cadence

ASIC Lab Manual

Revision 4.0

IES132

RC132

EDI132

Developed By

University Support Team

Cadence Design Systems, Bangalore

cadence

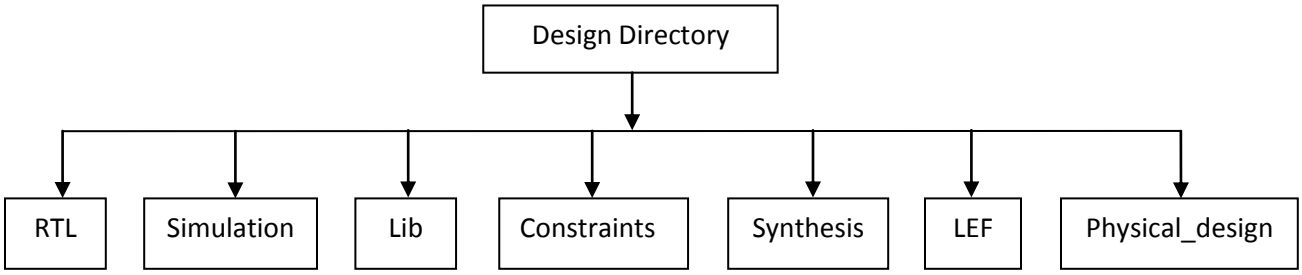
Table of Contents

1. **Prerequisites.....3**
   - Directory Structure.....3
   - Steps to Invoke Tool.....3
2. **Verification.....4**
   - Compilation.....6
   - Elaboration.....7
   - Simulation.....8
3. **Synthesis.....10**
   - Running Synthesis.....10
   - Timing Constraints or SDC file.....11
4. **Physical Design.....14**
   - Import Design.....14
   - Floor Planning..... 29
   - Power Planning.....30
   - Placement.....34
   - Pre-CTC Timing.....35
   - Clock Tree Synthesis.....37
   - Post-CTS timing.....39
   - Routing the Design.....39
   - Physical Verification.....40
   - Generating Stream File.....41

Prerequisites

Directory Structure

Before starting the flow, Please follow the below library structure. This is not mandatory but will help in separating design files, library files, log files and command files generated by tools.



Below figure (fig 1) is an example for the directory structure created for a counter design.

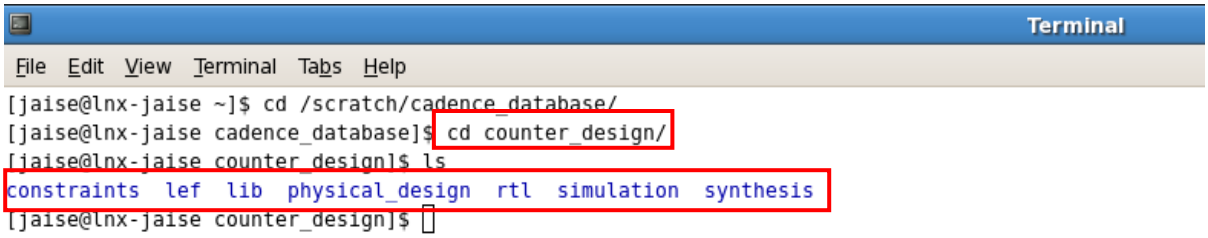


Fig 1

Please save your design (RTL files) inside the RTL directory and keep both RTL as well as test bench inside the simulation directory. Libraries, LEF and Constraint files (SDC files) are kept in respective directories. Simulation, Synthesis and Physical_design directories are used to run simulation, synthesis and physical design so that all log files, command files and other tool generated files won't get mixed up.

Steps to Invoke Tools

- Before invoking any tool, invoke C shell be typing 'csh' in terminal.
- Source the cshrc file by typing 'source <cshrc file>'
  e.g.:- source cshrc

Note: Every time you open a new terminal to invoke the tool, above steps should be followed.

Verification

IES (Incisive Enterprise Simulator) is the tool used for verification. Navigate to Simulation directory where you have kept your RTL and test bench (simulation directory).

- Invoke the tool by typing '*nclaunch -new*' in the terminal.

Below figure (fig 2) shows how to invoke C shell, source cshrc and invoke Incisive tool to simulate a counter design.

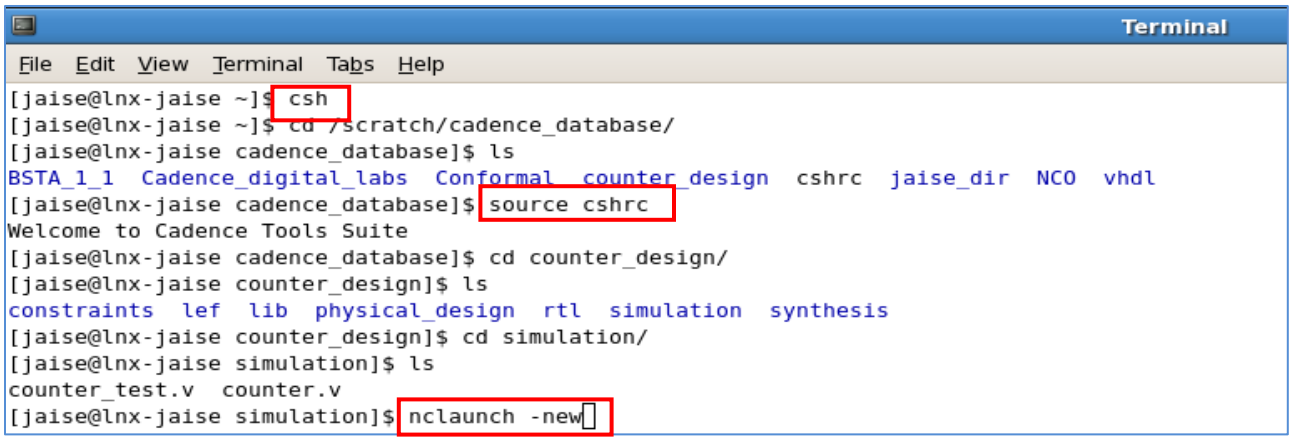


Fig 2

NCLaunch window will appear like in fig 3 and in the NCLaunch window, select 'Multiple Step' option.

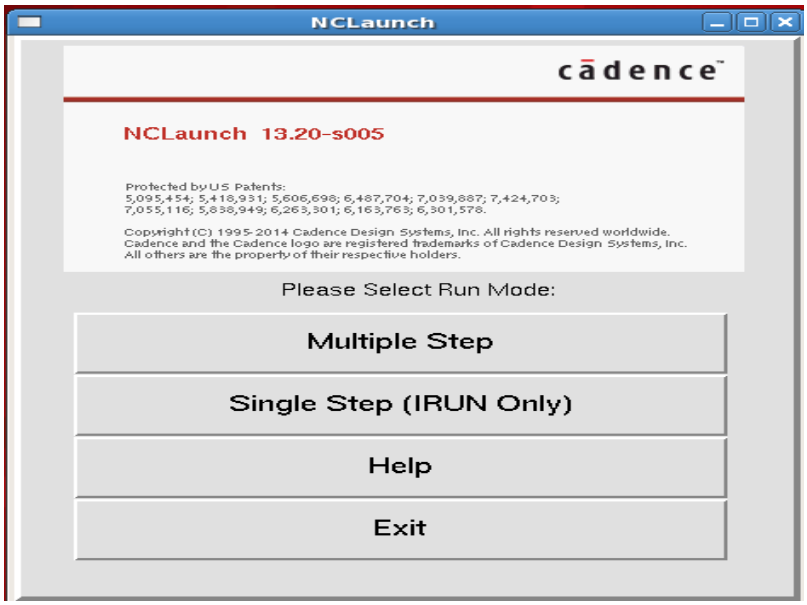


Fig 3

- On clicking the '*Multiple Step*' option, '*nclaunch: Open Design Directory*' window will appear as shown in fig 4.

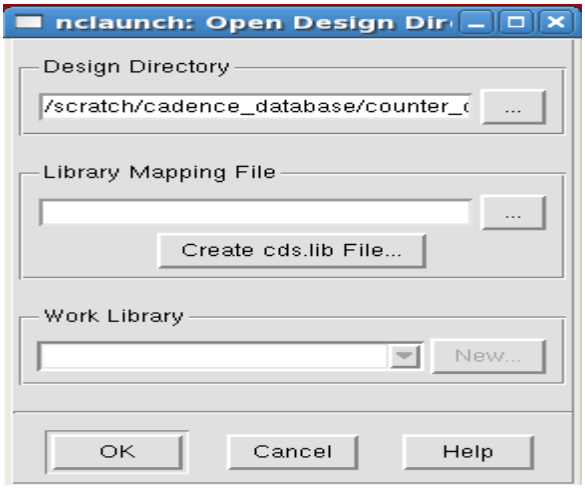


Fig 4

- Click on '*Create cds.lib File*' option and a '*Create a cds.lib file*' window will open. Click '*Save*' option.

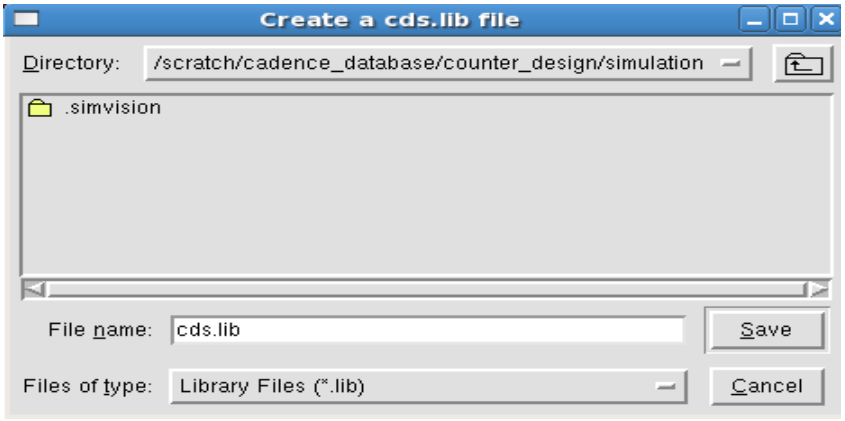


Fig 5

- A '*New cds.lib File*' window will appear. Click any of the three options available depending on your RTL and click '*OK*'. As the counter design is in verilog, the third option is selected.

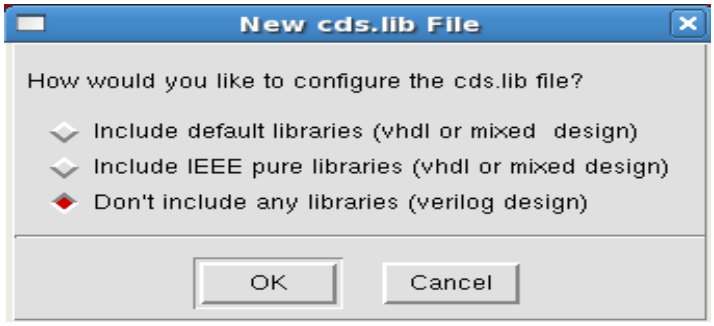


Fig 6

- Click '*OK*' in the '*nclaunch: Open Design Directory*' window.

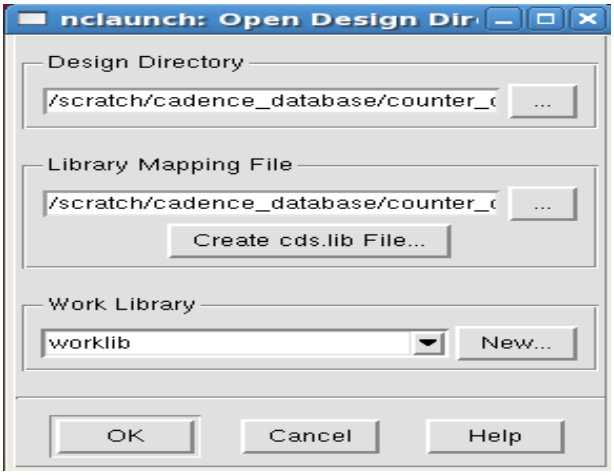


Fig 7

- In the NCLaunch window, we will be able to see the design as well as the testbench that we kept inside the simulation directory.

Compilation

- The next step is to compile (Checks syntax and semantics) the code. For this, select both the design and testbench and choose the appropriate compilers.
  - *ncvlog* for Verilog designs. (choose ncvlog for counter design as the design is in verilog)
  - *ncvhdl* for VHDL designs.

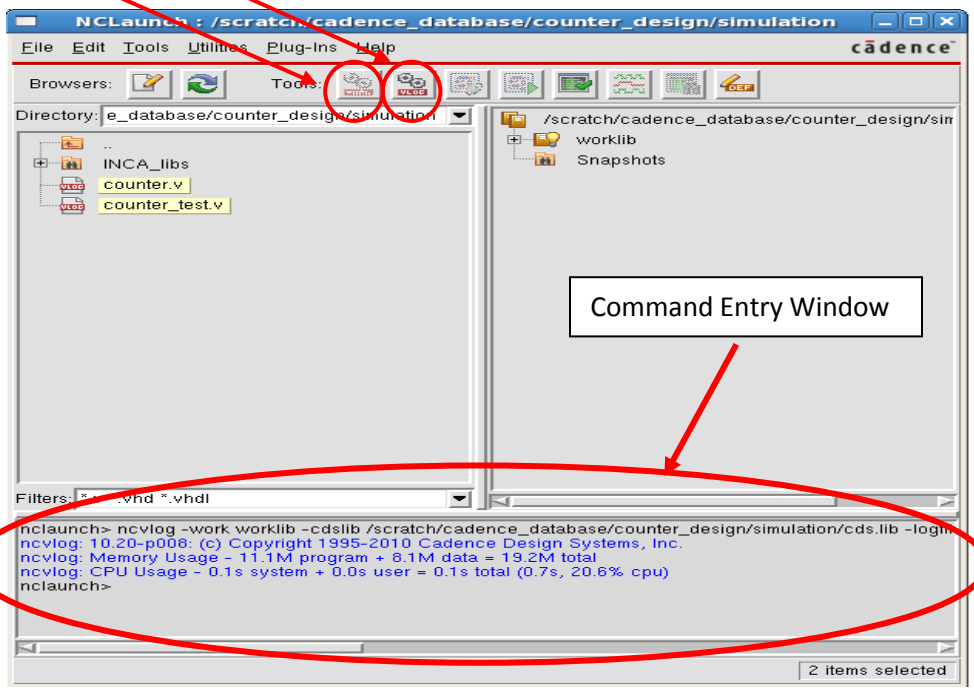


Fig 8

Any error in the code will be reported in the *'Command Entry Window'*.

Elaboration

- After rectifying the errors in the code, the next step is elaboration (constructs design hierarchy and connects signals). Once the compilation is successfully completed, open the *'worklib'* directory on the right side of the window and we can see the design objects created inside. Elaboration should be performed on the testbench as testbench is the top module at this stage and design is instantiated inside the testbench. Select the testbench module and select the '*launch elaborator'* (ncelab) key.

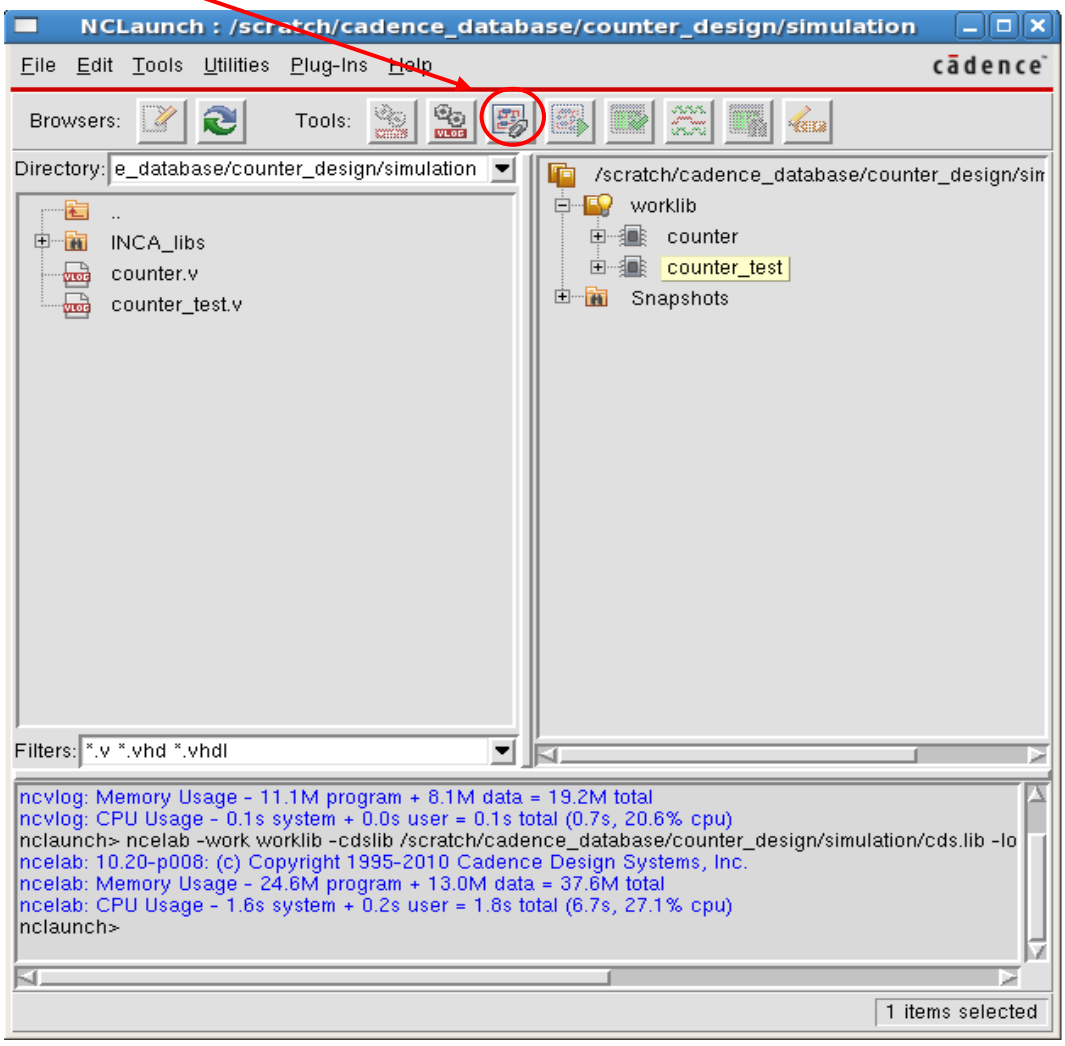


Fig 9

Simulation

- After elaboration, the next step is simulation (executes simulation code). For this we have to send the snapshot generated during elaboration to the simulator.
- Open the *'snapshots'* folder and select the snapshot and click on '*launch simulator*' option.

