APR FLOW

Layout Team

Yousry El-Maghraby
Mohamed Enany
Moataz Kadry
Haitham Ghoniem

March 2014
Version 1.2

- # Creating and preparing the input files

1- CD into your project SOS directory
2- If you can't find the "digital" directory after updating your SOS, create it and check it in the SOS
3- In the "digital" directory create a directory called "APR" then check in the SOS as well
4- The reason for checking in the project/digital/APR directory into the SOS is to avoid deleting the directory if another engineer created it and checked it in the SOS then you did an update ☺
5- In the "APR" directory create new directory by the top level module name of your digital block, here we will name it as "topModule"
6- In the "topModule" directory, create new directory named as "vx_date" (Example v2_041513)
7- In the directory */Project/digital/APR/topModule/vx_date/* ,Use the script "new_apr topModule"[1] to generate the whole APR structure as shown in figure (1)[2].
8- By now you have created the APR directory structure as below

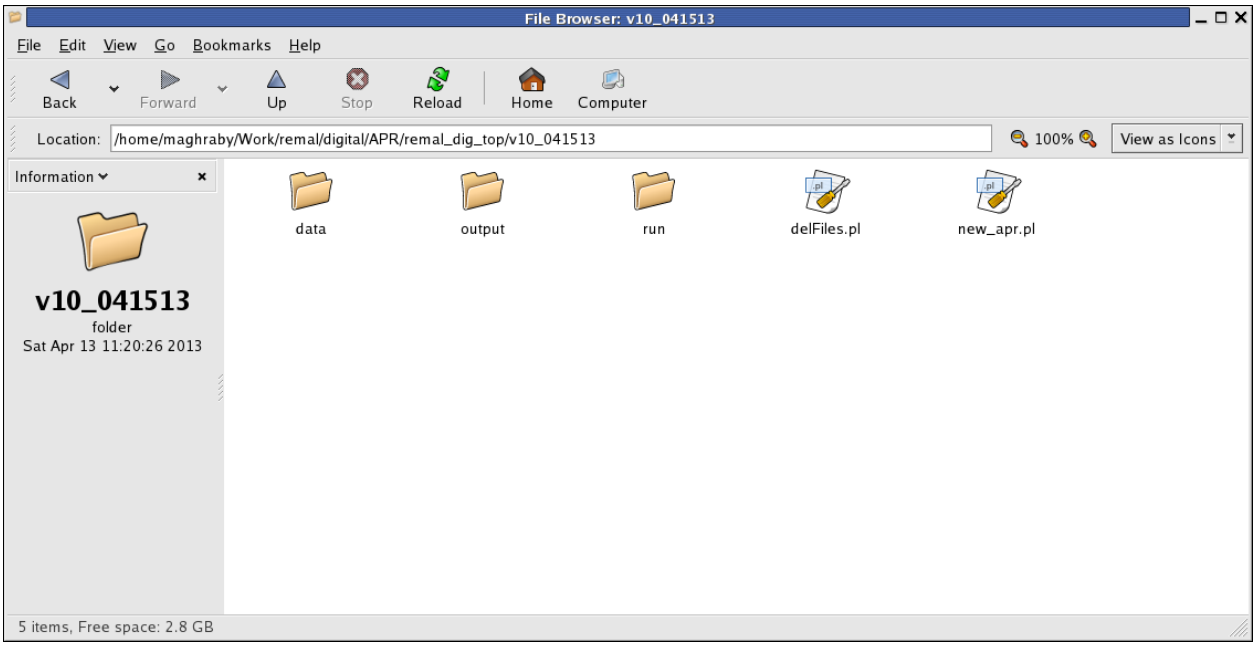

Figure 1 APR folder structure

9- Go to the data directory "/Project/digital/APR/topModule/vx_date/data", fill all the APR input data:
   - **Source File** directory define all the available lib file paths at different corners and the lef files as in the setup.cshrc file
   - **Config file** add the same variables in the source me file at the beginning. Then add the lib/lef variables in the corresponding lines in the file, Note that the min lib file corresponds to the fast corner and the max lib file corresponds to the slow corner, You should choose the most slow and fast corners (High temperature, low supply is always slower), At the end of the config file the vdd/vss names is defined, typically it's recommended to have the name vdd_ topModule, vss_ topModule, In case of multi-power domains, the default

[1] Whenever you need to delete the output and run directories contents you can use the "delFiles" script
[2] Please refer to Appendix 1 to know the content of each folder and the function of each file

module should have the names (vdd_ topModule, vss_ topModule) while the non-default supply connected to the module X fill be named as (vdd_X, vss_ X), Make sure that the supply pin names are compatible with the .vg file otherwise contact the designer to modify them

- **map.map file** get the layers number and data type from (The technology dump, technology document, technology directory, or the PIPO log file which is the most convenient if available)
- **procedure file** it's not generated automatically, simple you can copy it from other project and replace the top level module name by "topModule"
- You should of course place your netlist and sdc files in the "vg" and "sdc" directories
- In the sdc file make sure it does not include a library specific command, also it should be free of the "Set ideal network" or "Set don't touch" commands
- The directories "cts" and "icon_FP", just leave them now and we will fill them later.

