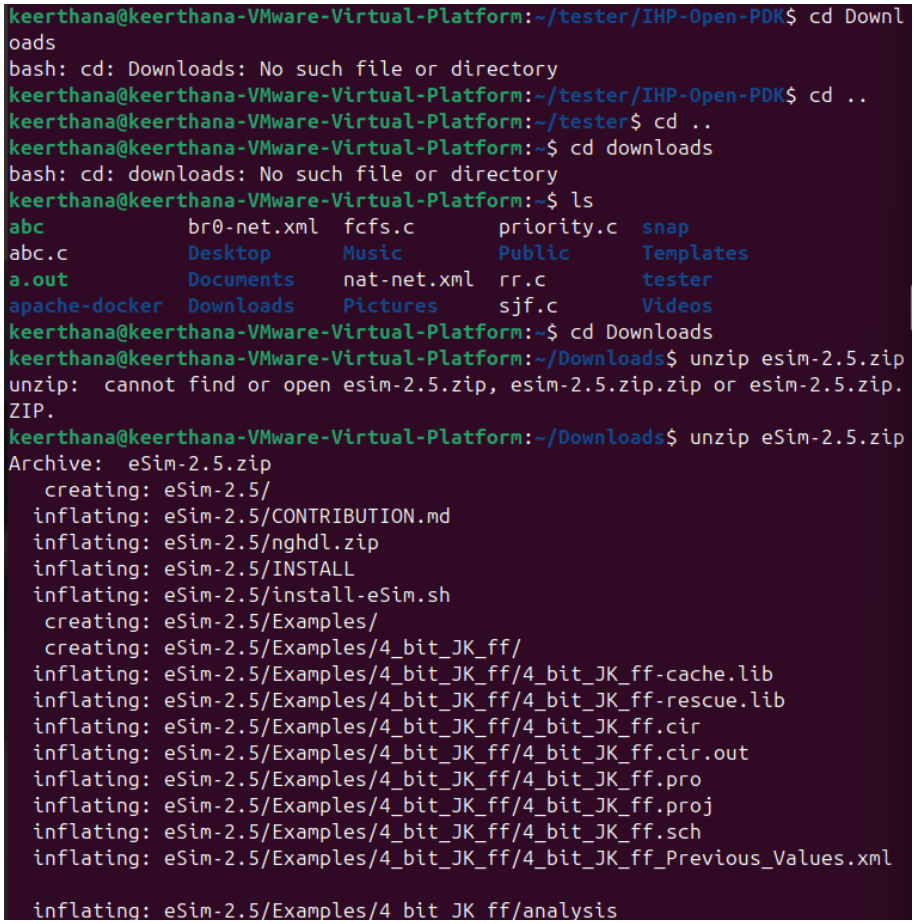

Figure 6.1: Installation

```
sudo apt-get install -y octave liboctave-dev
```

Build Dependencies

```
sudo apt-get install -y python3-sphinx python3-sphinx-autoapi python3-pandas python
sudo apt-get install -y libqt5xmlpatterns5-dev qtmultimedia5-dev libqt5multimediawi
sudo apt-get install -y tcl8.6 tcl-dev tcl8.6-dev
sudo apt-get install -y tk8.6 tk8.6-dev
sudo apt-get install -y flex clang gawk xdot pkg-config bison curl help2man perl ti
sudo apt-get install -y libxpm4 libxpm-dev libgtk-3-dev libffi-dev
sudo apt-get install -y libjpeg-dev libfl-dev libfl2
sudo apt-get install -y libreadline-dev gettext
sudo apt-get install -y libboost-system-dev libboost-python-dev libboost-filesystem
sudo apt-get install -y libx11-6 libx11-dev
sudo apt-get install -y libxrender1 libxrender-dev
sudo apt-get install -y libxcb1 libx11-xcb-dev
```

