Synthesis, Layout Generation, and Verification for Cardinal CMP Project Part 4

Assigned: April 17,2018
Due: April 30, 2018 11:59PM

Objective

The goal is to generate a layout for the Cardinal Chip Multiprocessor (CMP) using some of the CAD tools typically used in the creation of an ASIC design. While the time constraints of a one-semester class will not allow us to perform all steps, we will at least be performing synthesis, static timing analysis, gate-level simulation, logical equivalence checking, place & route, and back-annotated simulation.

Team Policies

Everyone should continue to work in the same teaming arrangement for the project.

What to do?

The following pages contain instructions for each step of the project. It is anticipated that you will be spending most of your time on the first step of simply obtaining a synthesizable design and optimizing the area-delay product of your synthesized design. For the synthesis portion of this work, use the synthesis environment you used for Homework 5 to develop a synthesized netlist targeting the NCSU 45nm standard cell library. Information about the library can be found at
http://www.eda.ncsu.edu/wiki/FreePDK45:Contents

What to submit?

(i) Gather up all the files requested in the Submission footers of each section of this assignment: synthesis/timing analysis/post-synthesis simulation, logical equivalency checking, place & route. Be sure to add all the files that are needed by cardinal_cmp.v (the top module).

(ii) "tar" the required files using the following command (tar_filename should include the names of both project partners => FirstName1_LastName1_FirstName2_LastName2)

```
viterbi-scf1 > tar cvf tar_filename.tar all_relevant_files
```

Be careful not to omit any required file. You may lose significant marks if a required file is omitted.

(iii) Submit the "tar" file using the uploading link of Project Part 4 for electronic submission.

Synthesis/Static Timing Analysis/Simulation

Logic Synthesis and timing

1. Modify the I/O port of your Cardinal CMP top module according to Figure 1. The new version removes the signals used for testing in Project Part 3.
2. Synthesize the Cardinal CMP design using the Synopsys Design Compiler. Use NCSU 45nm standard cell library as the target library.
3. Constrain the synthesis of your design using the appropriate optimization commands of Design Compiler. Your objective is to minimize the **product of area and delay** (clock period). The area is the total cell area including combinational and noncombinational area. The timing constraints are:
    - input delays = output delays = 1ns.
    - clock_latency = 0.5ns.
4. Generate the area, timing, and power reports (cardinal_cmp.area, cardinal_cmp.timing, cardinal_cmp.power). Analyze your design and determine the critical path (clearly describe it in the synthesis report). Iterate as necessary to further optimize the area-delay product of your design. In some cases, you may need to modify the RTL of your design to obtain a better synthesis result.
5. Use Synopsys "PrimeTime" to generate the static timing analysis report in the form of a histogram. You should use the same timing constraints that you used to synthesize the design. Submit the generated histogram in the synthesis report showing the details for the worst slack group, in the report.
6. At this stage you might want to go back and modify your logic design to achieve a better synthesis result, based on PrimeTime reports of the critical path. If your critical path is changed after this step, compare the final histogram with the first one in your report, indicating the result of your modifications. For RTL codes and scripts, you should submit the final versions only.

Post Synthesis Simulation

Plug in the synthesized module into your testbench for Project Part 3 (and please name your testbench tb_cardinal_cmp.v) and modify as necessary to work for your synthesized design. Use the standard delay format (sdf) file for back annotated simulations.

Synthesis Report

Include a synthesis report (syn_summary.doc/pdf) containing a table indicating your area- delay product (in units of μm^2*ns), area, delay and power consumption. Also report your critical path, generated histograms, any design optimization you made, and any part of this task you were unable to complete or special circumstances you encountered in this report.

