Winter Internship Report

On

Upgradation of eSim Installer to Ubuntu 24.04

Submitted by

Harsha Narayana P

Under the guidance of

Sumantho Kar

IIT Bombay

February 15, 2025

Acknowledgement

We would like to express our sincere gratitude to the **FOSSEE, IIT Bombay Team** for providing the opportunity to work on the **upgradation of the eSim installer to Ubuntu 24.04**. This project provided invaluable insights into open-source EDA tools and their role in circuit simulation.

We extend our sincere regards to **Mr. Sumanto Kar** for his valuable guidance and motivation throughout this project.

We would like to express our heartfelt appreciation to the entire FOSSEE team, includingmyour mentor Mr. Sumanto Kar, for their constant guidance and support throughout the duration of this internship. Mr. Sumanto Kar's accessibility and expertise were invaluable in debugging and resolving issues.

This experience has been a milestone in our pursuit of a successful career in software development and open-source contributions, enhancing our skills in scripting, debugging, and collaborative software engineering.

Contents

1. Introduction

1.1. Background of eSim

eSim is an open-source Electronic Design Automation (EDA) tool designed for circuit design, simulation, and PCB layout. Developed as part of the FOSSEE (Free/Libre and Open Source Software for Education) initiative at IIT Bombay, eSim integrates multiple open-source tools to provide a cost-effective alternative to proprietary EDA software.

1.1.1 Key Features

- **Schematic Design and Simulation:** eSim allows users to create circuit schematics and perform simulations using tools such as Ngspice. This enables detailed circuit analysis and verification.
- **PCB Layout and Fabrication:** The tool supports Printed Circuit Board (PCB) layout design using KiCad, facilitating multi-layer board development and generating manufacturing-ready Gerber files.
- **Mixed-Signal Simulation:** By integrating tools like GHDL and Verilator, eSim enables the co-simulation of both analog and digital circuits.
- **Customization and Extensibility:** Users can add new device models and subcircuits, allowing for enhanced simulation capabilities tailored to specific project requirements.
- **Cross-Platform Support:** eSim is available for both Linux (Ubuntu) and Windows, ensuring accessibility across different operating systems.

1.1.2 Significance and Adoption

eSim serves as a free alternative to commercial EDA tools such as OrCAD, Xpedition, and HSPICE. Its open-source nature makes it especially beneficial for academic institutions, research labs, and small businesses, reducing software costs while promoting collaborative development. By integrating various open-source technologies, eSim provides a comprehensive workflow for circuit design, from initial schematics to final PCB layouts.

1.1.3 Role of eSim in Open-Source Education

As part of the FOSSEE initiative, eSim promotes the adoption of open-source software in engineering education. It has been widely adopted by students, educators, and researchers for learning and developing electronic circuits. The tool's modular structure ensures continuous improvements through community contributions, making it a sustainable choice for long-term academic and professional use.

1.1.4 Why Upgrade is Required?

With the release of Ubuntu 24.04, certain dependencies and package management structures have changed, affecting the functionality of the existing eSim installer script. This

report addresses these challenges by identifying errors in the installation process, analyzing their root causes, and proposing effective solutions to ensure seamless installation on the latest Ubuntu version.

1.2. Objectives

This report aims to:

- Explain the installer script.
- Identify and analyze errors that prevent installation.
- Provide solutions and modifications to fix these issues.
- Present the final working script and installation guide.

2. Understanding the Installer Script

The eSim installer script (`install-eSim.sh`) is a Bash script that automates the installation and configuration of the eSim EDA Suite. This section provides a detailed explanation of the script's structure, functionality, and key components.

2.1. Script Header and Metadata

The script begins with a shebang (`#!/bin/bash`), indicating that it should be executed in the Bash shell. A comment block provides metadata about the script, including:

- **File Name:** `install-eSim.sh`
- **Usage:** `./install-eSim.sh --install` or `./install-eSim.sh --uninstall`
- **Description:** Automates the installation of the eSim EDA Suite.
- **Authors:** Contributors from the eSim Team at FOSSEE, IIT Bombay.
- **Revision Date:** Last modified on June 29, 2023.

2.2. Variable Declarations

Several variables are initialized at the start of the script:

- `config_dir`: Stores the configuration directory path (`$HOME/.esim`).
- `config_file`: Defines the configuration file name (`config.ini`).
- `eSim_Home`: Captures the current working directory.
- `ngspiceFlag`: A flag used to track the installation status of NGHDL.

2.3. Function Definitions

The script defines multiple functions to modularize various tasks:

2.3.1 Error Handling Function

```
error_exit() {
    echo -e "\n\nError! Kindly resolve above error(s) and try again."
    echo -e "\nAborting Installation...\n"
}
```

This function is called whenever an error occurs, aborting the installation.