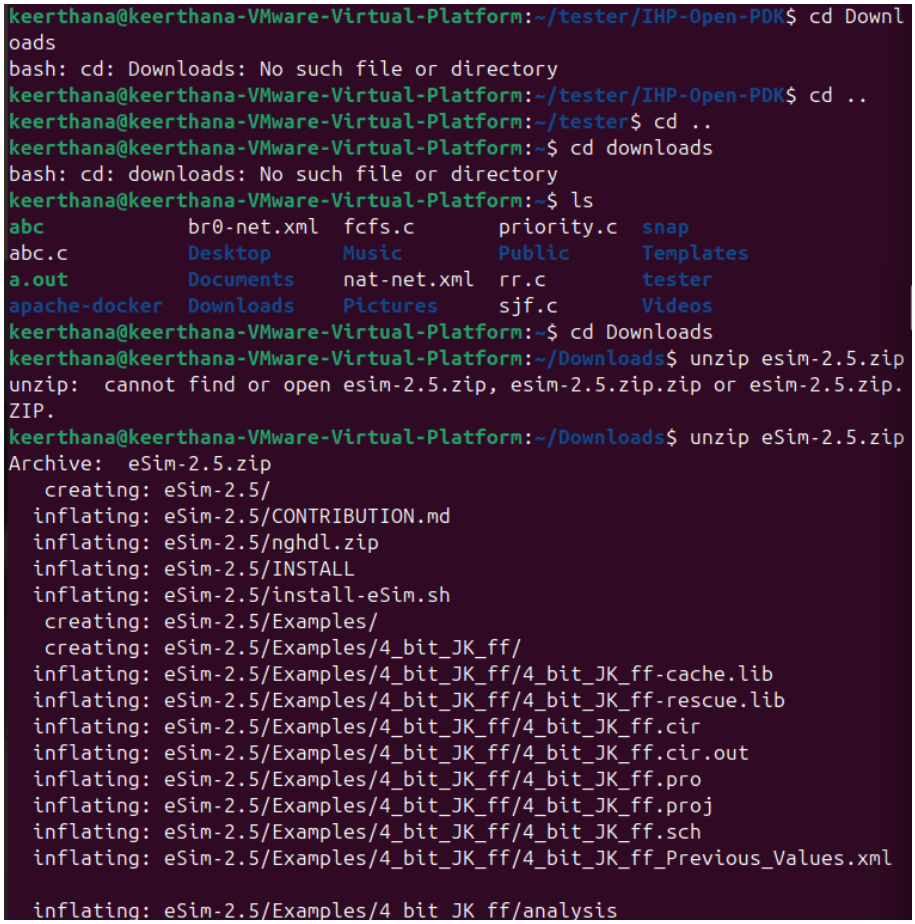

Figure 6.2: Installing dependencies

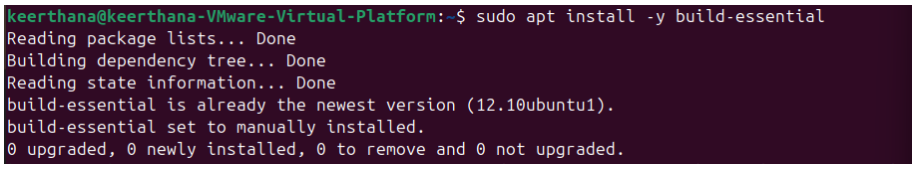

Figure 6.3: Build tools Installation

```
sudo apt-get install -y libcairo2 libcairo2-dev libxaw7-dev
sudo apt-get install -y libgz libfl2 libfl-dev zlibc zzlib1g zlib1g-dev libz-dev li
sudo apt-get install -y libgoogle-perftools-dev
sudo apt-get install -y gengetopt groff pod2pdf libhpdf-dev
sudo apt-get install -y libfftw3-dev
sudo apt-get install -y libxml-libxml-perl libgd-perl
sudo apt-get install -y libsuitesparse-dev gfortran swig libspdlog-dev libeigen3-de
```

```
keerthana@keerthana-VMware-Virtual-Platform:~$ sudo apt-get install -y qtbase5-dev qttools
5-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libclang1-15t64 libegl-dev libgl-dev libglu1-mesa-dev libglx-dev libllvm15t64
  libopengl-dev libqt5concurrent5t64 libqt5designer5 libqt5designercomponents5
  libqt5opengl5-dev libqt5positioning5 libqt5quickwidgets5 libqt5sensors5 libqt5test5t64
  libqt5webchannel5 libqt5webkit5 libvulkan-dev libxext-dev qdoc-qt5 qhelpgenerator-qt5
  qt5-assistant qt5-qmake qt5-qmake-bin qtattributionsscanner-qt5 qtbase5-dev-tools
  qtchooser qttools5-dev-tools
Suggested packages:
  libxext-doc qt5-doc default-libmysqlclient-dev firebird-dev libpq-dev libsqlite3-dev
  unixodbc-dev
The following NEW packages will be installed:
  libclang1-15t64 libegl-dev libgl-dev libglu1-mesa-dev libglx-dev libllvm15t64
  libopengl-dev libqt5concurrent5t64 libqt5designer5 libqt5designercomponents5
  libqt5opengl5-dev libqt5positioning5 libqt5quickwidgets5 libqt5sensors5 libqt5test5t64
  libqt5webchannel5 libqt5webkit5 libvulkan-dev libxext-dev qdoc-qt5 qhelpgenerator-qt5
  qt5-assistant qt5-qmake qt5-qmake-bin qtattributionsscanner-qt5 qtbase5-dev
  qtbase5-dev-tools qtchooser qttools5-dev qttools5-dev-tools
0 upgraded, 30 newly installed, 0 to remove and 0 not upgraded.
Need to get 57.0 MB of archives.
After this operation, 271 MB of additional disk space will be used.
Get:1 http://in.archive.ubuntu.com/ubuntu noble/universe amd64 libqt5positioning5 amd64 5.
15.13+dfsg-1 [222 kB]
```

