SUMMER FELLOWSHIP REPORT

ON

ESIM ON CLOUD

DYNAMIC CIRCUIT NETLIST EDITOR WITH WIRE MANAGEMENT AND UNDO/REDO FEATURES

Submitted by

Tharageshwaran S

5 years Integrated MSc Cyber Security

PSG College of Technology Coimbatore

Under the guidance of

Prof. Kannan Moudgalya

Chemical Engineering Department

IIT Bombay

May - Dec 2023

Acknowledgements

I, Tharagesh S a summer intern at FOSSEE - ESIM on CLOUD project, am extremely grateful to acknowledge all those who have been involved with the project and have assigned tasks believing in me and have helped me transform my ideas into something really unique.

I whole-heartedly thank **Mr. Nagesh Karmali** and **Ms. Firuza Aibara** for providing me with the opportunity to work on this project. I am really thankful to the mentors for their valuable suggestions. They were always there to show the right path and to help whenever I would hit a roadblock. With the help of their brilliant guidance and management I was able to complete all the tasks assigned to me properly and up to the mark.

I would like to extend my thanks to all my colleagues who did an excellent job of maintaining cohesion and communicating changes all through the fellowship period. Together we have been able to achieve difficult tasks and that too in a timely manner.

Table of Contents

1. INTRODUCTION

This report highlights the tasks undertaken, challenges faced, and solutions implemented during my internship as part of the eSim-Cloud project. My primary responsibility was to address issues and implement enhancements in the EDA (Electronic Design Automation) front-end system. The internship provided a valuable learning opportunity to work collaboratively on an open-source project, gaining hands-on experience in frontend and backend development using technologies like ReactJS, Django, and Docker.

2. PROJECT OVERVIEW

The eSim-Cloud project focuses on providing a robust platform for circuit simulation and design. My role was concentrated on enhancing the usability and functionality of the EDA front-end. The project involved debugging, adding features, and ensuring seamless integration with existing systems. The following sections outline the specific tasks and implementations undertaken during this period.

3. TASKS AND IMPLEMENTATIONS

3.1. NETLIST GENERATION FOR EXTERNAL JSON FILES

One of the initial challenges encountered was the failure to generate a proper netlist when uploading circuit designs through JSON files. To address this issue, I conducted an in-depth exploration of the system's file structure and codebase to gain a comprehensive understanding of the existing implementation. Through careful analysis, I pinpointed the root cause of the problem within the JavaScript files of the Electronic Design Automation (EDA) frontend. After identifying the issue, I implemented the necessary fixes to resolve the underlying defects, ensuring that the netlist generation process functioned as intended. To validate the solution, I rigorously tested the system using both the provided JSON files and newly created ones, confirming the effectiveness and reliability of the implemented fixes.

3.2. ADDING WIRES TO GROUND AND COMPONENT LABELING

The system encountered an issue where it failed to provide default labeling when connecting components to the ground or other components. To address this, I meticulously analyzed the codebase and pinpointed the absence of the necessary logic responsible for assigning default labels. I then implemented a solution to ensure proper labeling, such that wires connected to the ground were automatically labeled with '0'. To verify the effectiveness of the fix, I conducted extensive validation across various scenarios, including testing with pre-existing circuits and newly designed ones. This ensured that the labeling functionality worked seamlessly and consistently under diverse conditions.

3.3. UNDO AND REDO FUNCTIONALITY FOR CIRCUIT COMPONENTS

The initial undo and redo functionality faced several limitations. For instance, entire actions were reverted rather than step-by-step changes, and the functionality was disabled when existing files were opened. Additionally, stack size issues arose when managing a large number of changes or performing repeated undo/redo actions, leading to performance bottlenecks. To address these challenges, I restructured the functionality to appropriately disable undo/redo for newly opened files, ensuring a smoother user experience. I also resolved logical inconsistencies to align the behavior of undo and redo actions with user expectations, making the feature more intuitive and reliable. While I attempted to address the stack size issues by optimizing memory management and improving the underlying logic, some challenges persisted, indicating areas for further refinement.

3.4. SUB-CIRCUIT GENERATION

This new feature introduced the ability for users to create and integrate sub-circuits into the existing system, significantly enhancing its flexibility and usability. My contributions involved gaining a comprehensive understanding of the netlist generation process to ensure the feature aligned with the overall system architecture. I configured specific model parameters in the Django backend to facilitate seamless integration of sub-circuits, ensuring smooth communication between the backend and frontend components. Additionally, I collaborated closely with the team to test and validate the feature across various scenarios, focusing on both functionality and user-friendliness. This collective effort ensured the feature was robust, efficient, and met user expectations.