10- Source tools
11- CD into the directory "Project/cadence" (May be "oa") and source the setup.cshrc file
12- CD into the directory "/Project/digital/APR/topModule/vx_date/run", it should be initially normally empty
13- Invoke Encounter "encounter -vdi"[3]
14- You can always monitor the encounter.cmd* that includes all the commands done in your run either through command line or at GUI and encounter.log* for all the errors and warning and commands outputs. Those files are generated in the run directory directly.
15- The file encounter.cmd* is very useful when creating the procedure file.
16- Now you can see encounter window

[3] Don't open encounter in background mode (&) so as to be able to trace the errors and warnings on terminal

• Importing the design into Encounter

1- Define the lib/lef file paths variables by sourcing the source me file:
   - *source ../data/sourceFile/source_me*
2- Load the config file and make sure no syntax error are there (Always monitor errors in the log file):
   - *loadConfig ../data/conf/remal_dig_top.conf 0*
3- Commit the file to import the design
   - *commitConfig*
4- The design is now loaded into encounter as you can see in figure 2.

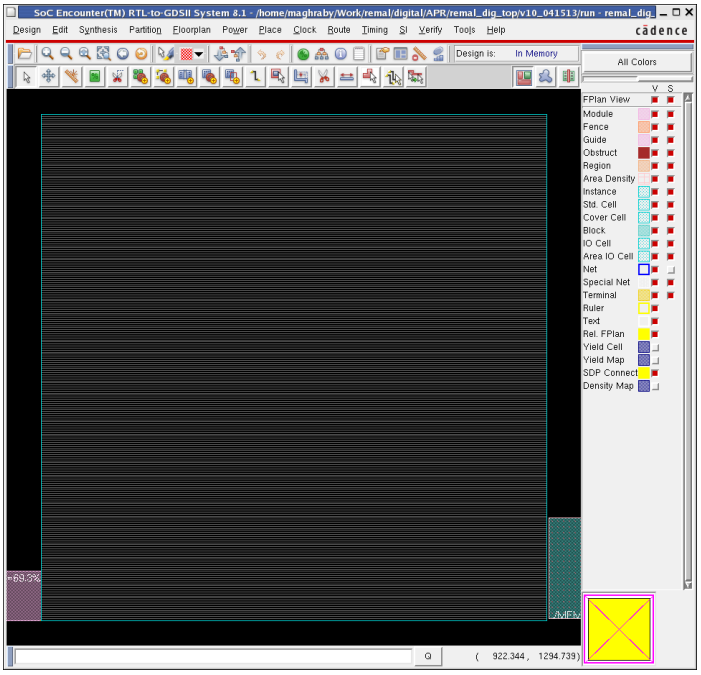

Figure 2 loaded design in encounter

• Design the floor plan

1- First step is to design an icon view at Virtuoso. The icon view should be defined by the drawing prboundary layer (Any shape can be used), each pin has pin layer square with center on boundary, and drawing layer square with center on boundary, plus a pin layer label with cross on boundary
2- Now you can run the getPins script from the skillXL menu and export the file at: /Project/digital/APR/topModule/vx_date/data/icon_FP/topModule.icfb
3- Concept wise each of the icon view pins may be on different layer, however the getPins script is now designed to read single layer, so until this is enhanced you can place all pins on the same metal layer
4- After that go back to Encounter, then from the GUI go to Floorplan/specify floorplan
5- In the specify floor plan window enter the maximum width and length of your floor plan in the fields dimensions width and height then OK

6- From the GUI as well go to the floor plan view and select the cut rectilinear icon, use it in coordination with the (move/resize/reshape) icon to design the same shape as your original icon view
7- Now you designed the icon view on Encounter as shown in figure 3.

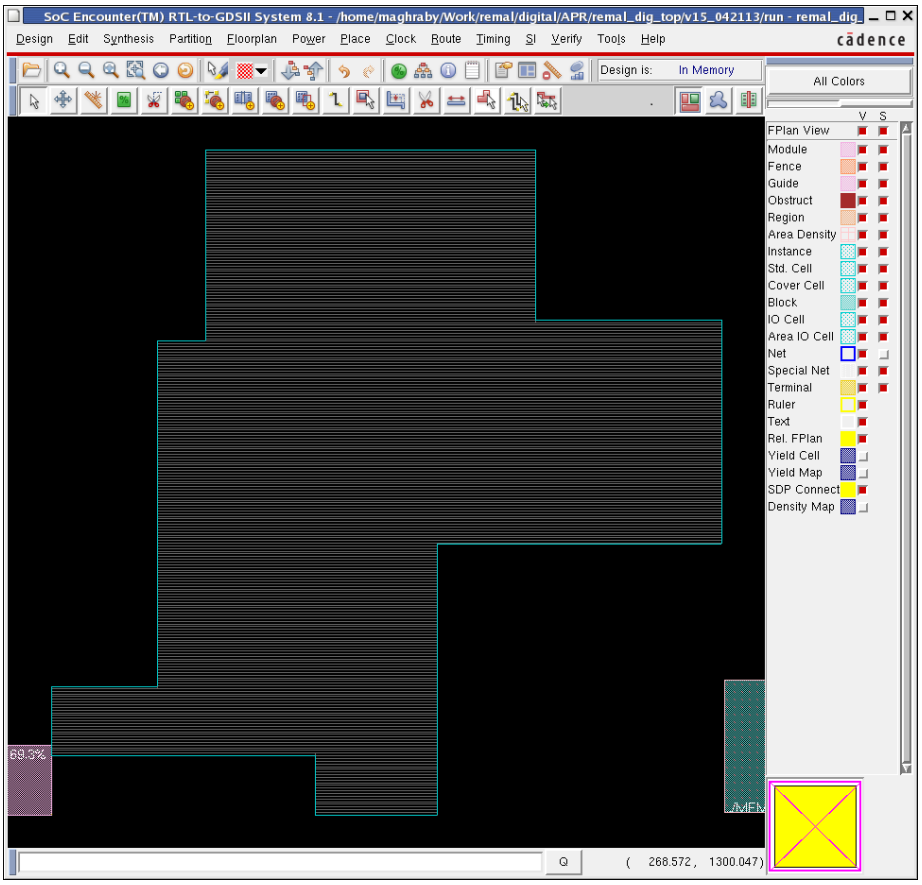

Figure 3 Floor plan design

8- Save the def file you have designed using the below command:
    - *defOut -floorplan ../data/icon_FP/ topModule_nopin.def*
9- Always remember that at Encounter the origin is at the lower left corner
10- Normally all pins are placed at the origin
11- Now open a terminal at the directory: /Project/digital/APR/topModule/vx_date/data/icon_FP
12- Use the script icon2def[4] to modify the def file for you true pins locations:
    - *icon2def topModule_nopin.def topModule.icfb topModule.def METAL_HORIZONTAL METAL_VERTICAL PIN_SIZE*

METAL_HORIZONTAL is the horizontal metal name as in the LSW window.
METAL_VERTICAL is the vertical metal name as in the LSW window.
PIN_SIZE is the min width of metal in the technology multiplied by a factor "y". For example, if the ratio in encounter's def file (the ratio between the coordinates in

[4] In the script icon2def, in case you don't have rectangular shape then you need to add the x coordinates of the vertical edges and y coordinates of the horizontal edges in the if condition that check for the pins direction ("N" for the horizontal edges of pins at bottom, "S" for the horizontal edges of pins at top, "E" for vertical edges of pins at left, "W" for vertical edges of pins at right) – Of course this is an enhancement room for this script

encounter's def file and the coordinates in the icfb file) is 2000 then y=2, another example if the ratio is 1000 then y=1

13- After running the script it will generate a new def file[5] "topModule.def" that can be loaded back into Encounter using the below command
    - *defIn ../data/icon_FP/ topModule.def*
14- Looking into the floor plan view you should notice that you have no more pins at the origin, this important to check that all pins where constrained.
15- Now you got the same icon view as in the one you did at Virtuoso, but still you can help Encounter doing the floor plan.
16- You can notice all top level module instances are placed as pink boxes in the floor plan view, you can just move/resize/reshape those blocks and organize within your boundary.
17- Once you select any of the top level instances you can see Encounter drawing it's connectivity to the different blocks and pins, this should guide you while doing the floor plan
18- You can specify whether the cell under constraint is guide/region or fence. Guide is a flexible constraint at which cell can go out and other cells can move in, region is harder constraint where cell cannot move out but other cells can move in, while fence is much harder where cell cannot move out and other cells cannot move in.
19- In case you are using IPs (RAM, OTP, … etc) you will find it placed
20- You can also add placement or routing blockage in your floor plan and move/resize/reshape those blockages to fit your desire.
21- Any routing blockage area can be edited to block specify layers and allow other, you can add many of them upon your top level needs
22- I's useful if you have many pins routed in small length at certain edge to block the placement near those pins

Power planning

1- Multiple power domains may be done using the CPF (Common Power Format) file and imported as below.
    - *loadCPF ../data/icon_FP/ topModule.cpf*
    - *commitCPF*
2- You can consult the below document for writing the CPF file (Template can be used from Remal project)
    - */home/maghraby/CPF_guide/cpfguide.pdf*
3- At the same directory you will find a model cpf file

    a. *At the cpf file begin you will define the power domains using the "create_power_domain" command, for all domains you should define at least one instant to be included in this domain except for the default domain that will include all uncategorized instances at the top level*

[5] You have to modify the def file by removing the lines including metals that are not required to be used in routing (example: to exclude routing using M5 and M6)

b. *Then for each of the defined power domains it can operate in different voltage level thus will need different lib files, you will define the lib file that corresponds to the cells in that domain using the command "*define_library_set*"*
c. *Follow other steps to define and assign power conditions for different domains using the commands "create_nominal_condition", "update_nominal_condition", "create_power_mode", "update_power_mode"*
d. *Then create the power and ground nets and assign them to the required power domains using the commands "create_bias_net", "create_ground_nets" and "create_global_connection"*
e. *At the end you may define the required isolation cells rule using the command "create_isolation_rule"*

4- After loading the CPF you should see each power domain marked with different color except for the default power domain
5- You can save the current def file so in next iterations you don't need to do all previous sequence once more, you need only to load the current def file.
6- Keep in mind that all shapes are feasible but extra complicated shapes may lead to routing congestion, in case you found routing congestion in certain block you can increase its area in the next run, thus you may need to readjust your floor plan on the go based upon your APR flow.
7- After the floor plan the cell will looks like:

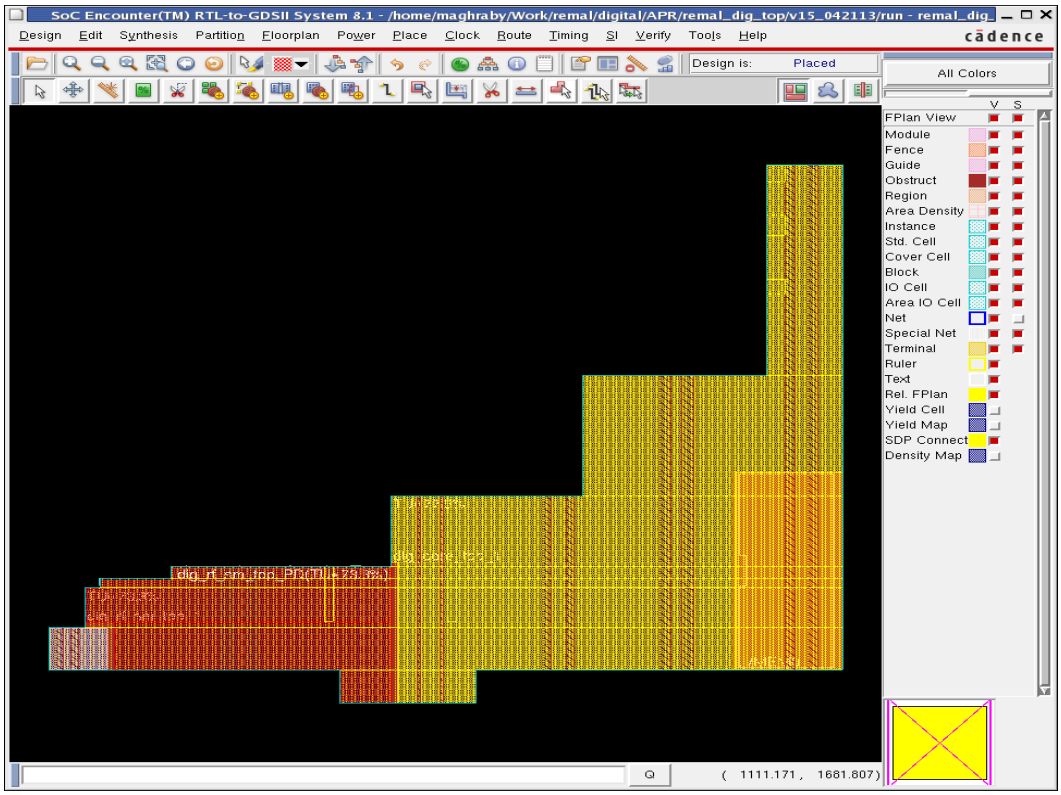

Figure 4 Multi power domains floor plan

8- For multi power domains make sure to leave some white space between different power domains to be able to add different DNW under the power domains if needed (15um should be enough)
9- In case you don't have multiple power you should define global connection for VDD/VSS pins.
    - *globalNetConnect vdd_topModule -type pgpin -pin VDD -inst * -module {}*
    - *globalNetConnect vdd_topModule -type tiehi -pin VDD -inst * -module {}*
    - *globalNetConnect vdd_topModule -type tielo -pin VDD -inst * -module {}*
    - *globalNetConnect vss_topModule -type pgpin -pin VSS -inst * -module {}*
    - *globalNetConnect vss_topModule -type tiehi -pin VSS -inst * -module {}*
    - *globalNetConnect vss_topModule -type tielo -pin VSS -inst * -module {}*

10- Power rails should also be added in the floor planning phase, this step is better to be done through the GUI to be visually traced
11- From the Power menu, select Power planning/Add stripes
12- For multiple power domains make sure that the supply net you are adding belongs to the power domain it is placed over
13- In case of IP usage, make sure the power rails you added is compatible with the IP cell (Connection between your rails and the cell VDD/VSS can be done smoothly) also it is nice if you can design it's pins direction to be the same Manhattan as your design.
14- It's not recommended to add vertical metal 2 as power rails, in case you use metal 2 you will not be able to place any STD cells under those rails
15- Note that routing is done as odd horizontal by default. So it's good from day one if you know you have to insert power rails odd vertical that you rotate your APR 90 degrees in the floor plan, another not recommended solution is to modify the lef file setting odd metals vertical
16- It's not an option to add metal strips in direction perpendicular to the STD cell metal 1 direction
17- Before adding the strips you should revise the post synthesis power report to make sure you will insert sufficient rails
18- It's not enough to just insert rails that can carry the current, but you should think of the current flow as well, for instance if the APR is very short and wide and you inserted one thick metal rail at the middle, the current will flow left and right from this rail only through the STD cells metal 1 to supply the whole STD cells at the same row
19- After finishing the power stripes addition, resave your def file this is important so as not to redo this step in the future and to keep the supply rails position fixed along the runs (Maybe it was already used in higher level integration)
20- In case you have a STD cells row just at the top of the floor plan, Encounter will not be able to connect it's supply properly, so it's recommended to leave some space above the upper STD cells row to be able to route its supply

• Placement

1- After designing the floor plan you can now place the design using the below command.
   - *placeDesign –prePlaceOpt*

2- Please make sure that the input to any of the STD cells is neither 1-bit (high) nor 0-bit (low) if this case exists you should use the following command to insert tiehi/tielo during placement, you can change the -maxDistance and -maxFanOut to control the number and the separation distance between ties.