Figure 6.4: qtbase Installation

```
keerthana@keerthana-VMware-Virtual-Platform:~$ sudo apt-get install -y clang cmake libtool
 autoconf
[sudo] password for keerthana:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
autoconf is already the newest version (2.71-3).
autoconf set to manually installed.
The following additional packages will be installed:
  clang-18 cmake-data icu-devtools lib32gcc-s1 lib32stdc++6 libc6-i386
  libclang-common-18-dev libclang-rt-18-dev libffi-dev libgc1 libicu-dev libjsoncpp25
  libltdl-dev libncurses-dev libobjc-13-dev libobjc4 libpfm4 librhash0 libxml2-dev
  libz3-4 libz3-dev llvm-18 llvm-18-dev llvm-18-linker-tools llvm-18-runtime
  llvm-18-tools
Suggested packages:
  clang-18-doc wasi-libc cmake-doc cmake-format elpa-cmake-mode ninja-build icu-doc
  libtool-doc ncurses-doc gfortran | fortran95-compiler gcj-jdk pkg-config llvm-18-doc
The following NEW packages will be installed:
  clang clang-18 cmake cmake-data icu-devtools lib32gcc-s1 lib32stdc++6 libc6-i386
  libclang-common-18-dev libclang-rt-18-dev libffi-dev libgc1 libicu-dev libjsoncpp25
  libltdl-dev libncurses-dev libobjc-13-dev libobjc4 libpfm4 librhash0 libtool
  libxml2-dev libz3-4 libz3-dev llvm-18 llvm-18-dev llvm-18-linker-tools llvm-18-runtime
  llvm-18-tools
0 upgraded, 29 newly installed, 0 to remove and 0 not upgraded.
Need to get 115 MB of archives.
After this operation, 669 MB of additional disk space will be used.
```

Figure 6.5: cmake Installation

Directory Management and Cloning

6.3 Environment Variable Configuration

Environment variables were configured to allow eSim and NgSpice to access the IHP OpenPDK directory. Absolute paths were used to avoid file inclusion errors during simulation. This step is crucial, as incorrect path configuration can result in missing model errors and simulation failure.

