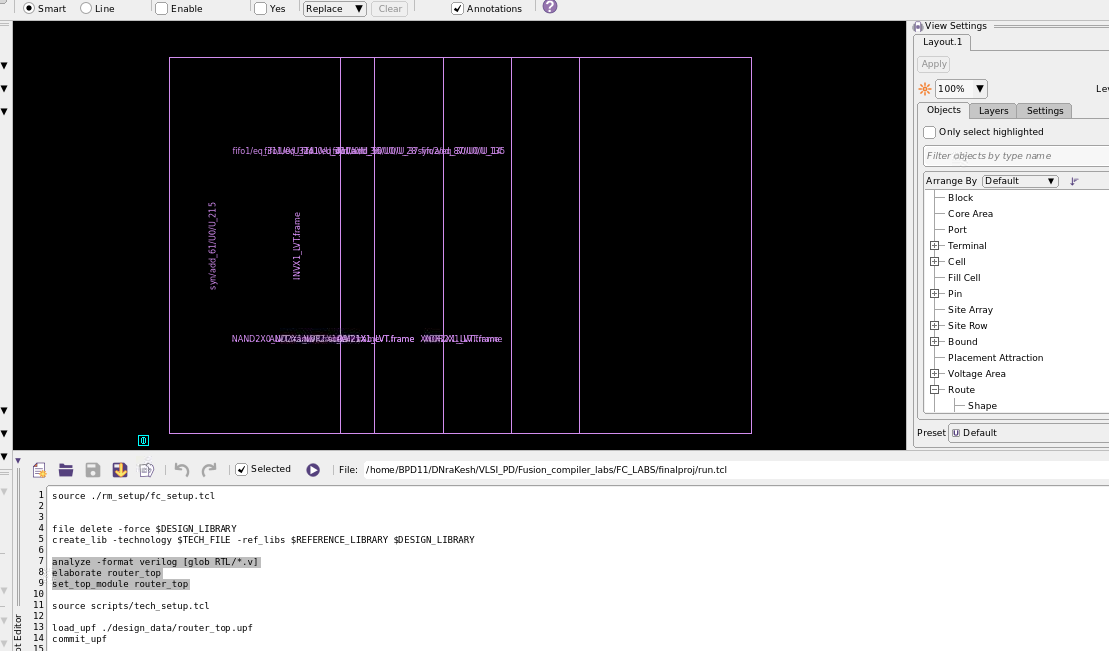
**Router3x1 project report**

Task1:: Create the Design Library, Read the RTL



Analyze the Verilog RTL files: analyze -format verilog [glob risc\_rtl1/\*.v] • Note: The glob command used above returns a list of all files that match the pattern. ⎫ Create a linked “top” block: elaborate risc\_core set\_top\_module risc\_core

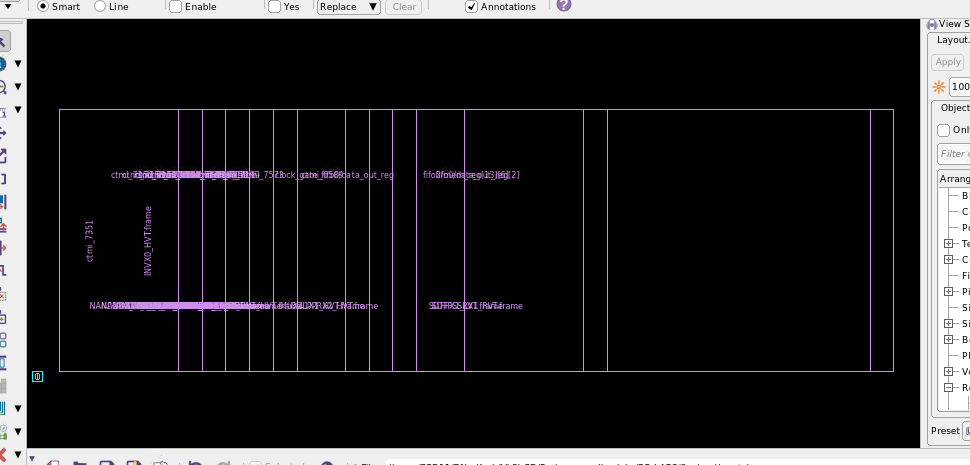


set\_auto\_floorplan\_constraints -core\_utilization 0.7 -side\_ratio {1 2} -core\_offset 10