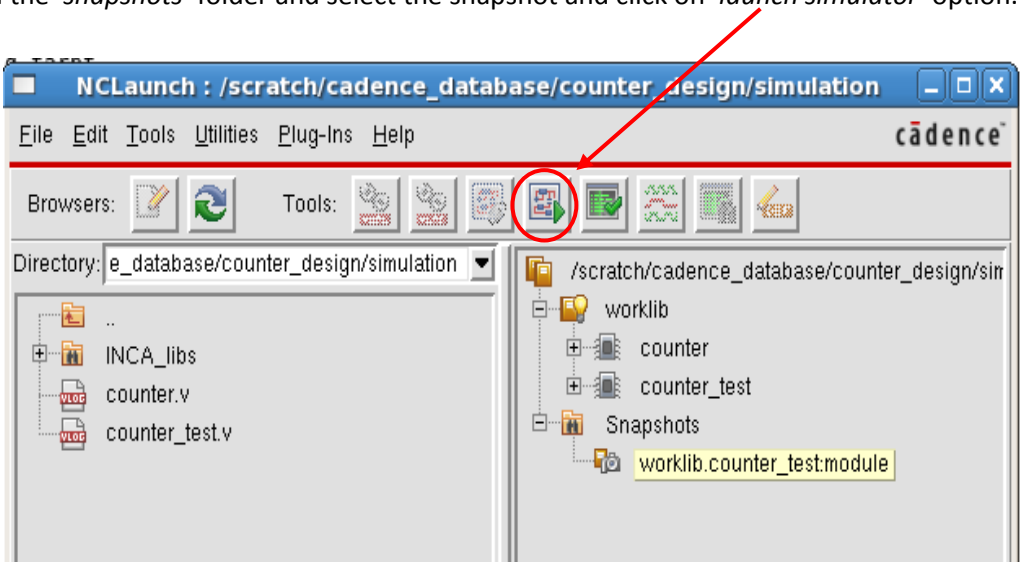
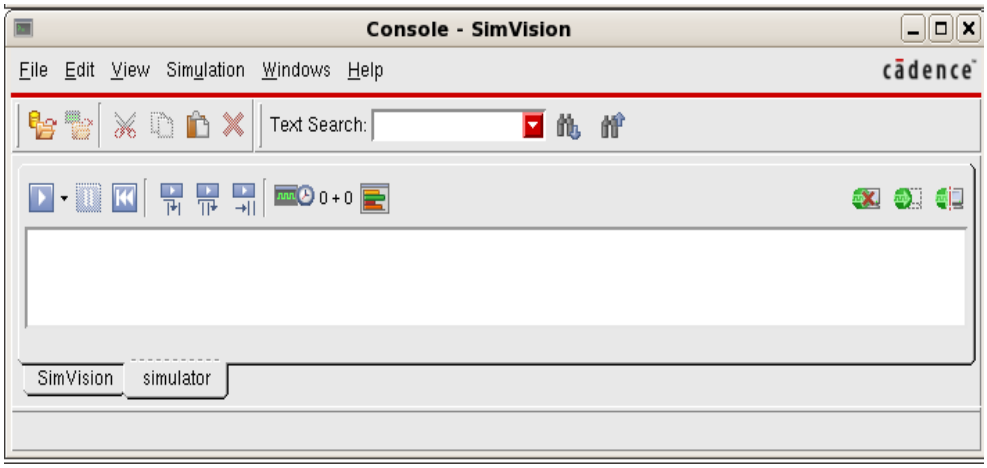


Fig 10

- 'Launch simulator' will open *'Design Browser'* and *'console'* windows. 'Console –SimVision' window can be used to perform simulation in command mode and hence can be minimized while using *'Design Browser –SimVision'* window to run simulation in GUI mode.



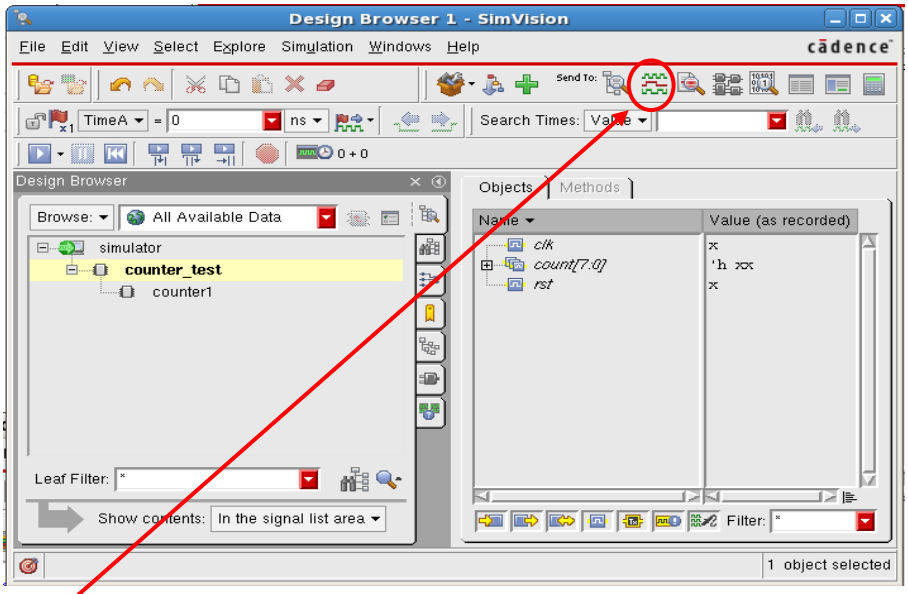


Fig 11

- In the Design Browser window, select the testbench module (counter_test) and select the 'waveform' option. A 'waveform –SimVison' window will appear.
- In the waveform window, we can see different ports in the design. Now click on the Run simulation key to start the simulation. Use the 'pause' key to interrupt or stop the simulation. Use different options like zoom in, zoom out etc to analyze the plot. There are many different options available in the 'Design Browser' as well as in the 'Waveform' window to analyze the design and debugging.

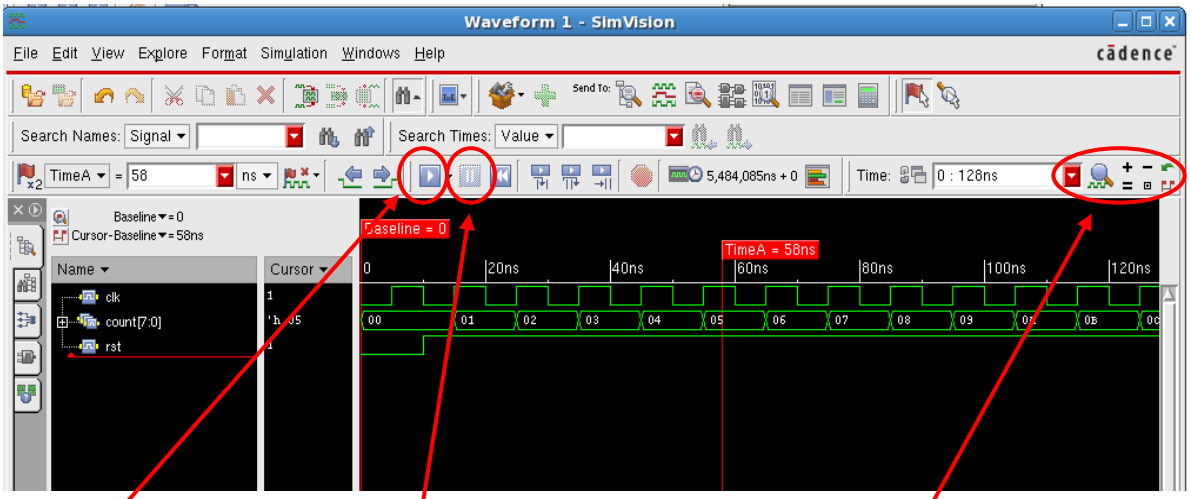


Fig 12

Run Simulation Interrupt Simulation Zoom in, Zoom out etc..

- After verifying the design, close the tools. We can now proceed for synthesis.

To know more about tool options, use help or pdfs present inside the doc directory of the tool.

Synthesis

The tool used for synthesis (converting RTL to gate level netlist) is RTL Compiler (RC).

Running Synthesis

Change the directory from simulation to synthesis and write a script file for synthesis.

Below is an example of a script file for synthesis.

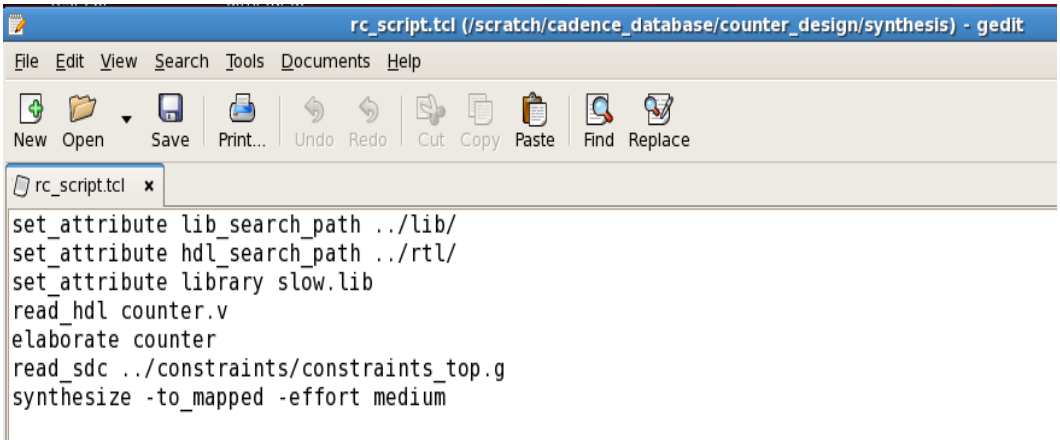


Fig 13

- The necessary inputs to perform synthesis are RTL, standard cell library and constraints.
- Let us see the usage and purpose of each command.
    - set_attribute lib_search_path <library path>
      This command will set the path for the standard cell library.
    - set_attribute hdl_search_path <rtl path>
      This command will set the path for rtl files.
    - set_attribute library <library name>
      This command will read the specified standard cell library from the specified library path.
    - read_hdl <rtl design>
      This command will read the rtl design.

      Note:- If the design is hierarchical or has multiple modules instantiated inside the top module, use curly braces '{ }' to mention all modules including the top design.

      E.g.:- read_hdl {top.v sub1.v sub2.v}
      Here top.v is the top module and sub1.v and sub2.v are the sub modules that are instantiated inside the top module.
    - elaborate
      The elaborate command constructs design hierarchy and connects signals.

- read_sdc < sdc file name with path>
  This command reads in the timing constraints file. Here we have to provide the constraints file name along with the path. Explanation on constraints file is provided
- Synthesize –to_mapped –effort medium
  This command will perform synthesis by combining the generic, mapped and incremental synthesis and – effort medium command specifies the synthesis effort. The effort can be set to 'low', 'medium' or 'high' depending upon the scenario.

➢ Include all the above commands in the script file.
Note: - In counter design, you can see a script file 'rc_script.tcl' inside the synthesis directory. Please open the script file for further understanding.

Timing Constraints or SDC file

Now let us understand the content of the Constraints or SDC file.

➢ Using SDC, we define clock period, pulse width, rise and fall time, uncertainty and also input and output delays for different signals. Below is the constraints file used in counter design.

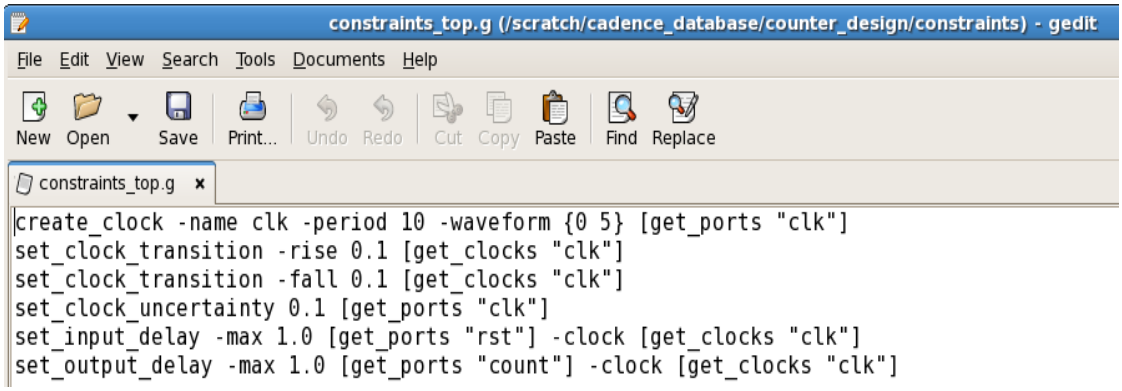


Fig 14

➢ Let us see the usage and purpose of each command.
- Create_clock –name –period 10 –waveform {0 5} {get_port "clk"}
  This command will define clock with period 10ns and 50% duty cycle and signal is high in the first half.
- 'Set_clock_transition –rise/fall' command defines the transition delay for clock.
- 'Set_clock_uncertainty' command will set the uncertainty due to (clock skew and jitter).
- 'Set_input/output_delay' command will specify the input and output delay used for timing slack calculations.

➢ Keep the constraints file inside the constraints directory.
Note: - To know more about writing timing constraints, please refer the rc_ta.pdf available inside the doc/rc_ta directory of the tool.
I.e. - <RC_tool directory>/doc/rc_ta/rc_ta.pdf

Once the script file to run synthesis and the constraints file are ready, we can initiate synthesis.

- Use the below command to invoke RTL compiler along with the script file.

  rc -f <script file name with path>

  'rc' is the command to invoke RTL Compiler and '-f' option is used to passes the script to RC at the time of launching the tool. RC will execute each commands mentioned inside the script file one by one.

  Note: - If the script file is in the current working directory (synthesis directory), we need not have to provide the path for the script.

  E.g. - In case of the counter design, the command will be 'rc –f rc_script.tcl'

- While performing synthesis, always check the RC terminal whether the tool is reporting any error. Figure below shows the RC terminal after synthesis.

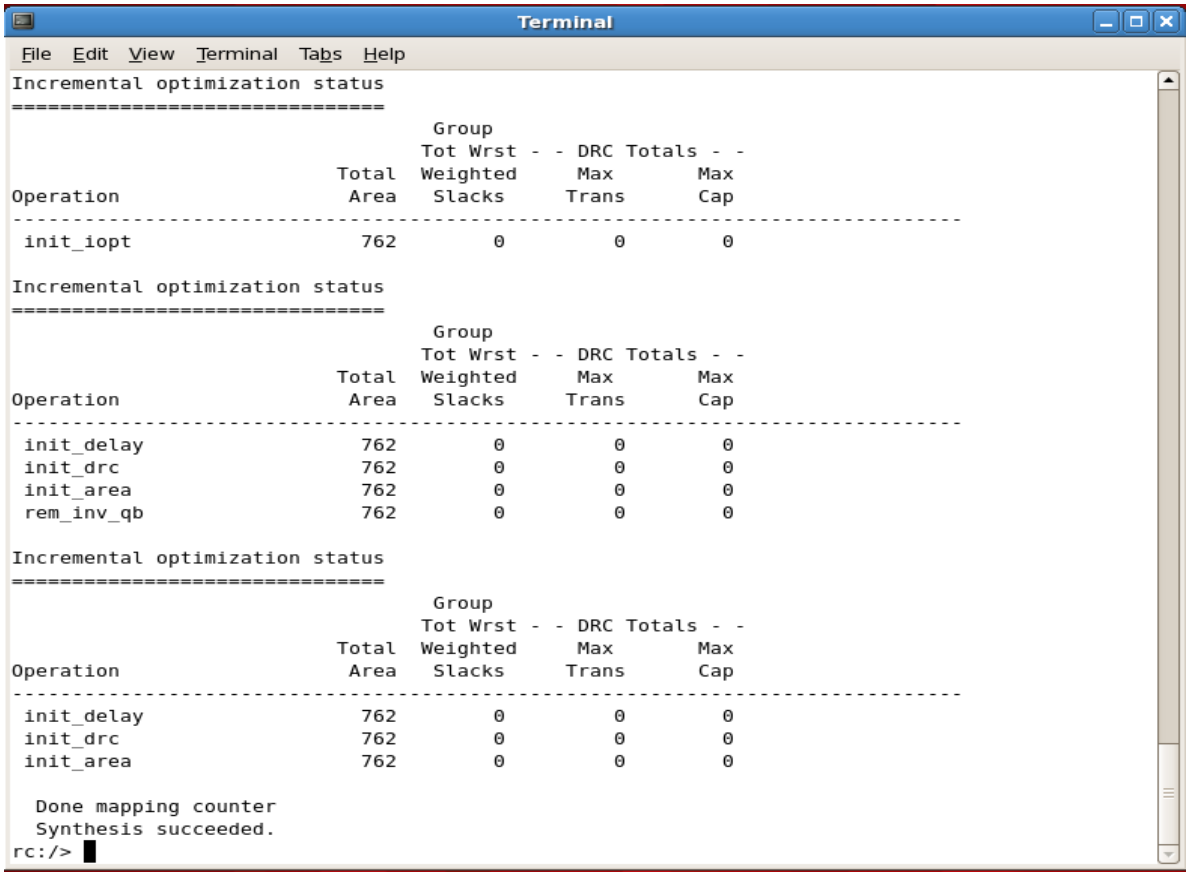


Fig 15

- After synthesis, to see the schematic, launch the gui using the below command. *'gui_show'* in the *'rc'* terminal. Use *'gui_hide'* command to close the gui.

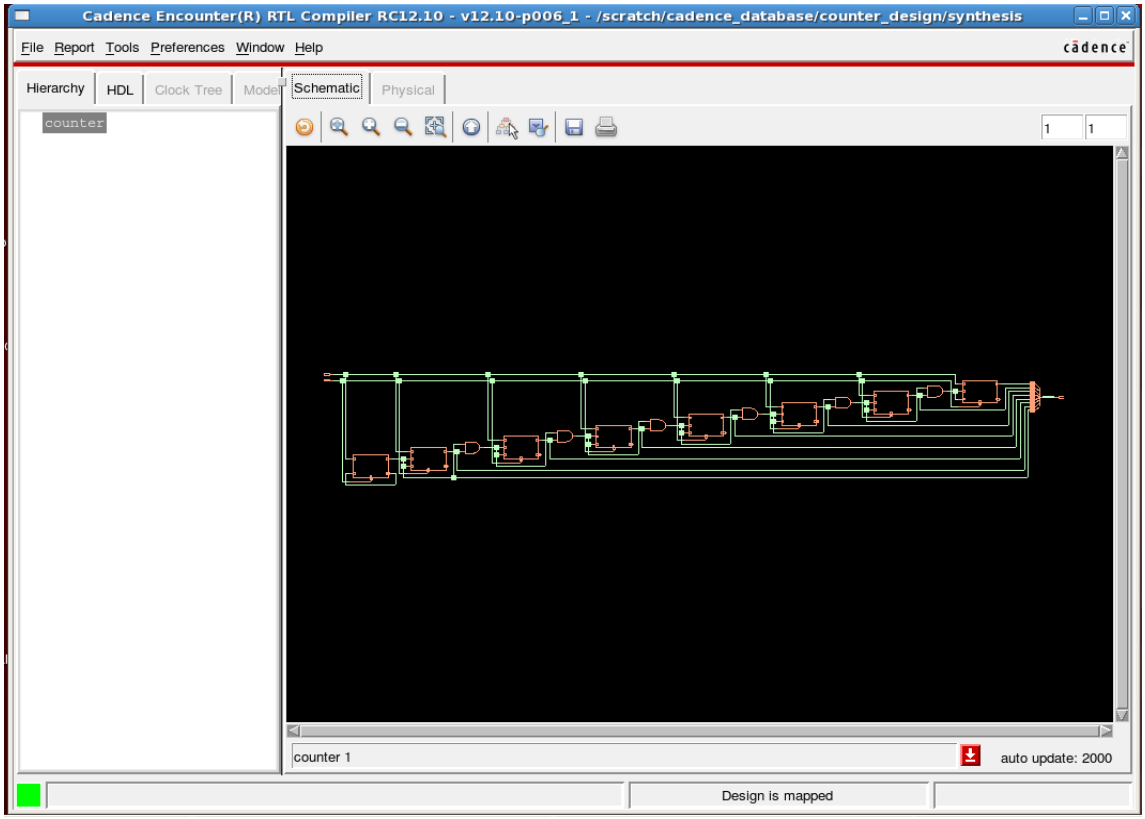


Fig 16

- Use 'report' command to write out the results.
  - 'report timing' to report the timing details.
  - 'report power' to dump out the power report.

Note: - Use just report command to know what all different reports you can dump out from RC.

- After completing synthesis, Use the below command to dump out netlist and SDC for Physical design.
  - *write_hdl >* netlist_name.v
    You can provide any name for the netlist file but the extension of the file should be *'.v'*
  - *write_sdc >* sdc_name.sdc
    You can provide any name for the SDC file but the extension of the file should be *'.sdc'*

The above commands will generate netlist and SDC in the synthesis directory.