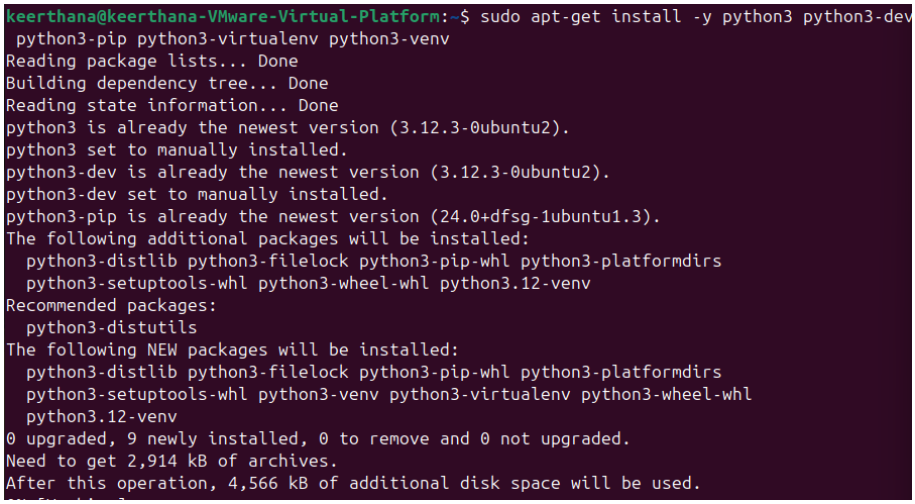

Figure 6.6: python packages installation

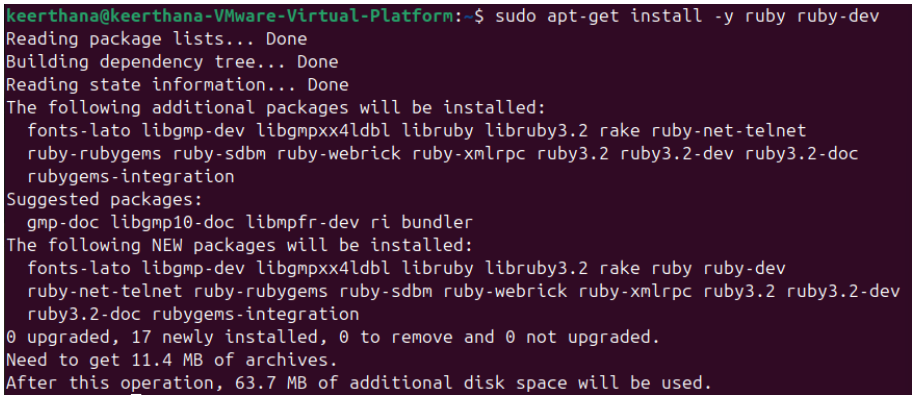

Figure 6.7: ruby Installation

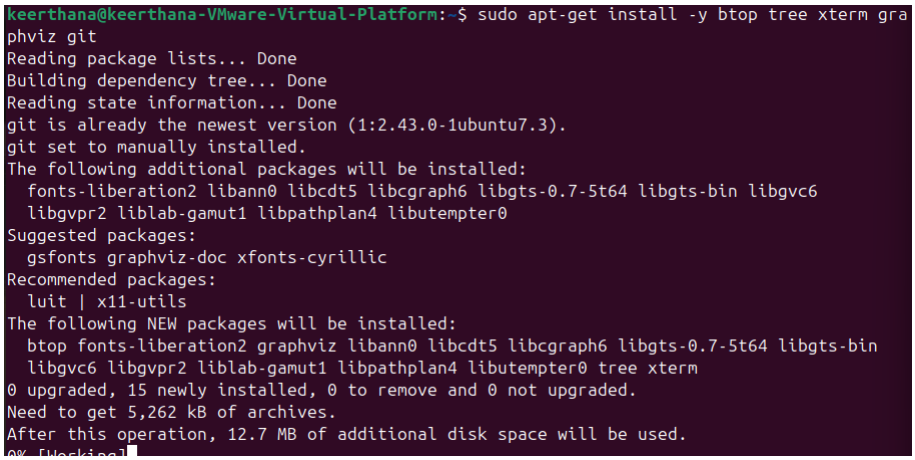

Figure 6.8: btop tree Installation

6.4 Verification of Installation

Installation verification was performed by including the IHP SPICE models in a simple test circuit. Successful simulation confirmed correct configuration and model

