

# Homework Assignment #6

---

Submission Due: 2024/06/02 23:59

## Objective

This homework will walk you through the APR flow with Cadence Innovus.

*Acknowledgement: This tutorial is adapted from the APR Flow of EE4292 IC Design Laboratory by Prof. Chao-Tsung Huang, NTHU.*

## Submission

Submit the three power reports and a written (PDF) report:

- `pre_layout_power_report.rpt`
- `post_sim_power_report.rpt`
- `pre_sim_power_report.rpt`
- `<YourStudentID>_hw6_report.pdf`
  - The PDF report should include the eleven checkpoints and their discussions (see the reference APR flow in the following) in your write-up.

Let's get started!

## Reference APR Flow

---

### Synthesis

1. Add your design (`lenet.v` & other hdl files) in Homework 4 to **hw6/hdl/**
2. Add your HDL file path to **hw6/sim/sim\_rtl.f**
3. Simulate and verify the RTL design to ensure that the design is correct.

```
$ cd hw6/sim  
$ make sim
```

4. Synthesize your design with **synthesis.tcl** and understand how the script is configured. Note: Clock period is 10ns in default, modify it if needed.

```
$ cd hw6/syn  
$ dc_shell -f synthesis.tcl
```

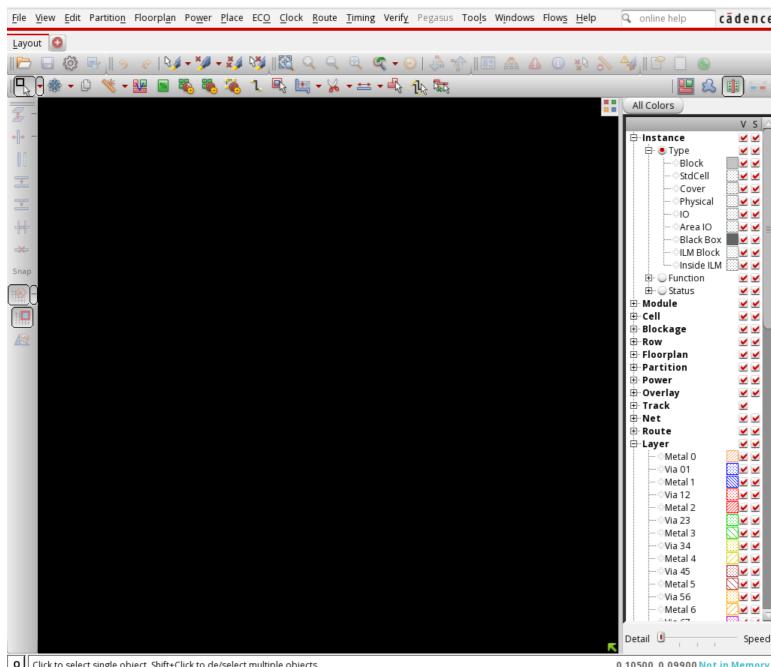
5. Do the gate-level simulation to make sure that the netlist is correct.

```
$ cd hw6/sim
$ make syn
```

Note: If you encounter problem in the APR flow, stay calm and don't panic. We have a reference netlist for you to proceed. You can replace **hw6/syn/netlist/** with **hw6/TA\_netlist/** and finish the assignment. However, using TA netlist will cause differences between the RTL code and the netlist. As a result, post-layout power analysis with the pre-sim waveform won't be possible, so points will be deducted for this part.

## Design Setup

1. Enter the **innovus** folder, launch the GUI interface of Innovus by:



```
$ cd hw6/apr/innovus
$ innovus
```

2. Import the Verilog design

**File->Import Design**

**Verilog**    **..../syn/netlist/lenet\_syn.v**

---

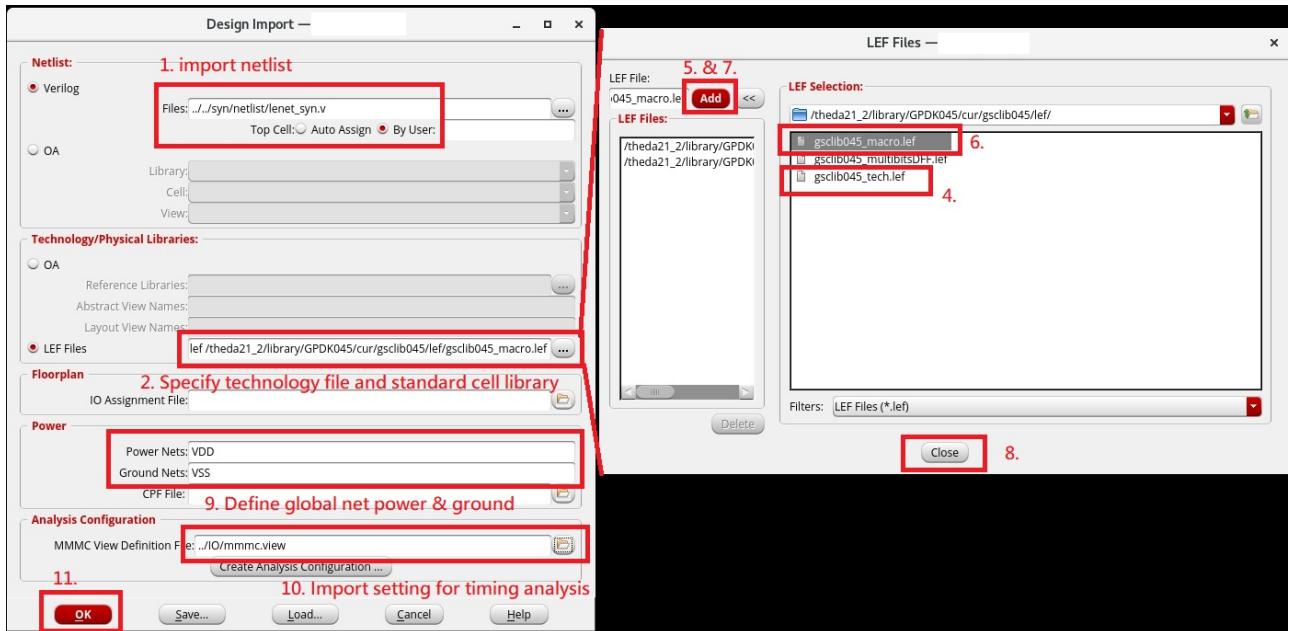
|                  |   |
|------------------|---|
| <b>LEF files</b> | <b>/theda21_2/library/GPDK045/cur/gsclib045/lef/gsclib045_tech.lef</b>  |
|                  | <b>/theda21_2/library/GPDK045/cur/gsclib045/lef/gsclib045_macro.lef</b> |

---

|              |                        |                         |
|--------------|------------------------|-------------------------|
| <b>Power</b> | <b>Power Nets: VDD</b> | <b>Ground Nets: VSS</b> |
|--------------|------------------------|-------------------------|

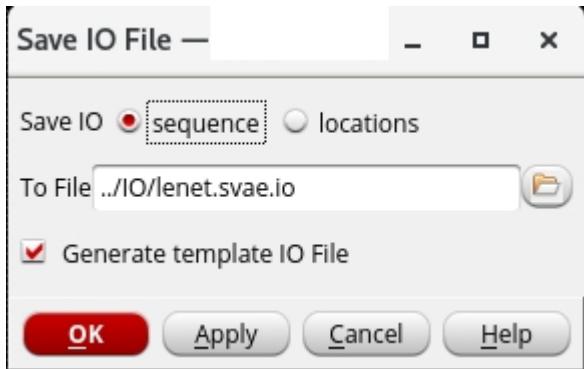
---

**MMMC**    **..../IO/mmmmc.view**



### 3. Create the IO constraint file

File->Save->I/O File



This step generates an IO template according to the IO declaration orders in your netlist.

### 4. Modify the IO constraint file

```

13  {iopin clk & rst_n should not place at corner      remove all "place_status"
14  {top
15  (pin name="clk" layer=2 width=0.0800 depth=0.2500 place_status
16  (pin name="rst_n" layer=2 width=0.0800 depth=0.2500 place_s
17  (pin name="compute_start" layer=2 width=0.0800 depth=0.2500
18  (pin name="compute_finish" layer=2 width=0.0800 depth=0.2500
19  (pin name="scale_CONV1[31]" layer=2 width=0.0800 depth=0.2500
20  (pin name="scale_CONV1[30]" layer=2 width=0.0800 depth=0.2500
21  (pin name="scale_CONV1[29]" layer=2 width=0.0800 depth=0.2500
22  (pin name="scale_CONV1[28]" layer=2 width=0.0800 depth=0.2500
23  (pin name="scale_CONV1[27]" layer=2 width=0.0800 depth=0.2500
24  (pin name="scale_CONV1[26]" layer=2 width=0.0800 depth=0.2500
25  (pin name="scale_CONV1[25]" layer=2 width=0.0800 depth=0.2500
26  (pin name="scale_CONV1[24]" layer=2 width=0.0800 depth=0.2500
27  (pin name="scale_CONV1[23]" layer=2 width=0.0800 depth=0.2500
28  (pin name="scale_CONV1[22]" layer=2 width=0.0800 depth=0.2500
29  (pin name="scale_CONV1[21]" layer=2 width=0.0800 depth=0.2500
30  (pin name="scale_CONV1[20]" layer=2 width=0.0800 depth=0.2500
31  (pin name="scale_CONV1[19]" layer=2 width=0.0800 depth=0.2500
32  (pin name="scale_CONV1[18]" layer=2 width=0.0800 depth=0.2500
33  (pin name="scale_CONV1[17]" layer=2 width=0.0800 depth=0.2500
34  (pin name="scale_CONV1[16]" layer=2 width=0.0800 depth=0.2500
35  (pin name="scale_CONV1[15]" layer=2 width=0.0800 depth=0.2500
36  (pin name="scale_CONV1[14]" layer=2 width=0.0800 depth=0.2500
37  (pin name="scale_CONV1[13]" layer=2 width=0.0800 depth=0.2500
38  (pin name="scale_CONV1[12]" layer=2 width=0.0800 depth=0.2500
39  (pin name="scale_CONV1[11]" layer=2 width=0.0800 depth=0.2500
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70  (pin name="scale_CONV2[10]" layer=2 width=0.0800 depth=0.2500 )
71  (pin name="scale_CONV2[9]" layer=2 width=0.0800 depth=0.2500 )
72  (pin name="scale_CONV2[8]" layer=2 width=0.0800 depth=0.2500 )
73  (pin name="scale_CONV2[7]" layer=2 width=0.0800 depth=0.2500 )
74  (pin name="scale_CONV2[6]" layer=2 width=0.0800 depth=0.2500 )
75  (pin name="scale_CONV2[5]" layer=2 width=0.0800 depth=0.2500 )
76  (pin name="scale_CONV2[4]" layer=2 width=0.0800 depth=0.2500 )
77  (pin name="scale_CONV2[3]" layer=2 width=0.0800 depth=0.2500 )
78  (pin name="scale_CONV2[2]" layer=2 width=0.0800 depth=0.2500 )
79  (pin name="scale_CONV2[1]" layer=2 width=0.0800 depth=0.2500 )
80  (pin name="scale_CONV2[0]" layer=2 width=0.0800 depth=0.2500 )
81  (pin name="scale_CONV3[31]" layer=2 width=0.0800 depth=0.2500 )
82  (pin name="scale_CONV3[30]" layer=2 width=0.0800 depth=0.2500 )
83  (pin name="scale_CONV3[29]" layer=2 width=0.0800 depth=0.2500 )
84  (pin name="scale_CONV3[28]" layer=2 width=0.0800 depth=0.2500 )
85  (pin name="clk" layer=2 width=0.0800 depth=0.2500 )
86  (pin name="rst_n" layer=2 width=0.0800 depth=0.2500 )
87  (pin name="scale_CONV3[27]" layer=2 width=0.0800 depth=0.2500 )
88  (pin name="scale_CONV3[26]" layer=2 width=0.0800 depth=0.2500 )
89  (pin name="scale_CONV3[25]" layer=2 width=0.0800 depth=0.2500 )
90  (pin name="scale_CONV3[24]" layer=2 width=0.0800 depth=0.2500 )
91  (pin name="scale_CONV3[23]" layer=2 width=0.0800 depth=0.2500 )
92  (pin name="scale_CONV3[22]" layer=2 width=0.0800 depth=0.2500 )
93  (pin name="scale_CONV3[21]" layer=2 width=0.0800 depth=0.2500 )
94  (pin name="scale_CONV3[20]" layer=2 width=0.0800 depth=0.2500 )
95  (pin name="scale_CONV3[19]" layer=2 width=0.0800 depth=0.2500 )
96  (pin name="scale_CONV3[18]" layer=2 width=0.0800 depth=0.2500 )

```

Clock and reset signals should locate in the middle of one side of the layout for a better timing balance.