floore plane is set using above command

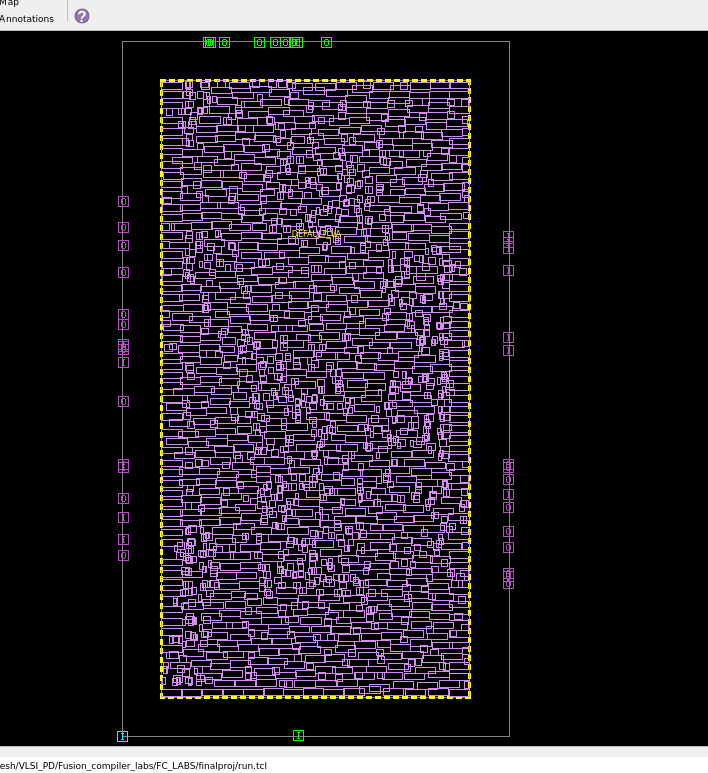
Task2: Run compile\_fusion initial\_map

Run stage 1 of compile, initial\_map:



Task 3: Run compile\_fusion logic\_opto

Run stage 2 of compile, logic\_opto:\_run\_compile\_stage\_save logic\_opto ,You should see that the floorplan has been created automatically, pins are placed along all 4 edges, and a first coarse placement has taken place (zoom in to examine). Timing-driven logic optimization occurs before and after placement. Here is a screenshot of the design after stage

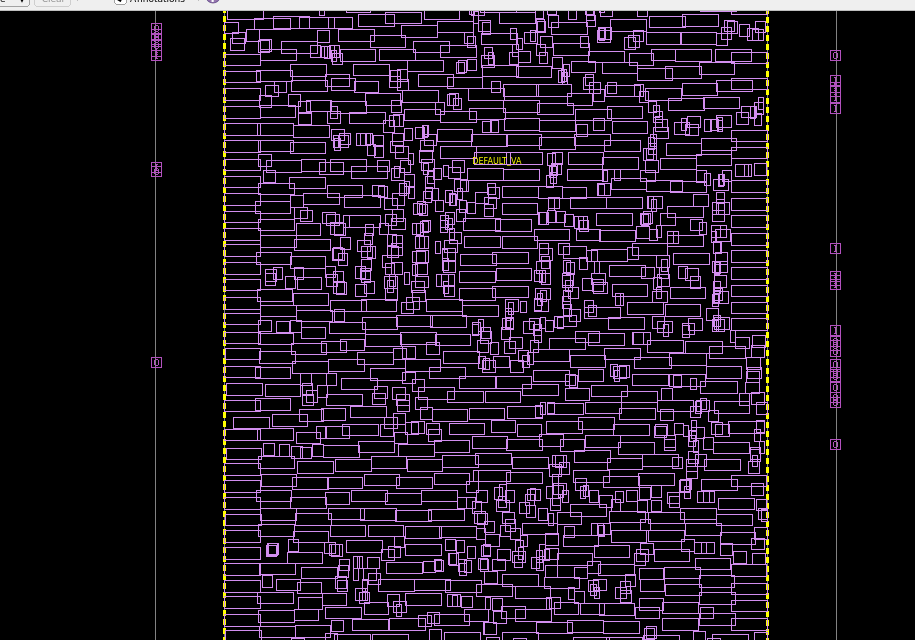


Task 4: Run compile\_fusion initial\_place

Run stage 3 of compile, initial\_place:\_run\_compile\_stage\_save initial\_place

The placement performed here is buffering-aware timing-driven placement(confirm by searching for the second “Placement Options” table in the log file (compile \_fusion.initial\_place.log).

Note that the placement is still coarse, the standard cells are not legalized.



Task5::Run compile\_fusion initial\_opto

Run stage 5 of compile, initial\_opto:\_run\_compile\_stage\_save initial\_opto

This stage performs many optimization steps, including CCD, scan insertion, etc.

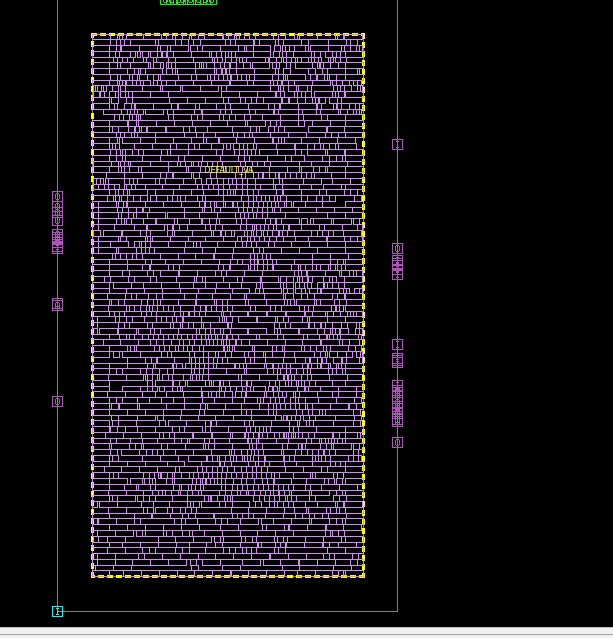
Zoom into the layout and you will see that placement seems to be fully legalized -standard cells are placed inside the placement rows and are not overlapping. Run the following command to check whether all cells are indeed placed legally:check\_legality