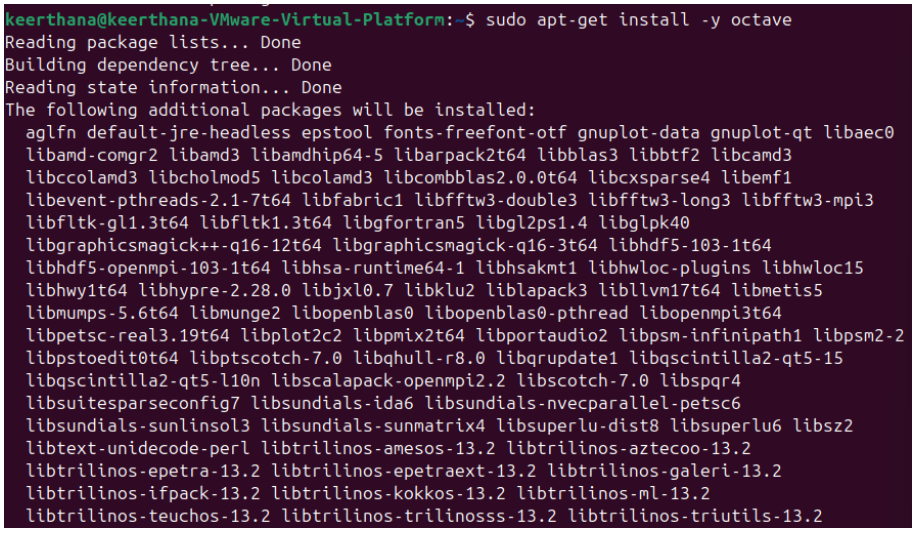

Figure 6.9: octave Installation

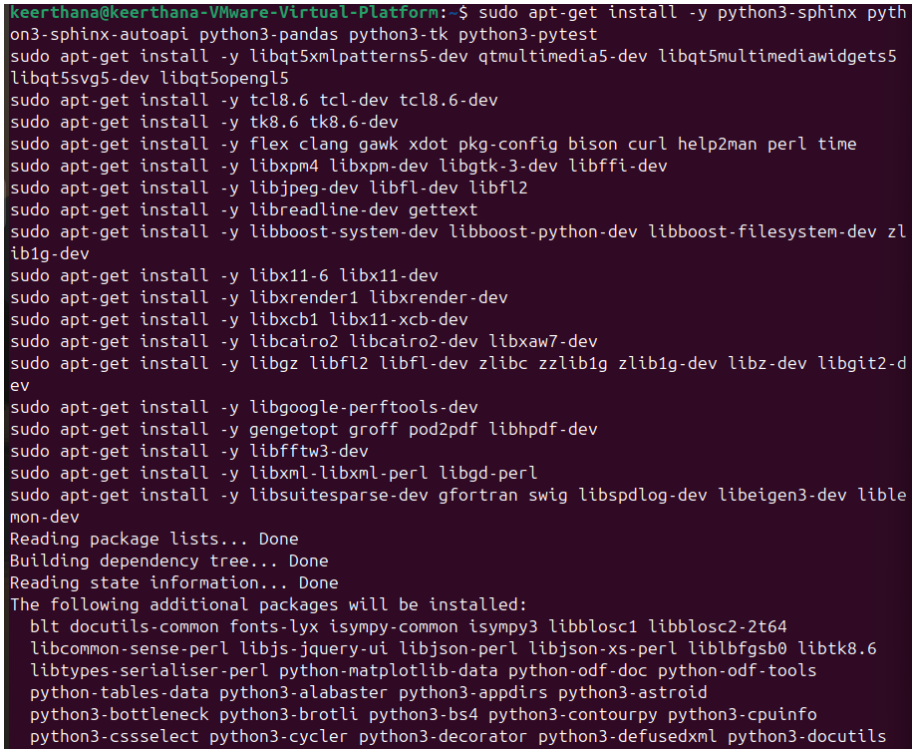

Figure 6.10: sphinx Installation

compatibility.

6.5 Common Installation Issues

Several common issues were encountered, such as missing dependencies, incorrect file paths, and permission errors. These were resolved by verifying directory structures,

```
keerthana@keerthana-VMware-Virtual-Platform:~$ mkdir tester
keerthana@keerthana-VMware-Virtual-Platform:~$ cd tester
keerthana@keerthana-VMware-Virtual-Platform:~/tester$ git clone --branch de
v --recurse-submodules https://github.com/IHP-GmbH/IHP-Open-PDK.git
Cloning into 'IHP-Open-PDK'...
fatal: unable to access 'https://github.com/IHP-GmbH/IHP-Open-PDK.git/': Co
uld not resolve host: github.com
keerthana@keerthana-VMware-Virtual-Platform:~/tester$ git clone --branch de
v --recurse-submodules https://github.com/IHP-GmbH/IHP-Open-PDK.git
Cloning into 'IHP-Open-PDK'...
remote: Enumerating objects: 15229, done.
remote: Counting objects: 100% (1964/1964), done.
remote: Compressing objects: 100% (421/421), done.
remote: Total 15229 (delta 1681), reused 1543 (delta 1543), pack-reused 132
65 (from 3)
Receiving objects: 100% (15229/15229), 398.69 MiB | 3.63 MiB/s, done.
Resolving deltas: 100% (9070/9070), done.
Updating files: 100% (4775/4775), done.
Submodule 'ihp-sg13g2/libs.tech/digital' (https://github.com/hneemann/IHP13
0.git) registered for path 'ihp-sg13g2/libs.tech/digital'
Submodule 'ihp-sg13g2/libs.tech/klayout/python/pycell4klayout-api' (https:/
/github.com/IHP-GmbH/pycell4klayout-api) registered for path 'ihp-sg13g2/li
bs.tech/klayout/python/pycell4klayout-api'
Submodule 'ihp-sg13g2/libs.tech/klayout/python/pypreprocessor' (https://git
hub.com/IHP-GmbH/pypreprocessor) registered for path 'ihp-sg13g2/libs.tech/
klayout/python/pypreprocessor'
Submodule 'ihp-sg13g2/libs.tech/openems/openems_ihp_sg13g2' (https://github
.com/VolkerMuehlhaus/openems_ihp_sg13g2) registered for path 'ihp-sg13g2/li
bs.tech/openems/openems_ihp_sg13g2'
Cloning into '/home/keerthana/tester/IHP-Open-PDK/ihp-sg13g2/libs.tech/digi
tal'...
remote: Enumerating objects: 61, done.
remote: Counting objects: 100% (61/61), done.
```

Figure 6.11: Directory management and git cloning

```
keerthana@keerthana-VMware-Virtual-Platform:~/tester$ cd IHP-Open-PDK
keerthana@keerthana-VMware-Virtual-Platform:~/tester/IHP-Open-PDK$ echo "ex
port PDK_ROOT=\$HOME/your_directory/IHP-Open-PDK" >> ~/.bashrc
echo "export PDK=ihp-sg13g2" >> ~/.bashrc
echo "export KLAYOUT_PATH=\"\$HOME/.klayout:\$PDK_ROOT/\$PDK/libs.tech/klay
out\"" >> ~/.bashrc
echo "export KLAYOUT_HOME=\$HOME/.klayout" >> ~/.bashrc
source ~/.bashrc
```

Figure 6.12: Directory modifications

using absolute paths, and ensuring correct file permissions.