- Use 'exit' command to close RC.

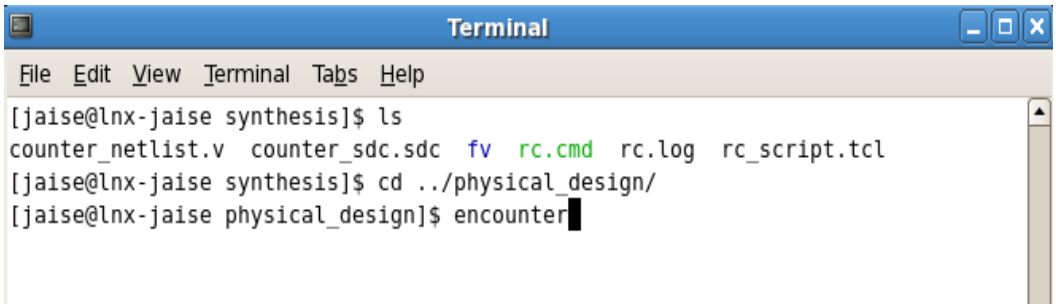
Note: - To know more about synthesis, please refer the rc_user.pdf available inside the doc/rc_ta directory of the tool.
I.e. - <RC_tool directory>/doc/rc_user/rc_user.pdf

After dumping out the netlist and SDC, we can proceed for Physical Design. Please close the RC tool.

Physical Design

Encounter Digital Implementation (EDI) is the tool used for physical implementation. Inputs required for physical implementation are Netlist, SDC and LEF (Library Exchange Format – This file contains the details the physical details for the standard cells).

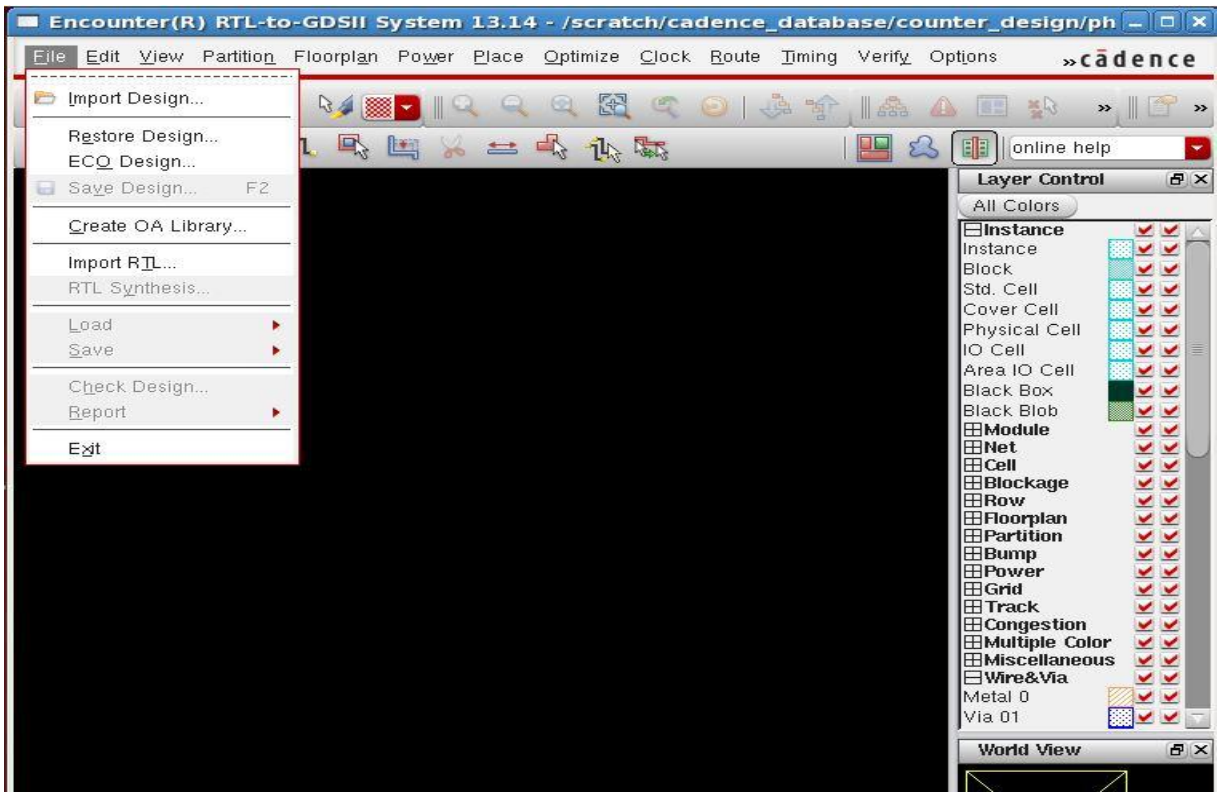
- Invoke Encounter tool inside the 'physical_design' directory typing 'encounter' then click enter



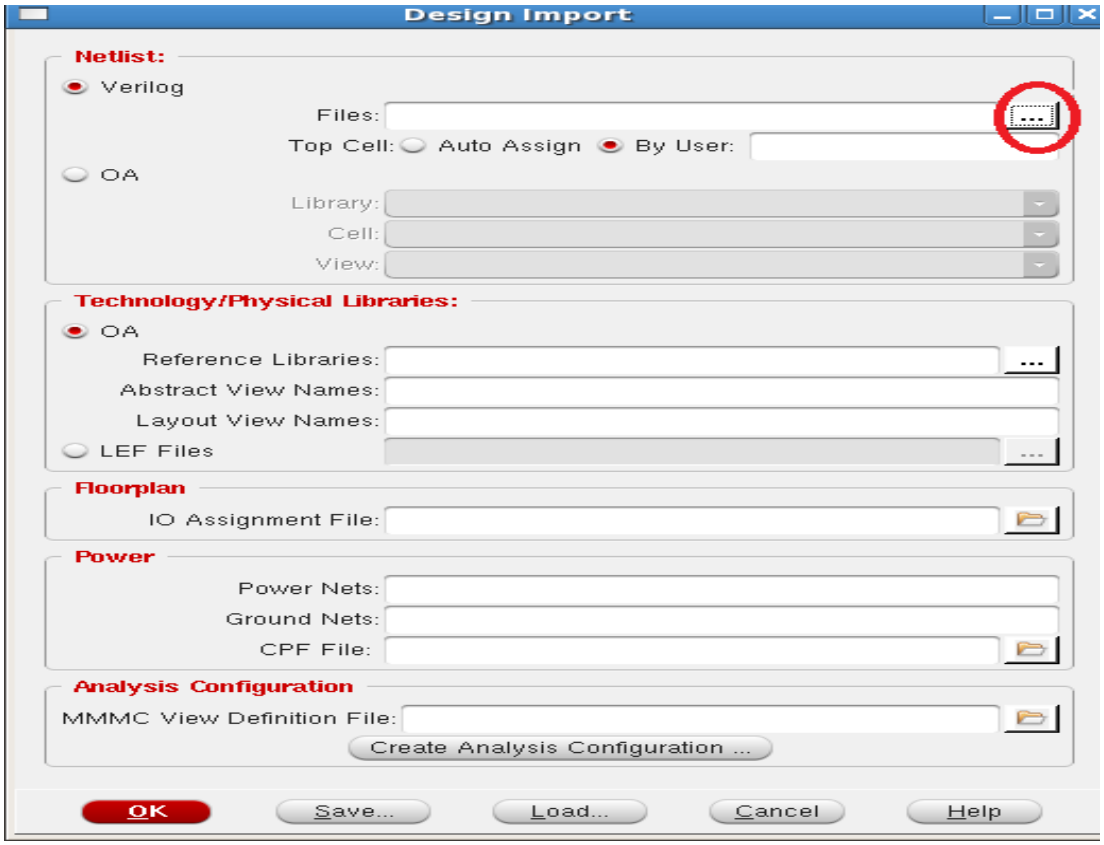
Encounter RTL-to-GDSII System <version> window will pop-up

Import Design

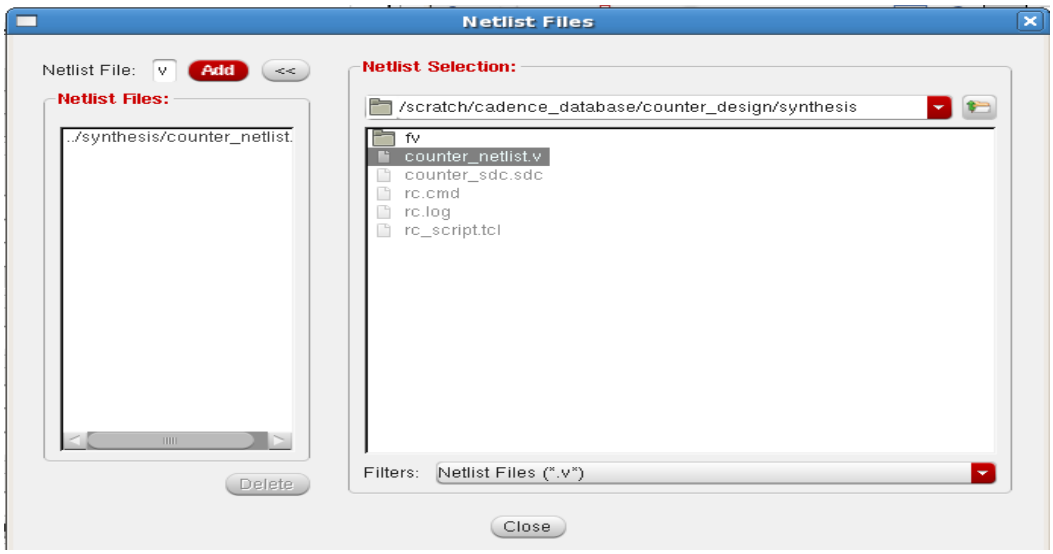
- To import the design click on File→ Import Design



- Design Import form will popup, browse to the gate level net-list (.v file) that you have created from the synthesis stage.
  - Click on highlighted portion

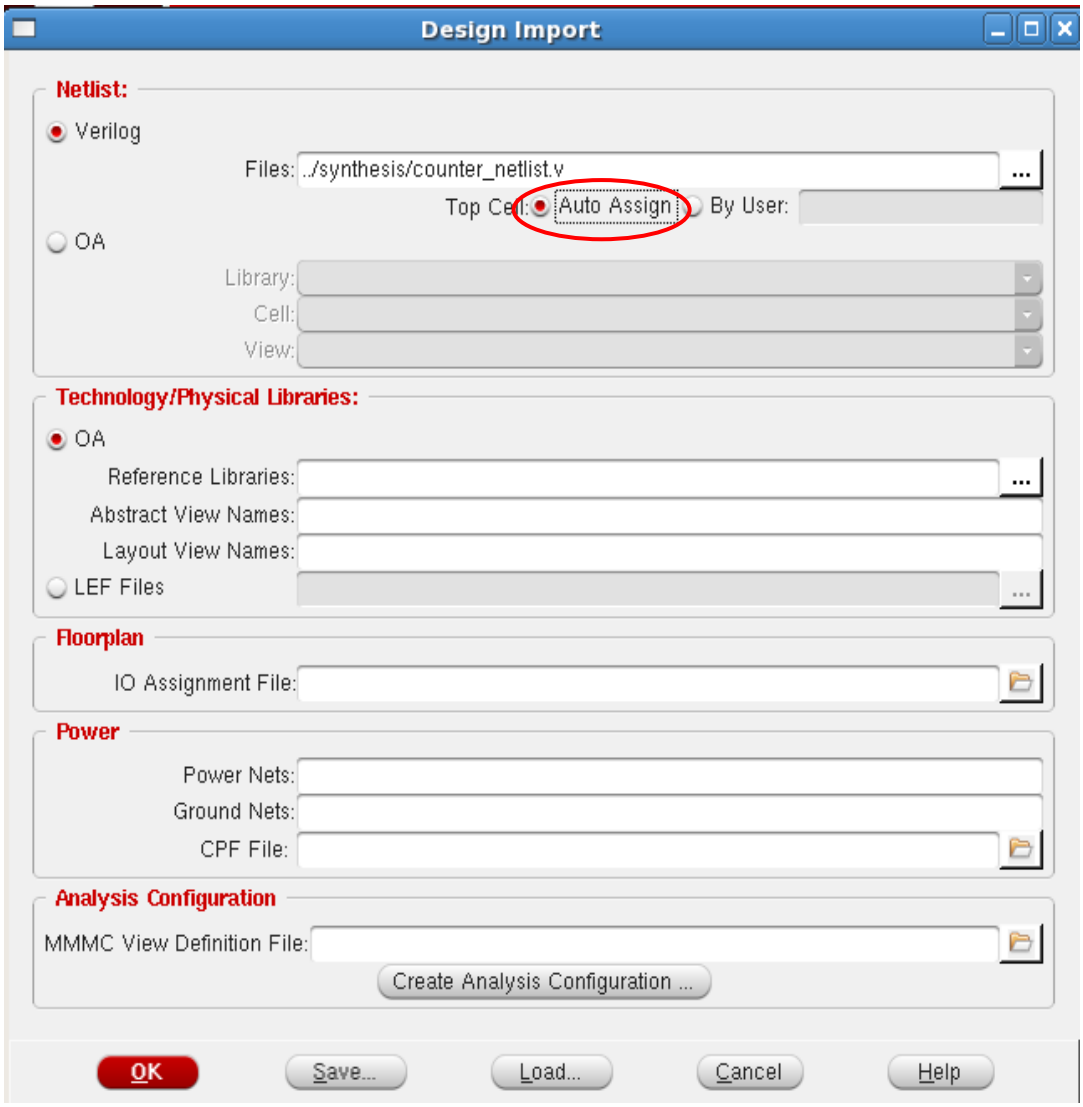


  - Under Netlist Selection browse to the .v file you have created using write_hdl command at synthesis stage. Then click on 'Add', then it should appear under the Netlist Files section as shown in the below figure.