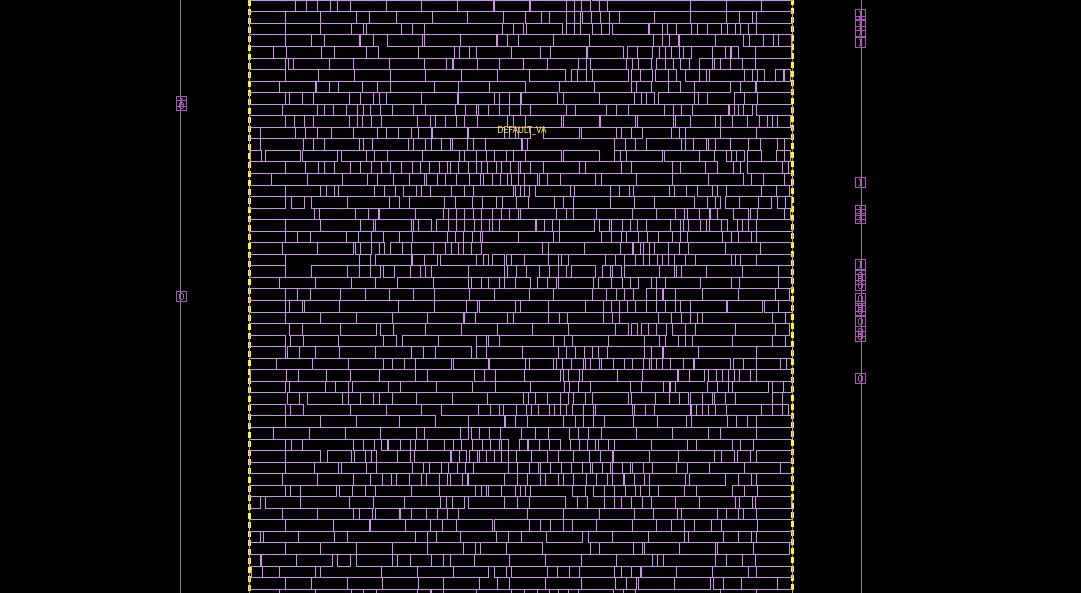
Task 6: Run compile\_fusion final\_place Run stage 6 of compile, final\_place:\_run\_compile\_stage\_save final\_place

Check placement legality again:check\_legality

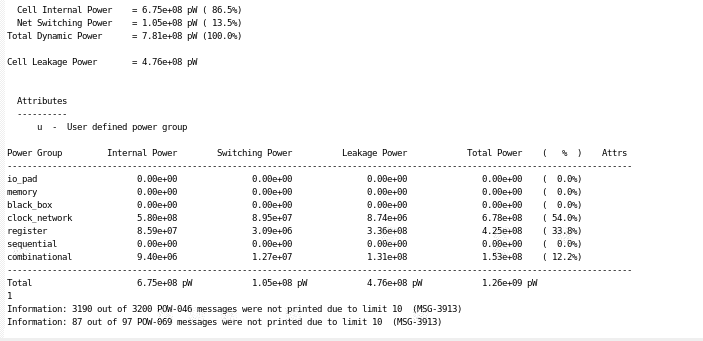
This time cell placement is fully legal. This is not the last stage though, more optimizations and incremental placement will occur during the 7th compile stage, after which final legalization takes place



* Run compile\_fusion final\_opt



Power report



Multi-Corner Multi-Mode Setup ⎫

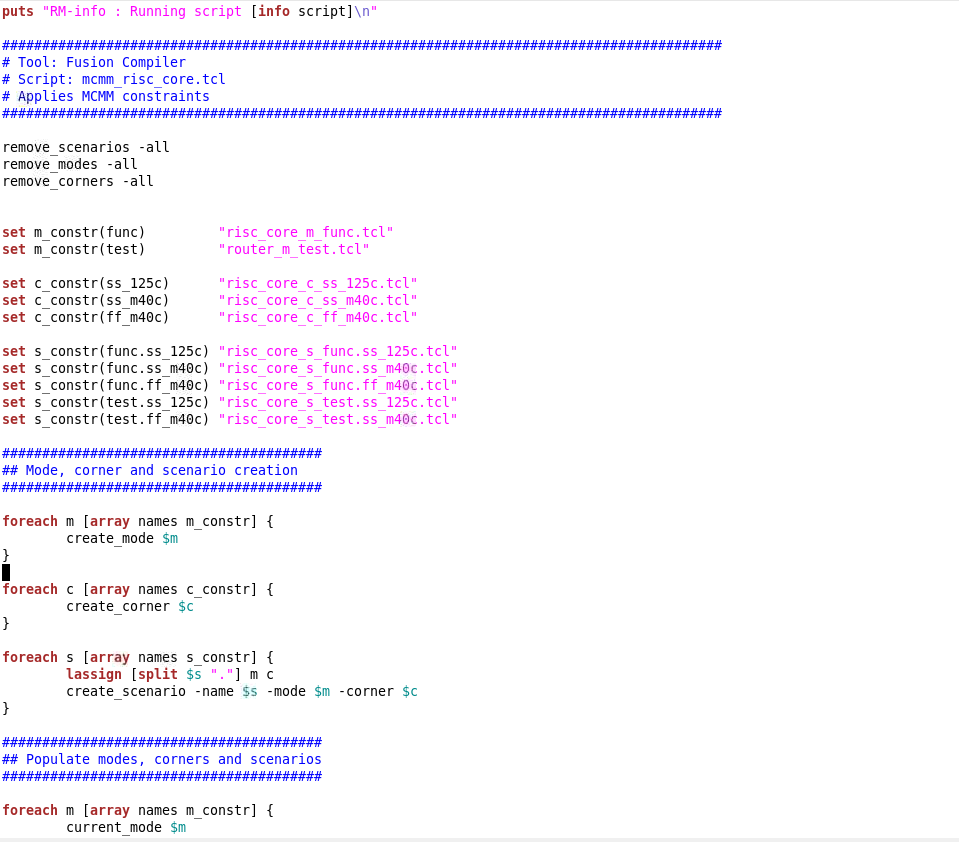
In a different terminal window, open the file: Design\_data/mcmm\_risc\_core.tcl ⎫

This script first creates the modes, corners, and scenarios needed for multi-corner multi-mode (MCMM) optimization of this design.