2.3.2 Configuration File Creation

```
function createConfigFile {
    if [ -d $config_dir ]; then
        rm $config_dir/$config_file && touch $config_dir/$config_file
    else
        mkdir $config_dir && touch $config_dir/$config_file
    fi

    echo "[eSim]" >> $config_dir/$config_file
    echo "eSim_HOME = $eSim_Home" >> $config_dir/$config_file
}
```

This function creates a configuration file containing installation details.

2.3.3 Installing NGHDL

```
function installNghdl {
    echo "Installing NGHDL..."
    unzip -o nghdl.zip
    cd nghdl/
    chmod +x install-nghdl.sh
    ./install-nghdl.sh --install
    cd ../
}
```

NGHDL is a tool that integrates with Ngspice for mixed-signal simulations, enabling VHDL-based circuit modeling.

2.3.4 Installing KiCad and Dependencies

```
function installKicad {
    echo "Installing KiCad..."
    sudo add-apt-repository -y ppa:kicad/kicad-6.0-releases
    sudo apt-get update
    sudo apt-get install -y --no-install-recommends kicad kicad-footprints
}
```

KiCad is a vital component of eSim, providing schematic capture and PCB layout capabilities.

2.4. Installing System Dependencies

The script installs multiple dependencies required for eSim:

```
sudo apt-get install -y xterm python3-psutil python3-pyqt5 python3-matplotlib python3-distut
```

Additionally, Python libraries such as Watchdog, Makerchip, and SandPiper-SaaS are installed via `pip3`.

2.5. Copying KiCad Libraries

```
function copyKicadLibrary {
    tar -xJf library/kicadLibrary.tar.xz
    sudo cp -r kicadLibrary/eSim-symbols/* /usr/share/kicad/symbols/
}
```

The function extracts and installs KiCad libraries required for eSim's circuit design capabilities.

2.6. Creating Desktop Shortcut

To simplify access, the script generates an executable startup script and a desktop shortcut:

```
echo '[Desktop Entry]' > esim.desktop
echo 'Exec=esim %u' >> esim.desktop
echo 'Terminal=true' >> esim.desktop
sudo cp -vp esim.desktop /usr/share/applications/
```

2.7. Main Execution Flow

The script executes different functions based on the command-line argument:

- **Installation** (`--install`): Calls functions sequentially to install dependencies, KiCad, NGHDL, and configuration files.
- **Uninstallation** (`--uninstall`): Removes all installed components and restores the system to its prior state.

2.8. Handling Proxy Settings

If the system is behind a proxy, the script prompts for proxy details and sets appropriate environment variables:

```
export http_proxy=http://$username:$passwd@$proxyHostname:$proxyPort
export https_proxy=http://$username:$passwd@$proxyHostname:$proxyPort
```

2.9. Error Handling and Permissions

The script uses traps to catch errors and ensure proper execution. It also modifies file ownership and permissions where necessary:

```
sudo chown -R $USER:$USER /usr/share/kicad/symbols/
```

3. Error Analysis and Resolution

During the installation of eSim on Ubuntu various errors were obtained. Below are the causes and solutions for each error encountered.

3.1. Error 1: Removal of `python3-distutils`

3.1.1 Error Message

When executing the following command:

```
sudo apt-get install -y python3-distutils
```

the installation failed with the following error:

Error Output

```
Package python3-distutils is not available, but is referred to by
↪  another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source.

E: Package 'python3-distutils' has no installation candidate

Error! Kindly resolve above error(s) and try again.

Aborting Installation...
```

3.1.2 Cause

Starting from Python 3.12, the `distutils` module has been deprecated and removed from the Python standard library. Since Ubuntu 24.04 ships with Python 3.12 by default, the package `python3-distutils` is no longer available in the system repositories. The recommended alternative is to use `setuptools`, which provides similar functionality [2].

3.1.3 Solution

To resolve this issue, the installer script was modified to install `python3-setuptools` instead:

```
sudo apt-get install -y python3-setuptools
```

This ensures that all functionality dependent on `distutils` remains available through `setuptools` [?].

