ECE451: VLSI Systems and Design
Lab 5: Controller Place & Route
Due: March 30, 2015 (before lecture)

1 Introduction

In this lab an industry-standard automatic placement and routing tool **Cadence First Encounter (FE)** is used to create a layout of the 2901 control logic synthesized to a Verilog netlist in the previous lab. Since hand layout is expensive and time consuming, as you figured out yourself in previous labs, it is generally reserved for the most timing and area critical, high-performance portions of a circuit. The vast majority of designs nowadays are implemented with automatic Place and Route tools.

The synthesis result produced in Lab 4, `control_syn.v`, is used as the input file for the subsequent place and route procedures. **You should use the one generated from the "flattening" synthesis approach.**

2 Automated Technology

The software has two major goals to accomplish once given a netlist and cell library: optimal placement of the components, and establishment of electrical connections between the gates. Additional tasks, such as power routing, accurate clock tree generation, etc. are also generally possible.

First Encounter uses *Amoeba* placement from another company called "Silicon Perspectives". Amoeba, as described in the documentation, "retains a design's logic hierarchy within the physical domain, clustering hierarchical modules into amoeba-like shapes conforming to neighboring clusters." This key feature helps reduce the length of wires needed to connect the cells and thus helps to making the routing process easier. For the purpose of this lab, we will not do timing-driven placement and routing, but rather area and connectivity based placement and routing.

With the availability of more metal layers, and the continuing impetus to minimize die size, the trend in routing technology is shifting away from channel based approaches, in which the cells are spaced slightly to make a preferred routing area, to more 'over the cell' maze-like methods. With 4 or 5 layers of metal, the cells can be abutted and the connections can be implemented uniformly in the upper metal layers unconstrained by routing channels. While this reduces chip size, the burden of effort shifts to the automatic tool. Simultaneously, the number of gates per design is rising, meaning the total number of grid points considered by the router increases geometrically. First Encounter uses *NanoRoute*, which is a Graph-based router, to accommodate these trends and complete the route quickly with limited memory use. The graph-based routers view a design in a similar way to grid-based routers in that they can follow grids in both the vertical and horizontal directions. However, a graph-based router, taking a step from shape-based routers that route around shapes, considers these grids only as guidelines for routing and is not required to use them. Off-grid routing or variable metal pitch does not pose a problem for graph-based routers, and they can handle complex routing rules and off-grid connection points.

The cell library we use, `ece451_cells`, is a 0.35um library with 4 layers of metal. For the purpose of this lab, the metal 4 layer is disabled for routing, so only metal 1 to metal 3 are available.

3 Fabrication Issues

Although designs are mostly done using automatic tools these days, failing to take physical effects into account when using automated tools could lead to manufacture of a non-functional design. When using full custom techniques, the designer is in complete control of placement and layout, and therefore any delays subsequently introduced. Automatic place and route requires 1) definition of constraints in order to describe design criteria to the tools, 2) accurate characterization of the process targeted, and 3) extraction of RC parameters for re-simulation to confirm that the goals were indeed met.

Use of pre-designed cells incurs a strong dependency upon the accuracy of the library, and correct operation is predicated upon a close interaction between the library vendor and the target fabrication process. In hand layout, compliance with the design rules is your responsibility, but in cell-based design, correct operation of the final circuit hinges on how closely the library producers adhered to the designated technology.

Each library vendor develops and maintains a working relationship with a number of foundries. Synopsys, for example, has various different fab partners, each with their own process, and more to the point for us, a specific and distinct library for just that process.

For low volume design efforts, often a multi-chip wafer is produced. Mosis (`www.mosis.org`) is a typical prototype fabrication service. Take some time to visit this site and understand the multi-project concept. Mosis contracts for fab services, and evaluates all of the production facilities in order to both develop and ensure compliance with published design rules.

Design rules are developed cooperatively by both cell designers and the fabrication process engineers.

4 Control Logic Place and Route

As is often the case, some care must be exercised to choose the proper design format recognizable by software from different vendors. VHDL is common in academic research, university settings, and European commercial work. Verilog HDL is more prevalent in industry. It is not surprising then that the VHDL control logic file is not a form acceptable by FE, but Verilog is. We already have a synthesized 2901 controller in Verilog format (`control_syn.v`). Now, do the following steps:

1. Start the software by typing `encounter` on the command line. Do not append & after the command; otherwise Encounter will not be responsive. Instead, use an xterm window that you will not use for anything else.