• Note: The script takes advantage of Tcl arrays (set array Name (var Name)) value) to create efficiently-coded for each loop.

Mcmm is loaded using below command

\*\*\*\*\*source scripts/mcmm/mcmm\_router.tcl

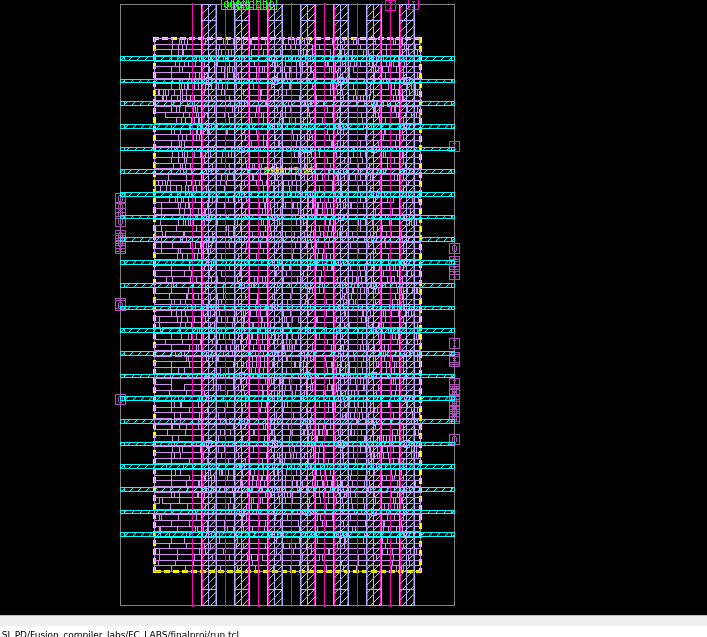


Power and Ground (PG) Prototyping

PG prototyping can help you create a basic PG mesh very quickly. All you need to specify are the PG net names, the layers, and the percentage of the layers you want to use for PG routing. This is most commonly used as a placeholder for the final power mesh, to check congestion issues that a PG mesh may cause. ⎫ From the Tasks palette, select PG Planning PG Prototyping