3.2. Error 2: `externally-managed-environment` During Package Installation

3.2.1 Error Message

While attempting to install Python packages using `pip3`, the following error was encountered:

Error Output

```
error: externally-managed-environment

× This environment is externally managed
> To install Python packages system-wide, try apt install
    python3-xyz, where xyz is the package you are trying to
    install.

If you wish to install a non-Debian-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have python3-full installed.

If you wish to install a non-Debian packaged Python application,
it may be easiest to use pipx install xyz, which will manage a
virtual environment for you. Make sure you have pipx installed.

See /usr/share/doc/python3.11/README.venv for more information.

note: If you believe this is a mistake, please contact your Python
↪  installation or OS distribution
provider. You can override this, at the risk of breaking your Python
↪  installation or OS, by passing
--break-system-packages.

hint: See PEP 668 for the detailed specification.

Error! Kindly resolve above error(s) and try again.

Aborting Installation...
```

3.2.2 Cause

This error arises due to the implementation of PEP (Python Enhancement Proposal) 668, which marks system-wide Python environments as "externally managed." This designation prevents the installation of Python packages directly into the system Python to avoid conflicts between the operating system's package manager and `pip`. Ubuntu 24.04, which includes Python 3.12, enforces this by restricting `pip3` from installing packages system-wide [1].

In the `install-eSim.sh` script, the `installDependency` function attempts to install

Python packages using `pip3`:

```
echo "Installing Watchdog........................"
pip3 install watchdog
echo "Installing Hdlparse........................"
pip3 install --upgrade https://github.com/hdl/pyhdlparser/tarball/master
echo "Installing Makerchip......................."
pip3 install makerchip-app
echo "Installing SandPiper Saas.................."
pip3 install sandpiper-saas
```

These commands trigger the `externally-managed-environment` error because they attempt to install packages directly into the system-managed Python environment.

3.2.3 Solution

To resolve this issue, it's necessary to install Python packages within a virtual environment, thereby isolating them from the system Python and adhering to the constraints imposed by PEP 668. The following modifications were made to the `install-eSim.sh` script:

1. **Install the `python3-venv` Package:**

Before creating a virtual environment, ensure that the `python3-venv` package is installed. Add the following lines after line 137 inside the `installDependency` function:

```
    echo "Installing Python-venv..................."
    sudo apt install python3-venv
```

2. **Ensure the Application Uses the Virtual Environment:**

Modify the `createDesktopStartScript` function. Add the following line before line 219:

```
 echo "source $eSim_Home/esim-venv/bin/activate" >> esim-start.sh
```

By implementing these changes, the installation process complies with PEP 668 guidelines, preventing the `externally-managed-environment` error and ensuring that all required Python packages are installed in an isolated environment, thereby maintaining system integrity.

3.3. Error 3: KiCad 6.0 PPA Not Found for Ubuntu 24.04 (Noble)

3.3.1 Error Message

During the installation of KiCad, the following error was encountered when attempting to add the KiCad 6.0 PPA:

Error Output

```
E: The repository
↪  'https://ppa.launchpadcontent.net/kicad/kicad-6.0-releases/ubuntu
↪  noble Release' does not have a Release file.
```

3.3.2 Cause

This error occurs because KiCad 6.0 is no longer supported on Ubuntu 24.04 (Noble), and the repository does not contain a release file for this version. The installation script tries to add an outdated PPA, leading to the failure.

In the `install-eSim.sh` script, the following function is responsible for installing KiCad:

```
function installKicad {
    echo "Installing KiCad..."

    kicadppa="kicad/kicad-6.0-releases"
    findppa=$(grep -h -r "^deb.*$kicadppa*" /etc/apt/sources.list* >
    ↪  /dev/null
    2>&1 || test $? = 1)
    if [ -z "$findppa" ]; then
        echo "Adding KiCad-6 ppa to local apt-repository"
        sudo add-apt-repository -y ppa:kicad/kicad-6.0-releases
        sudo apt-get update
    else
        echo "KiCad-6 is available in synaptic"
    fi

    sudo apt-get install -y --no-install-recommends kicad kicad-footprints \
    kicad-libraries kicad-symbols kicad-templates
}
```

Since the PPA is unavailable for Ubuntu 24.04, an alternative approach is required.