Submission files:

- Synthesis report (syn_summary.doc/pdf)
- Final version of cardinal_cmp.v (RTL) and any other necessary RTL files
- tb_cardinal_cmp.v.
- cardinal_cmp.syn.v (synthesized netlist)
- cardinal_cmp.tcl (synthesis script)
- Area/timing/power report files generated in Design Compiler

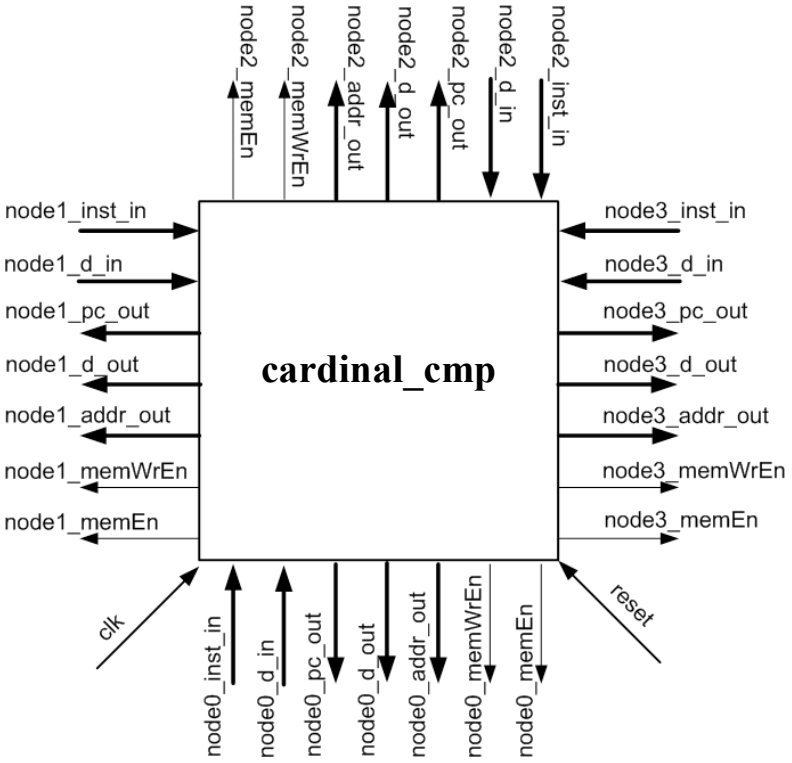

Figure 1: Top Level cardinal_cmp module.

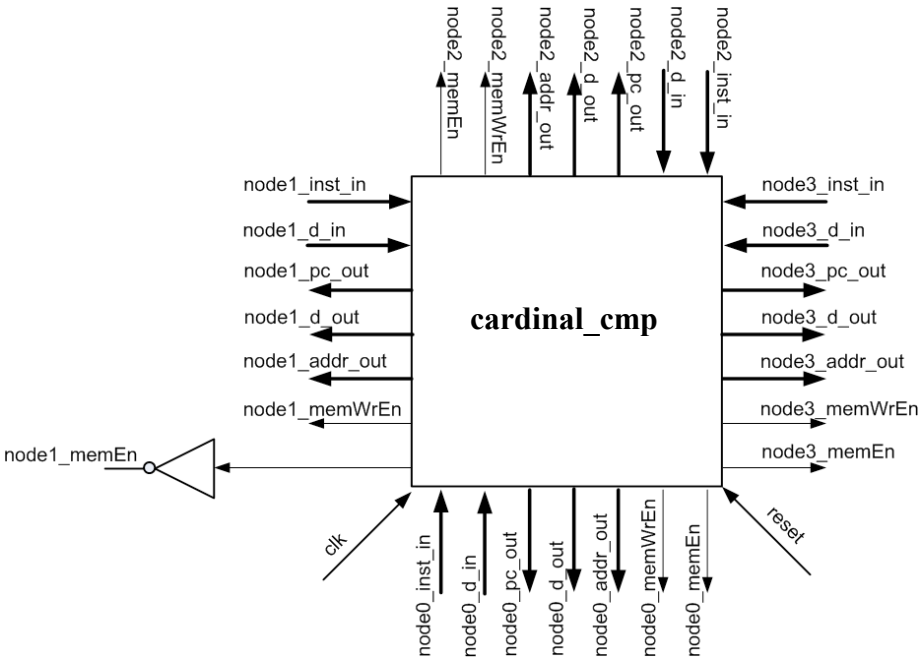

Figure 2: Erroneous Design for Logic Equivalence Verification

Logic Equivalence Verification Tool

To complete this part, we will be using the Cadence Conformal Logic Equivalence Checking to compare your RTL design (cardinal_cmp.v) and synthesized netlist (cardinal_cmp.syn.v). You may find a tutorial for using Cadence Conformal on the DEN website for EE 577B.

