Exploring the features of ORFS on Ibex at 7nm node

P Santosh Kumar Patra,
Physical Design Engineer,
santoshsonu36@gmail.com

Abstract- In this paper, I will explain my experience of using OpenROAD flow scripts (ORFS) on the ASAP7 platform. ORFS is a set of scripts and open-source tools used for RTL to GDS flow which supports nodes ranging from 180 to 7nm. We will focus on a specific design (ibex RISCV) and explore the capabilities of ORFS to increase the PPA of our design with customization of script or flow specific to the design. In the end, I will give recommendations on how this can be replicated in other projects.

Keywords: OpenROAD flow scripts, Physical design, Automated flow, NHIL.

I. Introduction

OpenROAD flow script is a set of scripts and tools for complete RTL to GDS flow. OpenROAD project aims to have an automated, no human-in-loop digital circuit design in 24 hours turnaround time. In this section, we will explain about components of ORFS and our experience while implementing an Ibex design in the Asap7 node.

A. Components of ORFS:

There are two main components in ORFS which makes IC design cheap and convenient, Tools and Flow scripts. Yosys(Synthesis tool with ABC support), Openroad(Floorplan, PnR, CTS tool), Klayout (GDS viewer tool) and Auto-tuner(hyper-parameter tuning to get optimal results) are the tools used in ORFS these tools are open-source and can be installed in any Linux machine making the cost of EDA tool free.

To work on these tools there are tcl scripts to complete the full RTL to GDS flow. These scripts take verilog files, SDC files, libraries, LEFs and other related files as input. These scripts make the flow automated, and require no human intervention in the loop.

B. Experience in using ORFS

My experience in using ORFS flow has been good overall. Similar to the flow, installing the tool on a local machine was also automated. I tried implementing the Ibex design on the Asap7 node, the runtime was approx 40min on my machine. I found that the scripts were in a common directory that is being used for all the designs for all scripts.

The Auto-tuner feature is very helpful in determining values of parameters in the design resulting in better PPA, thus reducing lots of manual time in experimentation.

II. Recommendation for future improvement

As discussed in the previous section common scripts are being used for all designs for various nodes, this means that all the designs are implemented in the almost same way. There are variables present in config files of the design that only control parameters like core utilization, placement density, etc.
I feel that with design-specific customization we can further improve the PPA of the design beyond what we are achieving through the general scripts of ORFS. After going through the flow I have identified some of the areas which can be targeted to increase the frequency of the design, they are listed below:

a. Adding IO constraint with design knowledge to fine-tune the placement of timing critical IOs.

b. Current PDN script for Asap7 is using M5 and M6 layers for stripes. Instead higher metal layers (M6 to M9) can be to free the M5 layer for signal and clock tree routing.

c. Timing recovery can be done during the initial floorplan stage with the use of the resynth_recover_timing variable.

d. At higher frequency with negative WNS we can tweak the setup and hold slack margins during the CTS stage to close our design with 0 WNS.

The above recommendations can be treated as an add-on to the existing flow. IC designers with knowledge of the RTL can adopt these recommendations on top of the ORFS flow for PPA improvement.

III. Conclusion

In this paper, I discussed the ORFS flow and my experience of using it to implement a RISCV Ibex design. The use of open-source tools and a full RTL to GDS flow scripts greatly reduces the cost and allows us to explore various aspects of physical design.
I have also discussed areas where we can target PPA improvement by adding design-based customization. Throughout the challenge, I will be trying these improvements and exploring more possibilities where there is a scope for PPA improvement. Finally, these solutions will be generalized to be used for other designs.

References

[1] https://openroad.readthedocs.io/en/latest/
[2] https://yosyshq.net/yosys/
[3] T. Ajayi, D. Blaauw, .et.al , “OpenROAD: Toward a Self-Driving, Open-Source Digital Layout Implementation Tool Chain", Proc. Government Microcircuit Applications and Critical Technology Conference, 2019, pp. 1105-1110.