3.3.3 Solution

To resolve this issue, the installation script was modified to detect the Ubuntu version and install the correct KiCad version. For Ubuntu 24.04, KiCad 8.0 should be used instead of KiCad 6.0. Below is the updated function:

```
function installKicad
{
    echo "Installing KiCad..........................."

    # Detect Ubuntu version
    ubuntu_version=$(lsb_release -rs)

    # Define KiCad PPAs based on Ubuntu version
```

```
    if [[ "$ubuntu_version" == "24.04" ]]; then
        echo "Ubuntu 24.04 detected."
        kicadppa="kicad/kicad-8.0-releases"

        # Check if KiCad is installed using dpkg-query for the main package
        if dpkg -s kicad &>/dev/null; then
            installed_version=$(dpkg-query -W -f='${Version}' kicad | cut
            ↪  -d'.' -f1)
            if [[ "$installed_version" != "8" ]]; then
                echo "A different version of KiCad ($installed_version) is
                ↪  installed."
                read -p "Do you want to remove it and install KiCad 8.0?
                ↪  (yes/no): " response

                if [[ "$response" =~ ^([Yy][Ee][Ss]|[Yy])$ ]]; then
                    echo "Removing KiCad $installed_version..."
                    sudo apt-get remove --purge -y kicad kicad-footprints
                    ↪  kicad-libraries kicad-symbols kicad-templates
                    sudo apt-get autoremove -y
                else
                    echo "Exiting installation. KiCad $installed_version
                    ↪  remains installed."
                    exit 1
                fi
            else
                echo "KiCad 8.0 is already installed."
                exit 0
            fi
        fi

    else
        kicadppa="kicad/kicad-6.0-releases"
    fi

    # Check if the PPA is already added
    if ! grep -q "^deb .*${kicadppa}" /etc/apt/sources.list
    ↪  /etc/apt/sources.list.d/* 2>/dev/null; then
        echo "Adding KiCad PPA to local apt repository: $kicadppa"
        sudo add-apt-repository -y "ppa:$kicadppa"
        sudo apt-get update
    else
        echo "KiCad PPA is already present in sources."
    fi

    # Install KiCad packages
    sudo apt-get install -y --no-install-recommends kicad kicad-footprints
    ↪  kicad-libraries kicad-symbols kicad-templates

    echo "KiCad installation completed successfully!"
}
```

This updated script ensures that:

- It correctly identifies Ubuntu 24.04 and installs KiCad 8.0 instead of 6.0.
- It prevents unnecessary reinstallation if KiCad 8.0 is already installed.
- It prompts the user before removing a different installed KiCad version.
- It prevents redundant PPA additions.
- It ensures that the installation process completes successfully with proper dependency handling.

3.4. Error 4: Unable to Locate Package `llvm-9` and `llvm-9-dev`

3.4.1 Error Message

During the installation process, the following error was encountered:

Error Output

```
E: Unable to locate package llvm-9
E: Unable to locate package llvm-9-dev
```

3.4.2 Cause

The error occurs because the installer script attempts to install `llvm-9` and `llvm-9-dev`, which are outdated packages and are no longer available in the latest Ubuntu 24.04 repositories.

The `install-eSim.sh` script includes a subscript called `install-nghdl.sh`, which is extracted from `nghdl.zip`. This script is responsible for installing NGHDL, a tool used for mixed-signal simulations.

The existing NGHDL package within the installer is based on GHDL version 0.37, which depends on LLVM 9. However, in a recent update to the NGHDL repository [4], the installation script was modified to use GHDL version 4.1.0. This updated version replaces LLVM 9 with the latest available LLVM packages, ensuring compatibility with modern systems.

Since the `install-nghdl.sh` script was last updated on **Tuesday, 31 December 2024**, the current eSim installer should be updated to reflect these changes by using:

- The latest NGHDL package: `ghdl-4.1.0.tar.gz`
- The updated `install-nghdl.sh` script from the NGHDL repository.

3.4.3 Solution

To resolve this issue, the existing `nghdl.zip` used in the eSim installer should be updated with the latest `ghdl-4.1.0.tar.gz` package and the improved `install-nghdl.sh` script. This ensures compatibility with the latest LLVM versions available in Ubuntu 24.04 and prevents installation failures.

3.5. Error 5: Verilator Compilation Failure (`unique_ptr` Not Found in `std`)

3.5.1 Error Message

During the build process of NGHDL, the following compilation error was encountered:

Error Output

```
../V3Const.cpp:244:22: error: 'unique_ptr' is not a member of 'std'
```

3.5.2 Cause

This error occurs during the compilation of Verilator in the `nghdl/verilator-4.210/src/V3Const.cpp` file. The error message indicates that the C++ standard library does not recognize `unique_ptr` as a member of the `std` namespace.

The root cause is the absence of the `<memory>` header file, which is required for `std::unique_ptr`. Without explicitly including this header, the C++ compiler fails to recognize the `unique_ptr` class, leading to a compilation failure.

To fix this issue, the necessary `#include <memory>` directive was added to the affected file in the NGHDL repository. This fix was applied via the following commit [5]:

- Commit ID: `5177de2ad8a5351bc1c31dedbe445305aede2995`
- Repository: `https://github.com/FOSSEE/nghdl`
- Branch: `installers`

3.5.3 Solution

The error has been resolved in the latest version of Verilator, which is included in the updated NGHDL package. To ensure successful installation, the eSim installer should use the updated `verilator-4.210.tar.gz` from the NGHDL repository.