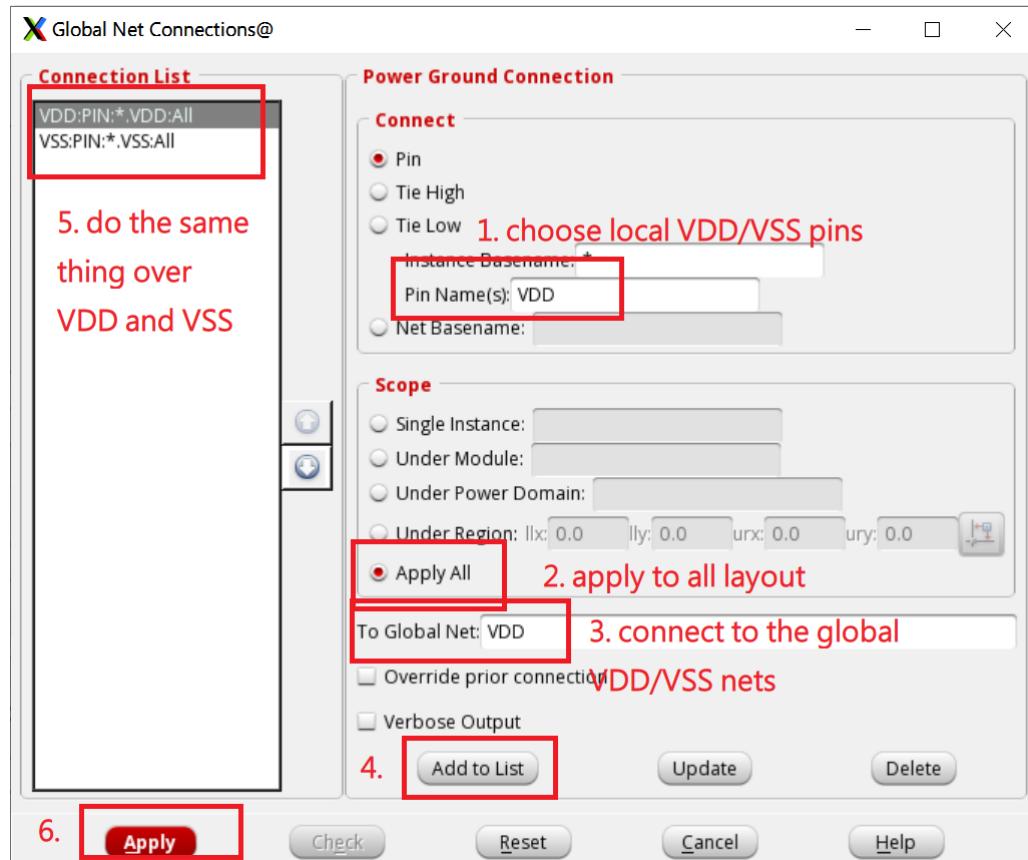
Also remove all "place\_status" configuration in IO file.

## 5. Set the process node

```
$innovus > setDesignMode -process 45
```

## 6. Power/Ground connection

Power-&gt;Connect Global Nets

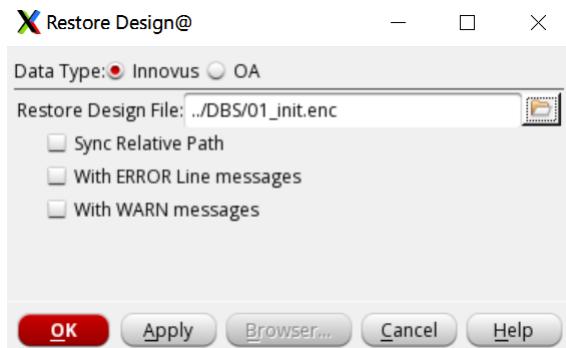


## 7. Save the design

File-&gt;Save Design



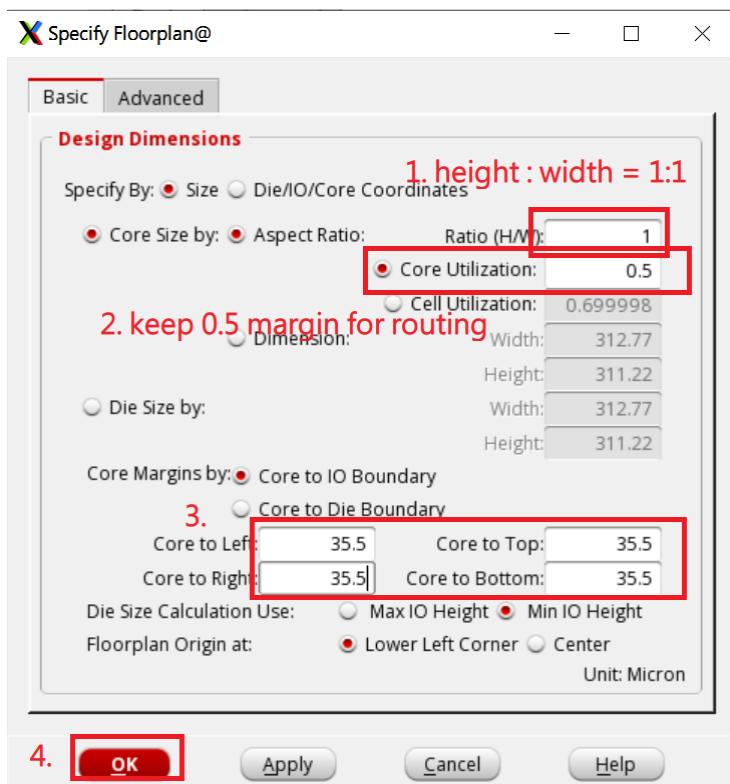
Note: You can restore your design by File->Restore Design



## Floorplan

### 1. Create the floorplan

Floorplan->Specify Floorplan



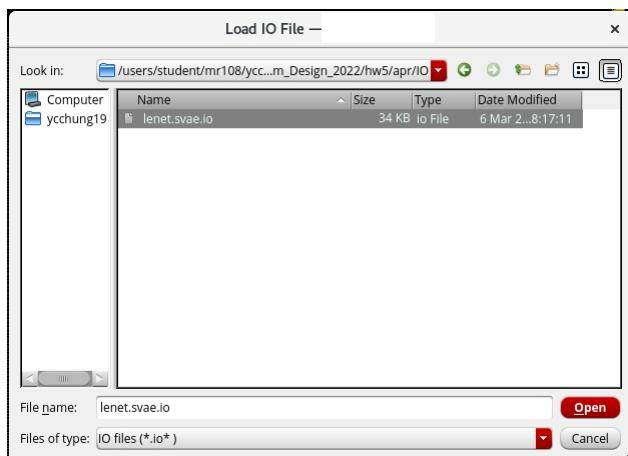
Note: "Core to IO boundary" indicates the space we preserve for the core ring. The space should be large enough for the offset and the width of the core rings (with proper internal space); it should be as small as possible for the area concern.

For example, we choose 35.5 for  $\$ \$ \text{ offset} + 32\text{ring} + 5 * \text{space} + \text{offset} = 2 + 64 + 51.5 + 2 = 35.5 \$ \$$

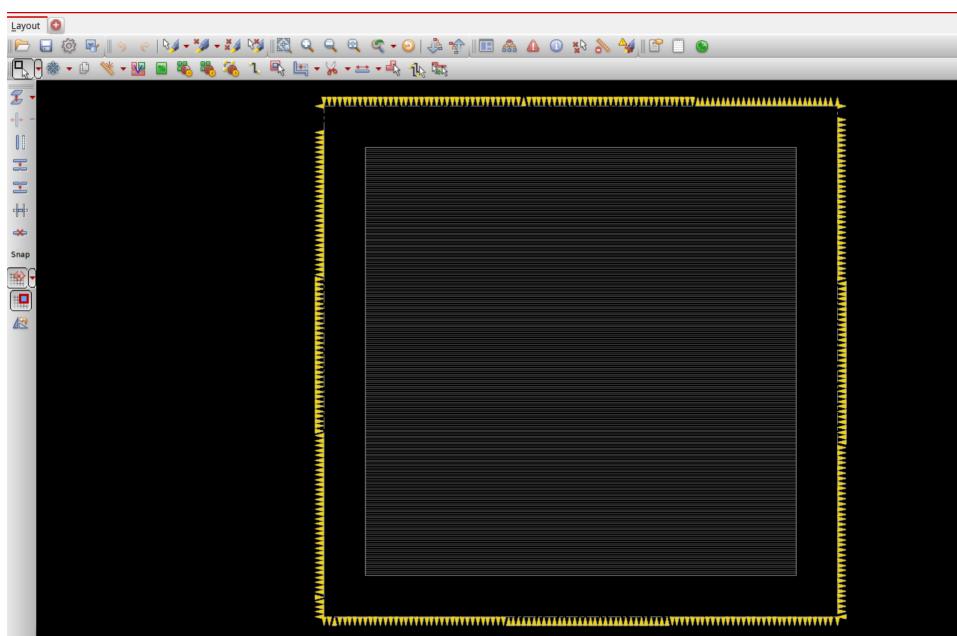
Note: The core utilization can be up to 70~80% practically (and even higher for a commercial chip). Here we set 50% simply for a faster APR in the lab. The core margins are reserved for the core power ring.

### 2. Read the IO constraint file

File->Load->I/O File

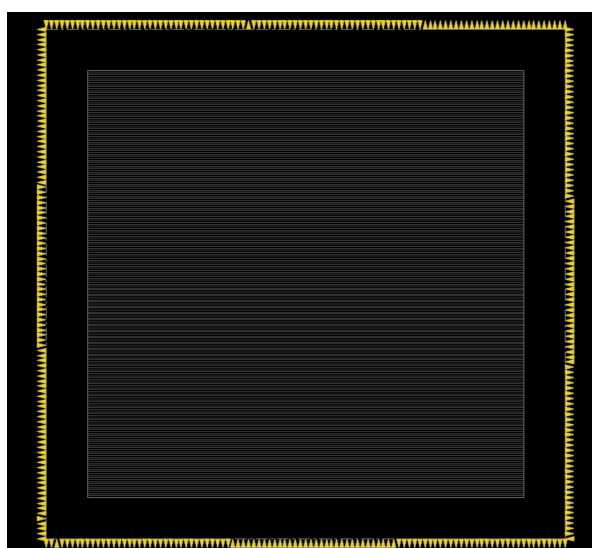


3. Press '**f**' or '**ctrl+r**' to redraw layout



*Note:* The tool automatically added IOs during the floorplan. But you still need to reload your IO file again to update the IO pin location.

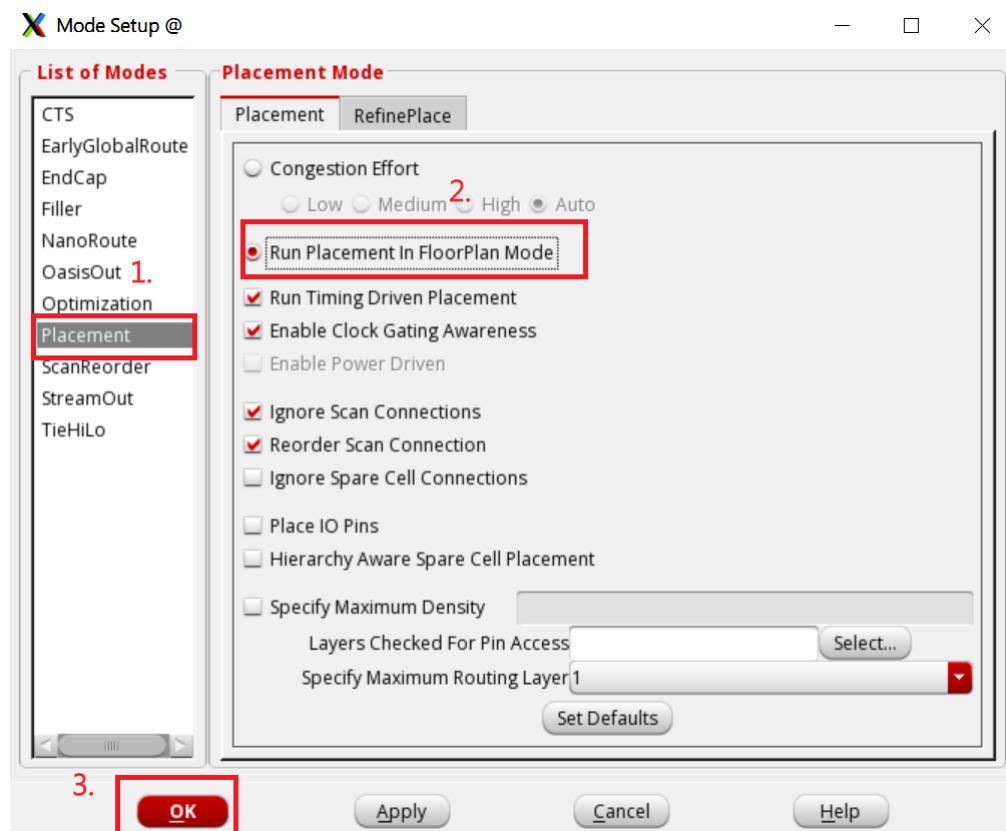
*Note:* If the IO pins are not balance-distributed, use **Floorplan->Specify Floorplan** and press **OK**



## Initial Placement

## 1. Set the placement to floorplan mode

Tool-> Set Mode->Mode Setup



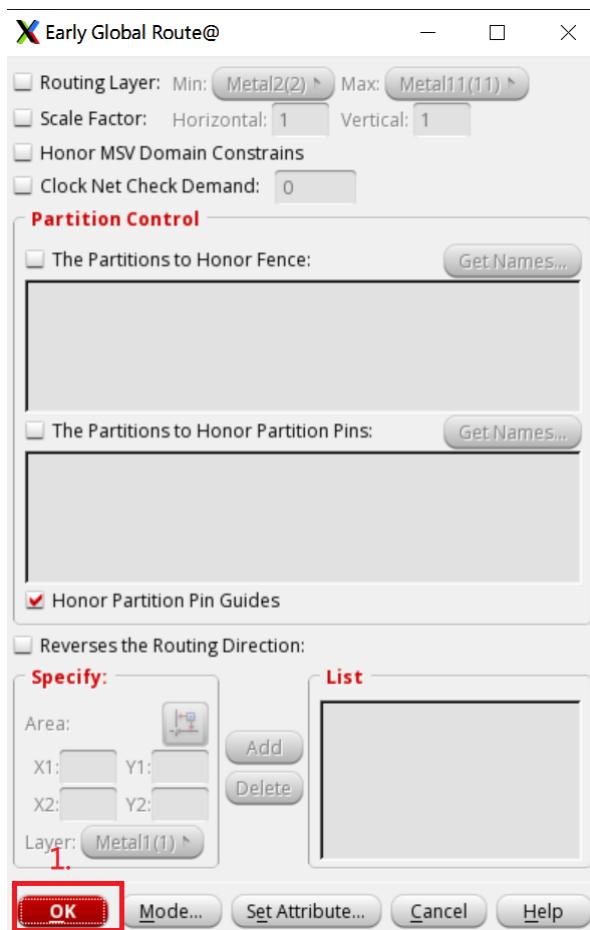
## 2. Execute the initial placement

```
$ innovus > place_design
$ innovus > refinePlace -checkRoute 1
```

