CS/EE 6710 Digital VLSI Design
CAD Assignment #6 (Group Assignment!)
Due Tuesday , October 17th, 5:00pm

Overview: In this assignment you will, as a group, add to the library that you started with the last assignment. In particular you will add a D-type master-slave flip-flop, a filler cell, and at least five other cells of your choosing. The result is that your standard cell library will include at least 12 cells (including the five from CAD 5). Follow the procedure from CAD5 to design and characterize your cells. This lab adds two new things to your cell library: a report from you on how you chose your additional cells, and a LEF file to go with your new, expanded, LIB file. The procedure for generating abstracts and LEF information are in Chapter 9 of the CAD manual.

Procedure: Follow the procedure from CAD 5 (CAD chapters 5 and 7) to add at least seven more cells to your standard cell library. The cells should be designed in all views necessary to be included in the library. In particular:

- cmos_sch view – transistor level schematics with transistor sizes
- layout view – following the standard cell template rules
- behavioral view – a view that has a Verilog behavioral description of your cell for behavioral simulation
- extracted and analog_extracted views - from the extraction/LVS process
- lib view – use SignalStorm or SpectreS to determine timing characteristics of your cells and include the new cell definitions in your library's .lib file
- verilog I/O view – include the I/O description in your library's .v file. This is generated in step3 of the SignalStorm flow, or you can do it manually.

New Views: In addition to these views, the process of producing abstract views with cad-abstract will generate:

- abstract view – This is generated by the cad-abstract program and contains a graphical view of the bounding box, connection, and blockage information that the place and route tool sees.
- LEF view – This is a text file, also generated by cad-abstract, that goes to the place and route tool. It's like a LIB file but it has place and route information instead of synthesis information. You should generate the cell macro

descriptions using cad-abstract, but use only one copy of the TechHeader.lef file that's in the /uusoc/facility/cad_common/local/Cadence/lef_files directory at the beginning of the file for technology information.

Note that you should generate these views with cad-abstract for all the cells in your library, including the five from CAD5. So, you might want to wait until you have your new cells designed before running cad-abstract (Chapter 9) on all of them.

The cells that you should add to your library are the following:

1. A positive edge-triggered D-type master-slave flip-flop with an active-low asynchronous clear. See CAD3 for details. You'll probably need to redo the layout from CAD3 to follow the standard cell template. You should include outputs for both Q and QBAR. Inputs are D, CLK, CLR.

2. A single-width filler cell. This cell is 2.4u wide and is nothing but vdd, gnd, and well layers. It provides the place and router with a way to fill gaps in the standard cell rows. The cell boundary should be 2.4u wide (one vertical routing grid wide), but the layers should overlap the official boundary by the amounts required in the template. This cell has no schematic view, and no simulation or lib view because it has no active devices in it. It does have layout, abstract, and lef views though.

3. Five more cells of your group's choice. You can decide what these cells are! In fact, how you choose should be documented as part of this lab. Think about what types of cells you might want to use if you were doing a design. Think about what special cells your project might need. Think about what type of cells might help the synthesis tool the most.

Procedure for Part 3: Please read: "Compact Yet High-Performance (CyHP) Library for Short Time to Market with New Technologies" by Nguyen Minh Duc and Takayasu Sakurai (linked to the class web site). This paper makes a case that a small (11 or 20 cells) library can perform almost as well as a 400 cell library when used with Synopsys design compiler and standard benchmarks. One way to try out different sets of cells is to make "fake" .lib files that have new cells listed there. As you probably figured out reading Chapter 7 on characterization, the .lib file is just an ASCII text file with some information in it about the cells. You could have made up this information from thin air and, after turning it into a .db file, Synopsys would happily use those "fake" cells (assuming you faked them with correct syntax). You could even use this technique to fiddle with the cells and see, for example, if Synopsys would be more likely to use a cell

if you could squeeze it a little smaller or make it a little faster.

In order to choose a good set of cells to add, you should perform a similar analysis to what Duc and Takayasu did: try out a few potential cells and see what you think. Create four behavioral Verilog files and try them out with different cells available in the synthesis target. Write behavioral Verilog for these four designs:

1. 2-input 4-bit-wide adder.
2. 8:1 mux.
3. 3:8 decoder.
4. 4-bit counter with a 4-bit limit register. Counts from zero to the limit.

Establish a baseline by creating structural verilog files (synthesizing with beh2str) for the four behavioral verilog targets. Find a figure of merit for the synthesis (Area is a good one. Speed is also good, but depends on "reasonable" speed estimates of your fake cells.). I would use the basic beh2str synthesis for this analysis just because it's easier to mess with the target libraries and requires less script-fiddling.

Now add a three input **NOR**and a three input **NAND** (these will be fictitious, but don't tell Synopsys) and see if the figure of merit improves.

Find out if Synopsys likes **OR** gates and **AND** gates.

Try and-or-invert gates :

```
Y = ~((A & B) | (C &D);   // four input   AOI4
Y = ~((A & B) | C);       // three input AOI3
```

Try or-and-invert gates:

```
Y = ~((A | B) & (C | D);   // four input OAI4
Y = ~((A | B) & C);        // three input OAI3
```

Try two complex cells

```
Y = A ^ B;                 // exclusive or
Y = (~C&A) | (C & B);      // 2:1 mux
```

Two years ago all libraries were required to include an **exclusive or** cell. Compare a library with **AOI4** against one with **exclusive or** in it. Fiddle with the areas (and other parameters) of these two (fictitious) cells to determine if an **exclusive or** gate should be required. Do the same comparison between the **2:1 mux** and the **AOI4.**

There are two example libraries (in .lib and .db format) that you can use as starting points or templates for your exploration. They are both in /uusoc/facility/cad_common/local/class/6710/synopsys/lib_files. The file named **foo** is the small example file from Chapter 7 of the CAD manual. The library named **osu05_stdcells** is a slightly larger cell library from Oklahoma State University. You can use these as starting points as you fiddle with the cells that are available in the libraries. Or you can write your own new cell descriptions from scratch. Think about "reasonable" sizes and speeds for any new cells that you consider. Of course, you'll only be using your best guess at this point. I'm not asking you to create, characterize, and throw away cells! I'm just asking you to fiddle with and fake some cells in the library and see how Synopsys reacts.

Please hand in a little report of your results. This report should list the cells you are including in your 12 cell library:

1. five basic cells that you already have
2. a flip-flop and a filler (cells the library must have)
3. five other cells that you will select. Justify the selection of these cells.

What to Turn In:

1. Documentation on all the new cells that you've added to your cell library: schematics, layouts, functional simulation
2. Final versions of your UofU_6170.lef, .lib, and .v files
3. Synopsys synthesis examples using your final 12-cell library on each of the four behavioral examples you used for deciding which cells to use
4. Your report on your methodology and conclusions on which cells to include. Build a simple model of how Synopsys chooses between **AOI4**, **2:1 mux** and **exclusive or**. Answer the questions, "Is **exclusive or** required?" "When does Synopsys prefer **exclusive or**?"