Chapter 7

Circuit Design Methodology

7.1 Introduction to CMOS Design

Complementary Metal-Oxide-Semiconductor (CMOS) technology is widely used in modern digital integrated circuits due to its low power consumption and high noise immunity. CMOS circuits utilize complementary pairs of NMOS and PMOS transistors, ensuring that only one transistor conducts at a time during steady-state operation.

The CMOS inverter is the most fundamental building block in digital logic design, forming the basis for more complex circuits such as logic gates, multiplexers, and sequential elements.

7.2 CMOS Inverter Architecture

A CMOS inverter consists of:

- A PMOS transistor connected between the supply voltage (VDD) and the output node
- An NMOS transistor connected between the output node and ground

The gates of both transistors are connected to the input signal. When the input is low, the PMOS conducts and pulls the output high. When the input is high, the NMOS conducts and pulls the output low. This complementary action ensures efficient switching and minimal static power consumption.

7.3 Selection of Design Parameters

Design parameters play a crucial role in determining inverter performance. In this work:

- A supply voltage of 1.8 V was selected based on IHP OpenPDK specifications
- Input voltage was varied between 0 V and 1.8 V
- Pulse input parameters were chosen to clearly observe transient switching behavior

These parameters were selected to ensure stable simulation and accurate representation of CMOS inverter characteristics.

7.4 Schematic Design Using eSim

The CMOS inverter schematic was created using the eSim schematic editor. NMOS and PMOS symbols were placed and connected according to CMOS inverter topology. Each transistor symbol was linked to the corresponding IHP OpenPDK SPICE model to ensure realistic simulation behavior.

Special care was taken to correctly assign node connections, power supply references, and ground nodes.

7.5 SPICE Model Integration

SPICE model integration was performed using .include statements within the NgSpice configuration. Absolute file paths were used to reference the IHP OpenPDK model files, preventing file-not-found errors during simulation.

This step is critical for ensuring that NgSpice correctly interprets the transistor models.

7.6 Netlist Generation

Once the schematic design was completed, eSim automatically generated the SPICE netlist required for simulation. The generated netlist was reviewed to verify correct component connections, model references, and simulation commands.