## 3. Early global route

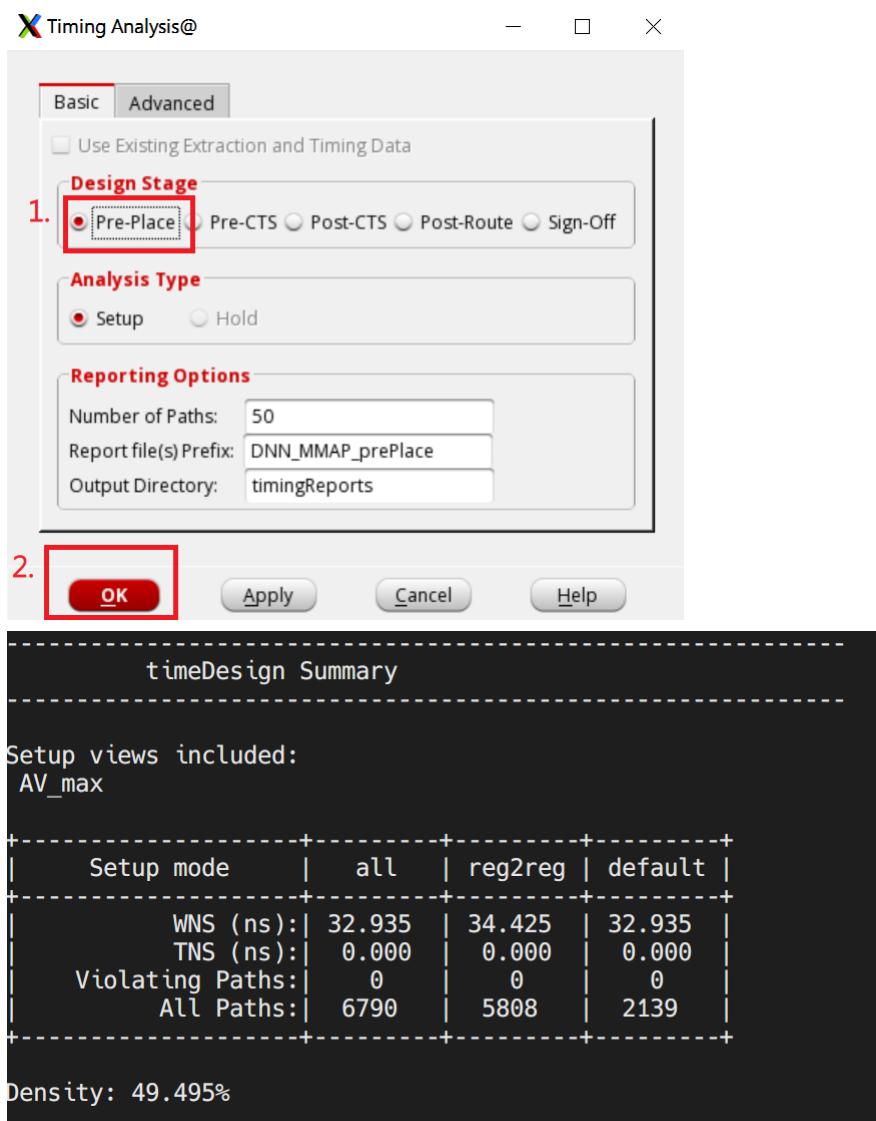
Route->Early Global Route

press OK



#### 4. Check the timing

[Timing->Report Timing](#)



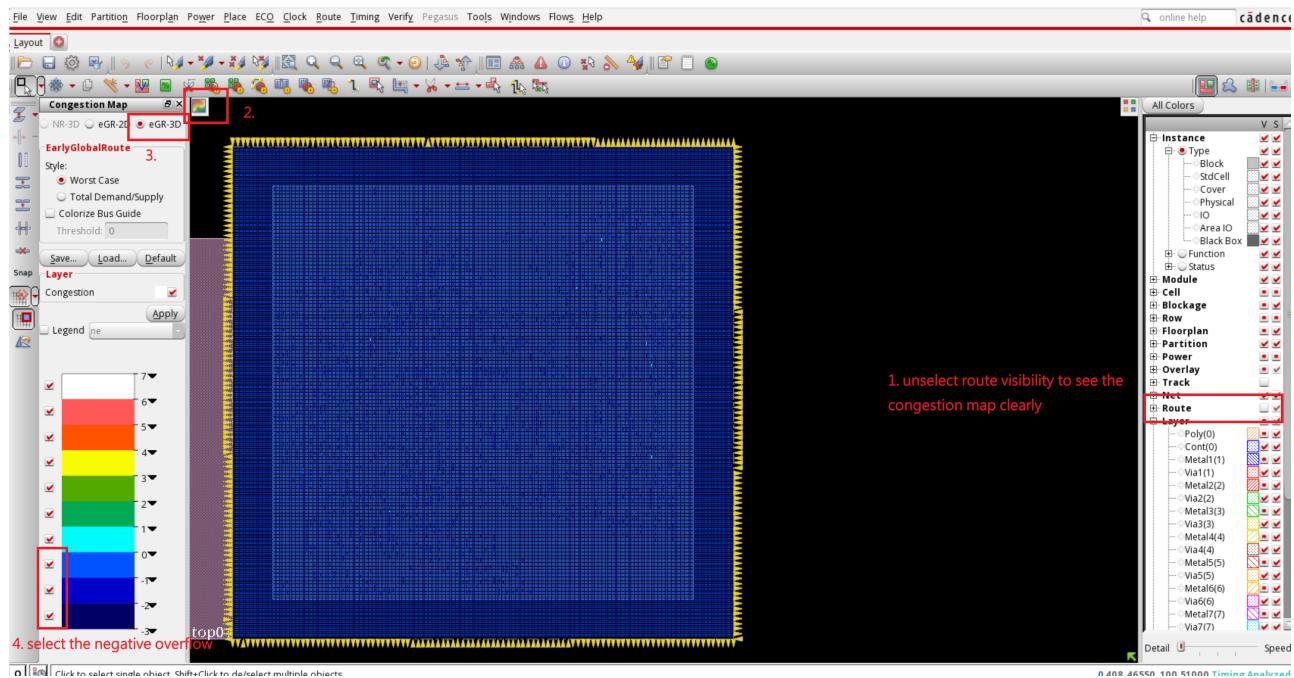
*Note:* Placement in the floorplan mode utilizes a low effort for congestion and applies the simplest routing for delay calculation. Therefore, there should not be any violations. If any violations occur, you can try the following steps:

- Reorder your IO
- Synthesize with a more proper design constraint
- Increase the clock period

## 5. Analyze the congestion

Route->NanoRoute->Analyze Congestion

**Checkpoint 1: Show your congestion map in the report.**



Note: The value of overflow = # nets crossing the gcell (global routing cell) - available routing tracks

Does the Congestion map show that our design is congested?

If it is, you can use the following command:

```
$ innovus > congRepair
```

Select **Route** visibility after viewing the congestion map.

6. Save your design

File->Save Design

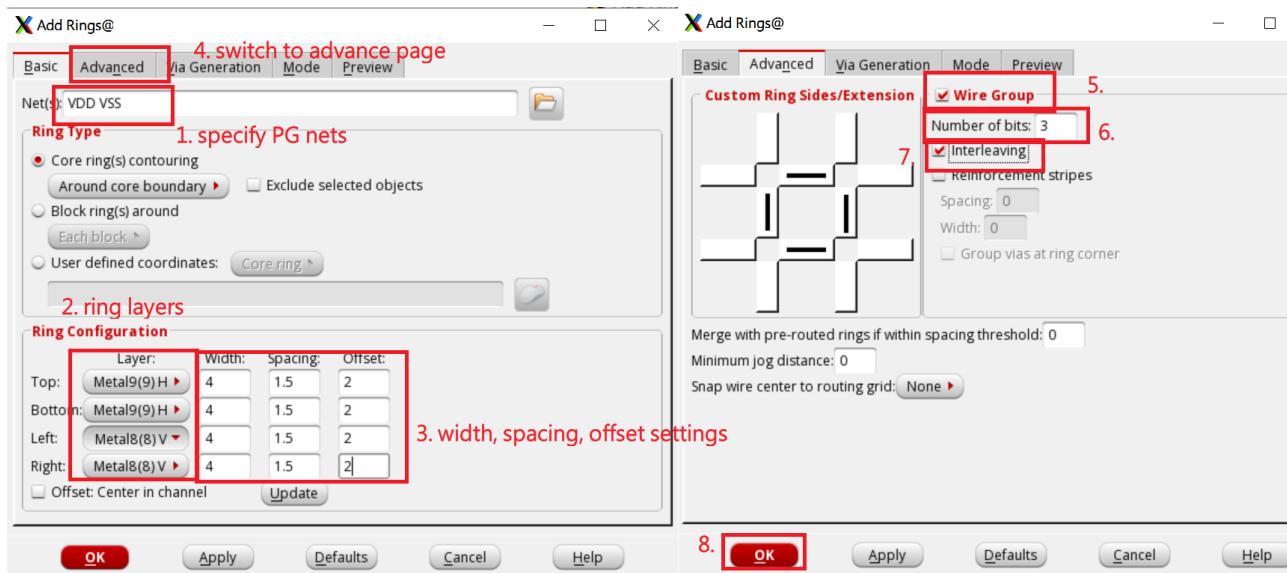


Note: Here, we save the design for the second time. Throughout the lab, the tutorial will guide you to repeatedly save the design, just if you need to resume the APR any time or redo the flow from the midpoint. You may remove the temporarily saved design later to release the disk space.

## Powerplan

1. Create the power ring

Power->Power Planning-> Add Ring



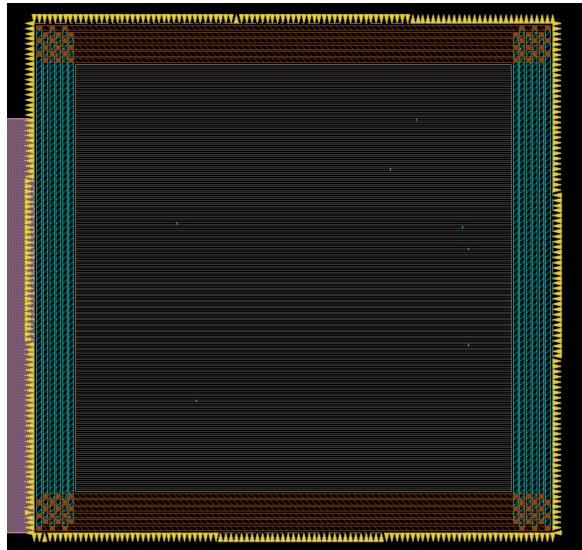
Note: The spacing is chosen based on \*\* "DRC: Metal spacing rules"\*\*.

Note: The ring width is chosen based on the power consumption of the whole chip.

Note: Due to \*\* "DRC: Metal spacing rules,"\*\* the ring width cannot exceed the maximum width. Thus, we use multiple pairs of Power rings to provide enough power supply for your design.

**Number of bits:** Specifies the number of nets for the specific power signal.

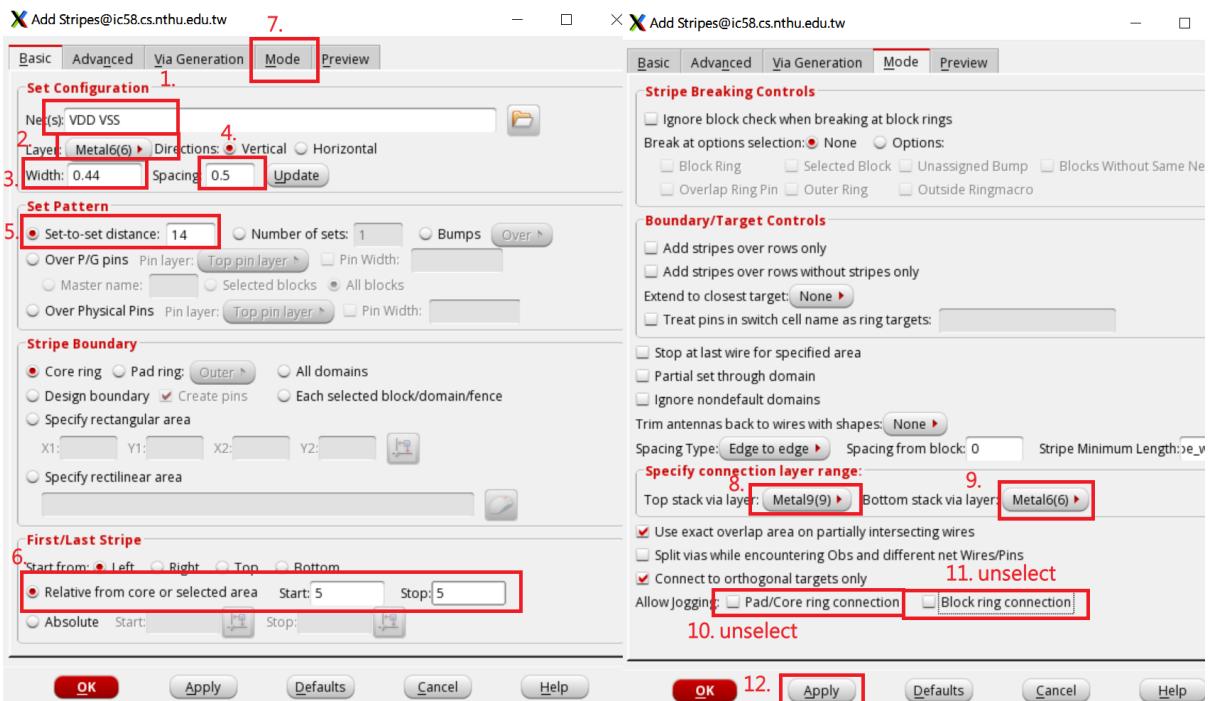
**Interleaving:** Alternate the VDD and VSS in wire groups.