4. Final Upgraded Installer Script

The complete upgraded script after fixing errors:

```
1 #!/bin/bash
2 #================================================================================
3 #          FILE: install-eSim.sh
4 #
```

```
#         USAGE: ./install-eSim.sh --install
#                                OR
#                ./install-eSim.sh --uninstall
#
#   DESCRIPTION: Installation script for eSim EDA Suite
#
#       OPTIONS: ---
#  REQUIREMENTS: ---
#          BUGS: ---
#         NOTES: ---
#       AUTHORS: Fahim Khan, Rahul Paknikar, Saurabh Bansode,
#                Sumanto Kar, Partha Singha Roy
#  ORGANIZATION: eSim Team, FOSSEE, IIT Bombay
#       CREATED: Wednesday 15 July 2015 15:26
#      REVISION: Tuesday 31 December 2024 17:28
#===============================================================================

# All variables goes here
config_dir="$HOME/.esim"
config_file="config.ini"
eSim_Home=`pwd`
ngspiceFlag=0

## All Functions goes here

error_exit()
{

    echo -e "\n\nError! Kindly resolve above error(s) and try again."
    echo -e "\nAborting Installation...\n"

}


function createConfigFile
{

    # Creating config.ini file and adding configuration information
    # Check if config file is present
    if [ -d $config_dir ];then
        rm $config_dir/$config_file && touch $config_dir/$config_file
    else
        mkdir $config_dir && touch $config_dir/$config_file
    fi

    echo "[eSim]" >> $config_dir/$config_file
    echo "eSim_HOME = $eSim_Home" >> $config_dir/$config_file
    echo "LICENSE = %(eSim_HOME)s/LICENSE" >> $config_dir/$config_file
    echo "KicadLib = %(eSim_HOME)s/library/kicadLibrary.tar.xz" >>
    ↪  $config_dir/$config_file
    echo "IMAGES = %(eSim_HOME)s/images" >> $config_dir/$config_file
```

```
55      echo "VERSION = %(eSim_HOME)s/VERSION" >> $config_dir/$config_file
56      echo "MODELICA_MAP_JSON =
        ↪  %(eSim_HOME)s/library/ngspicetoModelica/Mapping.json" >>
        ↪  $config_dir/$config_file
57
58  }
59
60
61  function installNghdl
62  {
63
64      echo "Installing NGHDL..........................."
65      unzip -o nghdl.zip
66      cd nghdl/
67      chmod +x install-nghdl.sh
68
69      # Do not trap on error of any command. Let NGHDL script handle its own
        ↪  errors.
70      trap "" ERR
71
72      ./install-nghdl.sh --install       # Install NGHDL
73
74      # Set trap again to error_exit function to exit on errors
75      trap error_exit ERR
76
77      ngspiceFlag=1
78      cd ../
79
80  }
81
82
83  function installSky130Pdk
84  {
85
86      echo "Installing SKY130 PDK....................."
87
88      # Extract SKY130 PDK
89      tar -xJf library/sky130_fd_pr.tar.xz
90
91      # Remove any previous sky130-fd-pdr instance, if any
92      sudo rm -rf /usr/share/local/sky130_fd_pr
93
94      # Copy SKY130 library
95      echo "Copying SKY130 PDK........................."
96
97      sudo mkdir -p /usr/share/local/
98      sudo mv sky130_fd_pr /usr/share/local/
99
100     # Change ownership from root to the user
101     sudo chown -R $USER:$USER /usr/share/local/sky130_fd_pr/
102
```

```
}


function installKicad
{
    echo "Installing KiCad..........................."

    # Detect Ubuntu version
    ubuntu_version=$(lsb_release -rs)

    # Define KiCad PPAs based on Ubuntu version
    if [[ "$ubuntu_version" == "24.04" ]]; then
        echo "Ubuntu 24.04 detected."
        kicadppa="kicad/kicad-8.0-releases"

        # Check if KiCad is installed using dpkg-query for the main package
        if dpkg -s kicad &>/dev/null; then
            installed_version=$(dpkg-query -W -f='${Version}' kicad | cut
            ↪  -d'.' -f1)
            if [[ "$installed_version" != "8" ]]; then
                echo "A different version of KiCad ($installed_version) is
                ↪  installed."
                read -p "Do you want to remove it and install KiCad 8.0?
                ↪  (yes/no): " response

                if [[ "$response" =~ ^([Yy][Ee][Ss]|[Yy])$ ]]; then
                    echo "Removing KiCad $installed_version..."
                    sudo apt-get remove --purge -y kicad kicad-footprints
                    ↪  kicad-libraries kicad-symbols kicad-templates
                    sudo apt-get autoremove -y
                else
                    echo "Exiting installation. KiCad $installed_version
                    ↪  remains installed."
                    exit 1
                fi
            else
                echo "KiCad 8.0 is already installed."
                exit 0
            fi
        fi

    else
        kicadppa="kicad/kicad-6.0-releases"
    fi

    # Check if the PPA is already added
    if ! grep -q "^deb .*${kicadppa}" /etc/apt/sources.list
    ↪  /etc/apt/sources.list.d/* 2>/dev/null; then
        echo "Adding KiCad PPA to local apt repository: $kicadppa"
        sudo add-apt-repository -y "ppa:$kicadppa"
        sudo apt-get update
```

```
148     else
149         echo "KiCad PPA is already present in sources."
150     fi
151 
152     # Install KiCad packages
153     sudo apt-get install -y --no-install-recommends kicad kicad-footprints
        ↪  kicad-libraries kicad-symbols kicad-templates
154 
155     echo "KiCad installation completed successfully!"
156 }
157 
158 
159 function installDependency
160 {
161 