PG is source using below command

source scripts/pns.tcl

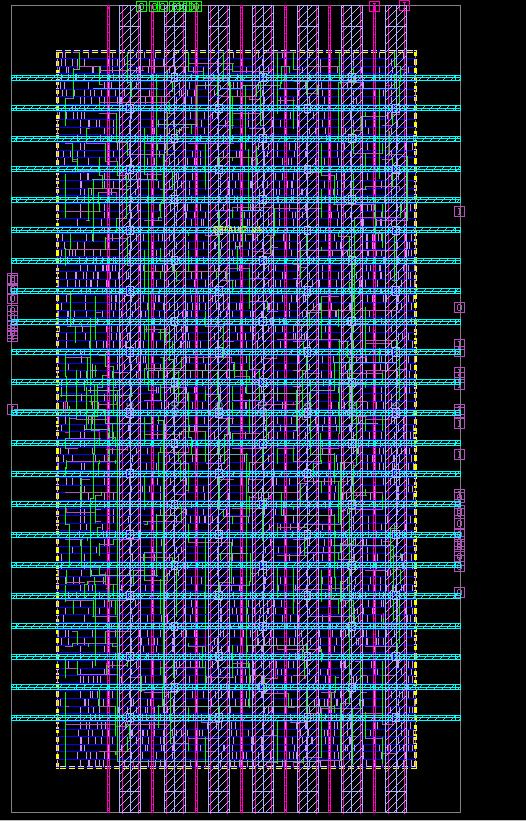


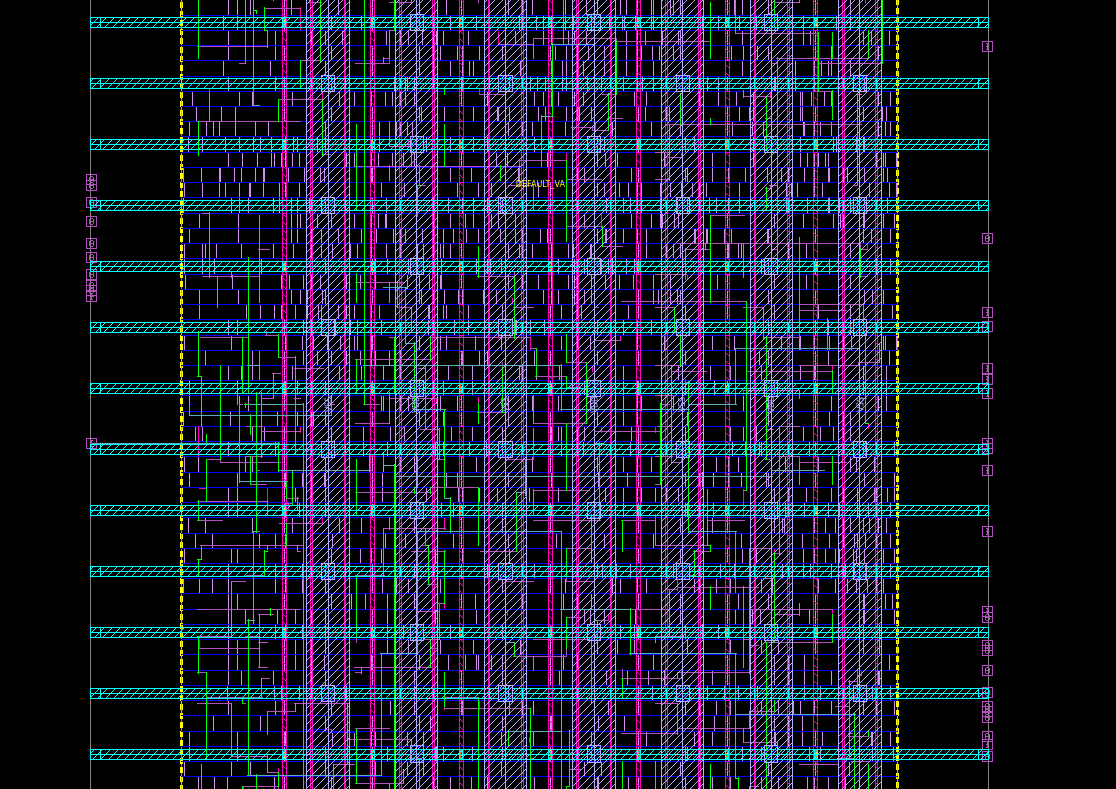
## CTS

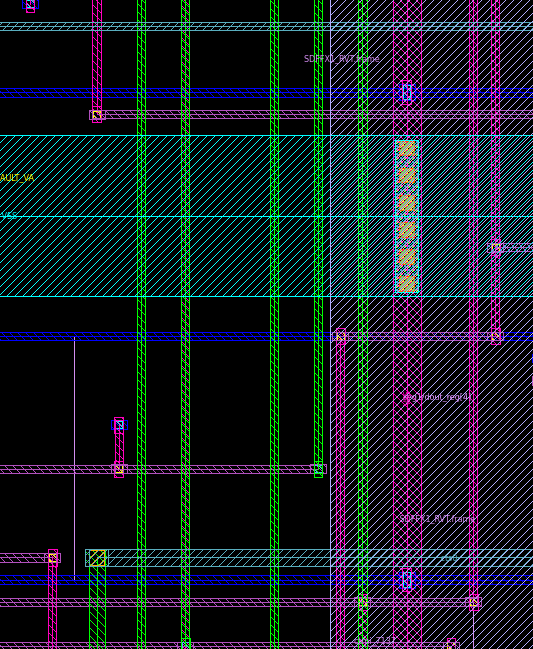
Define CTS Non-Default Routing Rules

we will specify CTS non-default routing rules, as well as clock cell spacing rules. ⎫ Open the file scripts/ndr.tcl in an editor

