**Lab 4 : ECE 5470/6470**

**Routing Using Cadence Innovus**

**Due March 23, 11:59 PM**

In this problem, you will be doing global and detailed routing for an already placed design. The sign-off routing tool ‘nanoroute’ is used for this purpose. This tool is integrated with Cadence Innovus.

1. **Global Route**

During this phase, the NanoRoute router breaks the routing portion of the design into rectangles called global routing cells (gcells) and assigns the signal nets to the gcells. The global router attempts to find the shortest path through the gcells, but it does not make actual connections or assign nets to specific tracks within the gcells. It tries to avoid assigning more nets to a gcell than the tracks can accommodate. The detailed router uses the global routing paths as a routing plan. The command for doing global routing on a placed design is **globalRoute.** You will notice that Gcell Congestion is reported in 'Congestion Analysis' on standard output. If there are any over-congested gcells, it reports number/percentage of such gcells on each metal layer. The first column, Layer, lists the metal layers that have over-congested gcells. The NanoRoute router marks a gcell as over-congested if the global router has assigned more nets to the gcell than the available tracks of the gcell. The columns, labeled OverCon #Gcell, list the number and percentage of gcells on each layer that are over-congested. The numbers in parentheses after OverCon #Gcell indicate how many additional tracks within the gcell are needed to accommodate the global routing assignments. For example, OverCon #Gcell (1-2) means that one or two additional tracks are needed to accommodate all the nets that the global router has assigned the gcells listed in the column.

1. **Detail Route**

During this phase, the NanoRoute router follows the global routing plan and lays down actual wires that connect the pins to their corresponding nets. The detailed router creates shorts or spacing violations rather than leave unconnected nets. The router runs search-and-repair routing during detailed routing. During search and repair, it locates shorts and spacing violations and reroutes the affected areas to eliminate as many of the violations as possible. The primary goal of detailed routing is to complete all of the required interconnects without leaving shorts or spacing violations. During detailed routing, the router divides the chip into areas called switch boxes (SBoxes), which align with the gcell boundaries. The SBoxes can be expressed in terms of gcells; for example, a 5x5 SBox is an SBox that encompasses 25 gcells. The SBoxes overlap with each other, and their size and amount of overlap might vary during search-and-repair iterations. The router also runs postroute optimization as part of detailed routing. During postroute optimization, it runs more rigorous search and repair steps. Detailed routing stops

automatically if it cannot make further progress on routing the design.

**The routing directory contains the following**:

* **s35932**: This is the design directory and contains all the input design data needed for placement of the design. The files present in this folder are s35932.conf, s35932.sdc and s35932\_post\_syn.v. These were explained in the placement pdf previously.
* **outputDir**: This is the outputDir generated during the placement stage. It contains to files: placed.def and s35932.palced.enc and one folder s35932.placed.enc.dat
* **s35932.placed.enc.dat**: This is the folder inside outputDir after placement is done.
* **s35932.placed.enc**: This is the file inside outputDir after placement is done.
* **env.tcl**: This file sets up the location parameters (which basically are tcl variables). The location parameters to be set are designDir, libDir, outDir. In short, it specifies the location of the input design data, input libraries required, timing constraints imposed and the directory to dump the intermediate results.

Commands in env.tcl are as follows:

1. ***set designDir ./s35932***: Set the following parameter to point to the location where the design data is located. Design data includes gate level netlist (.v), design floorplan (.fp and .fp.spr), design configuration file (.conf file), timing constraint file (.sdc).
2. ***set libDir /opt/software/cadence/library***: Set the following variable to point to the location where the libraries are located. These include one timing library (.lib) and two lef files(.lef).
3. ***set outDir ./outputDir***: Set the following variable to point to the directory where you want to store the intermediate results. The intermediate results include the design saved at various stages of the flow. e.g You may want to save the design after placement or routing or clock tree synthesis or in between while doing timing optimizations.
   * **route.tcl**: This is the basic script file containing the commands needed for routing, which is to run with the innovus. The commands within the route.tcl are as follows:
4. ***source env.tcl***: This command sources the env.tcl file. Thus the commands in this file will be executed.
5. ***restoreDesign s35932.placed.enc.dat s35932\_bench***: This command is used to load the saved design files of a previous design session from the specified s35932.placed.enc.dat directory.
6. ***optDesign -preCTS -outDir prectsOptTimngReports:*** This command is used to perform timing optimization before clock tree synthesis and generates timing reports which are stored in the directory prectsOptTimingReports.
7. ***create\_ccopt\_clock\_tree\_spec***: This command creates a clock tree network with the associated groups and the clock tree synthesis configuration.
8. ***ccopt\_design***: This command performs clock concurrent optimization on the loaded design. The command optimizes both the clock tree and the datapath to meet the global timing constraints.
9. ***timeDesign -postCTS -outDir postctsTimingReports***: This command runs Trial Route, extraction and timing analysis and generates detailed timing reports after clock tree synthesis. The generated timing reports are saved in ./timingReports directory or the directory you specify using the -o*utDir* parameter ( in this case postCTSTimingReports).
10. ***optDesign -postCTS -outDir postctsOptTimngReports***: This command is used to perform timing optimization after clock tree synthesis and generates timing reports which are stored in the directory postctsOptTimingReports.
11. ***globalRoute***: This command plans the interconnect by breaking the routing portion of the design into rectangles called global routing cells (gcells) and assigning the signal nets to the gcells.
12. ***setNanoRouteMode -routeWithTimingDriven true***: This command is used to minimize timing violations by analyzing the timing slack for each path, the drive strengths of each cell in the library and the maximum capacitance and maximum transition limits.
13. ***detailRoute***: This command uses the NanoRoute router to perform routing on the entire design.
14. ***SetAnalysisMode -analysisType onChipVariation***: This command sets the specified analysis mode for the timing analysis.
15. ***timeDesign -postRoute -outDir postrouteTimingReports***: This command runs Trial Route, extraction and timing analysis and generates detailed timing reports after routing. The generated timing reports are saved in ./timingReports directory or the directory you specify using the -o*utDir* parameter ( in this case postrouteTimingReports).
16. ***optDesign -postRoute -outDir postrouteOptTimingReports***: This command is used to perform timing optimization after routing and generates timing reports which are stored in the directory postrouteOptTimingReports.
17. ***verify\_drc***: This command checks width, spacing and internal geometry of objects and geometry between them. It also creates and saves violation markers in the design database.
18. ***verifyConnectivity***: This command detects conditions such as opens, unconnected wires, unconnected pins, loops, partial routing and unrouted nets. It also generates violation markers in the design window and thus reports violation.
19. ***report\_power -o outputPower***: This command report the power for different logical blocks. The report is stored in the directory outputPower.

The **route.tcl** script also contains commented commands at the end which can be used to fix the violations.

**The following commands have to be executed in order to do routing:**

1. Change the present working directory to the directory where your design is stored.
2. You should be able to access the ***s35932*** directory and outputDir (from lab3), one easy way is to copy these to folders to your lab4 working directory. Also copy env.tcl.
3. Open the Innovus using the command: ‘***innovus***’.
4. Make sure that the ***s35932.placed.enc.dat directory*** is in your working directory. Copy the ***s35932.placed.enc.dat*** directory from the output directory to your work directory. Or create a symbolic link (ln -s ./outputDir/s35932.placed.enc.dat).
5. Once the innovus opens, run the routing file using the file ‘***source route.tcl***’. You can also run each command from **route.tcl** one by one.
6. Notice the violations that occur during routing. These violations can be fixed in a variety of ways. One method is using the commands from the commented section in the ***route.tcl*** file (line 41). Other methods include increasing the timing delay/slack during synthesis, allowing a larger floorplan area, among others.

**Report the following:**

1. Total wire-length and via of global route results.
2. Total wire-length and via of detailed route results.
3. Slack.
4. Snapshot of the routed map.

***Note1****: Total wire length and vias for global route can be found at the end of the* globalRoute *command execution.*

*Alternatively, in case the tcl script has been used, then the most recent* ***innovus.log*** *file can be used. Search for “****Complete Global Routing****” in the most recent* ***innovus.log*** *file. The details for wire-length and vias can be found in the lines after the line containing the text “****Complete Global Routing****”.*

*Similarly Total wire length and vias can be found for the detailed route.*

***Note2****: Slack can be found at the end of the* optDesign -postRoute -outDir postrouteOptTimingReports*command execution under the heading of the table*

***“optDesign Final SI Timing Summary”****.*

*Alternatively, in case the tcl script has been used, then look for the same heading in the most recent* ***innovus.log*** *file.*