National Tsing Hua University
Department of Electrical Engineering
EE429200 IC Design Laboratory, Fall 2021

Lab 10: APR Flow with IC Compiler (Design Setup and Floorplan)

Assigned on Nov. 18, 2021

Due day in Nov 25, 2021

Objective

In this lab, you will learn how to use IC Compiler to perform a basic APR flow.

Demo checklist:

- ☐ Congestion map.
- ☐ Hierarchy visual mode.
- ☐ Questions:
  In coarse placement (floorplan stage), which constraint is more important, timing or congestion? Why?

Environment Setup

1. Copy lab file packages from ***~ee4292/***. Decompress the package and enter it.
   ```
   $ cp ~ee4292/iclab2021/lab10.zip .
   $ unzip lab10.zip
   $ cd lab10/lab10/
   $ cd icc/
   ```

   Note: Please run IC compiler in the ***icc_run/*** directory.

Data Preparation (already prepared)

Files	**Location**
Gate Level Netlist	./icc/pre_layout/design_data/CHIP_syn.v (generated later)
IO Constraint File	./icc/pre_layout/design_data/io_pin.tdf
Timing Constraint File	./icc/pre_layout/design_data/CHIP_syn.sdc
Technology File	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/milkyway/saed32nm_1p9m_mw.tf
Layer Mapping File	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/milkyway/saed32nm_1p9m_gdsout_mw.map
Milkyway library (Core)	/usr/cadtool/cad/synopsys/SAED32_EDK/lib/stdcell_hvt/milkyway/saed32nm_hvt_1p9m
TLU+ File	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/star_rcxt/saed32nm_1p9m_Cmax.tluplus

	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/star_rcxt/saed32nm_1p9m_Cmin.tluplus
TLU+ Mapping File	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/star_rcxt/saed32nm_tf_itf_tluplus.map

Action Items

I. Design Setup.

1. Wrap the synthesized netlist in ***icc/pre_layout/design_data/***.

   $ cd pre_layout/design_data/

   $ sh get_pnr_netlist.bat

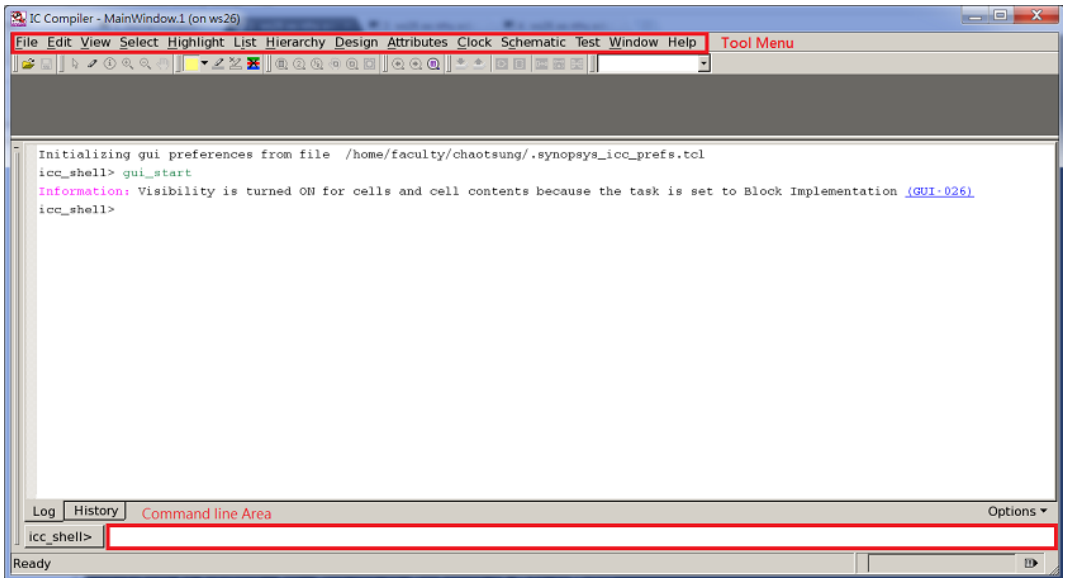
2. In the ***icc_run/***, open the GUI of ICC by:

   $ cd ../../icc_run/

   $ icc_shell -64 -gui

   Set link library and target library:

   $ source setup.tcl



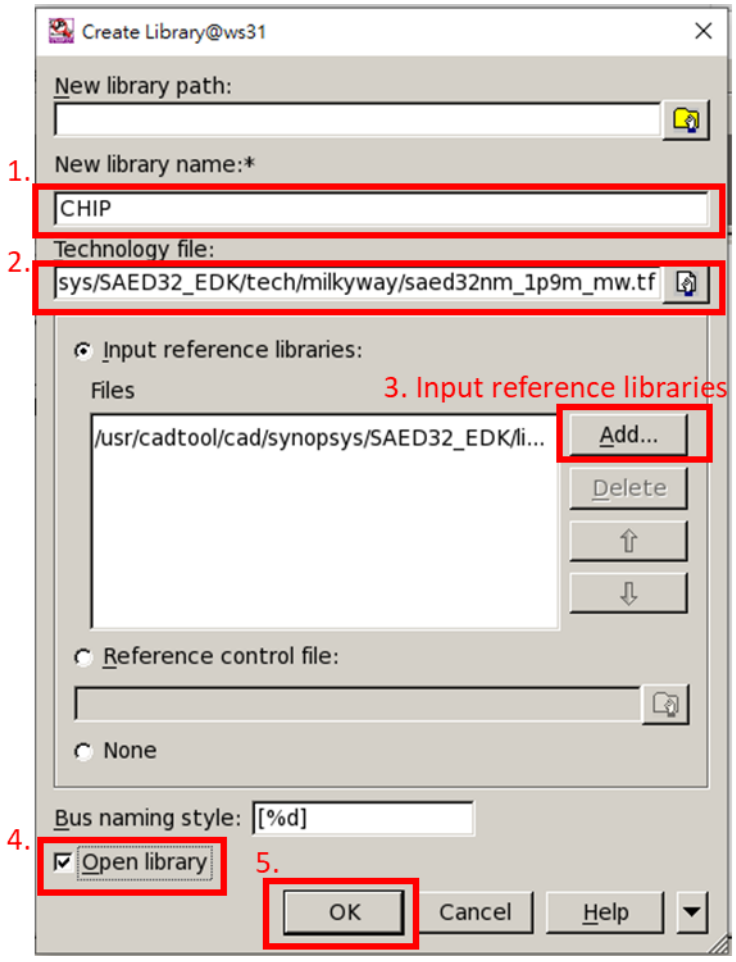
Note: **(Optional)** If you can't open ICC, check the .tcshrc file in your home directory and add the following command:

```
source /usr/cad/synopsys/CIC/icc.csh
```