the green color in below image is clock tree



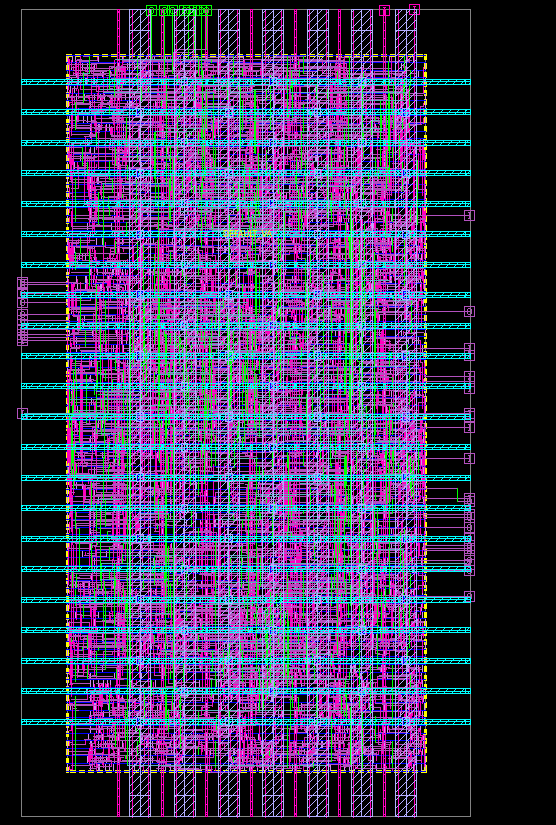


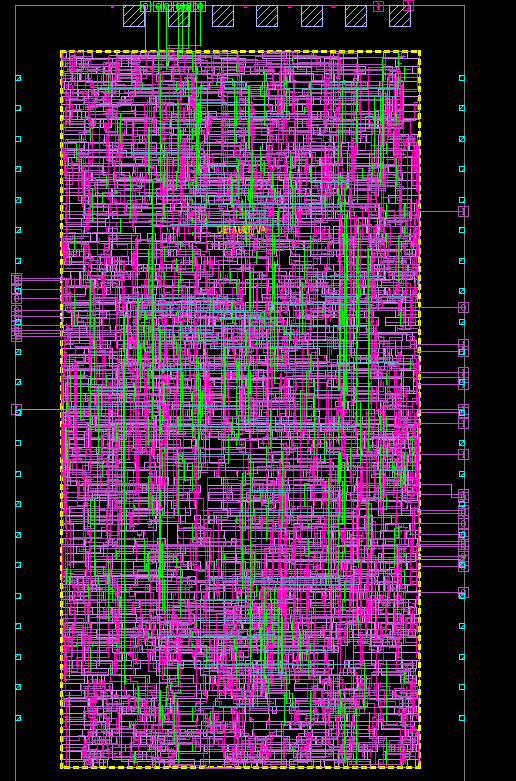


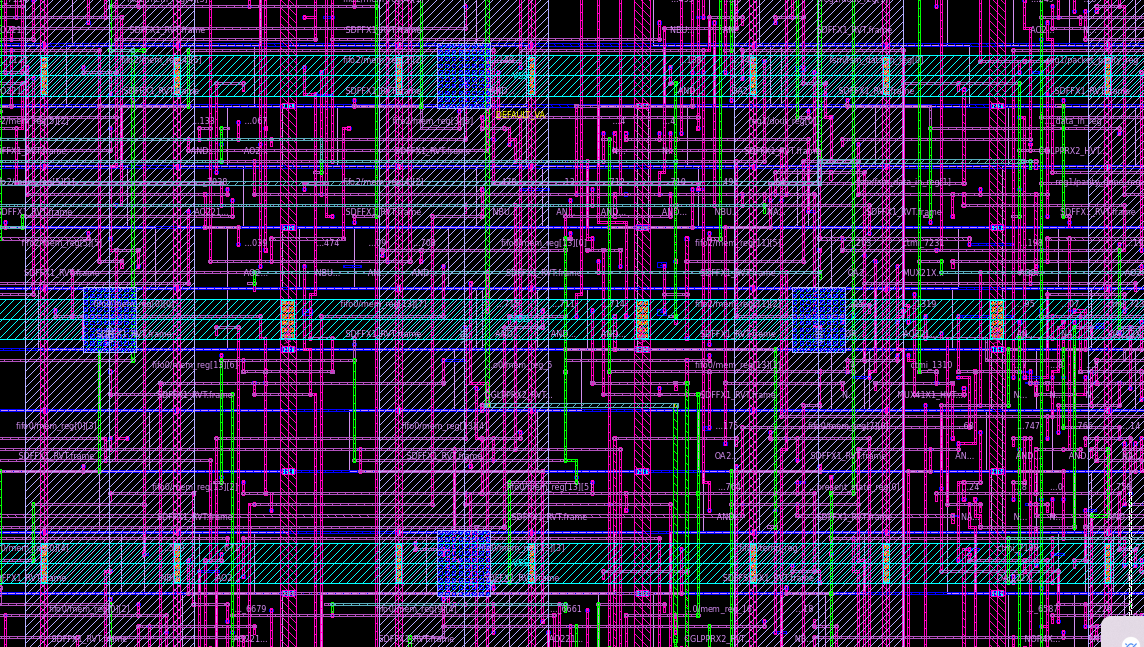