   - *setTieHiLoMode -cell { TIEHI TIELO } -maxDistance 50 -maxFanOut 50 -honorDontTouch false -createHierPort false*

3- But make sure that the STD cells does not need to insert tie cells, in case the STD cells does not include the ties to VDD/VSS you should insert the tap cells before placing your design
4- To insret the well taps go for Place/Physical Cells/Add well tap, select the well tape cell to be used, enter the maximum gap between cells, offset from the row start, number of sites to skip, and row number to start adding the cells. The last three options may be used to place well taps interleaved between rows (You should run this twice in this case) to decrease the number of tap cells without affecting the maximum distance between devices and taps
5- After inserting the required fill well taps, copy the command from the encounter.cmd file to your procedure file just before the *placeDesign* command to be done automatically the next runs
6- The below command is used to check the placement done
   - *checkPlace topModule.checkPlace*

7- Small report is generated at Encounter terminal where the block utilization is reported (Per power domain), for small blocks 85% utilization is quiet enough, but for large blocks you'd start by 75% utilization, while for non-regular shapes (Of high aspect ratio) utilization may need to go done to 30% in some cases to be able to rout and meet timing
8- The report also include placement violations if any
9- After placement you can go for the physical view and check that placement is done as desired
10- Now save a post placement version in the working directory just in case you need to revert back
   - *saveDesign topModule.placed.enc*

- # Clock tree synthesis

1- Before doing the clock tree use the below command to make sure that all your clocks will be in propagated mode (Not to be treated as idea clocks as in synthesis):
   - *set_propagated_clock [all_clocks]*

2- First step is to generate the clock tree synthesis spec file, this can be done through the GUI from Clock/Design Clock then Generate Spec, Select the buffers and inverters you need to be used in the clock tree (Normally they start by CLK or CK)
3- Alternatively you can use the below command to generate the clock tree synthesis spec file:
   - *addCTSCellList {LIST_OF_BUFFER_INVERTER_CELLS}*
   - *clockDesign -genSpecOnly Clock.ctstch*

4- Either through the GUI or by command you will find the required spec file under the run directory named "Clock.ctstch", now copy it to the following path "/Project/digital/APR/topModule/vx_date/data/cts"
5- Open the spec file you've just generated and make sure it includes all the clocks, also you may need to add clock tree for the reset signals in your design (Ask the designers for them).
6- For the reset tree you will set the NoGating option to rising or falling, normally the clock tree defines its leafs at the CLK inputs of the DFFs, however since the reset signal will not face any CLK pin it should terminate the tree and define pins at any gate it will face
7- Keep in mind you need to define new clock tree for the reset signal at every net this reset is gated through the design
8- After making sure your CTS spec file is good you can run the below command to execute the tree
   - *clockDesign -specFile ../data/cts/Clock.ctstch -outDir*
   - *clock_report -fixedInstBeforeCTS*

9- Make sure the log file is free of any errors and thus your tree is now done
10- You can trace your tree in the GUI at Clock/Clock Tree Browser
11- After Encounter finishes the CTS it try a quick trial route and mark the routing congested areas in red diagonals, this congestion diagram may sometimes lead you to a floor plan modification
12- In case of very tight area, sometimes you got placement violations after the CTS, so it's worth to check for placement now:
   - *checkPlace topModule.checkPlace*

13- Now save a post CTS version in the working directory just in case you need to revert back
   - *saveDesign topModule.cts.enc*

- # Timing analysis and optimization[6]

1- This is the first time we check for timing violations along the flow, I don't recommend to check before the CTS as the clock nets are too much loaded with gates and thus facing large delays that may lead to a fake setup violations
2- Before analyzing the timing, it's useful to add timing deration to give an additional margin in the timing analysis, the following command will derate the results by 50pSec for both setup and hold checks
    - *set_timing_derate -min -early 1.0 -late 1.05 -clock*
    - *set_timing_derate -max -early 0.95 -late 1.0 -clock*

3- You can always trace the derated and un-derated timing slack in the timing reports thus you can know if you violation is a real one or not. However of course it's useful to optimize to reach no violations after deration
4- Start by a timing analysis (Not optimization) using the below command for setup and hold:
    - *timeDesign -postCTS -pathReports -drvReports -slackReports -numPaths 50 -prefix topModule_postCTS -outDir timingReports*
    - *timeDesign -postCTS -hold -pathReports -slackReports -numPaths 50 -prefix topModule postCTS -outDir timingReports*

5- This will generate the setup and hold timing reports that include the worst slack 50 paths in your design, results are at the run folder in the "timingReports" directory
6- The setup report is named as "topModule_postCTS_all.tarpt" and hold report is named "topModule_postCTS_all_hold.tarpt"
7- On terminal you will get also a summary for the timing report that tells you the worst negative slack, total negative slack and number of violations
8- We cannot do a tape out if we have a single negative slack for setup or hold, however for maximum caps, maximum transitions or maximum fan out violations it can happen that we tape out with them
9- Concept wise maximum caps and fan out violations are not important if they are not affecting the maximum transition violation, also maximum transition violations are not critical if they are not leading to setup nor hold violation, however it's always admirable to get them all violations free
10- From experience maximum transition violation of -2nSec is the worst acceptable violation if you are not waiving it.
11- Anyways any violation in the max cap, trans, fanout, should be reported to the designer and get and an waiver approval from him
12- In case you have setup/hold violations you should not go for the optimization step unless you debug the existing violations first, I don't recommend to do optimization for more than -1nSec worst negative slack
13- You can always debug the violations before optimization with the designer, below are some common causes for the violations
    a. Trace the delay corresponding to each gate output, nominally it should be less than 0.5nSec, in case it's larger most likely this net is connected to many gates and need to be buffered either by a clock tree or by a buffer addition

[6] For more information please check the (APR Timing) document.

b. Normally each clock domain should not be interacting in the same violation, if you find one then most likely you need to add a set false path between those two clocks after consulting the designer
c. If the violation is due to an interaction between the normal and scan modes, you can duplicate the sdc setting the scan enable to 1 for scan mode and 0 for normal mode
d. In general non regular complex floor plan may lead to timing violations as the blocks are placed in an odd way
e. Most likely using both negative and positive edge triggered clocks leads to timing violations, this may need a design fix
f. In some processes the worst case corner is measured in toughly deviated conditions (Too low supply or too high temperature) and thus the delays are normally large (In the order of 5nSec) in such cases you may need to discard this tough corner and go for the next slow corner.
g. It's normal if you see some negative delays in the timing reports or the sdf files, this is due to latency in the rising or falling edges at some flops
h. In case you got timing violations between a static control signal and other signal you can set false path from this static control signal
i. For the scan enable signal if it is causing timing violations with some signals in the normal mode you can create two sdc files and set the scan enable signal to 1 in the scan mode and 0 in the normal mode and use both files to check timing

14- After making sure your violations are controllable you can got for optimization, optimize the max cap, trans and fan out by:
    - *optDesign -postCTS –drv*

15- Optimize for setup violations by:
    - *optDesign –postCTS*

16- Optimize for hold violations by:
    - *optDesign -postCTS –hold*

17- After optimization you should not get any violation, if you still have once, make sure it will not be optimized further more after routing, thus you should get a clean timing now
18- Sometimes the optimized is not able to add buffer in certain path due to a wrong "Set don't touch" constraint at the sdc
19- After optimization you can still solve 1 or 2 remaining violations manually by an ECO[7] fix (Will be explained)
20- Now save a post CTS OPT version in the working directory just in case you need to revert back
    - *saveDesign topModule.cts.opt.enc*

[7] You can find the way to make ECO at the end of the document

• Routing

1- Routing you can see in Encounter now is just a trial rout that you can normally see shorts and DRCs at, don't worry all that will be fixed
2- It's time for routing now, start with the special route for supply and ground rails connection:
   - *sroute -connect { blockPin padPin padRing corePin floatingStripe } -layerChangeRange { 1 5 } -blockPinTarget { nearestRingStripe nearestTarget } -padPinPortConnect { allPort oneGeom } -checkAlignedSecondaryPin 1 -blockPin useLef -allowJogging 1 -crossoverViaBottomLayer 1 -allowLayerChange 1 -targetViaTopLayer 6 -crossoverViaTopLayer 6 -targetViaBottomLayer 1 -nets { vss_topModule vdd_topModule }*

3- Now go for nano route:
   - *getNanoRouteMode -quiet*
   - *getNanoRouteMode -user -drouteEndIteration*
   - *getNanoRouteMode -user -drouteStartIteration*
   - *getNanoRouteMode -user -routeBottomRoutingLayer*
   - *getNanoRouteMode -user -routeTopRoutingLayer*
   - *getNanoRouteMode -quiet -envSuperthreading*
   - *getNanoRouteMode -quiet -drouteFixAntenna*
   - *getNanoRouteMode -quiet -routeInsertAntennaDiode*
   - *setNanoRouteMode -quiet -routeInsertAntennaDiode true*
   - *getNanoRouteMode -quiet -routeAntennaCellName*
   - *setNanoRouteMode -quiet -routeAntennaCellName ANTENNA*
   - *getNanoRouteMode -quiet -timingEngine*
   - *getNanoRouteMode -quiet -routeWithTimingDriven*
   - *getNanoRouteMode -quiet -routeWithEco*
   - *getNanoRouteMode -quiet -routeWithLithoDriven*
   - *getNanoRouteMode -quiet -droutePostRouteLithoRepair*
   - *getNanoRouteMode -quiet -routeWithSiDriven*
   - *getNanoRouteMode -quiet -routeTdrEffort*
   - *getNanoRouteMode -quiet -routeWithSiPostRouteFix*
   - *getNanoRouteMode -quiet -drouteAutoStop*
   - *getNanoRouteMode -quiet -routeSelectedNetOnly*
   - *getNanoRouteMode -quiet -drouteStartIteration*
   - *getNanoRouteMode -quiet -envNumberProcessor*
   - *getNanoRouteMode -quiet -envSuperthreading*
   - *getNanoRouteMode -quiet -routeTopRoutingLayer*
   - *setNanoRouteMode -quiet -routeTopRoutingLayer default*
   - *getNanoRouteMode -quiet -routeBottomRoutingLayer*
   - *setNanoRouteMode -quiet -routeBottomRoutingLayer default*
   - *getNanoRouteMode -quiet -drouteEndIteration*
   - *setNanoRouteMode -quiet -drouteEndIteration default*
   - *getNanoRouteMode -quiet -routeEcoOnlyInLayers*
   - *getNanoRouteMode -quiet -routeWithTimingDriven*
   - *setNanoRouteMode -quiet -routeWithTimingDriven false*
   - *getNanoRouteMode -quiet -routeWithSiDriven*
   - *setNanoRouteMode -quiet -routeWithSiDriven false*
   - *routeDesign –globalDetail*

4- After few minutes a small report is printed on terminal before the routing iterations the "Total Overcon" is a ratio that indicates the routing congestion, below 5% routing should be done easily, between 5% and 10% it is doable with risk, above 15% most likely routing will not converge
5- In case you've got many violations after routing try to conclude something from the place and the metal of violation, this will guide you how you are going to modify the floor plan to be able to route
6- Sometimes the reason of getting routing violations is that you did a pre CTS timing optimization starting a high negative slack that lead the tool to add a lot of buffers and thus complicate the routing
7- It happens that hold and setup violations can be solved after optimization smoothly but some high negative slack in the maximum transition lead to a lot of buffering after optimization, thus the tool was not able to route, the solution was to optimize for setup and hold only waiving the max transition violations
8- Always think of the available metals for routing in different areas in your floor plan, you should leave at least one vertical and other horizontal rail for routing to be feasible
9- For small number of routing violations, you may help the tool to fix them, for example:
    a. You may move STD cell under a routing violation and thus the router will find place to reroute in the white area under it
    b. Blockage layer may be adjusted in some parts to solve routing violations
    c. Some antenna violation can be solve manually by moving some cells to decrease the long signal path

10- In case you were timing clean post CTS, most likely you will be good after routing as well. If not you will need to optimize timing once more now
11- Optimize the max cap, trans and fan out by:
    - *optDesign -postRoute–drv*

12- Optimize for setup violations by:
    - *optDesign –postRoute*

13- Optimize for hold violations by:
    - *optDesign -postRoute–hold*

14- In case you are already good you don't need to do optimization now
15- Don't worry about routing, the timing optimizer should reroute the design after fixing the optimization

• Physical verification

1- After that you need to do some physical verifications
2- For geometry verification use:
   - *verifyGeometry*
3- For antenna verification use:
   - *verifyProcessAntenna -reportfile topModule.antenna.rpt -leffile topModule.antenna.lef -error 1000*

4- For connectivity verification use:
   - *verifyConnectivity -type all -noAntenna -error 1000 -warning 50*

5- You should clean geometry and antenna violations, if you find some you can do routing once more and recheck
6- For connectivity violations sometimes it reports fake open violations, I don't know why! ☺
7- After reaching a clean version physically and timing wise, you'd save temporarily version of this good enc:
   - *saveDesign topModule.enc*

8- Now it's time to place the filler cells and do metal filling using the below commands:
   - *addFiller -cell FILL_CELLS_NAMES -prefix FILLER -markFixed*
   - *addMetalFill -layer { 1 2 3 4} -squareShape*

9- Of course replace FILL_CELLS_NAMES by your filler cells names separated by spaces and make sure you include the list of the required metal layers number to be filled for density fix
10- In some technologies one of the filler cells you may use is the FILLCAP cell(s). It's always advisable to use it under the designer consultancy to introduce a decap between VDD/VSS. It's also feasible to design this cell if not found in your technology and import it as a lef file.
11- To delete metal filling use the command:
   - *deleteMetalFill -layer Metal number*

12- To delete all the metal fillings use the command
   - *deleteMetalFill.*

13- To delete the filler cells us the command
   - *deleteFiller*

• Output data preparation

1- You've now got a final version ready to extract output
2- Use the following command to same enc file for the final version in the output directory
   - *saveDesign ../output/enc/topModule.enc*

3- Use the below command to save the GDS in its output directory (Note that the GDS is extracted in tar format so if you extracted it then re-exported you will not have the GDS file updated and you will need to delete it and re-extract it once more):
   - *streamOut ../output/gds/topModule.gds.gz -libName DesignLib -structureName topModule -mapFile ../data/map/map.map*

4- Use the below command to export the logical netlist that doesn't include the filler cells nor the VDD/VSS data
   - *saveNetlist ../output/vg/topModule.log.vg*

5- Use the following command to export he physical netlist that include the filler cells nor the VDD/VSS data
   - *saveNetlist -phys -excludeLeafCell ../output/vg/topModule.phys.vg*

6- Now go for the output directory, open the create_enc.pl script, modify the your source and lib file names definition then open a terminal in the output directory and run the script after sourcing the project setup.cshrc
7- The script will generate a copy from the enc file in the output/enc directory for each corner and name it by the corner name, it will also generate the enc_commands text file to be sources at Encounter
8- Note that the create_enc default script is designed to have an OTP lib files, if you are not interested in this part you should remove it from the script
9- Source the enc_commands file and have a cup of tea until Encounter load each of the enc files and generate the timing reports and sdc at each corner:
   - *source ../output/enc_commands*