2. Select *File* →*Import Design*.

3. Press the three dots beside *Verilog: Files*. In the pop-up window, press >> beside the *Add* button. Now, browse to find `control_syn.v` and press *Add*. Then browse to find `/cad2/ece451/verilog/ece451_cells.v` and press *Add*. Press *Close* to close the window.

   `control_syn.v` only contains the description of the controller in terms of the library cells. It does not contain any Verilog description of each individual library cell. Therefore, we need to include `ece451_cells.v` here.

4. Beside *Top Cell*, choose *By User* and write **control**, which is the name of the module in `control_syn.v` that we want to place and route.

5. Press the three dots beside *LEF Files* and add file `/cad2/ece451/se/ece451_cells.lef`. Press *Close*.

   A LEF file is a standard data format for P&R tools to represent the cell library. It contains sections that 1) describe the available routing layers and rules, and 2) describe the abstract layout of each library cell and the obstruction regions on various routing layers for that cell.

6. Select the *Advanced* tab, choose *Power* from the list and enter **VDD** in the *Power Nets* text box and **VSS** in the *Ground Nets* text box.

7. Press *Save ...* and give the input configuration a name (**control.conf**) and press *Save*. If you ever need to restart the process, just load the saved configuration file instead of repeating Steps 3 to 6.

8. Press *OK*. There should be no errors. A square with horizontal lines through it should appear in FE. Also select *Tools →Design Browser* to check the design that was imported.

9. Select *Floorplan →Specify Floorplan ...*
   Ratio (H/W) : **0.5**
   Core Utilization : **0.85**
   Core to IO Boundary: **selected**
   Core to Left, Top, Right, Bottom: **2.5**

   Select the *Advanced* Tab
   Double-back rows: select the second option where VDD/VSS are shared between adjacent rows

   Select *OK*.

   To place standard cells, they have to be organized into "rows". If you look through the LEF file, you will notice that each library cell has the same height (so VDD and VSS rails can match with the neighbouring cells). The *Floorplan →Specify Floorplan* option specifies the various parameters governing these rows. "Ratio (H/W)" is the ratio of the height to the width of the rectangle that our cells will occupy. The ratio 0.5 sets the Width to be twice the Height. "Core Utilization" specifies how much of the row area should be used. Setting the "double-back rows" will combine the VDD and VSS rails from 2 rows into 1. "Core to Left, Top, Right, Bottom" provides space from the region where the cells will be placed to the location of the input/output pins.

10. Load the IO Assignment file by selecting *File →Load →IO File*.
    File name: **ioplace.ioc** (available on web site)
    Select *Open*.

    The I/O assignment file specifies where the various pins of the control block should be aligned along the sides of the rectangle. Look into the file to see how this information is specified. The pins are aligned such that the control word enters from the left and various control signals exit at the bottom respect to the 2901 datapath logic. **You should change this to match the ordering of control signals in your datapath.**

11. Select *Place →Place Standard Cell*. Use the default values and press *OK*.

    Amoeba placement is invoked to do the placement of the cells. Observe the message log (at the command prompt) and you will notice that Amoeba place uses a "gravity" metric to allow different cells to be placed close together.

12. To check if placement completes successfully, select *Place →Check Placement* and press *OK*.

    The message log should show 0 unplaced cells. If not, you have to select all the blocks and press *Delete* and then go back and re-specify the floorplan, by changing the size, row utilization rate, or the aspect ratio.

At this point you should also make sure that the pins do not go off the right or top of the datapath. If they do, change the offset in the IO Assignment file, reload the assignment file and re-place the cells.

13. Select *Power →Connect Global Nets ...*

    In this window we will set the global connections. First we will combine all the VDD pins by setting:

    Connect: select the ***Pin*** button
    Instance Basename: *****
    Pin Name(s): **VDD**
    Scope: ***Apply All***
    To Global Net: **VDD**

    Press *Add to List*. Now do the same thing for VSS:

    Connect: select the ***Pin*** button
    Instance Basename: *****
    Pin Name(s): **VSS**
    Scope: ***Apply All***
    To Global Net: **VSS**

    Press *Add to List*. We will also have to make sure that all the cell pins that are set to logic high or logic low (such as reset or preset pins) are also connected to VDD/VSS.

    Connect: select the ***Tie Low*** button
    Instance Basename: *****
    Pin Name(s):
    Scope: ***Apply All***
    To Global Net: **VSS**

    Press *Add to List*.

    Connect: select the ***Tie High*** button
    Instance Basename: *****
    Pin Name(s):
    Scope: ***Apply All***
    To Global Net: **VDD**

    Press *Add to List*. Finally we will have to set one other pin. Net L[0] is connected to VDD (If you don not understand why, go back to Lab 2 and check all the possible values for L[0]), and so we will need to make sure it connects to VDD.

    Connect: select the ***Net Basename*** button
    Net Basename: **L[0]**
    Scope: ***Apply All***
    To Global Net: **VDD**

    Press *Add to List*. Now press *Apply* and then *Check*. There should be no messages in the log. Now press *Cancel*.

14. Select *Route →Special Route ....* Use the default values and press *OK*.

    Before applying this step, you may notice that the VDD and VSS rails between the different cells are not connected fully. This "Special Nets Route" step will connect them up.

15. Select *Route →NanoRoute →Route ....* Use the default values and press *OK*.

    NanoRoute replaces the older route. It performs all the necessary routing steps and gives you a completely routed design if it finishes successfully. Check the log to see if the "Total number of fails"

is 0. If there are failures, you have to select the whole layout, press *Delete*, and re-specify the floorplan and try again until there is enough space (try to keep it as small as possible though!) After this step you will see the metal1 to metal3 routing wires all over the design.

16. Select *Verify* →*Verify Connectivity* .... Unselect *Dangling Wire (Antenna)* and press *OK*.

    This step checks if all the connections have been made. A log will be generated (saved in `control.conn.rpt`), and it describes all the connectivity problems in the design. You can safely ignore the error "Special Wires: Pieces of the net are not connected together" as long as it is related to net VDD/VSS (see `control.conn.rpt`). This is because we will manually connect the different VDD/VSS rings later. If there are other errors, you should select the whole design, press *Delete* and go back to Step 9 and try different parameters.

17. Select *Verify* →*Verify Geometry* ... and press *OK*.

    This step ensures that the routing does not violate any geometric design rules, especially when routes have been made off the routing grid. It is acceptable to have *Short errors*. The tool may report some that are not really errors. If there are any other errors, you should select the whole design, press *Delete* and go back to Step 9 and try different parameters.

18. Select *File* →*Save Design* ... and choose a design name (**control.enc**) and press *OK*. Remember this name as it will be needed for Lab 6.

19. Generate a summary report by selecting *File* →*Report* →*Summary* ... and saving the report to a text file.

20. Select *File* →*Save* →*DEF* .... Name the file with the default values and press *OK*. Assuming you saved the session as *control*, a file `control.def` will be generated.

21. Exit First Encounter.

22. Copy `cds.lib` from the web page into your working directory.

23. Start Cadence by typing `startCds`.

24. Press F6. You should see the `ece451_cells` and `ece451_pads` libraries.

25. In the ICFB window, select *File* →*New* →*Library*.
    Name: **design**
    Check *Attached to an existing techfile*
    Press *OK*.
    Select "ece451_cells" for the *Technology Library* in the pop-up box.
    Press *OK*.

26. Select *File* →*Import* →*DEF*.

    Library name: **design**
    Cell name: **control**
    View name: **layout**
    DEF File name: **control.def**

    Press *OK*. The design is imported into the Cadence library "design" with cell "control" and view "layout".

27. Press F6. Open the layout view of the design, by selecting "design" in the left column and double-clicking on "layout" in the right column.

28. Currently the menus of the layout window are for Place and Route. Switch to the layout menus by selecting *Tools →Layout*.

29. You should see that all the cells are in their abstract views, but not layout views. We need to swap them:

    Press *NV* in the LSW window to hide all the unwanted layers.
    Click on the layout editor window and press Ctrl-R and Ctrl-F.
    Select *Edit →Search*.
    Search for **inst** in *current_cellview*.
    Click *Add Criteria*.
    Select *libname* == **ece451_cells**, press *Apply*, then *Select All* and *Cancel*.

    All cells that belong to `ece451_cells` library are selected. Now:

    Select *Edit →Properties*,
    Check *Common* box, and change *View* from *abstract* to *layout*, and press *OK*
    Press *AV* in LSW and Shift-F in layout editor window to see the cells in layout view.
    Select *Edit →Select →Select All*.
    Select *Edit →Hierarchy →Flatten*,
    Check on *displayed levels* and *Flatten Pcells*, then press *OK*.

30. Make sure that all layers in the LSW window are active (they should all be light grey). For those layers that are dark grey, right-click on them to make them active. Now use the mouse to drag a box around the complete design and select *Edit →Other →Convert to Polygon*.

31. Save the design by selecting *Design →Save* and close the layout editor window.

32. In the ICFB window, select *File →Export →Stream*:

    Library name: **design**
    Top cell name: **control**
    Output File: **control.gds**

    Press *OK*.

33. Exit Cadence.

34. Launch MAX and select *File →Import File*. Choose "Load GDS file using current mmi25 max technology file". Enter **control.gds** as the *File Name* and press *Done*.

35. Save the design as **control**. If you turn on DRC check in MAX, you will see many DRC errors. This is expected because the control block is generated based on a different process technology (0.35um) than the default one we have been using (0.25um). You can safely ignore these errors.

36. Manually add labels to the locations where there should be a pin for input/output signals. Follow the pin locations in your `ioplace.ioc` file. The easiest way to do this is to open your `control.enc` in Encounter and use this to determine the pin locations in MAX. All the pins have been distributed evenly on the left and bottom sides.

    Some outputs of the control circuit are directly connected to its inputs. You should avoid "double-labeling" such nets in MAX. For example, if WriteEn[i] is directly connected to notFBEn[i], you should label this net only once with either WritenEn[i] or notFBEn[i], not both. Similarly, the multiple clock signals generated (phi1_1, phi1_2, ...) should not be labeled as they are directly connected to the raw input clock phi1. Last, L[0] is directly connected to VDD, so it should not be labeled explicitly.

37. Label VDD and VSS rails with *vdd* and *gnd* respectively. Since we choose "double-back rows" in Step 9, the VDD and VSS rails alternate with the bottom rail being VDD. **Be careful that the VSS rail must be labeled *gnd* as opposed to *vss*!** Otherwise, your simulation of the combined control and datapath will not work, because IRSIM does not recognize *vss*.

38. Copy the MAX design for the 4-bit datapath from Lab 3 to the current directory.

39. Connect the control signals between the datapath and control blocks. Flatten the two components and remove any duplicate control signal labels.

40. Run *Local →Extract Netlist* to get file `top_lay.sim`.

41. Exit MAX.

42. Do IRSIM simulation for the entire processor, by specifying the two clocks and the input control words. **Use the four test cases of the addition operation you made in Lab 3.**

    You will launch IRSIM on `top_lay.sim` directly, by using the following command:

    ```
    ece451-irsim top_lay.sim
    ```

    It is highly recommended that you first simulate the control block alone, before simulating the top-level design.

4.1 Scripting

All actions you perform in Encounter are recorded in the form of Tcl commands in a file `encounter.cmd` in the directory where you launched the tool. Before using it as a script, rename it to something else (e.g., `control_enc.tcl`) because `encounter.cmd` may be overwritten by your future actions in Encounter. To execute this script, type `encounter -nowin -init control_enc.tcl` at the command line.

You may also create a script for the steps in Cadence, by copying the commands recorded in the ICFB main window to a file (e.g., `control_icfb.il`). To execute this script, type `load("control_icfb.il")` in the ICFB main window.

5 Submission

You will perform three different P&R runs on the control. Change the row utilization and other parameters (in Step 9) for each run. Compare the total area.

For one of the runs, you will combine the generated control layout with your custom datapath layout and perform IRSIM simulation. The simulation plot should show the input and outputs of the control, and all intermediate datapath signals specified in Lab 3 handout. Essentially, the plot should look the same as that you submitted for Lab 3, with one additional signal – the input control bus.

The submission includes both electronic and paper parts. **Do not forget to include the UG account on your cover page.**

5.1 Electronic submission

Submit the following for **one P&R run**:

1. `control.max`: layout of the control.
2. `top.max`: layout of the top-level design (i.e., control + datapath).
3. `top_lay.cmd`: IRSIM script for simulating the top-level design.

You will submit the above files with the following command:

```
submitece451s 5 control.max top.max top_lay.cmd
```

You can check your submission with the command `submitece451s -l 5`

5.2 Paper submission

1. A post-routing summary for **each P&R run**, that includes:
   - Input parameters used (in Step 9).
   - Area of the generated layout (Standard cells, Core and Chip) and the effective Core utilization. They can be found in the summary report generated by *File →Report →Summary*.
   - Log from *Verify →Verify Connectivity*.
   - Log from *Verify →Verify Geometry*.
2. The IRSIM plot as a result of running `top_lay.cmd` on the top-level design you submit electronically. Annotate the simulation as usual.