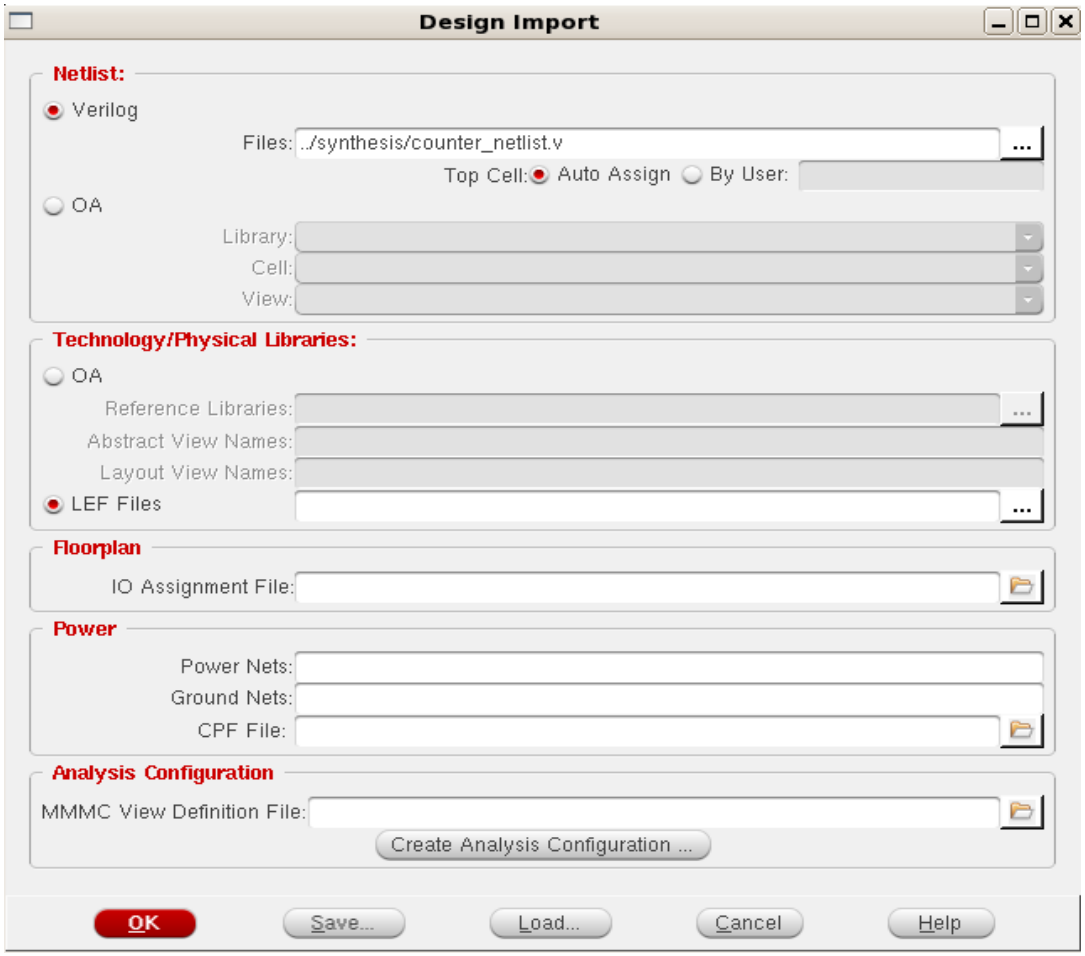


- Click on Close button

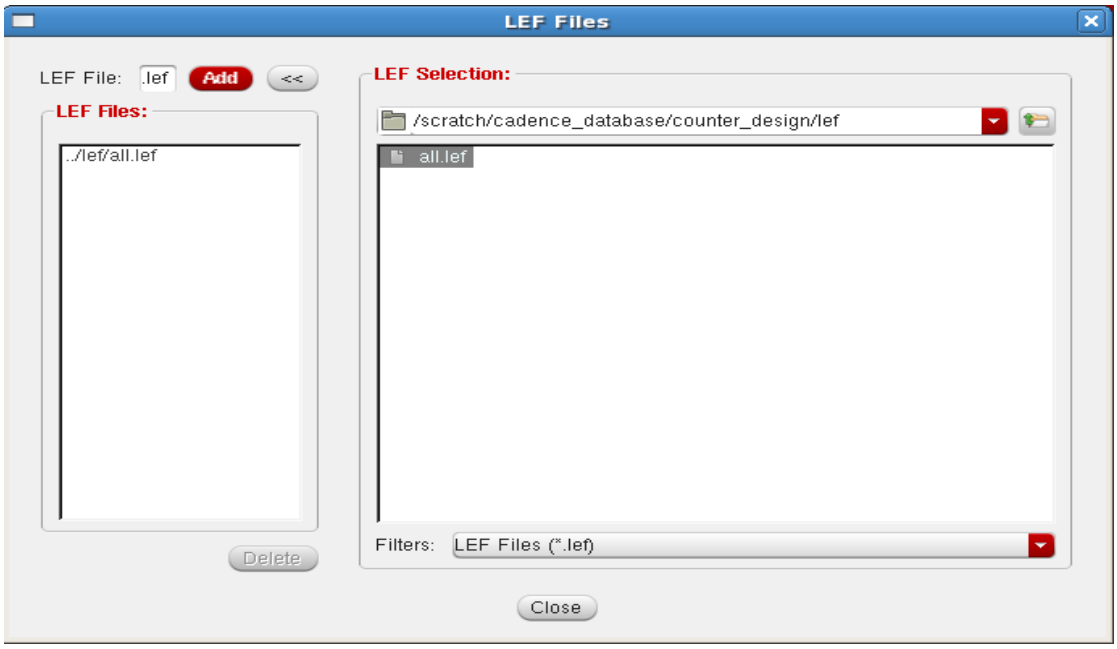
➢ Change Top Cell to Auto Assign



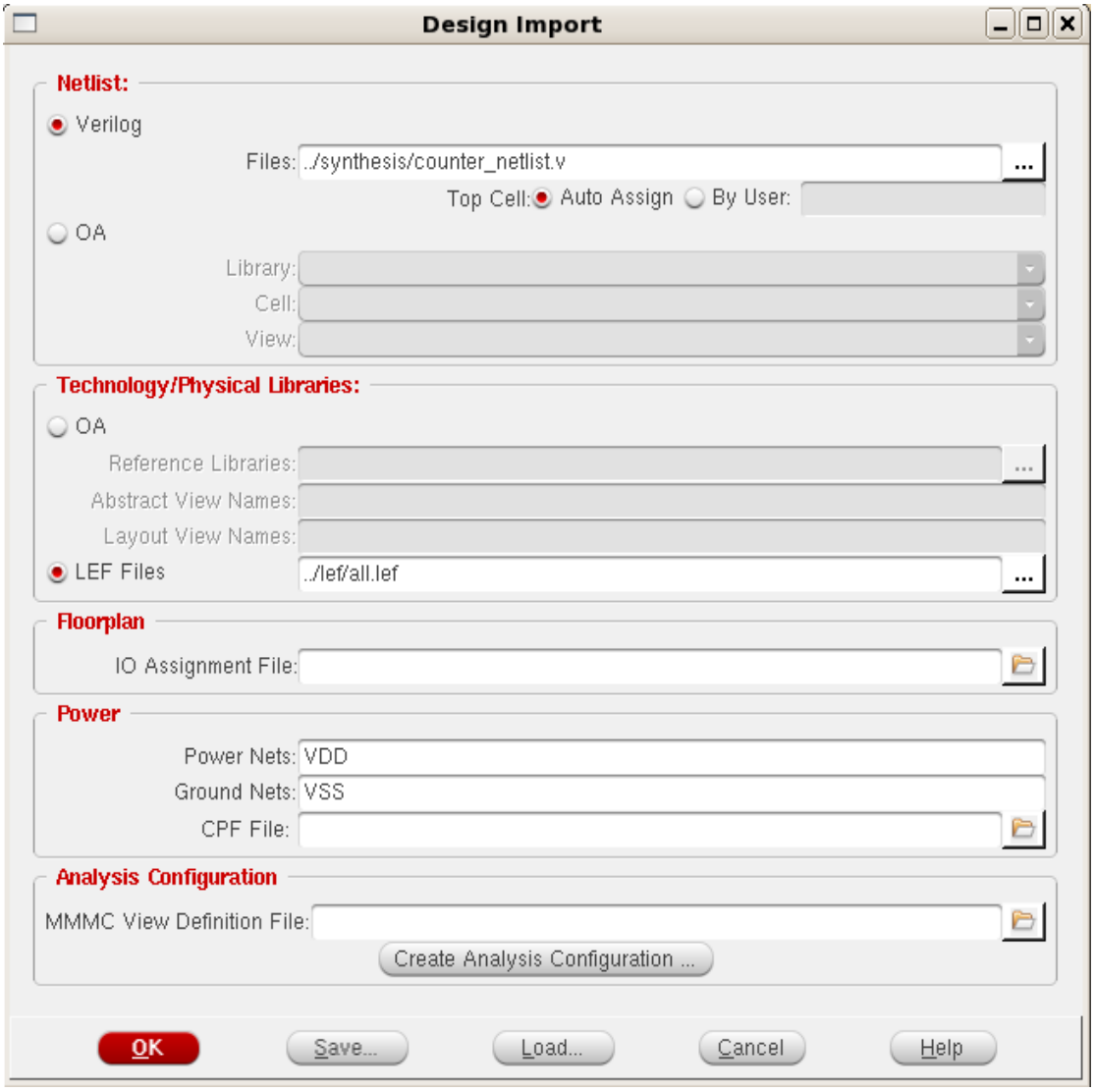
➢ Click on LEF Files



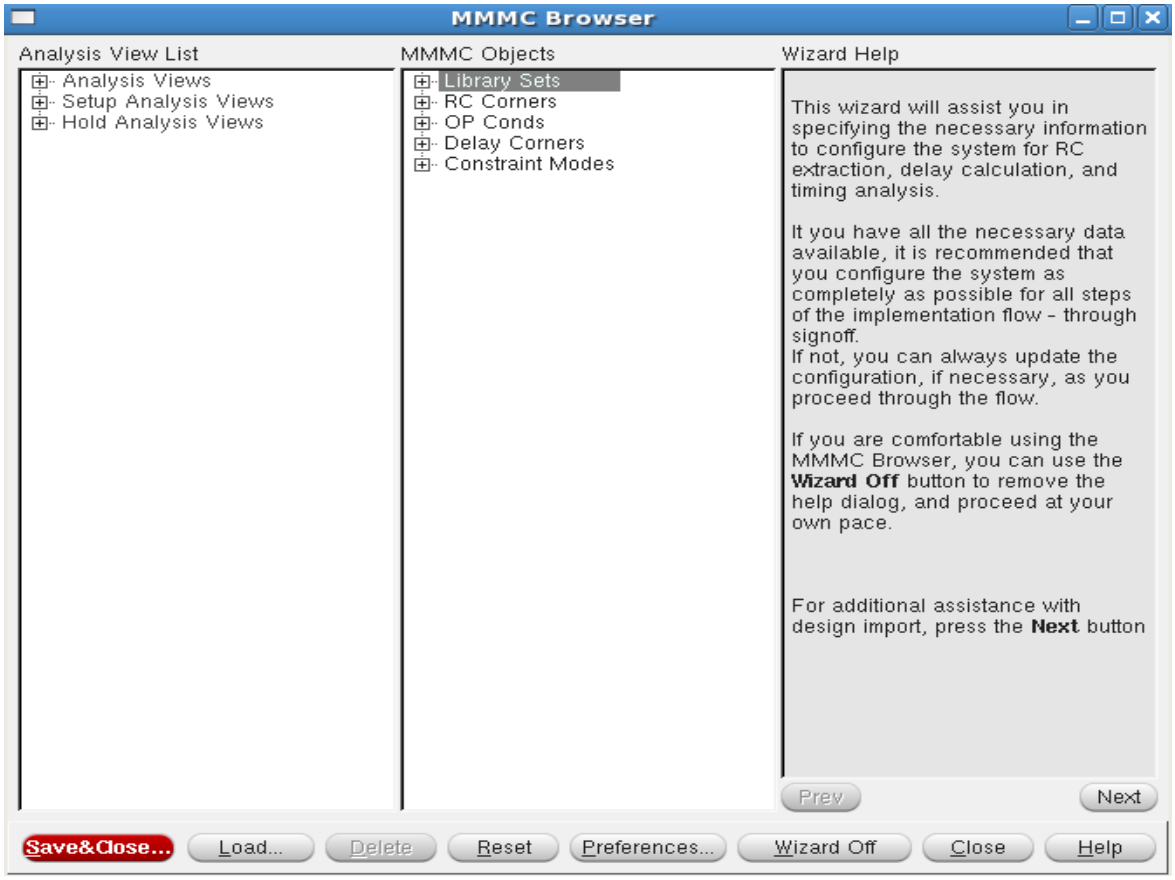
- Browse the LEF (Library Exchange Format) file and click close



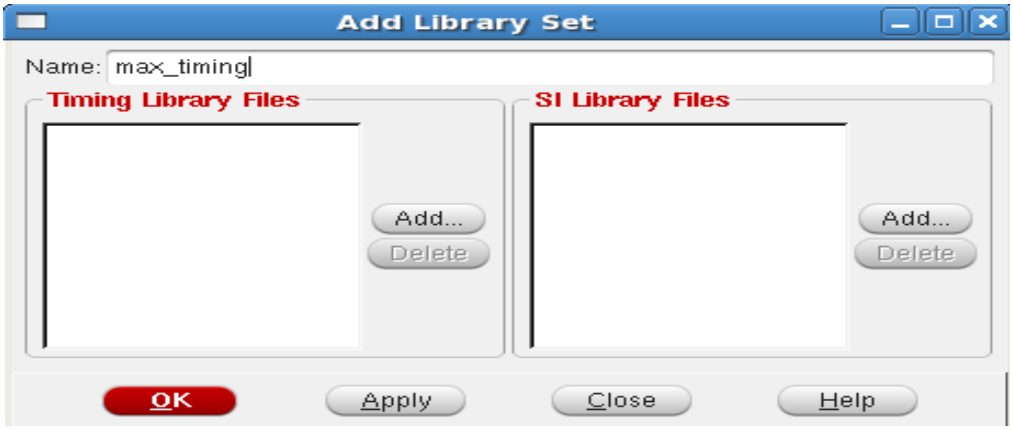
- Give names to Power and Ground nets as VDD and VSS respectively



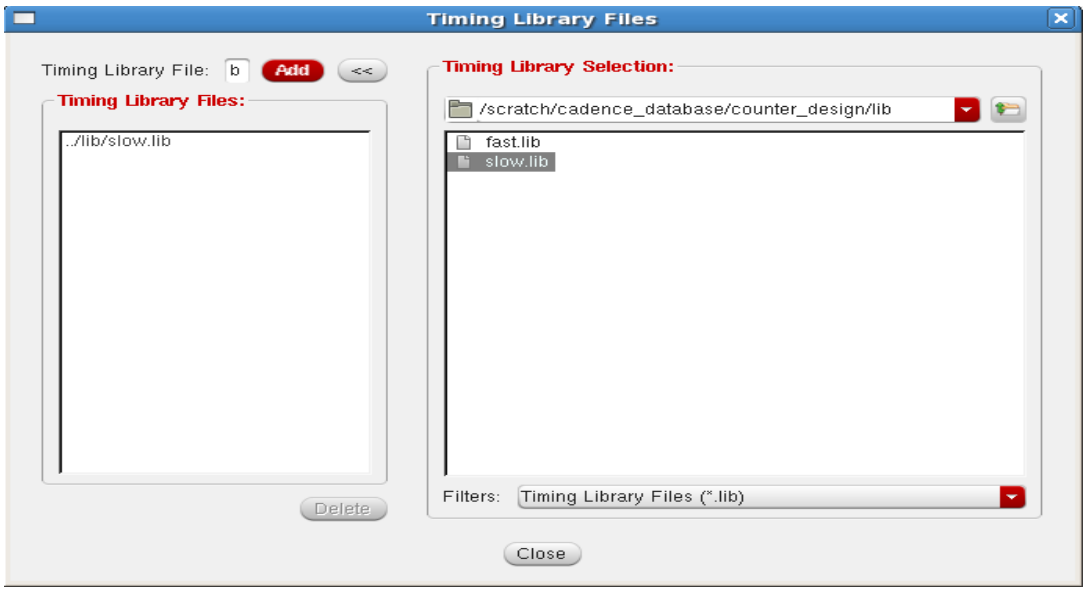
- Click on Create Analysis Configuration
  - A MMMC Browser form appears,
  - Double Click on Library Sets



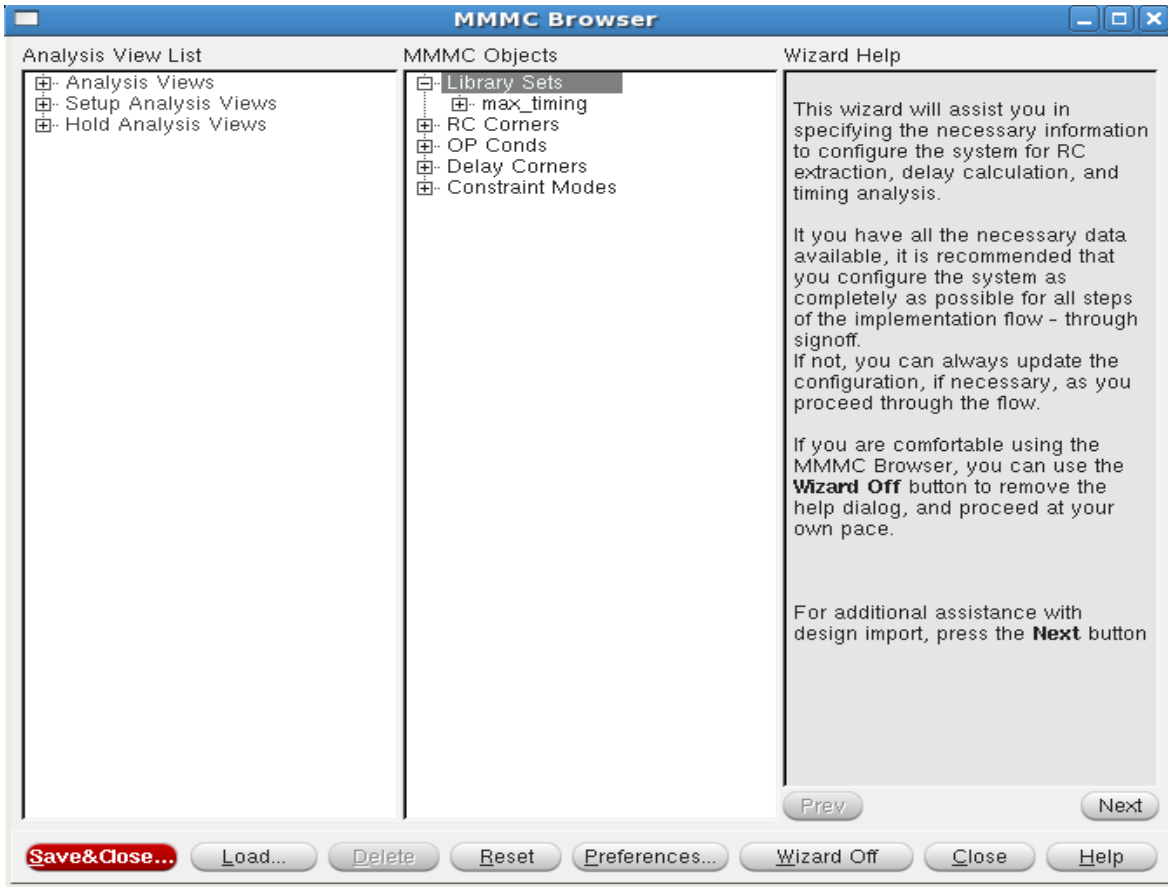
- In the Add Library Set give it name max_timing and click on 'Add…' under 'Timing Library Files'



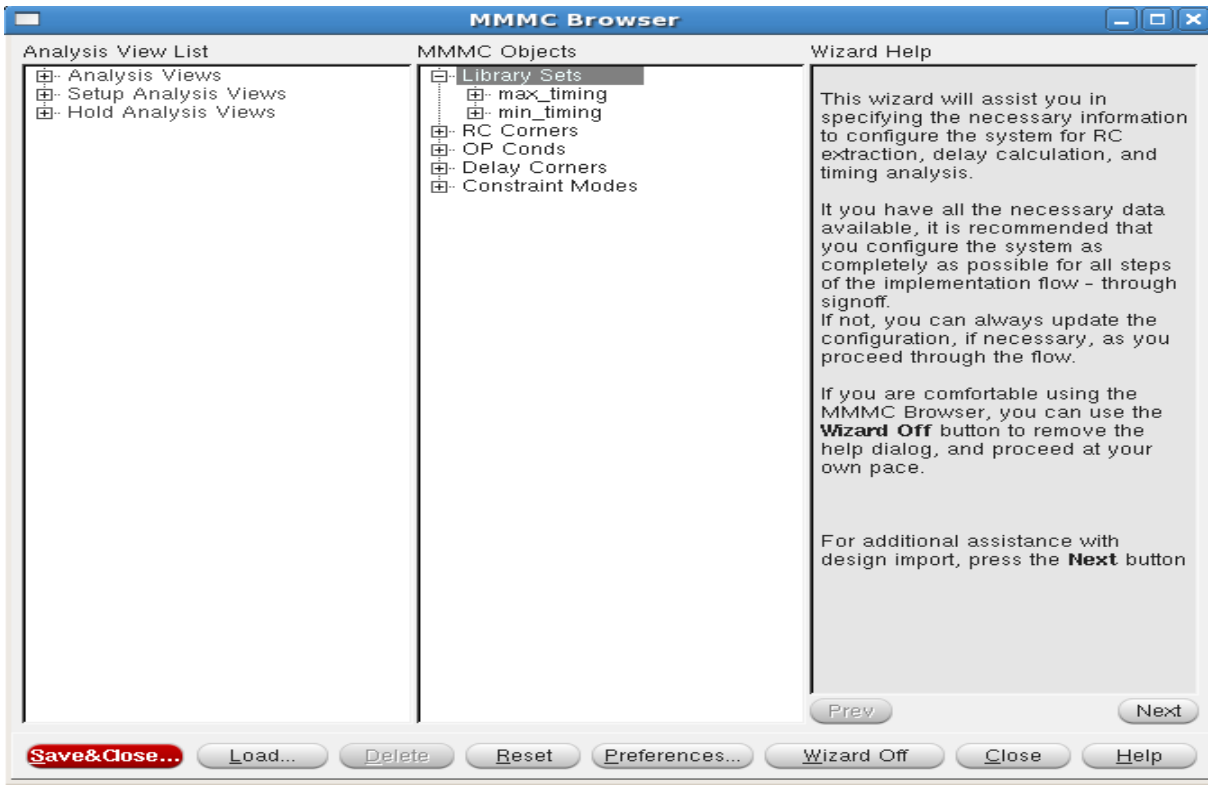
- Maximum timing would be provided by slow.lib. Browse to 'lib' directory then select slow.lib then click on 'Add'. (slow.lib should now appear under Timing library Files) click on Close



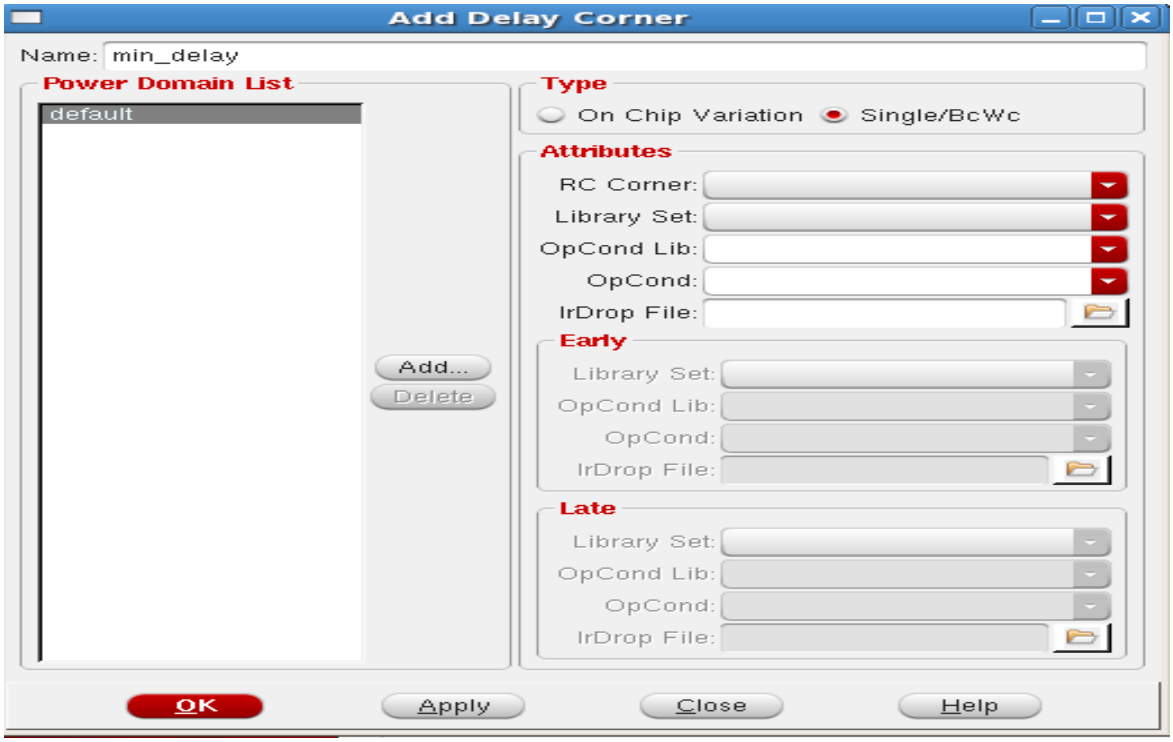
Upon clicking close the MMMC Browser form should look like the below figure



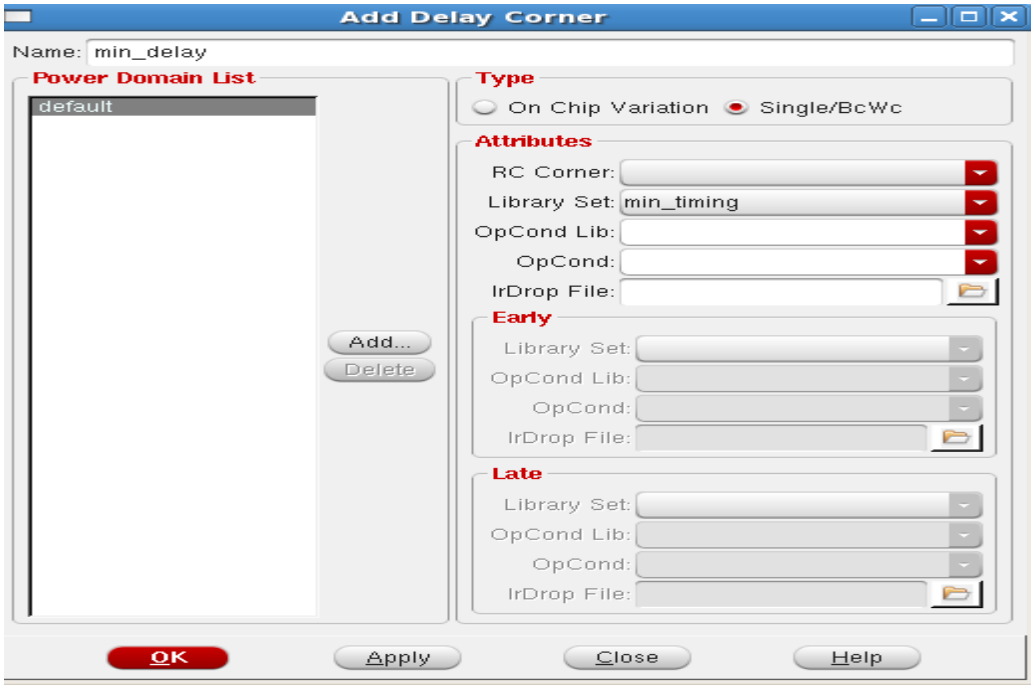
- Similarly for min timing repeat the above steps that we followed to set max timing to slow.lib (double click again on Library Sets , give the name as min_timing , Browse to 'lib' then select fast.lib then click on 'Add' then click on close).