162     set +e      # Temporary disable exit on error
163     trap "" ERR # Do not trap on error of any command
164 
165     # Update apt repository
166     echo "Updating apt index files..................."
167     sudo apt-get update
168 
169     set -e      # Re-enable exit on error
170     trap error_exit ERR
171 
172     echo "Instaling virtualenv......................."
173     sudo apt install python3-virtualenv
174 
175     echo "Creating virtual environment to isolate packages "
176     virtualenv $config_dir/env
177 
178     echo "Starting the virtual env..................."
179     source $config_dir/env/bin/activate
180 
181     echo "Upgrading Pip.............................."
182     pip install --upgrade pip
183 
184     echo "Installing Xterm..........................."
185     sudo apt-get install -y xterm
186 
187     echo "Installing Psutil.........................."
188     sudo apt-get install -y python3-psutil
189 
190     echo "Installing PyQt5..........................."
191     sudo apt-get install -y python3-pyqt5
192 
193     echo "Installing Matplotlib......................"
194     sudo apt-get install -y python3-matplotlib
195 
196     echo "Installing Setuptools....................."
197     sudo apt-get install -y python3-setuptools
```

```
198
199     # Install NgVeri Depedencies
200     echo "Installing Pip3.........................."
201     sudo apt install -y python3-pip
202
203     echo "Installing Watchdog......................."
204     pip3 install watchdog
205
206     echo "Installing Hdlparse......................."
207     pip3 install --upgrade https://github.com/hdl/pyhdlparser/tarball/master
208
209     echo "Installing Makerchip......................"
210     pip3 install makerchip-app
211
212     echo "Installing SandPiper Saas................."
213     pip3 install sandpiper-saas
214
215
216     echo "Installing Hdlparse......................."
217     pip3 install hdlparse
218
219     echo "Installing matplotlib................."
220     pip3 install matplotlib
221
222     echo "Installing PyQt5............."
223     pip3 install PyQt5
224 }
225
226
227 function copyKicadLibrary
228 {
229
230     #Extract custom KiCad Library
231     tar -xJf library/kicadLibrary.tar.xz
232
233     if [ -d ~/.config/kicad/6.0 ];then
234         echo "kicad config folder already exists"
235     else
236         echo ".config/kicad/6.0 does not exist"
237         mkdir -p ~/.config/kicad/6.0
238     fi
239
240     # Copy symbol table for eSim custom symbols
241     cp kicadLibrary/template/sym-lib-table ~/.config/kicad/6.0/
242     echo "symbol table copied in the directory"
243
244     # Copy KiCad symbols made for eSim
245     sudo cp -r kicadLibrary/eSim-symbols/* /usr/share/kicad/symbols/
246
247     set +e      # Temporary disable exit on error
248     trap "" ERR # Do not trap on error of any command
```

```
249
250     # Remove extracted KiCad Library - not needed anymore
251     rm -rf kicadLibrary
252
253     set -e      # Re-enable exit on error
254     trap error_exit ERR
255
256     #Change ownership from Root to the User
257     sudo chown -R $USER:$USER /usr/share/kicad/symbols/
258
259 }
260
261
262 function createDesktopStartScript
263 {
264
265     # Generating new esim-start.sh
266     echo '#!/bin/bash' > esim-start.sh
267     echo "cd $eSim_Home/src/frontEnd" >> esim-start.sh
268     echo "source $config_dir/env/bin/activate" >> esim-start.sh
269     echo "python3 Application.py" >> esim-start.sh
270
271     # Make it executable
272     sudo chmod 755 esim-start.sh
273     # Copy esim start script
274     sudo cp -vp esim-start.sh /usr/bin/esim
275     # Remove local copy of esim start script
276     rm esim-start.sh
277
278     # Generating esim.desktop file
279     echo "[Desktop Entry]" > esim.desktop
280     echo "Version=1.0" >> esim.desktop
281     echo "Name=eSim" >> esim.desktop
282     echo "Comment=EDA Tool" >> esim.desktop
283     echo "GenericName=eSim" >> esim.desktop
284     echo "Keywords=eda-tools" >> esim.desktop
285     echo "Exec=esim %u" >> esim.desktop
286     echo "Terminal=true" >> esim.desktop
287     echo "X-MultipleArgs=false" >> esim.desktop
288     echo "Type=Application" >> esim.desktop
289     getIcon="$config_dir/logo.png"
290     echo "Icon=$getIcon" >> esim.desktop
291     echo "Categories=Development;" >> esim.desktop
292     echo
      ↪  "MimeType=text/html;text/xml;application/xhtml+xml;application/xml;application/rss+x
      ↪  >> esim.desktop
293     echo "StartupNotify=true" >> esim.desktop
294
295     # Make esim.desktop file executable
296     sudo chmod 755 esim.desktop
297     # Copy desktop icon file to share applications
```

```
298     sudo cp -vp esim.desktop /usr/share/applications/
299     # Copy desktop icon file to Desktop
300     cp -vp esim.desktop $HOME/Desktop/
301 
302     set +e      # Temporary disable exit on error
303     trap "" ERR # Do not trap on error of any command
304 
305     # Make esim.desktop file as trusted application
306     gio set $HOME/Desktop/esim.desktop "metadata::trusted" true
307     # Set Permission and Execution bit
308     chmod a+x $HOME/Desktop/esim.desktop
309 
310     # Remove local copy of esim.desktop file
311     rm esim.desktop
312 
313     set -e      # Re-enable exit on error