3. Create a new library.

“File->Create Library”

New Library Name	CHIP
Technology file	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/milkyway/saed32nm_1p9m_mw.tf
Input reference libraries	/usr/cadtool/cad/synopsys/SAED32_EDK/lib/stdcell_hvt/milkyway/saed32nm_hvt_1p9m
Open library	Enable
Others	default

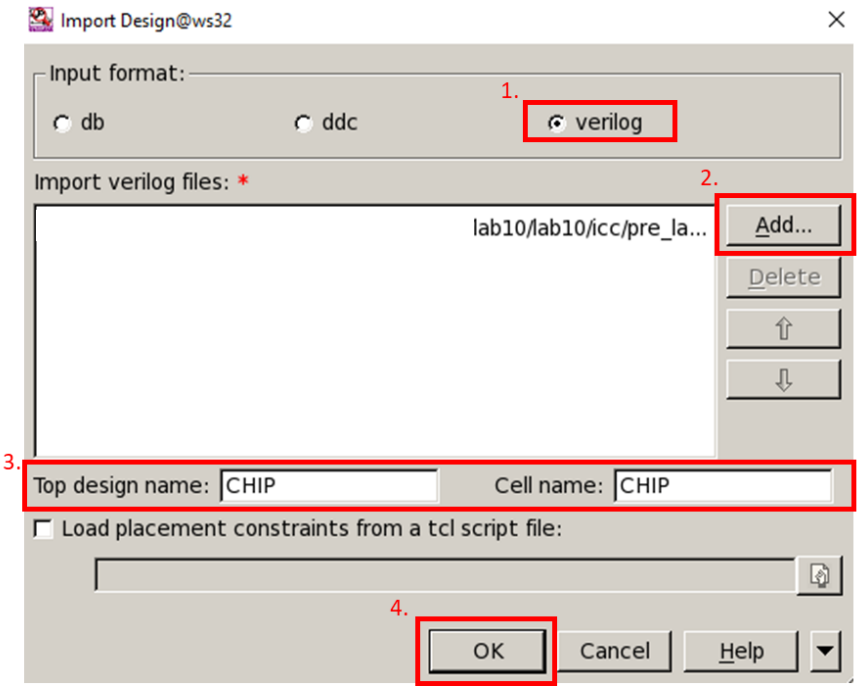


4. Import your design.

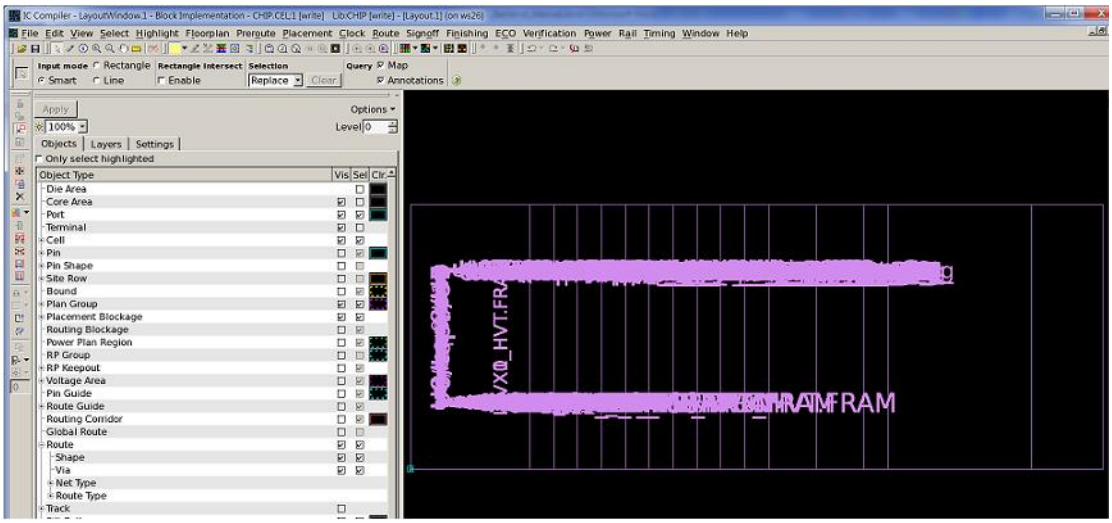
“File->Import Designs”

Input Format	verilog

Import Verilog Files	../pre_layout/design_data/CHIP_syn.v
Top design name	CHIP
Cell name	CHIP



It may take a while. Then the layout window will be opened.



