Advanced VLSI Project – stage 6

Table of contents

1. Reminder

So far, we have covered four stages:

- Synthesis stage – converting our RTL into logical cells
- STA stage – analyzing the timing constraints of our design.
- Floorplan – planning how we would want our chip to be constructed in a general view.
- Placement – place all the cells in a specific location (and some timing analysis practice).
- CTS – Generating a clock distribution network throughout the chip for synchronous operation of all components.

In the current stage (Routing) we want to connect to all wires in the design!

2. Setup

As we explained in previous stages of the project, we would like to copy the files for stage 6 from Elad's directory using the command:

```
cp -r /project/advvlsi/users/eladsimanian/ws/bitcoin/stage6 .
```

As explained in stage 3's document, you will work with a given NDM directories in this section too. For additional instruction on how to work with NDM directories please revisit the previous stage.

3. Stage 5: Routing

The Routing stage is a critical step in the process of designing a chip. After completing the Placement and CTS stages, the Routing stage involves connecting the pins of the placed cells to create a network of wires that interconnect the components of the design. This stage determines the timing, power, and area of the chip and plays a significant role in determining the overall performance of the chip.

The objective of the Routing stage is to optimize the routing of wires while meeting the specified design constraints, such as maximum wirelength and minimum via count.

In our project, we have completed the Placement and CTS stages for our bitcoin block, and now we need to perform the Routing stage to create the interconnects between the cells.

For your comfortable we marked all tasks needed for submission with:
[#P<Part_ID>_Q<Q_ID>] such as **[#P3.1_Q1]**

3.1 Warmup Questions

1. **[#P3.1_Q1]** What is the purpose of the Routing stage in physical design?
2. **[#P3.1_Q2]** What is the maze algorithm? How does it work?
3. **[#P3.1_Q3]** What is Cross talk? How can we minimize its effect?

3.2 Routing implementation

1. Open the NDM lib “bitcoin_stage_6” (see section “Working with NDM libraries” in previous project document for details).
2. This lib contains a similar snapshot of our design from the CTS stage.
3. The routing stage requires a lot of work when it comes to CPU calculations. Therefore, we will want to assign to each group 4 cores to work with for this stage.
   Run the command: “set_host_options -max_cores 4” (need to be set every time you open a new “Compile Fusion” window.
   Note: The micron servers have limited cpu cores and therefore you shouldn’t run your code with more than 4 cores.
4. Run the commands one by one in the given script.
5. Analyze the design:
   5.1. **[#P3.2_Q1]** How many Short and opens were found in our design?
   5.2. **[#P3.2_Q2]** What would be your recommendations for improving the current implementation?

4. Submission

1. If the team approves the design functionality, the team lead, which has the updated work area with all the teams' files working, should submit the assignment as follows.
2. Submit to Moodle a zip:
   Let's say you are group advvlsi_15 then:
   bitcoin_stage6_advvlsi_15.zip

   The zip must include a PDF file:
   bitcoin_stage6_advvlsi_15.pdf

   the pdf should include the following:
   2.1. The stage name and number (Stage 6 – Routing)
   2.2. Workarea path, such as: /project/advvlsi/users/$user/ws/bitcoin_project
   2.3. Team's info: members name, id, usernames (in micron servers), tau emails
   2.4. Answers for all questions and tasks marked with "[#P<>Q<>]"

Good Luck.

P.S the next stage (and last) will deal with Signoff!

Questions only

[#P3.1_Q1] What is the purpose of the Routing stage in physical design?

[#P3.1_Q2] What is the maze algorithm? How does it work?

[#P3.1_Q3] What is Cross talk? How can we minimize its effect?

[#P3.2_Q1] How many Short and opens were found in our design?

[#P3.2_Q2] What would be your recommendations for improving the current implementation?