314     trap error_exit ERR
315 
316     # Copying logo.png to .esim directory to access as icon
317     cp -vp images/logo.png $config_dir
318 
319 }
320 
321 
322 ##########################################################################
323 #                    MAIN START FROM HERE                                #
324 ##########################################################################
325 
326 ### Checking if file is passsed as argument to script
327 
328 if [ "$#" -eq 1 ];then
329     option=$1
330 else
331     echo "USAGE : "
332     echo "./install-eSim.sh --install"
333     echo "./install-eSim.sh --uninstall"
334     exit 1;
335 fi
336 
337 ## Checking flags
338 
339 if [ $option == "--install" ];then
340 
341     set -e  # Set exit option immediately on error
342     set -E  # inherit ERR trap by shell functions
343 
344     # Trap on function error_exit before exiting on error
345     trap error_exit ERR
346 
347 
348     echo "Enter proxy details if you are connected to internet thorugh proxy"
```

```
349
350     echo -n "Is your internet connection behind proxy? (y/n): "
351     read getProxy
352     if [ $getProxy == "y" -o $getProxy == "Y" ];then
353         echo -n 'Proxy Hostname :'
354         read proxyHostname
355
356         echo -n 'Proxy Port :'
357         read proxyPort
358
359         echo -n username@$proxyHostname:$proxyPort :
360         read username
361
362         echo -n 'Password :'
363         read -s passwd
364
365         unset http_proxy
366         unset https_proxy
367         unset HTTP_PROXY
368         unset HTTPS_PROXY
369         unset ftp_proxy
370         unset FTP_PROXY
371
372         export http_proxy=http://$username:$passwd@$proxyHostname:$proxyPort
373         export https_proxy=http://$username:$passwd@$proxyHostname:$proxyPort
374         export https_proxy=http://$username:$passwd@$proxyHostname:$proxyPort
375         export HTTP_PROXY=http://$username:$passwd@$proxyHostname:$proxyPort
376         export HTTPS_PROXY=http://$username:$passwd@$proxyHostname:$proxyPort
377         export ftp_proxy=http://$username:$passwd@$proxyHostname:$proxyPort
378         export FTP_PROXY=http://$username:$passwd@$proxyHostname:$proxyPort
379
380         echo "Install with proxy"
381
382     elif [ $getProxy == "n" -o $getProxy == "N" ];then
383         echo "Install without proxy"
384
385     else
386         echo "Please select the right option"
387         exit 0
388     fi
389
390     # Calling functions
391     createConfigFile
392     installDependency
393     installKicad
394     copyKicadLibrary
395     installNghdl
396     installSky130Pdk
397     createDesktopStartScript
398
399     if [ $? -ne 0 ];then
```

```
400         echo -e "\n\n\nERROR: Unable to install required packages. Please
            ↪ check your internet connection.\n\n"
401         exit 0
402     fi
403
404     echo "-----------------eSim Installed Successfully-----------------"
405     echo "Type \"esim\" in Terminal to launch it"
406     echo "or double click on \"eSim\" icon placed on Desktop"
407
408
409 elif [ $option == "--uninstall" ];then
410     echo -n "Are you sure? It will remove eSim completely including KiCad,
        ↪ Makerchip, NGHDL and SKY130 PDK along with their models and libraries
        ↪ (y/n):"
411     read getConfirmation
412     if [ $getConfirmation == "y" -o $getConfirmation == "Y" ];then
413         echo "Removing eSim..........................."
414         sudo rm -rf $HOME/.esim $HOME/Desktop/esim.desktop /usr/bin/esim
            ↪ /usr/share/applications/esim.desktop
415         echo "Removing KiCad..........................."
416         sudo apt purge -y kicad kicad-footprints kicad-libraries kicad-symbols
            ↪ kicad-templates
417         sudo rm -rf /usr/share/kicad
418         sudo rm /etc/apt/sources.list.d/kicad*
419         rm -rf $HOME/.config/kicad/6.0
420
421         echo "Removing Virtual env......................."
422         sudo rm -r $config_dir/env
423
424         echo "Removing SKY130 PDK......................."
425         sudo rm -R /usr/share/local/sky130_fd_pr
426
427         echo "Removing NGHDL..........................."
428         rm -rf library/modelParamXML/Nghdl/*
429         rm -rf library/modelParamXML/Ngveri/*
430         cd nghdl/
431         if [ $? -eq 0 ];then
432                 chmod +x install-nghdl.sh
433                 ./install-nghdl.sh --uninstall
434                 cd ../
435                 rm -rf nghdl
436             if [ $? -eq 0 ];then
437                 echo -e "-----------------eSim Uninstalled
                    ↪ Successfully-----------------"
438             else
439                 echo -e "\nError while removing some files/directories in
                    ↪ \"nghdl\". Please remove it manually"
440             fi
441         else
442             echo -e "\nCannot find \"nghdl\" directory. Please remove it
                ↪ manually"
```

```
443         fi
444     elif [ $getConfirmation == "n" -o $getConfirmation == "N" ];then
445         exit 0
446     else
447         echo "Please select the right option."
448         exit 0
449     fi
450 
451 else
452     echo "Please select the proper operation."
453     echo "--install"
454     echo "--uninstall"
455 fi
456 
```