# Routing and Post-Route Optimization

The routing is done by using below command

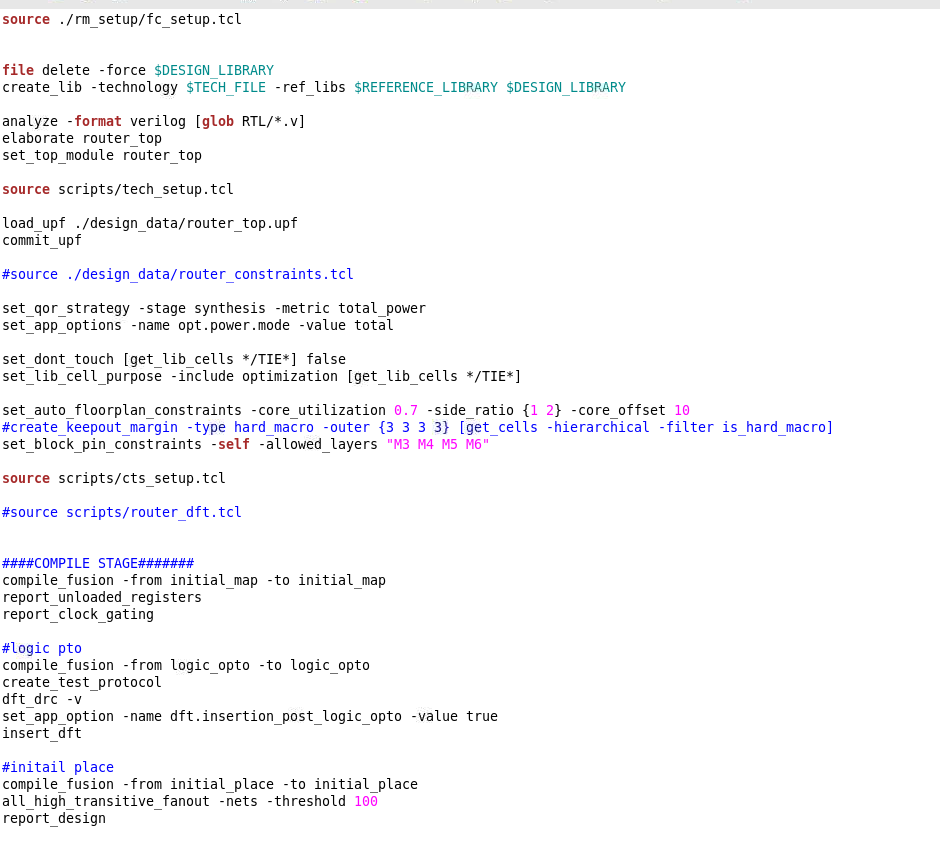
* route\_auto

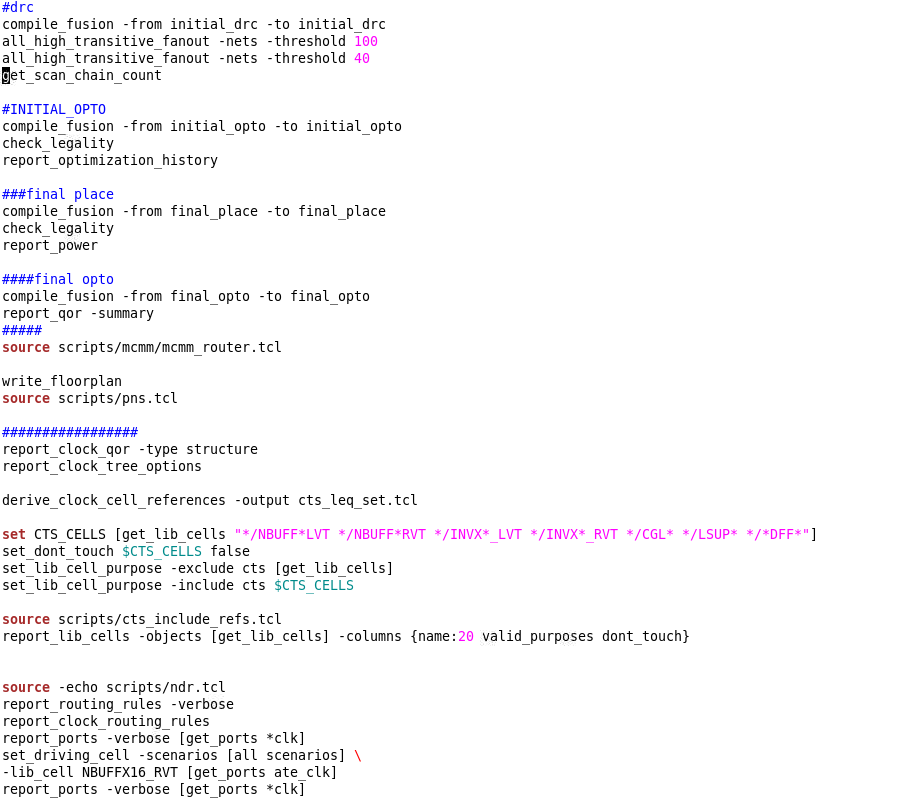


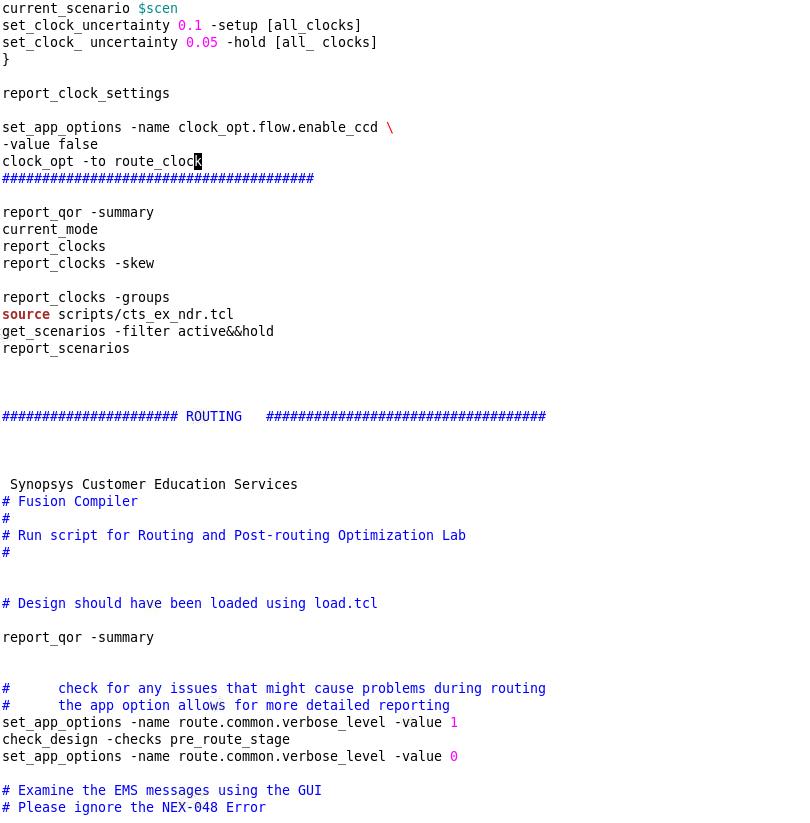