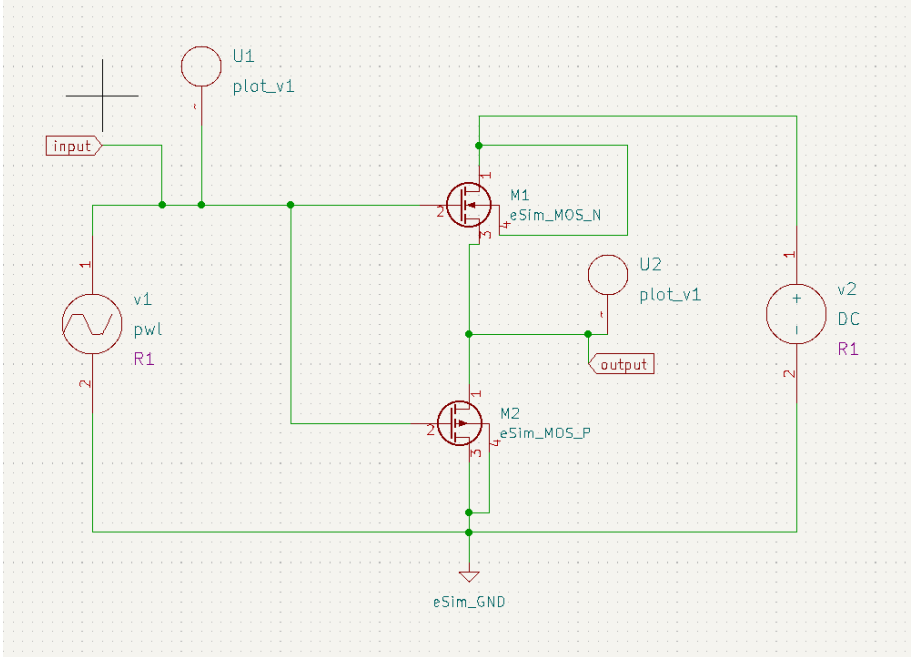

Figure 7.1: Circuit Diagram

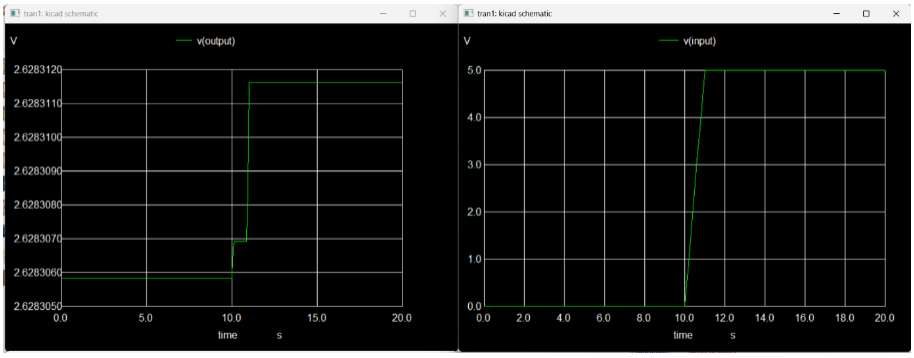

Figure 7.2: Output Waveform

Chapter 8

Simulation and Analysis

8.1 Introduction to SPICE Simulations

SPICE simulations allow designers to analyze circuit behavior before physical implementation. By applying mathematical models to circuit elements, SPICE predicts voltage and current behavior under different conditions.

In this project, SPICE simulations were used to validate both static and dynamic behavior of the CMOS inverter.

8.2 Transient Analysis

Transient analysis evaluates time-domain response of the circuit. A pulse input was applied to the inverter, and the output waveform was observed over time. The simulation confirmed correct logical inversion during rising and falling transitions of the input signal.

8.3 DC Sweep Analysis

DC sweep analysis was performed by varying the input voltage gradually from 0 V to 1.8 V. The resulting voltage transfer characteristic (VTC) curve provided insight into the switching threshold and noise margins of the inverter.

8.4 Simulation Parameters

Simulation accuracy was enhanced by selecting appropriate time steps and convergence settings. Smaller time steps were used to capture rapid voltage transitions and avoid numerical instability.

8.5 Interpretation of Simulation Results

The simulation results demonstrated:

- Correct inverter operation
- Stable switching behavior
- Switching threshold near mid-supply voltage

Chapter 9

Results and Discussion

9.1 Transient Response Analysis

Transient response analysis was performed to study the time-domain behavior of the CMOS inverter. A pulse input signal varying between 0 V and 1.8 V was applied at the inverter input. The corresponding output waveform was observed to analyze the switching behavior.

The simulation results showed that when the input voltage was low (logic 0), the output voltage remained high (logic 1), and when the input voltage was high (logic 1), the output voltage transitioned to low (logic 0). The rise time and fall time of the output waveform were observed to be smooth and continuous, indicating stable inverter operation.

Minor delays between input transitions and output responses were observed due to parasitic capacitances and realistic transistor characteristics present in the IHP OpenPDK models. These delays reflect real-world behavior and confirm the accuracy of the PDK-based simulation.

9.2 Voltage Transfer Characteristics (VTC)

The voltage transfer characteristic (VTC) of the CMOS inverter was obtained through DC sweep analysis by varying the input voltage from 0 V to 1.8 V while monitoring the output voltage.

The resulting VTC curve exhibited a sharp transition region between logic high and logic low output levels. The output voltage remained close to the supply voltage for low input voltages and transitioned rapidly to ground as the input voltage increased beyond the switching region.

The steep slope of the transition region indicates a high voltage gain, which is a

desirable property of CMOS inverters. The VTC characteristics confirmed that the inverter operates with strong noise immunity and reliable logic-level separation.

9.3 Switching Threshold Analysis

The switching threshold voltage is defined as the input voltage at which the output voltage equals the input voltage. From the VTC curve, the switching threshold was observed to be approximately near half of the supply voltage.

This balanced switching threshold indicates symmetric behavior between the NMOS and PMOS transistors. The threshold location confirms that the selected transistor dimensions and model parameters were appropriate for inverter design.

A well-centered switching threshold ensures improved noise margins and reliable digital operation, which is essential for larger digital circuit integration.

9.4 Performance Evaluation

The performance of the CMOS inverter was evaluated based on its switching behavior, stability, and signal integrity. The inverter demonstrated:

- Proper logical inversion
- Stable output levels
- Smooth transient response
- Minimal distortion in output waveform

Propagation delay and rise/fall time characteristics were consistent with expectations for CMOS circuits using realistic transistor models. The results indicate that the inverter design meets fundamental digital performance requirements.

Although detailed power and delay optimization was not performed, the inverter architecture inherently supports low static power consumption due to complementary transistor operation.