5. Summary and Conclusion

5.1. Summary

This report presents a detailed examination of the modifications required to make the eSim installer compatible with Ubuntu 24.04. The upgrade process involved identifying installation failures, diagnosing their underlying causes, and applying appropriate fixes.

Key improvements made to the installer include:

- Replacing the deprecated `python3-distutils` with `python3-setuptools`, ensuring Python compatibility.
- Addressing the `externally-managed-environment` error by implementing a virtual environment for package installations.
- Updating the installer to use KiCad 8.0 instead of the outdated KiCad 6.0, which is not supported on Ubuntu 24.04.
- Adapting the NGHDL installation to work with newer versions of LLVM and GHDL by integrating the latest `install-nghdl.sh` script.
- Resolving a Verilator compilation issue by incorporating the necessary `<memory>` header update from an upstream repository.

By integrating these fixes, the installer now ensures seamless deployment of eSim on Ubuntu 24.04 while maintaining compatibility with essential dependencies.

5.2. Conclusion

The migration to Ubuntu 24.04 introduced compatibility challenges due to deprecations, package removals, and stricter package management policies. Through systematic analysis and targeted modifications, the eSim installer has been successfully updated to accommodate these changes.

The enhanced installer now:

- Supports the latest Ubuntu version while preserving stability.
- Handles dependency installations efficiently within a virtual environment.
- Ensures proper package management, preventing conflicts and errors.

Future improvements could focus on broadening compatibility across different Linux distributions, refining dependency management, and introducing a more user-friendly installation approach. With these updates, eSim remains a reliable and accessible open-source EDA tool for circuit design and simulation.

5.3. References

References

[1] Python Enhancement Proposal 668, *Python Software Foundation*, `https://peps.python.org/pep-0668/`

[2] Python Packaging User Guide, *Python Packaging Authority*, https://packaging.python.org/guides/tool-recommendations/#use-setuptools

[3] KiCad PPA Documentation, *KiCad EDA*, https://launchpad.net/~kicad

[4] NGHDL Installer Update, *FOSSEE NGHDL Repository*, https://github.com/FOSSEE/nghdl/commit/57b9a2eea7a32e9f72a7b3d214d87823219fa284

[5] Verilator Compilation Fix, *FOSSEE NGHDL Repository*, https://github.com/FOSSEE/nghdl/commit/5177de2ad8a5351bc1c31dedbe445305aede2995