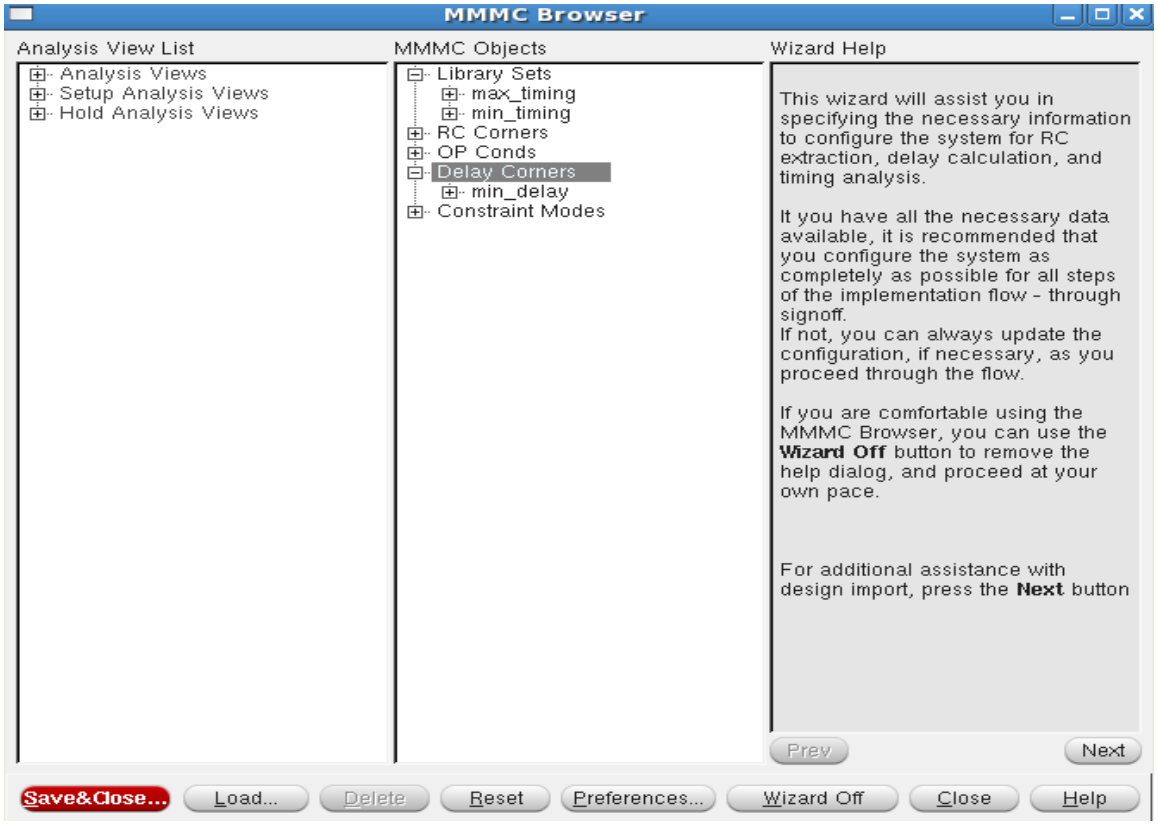


- Double click on 'Delay Corners' and 'Add Delay Corner' form will appear, give the name as min_delay.

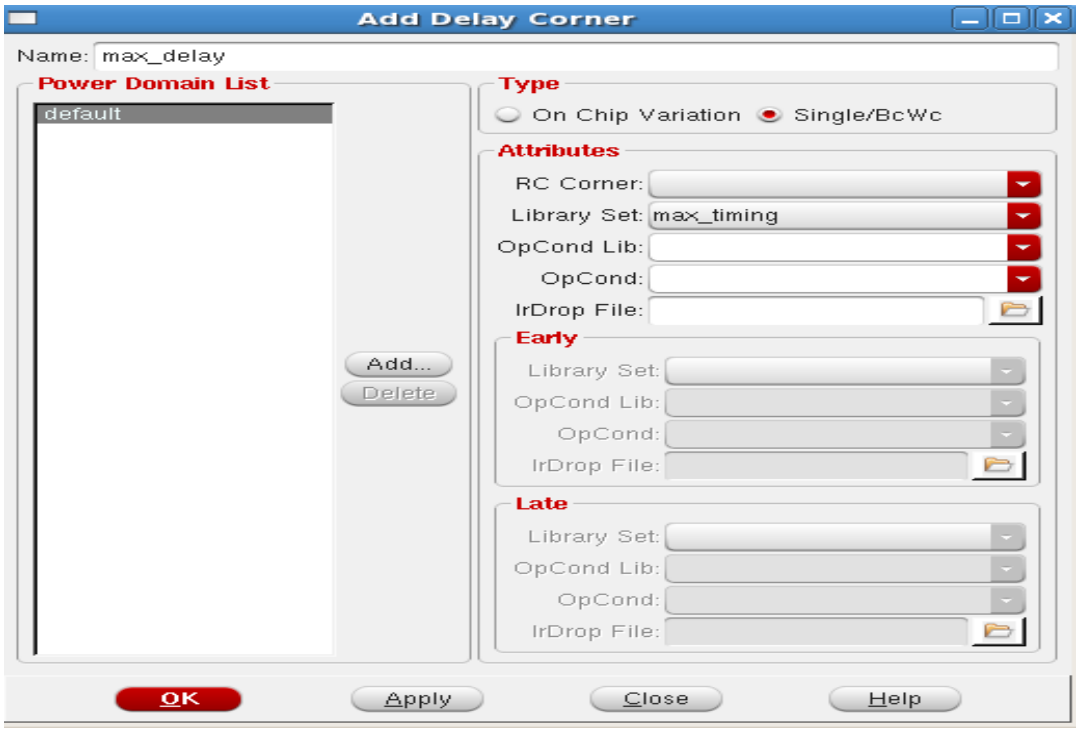


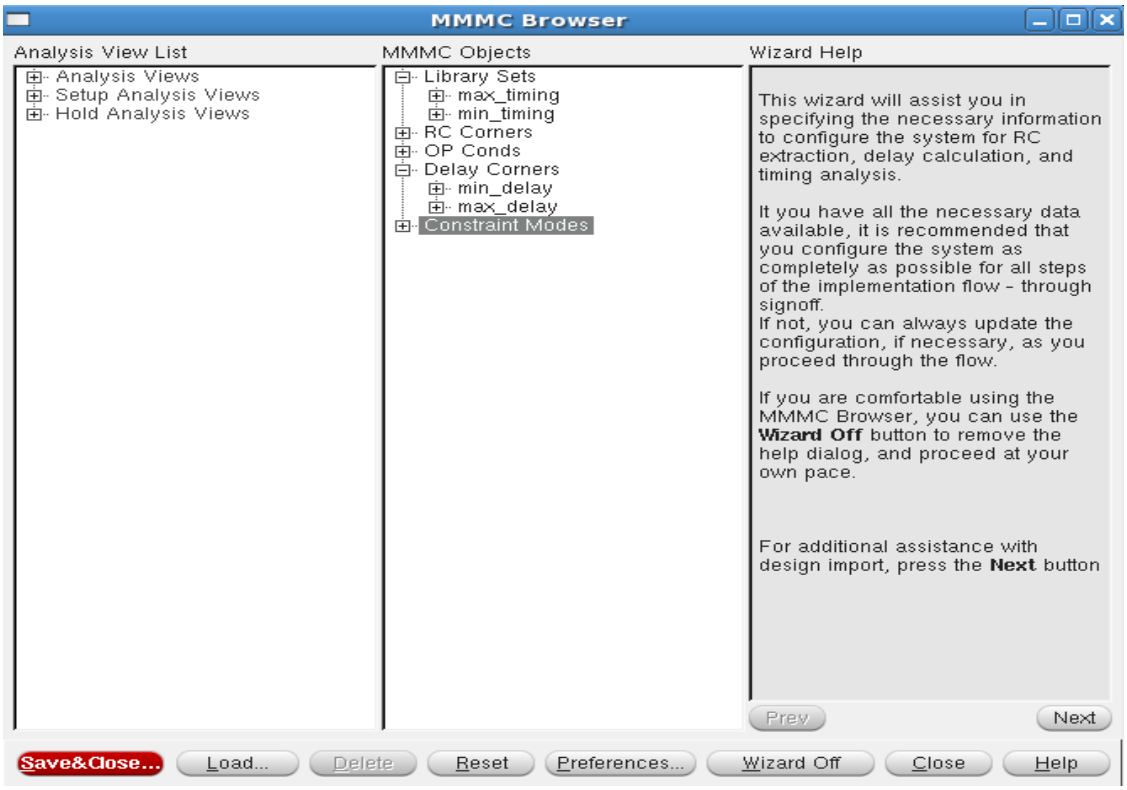
- In the Library Set option under Attributes scroll to min_timing then click on ‘OK’ the below should be the status.



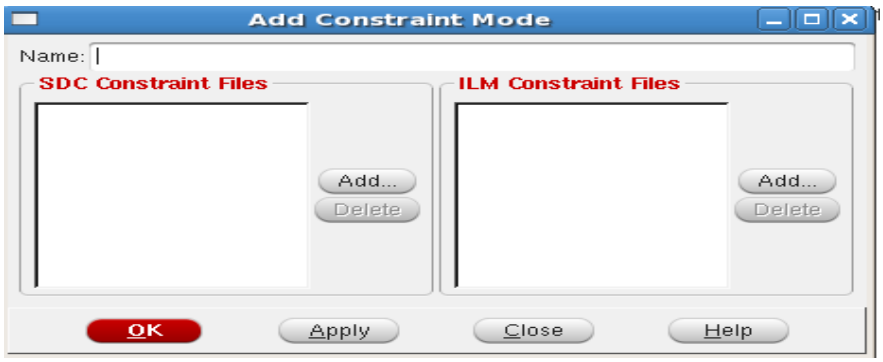


- Similarly again double click on ‘Delay Corners’ and this time give name as ‘max_delay’ and add ‘max_timing’ in Library set then click on ‘OK’.



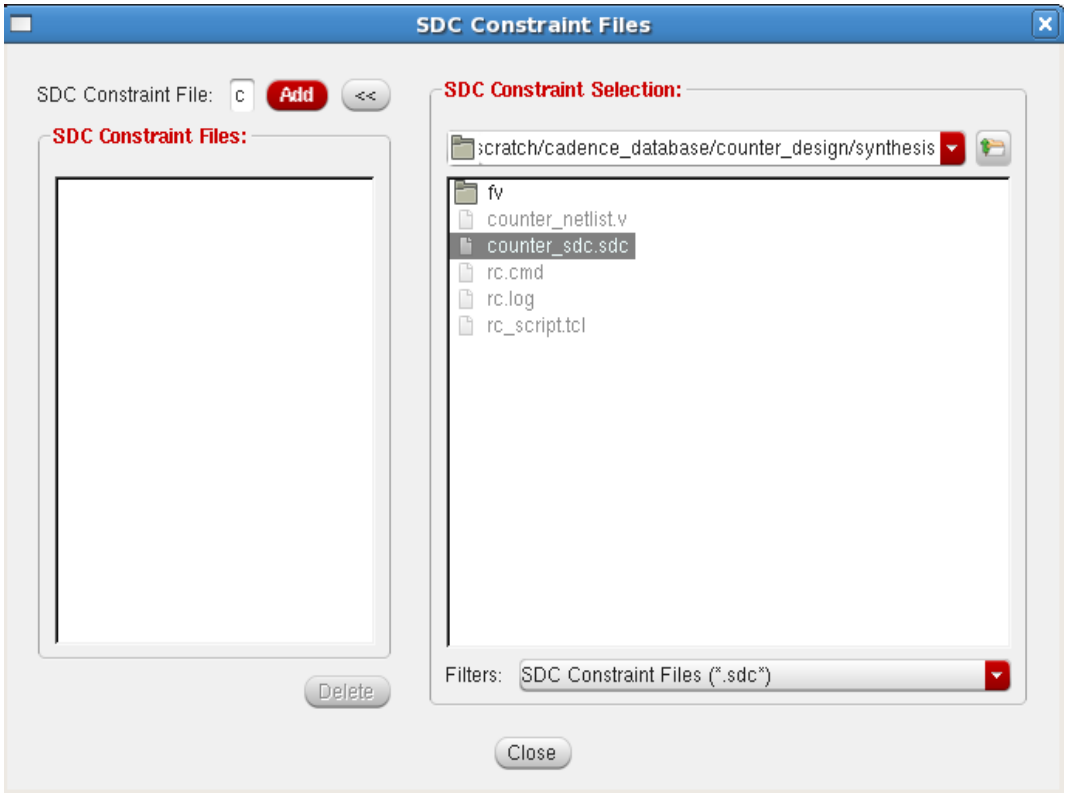


- Double click on Constraints Modes and an ‘Add Constraint Mode’ form appears.

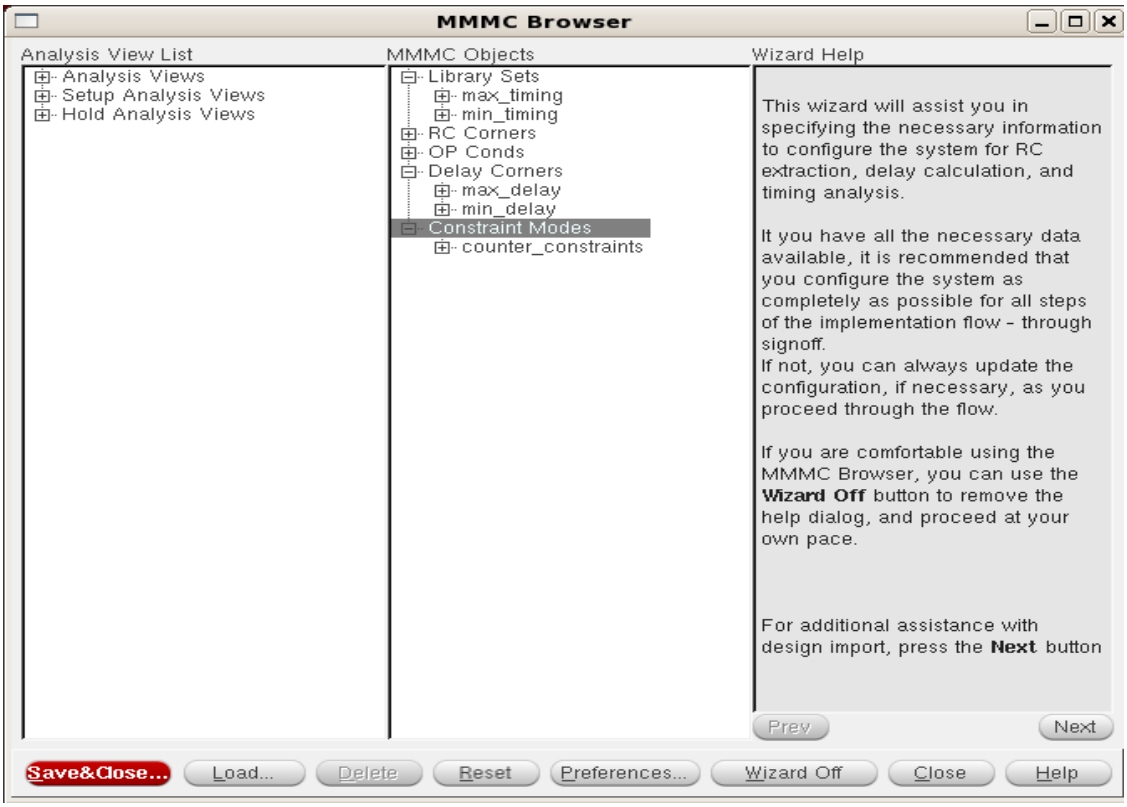


- Give any relevant name for e.g. counter_constraints

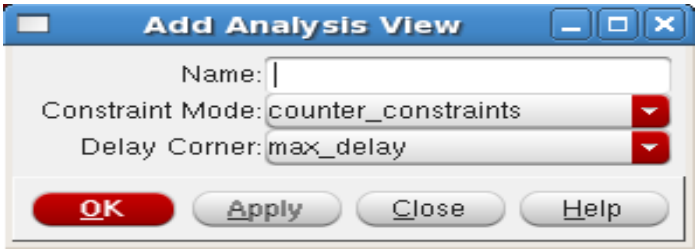
- Click on 'Add' under 'SDC Constraint Files' and browse to your constraints file(.sdc) that you have created from synthesis stage using 'write_sdc' command then click on 'Add' and click on 'close'



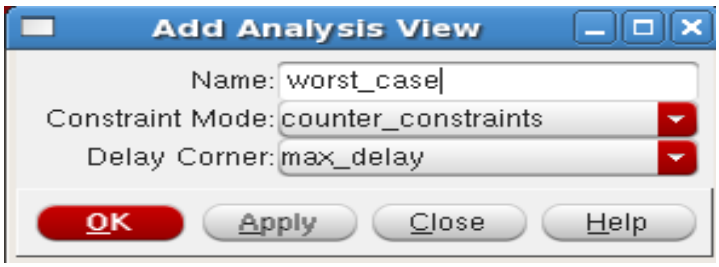
- After clicking on close the MMMC Browser should look like below figure.



