

# NCTU-EE IC Design LAB – Fall2023

## Lab12 Exercise: Innovus - From RTL to GDSII

In this exercise, you are going to perform the APR steps to the design : local binary pattern

### 1. Data Preparation

1. Complete rtl simulation, synthesis, and gate level simulation.
2. Change the directory to **05\_APR**
3. Prepare chip design netlist:
  - a. Open **CHIP\_SHELL.v**
    - ☐ The top module name is **CHIP**.
    - ☐ This **CHIP** contains the module **CAD**; and **I/O**, **I/O power**, **core power** pad.
    - ☐ Please calculate how many I/O and core power pads you need, and complete the netlist of pad cells (**size** and **action**).
    - ☐ After defining the pad cells, please run **%.00\_combine** to combine the CAD\_SYN.v with CHIP\_SHELL.v to be CHIP\_SYN.v
4. Prepare I/O pad location file:
  - a. Please see **CHIP.io** to know how to assign I/O pad location.
  - b. Open **CHIP.io** and complete the I/O pad location assignment according to the netlist of pad cells in **CHIP\_SYN.v** (**size** and **action**).
5. Prepare timing/driving/loading constraint file **CHIP.sdc**:
  - a. Copy it from the synthesis result:
    - ☐ Comment out the set\_clock\_uncertainty & set\_clock\_transition instruction

```
create_clock [get_ports clk] -period 20 -waveform {0 1}
#set_clock_uncertainty 0.1 [get_clocks clk]
#set_clock_transition -max -rise 0.1 [get_clocks clk]
#set_clock_transition -max -fall 0.1 [get_clocks clk]
#set_clock_transition -min -rise 0.1 [get_clocks clk]
#set_clock_transition -min -fall 0.1 [get_clocks clk]
set_input_delay -clock clk 0 [get_ports clk]
```
    - ☐ **cp ../02\_SYN/Netlist/CAD\_SYN.sdc CHIP.sdc**
    - ☐ Modify the contents to the desired constraint
6. Prepare linked library files:
  - a. **Timing libraries (LIB)**
    - ☐ **fsa0m\_a\_generic\_core\_ss1p62v125c.lib**
    - ☐ **fsa0m\_a\_generic\_core\_ff1p98vm40c.lib**
    - ☐ **fsa0m\_a\_t33\_generic\_io\_ss1p62v125c.lib**

- ☐ fsa0m\_a\_t33\_generic\_io\_ff1p98vm40c.lib
- ☐ Memory\_WC.lib
- ☐ Memory\_BC.lib
- b. Physical libraries (LEF)
  - ☐ header6\_V55\_20ka\_cic.lef
  - ☐ fsa0m\_a\_generic\_core.lef
  - ☐ FSA0M\_A\_GENERIC\_CORE\_ANT\_V55.lef
  - ☐ fsa0m\_a\_t33\_generic\_io.lef
  - ☐ FSA0M\_A\_T33\_GENERIC\_IO\_ANT\_V55.lef
  - ☐ BONDPAD.lef
  - ☐ Memory.lef
- c. RC extraction table/files
  - ☐ u18\_Faraday.CapTbl
  - ☐ icecaps.tch
- d. CeltIC libraries
  - ☐ u18\_ss.cdb
  - ☐ u18\_ff.cdb
- e. GDSII layout (Will not stream out in this course)
  - ☐ u18.gds2
  - ☐ u18io3v5v\_6lm.gds2
  - ☐ Memory.gds2 (not provided in this compiler version)

## 2. Reading Cell Library information and Netlist for APR

1. If you are running on ee servers, type below command on terminal to set OA\_HOME.

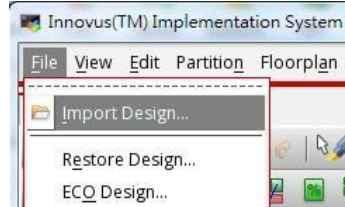
```
$ setenv OA_HOME
/RAID2/cad/cadence/INNOVUS/INNOVUS_20.15.000/share/oa
```

2. Start Innovus in the directory 05\_APR
  - % innovus (no &, do NOT use background execution)
3. Set uniqueness global variable to 1
  - % set init\_design\_uniquify 1
4. Change design mode to 0.18um process
  - % setDesignMode -process 180
5. Skip error messages like the following figure when reading lef files
  - % suppressMessage TECHLIB 1318
  - % suppressMessage ENCEXT-2799

```
**ERROR: (TECHLIB-1318): All the table values in the 'rise_transition' group are within '0.000010' of each other. (File /home/RAID2/COURSE/iclab/iclabta01/umc018/Lib/umc18io3v5v_slow.lib, Line 682)
**ERROR: (TECHLIB-1318): All the table values in the 'fall_transition' group are within '0.000010' of each other. (File /home/RAID2/COURSE/iclab/iclabta01/umc018/Lib/umc18io3v5v_slow.lib, Line 690)
**ERROR: (TECHLIB-1318): All the table values in the 'rise_transition' group are within '0.000010' of each other. (File /home/RAID2/COURSE/iclab/iclabta01/umc018/Lib/umc18io3v5v_slow.lib, Line 782)
```

This error message occurs since innovus assumes the values of rise / fall transition time should differ larger than 0.00001 ns in **the table** depending on the supply Voltage or the Temperature **in .lef file**. The error message can be ignored if the given .lef file is ensured to be correct. (Note: only suppress error when the reason is clearly verified instead of suppressing all errors)

6. In innovus menu, open **File → Import Design**



7. Fill the following field:

a. Netlist

◆ Verilog

☐ Files: **CHIP\_SYN.v**

☐ Top Cell: ◆ By User : **CHIP**

b. Technology / Physical Libraries

☐ LEF Files: **header6\_V55\_20ka\_cic.lef**  
**fsa0m\_a\_generic\_core.lef**  
**FSA0M\_A\_GENERIC\_CORE\_ANT\_V55.lef**  
**fsa0m\_a\_t33\_generic\_io.lef**  
**FSA0M\_A\_T33\_GENERIC\_IO\_ANT\_V55.lef**  
**BONDPAD.lef**  
**Memory.lef**

Note that (1) the standard cell lef file should be put in the first place since it contains the major metals / vias / polys technology definitions. Also (2) if there are antenna lef files, they should be put after the corresponded lef files without describing antenna layers. For example: umc18\_6lm.lef should be put in the first place; umc18\_6lm\_antenna.lef should be put after umc18\_6lm.lef.

c. Floorplan

☐ IO Assignment Files: **CHIP.io**

d. Power

☐ Power Nets: **VCC**

☐ Ground Nets: **GND**

e. Analysis Configuration

Press "Create Analysis Configuration" tab

☐ Double click **Library Sets** and include the max and min delay library :

**Max delay:**

Name: lib\_max

Timing Library:

**fsa0m\_a\_generic\_core\_ss1p62v125c.lib**

**fsa0m\_a\_t33\_generic\_io\_ss1p62v125c.lib**

**ib**

### Memory\_WC.lib

SI Library: **u18\_ss.cdb**

**Min delay:**

Name: lib\_min

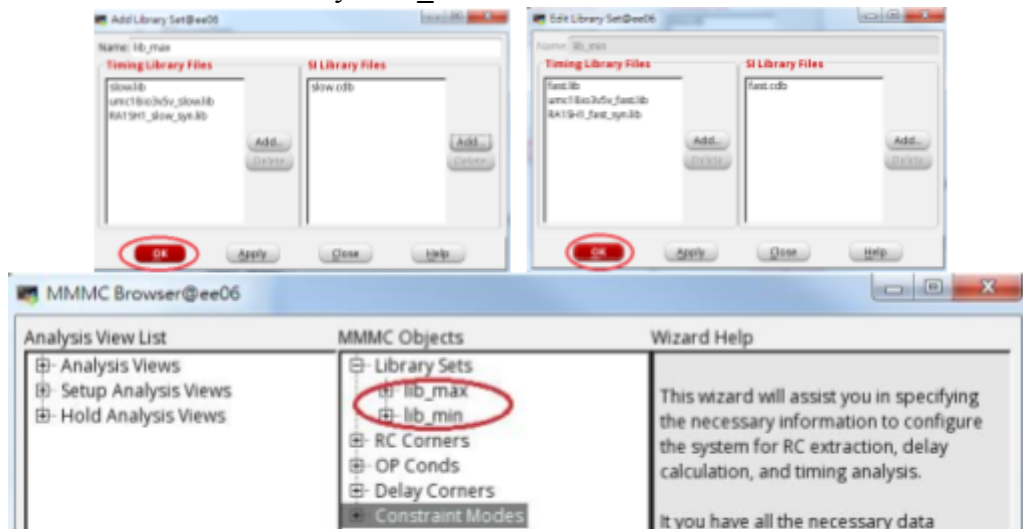
Timing Library:

**fsa0m\_a\_generic\_core\_ff1p98vm40c.lib**

**fsa0m\_a\_t33\_generic\_io\_ff1p98vm40c.lib**

### Memory\_BC.lib

SI Library: **u18\_ff.cdb**



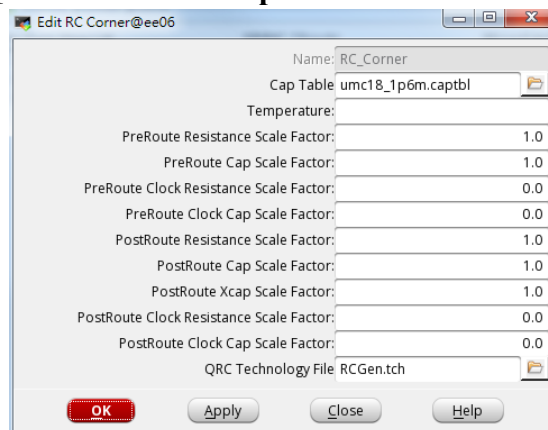
- Double click **RC Corners** to include the RC corner library

: Name: RC\_Corner

Cap Table:

**u18\_Faraday.CapTbl**

QRC Tech File: **icecaps.tch**



- Double click **Delay Corners** and create max and min delay constraints :

**Max delay:**

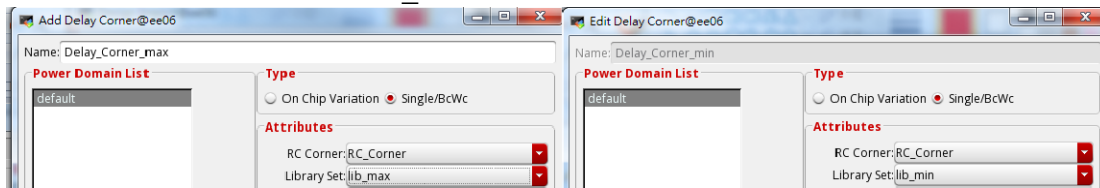
Name: Delay\_Corner\_max

RC Corner: RC\_Corner

Lib Set: lib\_max

## Min delay:

Name: Delay\_Corner\_min  
RC Corner: RC\_Corner  
Lib Set: lib\_min



- Double click **Constraints Mode** and create a function mode to place CHIP\_SYC.sdc :

Name: func\_mode  
SDC Constraint Files: CHIP.sdc



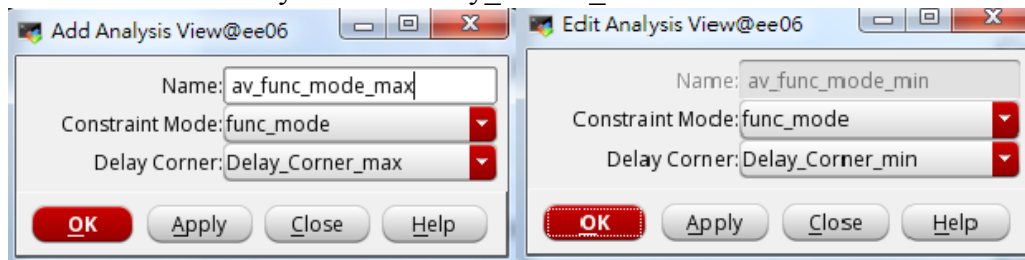
- Double click **Analysis Views** to create max and min delay analysis

## Max delay:

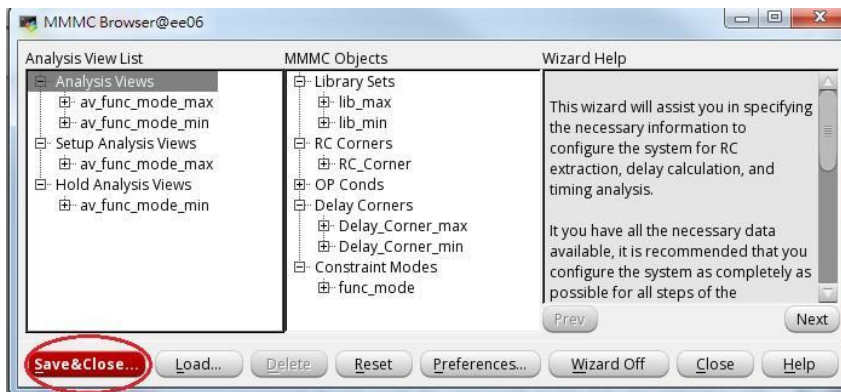
Name: av\_func\_mode\_max  
Constraint Mode: func\_mode  
Delay Corner: Delay\_Corner\_max

### Min delay:

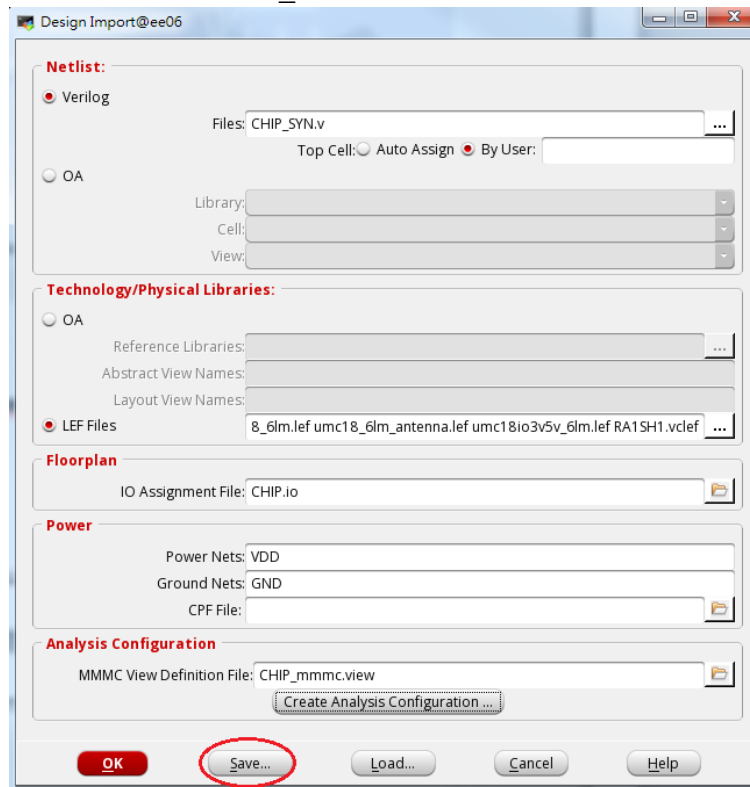
Name: av\_func\_mode\_min  
Constraint Mode: func\_mode  
Delay Corner: Delay\_Corner\_min



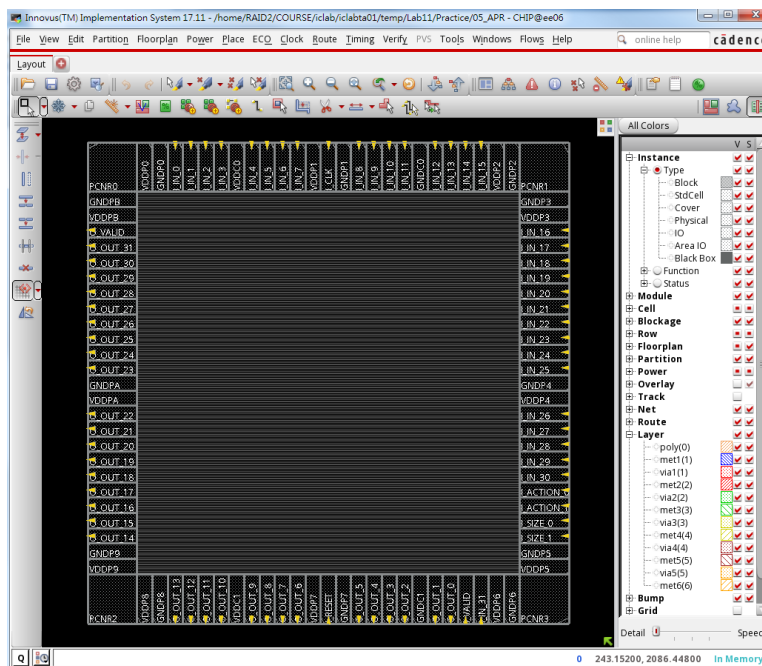
- Click **Setup Analysis View** and specify the max analysis mode  
Choose: av\_func\_mode\_max
- Click **Hold Analysis View** and specify the min analysis mode  
Choose: av\_func\_mode\_min



- Save as “CHIP\_mmmc.view”

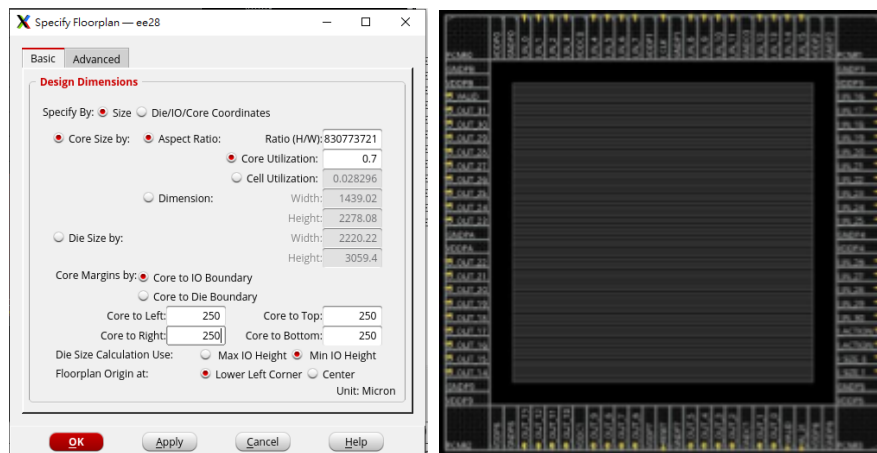


- Save current settings:
  - Click **Save...** button
    - File name: **CHIP.globals**
 If you want to reload the settings, you can just press Load... button
- Click **OK** button



### 3. Specify Chip Floorplan

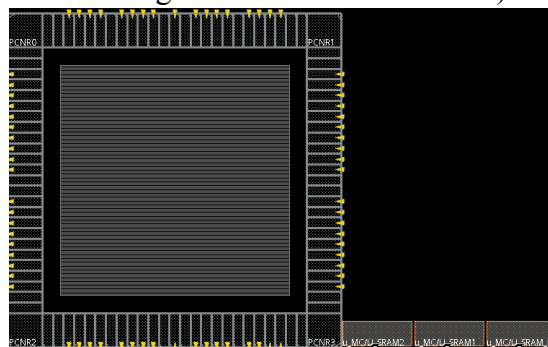
1. In innovus menu, open **Floorplan** → **Specify Floorplan**
2. Specify core size:
  - a. Core Utilization: Set any as your wish (0.7~0.8 is suggested)
3. Specify core margin:
  - b. ♦ Core to IO  
 Boundary Core to Left:  
**250**  
 Core to Right: 250  
 Core to Top: 250  
 Core to Bottom: 250
4. Apply the specification:
  - c. Click **OK** button



5. You can view designs in three ways; change to floorplan view:  
 The memory macro should appear (Besides zoom in and zoom out button

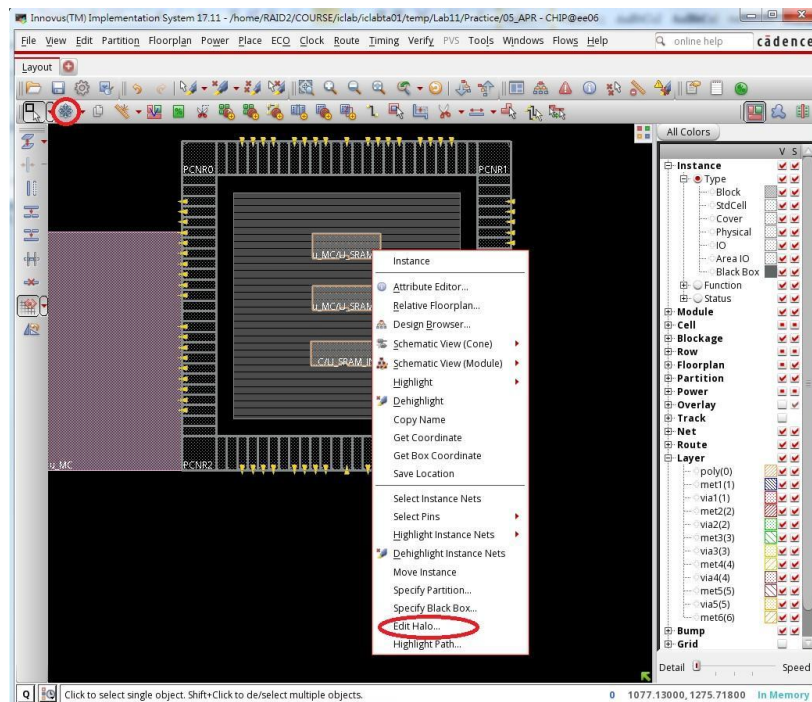


, you can use right click to zoom the view)



6. Move them to the desired places and make placement blockage:  
 right click on one macro, Choose **"Edit Halo"**

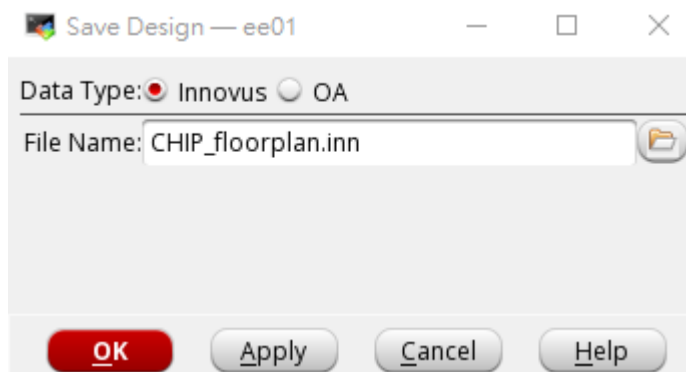




7. Specify Halo For: ◆ All Macros  
Add/Update Halo: Top, Bottom, Left, Right: 15um
8. You can also create the placement blockage yourself with the button:



9. Save the floorplan design:
    - a. File → Save Design
    - b. Data Type ◆ Innovus
- File Name: CHIP\_floorplan.inn
- c. Click **OK** button



If you do the wrong things in the following steps such as power planning, you can restore the design from the previous steps. However if the input files such as verilog file / io file / timing constraints are changed, you have to rerun all the APR steps.

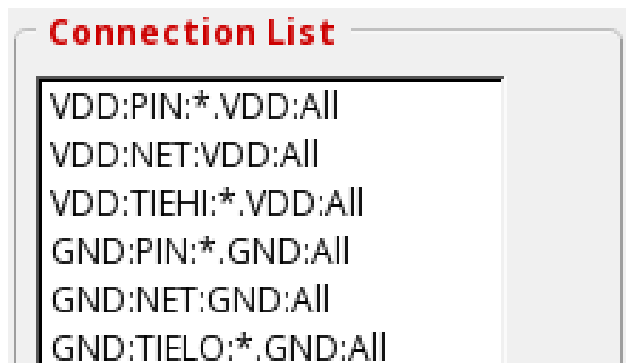
## 4. Connect/Define Global Net

1. In innovus menu, open **Power** → **Connect Global Nets...**



2. Add all VDD pins to Connection List:
  - a. Connect ◆ Pin
    - Instance Basename: \*
    - Pin Name(s): **VCC**
  - b. Scope ◆ Apply All
  - c. To Global Nets: VCC
  - d. Click Add to List button
3. Add all VDD nets to Connection List:
  - a. Connect ◆ Net Basename: **VCC**
    - b. Scope ◆ Apply All
    - c. To Global Nets: VCC
    - d. Click Add to List button
4. Add all Tie High pins to Connection List:
  - a. Connect ◆ Tie High
  - b. Scope ◆ Apply All
  - c. To Global Nets: VCC
  - d. Click Add to List button
5. Add all GND pins to Connection List:
  - a. Connect ◆ Pin
    - Instance Basename: \*
    - Pin Name(s): **GND**
  - b. Scope ◆ Apply All
  - c. To Global Nets: GND
  - d. Click Add to List button
6. Add all GND nets to Connection List:
  - a. Connect ◆ Net Basename: **GND**
    - b. Scope ◆ Apply All
    - c. To Global Nets: GND
    - d. Click Add to List button
7. Add all Tie Low pins to Connection List:
  - a. Connect ◆ Tie Low
  - b. Scope ◆ Apply All
  - c. To Global Nets: GND

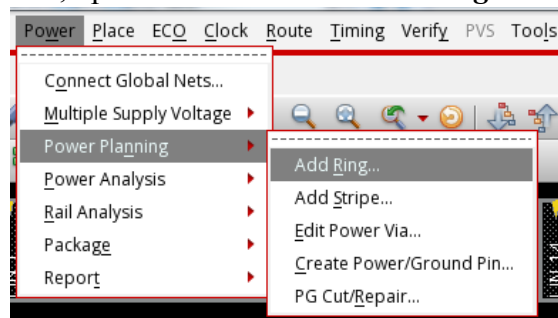
- d. Click Add to List button
- e. Click Add to List button



8. Apply the connection list and check:
  - e. Click **Apply** button
  - f. Click **Check** button
  - g. Click **Cancel** button

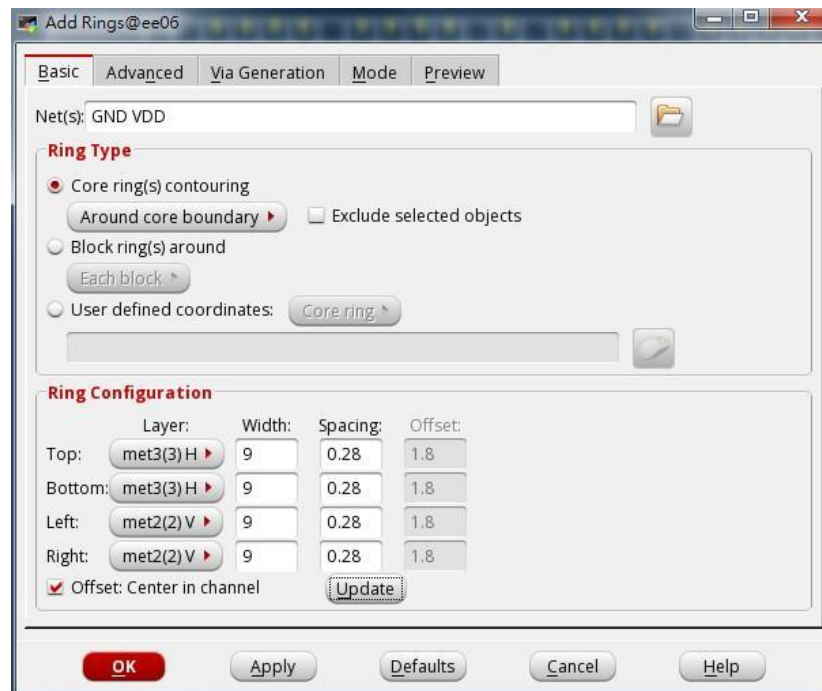
## 5. Power Planning (Add Core Power Rings)

1. In innovus menu, open **Power** → **Power Planning** → **Add Rings...**

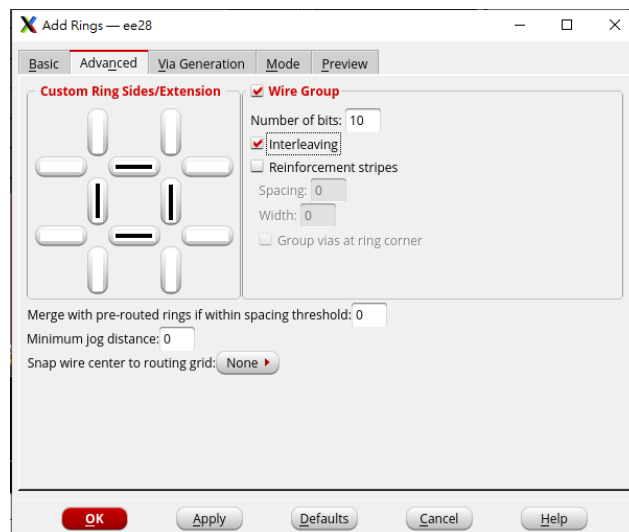



2. In the **Basic** tab, fill the following field:
  - a. Net(s): GND VCC
  - b. Ring Type: Core ring(s) contouring
  - c. Specify metal layers and width

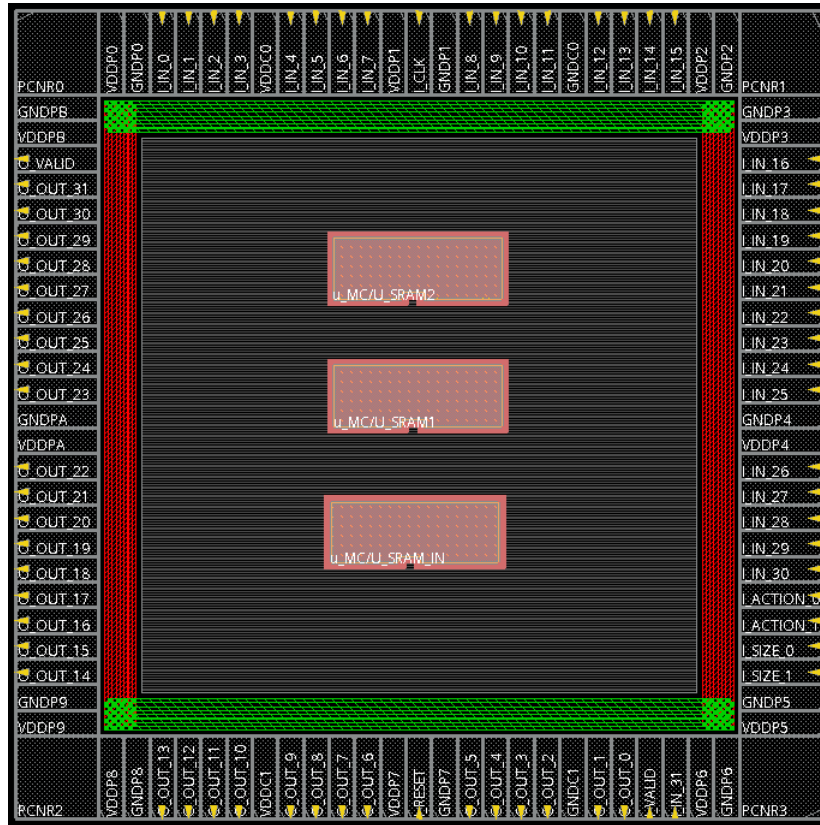
- |  |                 |
|--|-----------------|
| <input type="checkbox"/> Top Layer: <b>metal3 H</b>    | Width: <b>9</b> |
| <input type="checkbox"/> Bottom Layer: <b>metal3 H</b> | Width: <b>9</b> |
| <input type="checkbox"/> Left Layer: <b>metal2 V</b>   | Width: <b>9</b> |
| <input type="checkbox"/> Right Layer: <b>metal2 V</b>  | Width: <b>9</b> |
| <input type="checkbox"/> ◆ Offset: Center in channel   |                 |
| <input type="checkbox"/> Click <b>Update</b> button    |                 |



3. In the **Advanced** tab, fill the following field:
  - a. ♦ Wire Group
  - b. Number of bits: 10
  - c. ♦ Interleaving
4. Click **OK** button



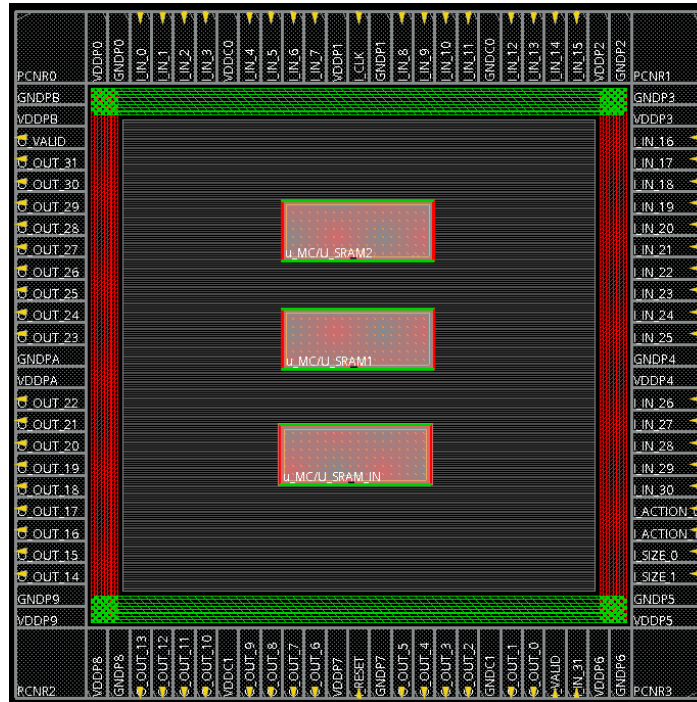
5. Check if the ring is correctly created. If not, click **undo** button  and repeat step 2~4 again.



## 6. Power Planning (Add Block Rings)

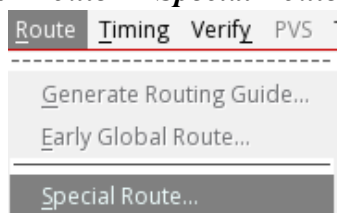
1. In innovus menu, open **Power** → **Power Planning** → **Add Rings...**
2. In the **Basic** tab, fill the following field:
  - a. Net(s): GND VCC
  - b. Ring Type: Block ring(s) around
  - c. Specify metal layers and width
 

<input type="checkbox"/> Top Layer: <b>met3(3) H</b>	Width: 2
<input type="checkbox"/> Bottom Layer: <b>met3(3) H</b>	Width: 2
<input type="checkbox"/> Left Layer: <b>met2(2) V</b>	Width: 2
<input type="checkbox"/> Right Layer: <b>met2(2) V</b>	Width: 2
<input type="checkbox"/> <b>◇ Offset: Center in channel</b>	
<input type="checkbox"/> Click <b>Update</b> button	
3. In the **Advanced** tab, disable the wire group:
  - a. **◇ Wire Group**
4. Click **OK** button



## 7. Connect Core Power Pin

1. In innovus menu, open **Route** → **Special Route...**



2. Connect core power: In **Basic** tab,
  - a. Net(s): GND VDD
  - b. Set the following configuration

**SRoute:**

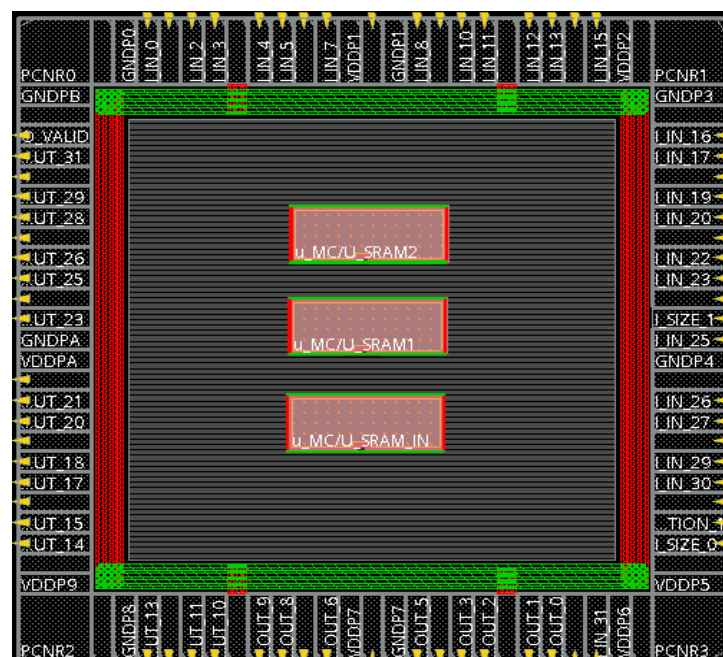
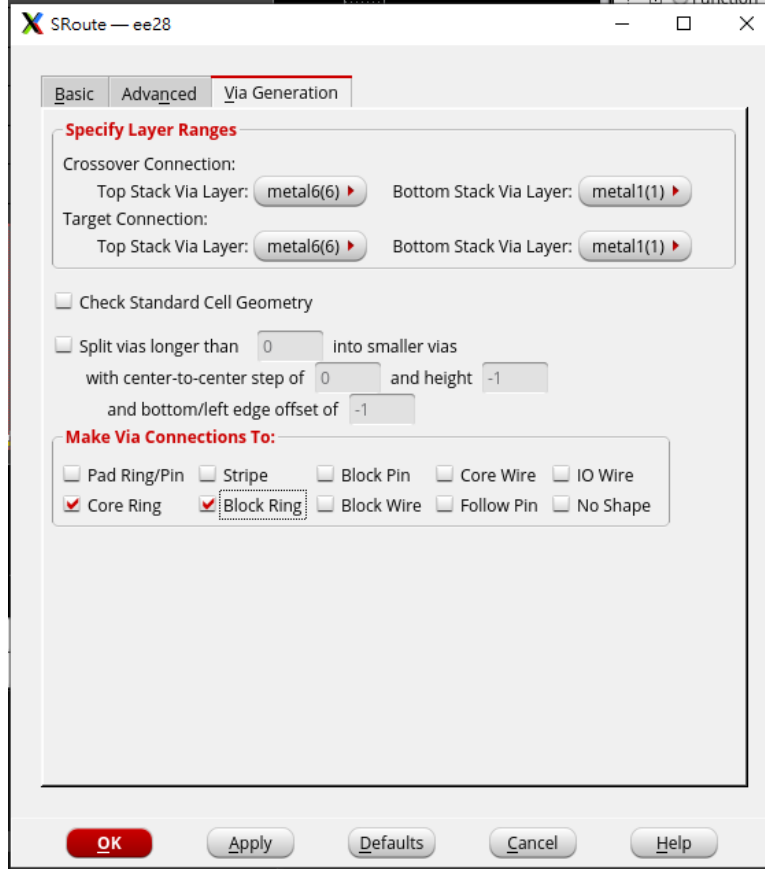
- ☐ ◆ Block pins
- ☐ ◇ Pad rings
- ☐ ◇ Floating Stripes
- ☐ ◆ Pad pins
- ☐ ◇ Follow

pins **In Via Generation** tab,

- c. Set the following configuration

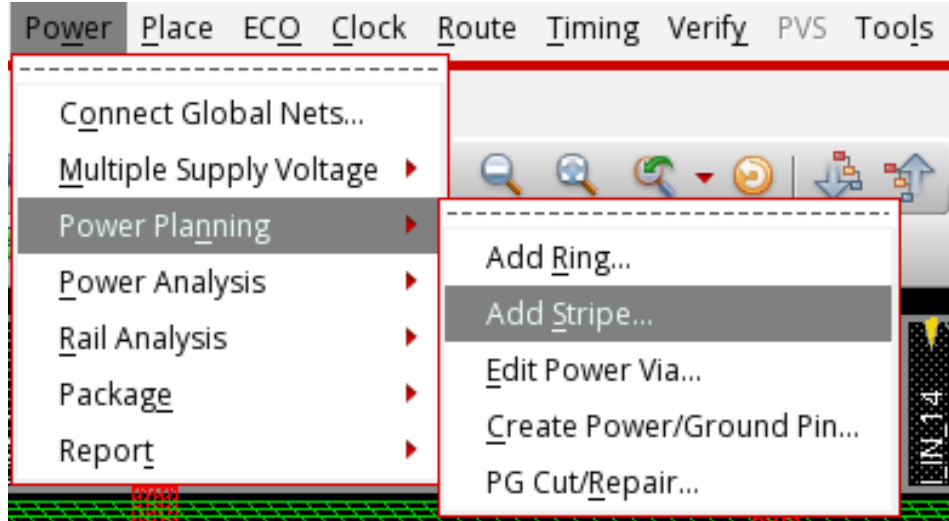
- ☐ ◆ Core Ring
- ☐ ◆ Block Ring

- d. Click **OK** button



## 8. Power Planning (Add Stripes)

1. In innovus menu, open **Power** → **Power Planning** → **Add Stripes...**





2. Create vertical power stripes:
  - a. Set Configuration
    - ☐ Net(s): GND VCC
    - ☐ Layer: **met2(2)**
    - ☐ Directions: **◆ Vertical**
    - ☐ Width: **4**
    - ☐ Click **Update** Button
  - b. Set Pattern
    - ☐ **◆ Set-to-set distance: 100**
  - c. First/Last Stripe
    - ☐ Start from: **◆ Left**
    - ☐ **◆ Relative from core or selected area**
    - ☐ Start: 50
  - d. Click **OK** button

**Add Stripes — ee28**

Basic | Advanced | Via Generation | Mode | Preview

**Set Configuration**

Net(s): GND VCC

Layer: metal2(2) | Directions: ☒ Vertical ☐ Horizontal

Width: 4 | Spacing: 0.28 | **Update**

**Set Pattern**

☒ Set-to-set distance: 100 | ☐ Number of sets: 1 | ☐ Bumps: Over

☐ Over P/G pins | Pin layer: Top pin layer | ☐ Pin Width:

☐ Master name: | ☐ Selected blocks ☒ All blocks

☐ Over Physical Pins | Pin layer: Top pin layer | ☐ Pin Width:

**Stripe Boundary**

☒ Core ring ☐ Pad ring: Outer | ☐ All domains

☐ Design boundary ☒ Create pins ☐ Each selected block/domain/fence

☐ Specify rectangular area

X1: | Y1: | X2: | Y2: |

☐ Specify rectilinear area |


**First/Last Stripe**


Start from: ☒ Left ☐ Right ☐ Top ☐ Bottom

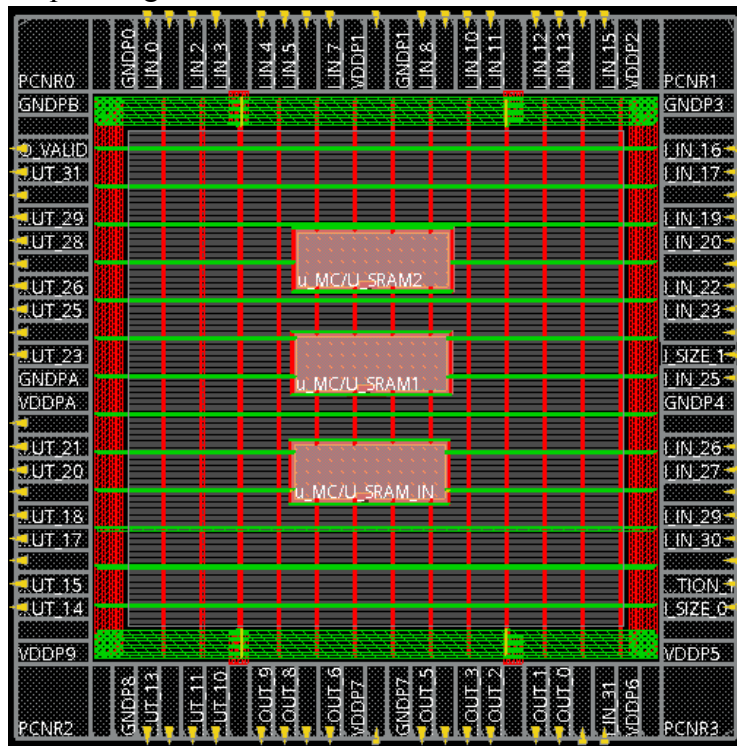
☒ Relative from core or selected area | Start: 50 | Stop:

☐ Absolute | Start: | Stop: |

**OK** | **Apply** | **Defaults** | **Cancel** | **Help**

3. Check if the stripes are correctly created. If not, click **undo** button 
4. Create horizontal stripes:
  - a. Set Configuration
    - ☐ Net(s): GND VCC
    - ☐ Layer: **met3(3)**
    - ☐ Directions: **◆ Horizontal**
    - ☐ Width: **4**
    - ☐ Click **Update** Button

- b. Set Pattern
    - ☐ ◆ Set-to-set distance: **100**
  - c. First/Last Stripe
    - ☐ Start from: ◆ Bottom
    - ☐ ◆ Relative from core or selected area
    - ☐ Start: 50
  - d. Click **OK** button
5. Check if the stripes are correctly created. If not, click **undo** button  and repeat step a~d again.



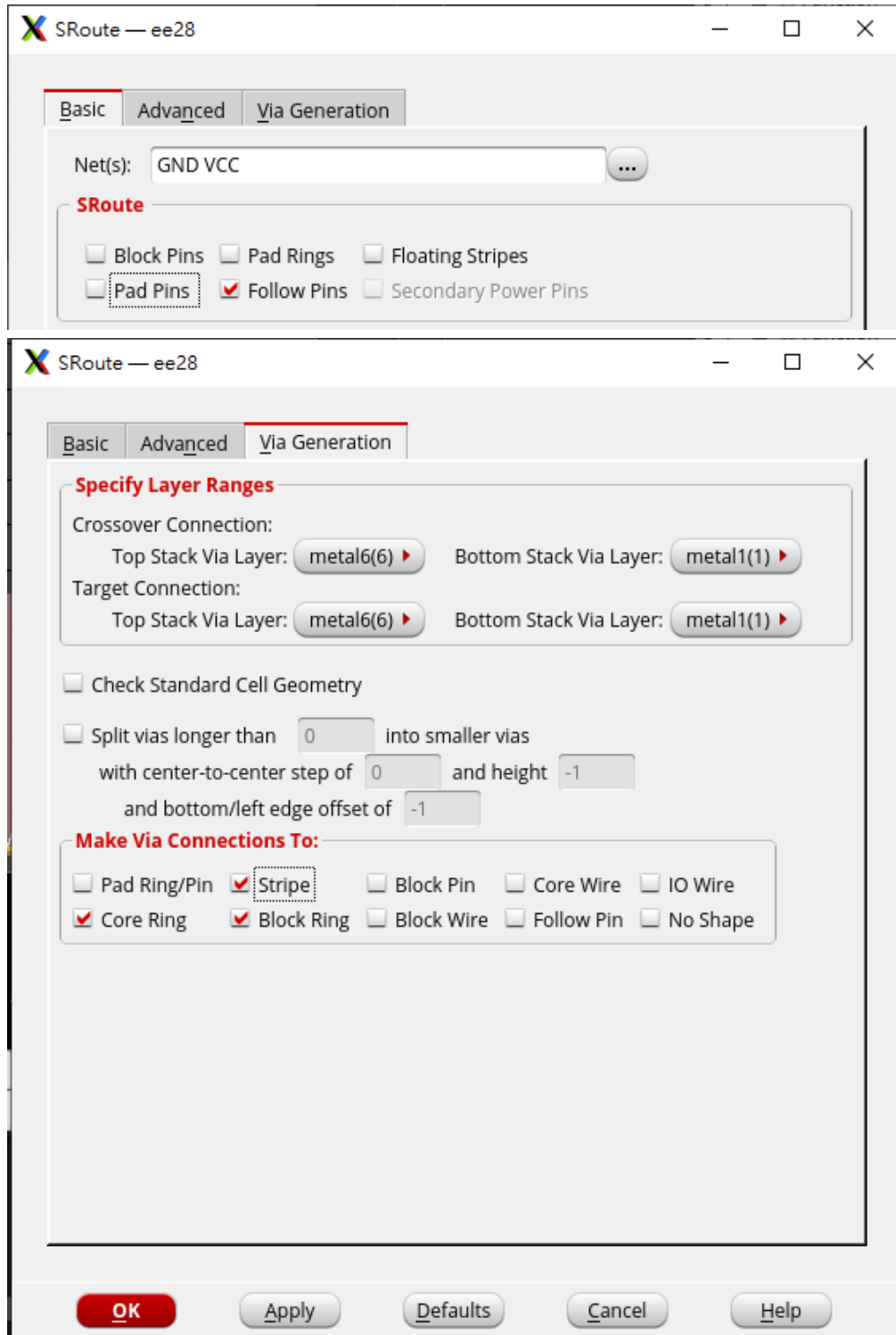
## 9. Connect Standard Cell Power Line

1. Make sure hard macro blockage has been added in floorplan.
2. In innovus menu, open **Route** → **Special Route...**
3. Connect core power: In **Basic** tab,
  - a. Net(s): GND VDD
  - b. Set the following configuration
    - ☐ ◇ Block pins
    - ☐ ◇ Pad rings
    - ☐ ◇ Floating Stripes
    - ☐ ◇ Pad pins
    - ☐ ◆ Follow
- pins **In Via Generation** tab,
  - c. Set the following configuration
    - ☐ ◆ Stripe

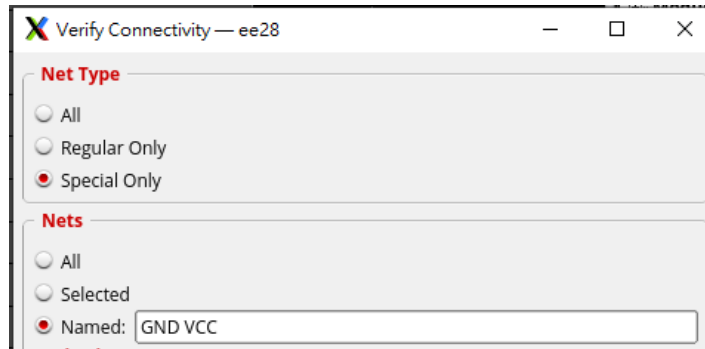
☐ ◆ Core Ring

☐ ◆ Block Ring

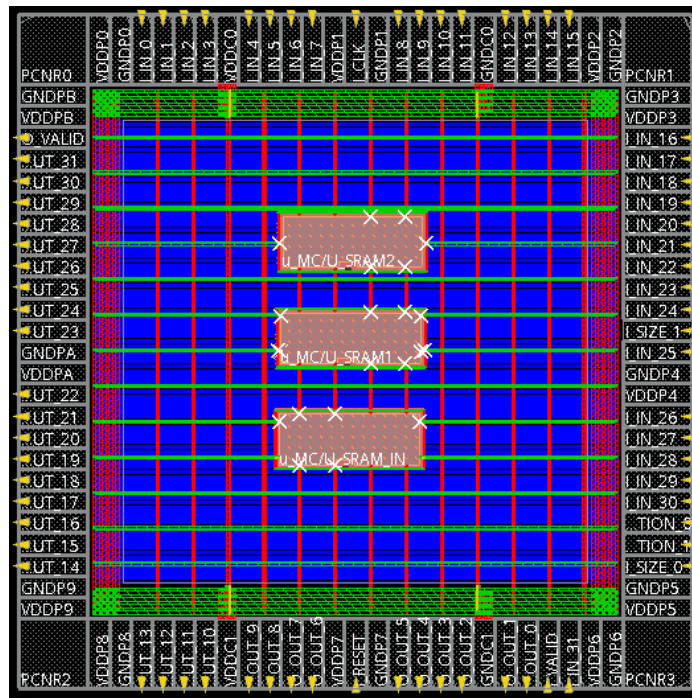
d. Click **OK** button





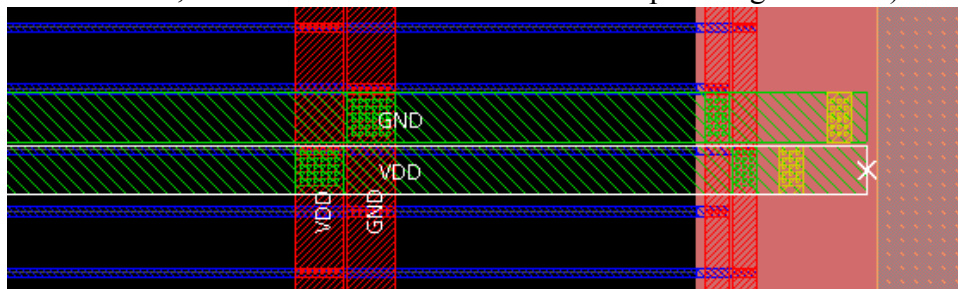


3. If the problem is the dangling wire (floating wire) around memory, like:

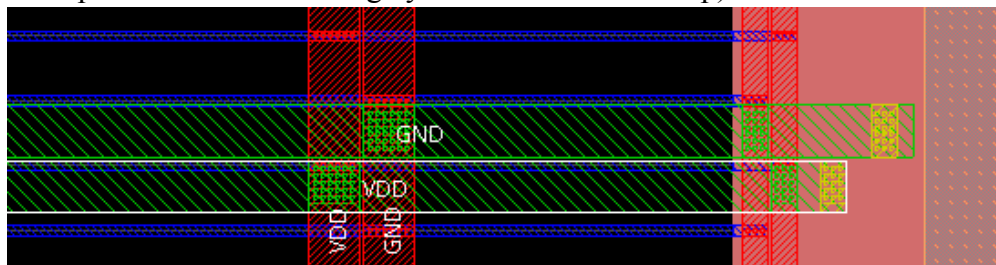


Zoom in and left click to select the highlighted wire:

(In innovus menu, **Tools** → **Violation Browser** can help finding locations)



Press “shift+t”, the dangling wire will be adjusted. (Do not delete those wires or else the power cannot be averagely distributed on the chip)



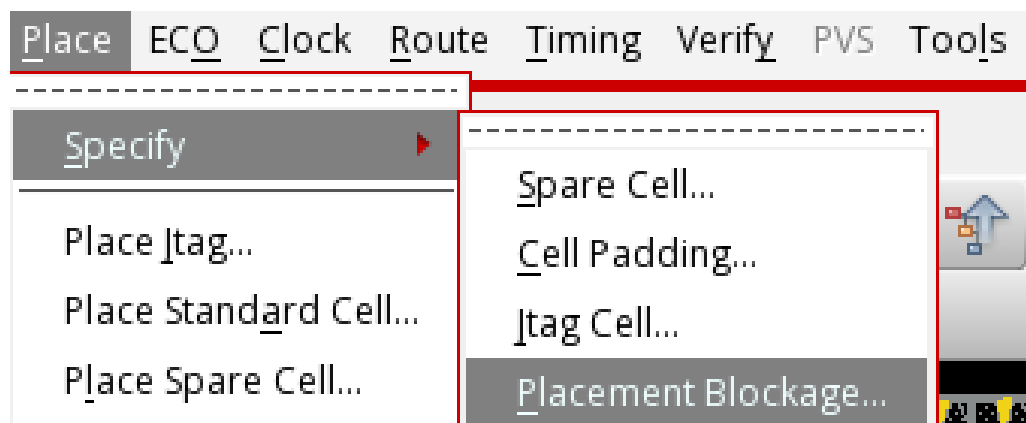
4. After fixing all the dangling wires, run step 2. Again to ensure the connectivity correctness.

```
***** End: VERIFY CONNECTIVITY *****  
Verification Complete : 0 Viols. 0 Wrngs.  
(CPU Time: 0:00:00.0 MEM: 0.000M)
```

5. Save design as CHIP\_powerplan.inn

## 11. Place Standard Cells

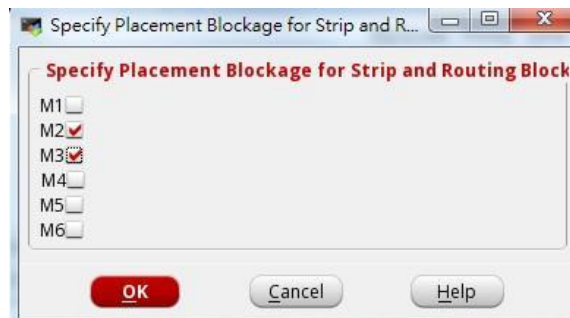
1. In innovus menu, open *Place* → *Specify* → *Placement Blockage*



2. Specify placement blockage for stripes
  - a. Specify placement blockage under metal2 and metal3

- ◇ M1
- ◆ M2
- ◆ M3
- ◇ M4
- ◇ M5
- ◇ M6

- b. Click **OK** button



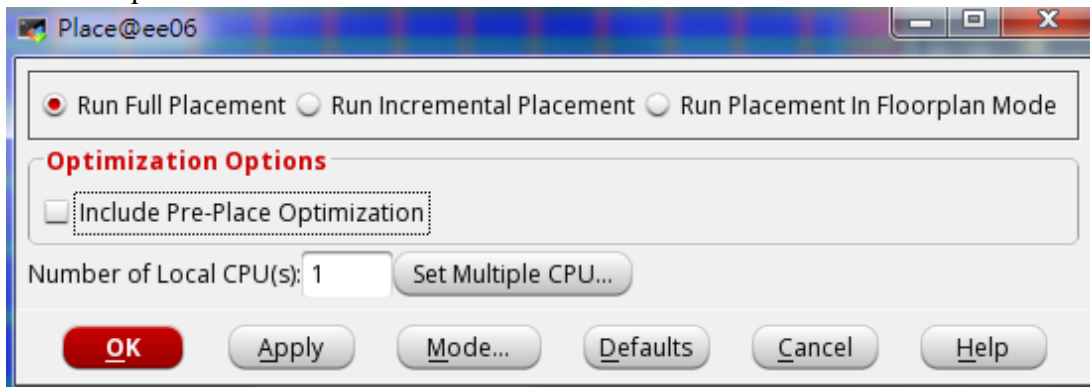
3. In innovus menu, open **Place** → **Place Standard Cells...**

- ☐ ◆ Run Full

Placement Optimization Options

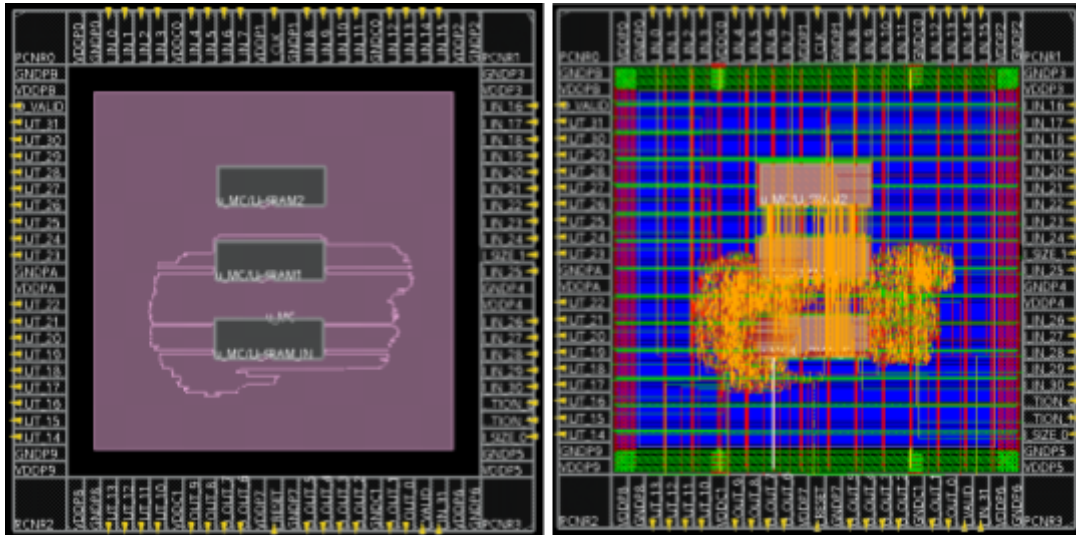
- ☐ ◇ Include Pre-Place

Optimization Click **OK** button



The following figures show the results of Amoeba view and Physical view



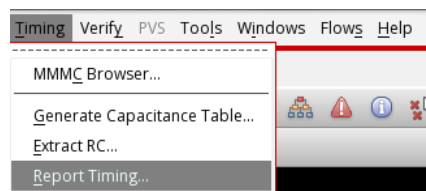


4. Save design as CHIP\_placement.inn

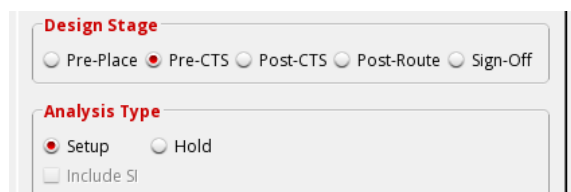
## 12. In-Place Optimization (IPO)

### - Before Clock Tree Synthesis

1. In innovus menu, open **Timing** → **Report Timing...**



2. Perform trial route to model the interconnection RC effects
  - a. Design Stage ♦ pre-CTS
  - b. Analysis Type ♦ Setup
  - c. Click **OK** button



3. See timing reports in **timingReports/** directory, **CHIP\_preCTS.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_preCTS\_all.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.
4. If the timing slack is negative, or there are DRVs, open **ECO** → **Optimize Design...** in innovus menu
5. Perform pre-CTS IPO
  - a. Design Stage ♦ Pre-CTS
  - b. Optimization Type
    - ☐ ♦ Setup
    - ☐ ♦ Design Rule Violations
    - ☒ ♦ Max Cap



- ◆ Max Tran
- ◆ Max Fanout

c. Click **OK** button



#### 6. Save design as CHIP\_preCTS.inn

In preCTS timing report, the max cap violation of DRV may be caused by reset\_n or clk, which connects to a lot of registers. After ECO, the rst\_n DRV may be fixed by inserting buffers. Whereas the clk may not since the clk buffers will be inserted in the next step: Clock Tree Synthesis (CTS) stage. The remaining clk DRV will be shown like:

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	0 (0)
max_fanout	0 (0)	0	0 (0)
max_length	0 (0)	0	0 (0)

and can be view in the file *timingReports/CHIP\_preCTS.tran.gz*. This DRV should be fixed after synthesizing the clock tree, thus don't worry in this stage. You only have to worry if this DRV cannot be fixed after CTS.

You will also see **Density** and **Routing Overflow** terms at the bottom of the timing report.

```
Density: 8.473%
Routing Overflow: 0.00% H and 0.00% V
```

The lower the density is, the lower the standard cells (excluding Hardmacros) are utilizing the core, which means it provides larger flexibility in further routing and optimization steps.

After the placement and in the preCTS stages, innovus performs **trial route** to give the rough delay calculation of path between every registers / input output ports so the preCTS can be done. Similar to the density which is the placement utilization, the routing overflow represents the wiring congestion level. The higher the routing overflow is, the harder of the coming CTS and detail routing (**nanoroute**). Usually when Horizontal and Vertical routing overflow are both < 0.5% ~ 1.0%, the further routing problem will be small. The routing overflow term will disappear after **nanorouting** since the **detail route** will be given to replace the **trial route**.

## 13. Clock Tree Synthesis (CTS)

➤ source ./cmd/ccopt.cmd

## 14. In-Place Optimization (IPO)

### - After Clock Tree Synthesis

1. In innovus menu, open **Timing** → **Report Timing...**
2. Perform trial route to model the interconnection RC effects
  - a. Design Stage ◆ Post-CTS
  - b. Analysis Type ◆ Setup
  - c. Click **OK** button
3. See timing reports in **timingReports/** directory, **CHIP\_postCTS.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_postCTS\_all.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.
4. If the timing slack is negative, or there are DRVs, open **ECO** → **Optimize**

**Design...** in innovus menu

5. Perform post-CTS IPO
  - a. Design Stage ◆ Post-CTS
  - b. Optimization Type
    - ☐ ◆ Setup
    - ☐ ◆ Design Rule Violations
      - ◆ Max Cap
      - ◆ Max Tran
      - ◆ Max Fanout
  - c. Click **OK** button

**From post CTS steps, hold time checking is also required**

6. In innovus menu, open **Timing** → **Report Timing...**
7. Perform trial route to model the interconnection RC effects
  - a. Design Stage ◆ Post-CTS
  - b. Analysis Type ◆ Hold
  - c. Click **OK** button
8. See timing reports in **timingReports/** directory, **CHIP\_postCTS\_hold.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_postCTS\_all\_hold.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.
9. If the timing slack is negative, or there are DRVs, open **ECO** → **Optimize Design...** in innovus menu
10. Perform Post-CTS IPO
  - a. Design Stage ◆ Post-CTS
  - b. Optimization Type
    - ☐ ◆ Hold
    - ☐ ◆ Design Rule Violations
      - ◆ Max Cap
      - ◆ Max Tran

◆ Max Fanout

c. Click **OK** button

11. Save design as CHIP\_postCTS.inn

## 15. Add PAD Filler

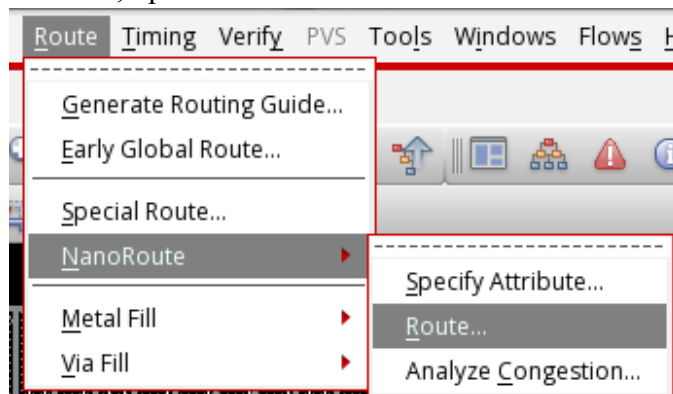
1. In innovus command prompt, execute the following commands:

☐ **source ./cmd/addIOFiller.cmd**

**-fillAnyGap** PAD filler must be added before detail route, or there may have some DRC/LVS violations after PAD filler insertion

## 16. SI-Prevention Detail Route (NanoRoute)

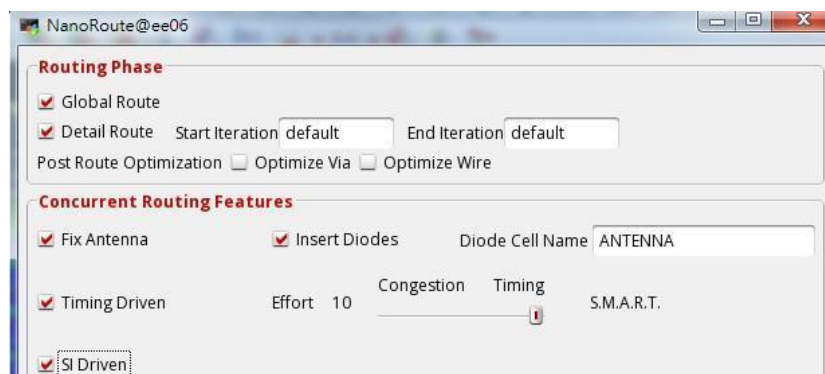
1. In innovus menu, open **Route** → **NanoRoute** → **Route**



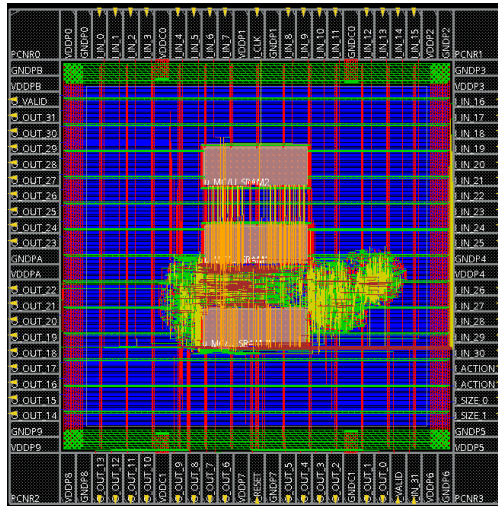
2. Nanoroute can prevent cross talk effects and fix antenna rule violations, also it routes design to meet timing constraints.

a. Configure routing features

- ◆ Fix Antenna
- ◆ Insert Diodes      Diode Cell Name: ANTENNA
- ◆ Timing Driven      Effort: 10
- ◆ SI Driven

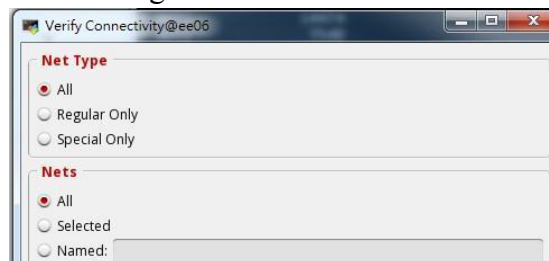


b. Click **OK** button



3. In innovus menu, open **Verify** → **Connectivity**

- ☐ Net Type: ♦ All
- ☐ Nets: ♦ All
- ☐ Click **OK** button
- ☐ Check routing for LVS error



4. In innovus menu, open **Verify** → **DRC**

- ☐ Click **OK** button
- ☐ Check routing for DRC error

Verification Complete : 0 Viols.