(1) Perform an equivalence check between the RTL (.v) and gate level netlist (.syn.v). Use the final RTL and resulting netlist from the synthesis step. Follow the instructions in the tutorial for your multiprocessor design and save the result as "lec_report1.txt". If you are unable to attain a positive equivalence result, describe the steps you took to debug your design. Create a file "lec_report1_summary.txt" that contains this description. If you attain a positive equivalence result, the file can be empty.

(2) To see what happens when designs are not equivalent, perform an experiment to manually edit a copied version of the netlist to force it to be erroneous. Edit the synthesized netlist to manually attach an inverter to the *node0_memEn* output. To add an inverter, you need to declare a wire, say *n240* (or some other name that doesn't already exist in the netlist), to replace the original *node0_memEn* output and move *node0_memEn* to the output of that inverter (see Figure 2). Now run Cadence Conformal with the RTL and this erroneous netlist. In this case, the tool will report a non-equivalent result. Save the result as "lec_report2.txt".

Submission:

- lec_report1.txt
- lec_report2.txt
- lec_report1_summary.txt

Automatic Place and Route

For place & route, we will be using Cadence SOC Innovus. Please refer to the Innovus tutorial posted on BB to learn how to work with the tool. Once you are comfortable with the tutorial, use Cadence SOC innovus to place and route the final synthesized netlist that resulted from the synthesis step (cardinal_cmp.syn.v).

Setup paths and timing constrains for Innovus

When you import your design, please insert the following path into ***Common Timing Libraries***:
/home/scf-22/ee577/NCSU45PDK/FreePDK45/osu_soc/lib/files/gscl45nm.tlf
and insert the following path into ***LEF Files***:
/home/scf-22/ee577/NCSU45PDK/FreePDK45/osu_soc/lib/files/gscl45nm.lef

For the Timing Constraint File, please see the mmc tutourial and example.

You may want to use a slower clock if there are timing violation or the place and route takes too long).

Perform pre-CTS timing analysis

Follow the steps in the tutorial until the standard cells have been placed. Go to *Timing->Analyzing Timing*, select *Pre-CTS* and *Setup* to generate preCTS setup time analysis report. Open your preCTS timing summary file (cardinal_ring_preCTS_all.summary). It is inside the subdirectory timingReports. Make sure there is zero "Violating Paths" shown in cardinal_ring_preCTS.summary. Copy the content of this file into ***pnrReport.txt***.

Perform post-CTS timing analysis

(optinal)

Continue the tutorial and do clock tree synthesis for your design. Generate the Post-CTS hold time analysis report. Open your postCTS hold timing summary file (cardinal_ring_postCTS_hold.summary). Copy the information inside your postCTS file into your ***pnrReport.txt***. Now open your clock tree synthesis report file (clock_report/clock.report). In this file, you can find many useful design data such as clock rise and fall delay, clock skew, etc. Copy lines 24-47 into your ***pnrReport.txt*** (Rise Phase Delay, Fall Phase Delay, Trig. Edge Skew, Rise Skew, Fall Skew, Max. Rise, Fall Buffer Tran, Max. Fall, Rise Sink Tran, Min. Rise, Fall Buffer Tran and Min. Rise, Fall Sink Tran along with Transition Time Vilations, etc.).

Go to Design->Report->Summary, and create a summaryReport.rpt text file and click ok. Open the summaryReport.rpt file and copy the lines that give Floorplan/Placement Information and Wire Length Distribution into your ***pnrReport.txt.***

Simulate the netlist with back annotation

(optional)

Type the command "saveNetlist cardinal_cmp.vo.v" at the Innovus console to save a Verilog file after place and route. Run your testbench from part 1 to test this Verilog file. In the simulation, use the sdf file generated by Encounter to back-annotate the netlist. Name the sdf file: cardinal_cmp_pnr.sdf. Name your testbench tb_cardinal_cmp_pnr.v. You may need to change the delay and clock period in the testbench based on the changes you make to your constraint file. Explain any testbench modifications you make in the ***pnrReport_summary.txt*** file.

Include information about any other anomalies or any part of the task you are not able to complete in your ***pnrReport_summary.txt*** file.

Submission:

- pnrReport.txt
- pnrReport_summary.txt
- cardinal_cmp.vo.v
- tb_cardinal_cmp_pnr.v
- cardinal_cmp_pnr_sdf
- cardinal_cmp_pnr.tcl