4. CHALLENGES FACED

Understanding the complex codebase of the EDA frontend and backend systems was a time-intensive process, requiring meticulous analysis to grasp the intricate interdependencies and workflows. Debugging issues with undo and redo functionalities presented additional challenges, particularly in addressing stack size limitations and ensuring the feature operated seamlessly under various scenarios. Furthermore, the integration of new features, such as sub-circuits, demanded extensive testing and troubleshooting to ensure smooth functionality, compatibility with existing features, and adherence to user expectations. Each of these tasks required a methodical approach and collaborative effort to achieve a robust and reliable system.

5. KEY LEARNINGS

During this project, I developed proficiency in version control using Git, enabling efficient and collaborative development within the team. I enhanced my understanding of frontend and backend technologies, including ReactJS, Django, and Docker, gaining practical experience in their integration and application. Additionally, I acquired valuable insight into debugging complex systems and implementing new features within an open-source framework, navigating challenges with systematic approaches. This experience also strengthened my problem-solving and collaboration skills, as I worked closely with a diverse team to address issues, optimize workflows, and deliver high-quality solutions.

6. TECHNOLOGIES UTILIZED

This section outlines the technologies used in the project, highlighting their purpose and functionality.

6.1 DJANGO

Django is a high-level Python web framework that encourages rapid development and clean, pragmatic design. It provides a robust set of features for building secure and scalable web applications, including ORM for database interactions, built-in authentication, and an admin interface for easy application management.

6.2 POSTGRESQL

PostgreSQL is an advanced open-source relational database management system (RDBMS) known for its reliability, scalability, and compliance with SQL standards. It supports complex queries, transactions, and extensive data types, making it ideal for data-intensive applications.

6.3 CELERY

Celery is an open-source distributed task queue and scheduling tool used for handling asynchronous tasks. It integrates seamlessly with Django, enabling the offloading of time-consuming tasks like sending emails, data processing, or report generation to background workers.

6.4 REDIS

Redis is an in-memory data structure store commonly used as a caching layer, message broker, or database. It is widely used with Celery for task queue management and supports advanced data structures like strings, hashes, lists, and sets.

6.5 REACTJS

ReactJS is a JavaScript library for building dynamic, user-friendly front-end interfaces. Developed by Facebook, it is based on a component-driven architecture and enables developers to create reusable UI components, ensuring efficient rendering and seamless user interactions.

6.6 MXGRAPH

mxGraph is a JavaScript-based library for building interactive diagramming and graphing applications. It provides robust capabilities for creating, editing, and visualizing graphs, flowcharts, and diagrams in a web browser, making it ideal for applications requiring complex visualizations.

6.7 NGSPICE

ngSpice is a powerful circuit simulation tool based on the SPICE engine. It is widely used in electronic design automation (EDA) for simulating and analyzing circuit designs. Its integration into web or desktop applications aids in providing real-time feedback on circuit performance.

6.8 DOCKER AND DOCKER COMPOSE

Docker is a containerization platform that simplifies application deployment by packaging code and dependencies into lightweight, portable containers. Docker Compose further facilitates the orchestration of multi-container applications, enabling easy configuration and scaling of services like Django, PostgreSQL, and Redis.

6.9 NGINX

Nginx is a high-performance web server and reverse proxy server widely used for serving static content, load balancing, and managing API gateway functions. It complements Django applications by efficiently handling traffic, ensuring high availability, and improving scalability.

7. CONCLUSION

The internship offered a comprehensive and enriching learning experience, enabling me to make meaningful contributions to the eSim-Cloud project. By tackling critical issues, such as debugging existing functionalities, and implementing new features, I significantly improved the system's overall usability and functionality. This hands-on experience not only deepened my technical knowledge but also enhanced my problem-solving, collaboration, and project management skills. The expertise I gained during this period has greatly bolstered my proficiency in software development and serves as a valuable addition to my professional portfolio and resume.

REFERENCES

1. eSim-Cloud Project Documentation (https://esim-cloud.readthedocs.io/)
2. GitHub Repository for eSim-Cloud (https://github.com/frg-fossee/eSim-Cloud)
3. ReactJS Official Documentation (https://react.dev)
4. Django Official Documentation (https://docs.djangoproject.com)
5. Docker Official Documentation (https://docker.com)