10- After that you will get the output/timingReports_auto and output/sdf_auto ready, please check
11- You can nedit output/timingReports_auto/*.summary to brows if you have any setup/hold timing violation at different corners, it's worthy to think for a Perl script to check the timing violation output reports and generate a summary via corners
12- In case you have got timing violation at certain corner, you can browse the output/timingReports_auto directory to trace the violation, best solution is to load the topModule.enc file saved previously at the run directory (This one does not include the filling), then do an ECO fix to solve the violation the restart the output data preparation flow
13- Gtar the timingReports_auto, vg, sdf_auto directories from the output folder into topModule.vx_date.apr.tar.gz and send it by mail to the designer
14- After tape out you should check in the final APR folder at which you did your taped out APR run, this will insure anyone can work on any fix in the same APR in the future without even asking you

• Importing database into Virtuoso

1- In your first APR run, it's advisable to take the flow till the end to make sure you are good with the floor plan, supply routing, technology DRC/LVS/ANT
2- In your iterations, it's not recommended you go the flow till the end unless you got a confirmation from the designers that your APR output is good to go. However it's advisable to import the design each time you made change in the floor plan in your iterations
3- To import the design, un-tar the gds file in the output/gds directory (make sure you will un-tar it from the latest version as sometimes you are using an old un-tarred version while the tarred version got an update), then open the file output/vg/topModule.phys.vg and remove the filler cells (It is always placed as a list of lines at the file end), but make sure to keep the FILL CAP cells if found. You may remove the filler cells using nedit regular expression matching or a simple Perl script
4- The importing flow in the new and old Virtuoso versions differ slightly, for version 5 and below:

➢ **Layout**

a. From ICFB window go to File/Import/Stream to import the gds
b. Enter the input file name, library name
c. From the options menu, check the "Replace [] with <>" box, and check the "Retain Reference Library", and enter the reference library order
d. The concept of the reference libraries is that any imported cell should see the corresponding layout view for this cell to be imported otherwise it will not do the import
e. In the reference library order you should list the libraries that include the sub cells to be imported, please enter them in the following order (Technology library, STD cells library, IPs library, the target library)
f. In case you need to place the cell in DNW you can refer to the DNW STD library instead
g. After importing open the layout to make sure it's OK

➢ **Schematic**

a. From ICFB window go to File/Import/Verilog
b. Brows to your input physical vg file, select the file and click add netlist
c. Enter the target library name and the reference libraries the same order as in the gds import
d. The concept of the reference libraries is that any imported cell should see the corresponding symbol view for this cell to be imported otherwise it will not do the import
e. Modify the vdd/vss nets name to VDDD, GNDD which are not in your design
f. After the import open the verilogIn.log file in cadence directory to see if you have errors, if it's OK you should see the top level schematic generated
g. In case you have ANTENNA cell which has no symbol in its views you should create the symbol view first before importing (Either in the STD cells directory or in the same target library as a temporary solution)
h. Remember that you should remove the FILL cells before importing the netlist as the FILL cells done not have symbol nor schematic views

i. In case you have an IP cell, best solution is to create an empty schematic and symbol view for this cell that only contains the correct pin list, after that include the cdl netlist for this cell in your top level netlist after modifying the pin order according to your new empty symbol view you've generated
j. You can ask the CAD guy to include the IP cdl netlist by default in any exported CDL netlist

5- For Virtuoso version 6 and above the same flow will apply, but for the stream in menu you should select the "Text Namespace" to be Verilog, turn on the automatic detect via option, and enter the reference libraries in the library list

- **ECO fix**

1- Delay may be added manually at certain path to pass timing by adding buffers (In this case you will need to add a low drivable buffers and will need to place it away from the required position to increase its delay)
2- Also you may add buffers to decrease the delay on certain highly loaded net (In this case you will add a high drivable buffer and place it near from the required position to decease the delay)
3- Commonly the ECO is done through GUI to be able to place the buffers in the required locations visually
4- Open the design browser (Tools/Design Browser)
5- Brows for the required not to add buffers on, select it and click on select net to select it in Encounter
6- Open the ECO menu at Timing/Interactive ECO/Add repeater tab
7- Click the get selected net to get the net you have selected by the design browser
8- Choose either you will add the buffer between this net and all connected outputs or for a single destination (In this case you will enter the destination port path)
9- Select the required buffer to be added form the list
10- And select the location from the space mode options to be able to select the required location this buffer will be added
11- Click apply and look for the added buffer, it may be placed inverted and you need to move it manually to fix this
12- Now rerun timing analysis to make sure the violation is gone, it not you may need to add more buffers (Note that the net name may now be changed and thus you need to re-get its new name from the design browser)
13- If you need to delete a repeater you can use the Delete Repeater tab in the same window
14- It's worth to check for placement now to make sure no violations exist:
    - *checkPlace topModule.checkPlace*

15- If everything is now fixed you can save an enc version:
    - *saveDesign topModule.cts.opt.enc*