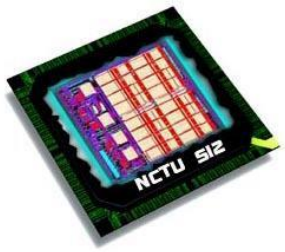
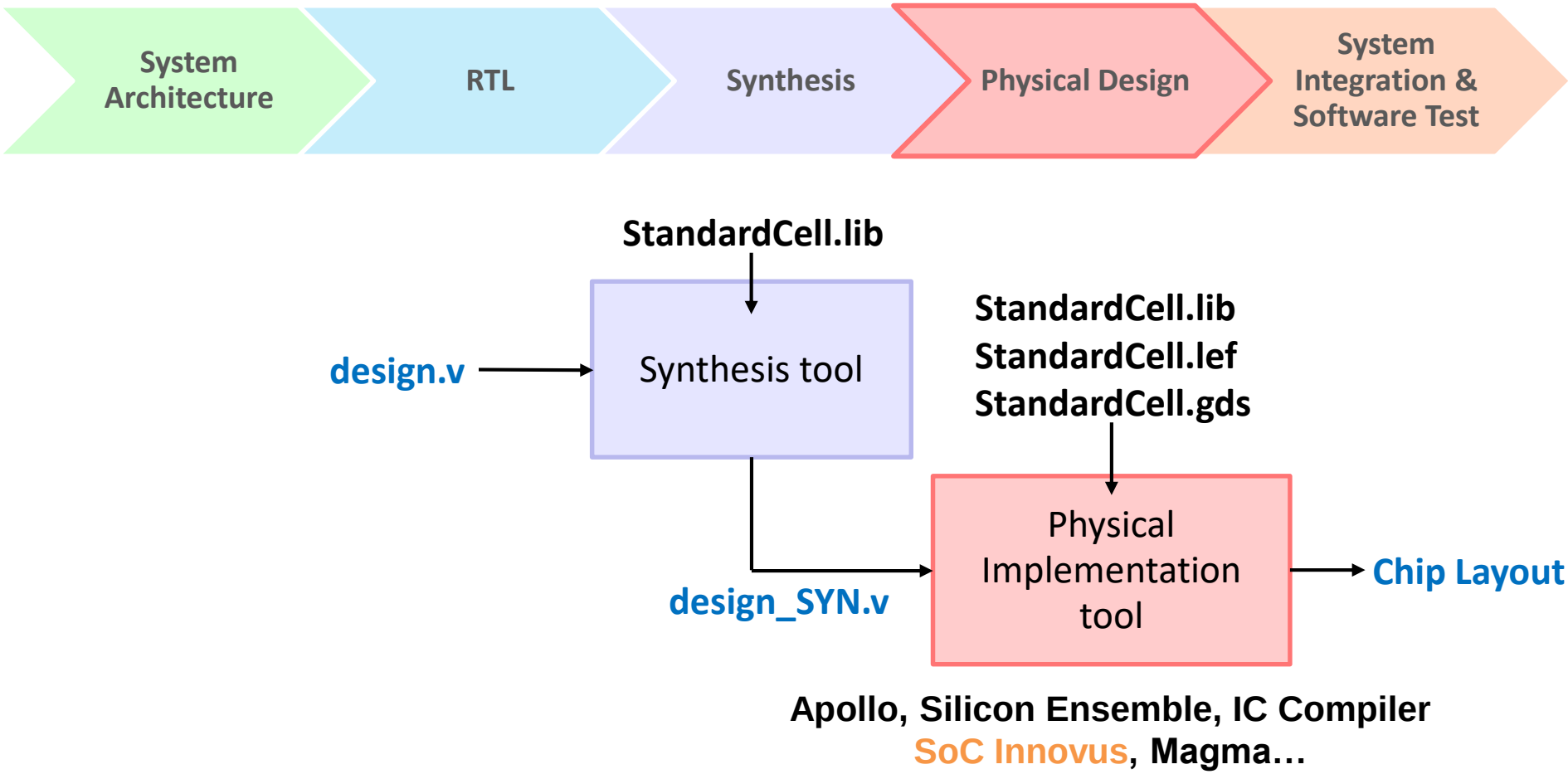
Cell-Based APR Design Flow

NCTU-EE IC Lab Spring-2018



Recall: Design Flow

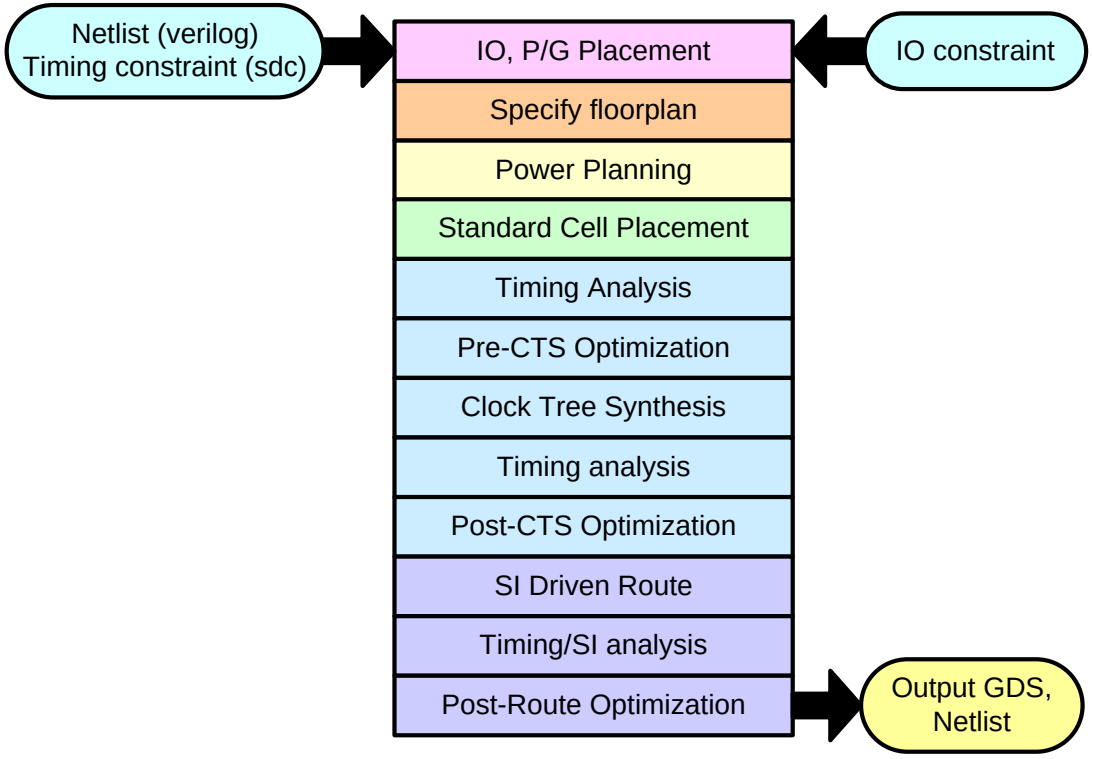
✓ **Cell-based Design Flow**

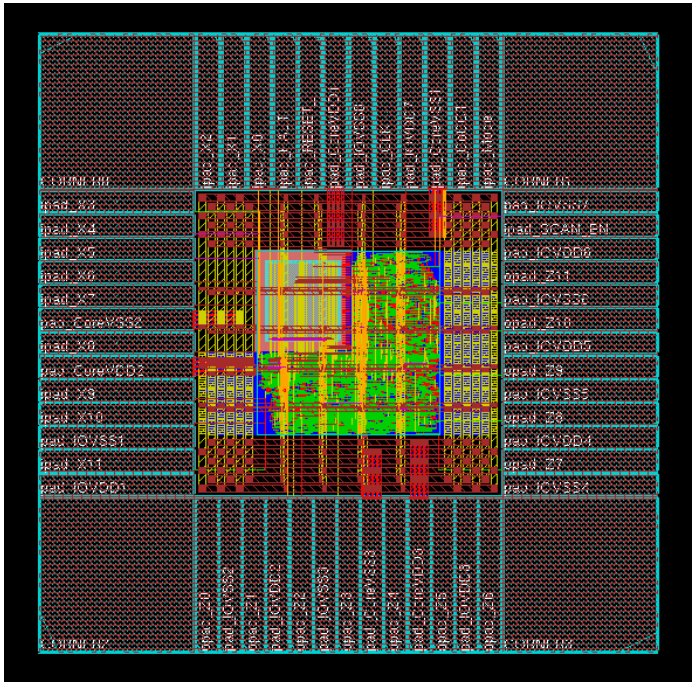


APR Design Flow

import
floorplan
powerplan
placement
CTS
routing

✓ SoC Innovus Design Flow

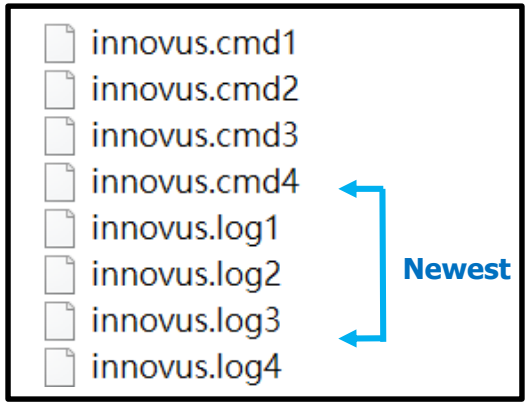




Data Preparation

Tips for SoC Innovus

- ✓ **Do not use background execution !!**
- ✓ **The executed commands are saved in innovus.cmd#**
- ✓ **The executed log files are saved in innovus.log#**
- ✓ **Save design after finishing each step !**
- ✓ **Restart from each step by freeing/reloading design**
  - Save Design
    - innovus > saveDesign xxx.inn
  - Free Design
    - innovus > freeDesign
  - Restore Design
    - innovus > restoreDesign xxx.inn.dat CHIP
- ✓ **Online document reference**
  - **unix%** /usr/EDI/tools/bin/cdnshelp &



0. Data Preparation

- ✓ **Library files**
  - Timing libraries (LIB)
    - NangateOpenCellLibrary_fast.lib
    - NangateOpenCellLibrary_slow.lib
    - tpz_fast.lib
    - tpz_slow.lib
    - HardMacro.lib
  - Physical libraries (LEF)
    - NangateOpenCellLibrary.macro.lef
    - NangateOpenCellLibrary.tech.lef
    - tpz.lef
    - HardMacro.vclef
  - RC extraction
    - best.captbl
    - worst.captbl
    - best.tch
    - worst.tch
  - CeltIC libraries
    - fast.cdb
    - slow.cdb
  - GDSII layout
    - NangateOpenCellLibrary.macro.gds
    - tpz.gds
    - HardMacro.gds
- ✓ **Timing constraint files (User Data)**
  - Generate timing constraint files from Design Complier
    - dc_shell-t> write_sdc design_SYN.sdc
- ✓ **Design netlist files (User Data)**
  - Synthesized design netlist
    - designName_SYN.v
  - Chip design netlist
    (combine core_syn.v&chip_shell.v)
    - CHIP_SYN.v
  - Uniquified design netlist
    (from chip_syn.v)
    - CHIP_unique.v
  - IO pad location file
    - CHIP.ioc

0. Data Preparation - Libraries

- ✓ **LEF Library : physical process and cell information**
  - Layer definition, width and spacing, default direction
  - Cell size, pin position, pin metal layer
- ✓ **LIB Library : timing information**
  - Operating condition, pin capacitance
  - path delay, transition delay, timing constraints (ex: setup/hold)
- ✓ **CeltIC Library (*.cdb)**
  - cdB model (noise model for each cell used in SI optimization phase)
- ✓ **RC Extraction (*.captbl)**
  - Capacitance table
  - RC extraction for further power analysis, simulation etc

0. Data Preparation - Design Netlist

- ✓ **Chip design netlist: CHIP_SYN.v**
  - Add pad cells to synthesized design netlist
  - Combine design_SYN.v and CHIP_SHELL.v
    - `cat CHIP_SHELL .v CORE_SYN.v > CHIP_SYN.v`
  - CHIP_SHELL.v contains I/O pads information
- ✓ **Uniquified design netlist : CHIP_unique.v**
  - Uniquifies the design netlist (CHIP_SYN.v)
    and writes the uniquified design to a netlist file (CHIP_unique.v)
    - `uniquifyNetlist -top CHIP CHIP_unique.v CHIP_SYN.v`
  - The chip design netlist must be unique for running Clock Tree Synthesis (CTS), Scan Reorder, and In-Placement Optimization(IPO)
  - When you uniquify a netlist, you create a unique module definition for every module in the netlist.

0. Data Preparation – I/O File

- **IO pad location file: CHIP.ioc**
  - Edit IO pad location file
    - CHIP.ioc
  - Syntax
    - `Pad: pad_instance_name direction [pad_type]`
  - Categories

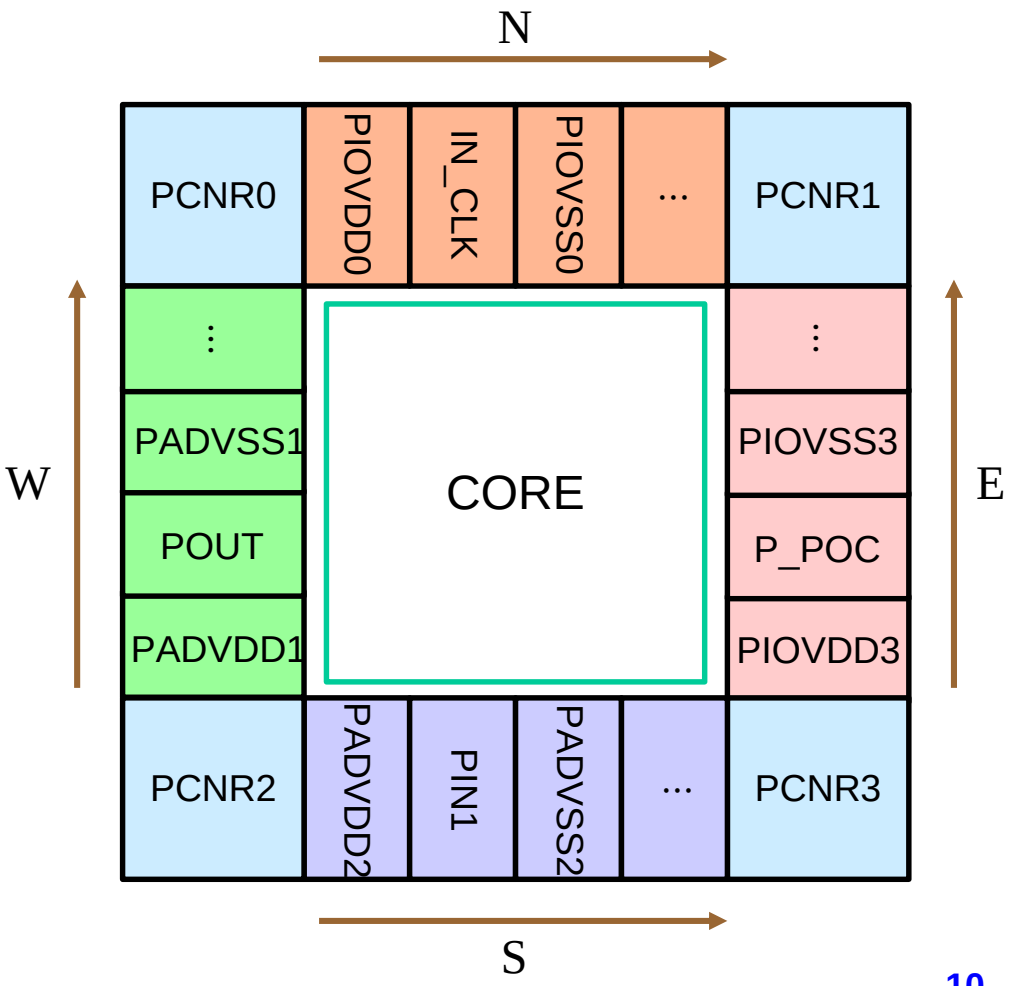
Field	Usage	
pad_instance_name	Corresponding instance name of pad in the chip netlist	
direction	Specify pad location	S (southern pads)
		E (eastern pads)
		W (western pads)
		N (northern pads)
pad_type	Specify pad type (corner pad, IO power pad, core power pad, and pad filler)	PADCORNER (corner pads)
		PADIOVDD (IO power pads)
		PADIOVSS (IO power pads)
		PADVDD (core power pads)
		PADVSS (core power pads)
		PADFILLER (pad fillers)
		PADIOVDDPOC

0. Data Preparation – I/O File

✓ **CHIP.ioc**

Note:
1. It's recommended to put at least one power pad pairs on each direction
2. *PADIOVDDPOC* must be included once in each power domain

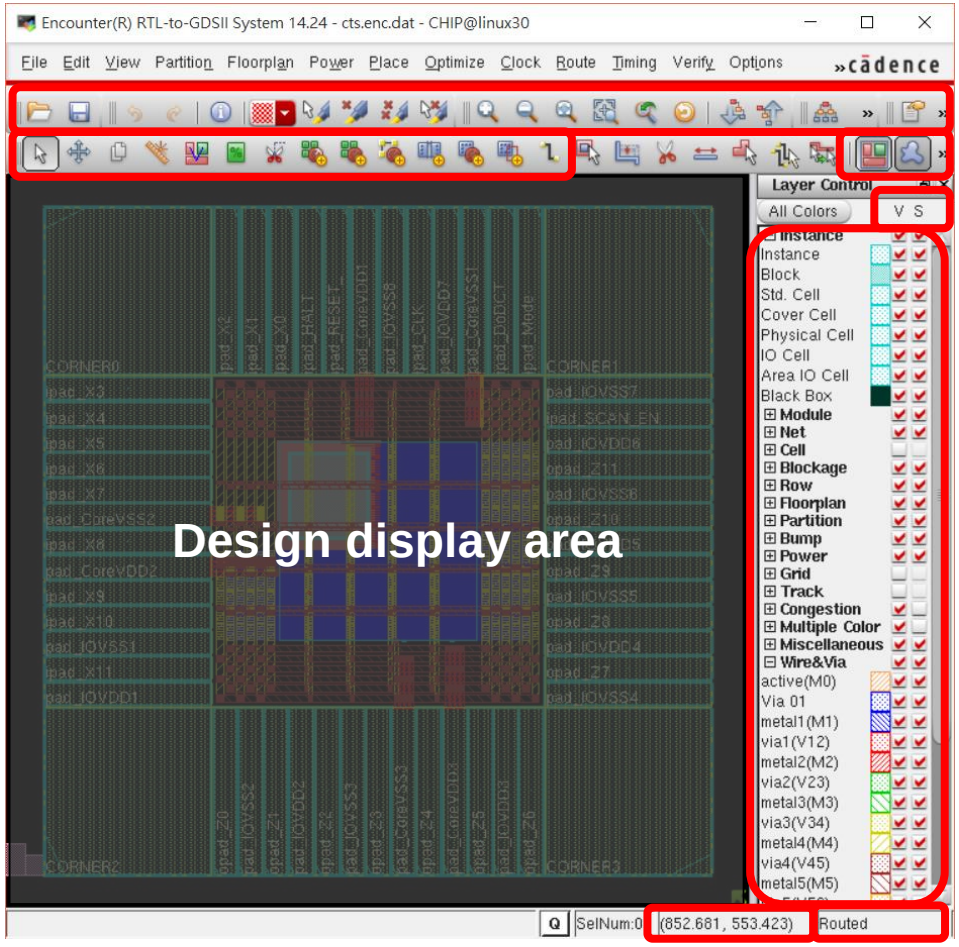
```
      pad
Pad:  instance     dir. [pad_type]
      name
Pad:  PCNR0        NW   PADCORNER
Pad:  PIOVDD0      N    PADIOVDD
Pad:  IN_CLK       N
Pad:  PIOVSS0      N    PADIOVSS
…
Pad:  PCNR1        NE   PADCORNER
Pad:  PIOVDD3      E    PADIOVDD
Pad:  P_POC        E    PADIOVDDPOC
Pad:  PIOVSS3      E    PADIOVSS
…
Pad:  PCNR2        SW   PADCORNER
Pad:  PADVDD2      S    PADVDD
Pad:  PIN1         S
Pad:  PADVSS2      S    PADVSS
…
Pad:  PCNR3        SE   PADCORNER
Pad:  PADVDD1      W    PADVDD
Pad:  POUT         W
Pad:  PADVSS1      W    PADVSS
…
```



Import Design

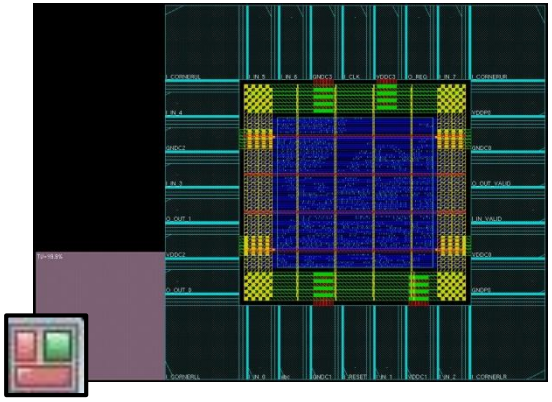
1. Invoke SoC Innovus

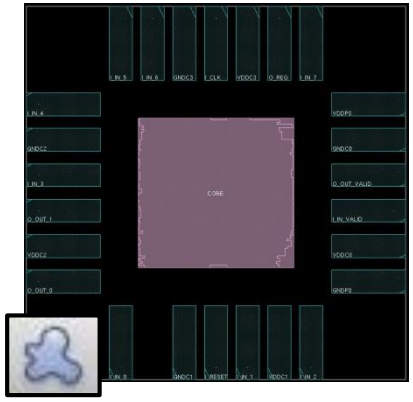
- ✓ **Command to start an Encounter session**
  - unix% **innovus**

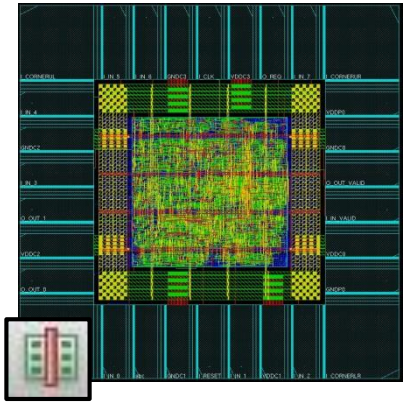


1. Invoke SoC Innovus

- ✓ **Design Views**
  - –  Floorplan View: displays the hierarchical module and block guides, connection flight lines and floorplan objects
  - –  Amoeba View: displays the outline of modules after placement
  - –  Physical View: displays the detailed placements of cells, blocks.

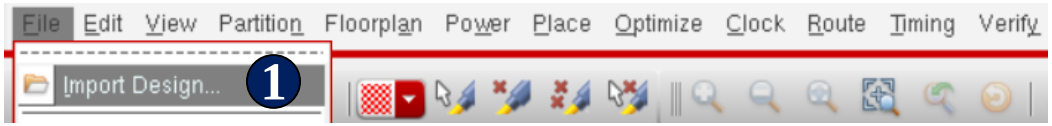


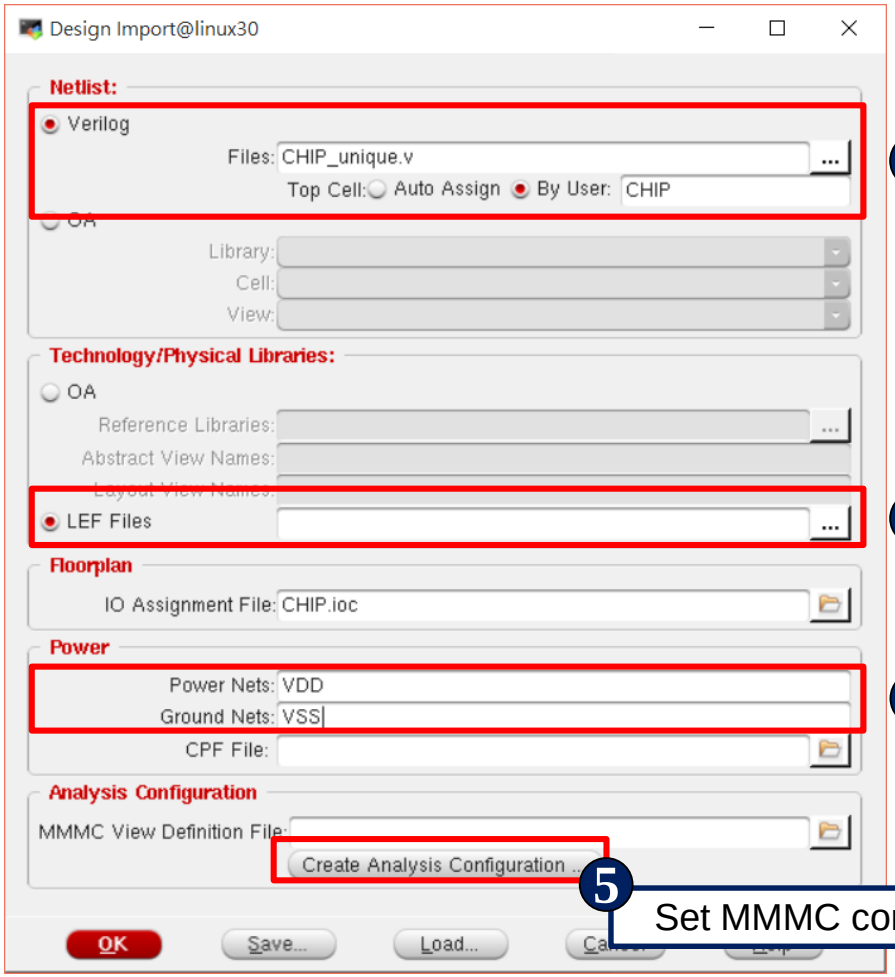




2. Import Design

✓ **Import design files into Encounter environment**





2 Files: CHIP_unique.v
Top Cell: ◆ By user: CHIP

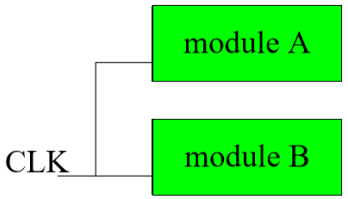
3 Specify LEF files

4 Fill in power/GND nets

5 Set MMMC configuration

2. Import Design - MMMC

- ✓ **Multi-mode multi-corner (MMMC) timing analysis and optimization**
  - Multi-mode



Operation Mode1： moduleA runs on 100MHz
moduleB not use
Operation Mode2： moduleA runs on 50MHz
moduleB runs on 50MHz

  - Multi-corner

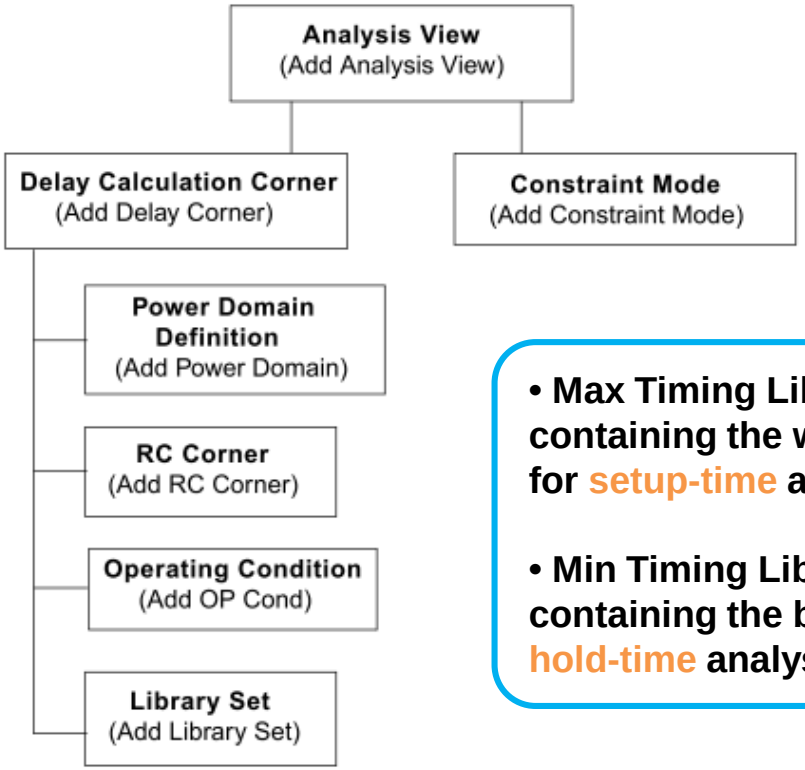
Example

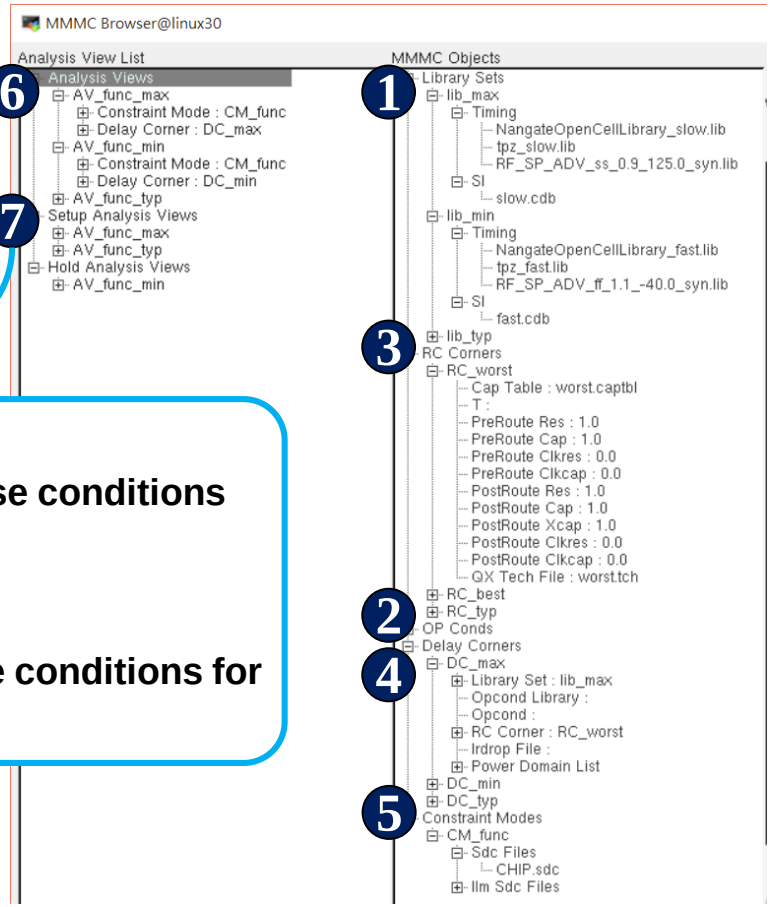
A design is required to meet 3 operating corners:

- Corner 1 - 1.1V, 0°C
- Corner 2 – 0.9V, 100°C
- Corner 3 – 1.1V, 100°C

2. Import Design - MMMC

- ✓ **Multi-mode multi-corner (MMMC) timing analysis and optimization**
  - In MMMC environment you must specify the lower-level information first

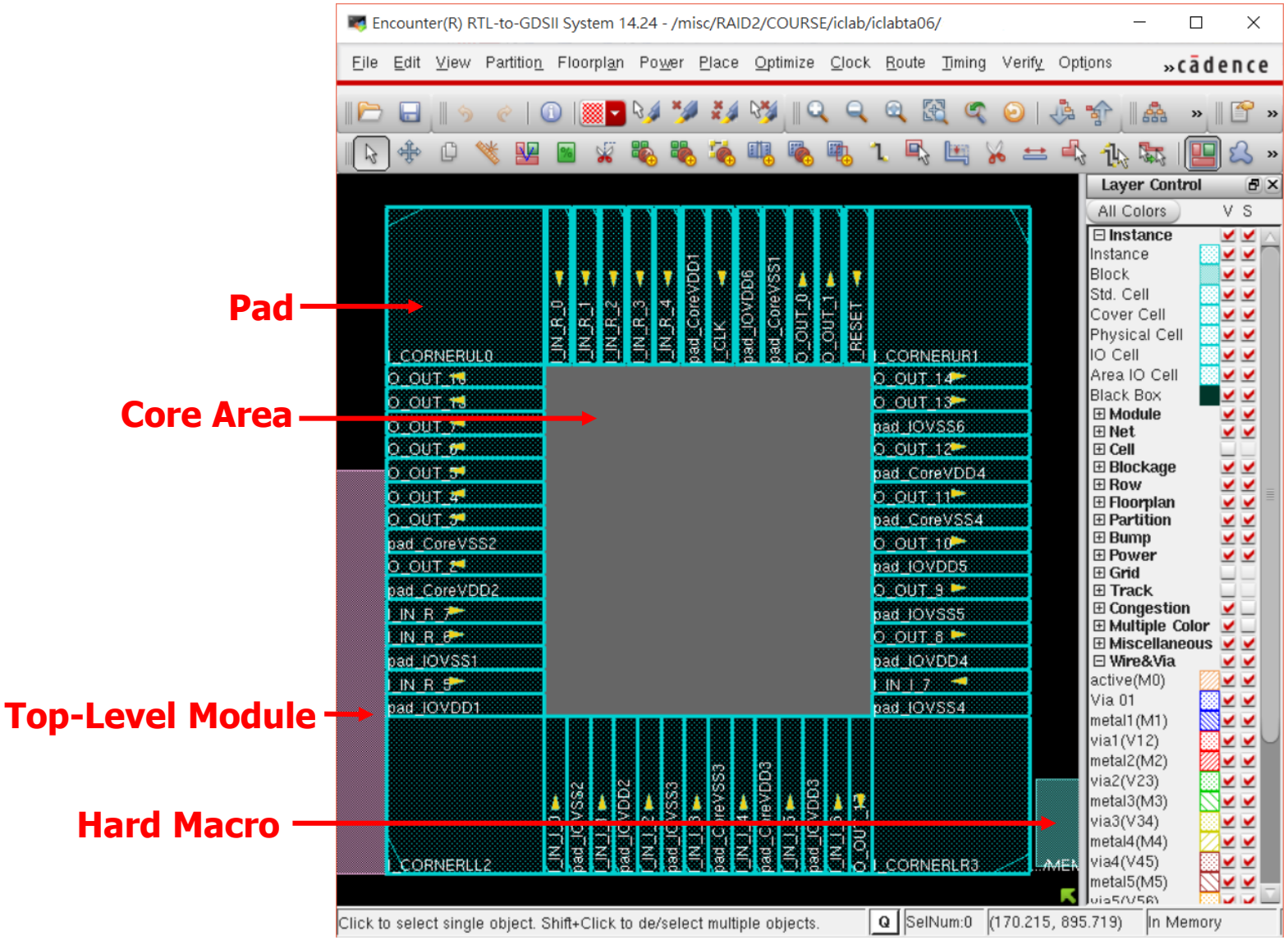




- **Max Timing Libraries: containing the worst-case conditions for setup-time analysis**
- **Min Timing Libraries: containing the best-case conditions for hold-time analysis**

Note:
You can save the settings to ***CHIP_mmmc.view*** by clicking **save** bottom, then you can load it by using **load** bottom next time without configuration set up again

2. Import Design - Result



Floorplan

3. Floorplan – Intro.

✓ Core size determination

- When calculating core size of standard cells, the core utilization must be decided first. Usually the core utilization is higher than **85%**
- The core size can be calculated as follows

$$\text{Core Size of Standard Cell} = \frac{\text{macro} + \text{standard cell area}}{\text{core utilization}}$$

- The recommended core shape is a **square**, i.e. Core Aspect Ratio = 1. Hence the width and height can be calculated as

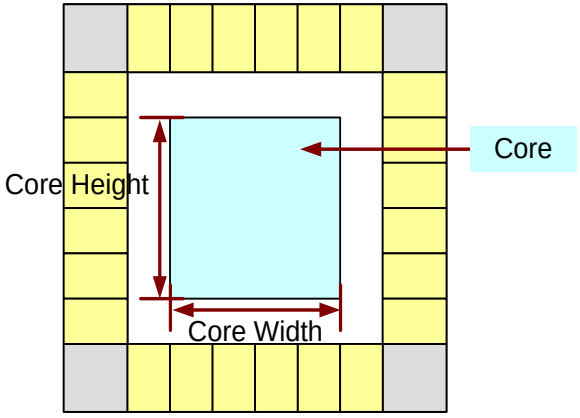
$$\text{Width} = \text{Height} = \sqrt{\text{Core Size of Standard Cell}}$$

  - Note: because stripes and macros will be added, width and height are usually set larger than the value calculated above

- E.g.
  - No mStandard cell area = 2,000,000
  - Core utilization demanded = 85%
  - acros

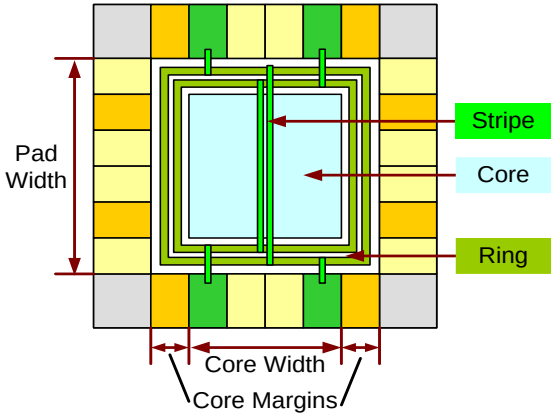
$$\text{Core Size of Standard Cell} = \frac{2{,}000{,}000}{0.85} = 2{,}352{,}941$$

$$\text{Width} = \text{Height} = \sqrt{2{,}352{,}941} = 1534$$

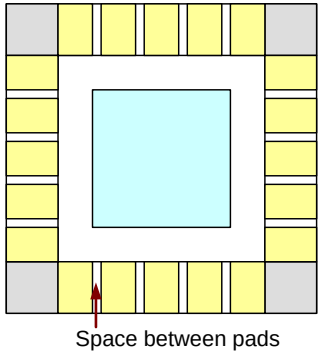


3. Floorplan – Intro.

- **Core margins**
  - When setup the floorplan, remember to leave enough space for power ring
- **Core limited design or Pad limited design**
  - Die size determination
    - When pad width > (core width + core margin), die size is determined by pads.
      And it is called **pad limited design**
    - When pad width < (core width + core margin), die size is determined by core.
      And it is called **core limited design**



  - Adding pad filler
    - Provide power to pad
    - There should be no spacing between pads.
      Pad filler is necessary for core limited design

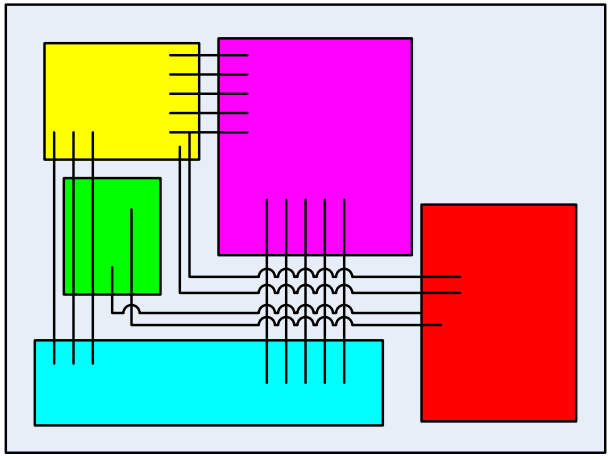


Core limited design

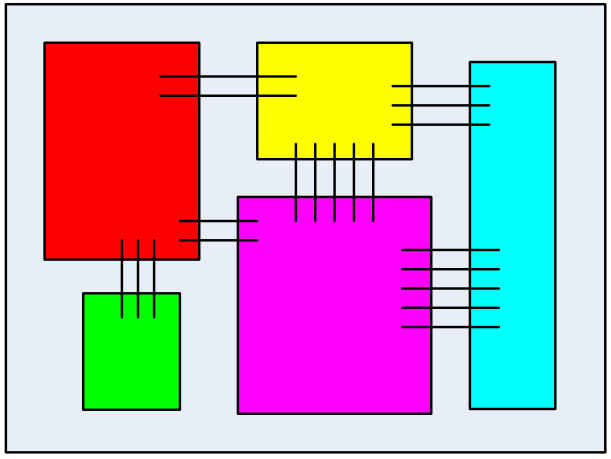
3. Floorplan – Intro.

✓ Floorplan Purposes

- Develop early physical layout to ensure design objective can be achieved
  - Minimum area for low cost
  - Minimum congestion for routing
  - Estimate parasitic capacitance for delay calculation
  - Analysis power for reliability
- Gain early visibility into implementation issues
- Different floorplan, different performance



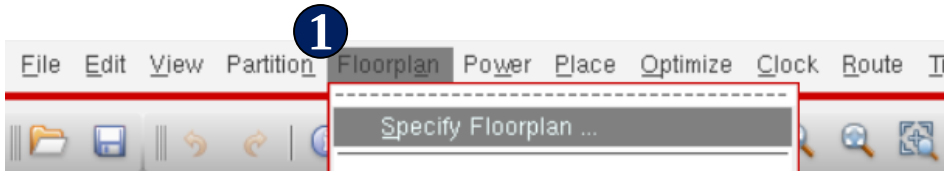
Bad floorplan

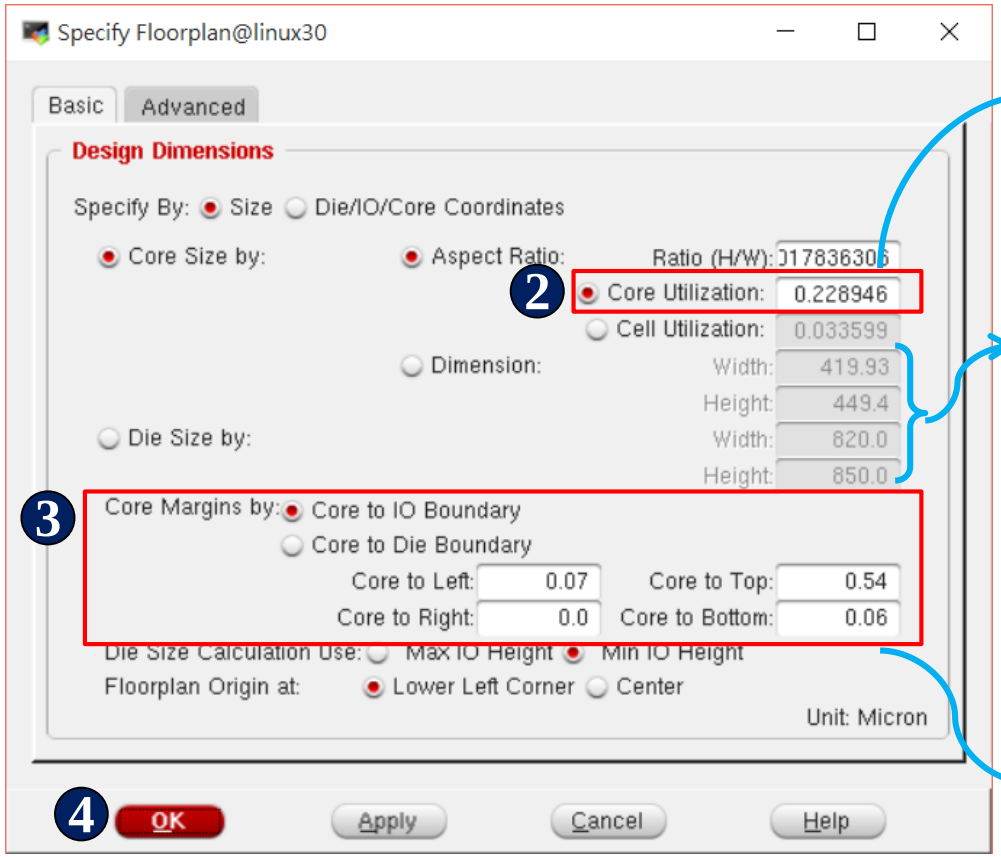


Good floorplan

3. Floorplan

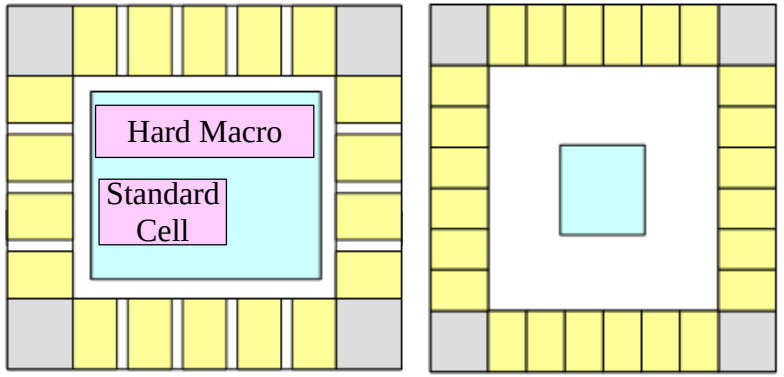
✓ Specify floorplan





Some space are remained for routing.

Set a demanded value of width and height when Hard Macro takes most of the place.

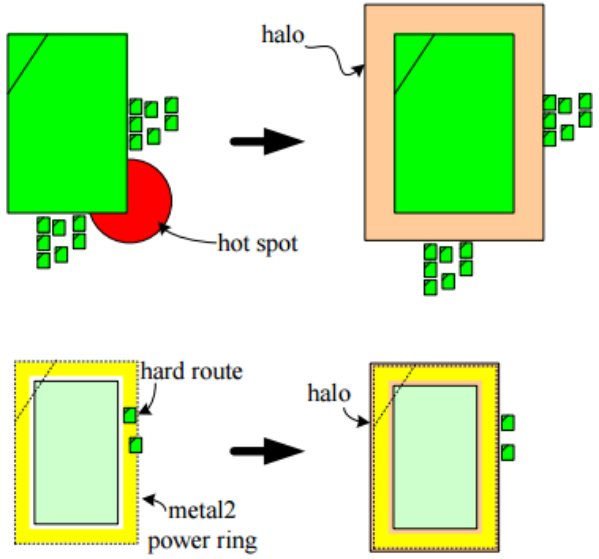


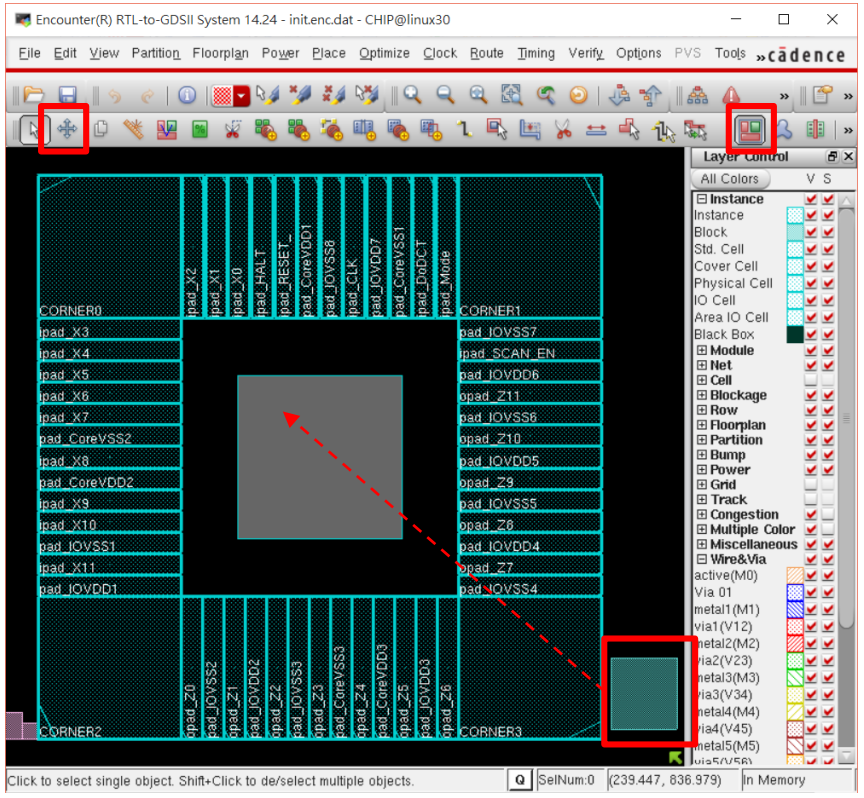
• Core Limited Design • Pad Limited Design

Some space are remained for power rings. (design dependent)

3. Floorplan – Hard Macro

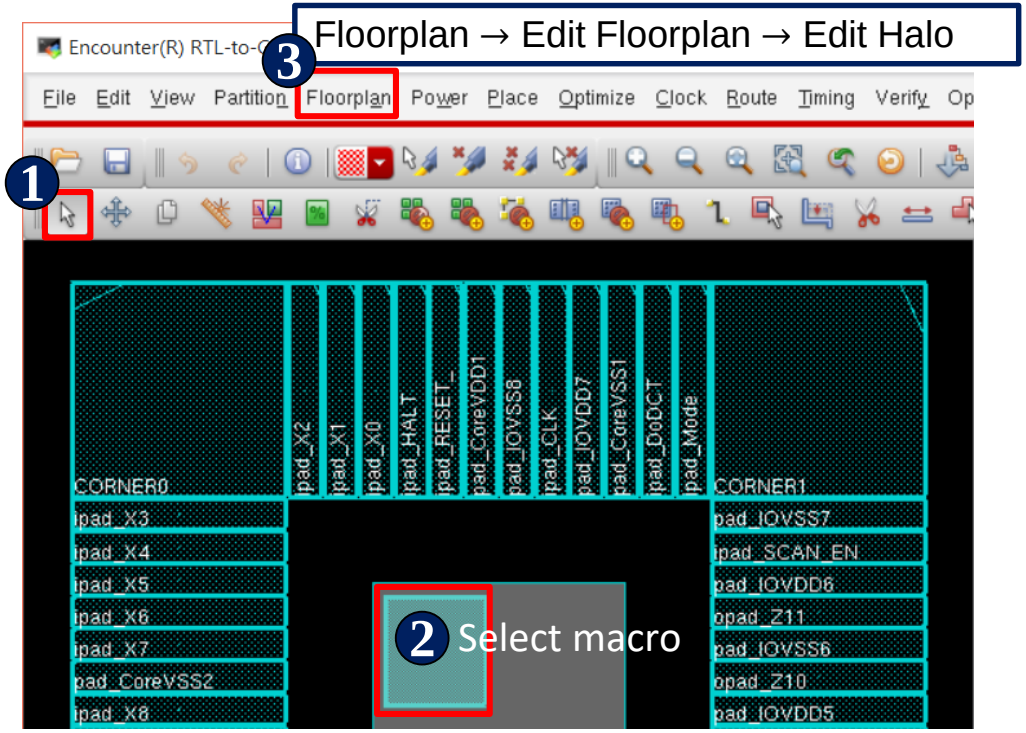
- **Floorplan with Hard Macro**
  - Move macro blocks to proper positions
  - Place macro around the corner to improve routability
- **Add placement blockage for macros**
  - Prevent the placement of blocks and cells to reduce congestion around a block

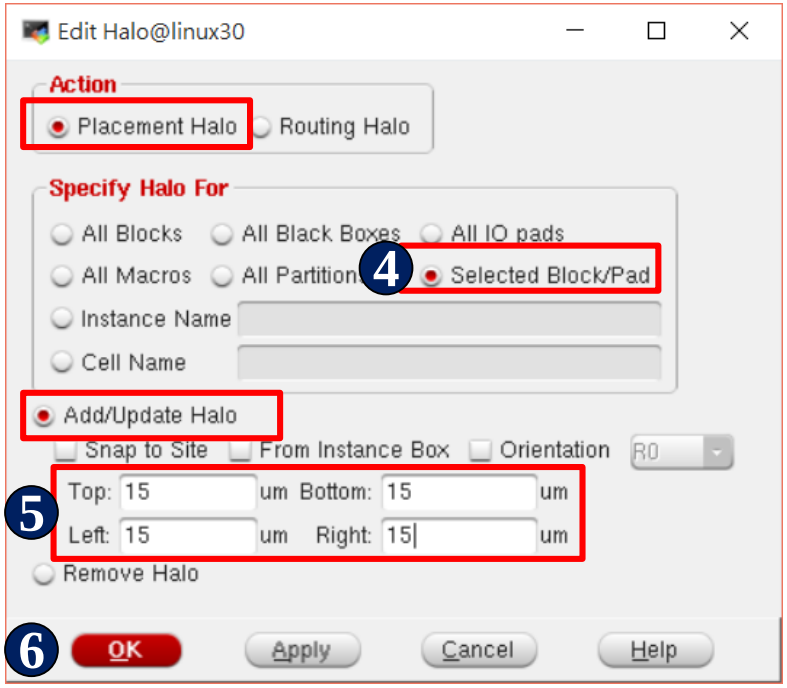




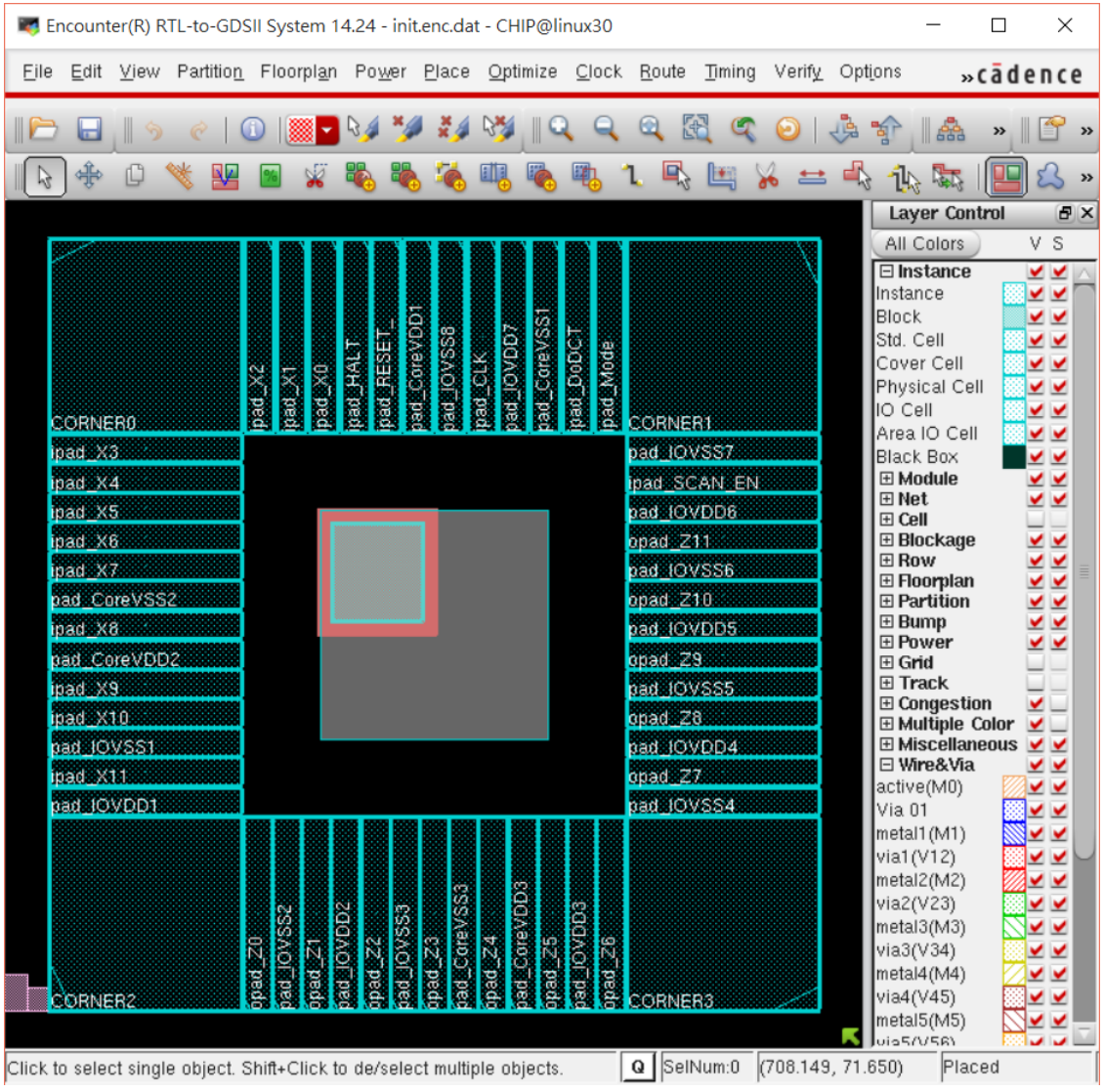
3. Floorplan – Add Halo to Macro

- ✓ **Add Halo**





3. Floorplan - Result



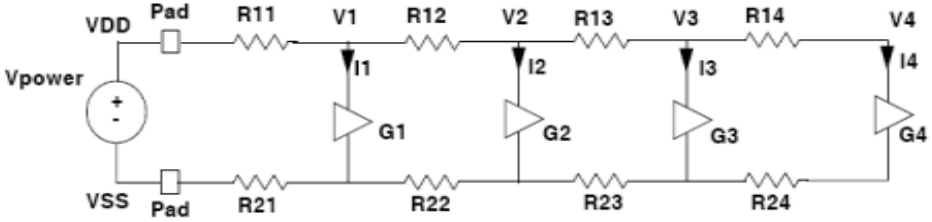
Powerplan

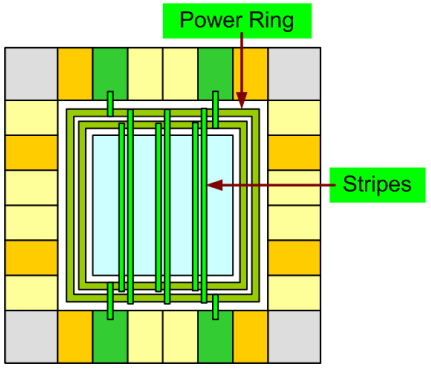
4. Powerplan – Intro.

✓ Power issue

– IR drop

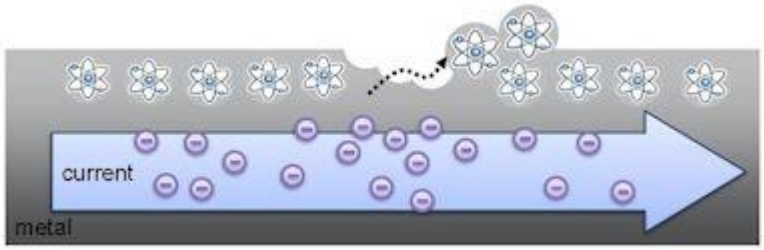
- IR drop is the problem of voltage drop of the power and ground due to high current flowing through the power-ground resistive network
- When there are excessive voltage drops in the power network or voltage rises in the ground network, the device will run at **slower speed**
- IR drop can cause the chip to fail due to
  - Performance (circuit running slower than specification)
  - Functionality problem (setup or hold violations)
  - Unreliable operation (less noise margin)
  - Power consumption (leakage power)
  - Latch up (short circuit)
- Prevention: adding stripes to avoid IR drop on cell's power line

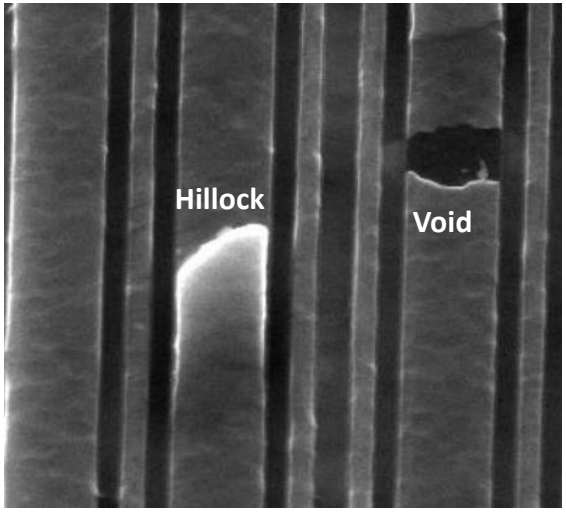




4. Powerplan – Intro.

- ✓ **Power issue**
  - **Metal migration (electromigration)**
    - Under **high currents**, electron collisions with metal grains cause the metal to move. The metal wire may be **open circuit** or **short circuit**.
    - Prevention: sizing power supply lines to ensure that the chip does not fail
    - Experience: make current density of power ring < 1mA/μm

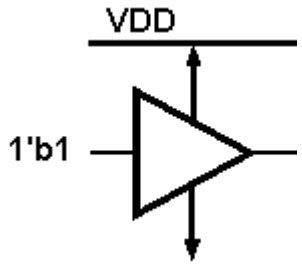


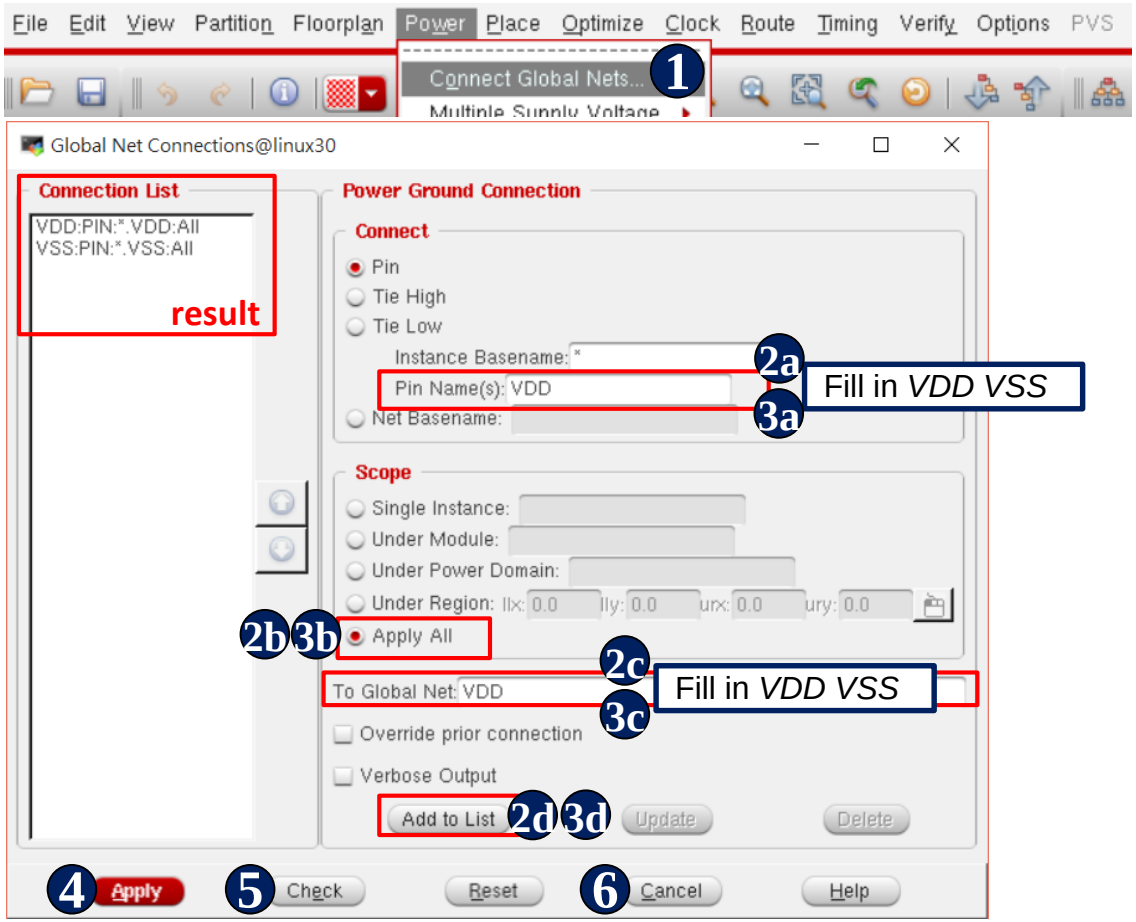


4.1. Powerplan – Global Nets

✓ 4.1 Connect global nets

- Connect all the 1'b1 to VDD and 1'b0 to VSS
- Add power nets and pins connection list
- Add ground nets and pins connection list

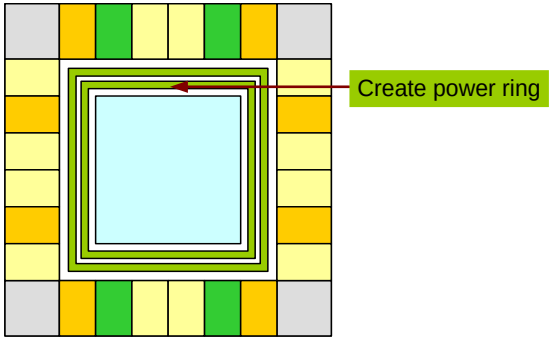




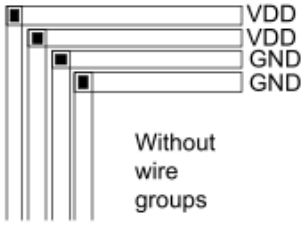
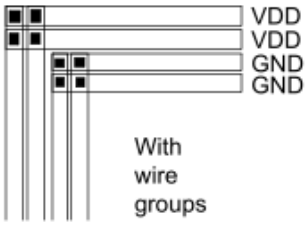
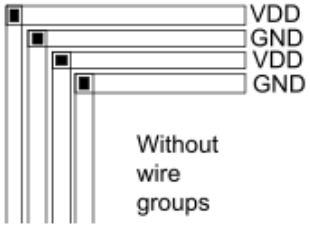
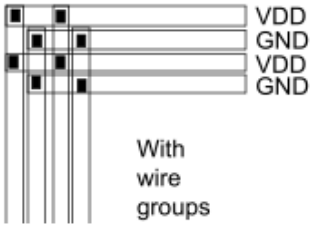
Create 2 global nets connection list
2 Connect ◆ Pins: *VDD* To Global Net: *VDD*
3 Connect ◆ Pins: *VSS* To Global Net: *VSS*

4.2. Powerplan – Power Ring

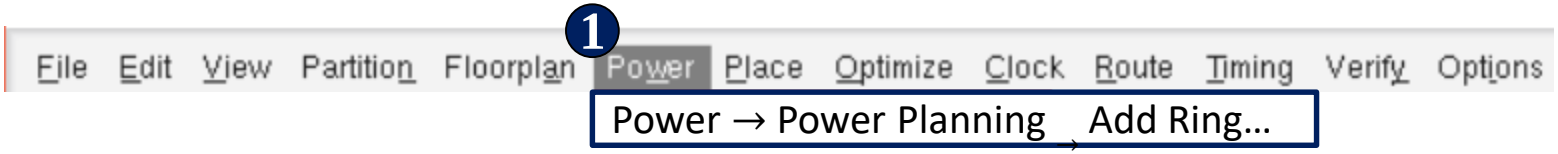
✓ 4.2 Core power ring



– Interleaving and Wire group

	Without Wire group	With Wire group
Without Interleaving		
Interleaving		

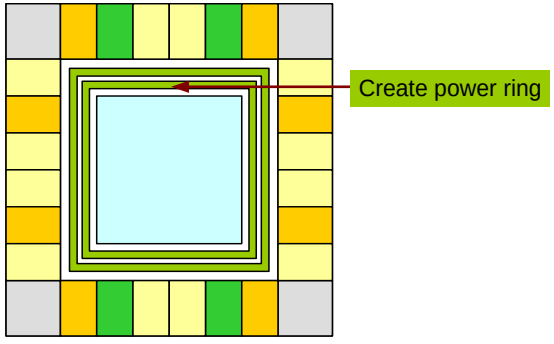
4.2. Powerplan – Power Ring

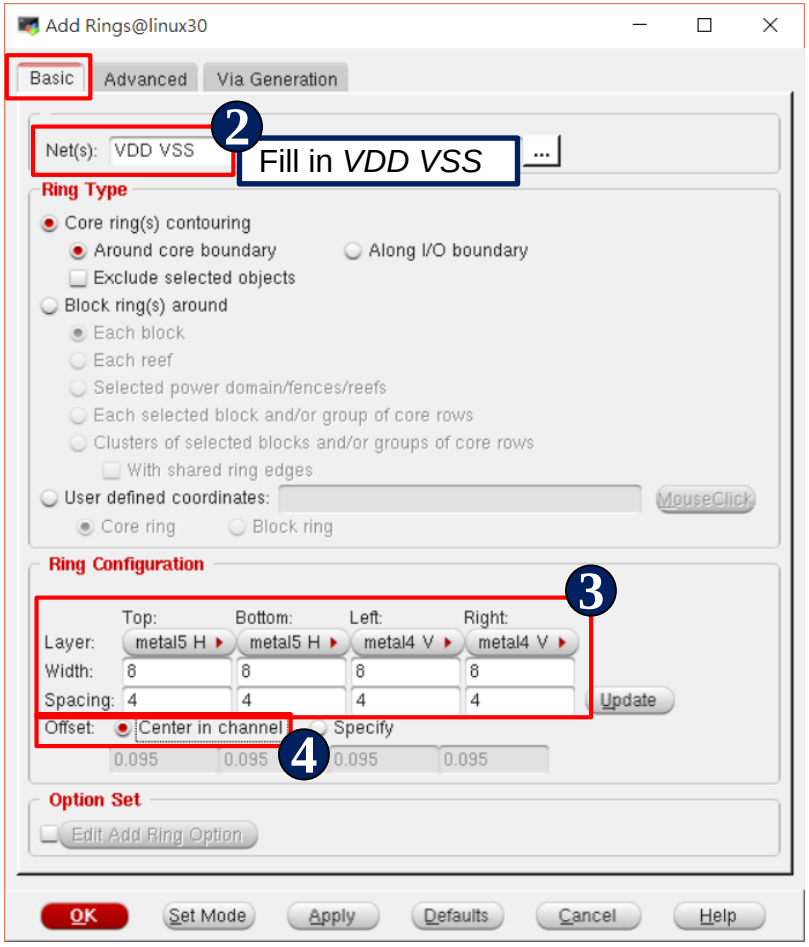


– In Basic tab

3

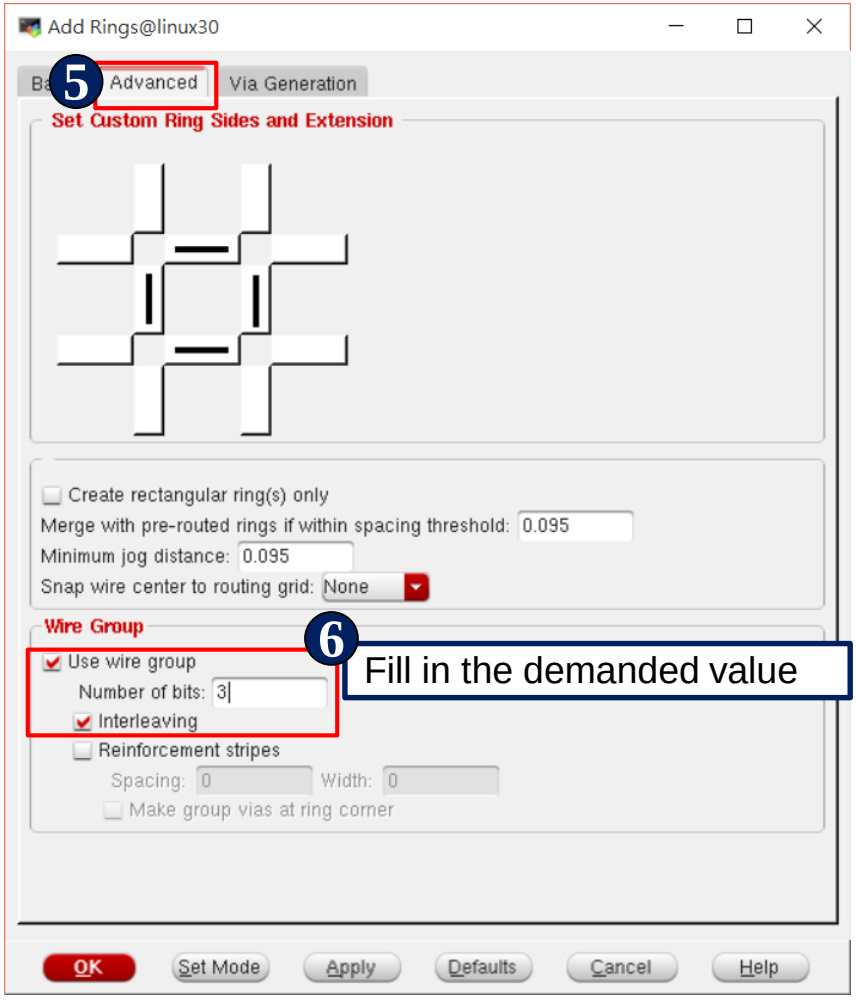
Field	Fill In
Top and Bottom Layer	metal 5
Left and Right Layer	metal 4
Width	8
Spacing	4





4.2. Powerplan – Power Ring

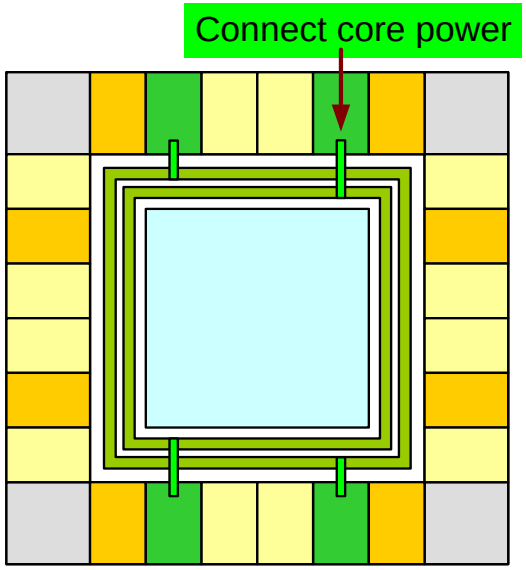
- In Advanced tab



Tip:
Click **Apply** then you can undo this step
Click **OK** then undo is not allowed

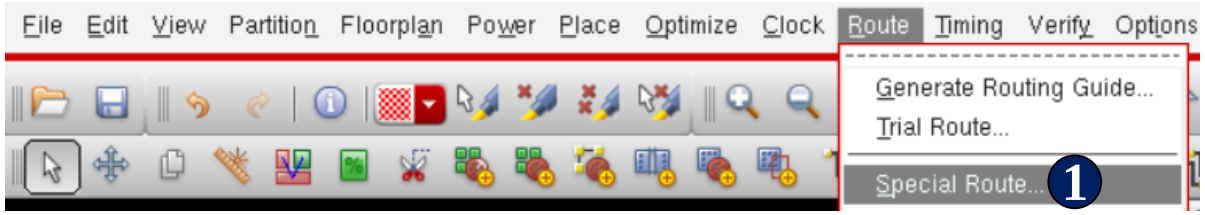
4.3. Powerplan – Core Power Pin

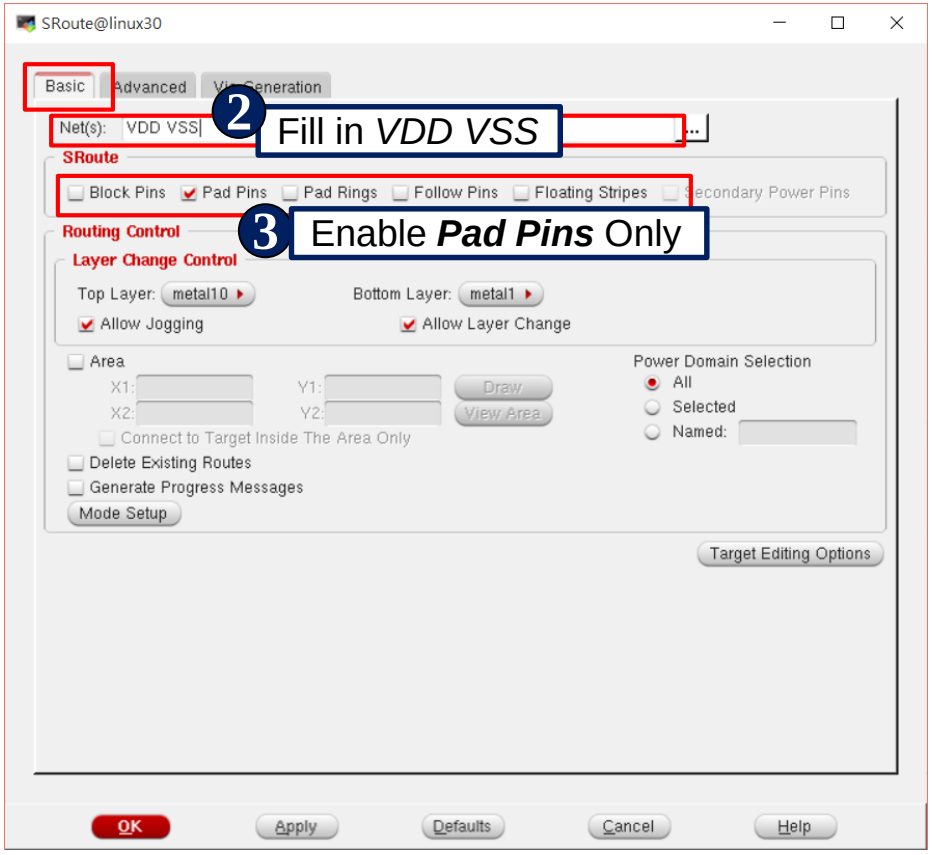
- ✓ **4.3 Core power pin**
  - Connect core power pads to power rings

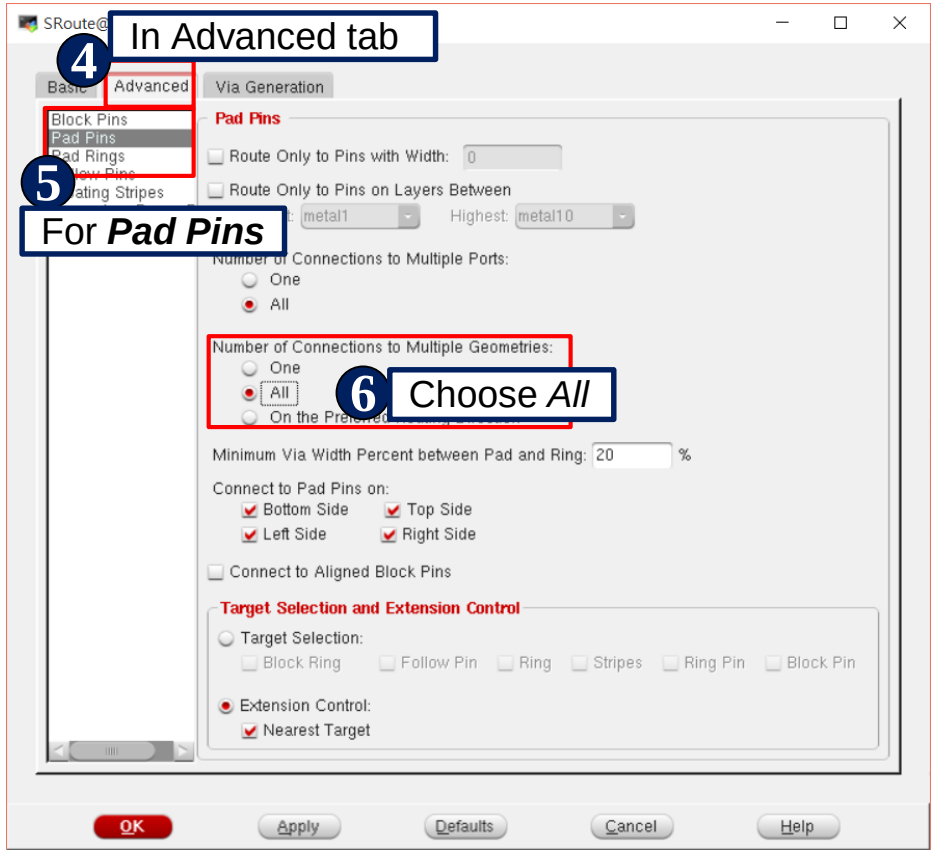


4.3. Powerplan – Core Power Pin

- ✓ **Connect core power pin**

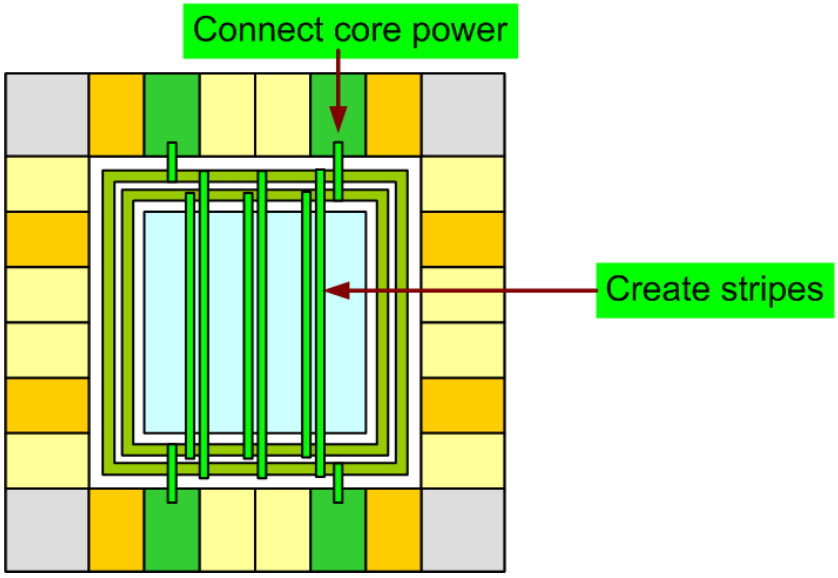






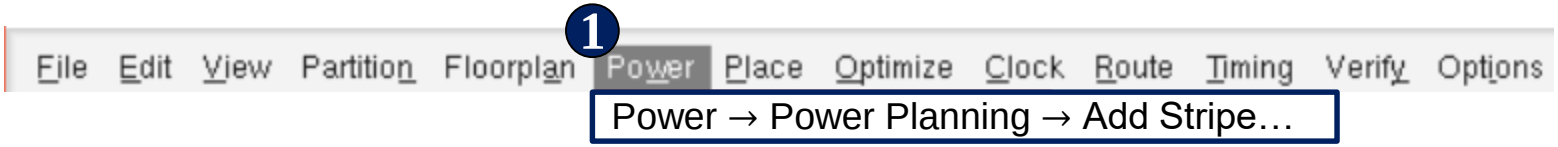
4.4. Powerplan – Stripes

- ✓ **4.4 Stripes**
  - IR drop prevention

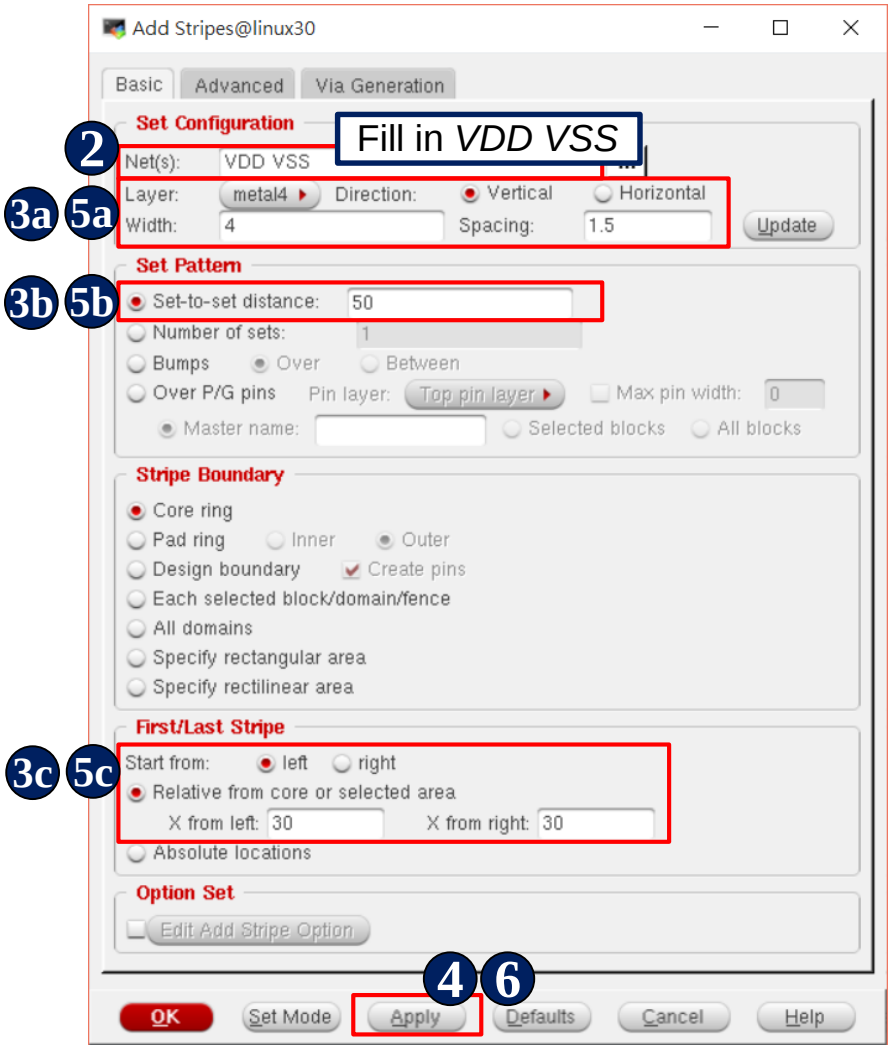


Note:
Stripes should not be mixed with core power pins

4.4. Powerplan – Stripes

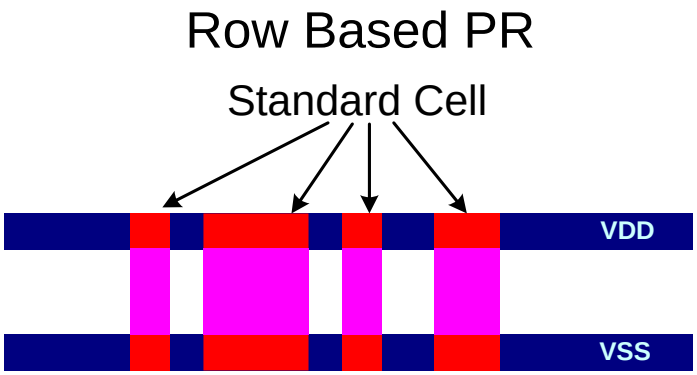


Note:
Use even metal for vertical stripe
Odd metal for horizontal stripe

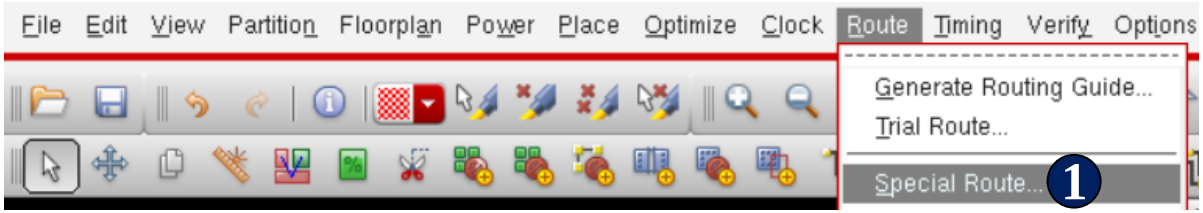


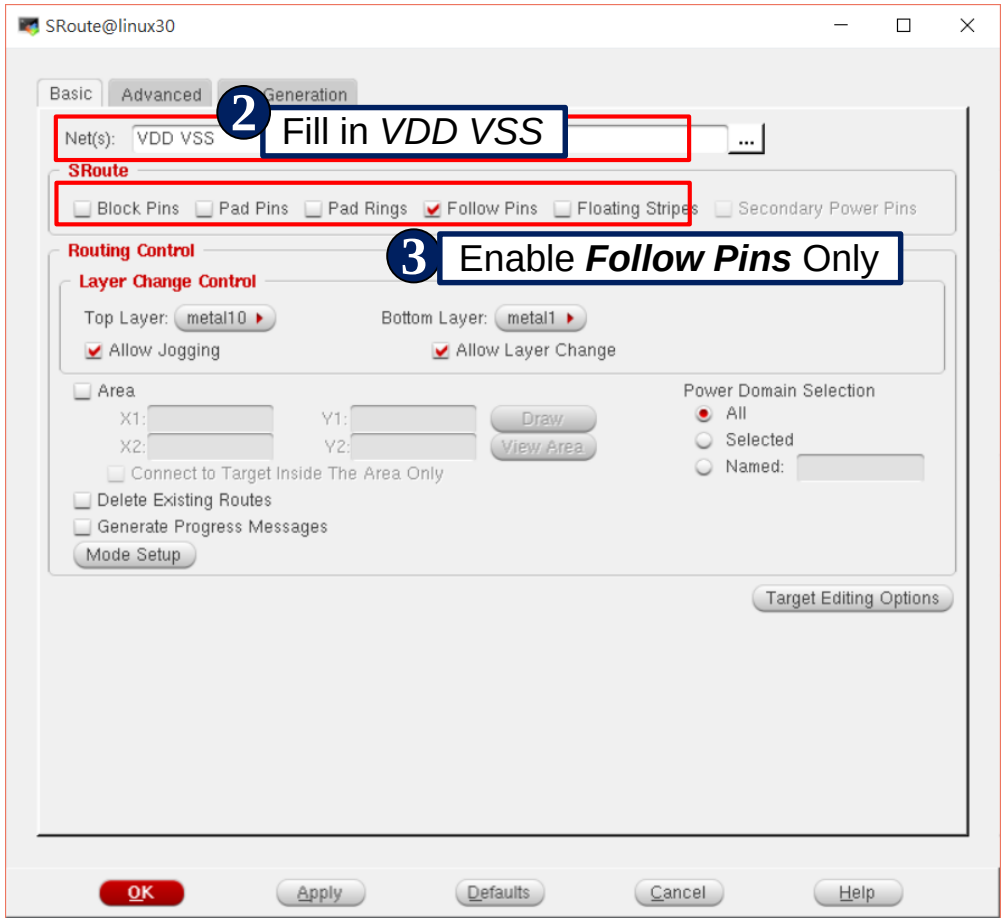
4.5. Powerplan – Std. Cell Power Line

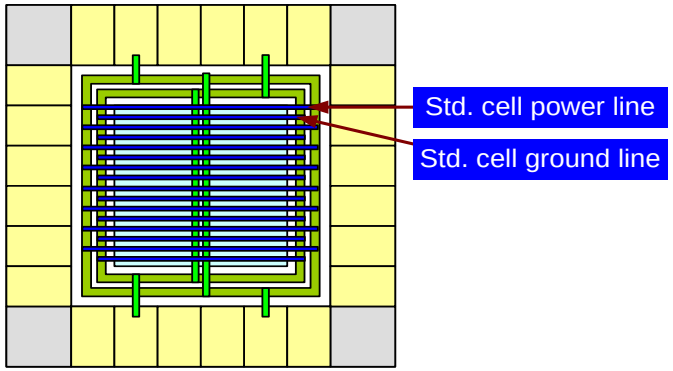
- ✓ **Connect standard cell power line**



4.5. Powerplan – Std. Cell Power Line



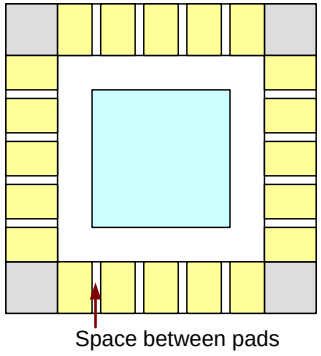




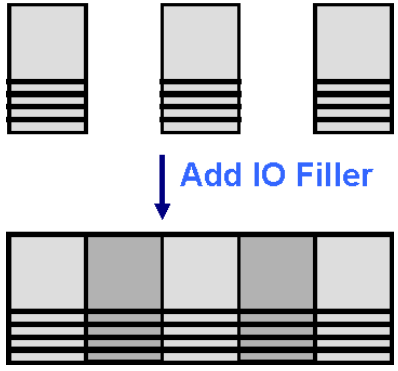
4.6. Add Pad Filler

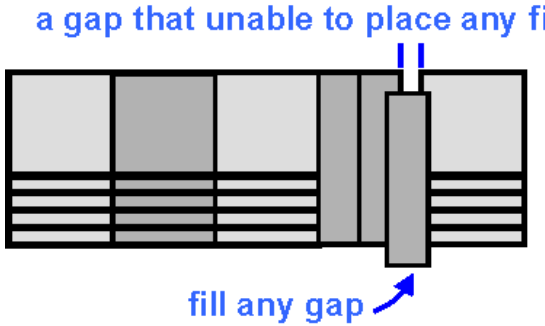
✓ 4.6 Add pad filler

- There should be no spacing between pads.
  Therefore adding pad filler is necessary for core limited design
- Pad filler must be added before detailed routing, otherwise there may be some DRC/LVS violations after pad filler insertion



Core limited design





Note: Only the smallest filler can use –fillAnyGap option

- Text command

From wider fillers to narrower ones

```
innovus > addIoFiller –cell PADFILLER20 –prefix IOFILLER
innovus > addIoFiller –cell PADFILLER10 –prefix IOFILLER
innovus > addIoFiller –cell PADFILLER5 –prefix IOFILLER
innovus > addIoFiller –cell PADFILLER1 –prefix IOFILLER
innovus > addIoFiller –cell PADFILLER05 –prefix IOFILLER
innovus > addIoFiller –cell PADFILLER005 –prefix IOFILLER -fillAnyGap
```

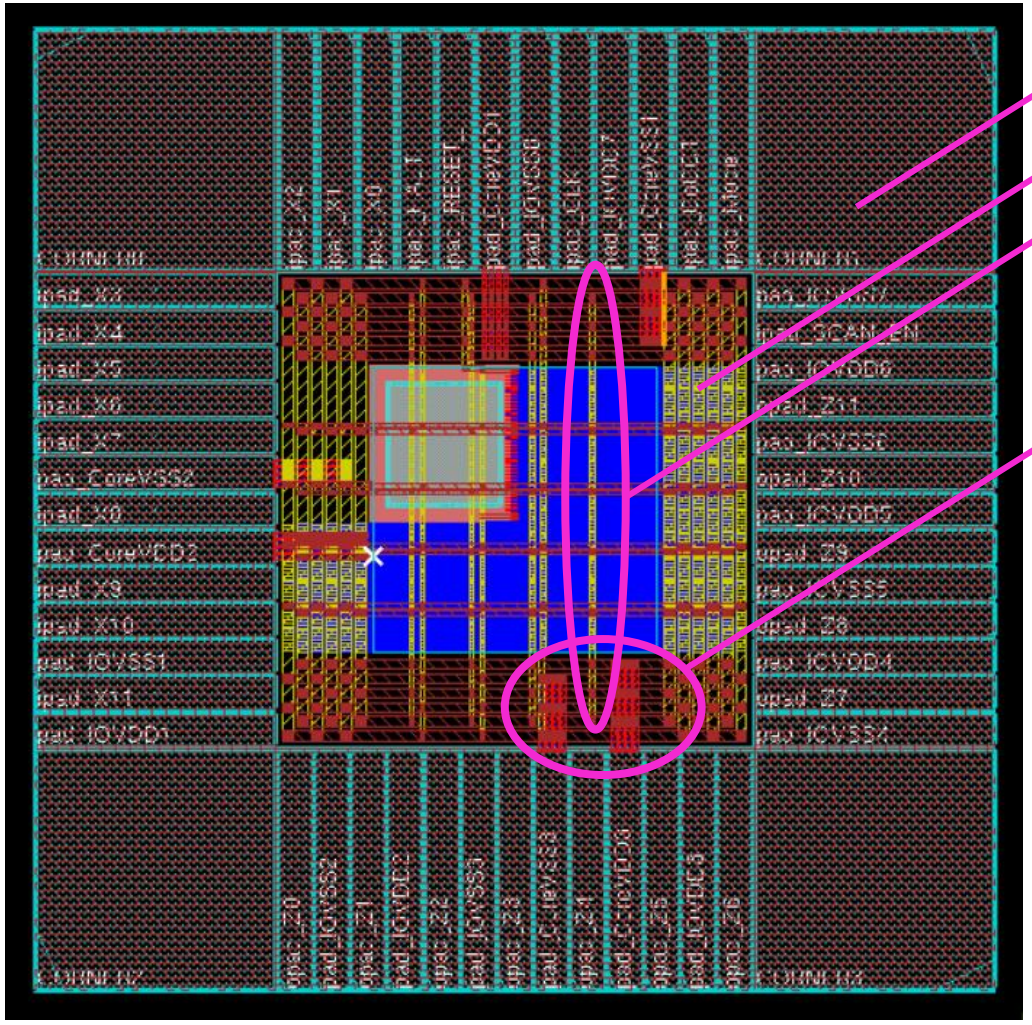
4.7. Routing Blockage and Verification

- ✓ **Add routing blockage**
  - To prevent wires routed on IO pads, we add routing blockage before routing
- ✓ **Verify geometry**
  - Check if the layout so far does not violate technology design rules
- ✓ **Verify wire connectivity**
  - Check if potential flaws with interconnects and pins exist
- ✓ **Calibre DRC**
  - Design rule check with more rigorous rules
  - For advanced process, the verifications provided by encounter may be insufficient, so we must check Calibre DRC

Note:
It's suggested to do the checks after powerplan

4. Powerplan - Result

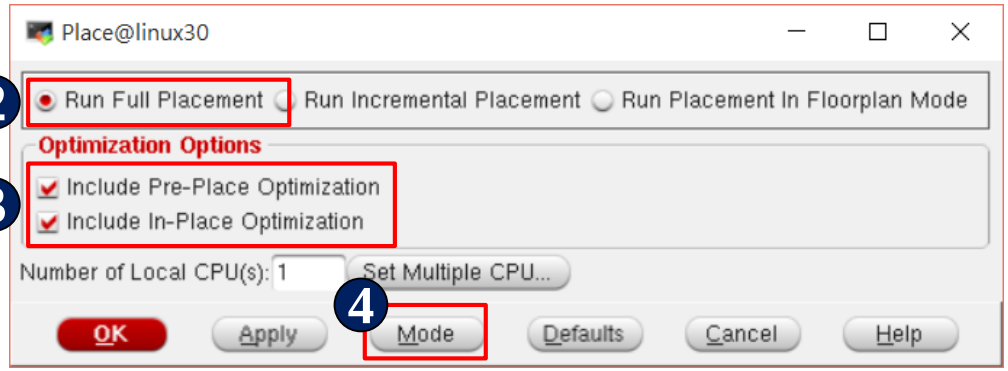
- ✓ **Tip: Save your design before adding std. cell power lines**

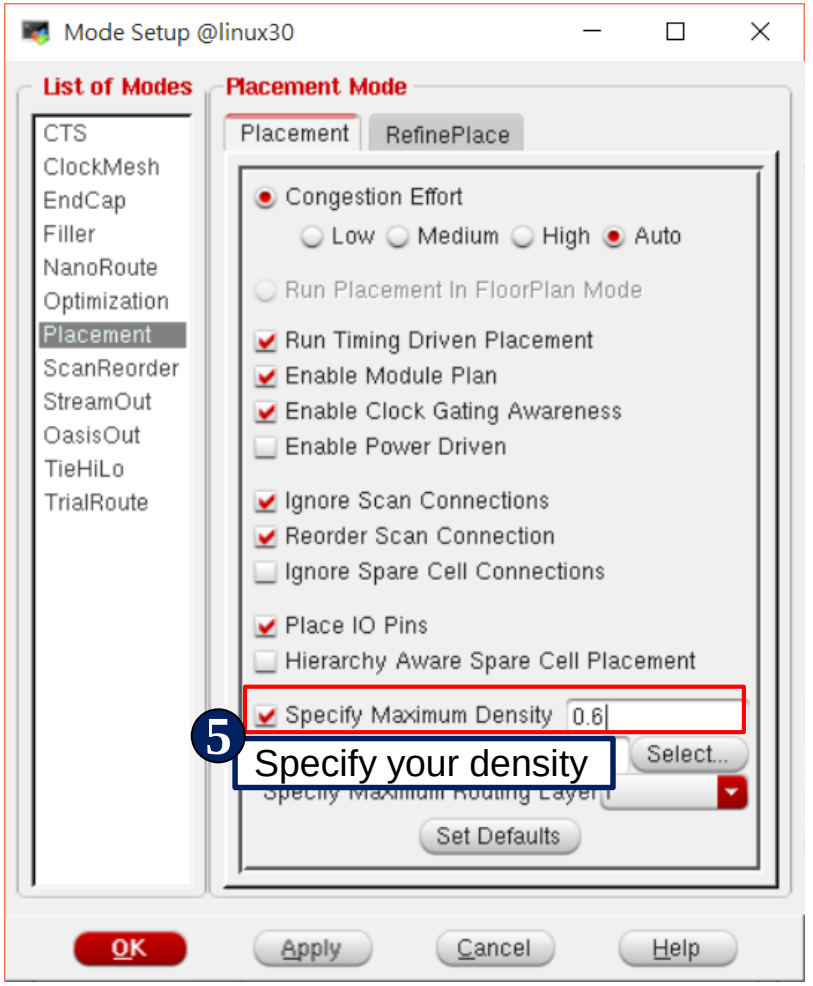


Placement

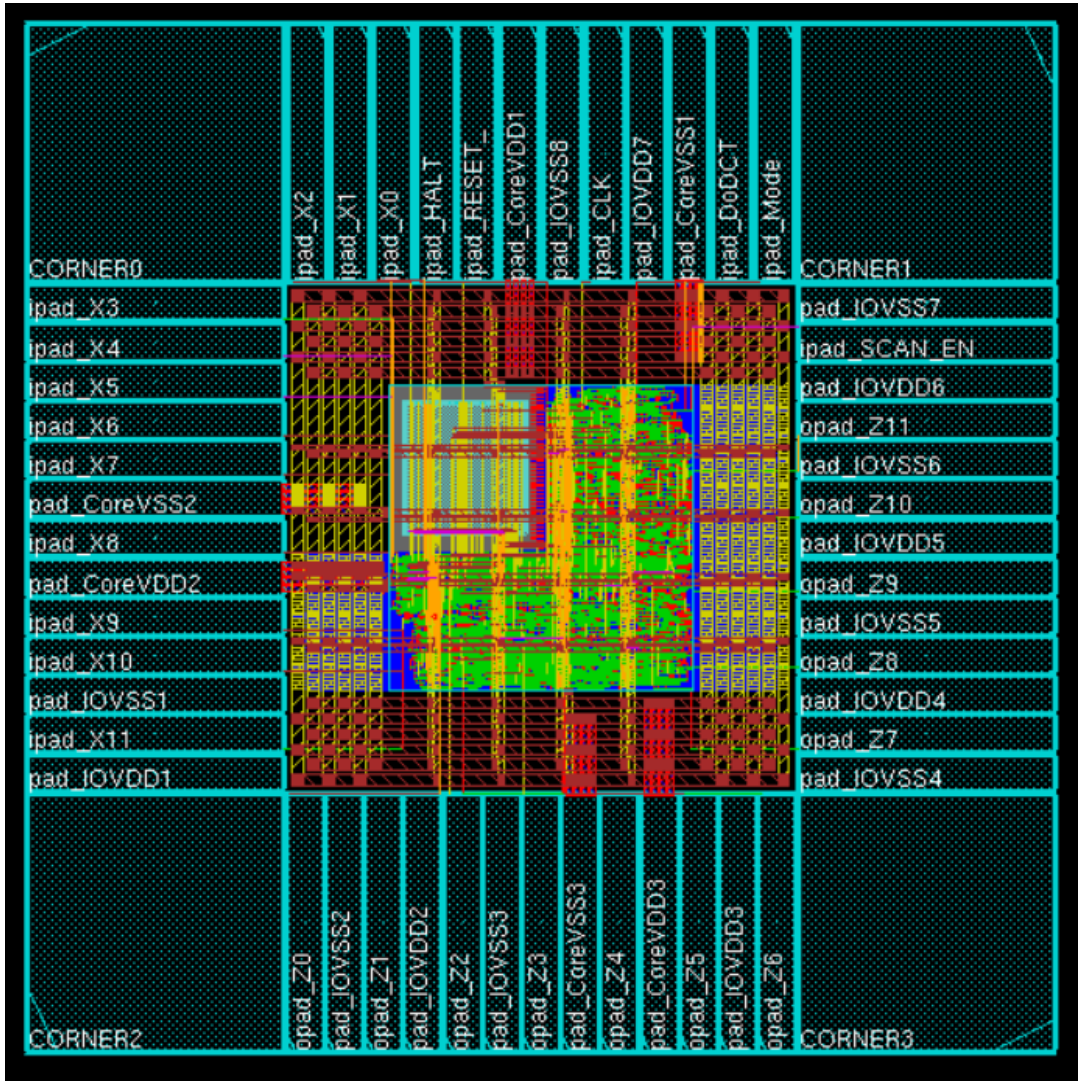
5. Standard Cell Placement

✓ **Place standard cells**





5. Standard Cell Placement - Result



Clock Tree Synthesis

6. Pre-CTS Timing Analysis

- ✓ **Timing analysis**
  - Open *Timing* → *Report Timing…*
  - Design Stage ◆ Pre-CTS
- ✓ **Operations**
  - Trial route
  - RC extraction
  - Timing analysis
- ✓ **Reports**
  - Generated reports are saved in ./timingReports/
    - CHIP_preCTS.cap
    - CHIP_preCTS.fanout
    - CHIP_preCTS.tran
    - CHIP_preCTS_all.tarpt
  - Timing summary and DRVs (design rule violations)
  - If the timing is MET, pre-CTS optimization can be skipped

timeDesign Summary

Setup mode	all	reg2reg	default
WNS (ns):	2.239	2.239	2.853
TNS (ns):	0.000	0.000	0.000
Violating Paths:	0	0	0
All Paths:	523	489	415

Negative slack should be fixed

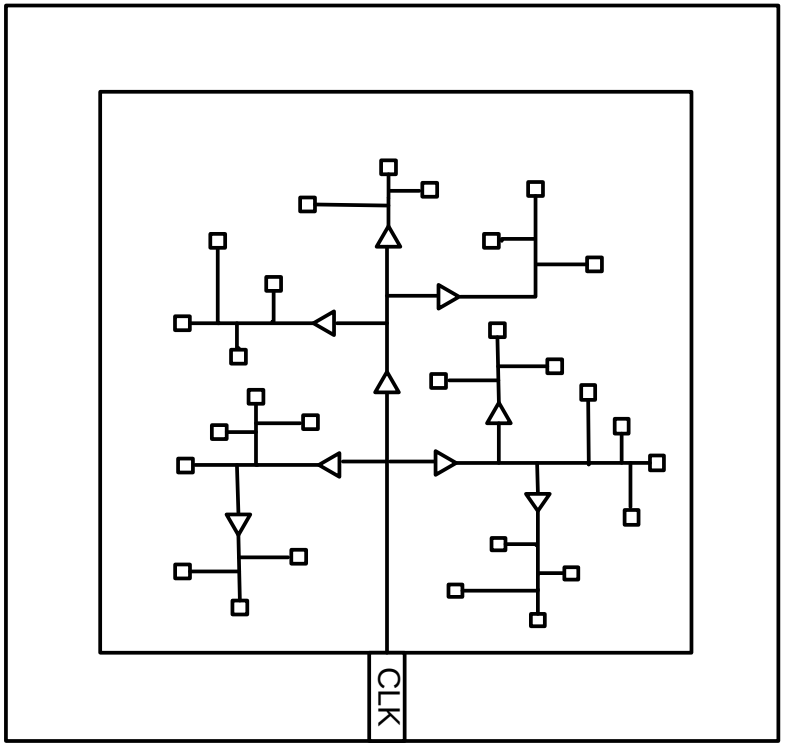
DRVs	Real		Total
	Nr nets(terms)	Worst Vio	Nr nets(terms)
max_cap	0 (0)	0.000	0 (0)
max_tran	17 (18)	-0.192	17 (18)
max_fanout	0 (0)	0	0 (0)
max_length	0 (0)	0	0 (0)

6. Pre-CTS Optimization

- ✓ **Optimize timing with ideal clocks**
  - Open *Optimize* → *Optimize Design…*
  - Design Stage ◆ Pre-CTS
  - Optimization Type ◆ Max Cap ◆ Max Tran ◆ Max Fanout
- ✓ **Operation**
  - Repair Setup slack, Setup times
  - Repair Design rule violations (DRVs)

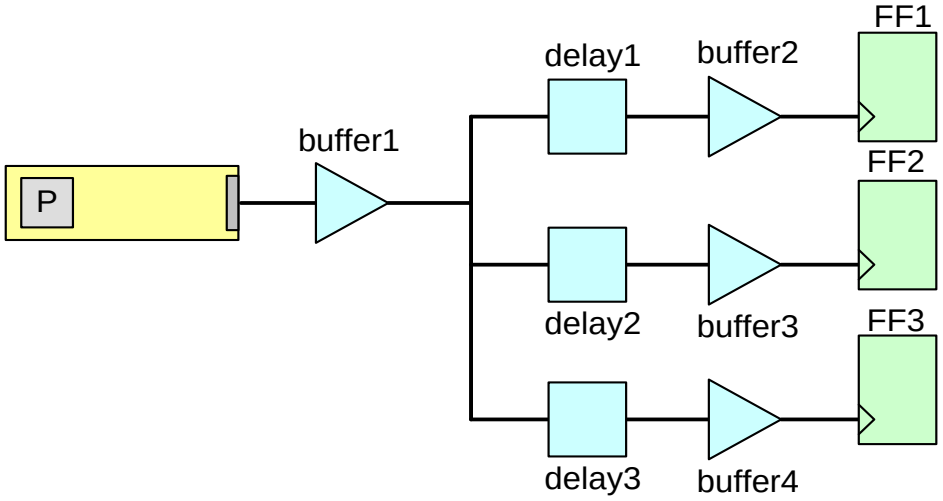
7. Clock Tree Synthesis – Intro.

- ✓ **Clock problem**
  - Heavy clock net loading
  - Long clock insertion delay
  - Clock skew
  - Skew across clocks
  - Clock is power hungry
  - Electromigration on clock net



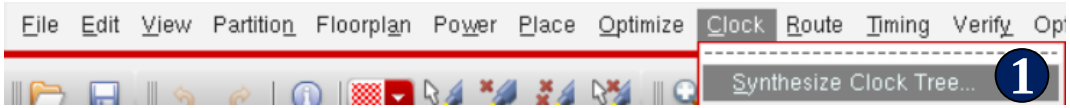
7. Clock Tree Synthesis – Intro.

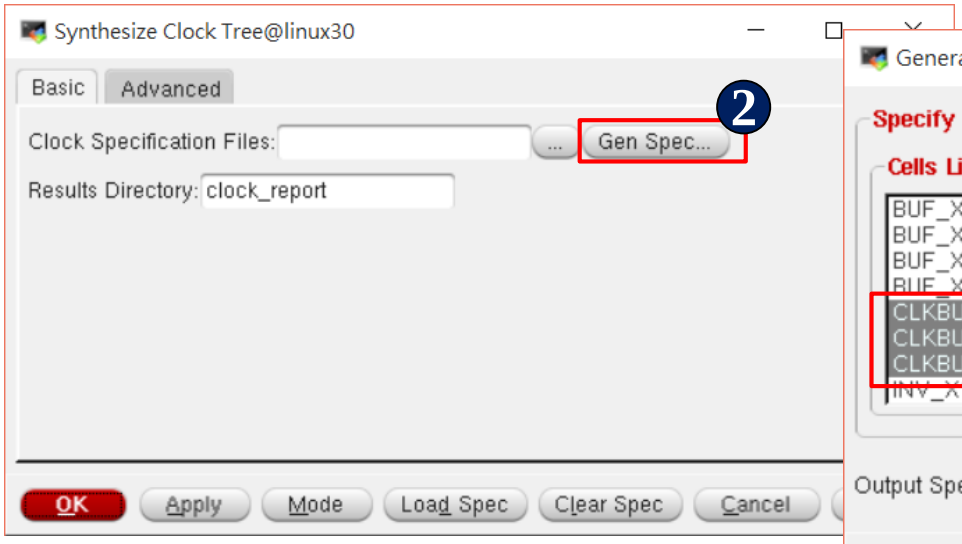
- **The process of clock tree synthesis includes**
  - Create clock tree spec file
  - Build a buffer distribution network
  - Route clock nets using CTS-NanoRoute
- **In automatic CTS mode, Innovus will do the following things**
  - Build the clock buffer tree according to the clock tree specification file
  - Balance the clock phase delay with appropriately sized, inserted clock buffers

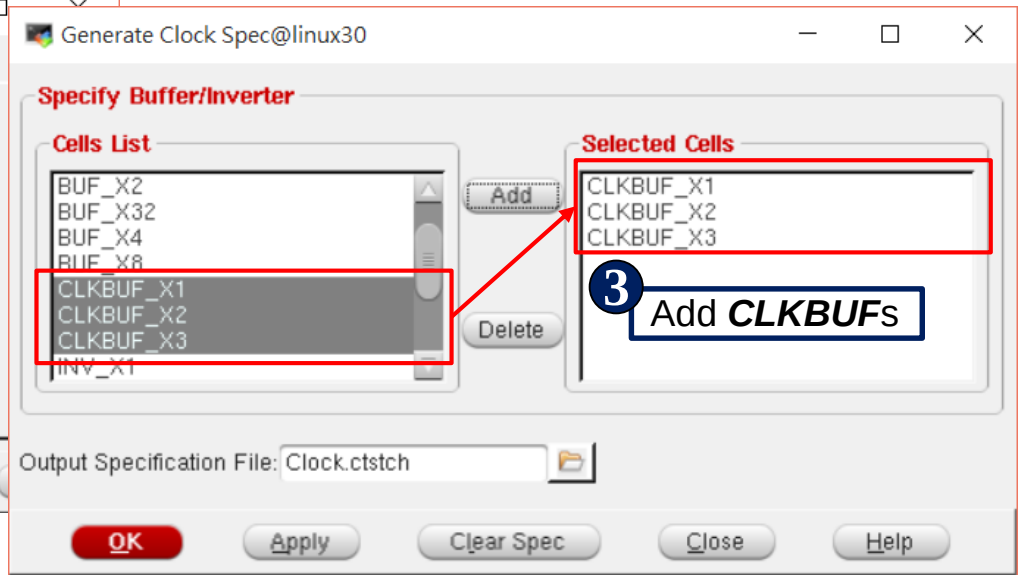


7. Clock Tree Synthesis

- ✓ **Update sdc constraint**
  - Add the **set_propagated_clock** command
- ✓ **Generate clock spec. file**







- ✓ **Synthesize clock tree**
  - encounter > setCTSMode -engine ccopt_from_edi_spec
  - encounter > ccopt_design -cts -ckSpec

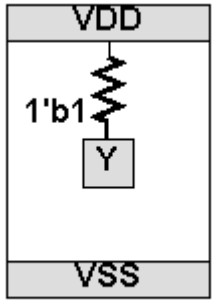
8. Post-CTS Timing Analysis

- ✓ **Timing analysis**
  - Open *Timing* → *Report Timing…*
  - Design Stage ◆ Post-CTS
- ✓ **Reports**
  - Generated reports are saved in ./timingReports/
  - Timing summary and DRVs (design rule violations)
  - If the timing is MET, post-CTS optimization can be skipped

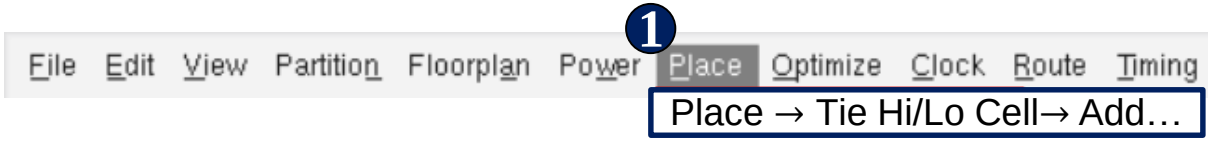
  - CHIP_postCTS.cap
  - CHIP_postCTS.fanout
  - CHIP_postCTS.tran
  - CHIP_postCTS_all.tarpt
- ✓ **Optimization**
  - Open *Optimize* → *Optimize Design…*
  - Design Stage ◆ Post-CTS
  - Optimization Type ◆ Max Cap ◆ Max Tran ◆ Max Fanout
- ✓ **Operation**
  - Repair Setup slack, Setup times
  - Repair Design rule violations (DRVs)

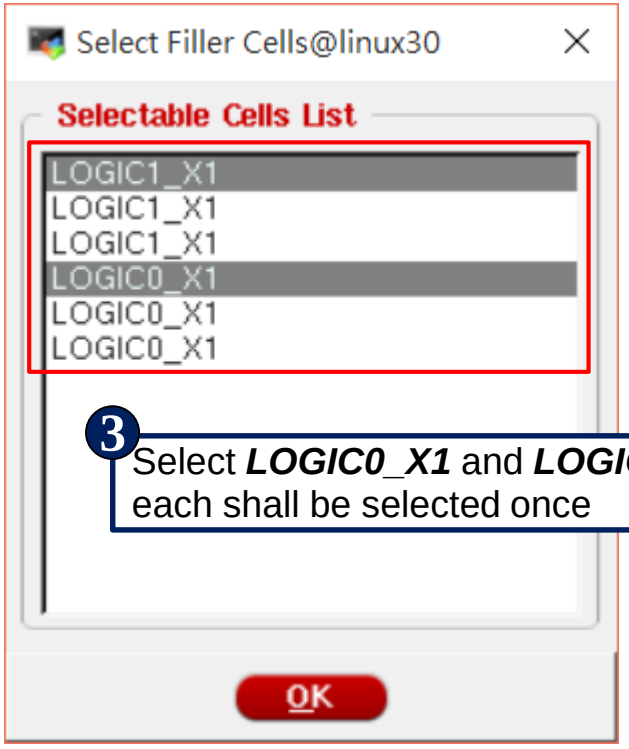
9. Add Tie Hi/Lo Cell

- **Add Tie Hi/Lo Cell**
  - Connect to supply voltage or ground with resister
  - Added for ESD protection



Tie High Cell

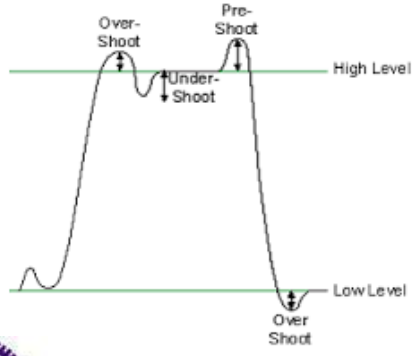


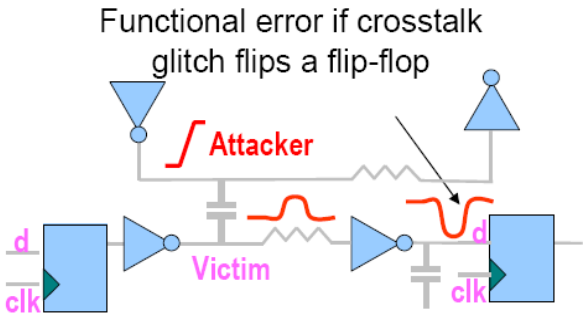


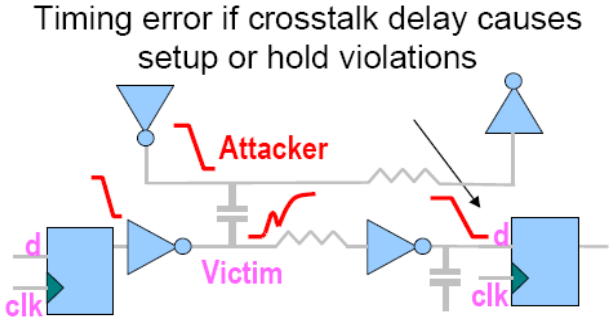
Routing

10. NanoRoute – Signal Integrity

- ✓ **Signals are subject to noise, distortion, loss and so forth in real world**
- ✓ **Signal integrity is a set of measures of the quality of signal**
- ✓ **Signal Integrity (SI) Issue**
  - Crosstalk
  - Charge sharing
  - Supply noise
  - Leakage
  - Overshoot
  - Under shoot



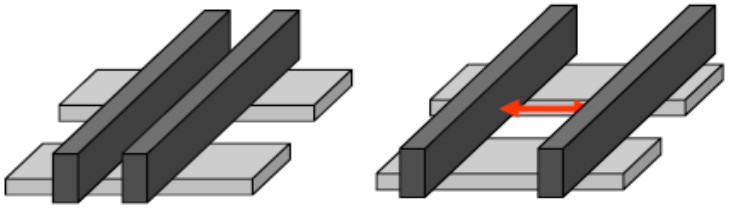




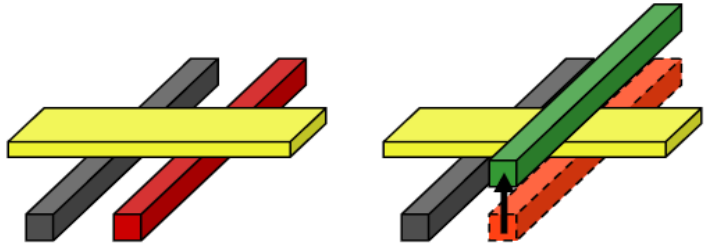
10. NanoRoute – Signal Integrity

✓ **Routing-based SI prevention**

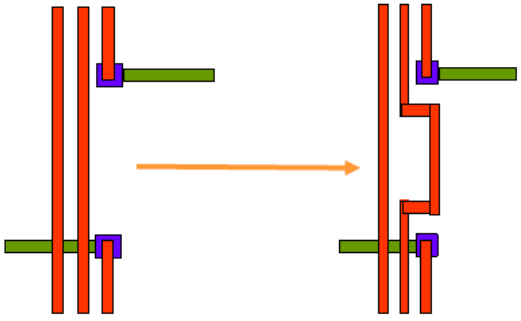
- Wiring Spacing



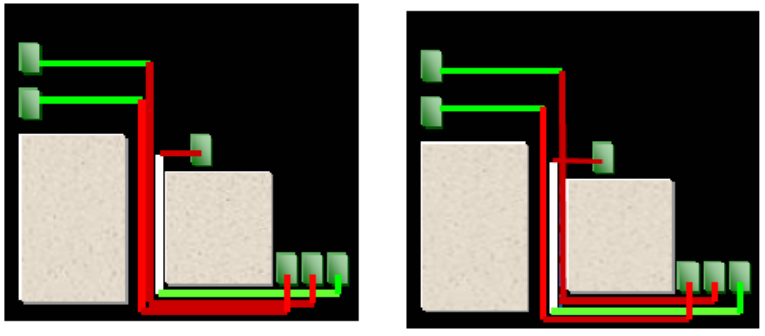
- Layer Switching



- Parallel Wires Reducing

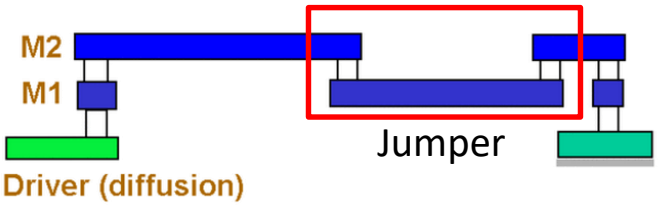


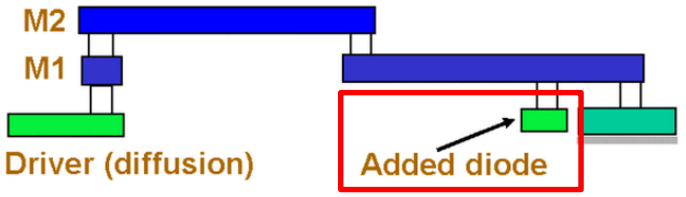
- Net Re-ordering



10. NanoRoute – Antenna Effect

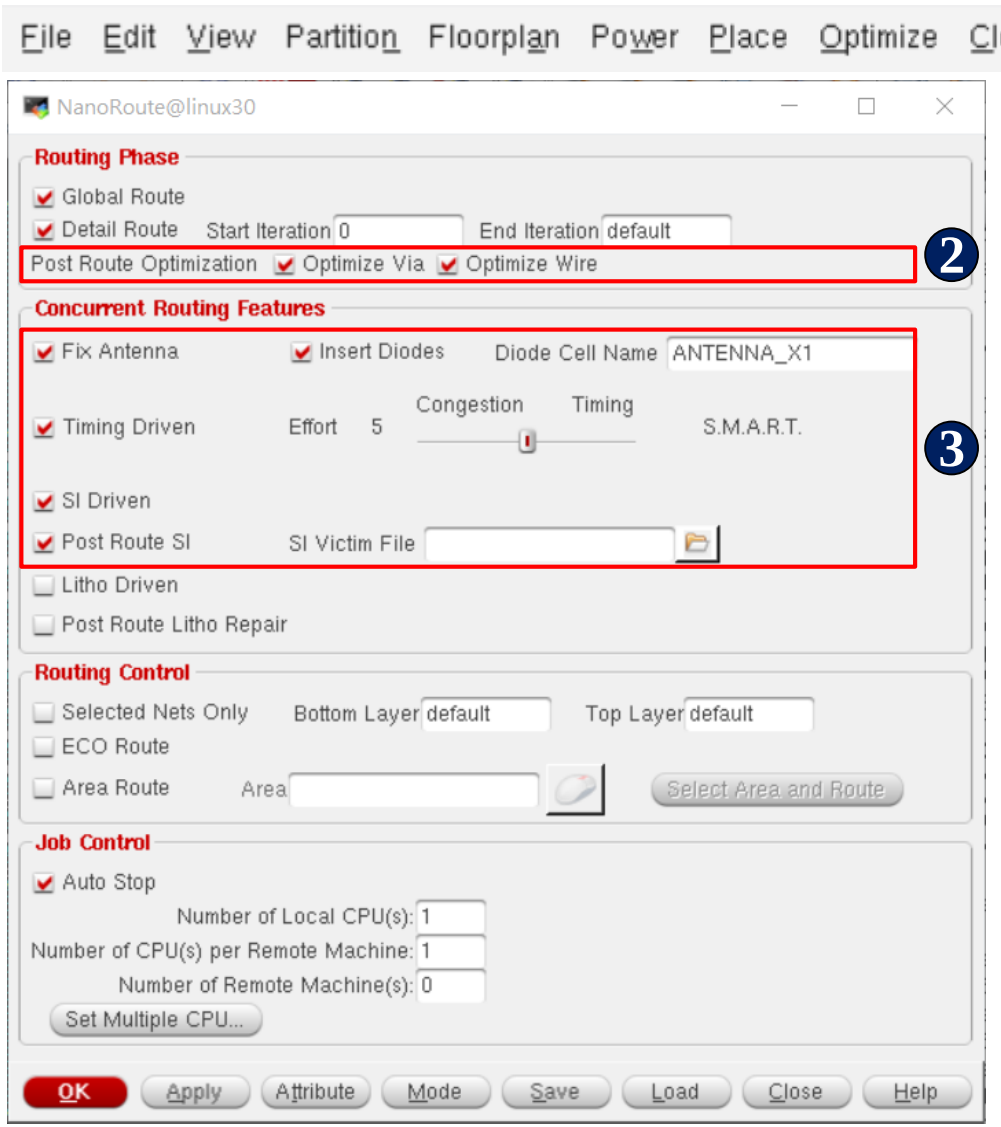
- ✓ **Antenna Effect**
  - In a chip manufacturing process, metal is initially deposited so it covers the entire chip
  - Then, the unneeded portions of the metal are removed by etching, typically in plasma (charged particles).
  - The exposed metal collect charge from plasma and form voltage potential.
  - If the voltage potential across the gate oxide becomes large enough, the current can damage the gate oxide.
- ✓ **Solution**
  - Add jumper (change metal layer)
  - Add antenna cell (**diode**)





10. NanoRoute

✓ NanoRoute



NanoRoute

Routing Phase

2
- Optimize Via
- Optimize Wire

Concurrent Routing Features

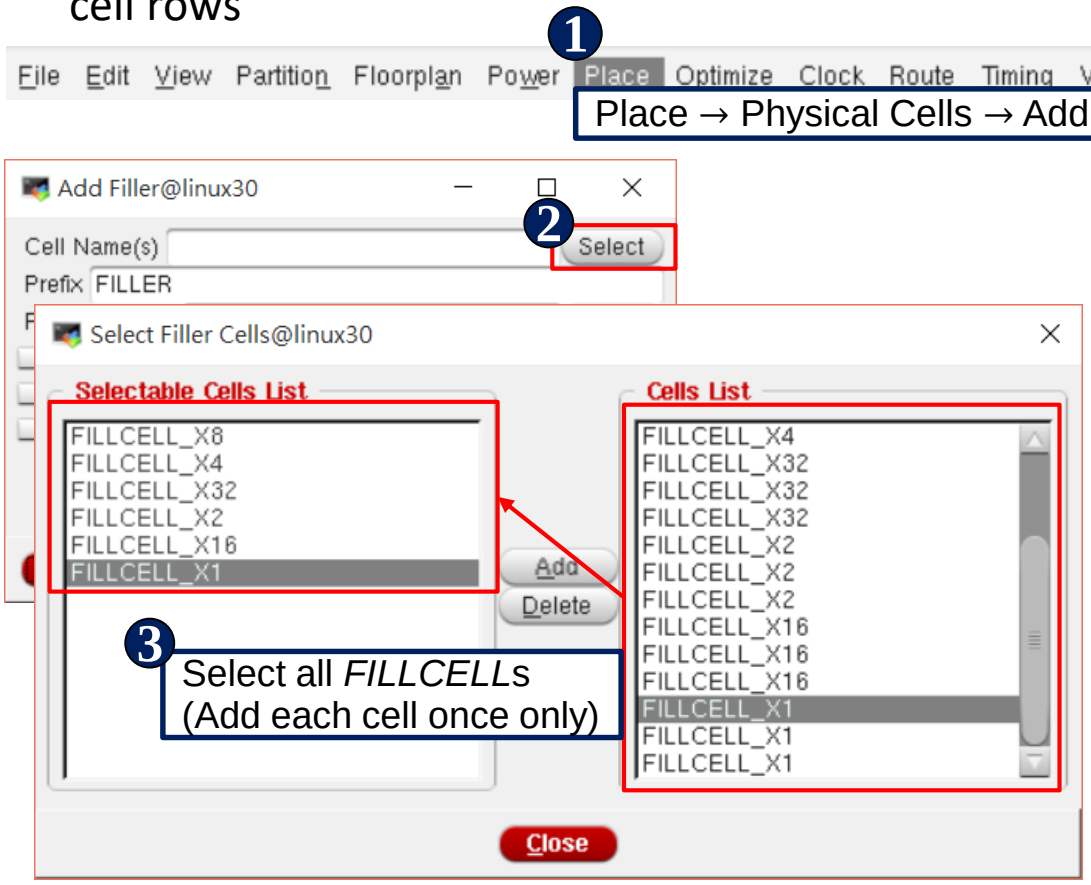
3
- Fix Antenna
- Insert Diodes
  - Diode Cell Name: ANTENNA_X1
- Timing Driven
- SI Driven
- Post Route SI

11. Post-Route Timing Analysis

- ✓ **Timing analysis (Setup)**
  - Open *Timing* → *Report Timing…*
  - Design Stage ◆ Post-Route
  - Analysis Type ◆ Setup
- ✓ **Optimization (Setup)**
  - Open *Optimize* → *Optimize Design…*
  - Design Stage ◆ Post-Route
  - Optimization Type ◆ Max Cap ◆ Max Tran ◆ Max Fanout
- ✓ **Timing analysis (Hold)**
  - Open *Timing* → *Report Timing…*
  - Design Stage ◆ Post-Route
  - Analysis Type ◆ Hold
- ✓ **Optimization (Hold)**
  - Open *Optimize* → *Optimize Design…*
  - Optimization Type ◆ Setup ◆ Hold ◆ Max Cap ◆ Max Tran ◆ Max Fanout

12. Add Filler Cells

- **Add filler cells**
  - Fill all the gaps between standard cell instances
  - Provide decoupling capacitances to complete connections in the standard cell rows

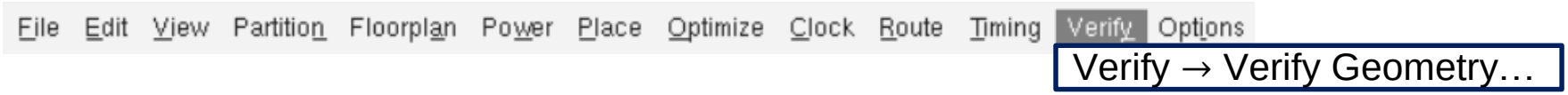


- **Metal filler inserted after routing, but before GDSII stream out**

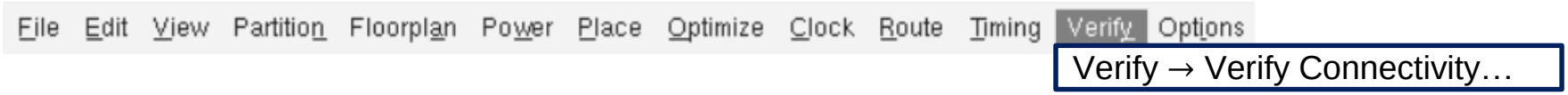
Stream Out

13. Before Output: Verification

- ✓ **Verify geometry**
  - Check if the layout so far does not violate technology design rules



- ✓ **Verify wire connectivity**
  - Check if potential flaws with interconnects and pins exist



- ✓ **Calibre DRC**
  - Design rule check with more rigorous rules
  - For advanced process, the verifications provided by encounter may be insufficient, so we must check Calibre DRC

14. Stream Out Files

- ✓ **The following files will be generated after APR design flow**
  - CHIP_pr.v
    - Netlist file for post-layout gate-level simulation
  - CHIP.sdf
    - Design timing file for post-layout gate-level simulation
  - CHIP.def
    - Design exchange format relevant to physical layout
  - CHIP.gds
    - GDSII stream file for Calibre-DRC (Design Rule Check), LVS, and tape out