5. Read Synopsys Design Constraint (SDC) file.

"File->Import->Read SDC…"

Input File Name	../pre_layout/design_data/CHIP_syn.sdc
Version	1.9
Other	default

Note: Please take a look at */design_data/CHIP_syn.sdc* to find the difference from the */dc/top_pipe_syn.sdc* which is generated by DC. The clock period is relaxed to 3.8ns (from 3.2ns in DC), and besides the max library for setup time checking, the min library for hold time checking is also identified.

6. TLU+ model setup.

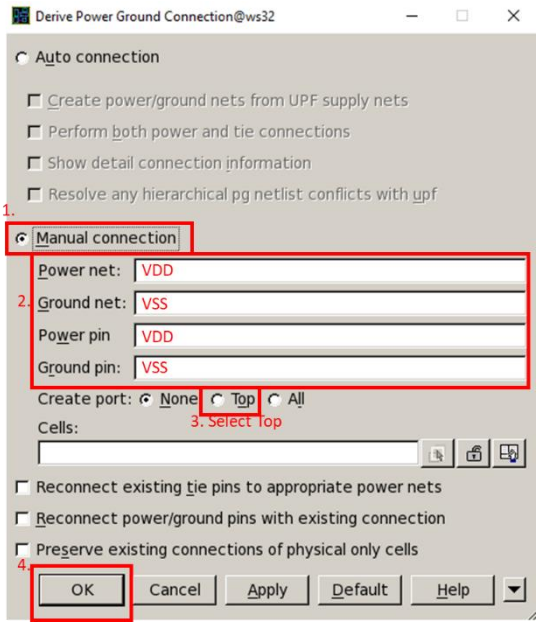
   "File->Set TLU+"

Max TLU+ File	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/star_rcxt/saed32nm_1p9m_Cmax.tluplus
Min TLU+ File	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/star_rcxt/saed32nm_1p9m_Cmin.tluplus
Layer name mapping file between technology library and ITF file	/usr/cadtool/cad/synopsys/SAED32_EDK/tech/star_rcxt/saed32nm_tf_itf_tluplus.map

7. Power/Ground connection.

   "Preroute->Derive PG Connection" (in Layout window)

Manual connection	Selected
Power net	VDD
Ground net	VSS
Power pin	VDD
Ground pin	VSS
Create port	Top



Note: This step connects (logically, not physically routed yet) power and

ground for all the nets or pins with names VDD and VSS respectively.

8. Save your design.

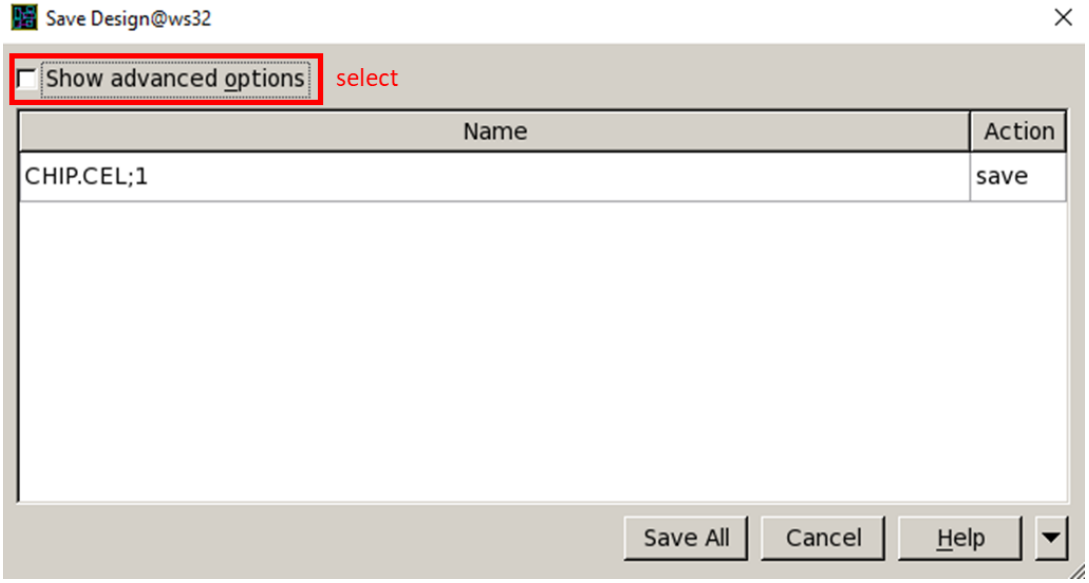
   "File->Save Design" and click **"Save all."** Or you can save your design with command line interface.

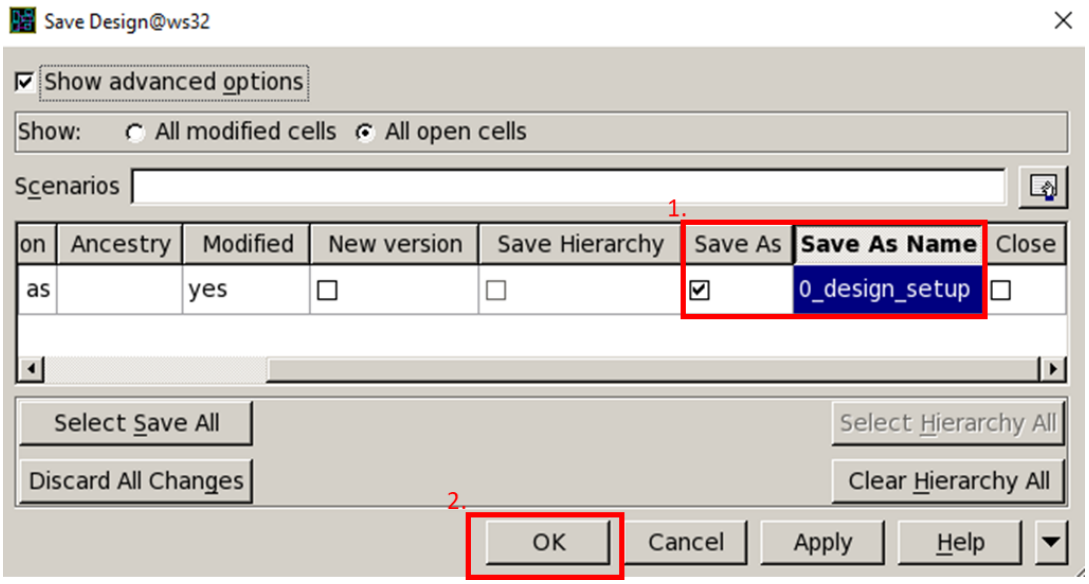
   $ save_mw_cel CHIP

9. Save your design.

   "File->Save Design" and check **"Show advanced options"**

Save as	Enable
Save as name	0_design_setup





Or you can save your design with command line interface.

```
$ save_mw_cel -as 0_design_setup
```

Note: Remember to save your design frequently such that you can go back to a clean stage when something wrong happens.

II. Floorplan

1. Read I/O physical constraint.

"Floorplan->Read Pin/Pad Physical Constraints"

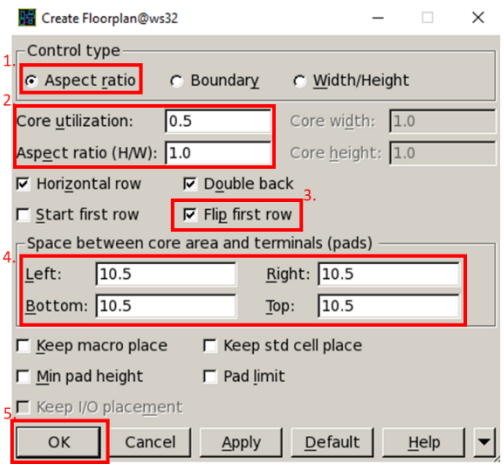
Input file name	../pre_layout/design_data/io_pin.tdf

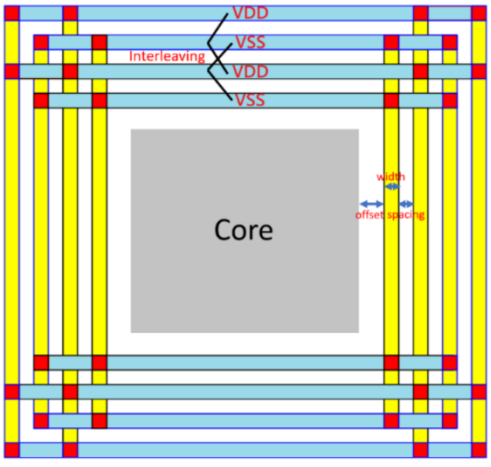
Note: You will need to prepare IO arrangement for your design in final project.

Note: IO placement may affect timing performance and routing complexity. Reference the lecture for more details.

2. Floorplan.

"Floorplan->Create Floorplan"





Note: "**Core to xxx**" means the space we preserve for core ring. The space should be large enough for offset, space and width of core rings, and should be as small as possible for area concern.

Note: Here we choose 10.5 for {offset + ring width + space + ring width + offset} = {1 + 4 + 0.5 + 4 + 1}

Note: The core utilization can be set up to 70~80% normally. Here we set 50%

only for faster P&R for the lab. The core margins are reserved for the core power ring.

3. Identify clock-gating for further P&R.

   Execute the following three commands in the command line area sequentially:

   $ identify_clock_gating

   $ report_clock_gating

   Switch to Main Window to find:

   Number of Gated Registers =_____? Number of Ungated Registers =_____?

   Note: Executing $ man report_clock_gating can invoke the "**Man Page Viewer**" which shows the details of this instruction. Obviously applicable to all commands.

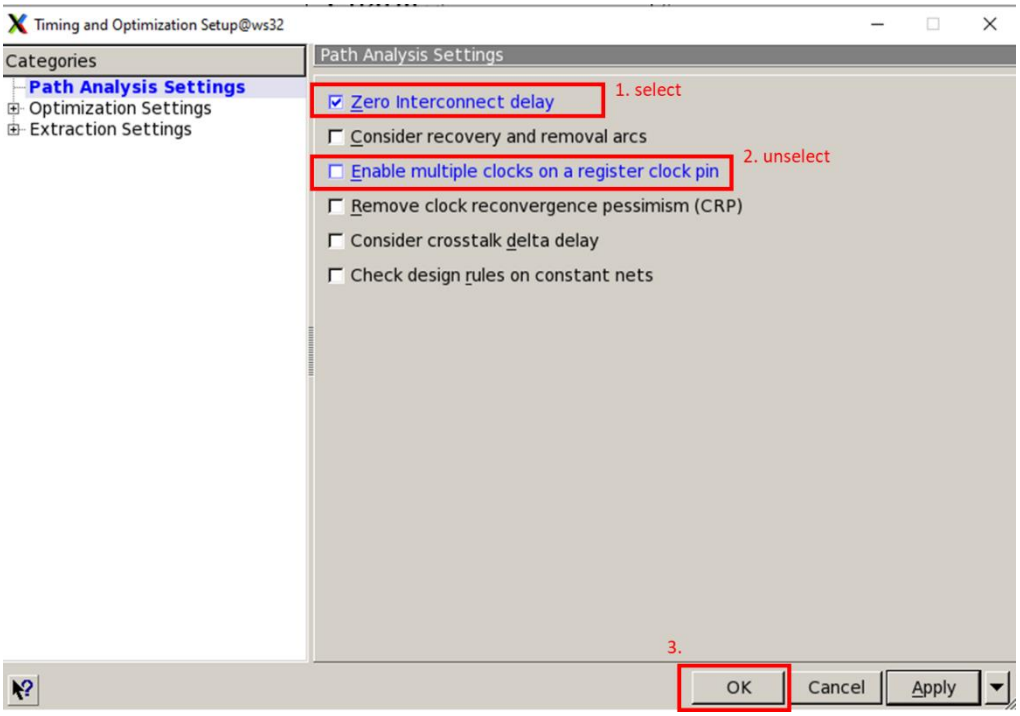
4. Virtual flat placement.

   Execute the following command in the Command Line:

   $ create_fp_placement -timing_driven

5. Check the timing report.

   "Timing>Timing and Optimization Setup…"



Then execute the following command for timing analysis.

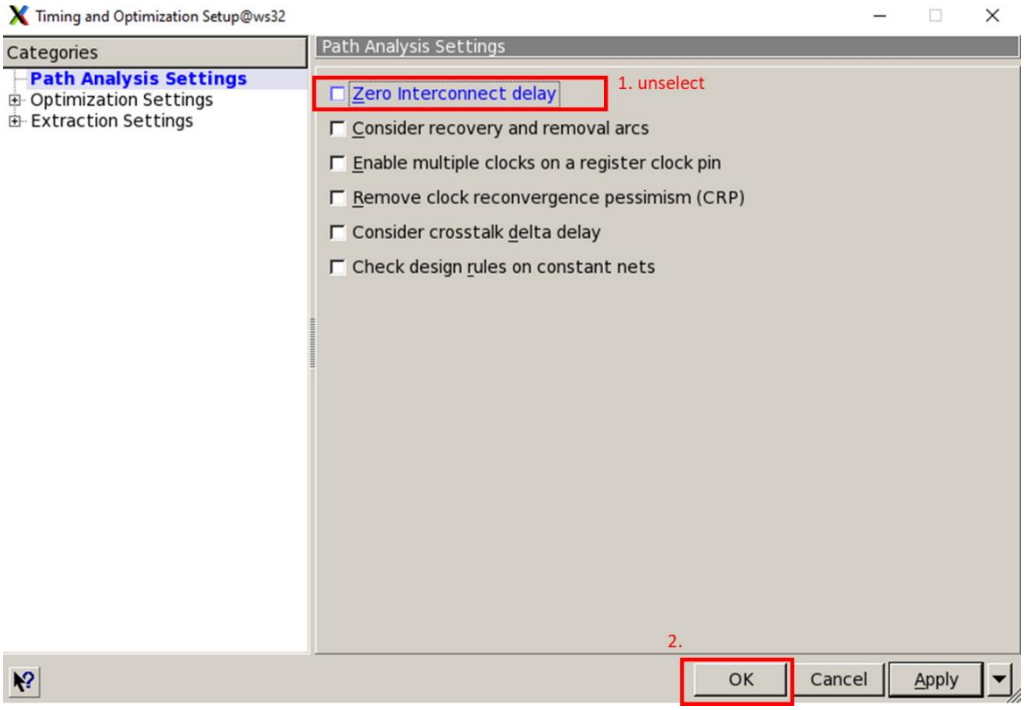
$ report_timing

The worst timing slack = _____?

Note: This step assumes there is no net delay, so the timing analysis should pass.

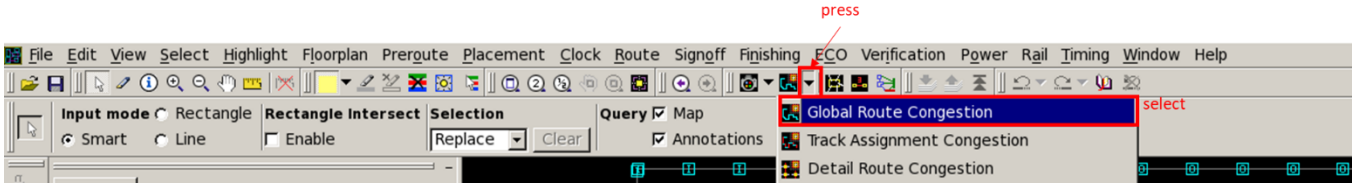
6. Set the net delay back to false.

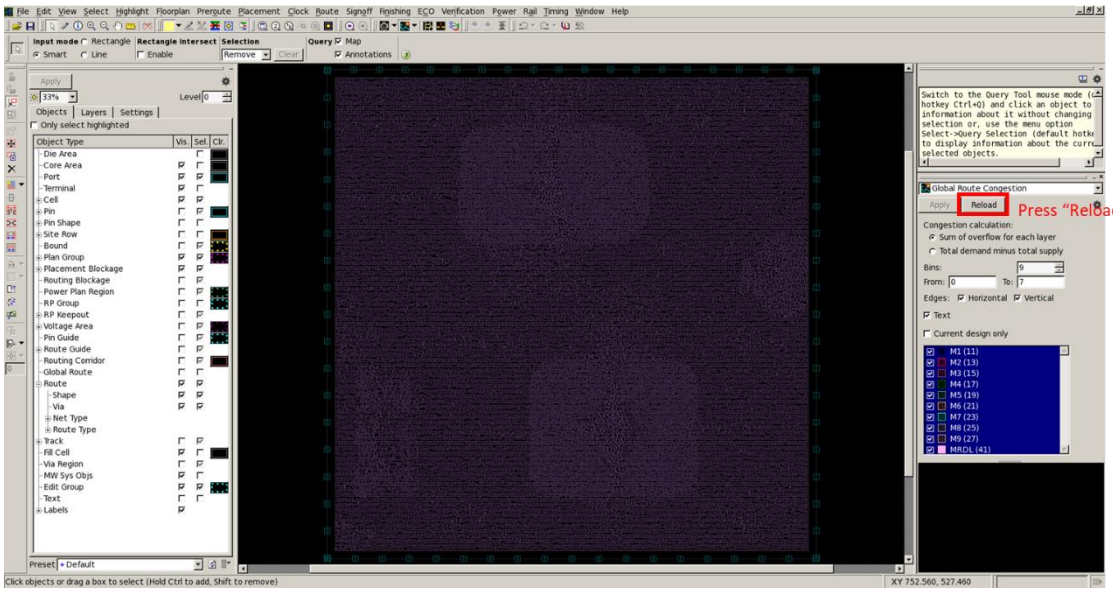
"Timing>Timing and Optimization Setup…"



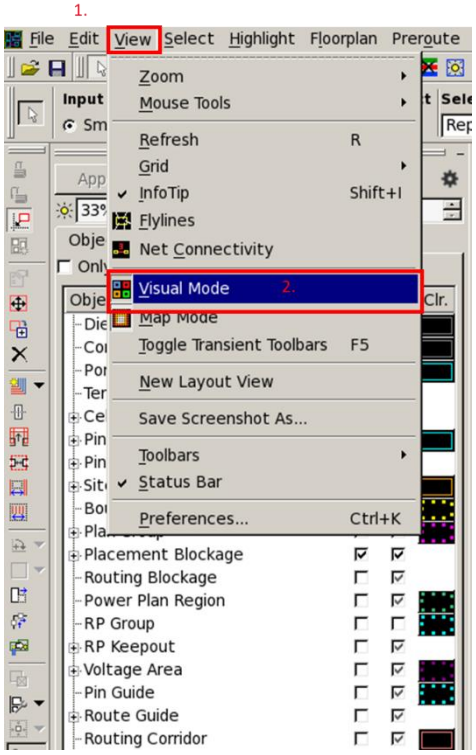
7. Analyze congestion.

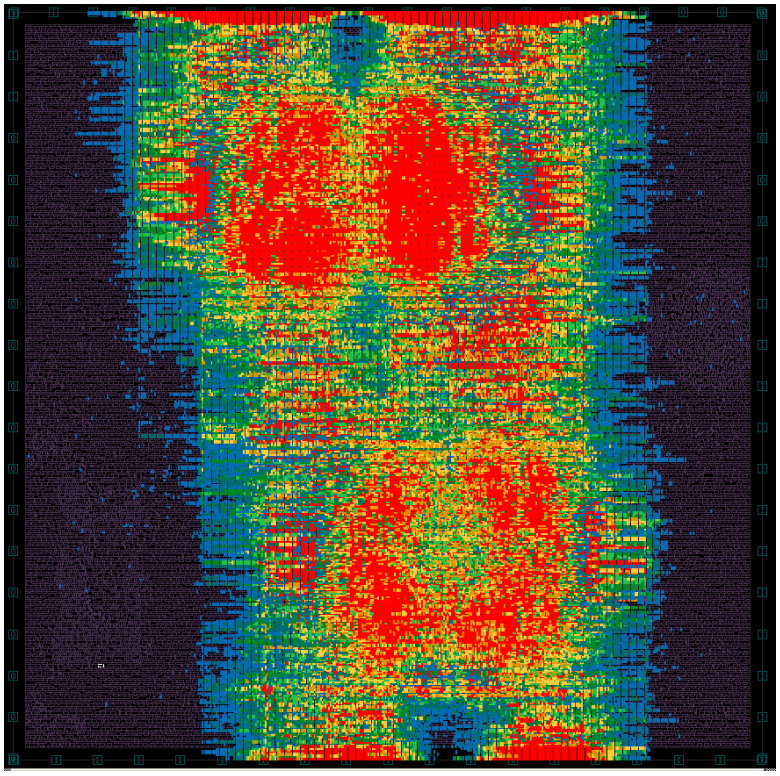
Open the **Global Route Congestion menu** and then click **"Reload"** on the right hand side. (This step may take a few minutes)

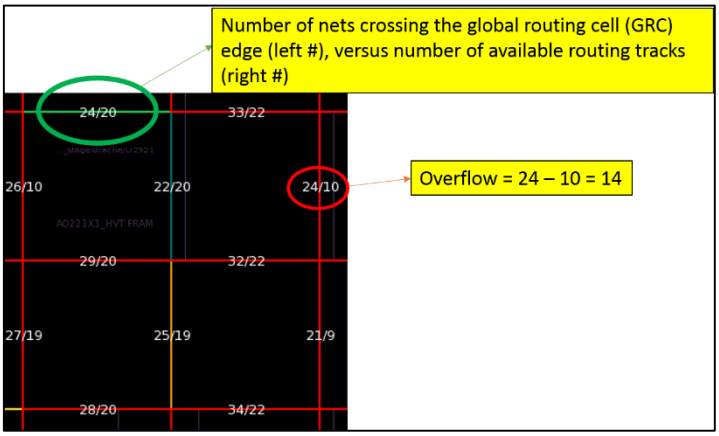




“View>Visual Mode” to view congestion map







Obviously, the design is **congested**.

8. Congestion can be serious or unacceptable if one or more of the following are true:

   (1) Large or many "**hotspots**" in the congestion map. (Serious routability challenge.)

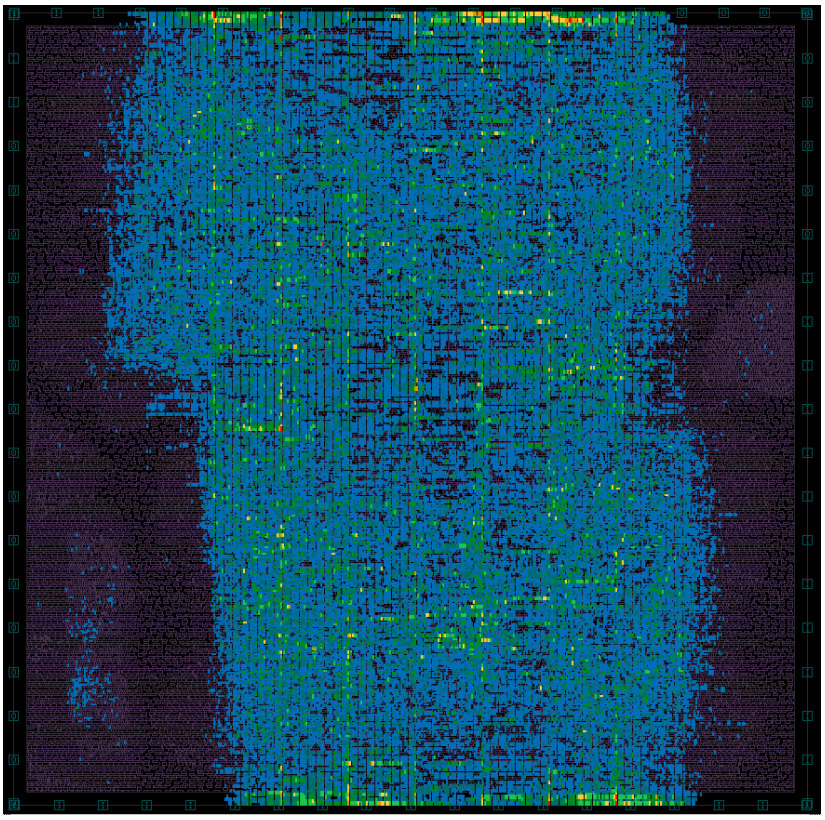
   (2) Any GRC (global routing cell) "**overflow**" larger than around 10%. (Possible non-routable nets)

   (3) About 2%, or more, GRC edges have an overflow. (Possible signal integrity or timing degradation issues)

   In order to reduce congestion, execute the following command:

```
$ create_fp_placement -congestion_driven
```

Then, check the congestion map again (reload congestion map again).



9. **(Optinoal)** If the design is still congested, execute the following command:

```
$ create_fp_placement -congestion_driven -incremental all
```

Or you have to modify the floorplan, which means you should reduce the **utilization.** In this lab, we choose utilization=0.5 to prevent congestion.

10. Save your design.

"File->Save Design" and click **"Save all."**

Or you can save your design with command line interface.

```
$ save_mw_cel CHIP
```

11. Save your design.

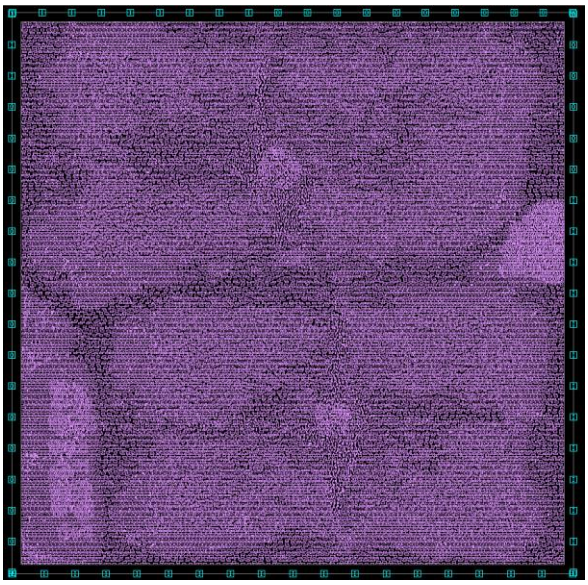
"File->Save Design" and check **"Show advanced options"**

Save as	Enable
Save as name	1_floorplan

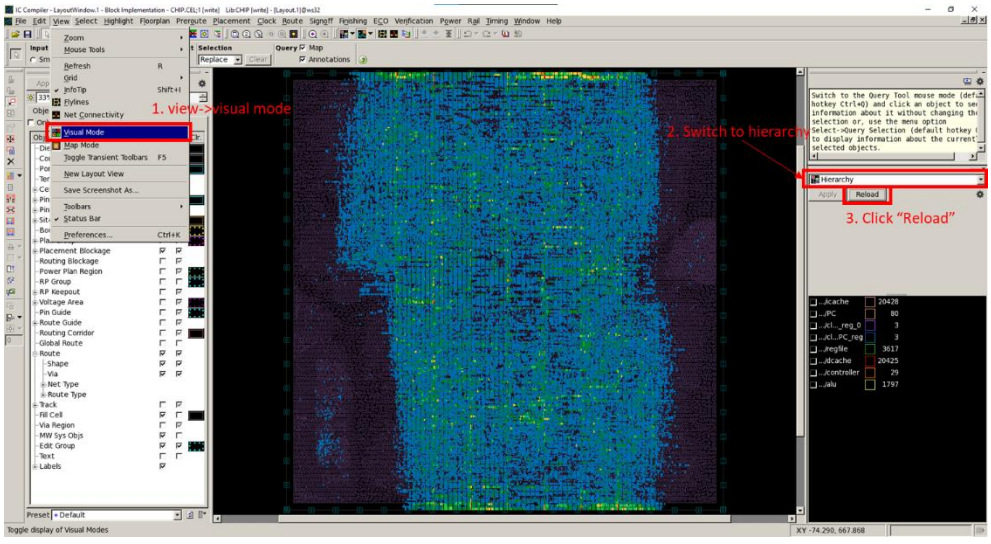
Or you can save your design with command line interface.

```
$ save_mw_cel -as 1_floorplan
```

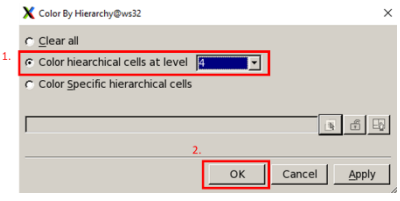
12. You can see the floorplan and flat placement result in Layout Window where the core area for standard cells is surrounded by the pins.



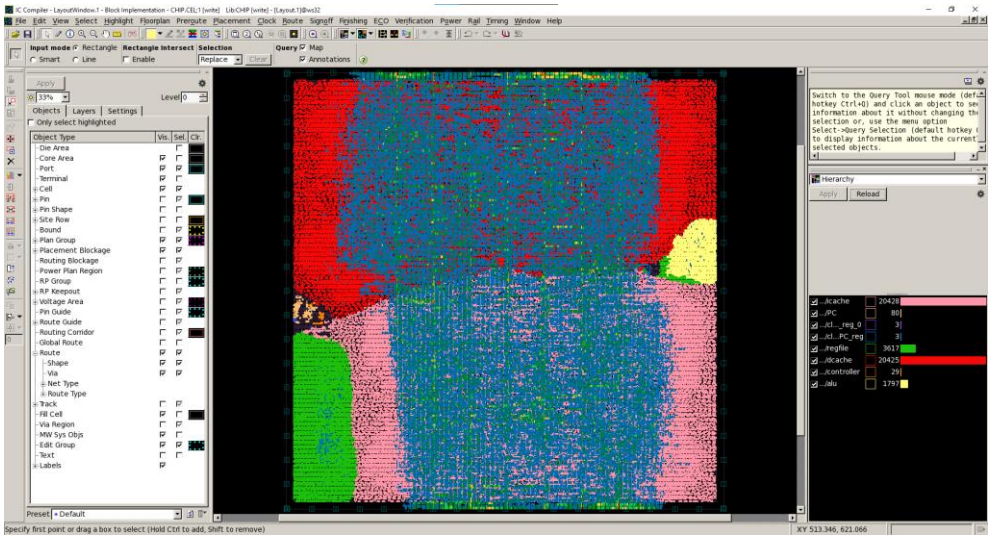
We can find out the layout partition by first selecting **"View->Visual Mode"** and switch the option on the pop-up window on the right from **"RPNet Connection"** to **"Hierarchy."** Then click **"Reload"**.



And at pop-up window set color hierarchical cells at level **4**



Press “F” on the keyboard to show the full layout. Then you can see the hierarchical partition:



Selecting **“View->Visual Mode”** again can switch back to the default mode.

13. Close ICC.

    “File->Close Design,” “File->Close Library,” and then type **exit** in the icc_shell terminal **(make sure to close library before you leave)**

Appendix

Here is the file list of the lab package. Please check if anything missing!

Directory	**Filename**	**Description**
icc/icc_run	setup.tcl	Environment setup
icc/pre_layout/dc	top_pipe_syn.sdc	Synthesis SDC
icc/pre_layout/dc	top_pipe_syn.v	Synthesized
icc/pre_layout/design_data	add_tie.tcl	Tie cells insertion
icc/pre_layout/design_data	addCoreFiller.tcl	Core filler insertion
icc/pre_layout/design_data	CHIP_syn.f	Filelist for simulation
icc/pre_layout/design_data	CHIP_syn.sdc	SDC for ICC
icc/pre_layout/design_data	CHIP_syn_app.v	Netlist wrapper
icc/pre_layout/design_data	get_pnr_netlist.bat	Batch file for merging netlist
icc/pre_layout/design_data	io_pin.tdf	Pin Constraint
sim	instruction.txt	CPU simulation pattern
sim	presim.f	Filelist for pre-simulation
sim	test_top.v	Testbench for the CPU

sim	top.f	Filelist for the CPU
source	top.v	Top module for this CPU
source	top_pipe.v	Pipelined design for this CPU
source	ID_stage.v	RTL for ID stage
source	IF_stage.v	RTL for IF stage
source	controller.v	RTL for control signal
source	regfile.v	RTL for register file
source	ID_EXE.v	RTL for ID to EXE flip flop
source	EXE_stage.v	RTL for EXE stage
source	alu.v	RTL for alu module
source	CPU_define.v	Definition file
source	dsram.v	RTL for memory
source	PC.v	RTL for program counter
source	IF_ID.v	RTL for IF to ID flip flop
syn	.synopsys_dc.setup	DC setup file
syn	*.tcl	Synthesis scripts
syn	*.log	Log file
syn	run_dc.bat	Batch file for running synthesis
syn/report	*.out	Report files
syn/report	*.log	Log files
syn/netlist	top_pipe_syn.ddc	Netlist & constraints can be read by DC
syn/netlist	top_pipe_syn.saif	Switching activity interchange format (for power)
syn/netlist	top_pipe_syn.sdc	Synopsys design constrains format
syn/netlist	top_pipe_syn.sdf	Standard delay format
syn/netlist	top_pipe_syn.v	Netlist in verilog