9.5 Validation of Results

The simulation results were validated by comparing them with theoretical CMOS inverter behavior and standard digital design principles. The observed waveforms, VTC characteristics, and switching threshold closely matched expected results.

Minor deviations from ideal behavior were attributed to non-ideal transistor characteristics modeled in the IHP OpenPDK, such as parasitic effects and finite mobility. These deviations further validate the realism of the simulation environment.

Overall, the results confirm that the integration of IHP OpenPDK with eSim provides accurate and reliable simulation capabilities suitable for academic and research-oriented CMOS circuit design.

Chapter 10

Issues Faced and Troubleshooting

10.1 File Path and Model Inclusion Errors

One of the major challenges encountered was related to incorrect file paths and improper inclusion of SPICE model files. The eSim environment relies on accurate .include statements to access external model libraries. Even minor errors in directory paths resulted in simulation failures.

To resolve these issues:

- Absolute file paths were used instead of relative paths
- File locations were verified prior to simulation
- All required model files were placed in organized directories

Proper file management significantly reduced model inclusion errors and ensured smooth simulations.

10.2 Convergence Issues in NgSpice

Several simulation issues were traced back to errors in the automatically generated SPICE netlists. These included incorrect node names, syntax inconsistencies, and mismatched model references.

To resolve these issues:

- The generated netlist was manually inspected
- Simulations were performed incrementally
- Individual components were tested before full integration

This systematic debugging approach proved effective in isolating and correcting errors, significantly improving simulation reliability.

10.3 Simulation Timing and Accuracy Issues

Simulation inaccuracies were observed when inappropriate time-step values were selected. Larger time steps caused waveform distortions and missed switching events, affecting the accuracy of transient response analysis.

To improve timing accuracy:

- Time-step values were reduced
- Input signal rise and fall times were adjusted
- Simulation duration was extended for better observation

These changes resulted in clearer and more accurate waveform representations.

10.4 Debugging Strategies Adopted

A structured debugging methodology was adopted to identify and resolve errors efficiently. Rather than debugging the entire circuit at once, problems were isolated and addressed incrementally.

Key strategies included:

- Simulating individual components before full integration
- Manually inspecting SPICE netlists generated by eSim
- Verifying node connections and model references
- Testing simplified versions of the circuit

This systematic approach significantly reduced debugging time and improved simulation reliability.

10.5 Lessons Learned from Troubleshooting

The troubleshooting process provided valuable insights into effective circuit simulation practices. The key lessons learned include:

- Accurate file organization is critical for PDK integration
- Understanding SPICE error messages aids faster debugging
- Incremental testing improves design reliability
- Proper documentation simplifies future modifications

These lessons enhanced both technical proficiency and problem-solving skills.

Chapter 11

Conclusion

11.1 Summary of Work Completed

The major tasks completed during this internship include:

- Installation and configuration of eSim and IHP OpenPDK
- Integration of SPICE transistor models into eSim
- Design of a CMOS inverter using eSim schematic editor
- Generation and verification of SPICE netlists
- Simulation and analysis of inverter performance

Each task contributed to a comprehensive understanding of the open-source VLSI design workflow.

11.2 Key Achievements

The key achievements of this internship are:

- Successful integration of IHP OpenPDK with eSim
- Accurate simulation of CMOS inverter behavior
- Validation of PDK-based simulation accuracy
- Development of structured debugging and analysis skills

These achievements demonstrate the technical depth and practical relevance of the work.

11.3 Overall Impact of the Internship

This internship significantly enhanced both technical and professional skills. It provided hands-on exposure to realistic semiconductor design workflows and reinforced the importance of open-source EDA tools in academic and research environments.

The experience gained during this internship has laid a strong foundation for advanced studies and research in VLSI design, semiconductor technology, and electronic system design.

Bibliography

1. FOSSEE eSim Project. eSim Open Source EDA Tool for Circuit Design and Simulation. Available at: https://esim.fossee.in/resources

2. IHP Microelectronics. IHP Open Process Design Kit (OpenPDK) Documentation. Available at: https://www.ihp-microelectronics.com/services/research-and-prototyping-service/process-design-kits

3. Ngspice Development Team. Ngspice Open Source SPICE Simulator for Electric and Electronic Circuits. Available at: https://ngspice.sourceforge.io/docs/ngspice-manual.pdf

4. Wikipedia Contributors. CMOS Inverter Working and Characteristics. Available at: https://en.wikipedia.org/wiki/CMOS

5. SPICE Simulation Fundamentals. SPICE Modeling and Circuit Simulation Concepts. Available at: https://www.electronics-tutorials.ws/simulation/spice-simulation.html

6. Open Source EDA Community. Introduction to Process Design Kits (PDKs). Available at: https://openroad.readthedocs.io/en/latest/