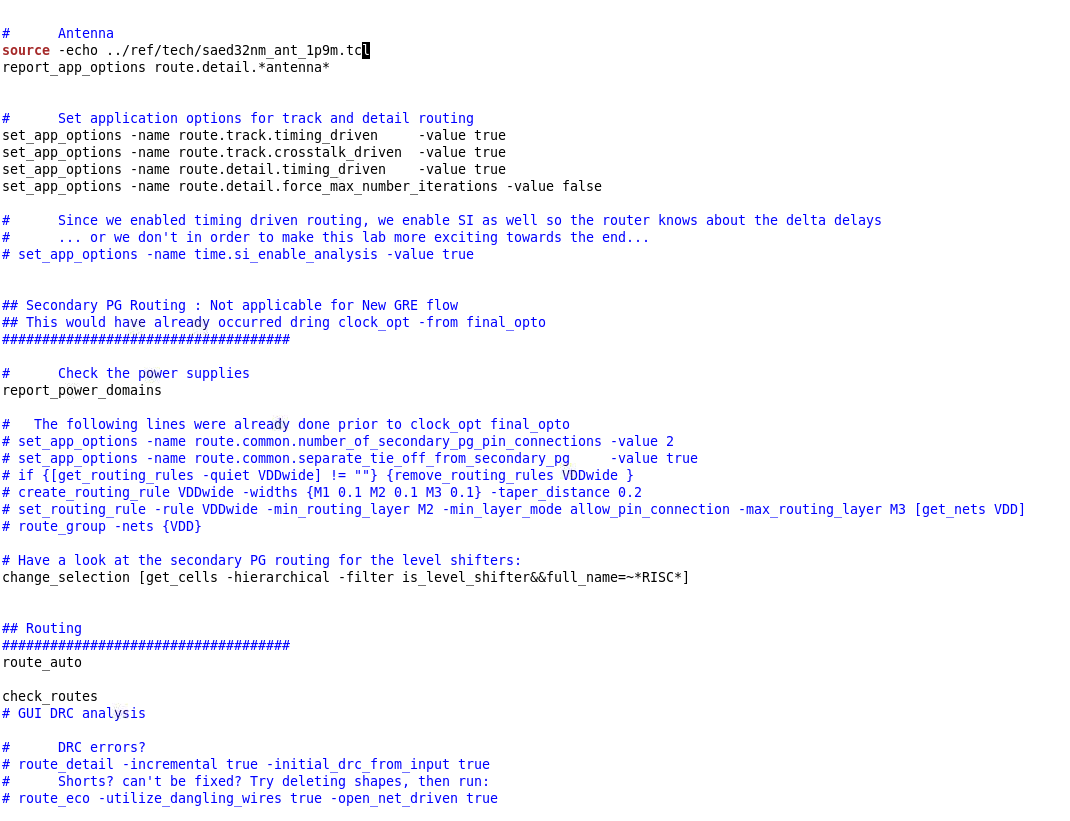


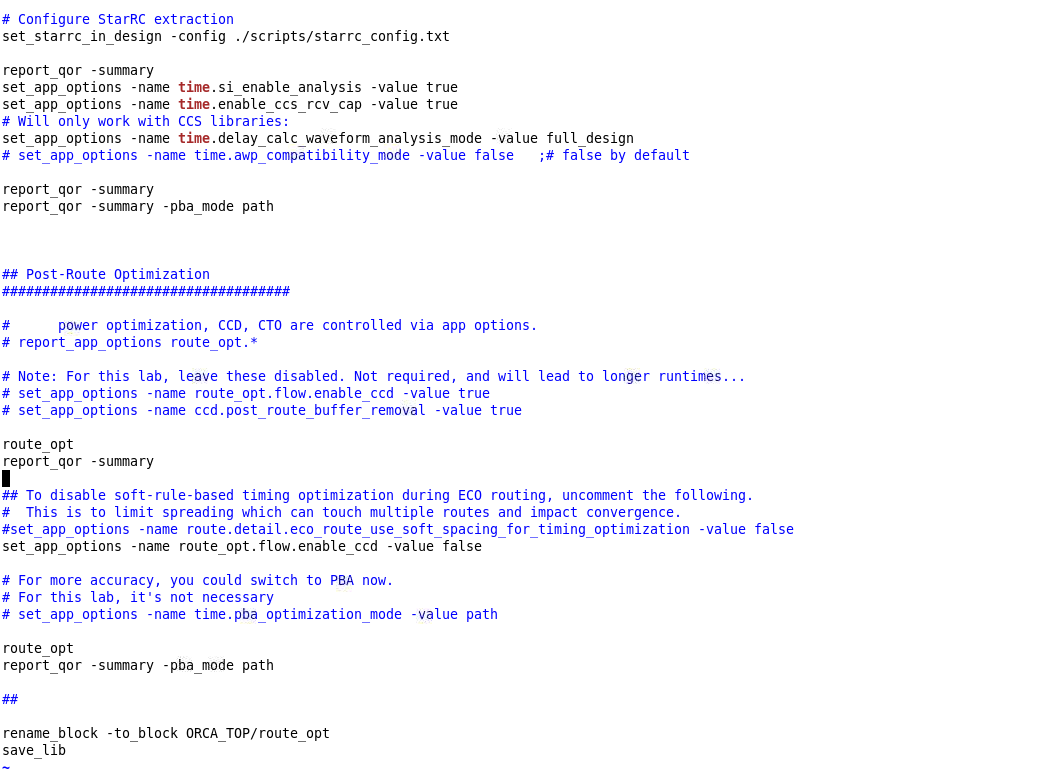
The RUN script used for our design











After completion of routing the content in project folder are