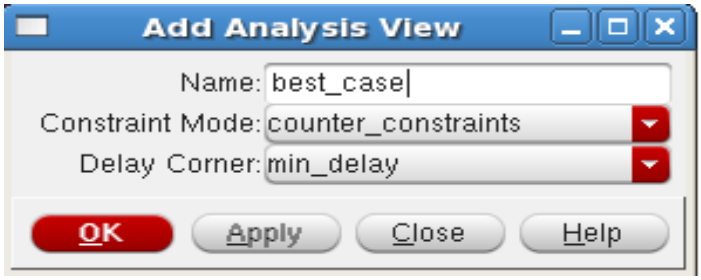
- Now we need to setup analysis for best case and worst case scenario, we will do worst-case analysis for setup and best case analysis for hold.
- Double click on 'Analysis Views', a new Add Analysis View form appears



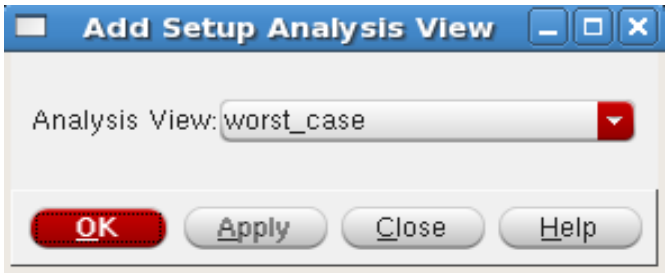
- Give name as worst_case and select max_delay for Delay Corner and click 'OK'.



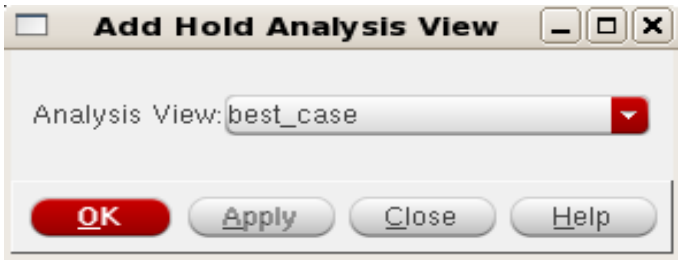
- Again double click on ‘Analysis Views’ a new ‘Add Analysis View’ form appears, give name as best_case ,scroll to min_delay and then click ‘OK’.



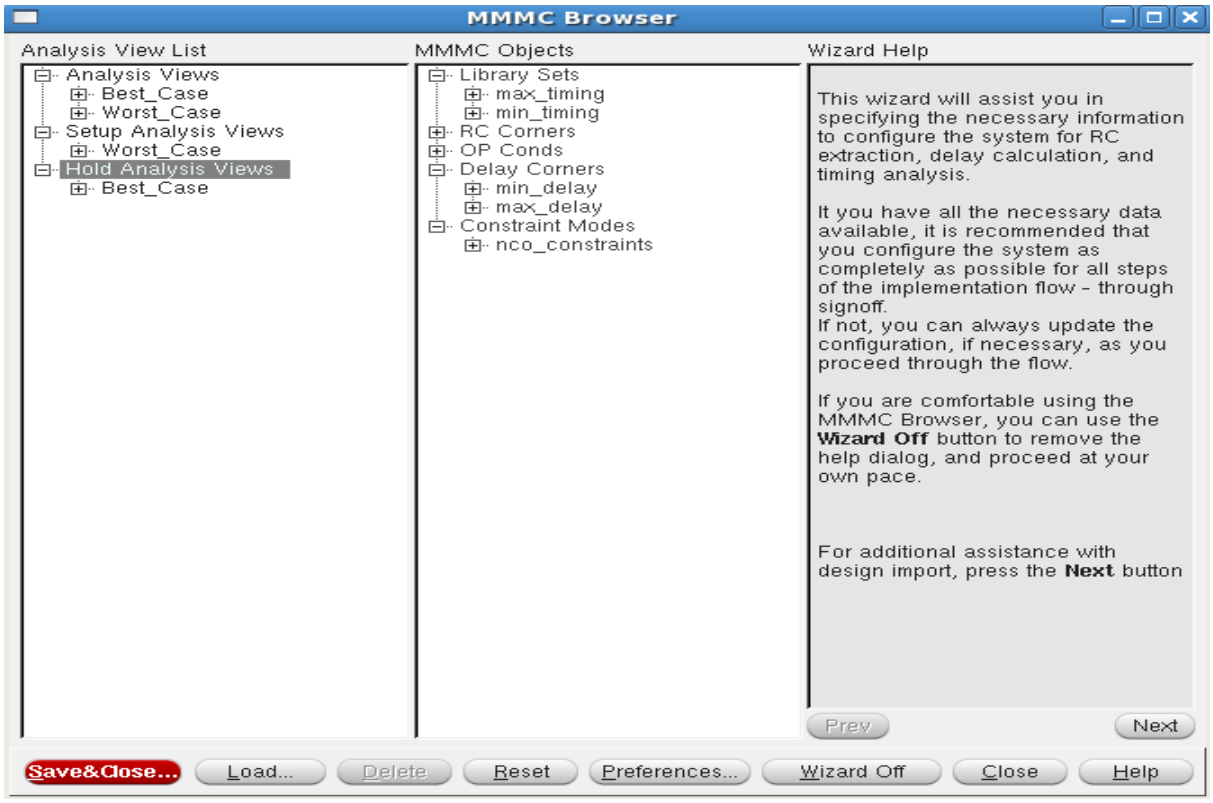
- Double click on Setup Analysis Views , in the ‘Add Setup Analysis View’, the Analysis view should be set to worst_case, if it is set to best_case then scroll to worst_case and click ‘OK’.



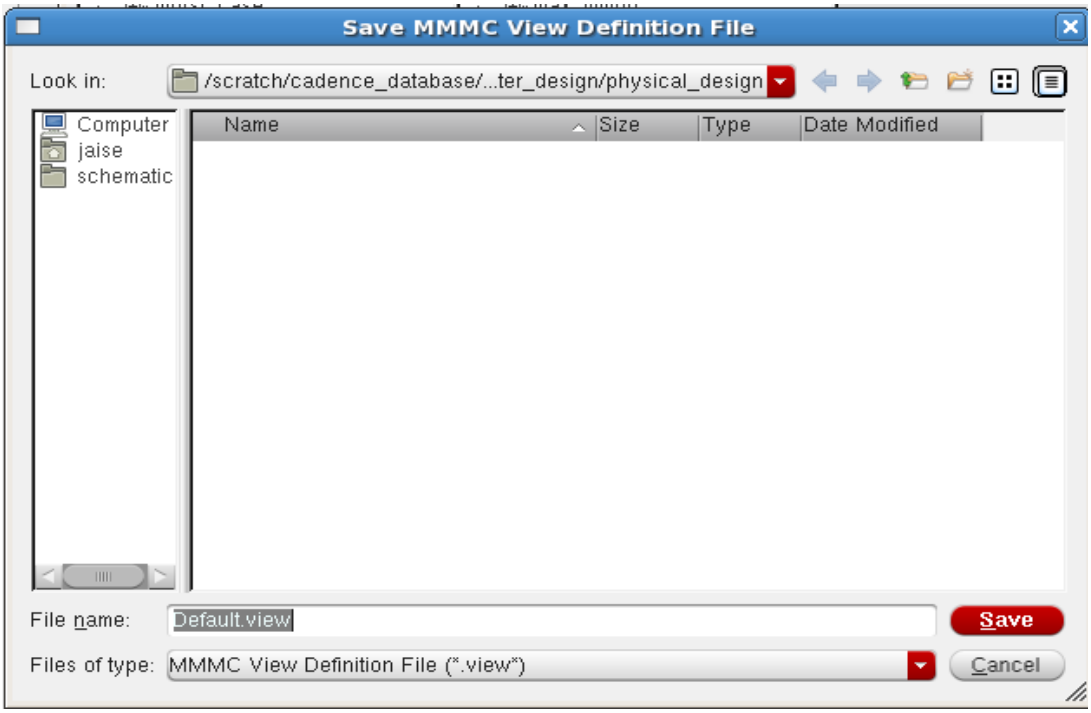
- Double click on ‘Hold Analysis Views’, in the ‘Add Hold Analysis View’ the Analysis view should be set to best_case, if it is set to worst_case then scroll to best_case and click ‘OK’.



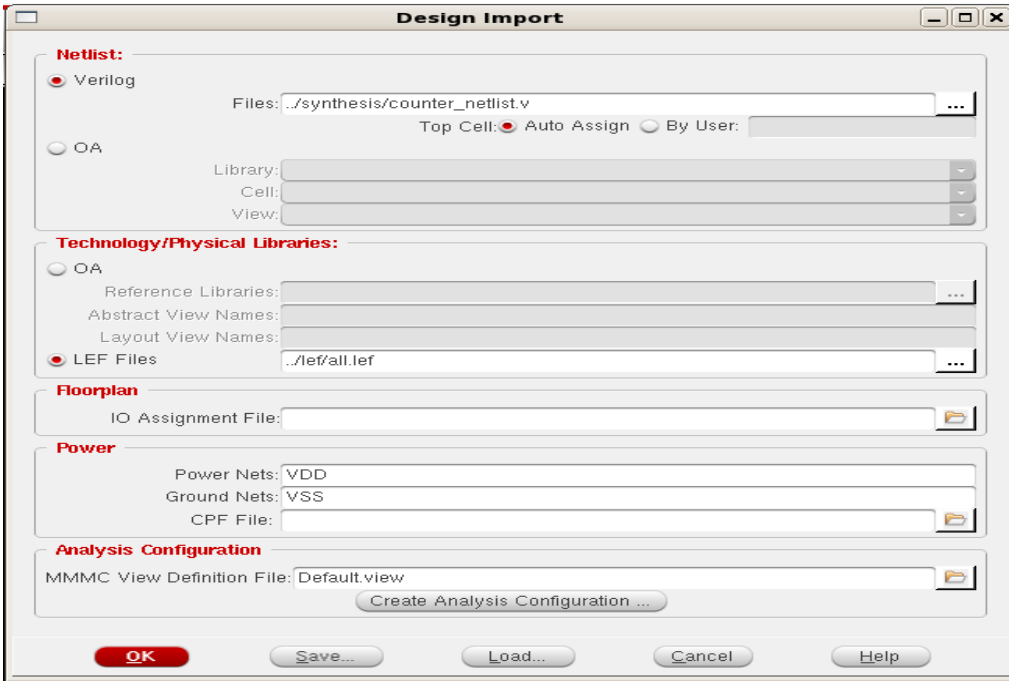
- After clicking ok the MMMC Browser should look like below.



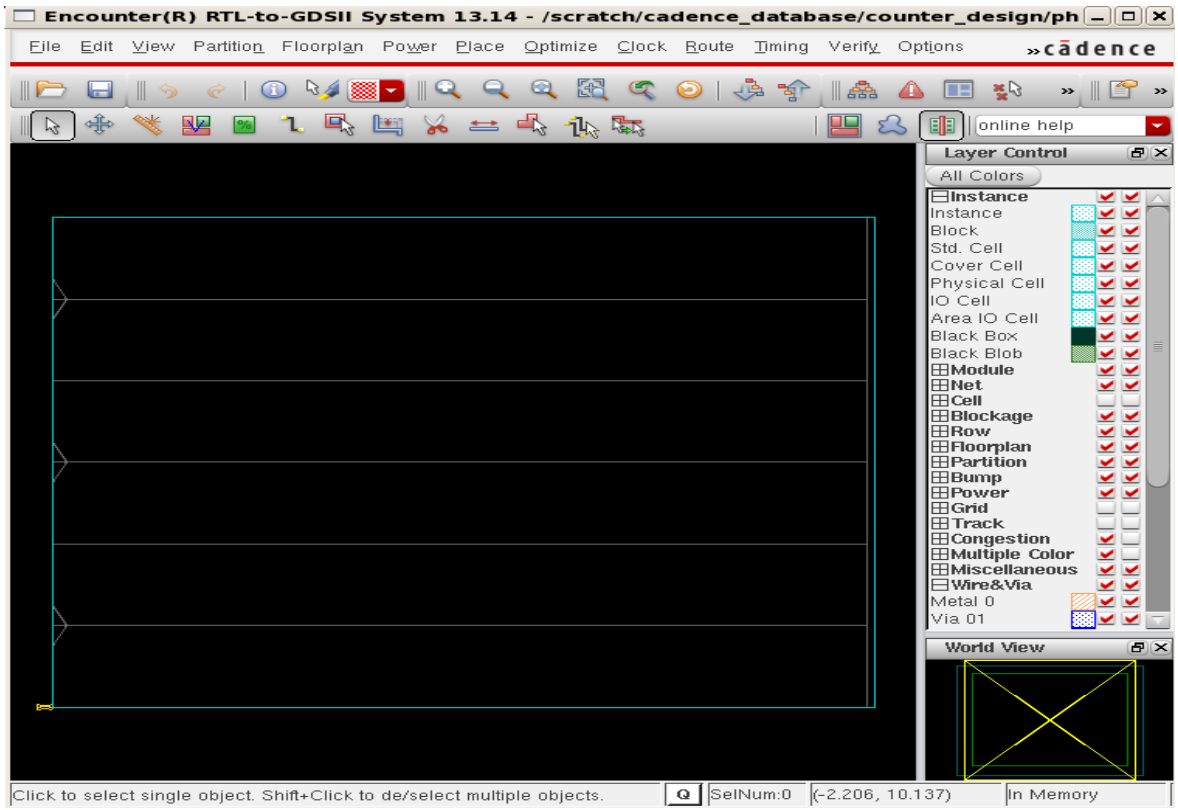
- Click on Save & Close



- After saving the design import form appears and it should look like the below figure and Click on 'OK'.



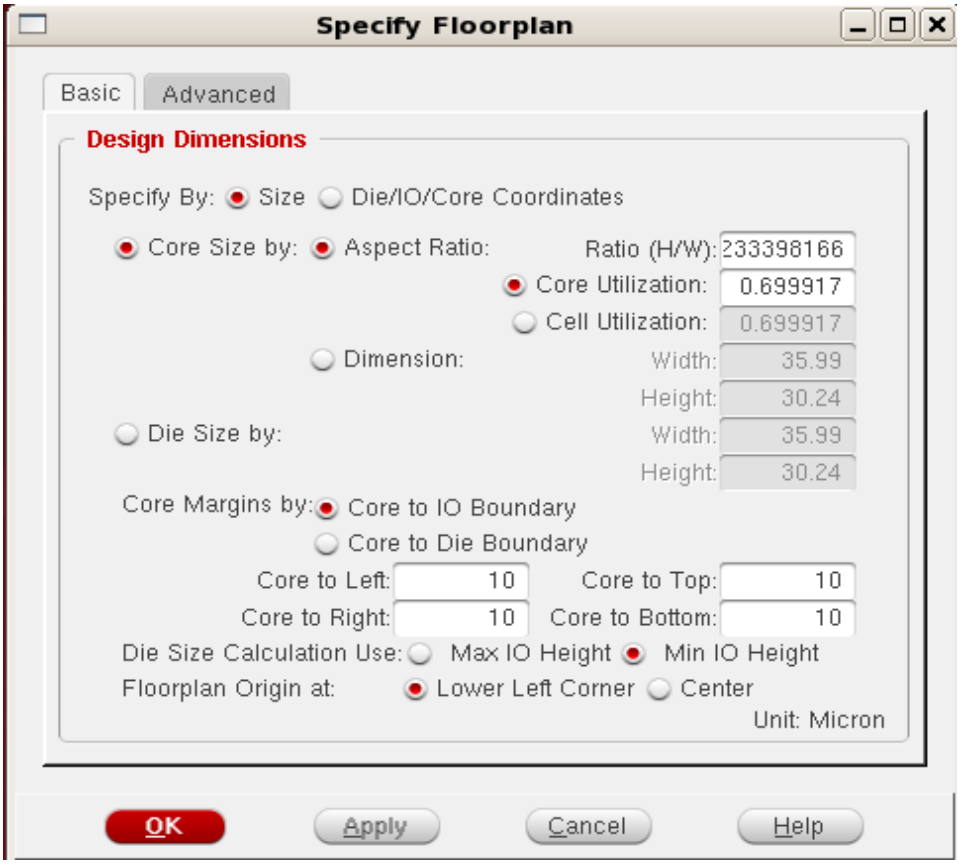
- The below window appears and you can notice at the bottom right corner that the design is in memory now i.e. the design has successfully been imported. Press f (to fit it screen).



Floor Planning

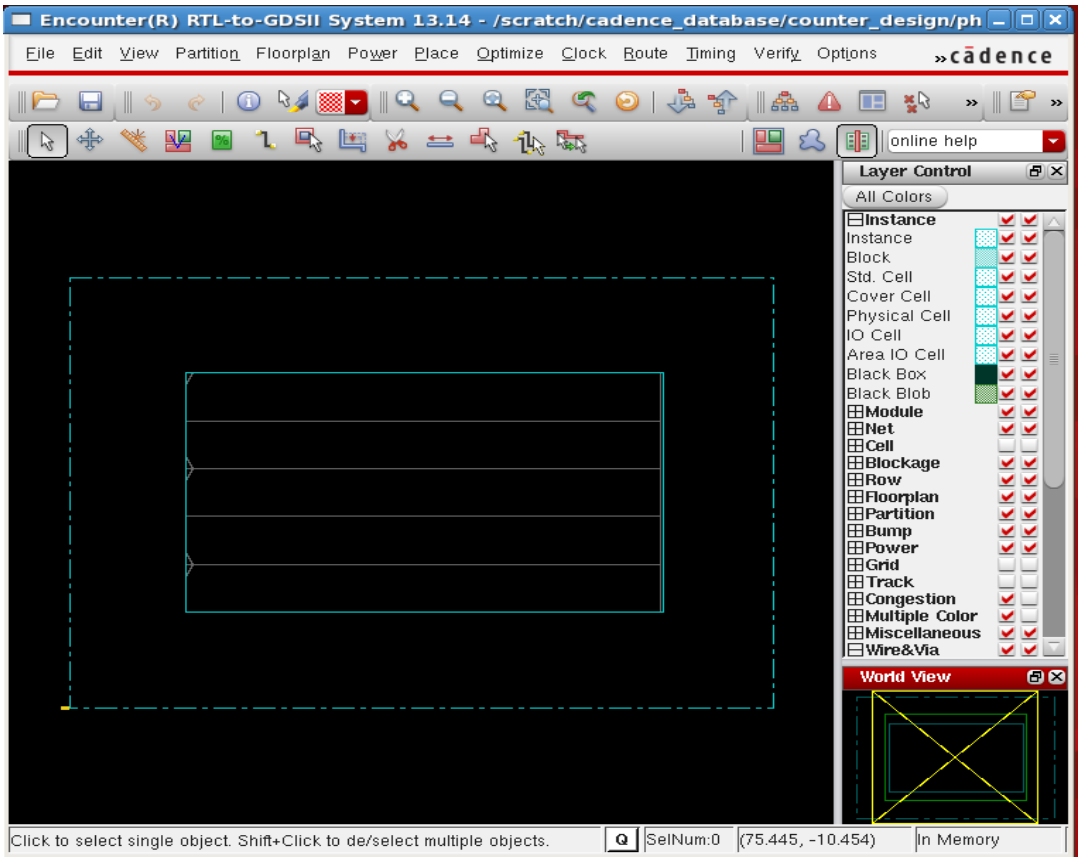
Floor Plan defines the actual form or aspect ratio, the layout will take, the global and detailed routing grids, the rows to host the core cells and the I/O pad cells (if required), the area for power rings, the (pre)placement of blocks/macros, and the location of the corner cells (if required). After the design import, an initial floorplan is displayed in the display area.