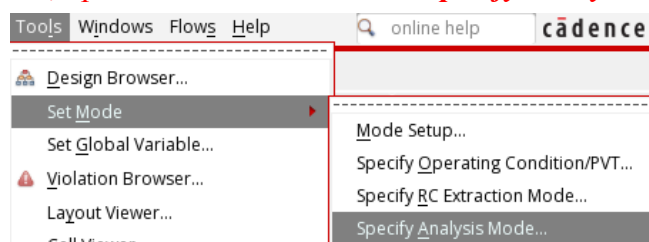
\*\*\* End Verify DRC (CPU: 0:00:00.3 ELAPSED TIME: 0.00 MEM: 1.0M) \*\*\*

5. Save design as CHIP\_nanoRoute.inn

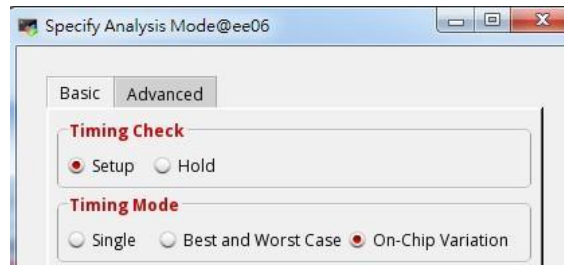
## 17. In-Place Optimization (consider crosstalk effects)

- After Detail Route

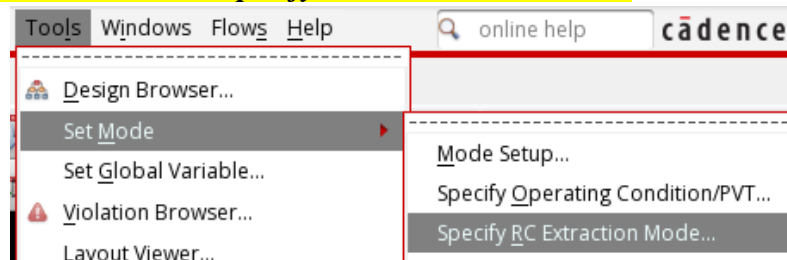
1. In innovus menu, open **Tools** → **Set Mode** → **Specify Analysis Mode...**



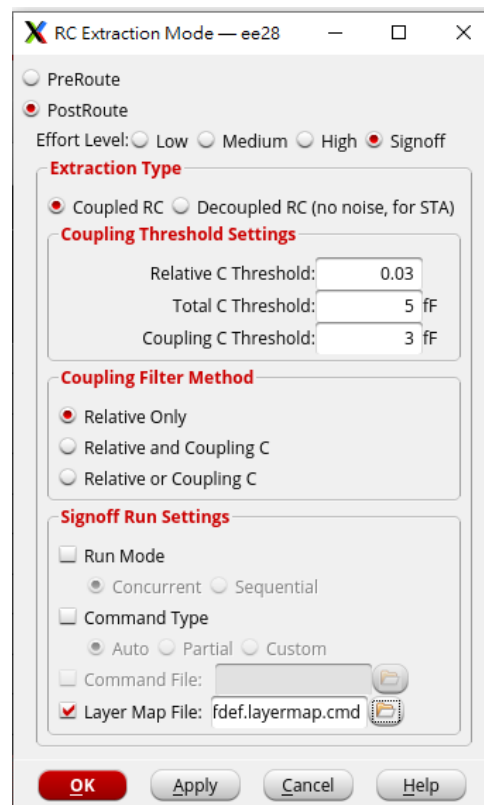
- a. Timing Mode: ♦ On-Chip Variation
- b. Click **OK** button



2. In innovus menu, open **Tools → Set Mode → Specify RC Extraction Mode...**



- a. **◆ PostRoute**
- b. **Effort Level: ◆ Signoff**
- c. **Extraction Type: ◆ Coupled RC**
- d. **Signoff Run Settings:**
- ◆ **Layer Map File: lefdef.layermap.cmd**
- e. **Click OK button**



3. Setting rc and si
  - ☐ innovus > set\_db extract\_rc\_engine post\_route
  - ☐ innovus > set\_db extract\_rc\_effort\_level high
  - ☐ innovus > set\_db delaycal\_enable\_si true
4. In innovus menu, open **Timing → Report Timing...**
5. Perform trial route to model the interconnection RC effects
  - a. Design Stage **◆ Post-Route**
  - b. Analysis Type **◆ Setup**

- c. Click **OK** button
- 6. See timing reports in **timingReports/** directory, **CHIP\_postRoute.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_postRoute\_all.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.

**\*\* You can only run postRoute and signoff timing analysis on ee servers \*\***

- 7. If the timing slack is negative, or there are DRVs, open **ECO→Optimize Design...** in innovus menu
- 8. Perform post-Route IPO
  - a. Design Stage ◆ post-Route
  - b. Optimization Type
    - ☐ ◆ Setup
    - ☐ ◆ Design Rule Violations
      - ◆ Max Cap
      - ◆ Max Tran
      - ◆ Max Fanout
  - c. Click **OK** button
- 9. In innovus menu, open **Timing → Report Timing...**
- 10. Perform trial route to model the interconnection RC effects
  - a. Design Stage ◆ Post-Route
  - b. Analysis Type ◆ Hold
  - c. Click **OK** button
- 11. See timing reports in **timingReports/** directory, **CHIP\_postRoute\_hold.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_postRoute\_all\_hold.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.
- 12. If the timing slack is negative, or there are DRVs, open **ECO→Optimize Design...** in innovus menu
- 13. Perform post-Route IPO
  - a. Design Stage ◆ post-Route
  - b. Optimization Type
    - ☐ ◆ Hold
    - ☐ ◆ Design Rule Violations
      - ◆ Max Cap
      - ◆ Max Tran
      - ◆ Max Fanout
  - c. Click **OK** button
- 14. Save design as **CHIP\_postRoute.inn**

## 18. Timing Analysis (Signoff)

### - Optional in this Practice

Signoff timing analysis is similar to postRoute timing analysis. In addition to invoke Quantus QRC Extraction tool to calculate the RC delay as postRoute, it also consider the SI affect with the signoff RC.

Currently, the delay considering the SI affect with signoff RC can be only

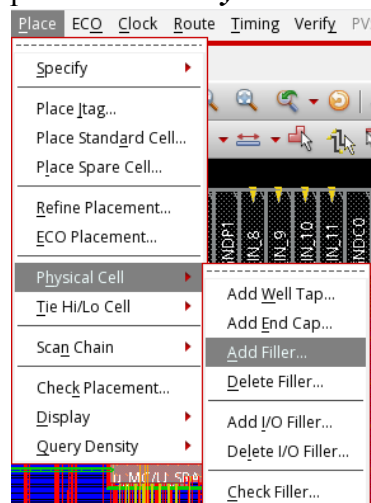
used for timing analysis, but cannot be used for timing optimization. If you want to perform timing analysis or timing optimization for postRoute again after signoff timing analysis, you need to reset the SI mode by the command:  
 innovus > **setDelayCalMode -siMode signoff**

1. In innovus menu, open **Timing → Report Timing...**
2. Perform trial route to model the interconnection RC effects
  - a. **Design Stage ♦ Sign-Off**
  - b. Analysis Type ♦ Setup
  - c. Click **OK** button
3. See timing reports in **timingReports/** directory, **CHIP\_signOff.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_signOff\_all.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.
4. In innovus menu, open **Timing → Report Timing...**
5. Perform trial route to model the interconnection RC effects
  - a. **Design Stage ♦ Sign-Off**
  - b. Analysis Type ♦ Hold
  - c. Click **OK** button
6. See timing reports in **timingReports/** directory, **CHIP\_signOff\_hold.slk** shows the timing analysis results. All slack values must be positive value in this file. Moreover, for detail path report, see **CHIP\_signOff\_all\_hold.tarpt**. DRVs report files: **\*.cap**, **\*.fanout**, and **\*.tran**.

You can perform the above steps to see if there is any difference between the results between the signoff timing analysis and the postRoute timing analysis. In most experiences and in this practice, signoff timing analysis result is very closed the postRoute timing analysis. Thus in this practice you can choose to perform this step or not.

## 19. Add CORE Filler Cells

1. In innovus menu, open **Place → Physical Cell → Add Filler**



2. Add core filler to improve electric effects of NWELL and PWELL:
  - a. Click **Select** button next to Cell Name(s)
  - b. Select core filler cells
  - c. Click **Add** button
  - d. Click **Close** button
  - e. Click **OK** button





## 20. Stream Out and Write Netlist

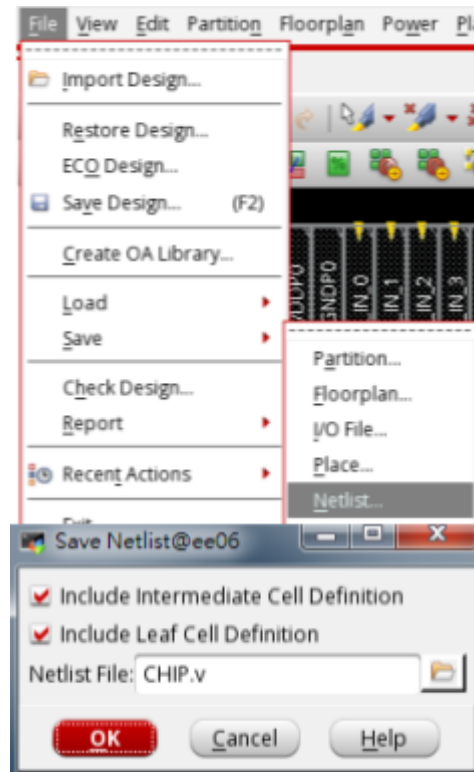
1. Save design as **CHIP.inn**
2. Write CHIP.sdf
  - a. In innovus menu, open **Timing** → **Write SDF**
  - b. Delay Calculation Option
    - ☐ ☒ **Ideal Clock**

**Ideal Clock should be disabled**
  - c. Click **OK** button



(or you can use the command "write\_sdf CHIP.sdf")

3. Save design netlist CHIP.v for post-layout simulation:
  - a. In innovus menu, open **File** → **Save** → **Netlist...**
    - ☐ ☒ Include Intermediate Cell Definition
    - ☐ ☒ Include Leaf Cell Definition
    - ☐ Netlist File: **CHIP.v**
    - ☐ Click **OK** button



## 21. Post-Layout Gate-Level Simulation

1. Change to directory **06\_POST**
2. Perform Post-Layout Gate-level simulation of CHIP.v  
 % ./01\_run\_vcs\_post The latency should be the same as gate level simulation

### Appendix: IO Pad assignment

CHIP.io:

```
Version: 1
Spacing: 16
Orient: R90
pad: cornerUL          NW      CORNERC
Orient: R0
pad: cornerUR          NE      CORNERC
Orient: R270
pad: cornerLR          SE      CORNERC
Orient: R180
pad: cornerLL          SW      CORNERC
pad: opad_QULLR0        W
pad: io_vss1            W      GNDIOC
.
.
pad: opad_QULLR5        W
pad: io_vss2            W      GNDIOC
pad: io_vdd3            N      VCC3IOC
pad: ipad_CODEWORD      N
.
.
.
pad: io_vdd4            N      VCC3IOC
pad: io_vss4            N      GNDIOC
pad: opad_QULLR6        N
```

**You can execute `>> source ./cmd/run_apr.cmd` in innovus interface to do **Step 2.** and then you can directly go to Floorplan Step!**