Practical observations on OpenROAD Tool

Bhargavi Sri Teja A
ICT Dept.
University Of Hyderabad
Hyderabad, India
bhargavisriteja@gmail.com

Abstract

Through this paper, I want to share the observations on using OpenROAD tool. OpenROAD tool is an automated open-source tool to build the RTL-to-GDSII flow using OpenROAD Flow Scripts. Here, I explained the usage and observations of OpenROAD Flow Scripts.

Keywords— OpenROAD tool, open-source tool, observations, ORFS, Physical aware synthesis, MLMP, runtime, QoR.

I. INTRODUCTION

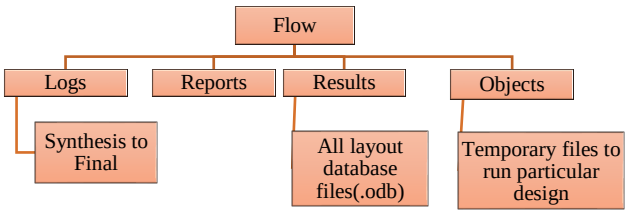
OpenROAD uses the OpenDB database and OpenSTA for static timing analysis. OpenROAD tool can be used by hardware designers, industry collaborators, enthusiasts, academics, and researchers. The execution of the tool is as follows 1. Setup the environment 1.a. By opting Build locally (i.e., installed Ubuntu22.04 LTS). 2. Executing the design flow. 3. Reviewing the data. Through this paper, we share our observations on the approach of ORFS to build the design.

II. RUN-THROUGH FLOW SCRIPTS

Build the openROAD tool (i.e., OpenROAD_flow_scripts) which consists of directories as flow, docs, etc, tools, jenkins. Once we build the ORFS check and verify whether all the required directories got created. Apart from this there few scripts which are important to check before we run. They are:

1. setup_env.sh file which source all the opensource tools to run from RTL-to-GDS II flow
2. makefile User can change the design config and other variables based on their requirement
3. config.mk this consists of design recipe information like platform, .v files, core_utilization, LEF, LIB files directory.
4. constraint.sdc it consists of timing like clock period, I/O port delays

Run make command. Once it executed successfully logs, reports, result files will be generated in flow directory.



III. CONCLUSIONS

ORFS has been easy to start with by following the tutorials and documentation. We got the positive results for few designs.

Based on using ORFS we recommend using 1 CPU Core and 16 GB RAM rather than 6GB RAM as the process is getting killed for complex design.

IV. SUGGESTIONS

In this tool design flow is logic aware synthesis followed by floorplan, place, cts, route and final. However, if we follow physical aware synthesis (i.e Consider physical parameters in the early phase of optimization of the design. Placed netlist within the setup timing margin) this helps in reducing the runtime and improves the Quality of Report (QoR).

For complex design which include pre-defined cells (i.e., macros) this tool uses hierarchical macro placer which runs hierarchy rtl.mp file by varying the variable values in metadata_base-ok.json. For better optimization we have to run n number of iterations which consumes more runtime. In place of this we can use MLMP (i.e., Machine Learning based Macro Placement) with few switches like mode-congestion based or timing based, style- Edge or middle or hybrid, effort- high or low or medium which in turn reduces the number iterations and overall runtime.

V. REFERENCES

1. https://openroad-flow-scripts.readthedocs.io/en/latest/tutorials
2. https://vsdiat.com/course_content?uniqueid=20230310071525
3. https://www.udemy.com/course/vsd-a-complete-guide-to-install-open-source-eda-tools/learn/lecture/15616198
4. https://www.udemy.com/course/vsd-a-complete-guide-to-install-openlane-and-sky130nm-pdk/learn/lecture/21989274#overview