- Click 'Floorplan' -> 'specify floorplan' and a 'Specify Floorplan' window will open.



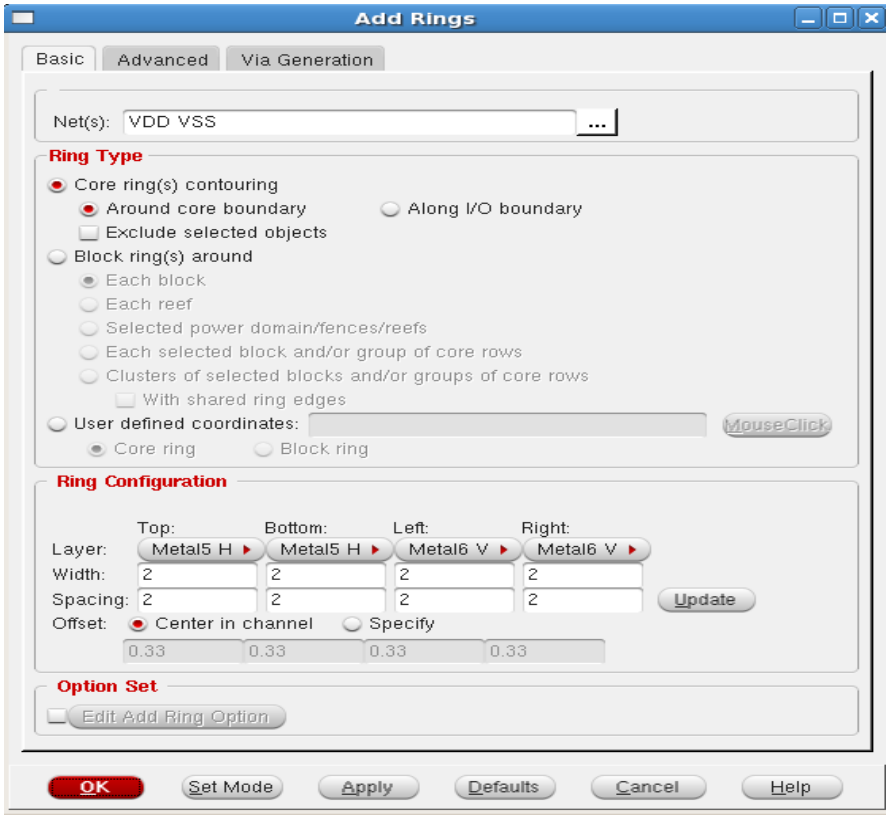
- Select the Aspect Ratio as per the requirement. Give some dimension in "Core to left", "Core to right", "Core to top","Core to bottom". E.g. give 10 to each. This is to create the space for Power rings which will be created in power planning. After defining core area, click 'OK'

- After Floor Planning, the Encounter window will look like the below image.

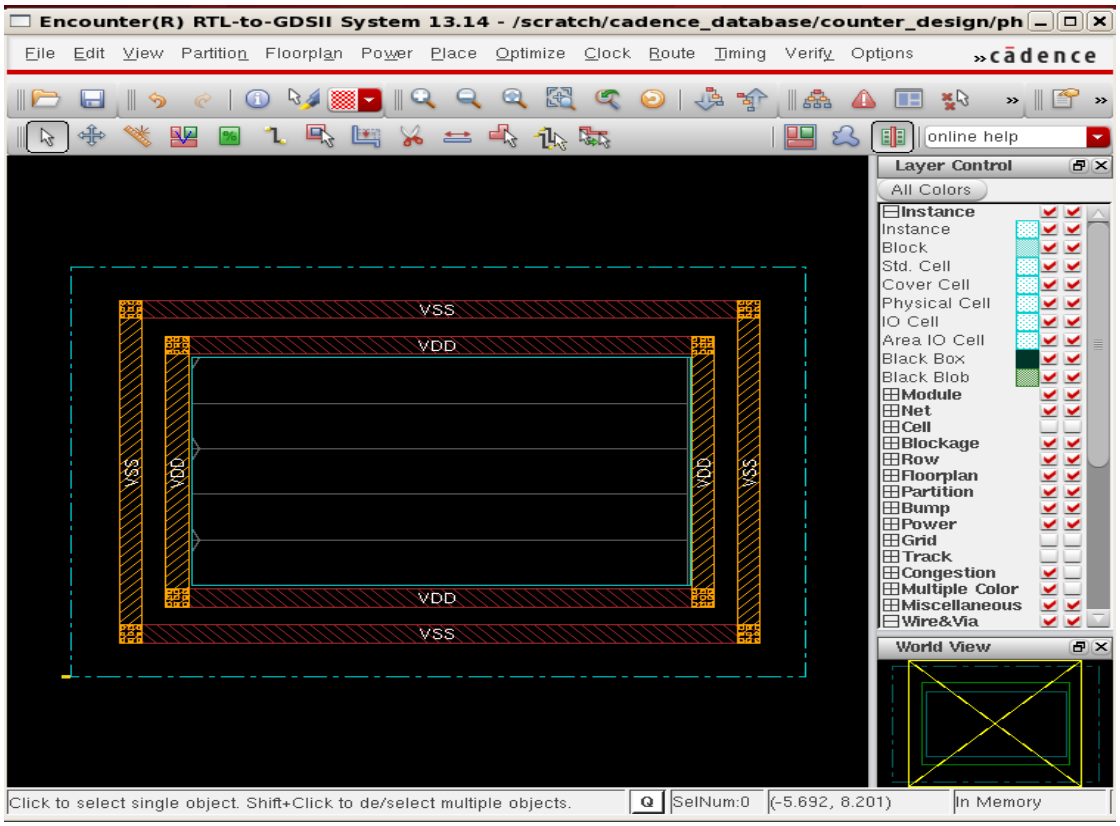


Power Planning

- Click on power -> power planning -> Add Rings and the ring window pops up.
- Select the VDD and VSS power nets under the net option using the browse keys.
- Select the top and bottom layer as Metal5, Left and Right as Metal6 (for 180nm total 6 metal layers top layers are metal6 and metal5. Set the width as per the requirement and taking the space between core boundary and I/O pad considerations. Select the option for offset as “center in channel” and click OK.



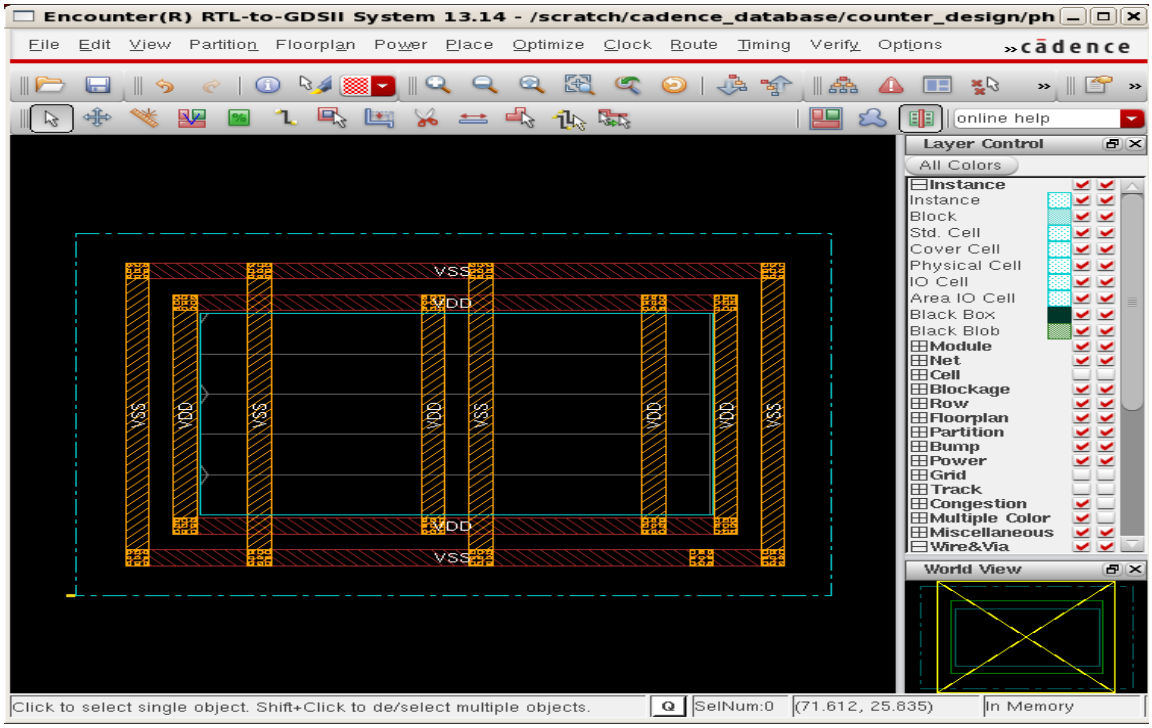
- The power ring will get created as shown in the below image.



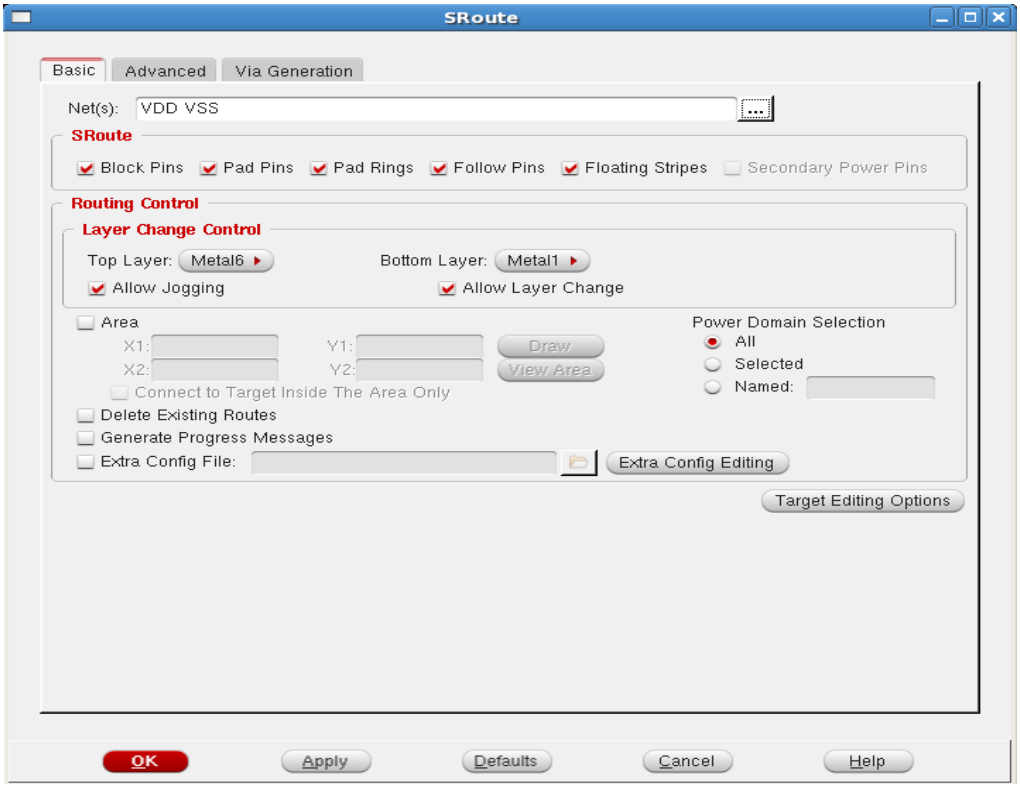
- The next step in power planning is to create power strips. Select Power, click Power Planning and click Add Stripe.

- Select the VDD and VSS power nets under the net option using the browse keys.
- For adding the stripes, select metal layer as Metal 6 and chose direction as vertical (if direction chosen is horizontal, chose metal layer as Metal 5). Click OK and the design will get the vertical thin strips of type Metal 6. Top metal layers are selected for power routing because they compensate very less resistance, so that drop can be reduced.
- Specify the width and spacing between the stripes and also specify the number of sets (of stripes) under the set pattern option. And click 'OK'.

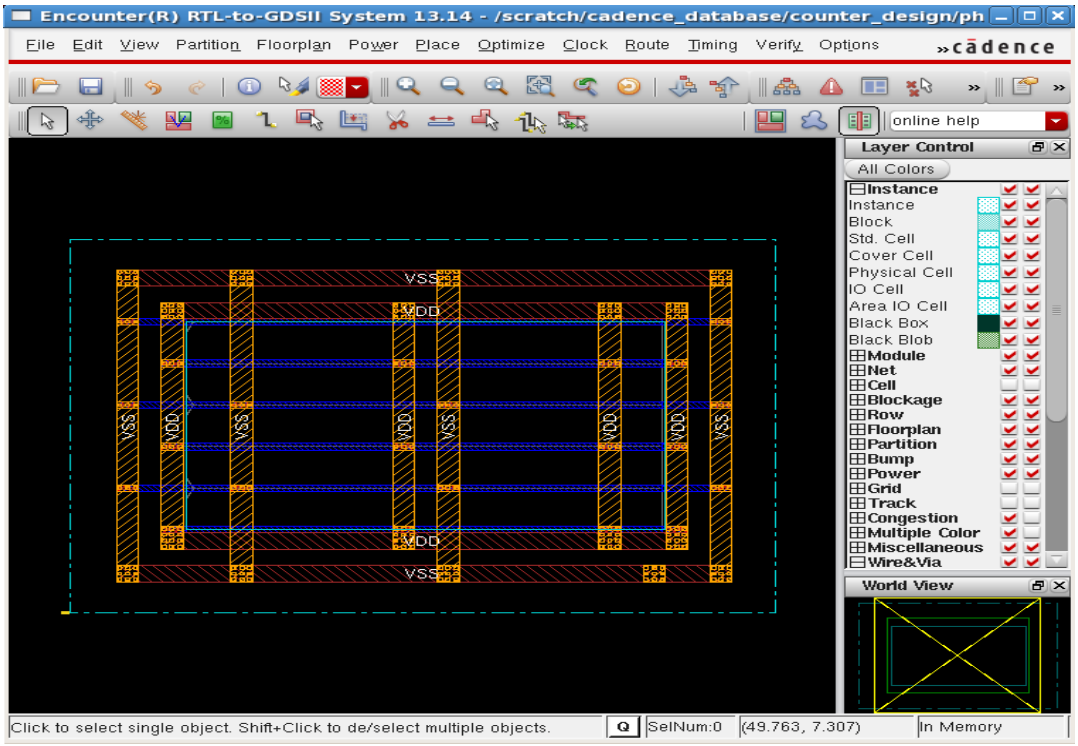
- After adding the power stripes, the Encounter window will look like the image shown below.



- After the power planning, go to Route -> Special Route. A new Window Sroute will appear.
- Select the VDD and VSS power nets.



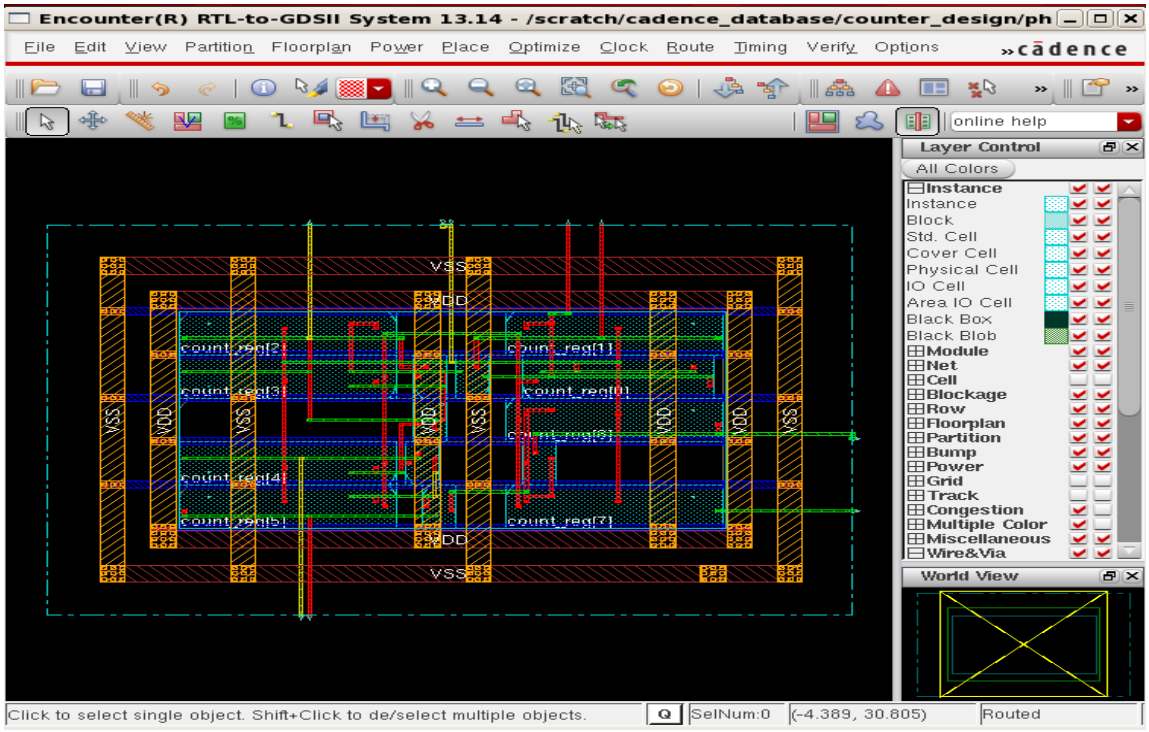
- Click OK with all default settings. This is done to provide power to standard cells. The horizontal blue colored metal1 stripes as shown in the below image are created as a result of Special Route.



Placement

- This step places the standard cells in the rows, according to the imported Verilog netlist.
- Select 'Place' -> 'Place Standard Cells' in the main menu.

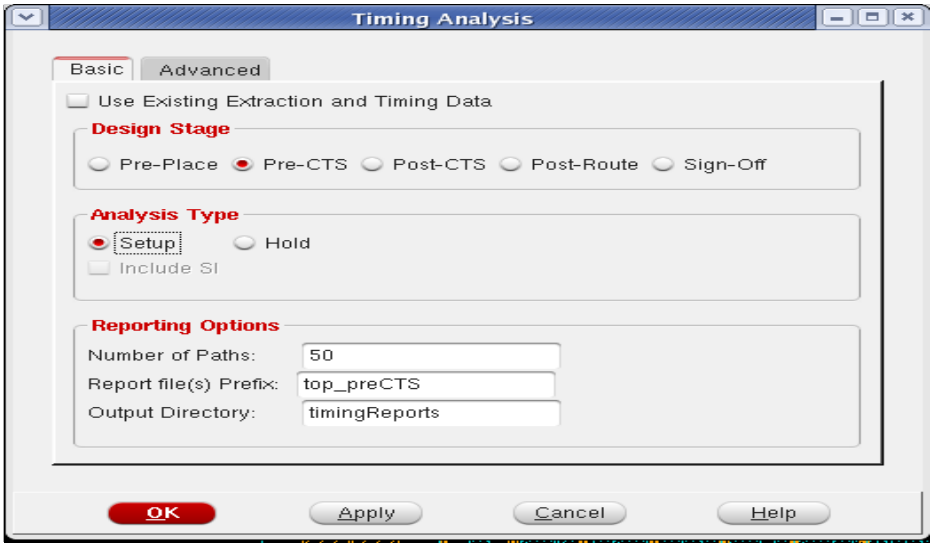
- Select Place > Check Placement... in the main menu to get information regarding Placement density, unplaced and placed cell status.
- Click OK on Place window and in physical view the blue colored standard cells can be seen as a result of placement of standard cells.