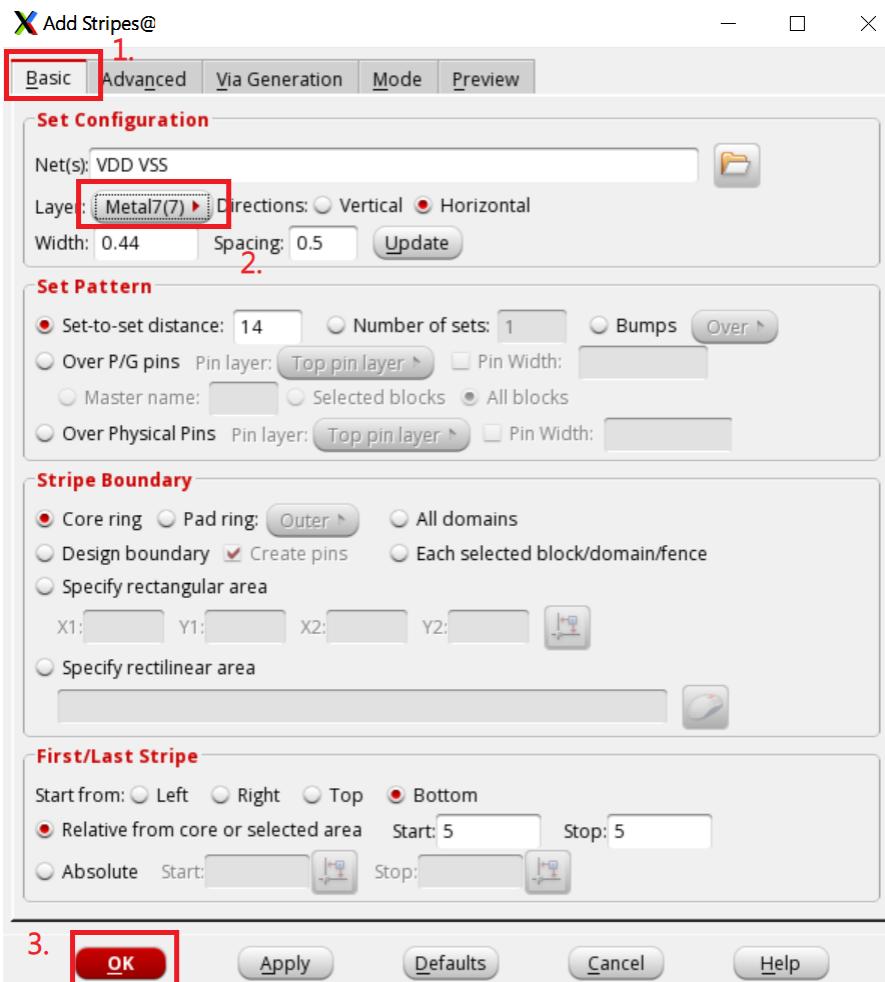
## 2. Create the power stripes

Power->Power Planning-> Add Stripes

### 1. Add vertical stripes

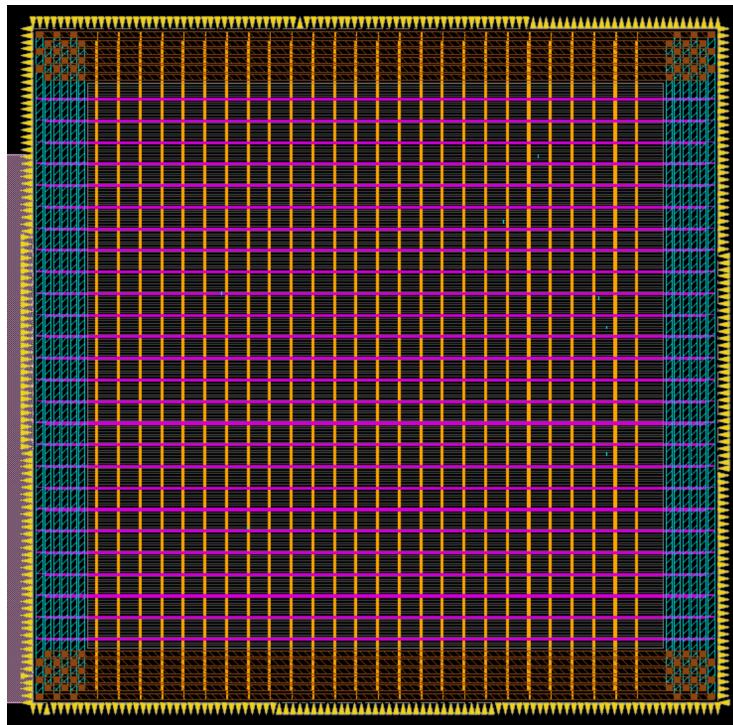


## 2. Add horizontal stripes



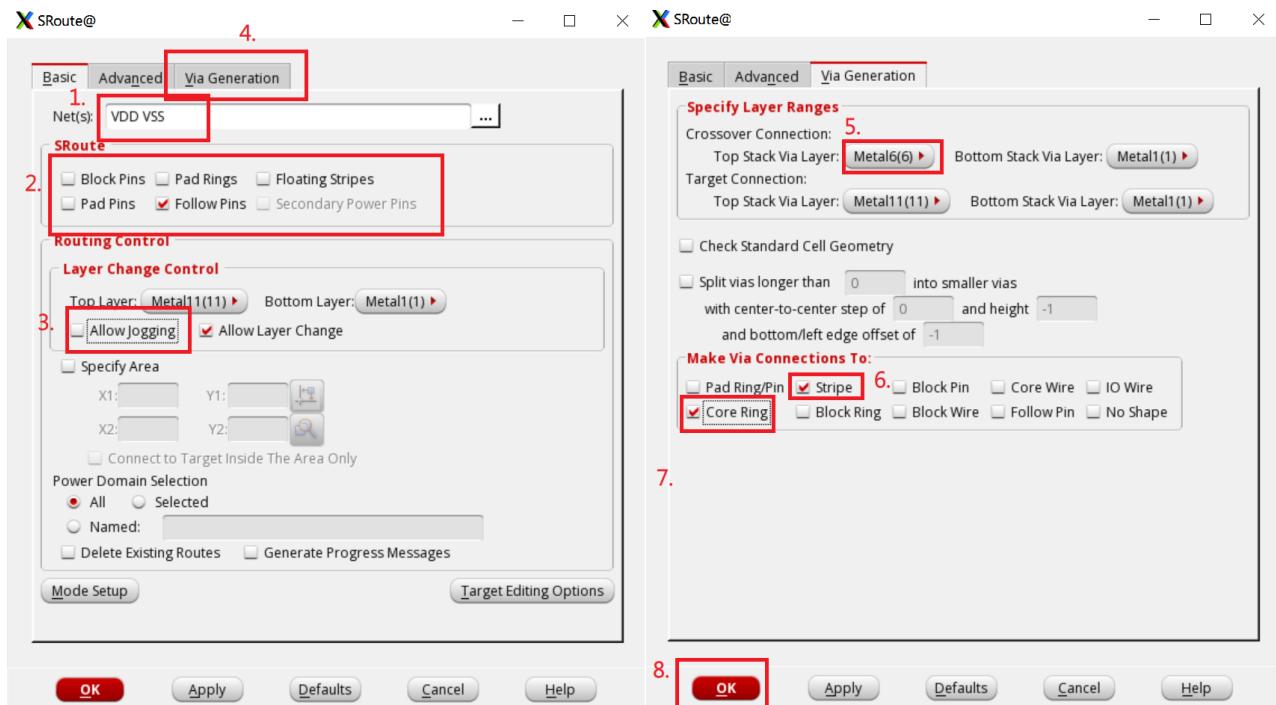
Note: Standard cells are surrounded by Power/Ground-rail (M1) segments on top and bottom. To reduce routing complexity, each metal layer's direction in the APR flow will follow the rail direction of the same metal layer inside the standard cells.

In our case, odd-numbered metal layers (e.g., M1, M3, ..., M9) route horizontally, and even-numbered metal layers (e.g., M2, M4, ..., M8) vertically. Though this is not a strict rule, violating the metal direction guideline will introduce routing difficulty (i.e., more routing resources will be required).



### 3. Connect the Follow Pin

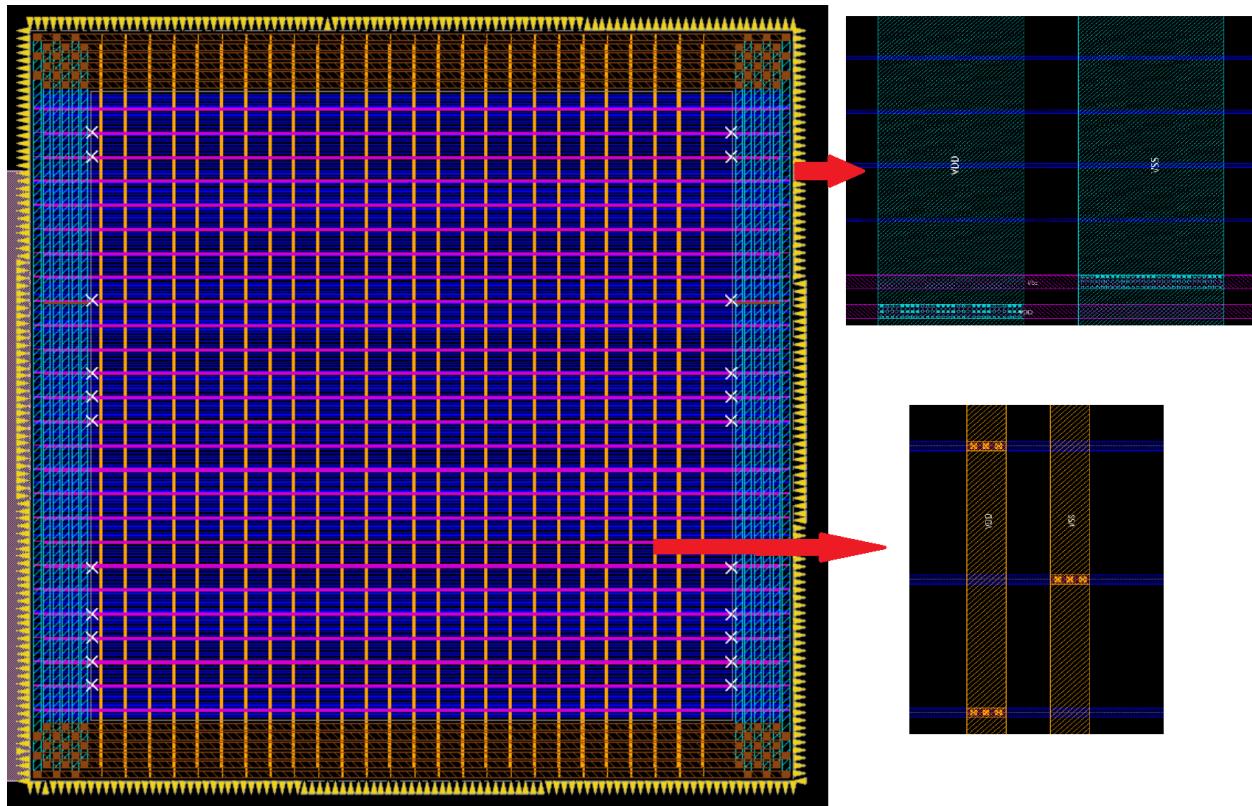
#### Route->Special Route



Some **Xs** show up because the stripe's via is blocking some rails; since our power mesh is dense enough for an acceptable IR drop, press "**shift+v**" to neglect it.

After the power rails have been created, please check if the rails connect to power stripes and core rings correctly.

Checkpoint 2: Show a part of your layout with power rails in the report.



#### 4. Save your design



## Placement

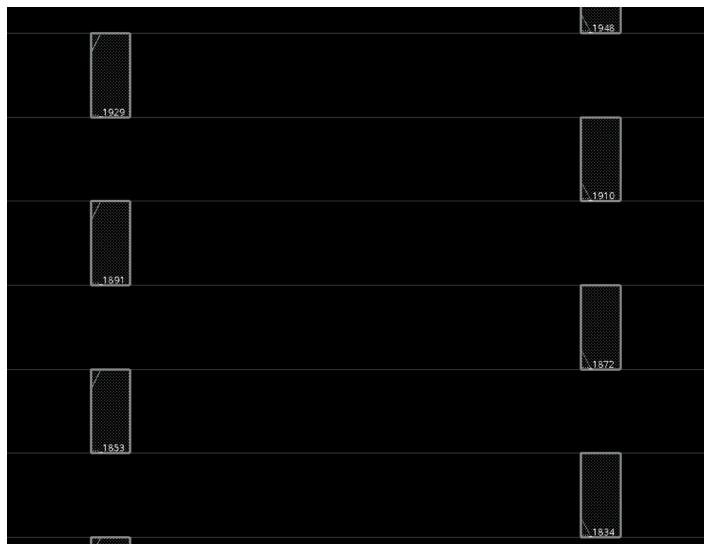
#### 1. Create the Path Groups

```
$ innovus > createBasicPathGroups -expanded
$ innovus > get_path_groups
```

#### 2. Add the Tap cells

```
$ innovus > addWellTap -cell FILL4 -cellInterval 20 -checkerBoard
```

Where are the Tap cells placed?



