Experience of Using OpenROAD Flow Scripts on a Specific Design

Kunal Kokate
ECE Department,
Vidyalankar Institute of Technology, Mumbai, India
kunalkokate03@gmail.com

***Abstract*— In this paper, I have demonstrated how OpenROAD Flow Scripts (ORFS) is useful in physical designs for specific projects. It also reflects my experience of ORFS installation and usage of the tools it uses for RTL to GDSII file generation and discusses a few limitations and recommendations for its scope of improvements ahead.**

Keywords*— *OpenROAD Flow Scripts, RTL-to-GDSII flow, open-source tools, automated design, no-human-in-the-loop.

I. Introduction

OpenROAD flow scripts is a flow controller supported by OpenROAD with a collection of tools required for design automation for a complete RTL to GDSII flow. The ORFS project aims to improve its algorithms for minimizing the area by reducing the wire length and increasing speed with low power consumption. In this paper, I described my experience and overview of using OpenROAD Flow Scripts, highlighting the benefits, challenges, and future recommendations.

II. Design Flow Overview

In physical design, 'Verilog' is used as a basic input method to do the design entries for any specific design project. Once ORFS is installed, we can specify the design parameters, like the technology node, the design constraints, and tool settings to meet the user's design projects. As an open-source tool, it gives users the option to make changes to the code for improvement. ORFS provides PDK support in the repository such as Nangate45, Sky130, Intel22, Intel16, and ASAP7 PDK to assist users with this.

Tools and dependencies required in the overall design, from synthesis to layout, can be installed directly via a single shell script command. However, there are some limitations I observed during the installation for certain tools or different versions of OS and the compatibility of the script with them.

III. Experience of Using ORFS

Only certain Linux scripts are needed to be run for the installation of ORFS and its tools. Any new user doesn't need to worry about manually installing anything other than the scripts given in the ORFS documentation, which makes it easier for beginners to get started with ORFS. However, there are many users who still struggled with the installation and had some errors on their machine even when using compatible operating systems as mentioned in the ORFS documentation.

Hence, it has scope for improvement in tool installations and OS version support for hassle free installation for all the users.

Apart from this, the source codes written for performing the tasks are in C++ and TCL. In ORFS, we just need to give inputs to the tool with the values for a certain design, and then ORFS will run the make file and other tools for the required design by itself, and then within 24 hours, depending on the operating system specifications of the user, it will generate the GDSII file. The higher the hardware specifications, the lower the run time. OpenROAD offers an interactive GUI which can be viewed by running 'make gui_final' command in linux. It helps to analyze the generated GDSII layout. The three major metrics that we need to look for are timing, power, and area.

IV. Conclusions and Recommendations

In conclusion, the ORFS, if used as an open-source tool, will be extremely useful for future designs without any human intervention, yielding speed and optimal outputs for physical design flow. Most importantly, it is also a time-saving, favorable, and cost-effective tool. As of now, the ORFS design supports three processor designs: riscv32i, ibex, and swerve_wrapper. In the future, I hope it becomes compatible with a larger number of processor designs and supports every other upcoming design so that everyone can use it without restriction. .

Even with a few limitations, ORFS is capable of running designs for 7nm technology nodes composed of FinFET-based compatible device models. And I truly believe that this open-source tool will be largely useful in the coming years in the domain of physical design.

References

[1] https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts
[2] https://openroad.readthedocs.io/en/latest/index.html
[3] https://openroad.readthedocs.io/en/latest/contrib/GettingInvolved.html#licensing-contributions
[4] https://github.com/The-OpenROAD-Project/asap5
[5] https://openroad.readthedocs.io/en/latest/main/README.html#getting-started-with-openroad

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE