Jordan Bisasky
August 26, 2012

Tutorial for Synthesis and Encounter Place+Route

Setting up your account: (only need to do one time)

Edit .cshrc (only need to do one time)

1. Open up a Terminal in linuxserver1 (linuxserver1@cs.umbc.edu)
2. In your home directory (/home/csee1/[your username]), open .cshrc to edit using vi, emacs, xemacs, etc
    a. So others in the group can read/write/execute your files, make sure the umask is set to 002
    b. Someone near the bottom of the .cshrc file, add the following two lines:
        i. source ~/cshrc.cadence
        ii. source /afs/umbc.edu/software/cadence/etc/setup_2008/cshrc.cadence
    c. (As a reference, I also put a copy of my .cshrc in /data/eehpc0/software called cshrc.txt. You could copy that file to your home directory and rename it .cshrc instead of the preceding steps)

Add cshrc.cadence to your home directory (only need to do one time)

1. Go to /data/eehpc0/software and copy cshrc.cadence to your home directory

Run changes (only need to do one time)

1. In your home directory terminal, type:
    a. source .cshrc
2. Close all of your open terminals
3. If you can reopen them and you can access your directory and use linux commands in the terminal, it should have worked. If something bad happened, you'll probably have to send a help ticket to the CSEE IT dept for help

Jordan Bisasky
August 26, 2012

Set up directory for synthesis/layout

1. In a directory (preferably in your /data/eehpc0 folder), make a working directory (e.g. mkdir testDir)
2. Go into the directory and create three more directories; verilog, synth, layout
3. Move your .v Verilog files to the verilog directory

Jordan Bisasky
August 26, 2012

Set up and run synthesis script

1. I placed a sample script rtl.tcl in /data/eehpc0/software/encounterScript/synth
2. Copy the rtl.tcl to your synth directory
3. Open the script. I placed a "TODO" above every line you need to/should modify
    a. Path for your Verilog code, change the location between the curly braces { }, to your working directory,
    ex. set attribute hdl_search_path {/data/eehpc0/bisasky/testDir/verilog}
    b. Set myfiles - List all of your Verilog files between the square brackets [ ]. Do not include test files.
    c. Set basename – Add the name of your top level module. Or if you only have one Verilog module, name that module
    d. Set myclk – Leave as clk unless you use a different name for clock
    e. Set myPeriod_ps – Set the clock period. Leave at 1 GHz/1000 ps as a default. Can later increase/decrease that number to adjust the speed/power/area
    f. Additionally, there's "set_attribute lib_search_path {}" and "set_attribute library {}" which have the path and library names. They presently have 3 libraries/paths listed each. The first is library for the standard cells which will be used. The second and third and the libraries for SRAM modules which will not be used. I left them in case you need to use the SRAM modules in the future.
4. Run the script by typing in your synth folder:
    a. rc09 –f rtl.tcl
    b. Open the rc.log file and check for any errors (search for "error"). Warnings are generally ok.
    c. If it ran successfully, you should see many files added named after your basename.
    Ex. top.pow, top.sdc, top.cel, top_synthesize.v
5. Troubleshooting errors:
    a. Check the rc.log for errors.
    b. You're running the script in your synth folder and your files are in the verilog folder
    c. Make sure you modified the tcl script file correctly. i.e. the path to your code is correct, path to libraries and library names are correct, all of your Verilog files are listed, the basename is correct.
    d. If there are Verilog syntax errors, it will point them out to the screen and rc.log.
6. Files generated
    a. .area- approximate area breakdown for each module
    b. .cel - list of standard cells used and approximate area and area breakown
    c. .pow – approximate power consumption
    d. .sdc – timing file, used in Encounter layout
    e. _synthesize.v – RTL Verilog file to be used in Encounter layout
    f. .tim – Time to complete the critical path and the timing slack based on the clock period set in the rtl.tcl file. If there is a negative slack, then the clock period is set too low

7. Fine tuning – there is a balance between the clock period set in the rtl.tcl and the power/area. A lower clock period will increase the clock speed but also increase the area and power.

Jordan Bisasky
August 26, 2012

Set up and Run Encounter Place+Route

After each step, make sure to check the encounter.log file for errors in place and route. Keep in mind that the step may appear to complete successfully even if there were errors.