### 3. Run the placement

*Note:* This step might take about 10 minutes.

```
$ innovus > setPlaceMode -reset
$ innovus > place_opt_design
```

### 4. Check to make sure there are no violating paths

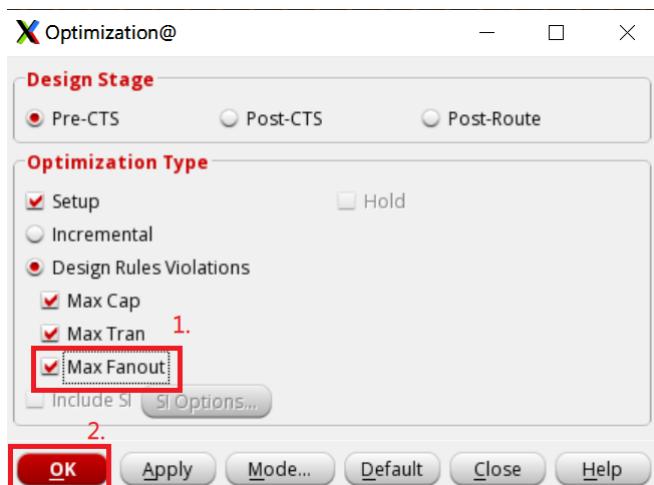
| optDesign Final Summary         |        |         |        |         |        |         |
|---------------------------------|--------|---------|--------|---------|--------|---------|
| Setup views included:<br>AV_max |        |         |        |         |        |         |
| Setup mode                      | all    | reg2reg | in2reg | reg2out | in2out | default |
| WNS (ns):                       | 32.390 | 33.645  | 33.138 | 32.390  | N/A    | 0.000   |
| TNS (ns):                       | 0.000  | 0.000   | 0.000  | 0.000   | N/A    | 0.000   |
| Violating Paths:                | 0      | 0       | 0      | 0       | N/A    | 0       |
| All Paths:                      | 6694   | 5712    | 2008   | 131     | N/A    | 0       |

| DRVs       | Real           |           | Total          |
|------------|----------------|-----------|----------------|
|            | Nr nets(terms) | Worst Vio | Nr nets(terms) |
| max_cap    | 0 (0)          | 0.000     | 1 (1)          |
| max_tran   | 0 (0)          | 0.000     | 1 (1)          |
| max_fanout | 219 (219)      | -55       | 220 (220)      |
| max_length | 0 (0)          | 0         | 0 (0)          |

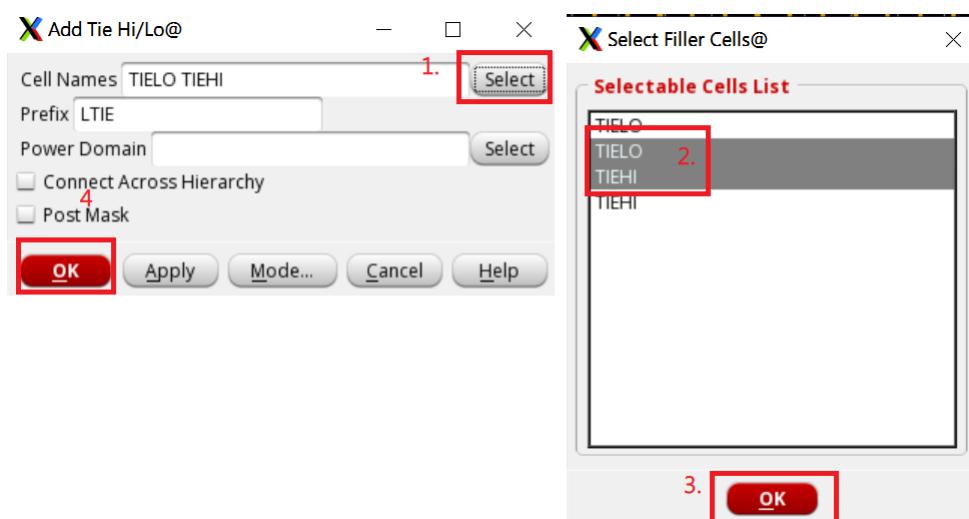
### 5. (Optional) Optimize Design (if the timing did not pass during the previous steps.)

[ECO->Optimize Design](#)



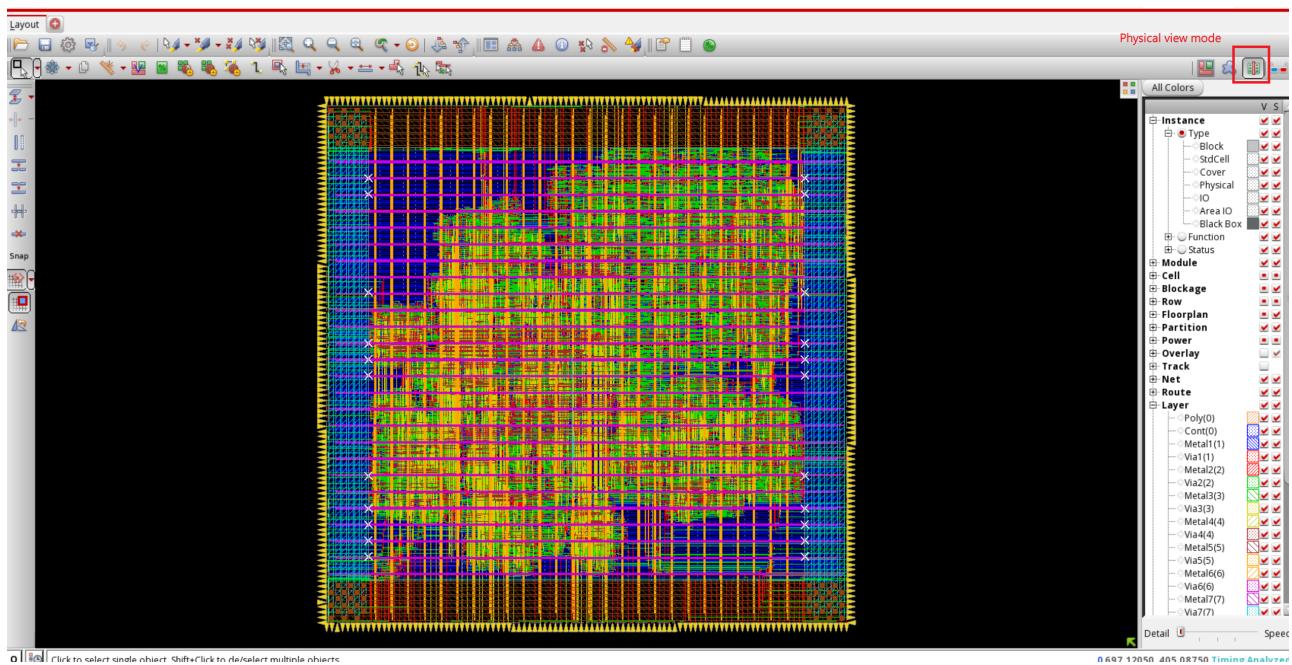
## 6. Add the Tie High/Low Cells

Place->Tie HI/LO cell->Add



Note : There are duplicated options in the GUI window (e.g., two TIELO options and two TIEHI ones). It is OK to select the ones we need arbitrarily.

## 7. Switch to physical view mode to see whether the standard cells have been placed on the layout.

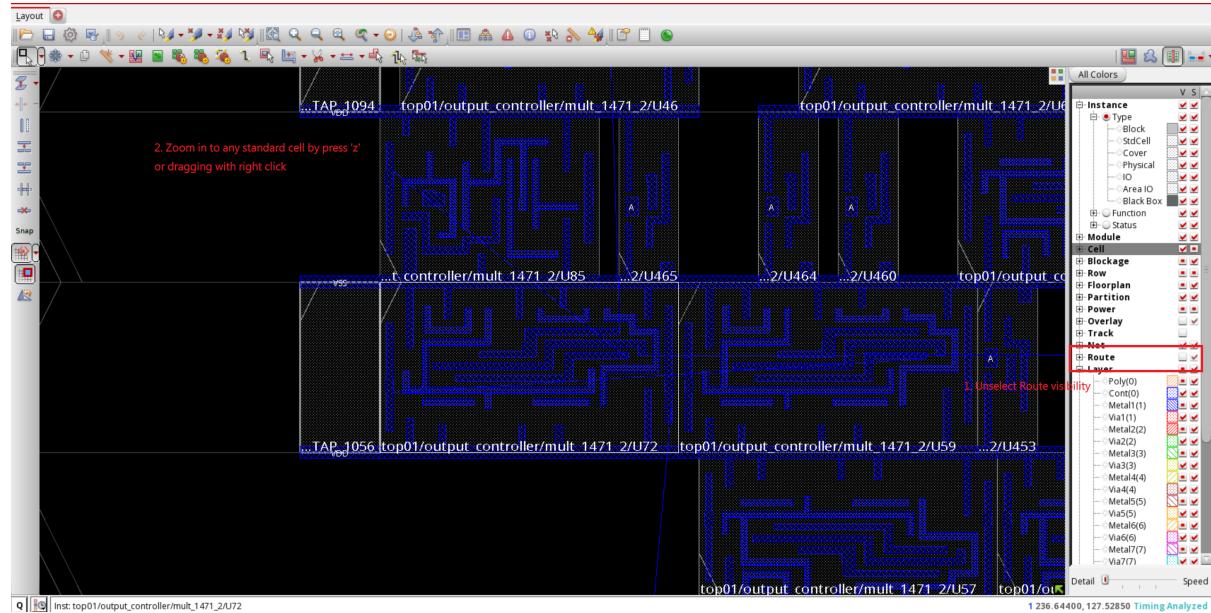


## 8. Observation

### 1. Show the standard cell FRAM view

Checkpoint 3: Show a part of your standard cells in the report.

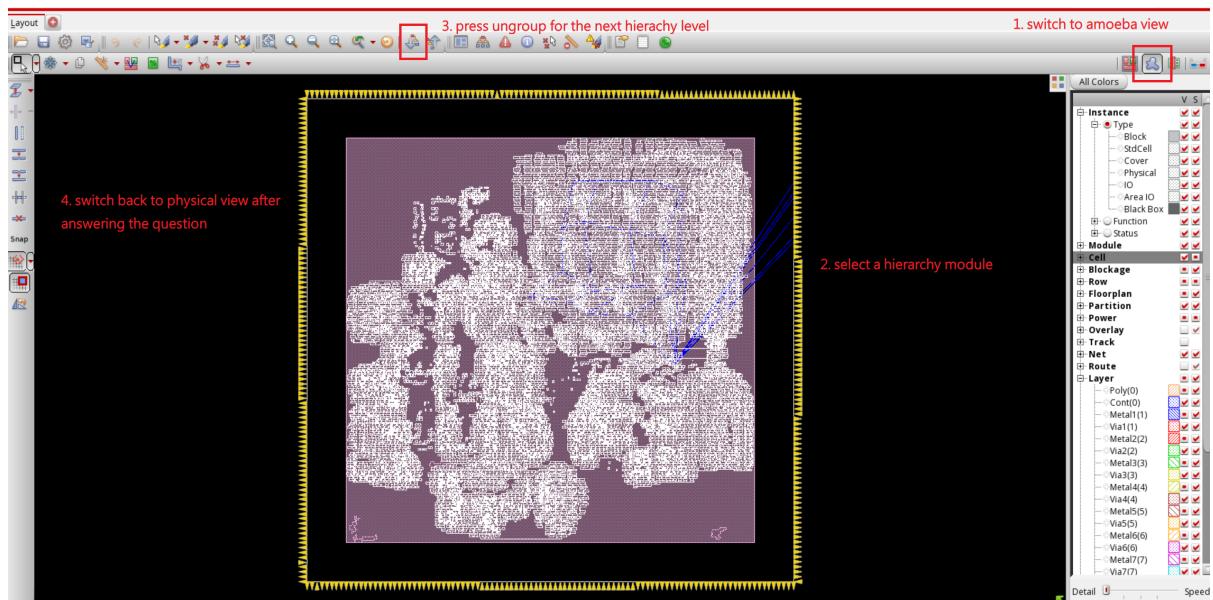
Q: How do standard cells connect with power/ground pins?



### 2. Show the hierarchy groups

Checkpoint 4: Show your layout in amoeba view in the report.

Q: How does the layout partitioned?



## 9. Save the design

File->Save Design



## Static Power Analysis and Early Rail Analysis

**Early Rail Analysis could be done at floorplanning stage, after placement, and postroute** to analyze power-grid integrity.

### Static Power Analysis

Static power analysis estimates the power consumption with the toggle rate of the signals. Although it can be conducted without any waveform, Innovus allows us to obtain a more accurate toggling rate with the waveform.

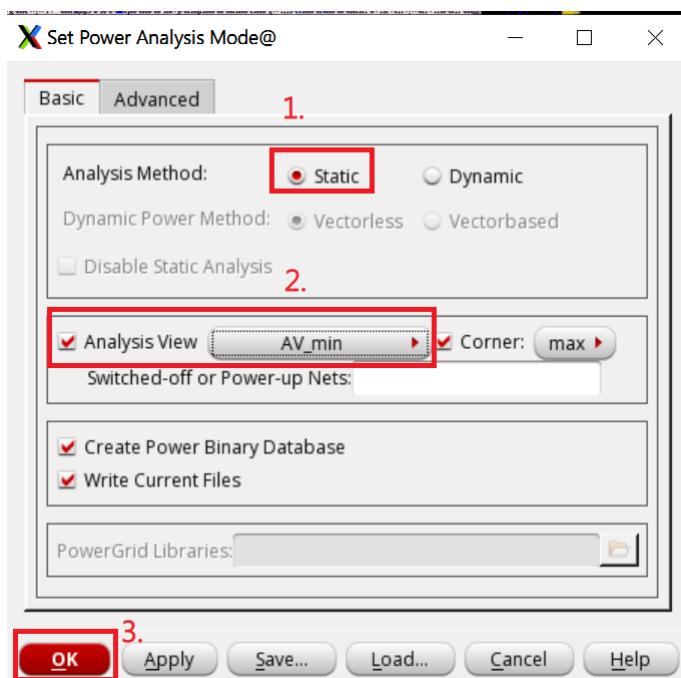
#### 1. Generate the waveform

Open another terminal

```
$ cd hw6/sim
$ make syn
```

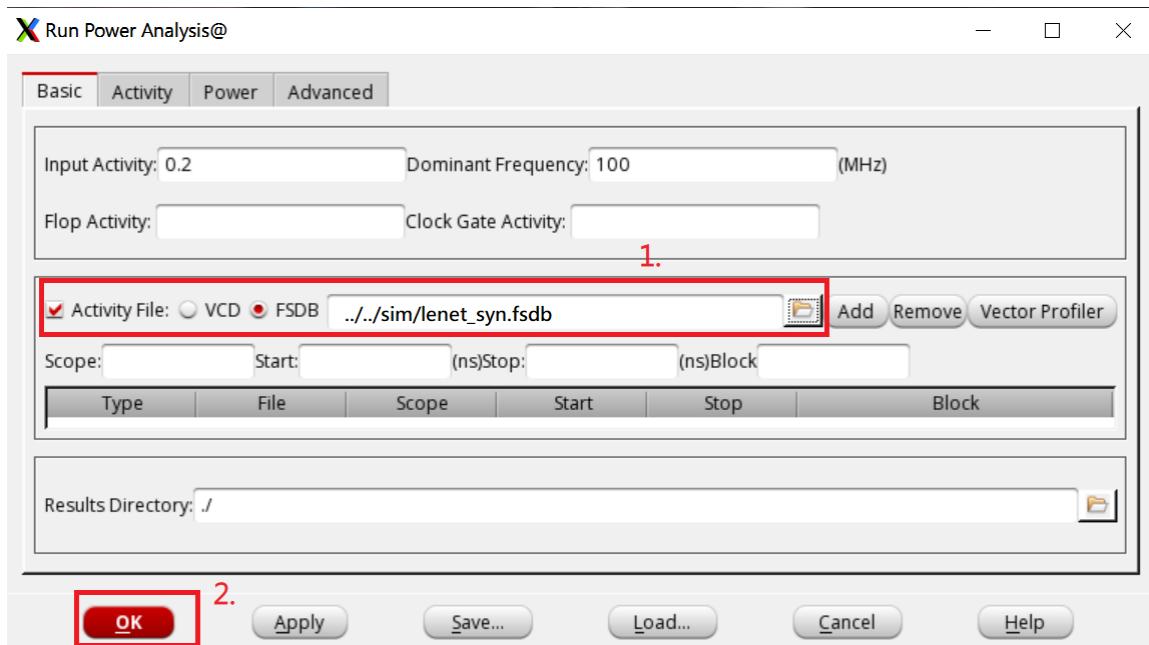
#### 2. Setup the power analysis

Power->Power Analysis->Setup



### 3. Run the power analysis

Power->Power Analysis->Run



### 4. Check the log in the terminal to find the power analysis result

**Total internal power (mW)**

**Total switching power (mW)**

**Total leakage power (mW)**

**Total power (mW)**

```
Total Power
-----
Total Internal Power:      0.63063174      73.6574%
Total Switching Power:    0.21659559      25.2982%
Total Leakage Power:       0.00894099      1.0443%
Total Power:                0.85616832

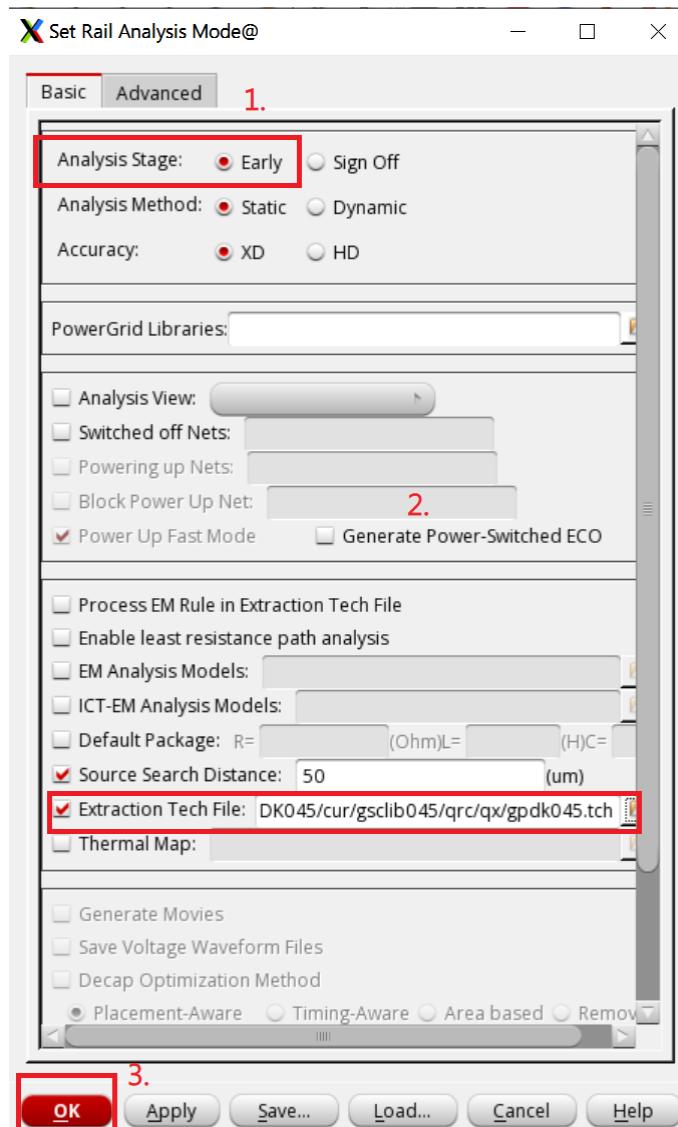
-----
Ended Static Power Report Generation: (cpu=0:00:00, real=0:00:00,
mem(process/total/peak)=1172.61MB/2261.07MB/1317.79MB)
```

## Rail Analysis

### 1. Setup the Rail analysis

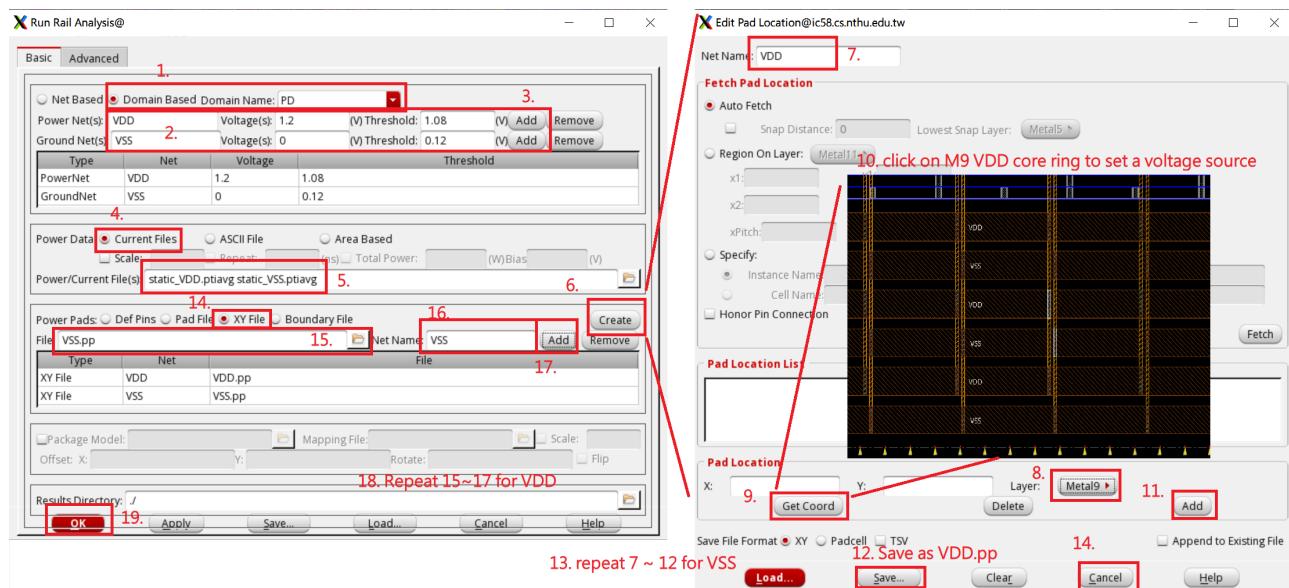
Power->Rail Analysis->Setup Rail Analysis

**Extraction Tech file:** /theda21\_2/library/GPDK045/cur/gsclib045/qrc/qx/gpdk045.tch



### 2. Run the rail analysis

Power->Rail Analysis->Run Rail Analysis

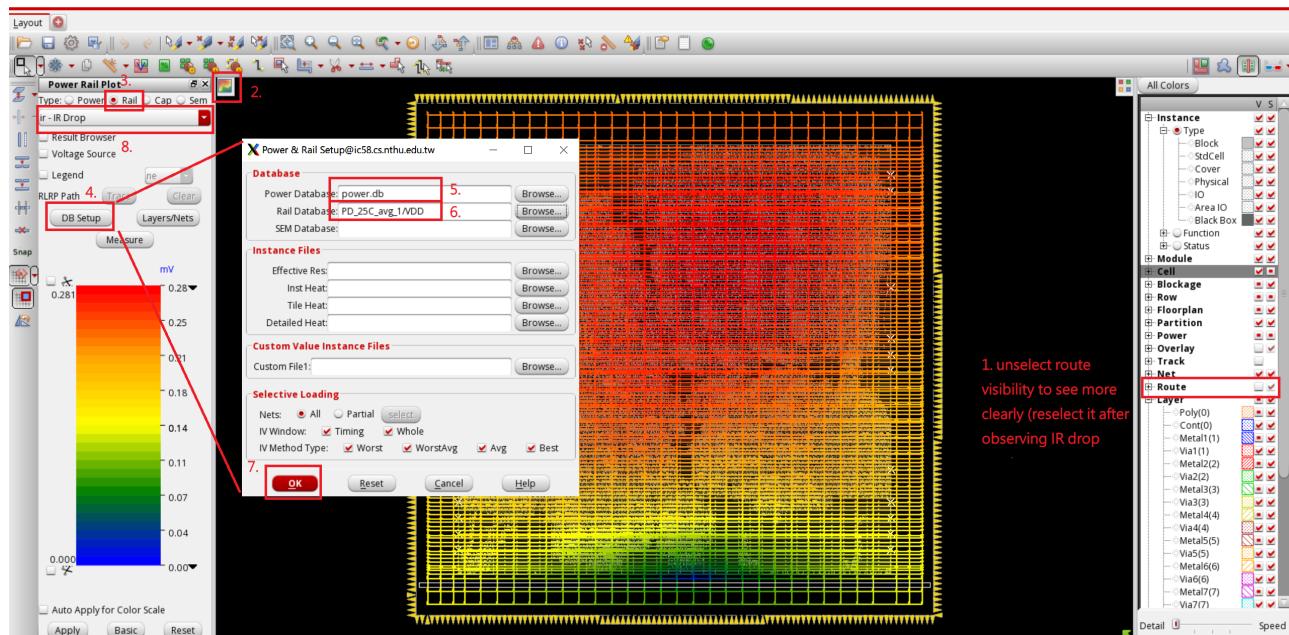


3. Check the log in the terminal to find the IR drop

4. Visualize the IR drop map

**Checkpoint 5: Show your IR drop map in the report.**

Power -> Report -> Power Rail Result



Q: What is the worst IR drop? (Optional: Is it acceptable?)

## Clock Tree Synthesis

1. Remove the clock latency from the SDC constraints

```
$innovus > set_interactive_constraint_modes [all_constraint_modes]
$innovus > reset_clock_latency [all_clocks]
```

2. Run the CTS (it might take about 30 minutes)

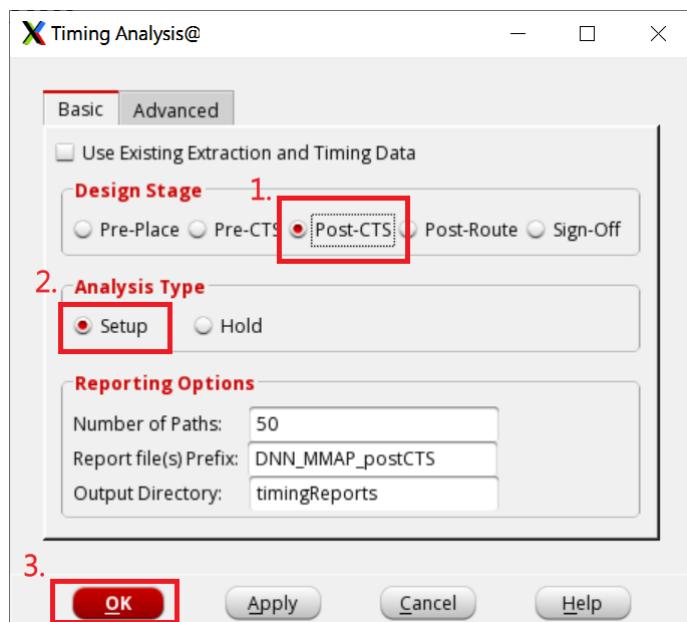
Understand the command in hw6/apr/cts/ccopt.cmd

```
$innovus > source ../../cts/ccopt.cmd
```

3. Report timing

1. Setup time

Timing->Report Timing



Check if setup time meets the timing requirement.

```

timeDesign Summary

Setup views included:
AV_max

+-----+-----+-----+-----+-----+-----+
| Setup mode | all | reg2reg | in2reg | reg2out | in2out | default |
+-----+-----+-----+-----+-----+-----+
| WNS (ns):| 31.985 | 33.610 | 33.433 | 31.985 | N/A | 0.000 |
| TNS (ns):| 0.000 | 0.000 | 0.000 | 0.000 | N/A | 0.000 |
| Violating Paths:| 0 | 0 | 0 | 0 | N/A | 0 |
| All Paths:| 6694 | 5712 | 2008 | 131 | N/A | 0 |
+-----+-----+-----+-----+-----+-----+

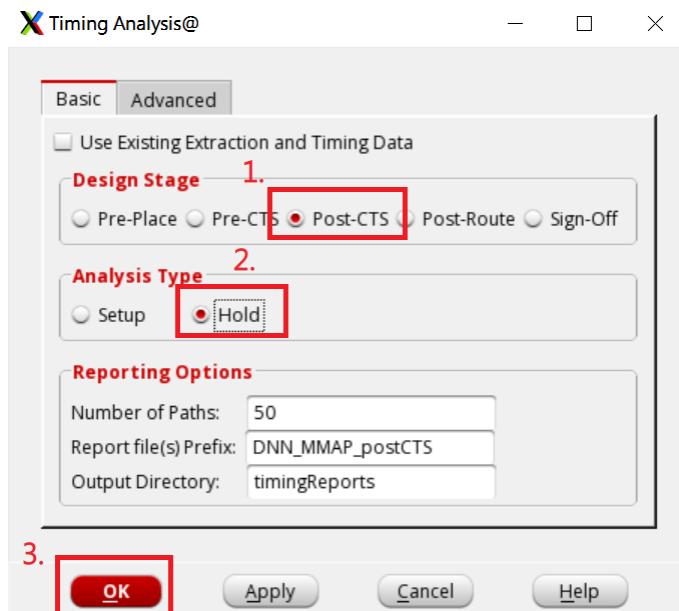
+-----+-----+-----+
| DRVs | Real | Total |
+-----+-----+-----+
| | Nr nets(terms) | Worst Vio | Nr nets(terms) |
+-----+-----+-----+
| max_cap | 0 (0) | 0.000 | 0 (0) |
| max_tran | 0 (0) | 0.000 | 0 (0) |
| max_fanout | 0 (0) | 0 | 26 (26) |
| max_length | 0 (0) | 0 | 0 (0) |
+-----+-----+-----+

Density: 52.727%
Routing Overflow: 0.00% H and 0.00% V

```

## 2. Hold time

### Timing->Report Timing



Check if hold time meets the timing requirement.

```

timeDesign Summary

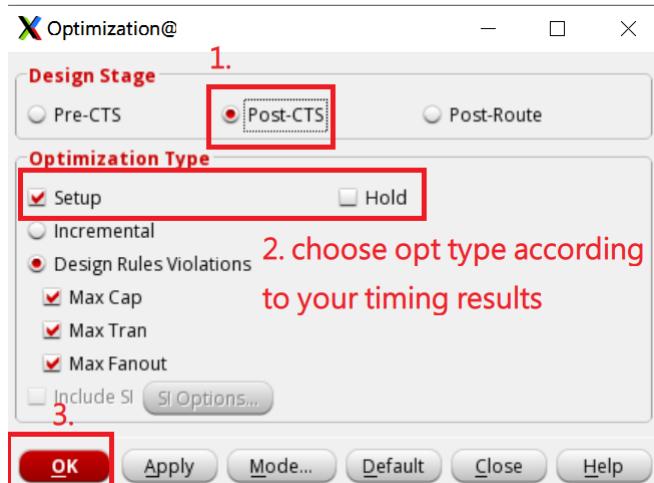
Hold views included:
AV_min

+-----+-----+-----+-----+-----+-----+-----+
| Hold mode | all | reg2reg | in2reg | reg2out | in2out | default |
+-----+-----+-----+-----+-----+-----+-----+
| WNS (ns): | 0.064 | 0.064 | 1.961 | 1.171 | N/A | 0.000 |
| TNS (ns): | 0.000 | 0.000 | 0.000 | 0.000 | N/A | 0.000 |
| Violating Paths: | 0 | 0 | 0 | 0 | N/A | 0 |
| All Paths: | 6694 | 5712 | 2008 | 131 | N/A | 0 |
+-----+-----+-----+-----+-----+-----+-----+

Density: 52.727%
Routing Overflow: 0.00% H and 0.00% V

```

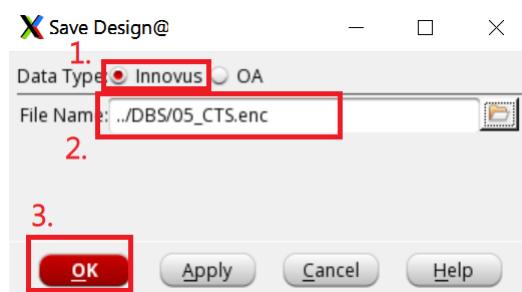
#### 4. (Optional) Optimize Design (if the timing did not pass during the previous steps.)



Then repeat 3.1 or 3.2 to check your design meets the timing requirements.

#### 5. Save the design

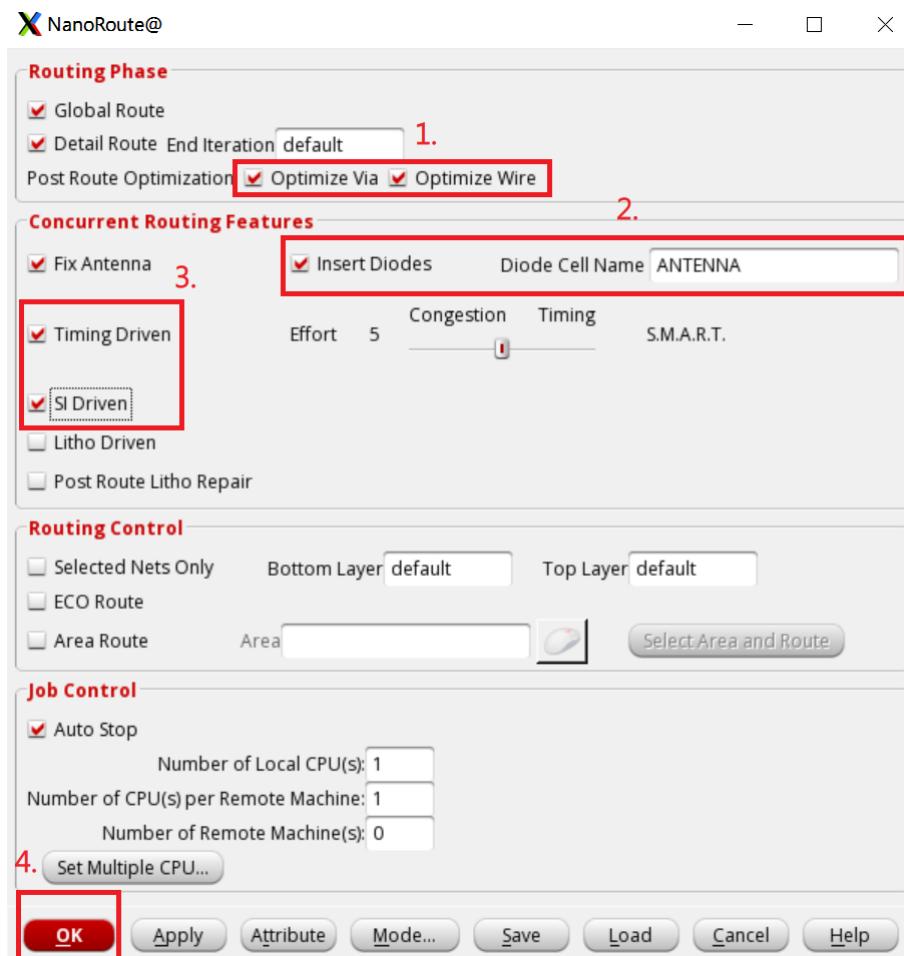
File->Save Design



## Route

#### 1. Perform the routing (it might take about 20-30 minutes)

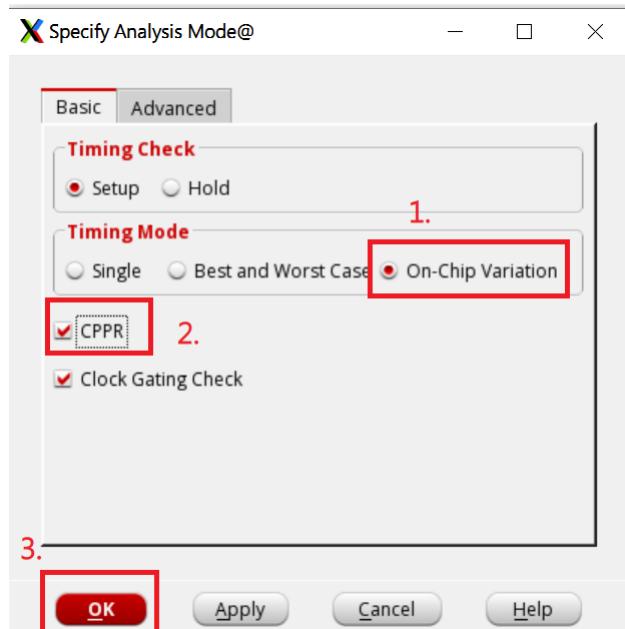
Route->NanoRoute->Route



## 2. Check the timing

1. Firstly, we need to change the timing analysis mode.

Tool->Set Mode->Specify Analysis Mode



## 2. Activate SI aware

```
$innovus > setDelayCalMode -SIAware true
```

### 3. Check the setup time

**Timing Analysis@**

Basic Advanced

Use Existing Extraction and Timing Data

**Design Stage**

Pre-Place  Pre-CTS  Post-CTS  Post-Route  Sign-Off

**Analysis Type**

Setup  Hold

**Reporting Options**

Number of Paths: 50  
Report file(s) Prefix: DNN\_MMAP\_postRoute  
Output Directory: timingReports

**OK** **Apply** **Cancel** **Help**

-----

timeDesign Summary

Setup views included:

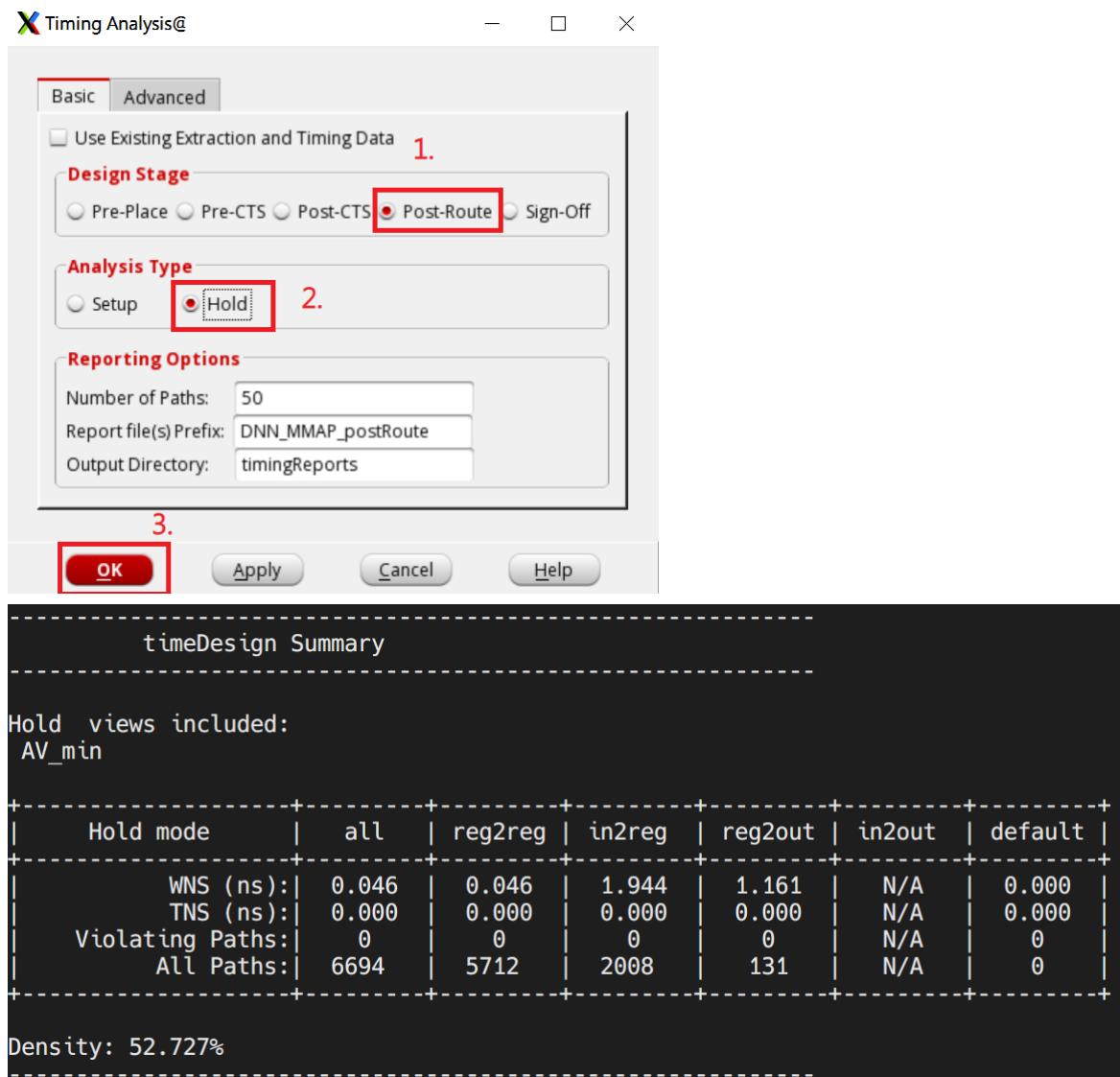
AV\_max

| Setup mode       | all    | reg2reg | in2reg | reg2out | in2out | default |
|------------------|--------|---------|--------|---------|--------|---------|
| WNS (ns):        | 32.035 | 33.455  | 33.366 | 32.035  | N/A    | 0.000   |
| TNS (ns):        | 0.000  | 0.000   | 0.000  | 0.000   | N/A    | 0.000   |
| Violating Paths: | 0      | 0       | 0      | 0       | N/A    | 0       |
| All Paths:       | 6694   | 5712    | 2008   | 131     | N/A    | 0       |

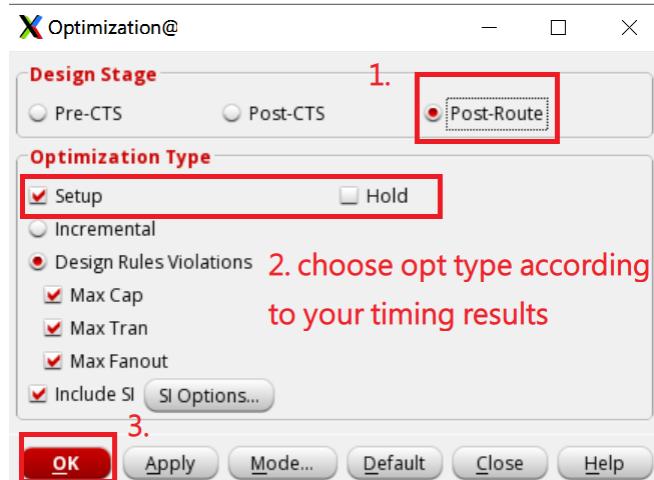
| DRVs       | Real           |           | Total          |
|------------|----------------|-----------|----------------|
|            | Nr nets(terms) | Worst Vio | Nr nets(terms) |
| max_cap    | 0 (0)          | 0.000     | 0 (0)          |
| max_tran   | 0 (0)          | 0.000     | 0 (0)          |
| max_fanout | 0 (0)          | 0         | 26 (26)        |
| max_length | 0 (0)          | 0         | 0 (0)          |

Density: 52.727%  
Total number of glitch violations: 0

### 4. Check the hold time



3. (Optional) Optimize Design (if the timing did not pass during the previous steps.)



Then repeat 2.3 or 2.4 to check your design meets the timing requirements.

1. Check the core utilization

```
$ innovus > checkFPlan -reportUtil
```

What is your final utilization?

Checkpoint 6: Show your core utilization log in the report.

```
Reporting Utilizations.....
Core utilization = 54.683274
Effective Utilizations
Average module density = 0.527.
Density for the design = 0.527.
    = stdcell_area 201481 sites (68907 um^2) / alloc_area 382118 sites (130684 um^2).
Pin Density = 0.2137.
    = total # of pins 85183 / total area 398610.
*** Message Summary: 0 warning(s), 0 error(s)
```

## 2. Verification

Check that no violation is reported in the terminal

### 1. DRC

```
$ innovus > verify_drc
```

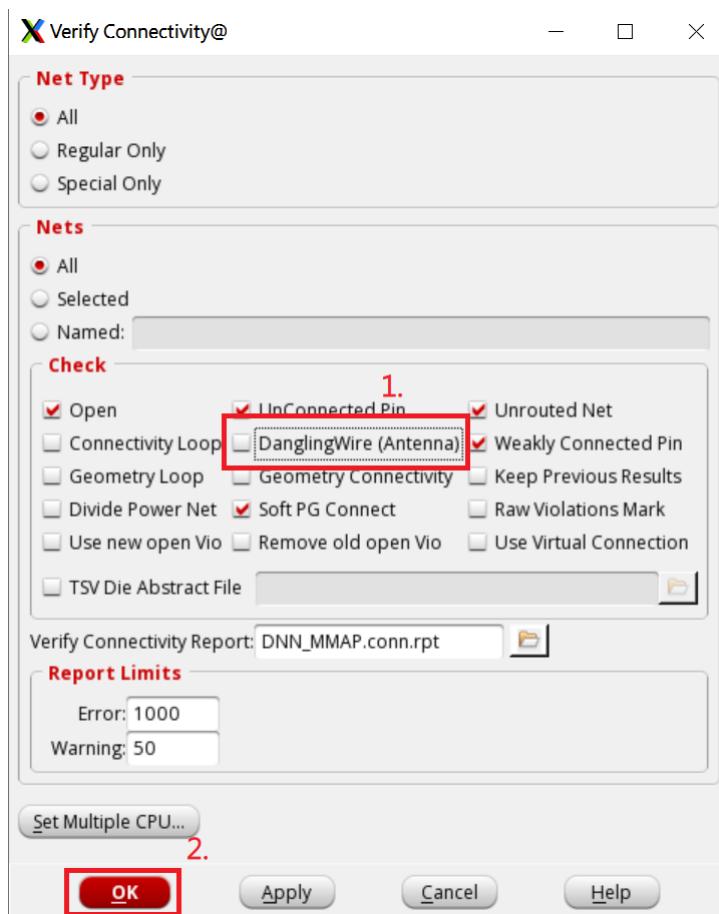
```
VERIFY DRC ..... Sub-Area: {369.600 353.280 442.000 438.710} 30 of 30
VERIFY DRC ..... Sub-Area : 30 complete 0 Viols.

Verification Complete : 0 Viols.

*** End Verify DRC (CPU: 0:00:18.4 ELAPSED TIME: 18.00 MEM: 0.5M) ***
```

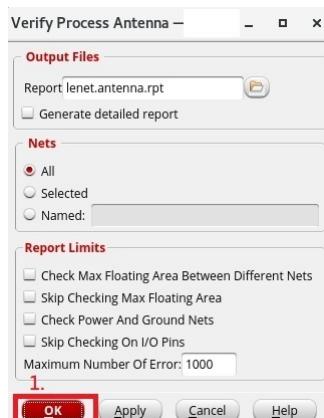
### 2. Connectivity

Verify->Verify Connectivity, cancel **DanglingWire (Antenna)**, press **OK**



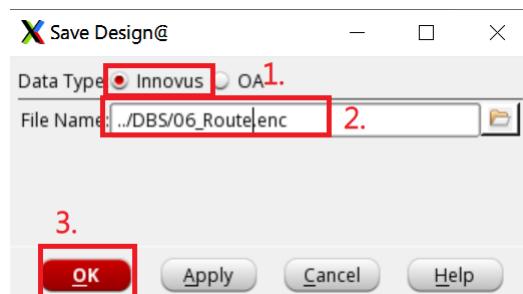
### 3. Antenna

Verify->Verify Process Antenna, press OK



### 3. Save the design

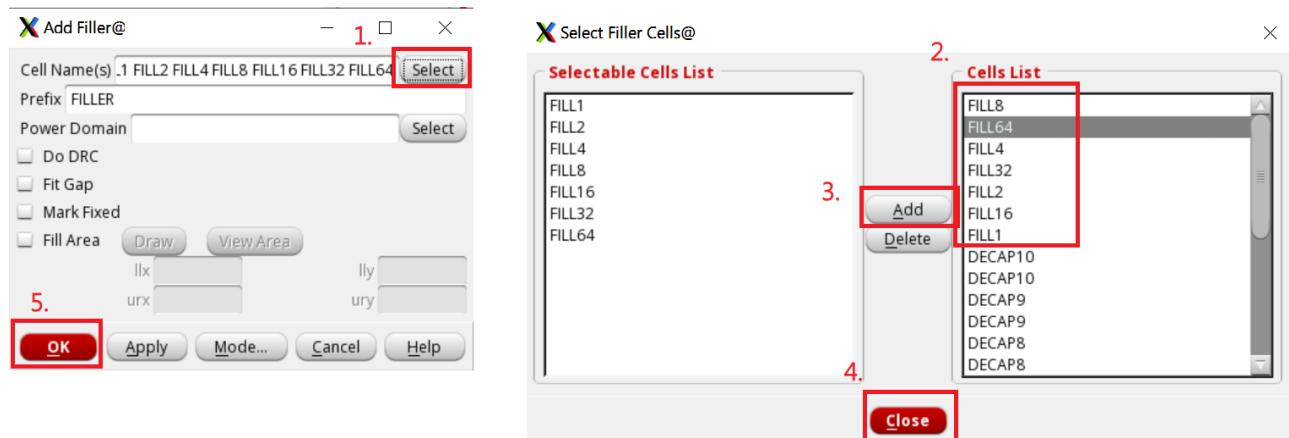
File->Save Design



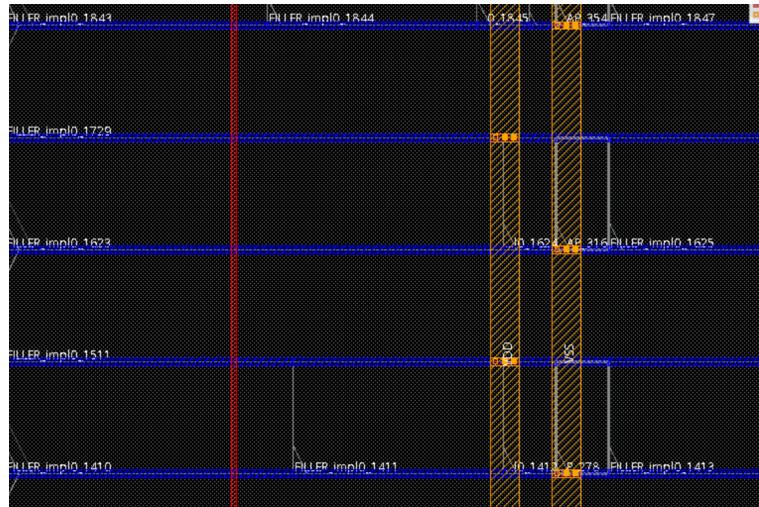
# Design for Manufacturability

## 1. Add Filler

Place->Physical Cells->Add Filler



## 2. Check if your layout has been filled by the fillers.



## Export the design

1. Export the netlist and the corresponding timing data in SDF (Standard Delay Format) for the post-layout simulation

```
$innovus > source ../ref_script/savesdf.cmd
```

2. Export the GDSII layout

```
$innovus > source ../ref_script/savegds.cmd
```

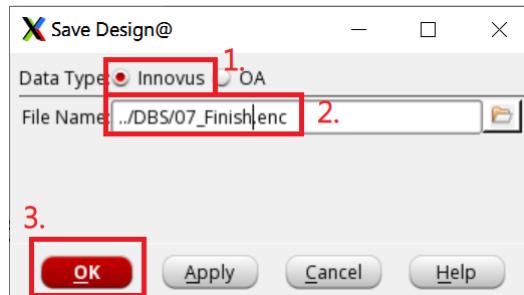
*Note:* The GDSII layout will be processed and verified by other tools in a practical backend flow. But we will not involve these operations in this lab.

3. Export the design constraint (SDC) and parasitic information (SPEF) for the following power simulation

```
$innovus > write_sdc ../netlist/CHIP_layout.sdc
$innovus > setExtractRCMode -engine postRoute
$innovus > reset_parasitics
$innovus > extractRC
$innovus > rcOut -rc_corner RC -spef ../netlist/CHIP_layout.gz
```

4. Save the design

File->Save Design



For a more detailed Innovus guide, type the following command in the terminal:

```
$unix > /usr/cadence/INNOVUS/cur/bin/cdnshelp &
```

## Post-layout simulation

1. In the **post\_sim/** directory, read these files (Makefile, header) and understand how the simulation environment is configured. *Note:* In the testbench, we need to annotate the SDF file exported by Innovus to the CHIP instance (see lines 492-496). The SDF file records the delay information of gates and nets.
2. Run the simulation with the following commands, and then check whether the simulation result is correct or not.

**Checkpoint 7: Show your post simulation result in the report.**

```
$ cd hw6/sim
$ make post
```

## Logic Equivalence Check

Assume we have verified the logic equivalence between RTL and gate-level netlist before P&R (you can also run LEC by yourself for the RTL and gate-level netlist for a complete verification flow). After the physical design, we need to check the logic equivalence between the gate-level netlist and layout by Conformal LEC since running all possible cases in post-layout simulation is not affordable.

1. In the **lec/** directory, read all the files and understand how the environment is configured.
2. Run the LEC between the gate-level and post-layout netlist with the following commands, and then check whether they are equivalent or not.

**Checkpoint 8:** Show your LEC result in the report.

```
$ cd hw6/apr/lec/
$ source /usr/cad/cadence/ANLS/CIC/confrm1.cshrc
$ ./run_lec.bat
```

*Note:* We recommend running LEC on ic55 and ic56 machines. *Note:* If you encounter Permission deny, try

```
$ chmod u+x run_lec.bat
```

## Time-based Power Analysis with PrimeTime

Unlike the static power analysis in Design Compiler and Innovus, time-based power analysis estimates the power by the actual waveform. The dynamic signal activities make the estimation more realistic.

To run the power analysis, you need to prepare the following files:

- Netlist(\*.v, generated by DC/Innovus)
- Design Constraint file (\*.sdc)
- Waveform (\*.fsdb), waveform by the pre-/post-layout simulation
- SPEF file (Standard Parasitic Exchange format, generated by Innovus)

1. Pre-layout power analysis Practically, this step is done before the APR to estimate the power needed. However, we perform the pre-layout power analysis after the APR simply for convenience.

1. Run the gate-level simulation and obtain the waveform.

```
$ cd hw6/sim
$ make syn
```

2. Run the pre-layout power estimation

**Checkpoint 9:** Show the time-based power analysis result for your pre-layout in the report.

Summit "**pre\_layout\_power\_report.rpt**"

```
$ cd hw6/apr/primetime/pre_layout
$ fsdb2vcf ../../sim/lenet_syn.fsdb -o ../../sim/lenet_syn.vcd
#convert fsdb to vcd
$ pt_shell -f pt_px.tcl
```

**Note:** SPEF is not needed here because we don't have physical information about wiring. The SPEF file records parasitic information of the post-layout netlist, which provides essential parasitic information to PrimeTime.

## 2. Post-layout power analysis with the post-sim waveform

- Run the post simulation again, and obtain the waveform.

```
$ cd hw6/sim/
$ make post
```

- Modify the SDC generated by Innovus. Some statements are not supported in PrimeTime.

Open hw6/apr/netlist/CHIP\_layout.sdc

Comment the following two lines out

```
# current_design lenet
...
# set_max_fanout 20 [get_designs {lenet}]
```

## 3. Run the post-layout power estimation with the post-sim waveform

**Checkpoint 10:** Show the time-based power analysis result for your post-layout with post-sim waveform in the report.

Summit "[post\\_sim\\_power\\_report.rpt](#)"

```
$ cd hw6/apr/primetime/post_sim_waveform
$ pt_shell -f pt_px.tcl
```

## 3. Post-layout power analysis with the pre-sim waveform

- Run the RTL simulation, and obtain the waveform.

```
$ cd hw6/sim
$ make sim
```

- Use the same SDC as the previous step. Make sure the SDC file has been properly modified.

- Run the post-layout power estimation with the pre-sim waveform

**Checkpoint 11:** Show the time-based power analysis result for your post-layout with pre-sim waveform in the report.

#### Summit "pre\_sim\_power\_report.rpt"

```
$ cd hw6/apr/primetime/pre_sim_waveform
$ pt_shell -f pt_px.tcl
```

*Note:* When we want to use the pre-sim waveform for power analysis in PrimeTime, we have to add the argument "**-rtl**" in "read\_vcd" command (see line 20 in pt\_px.tcl) to specify that the VCD/FSDB file comes from an RTL simulation.

## PDF report

1. Briefly explain what each step does in the APR flow.

- a. Design setup
- b. Floorplan
- c. Powerplan
- d. Placement
- e. Clock tree synthesis
- f. Route
- g. Filler adding

2. What is the purpose of these generated files in **hw6/apr/netlist/** ?

- a. CHIP.v
- b. CHIP\_layout.sdc
- c. CHIP\_layout.sdf
- d. CHIP\_layout.gds
- e. CHIP\_layout.gz

3. Please explain why the RTL simulation (**make sim**) doesn't need an SDF file, but the pre-layout (**make syn**) and post-layout simulation (**make post**) require this file.

4. Please compare three kinds of power analysis in your report. Also, try to explain why they have different results.

5. You should include the screenshots of every checkpoint.

**That's it! Happy APR-ing!!**