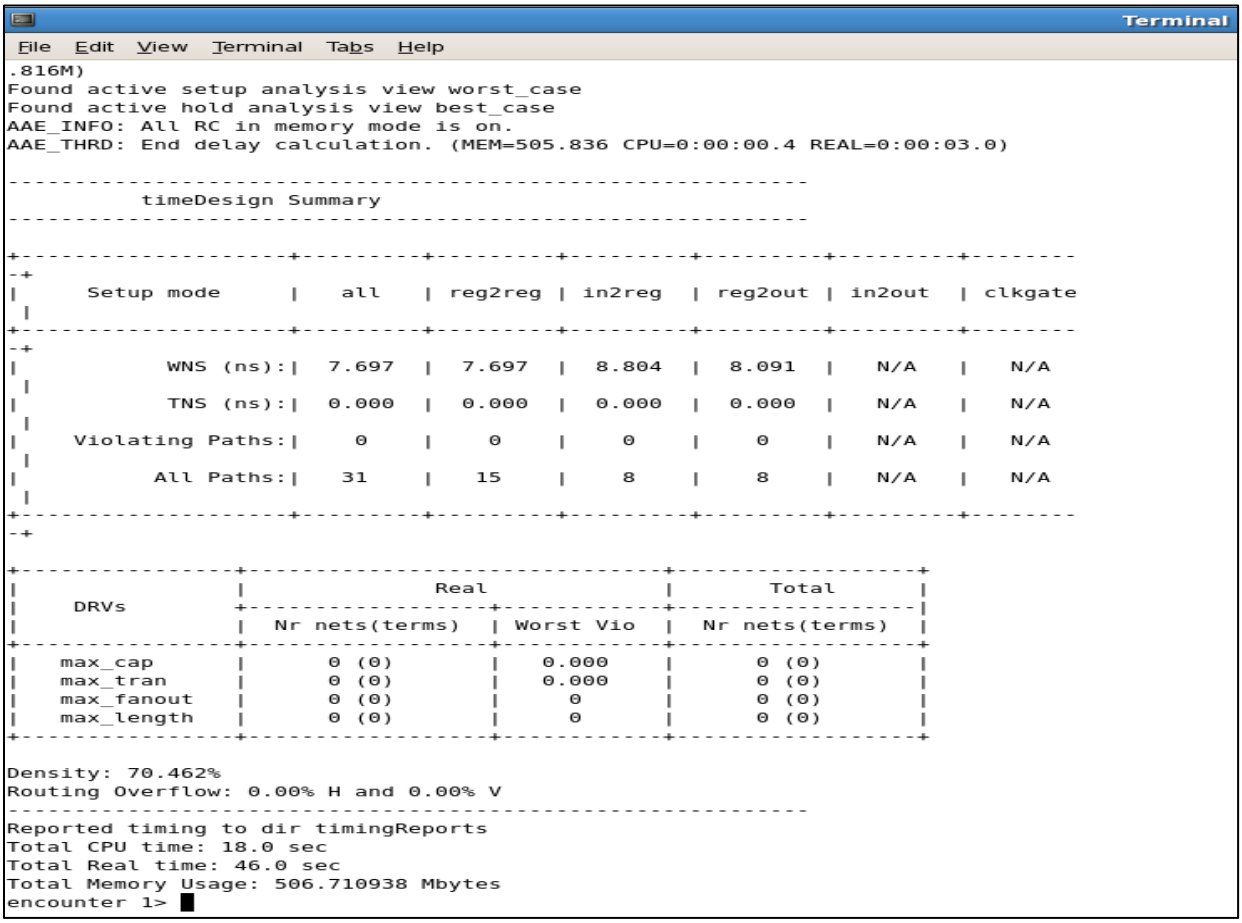
- The multi-coloured lines visible in the tool window are the connections between standard cells using metal layers. If any part of this design is Zoom-in, metal layers can be viewed easily.

Pre-CTS Timing

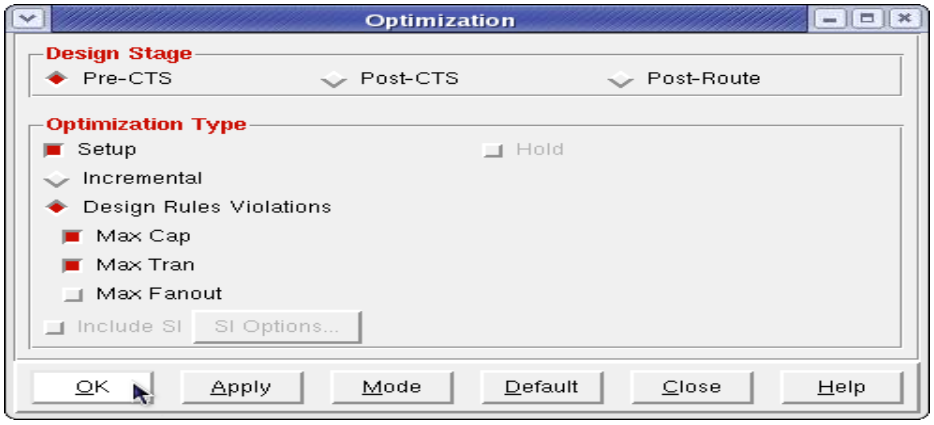
- Before CTS, timing analysis has to be done for any setup violations.
- Click on Timing, and select Report Timing. A 'Timing Analysis' window will get open. In the window select the 'Pre-CTS' as Design Stage and select the 'Setup' as Analysis Type.



- Click OK to complete the Timing analysis. The timing information will get display on terminal in tabular form. In the table displayed on the terminal under “timeDesign Summary”, check for any negative value under WNS (Worst Negative Slack) and TNS (Total Negative Slack). The terminal will look as the image below.



- If there is any of the negative slack value under WNS or TNS, we need to optimize our design.
- Select ‘Optimize’ -> ‘Optimize Design’ in the main menu. Check the Pre-CTS box. The other default selection of boxes asks to correct setup, max capacitance and max transitions violations. The tool will optimize the design and the optimized timing results will be displayed over terminal again.

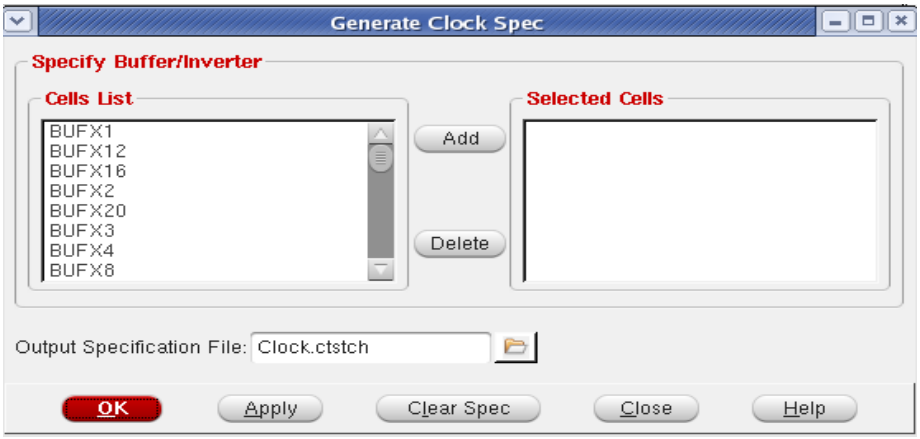


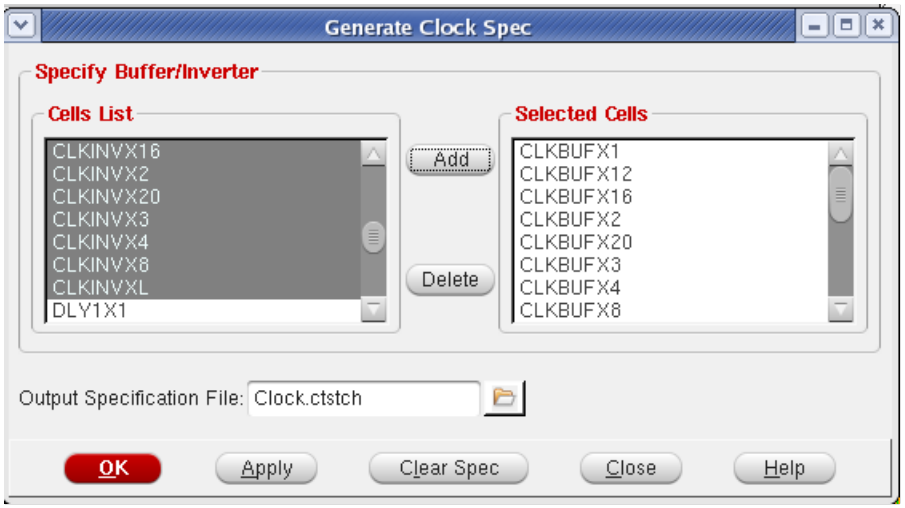
- During optimization the tool will perform following things.
  - Adding buffers
  - Resizing gates
  - Reconstructing the circuit
  - Remapping the logic
  - Swapping the pins
  - Deleting the buffers
  - Moving the instances
- Once you done with timing analysis tool internally dump out timing report directory.

Clock Tree Synthesis

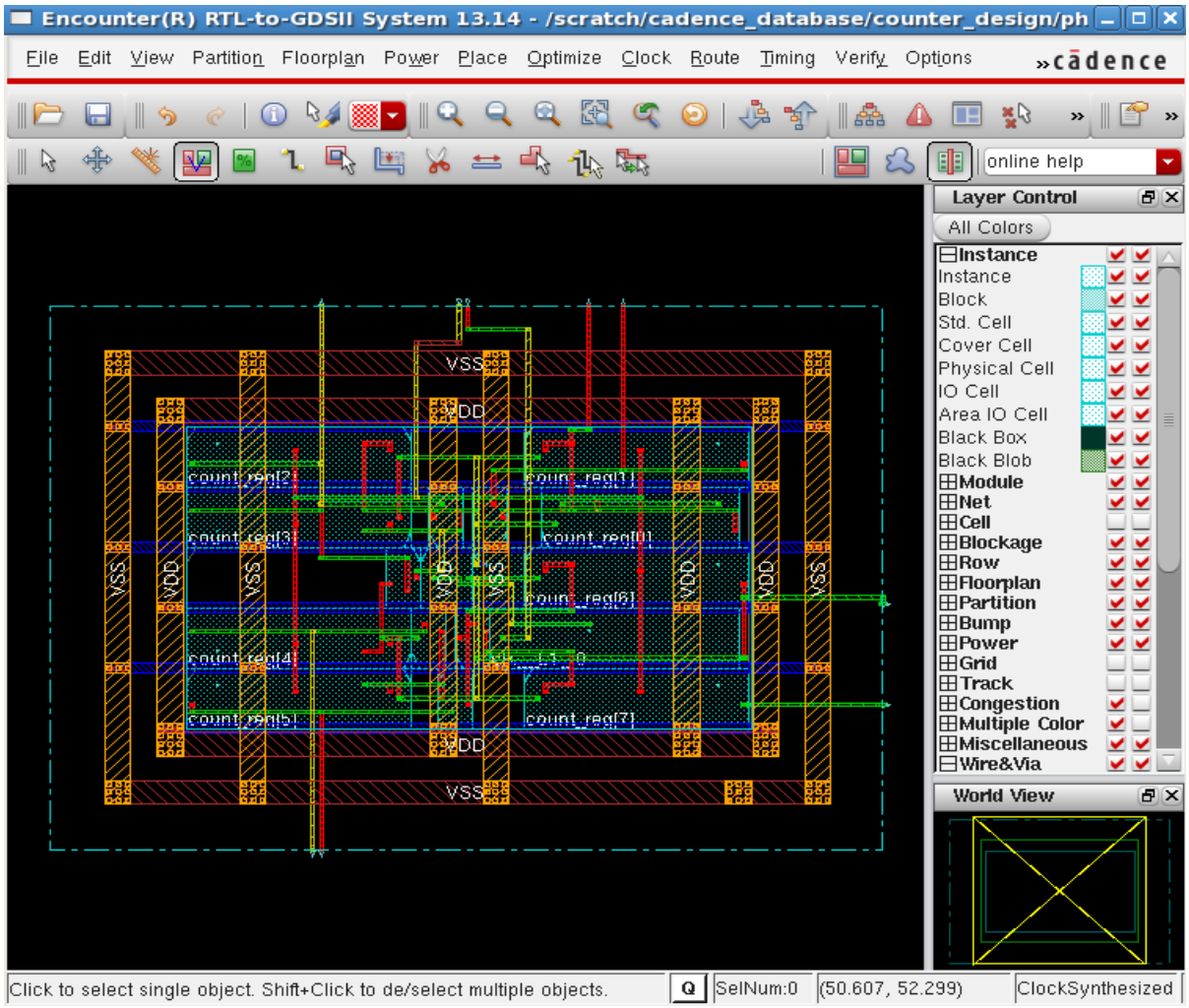
- As the paths that will propagate the clock signal in the design are not necessarily balanced, some registers may receive the active clock edge later than others (clock skew) and may therefore violate the assumed synchronous design operation.
- To create a balanced clock tree, you have first to create a clock tree specification file. Encounter can create a first draft version of the file you can then edit to specify design specific data.
- Select 'Clock' -> 'Synthesis Clock Tree...' in the main menu. Then, in the Basic tab, click the 'Gen Spec' button. A new window "Generate Clock Spec" will open.

- From Cells List, Select all cells starting with "CLK" and click on Add button to add them to the Selected Cells. Select a name for Output specification and click 'OK'.





- Specify a name for 'Results Directory' and click OK. The tool window looks like the image below.

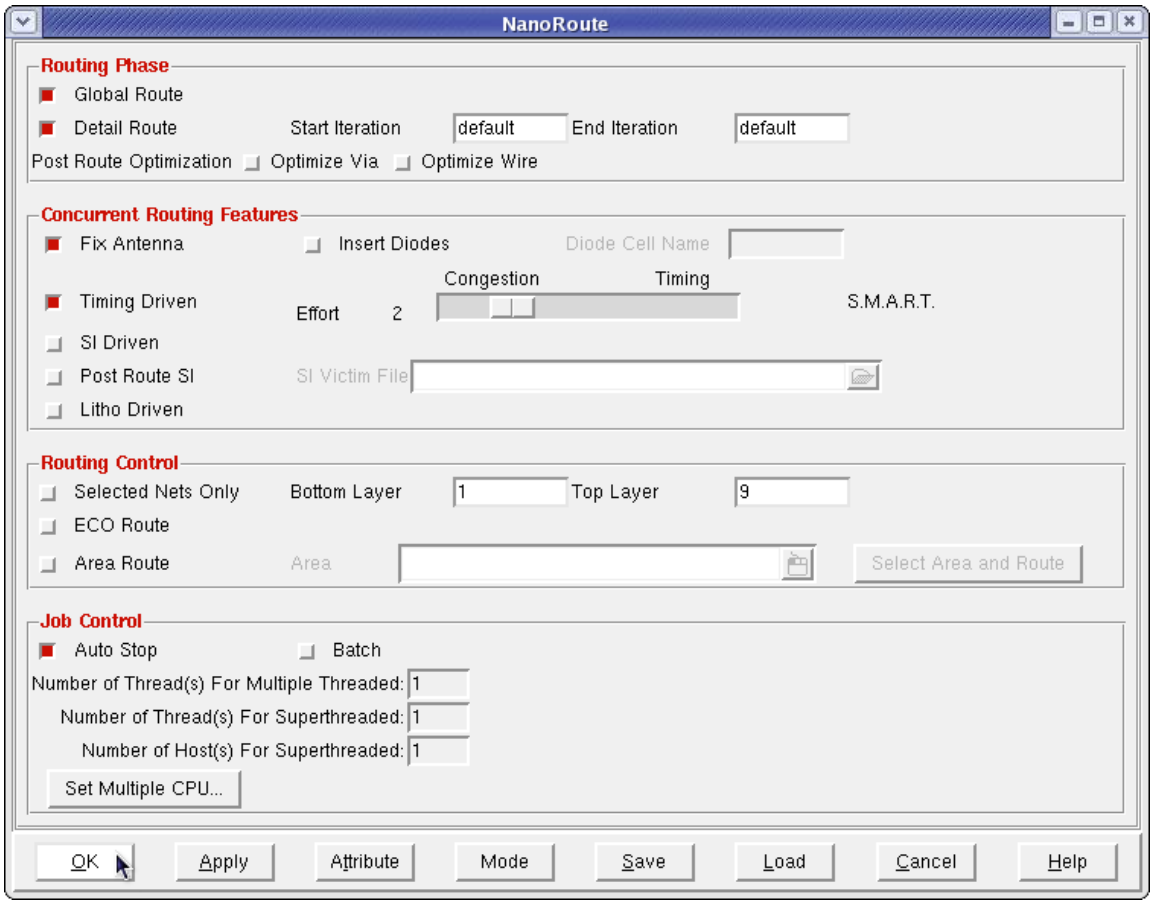


Post-CTS Timing

- Again Perform the Timing by clicking on Timing and selecting Report Timing. Select 'Post-CTS' under Design Stage and do the select 'Set-up' as Analysis Type. Check the terminal for timing violations and optimize the design if any violation exists.
- Similarly check for Hold violations and optimize if violation exists.

Routing the Design

- This step generates all the wires that are required to connect the cells as defined in the imported Verilog netlist.
- Select 'Route' -> 'NanoRoute' -> 'Route' in the main menu. Check the Timing Driven box. A higher value increases the effort toward meeting the timing constraints and decreases the effort toward relieving congestion. Click OK to start the routing.



- The tool will perform the Routing and the Routing statistics can be seen on terminal window including DRC violations.

- Perform the timing analysis again to check for violations and optimize if any.

Physical Verification

The Verify menu has a number of items to check that the design has been properly placed and routed.

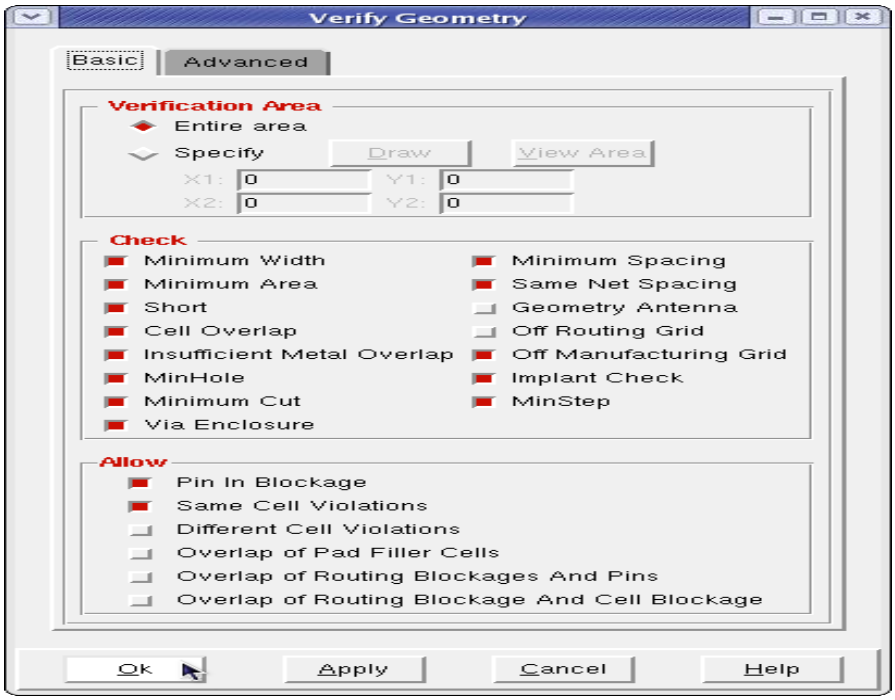
Verify connectivity

- Select ‘Verify’ -> ‘Verify Connectivity’ in the main menu. Click ‘OK’.

- If you face any violation and check it out those violations to tools->violation browser

Verify Geometry

- Select 'Verify' -> 'Verify Geometry' in the main menu.
- If you face any violation and check it out those violations to tools->violation browser



Generating Stream file

- Once everything has been completed next stage is generation of GDSII file. Select 'save'-> 'gds'.