1. There are two sample files, top.conf and top.tcl, in /data/eehpc0/software. Copy those two files to your layout directory
2. Top.conf is the initial script that loads the necessary files to Encounter
    a. Set rda_Input(ui_topcell) "top" - set the topcell to the top module in your design (do not remove the quotations " ")
    b. Everything else in top.conf can remain the same as long as you're using the same working directory set up described in the aforementioned sections
    c. **NOTE: The .tcl and .conf should be changed to the name of your top module. top.conf and top.tcl assumes your top module name is "top"**
3. Open top.tcl in layout. Again I placed TODO markers in the file. We will be running the script in parts so we can modify the top.tcl in parallel to running Encounter
4. Go to your layout directory, type:
    a. encounter
    b. Wait for Encounter to load. An Encounter GUI window should open up. If it fails to load, make sure you set up your account correctly
    c. You can use either the GUI or the command line in the terminal. We will be using the command line but we can use the GUI to visually check for errors and look at our design
5. Load design
    a. In top.tcl (line 10) - set DesignName – add the top level module in your design
    b. If you go to approximately line 81, you will see an IF statement with STEP's inside. We will set the constant STEP to the string. If we do "load", it will only run what is inside of that IF statement. If we do "one", it will run all of the IF statements with "one" as an argument (load, floor plan, power plan, pre place, pre time, pre power). Finally, "big" will run all of the commands in the script.
    c. To load the top.conf, terminal – type:
        i. set STEP load
        ii. source top.tcl
    d. You should see a rectangle in the GUI window if it loaded correctly.
    e. **TROUBLESHOOTING**
        i. Did the synthesis run correctly without error?
        ii. You modified the top.conf file as specified in step 2a
6. Load floor plan
    a. In the GUI, go to Floorplan->Specify Floorplan
        i. You can adjust the height/width in the menu (prefer to keep the height and width equal) and then hit apply

ii. It will show the core utilization percentage. Ideally we want that as close to 100% as possible. However probably adjust the height/width for now so the utilization is around 90%

b. In top.tcl (line 42), set corewidth/height –Set the width/height to the values you found from the GUI

c. Type in the terminal
   i. set STEP fp
   ii. source top.tcl

d. **TROUBLESHOOTING**
   i. If you preplaced standard cells (ex. SRAM), they fit inside the floorplan width/height

7. Load power plan
   a. Looking at your design, determine where you want your vertical/horizontal VDD/GND rails.
   b. In top.tcl (line 176), go to vertical stripes and horizontal stripes
      i. For vertical stripes (metal 6):
         1. start_x [where you want first vertical stripe relative to x axis] – presently it's set to 135 where 0 on the x-axis is the left edge
         2. set_to_set_distance [distance between the next vertical rail] – I have it set to 400 (much wider than the design) so there is only one rail
         3. number_of_sets [number of vertical rails]
         4. You can copy/paste this code again if you want to more precisely place down many vertical stripes
      ii. For horizontal stripes (metal 5):
         1. Same thing, adjust start_y, set_to_set_distance, and number_of_sets. The # of sets field doesn't seem to be included in this stripe but you can add the field to the line
         2. I have the horizontal stripe commented out since it requires some additional work to prevent place and route from causing design errors when you have both horizontal and vertical stripes
      iii. For both cases, this is one line of code, the '\' character goes to the next line but Encounter reads it as a single line command
      iv. CreateObstruct – Place a rectangle where a standard cell cannot be placed. This is useful to prevent issues from having both a horizontal and vertical rail. Right now the lines are commented out but to place one, you specify the (x,y) of the lower left hand corner of the rectangle and the (x,y) of the upper right hand corner to form the coordinates for a rectangle
   c. Type:
      i. set STEP powerplan
      ii. source top.tcl
   d. **TROUBLESHOOTING**
      i. VDD/GND rails fit within the dimensions of the design

ii. VDD/GND rails do not overlap each other

8. To finish the rest of step one, set the step to “preplace” and run the tcl script. Then set to “pretime” and run the script. Then “prepower” and run the script.
    a. **In top.tcl (line 407) under prepower, set the speed and gate switching**
        i. {clk 1170 0.2} refers to a clock frequency of 1170 MHz with 20% of gates switching every cycle
        ii. This affects the power consumption calculated, the speed and activity factor
    b. **TROUBLESHOOTING**
        i. There could be errors from a previous step causing issues
        ii. Looking at the GUI is helpful to visually locate the errors. They are marked with a white X
        iii. **Lots of white X’s on the design** - The core width/height may be too small. Usually the floor plan utilization underestimates the percentage used, so that’s why I usually start at 90% and gradually decrease the height/width. Keep in mind that if you use other libraries (ex. SRAM), the floor plan utilization will not be useful at all since it will not estimate the area occupied by other cells correctly
        iv. **A few white X’s on the design** - If you still get errors, there are issues with the VDD/GND rails or placement of blocks from other libraries. Encounter may try to place a standard cell where you do not want it to be. You may need to adjust the placement of the rails/library blocks and/or use the createObstruct command
9. Then set to step “two” and run the script, then step “three”, and then step “four”
    a. **TROUBLESHOOTING**
        i. Same as the previous section. Make sure there are no errors from previous steps. Clock tree synthesis is performed in step three which adds additional routing. This step could result in errors from the width/height being too small

When you are more confident the design will place and route correctly, you can start using “one” instead of “load”, “fp”, etc.

Instead of having to run steps over and over, it will save also your design in .enc files which can be loaded. This saves us from having to rerun the earlier steps over and over again. Go to File -> Restore Design and select the .enc file. You can go to the top.tcl script and search for “saveDesign” to see where this is performed.

Jordan Bisasky
August 26, 2012

Post Power Analysis

The area is specified by the floorplan width/height provided and the speed was set in the rtl.tcl script in the synth folder. The power numbers are calculated based on speed and area specificiations.

In the layout/postpower folder that is created, the resultant power and area numbers are given. Skimming through the files provides the power numbers.