<SOC Encounter>
LAB

Aug. 2008 TW
Nov. 2010 CS
Dec. 2011 CS
Jul. 2012 CS & KT

- After the synthesis, you should have files as follows:

LAB File List for Front-End Design		
Folder	**Name**	**Description**
MEM	**SRAM_SP_ADV.ps**	**Post script file**
	SRAM_SP_ADV.spec	**RAM specification**
	SRAM_SP_ADV.v	**Verilog model**
	SRAM_SP_ADV.vclef	**Physical geometry file**
	SRAM_SP_ADV_ff_1.1_-40.0_syn.lib	**Timing files for SOC encounter**
	SRAM_SP_ADV_ff_1.1_.0_syn.lib	
	SRAM_SP_ADV_ss_0.9_125.0_syn.lib	
	SRAM_SP_ADV_tt_1.0_25.0_syn.lib	
	SRAM_SP_ADV_ff_1.1_-40.0_syn.db	**Timing files for Design Compiler**
	SRAM_SP_ADV_ff_1.1_.0_syn.db	
	SRAM_SP_ADV_ss_0.9_125.0_syn.db	
	SRAM_SP_ADV_tt_1.0_25.0_syn.db	
RTL	**core.v top_lv_bisr.v lv*.v**	**Verilog netlist**
		Test bench
		Signal configuration file
		Waveform aliasing file
	SRAM_SP_ADV.v	**Verilog model**
GTL	**run.script**	**Synthesis script file**
	core.v top_lv_bisr.v lv*.v	**Verilog netlist**
	.synopsys_dc.setup	**Design compiler setup file**
		Test bench
	core.sdc	**Timing constraint file**
	core.sdf	**Gate-level timing file**
	core.spf	**Scan chain configuration**

	core.vg	Gate-level netlist
GTL/ SIMULATION	tsmc090.v	Verilog model of standard cells
	core.vg	Gate-level netlist
	core.sdf	Gate-level timing file
	SRAM_SP_ADV.v	Verilog model
		Test bench

LAB File List for Back-End Design		
Folder	Name	Description
SOC/lef	antenna.lef	LEF file for antenna rules
	tpzn90gv3_9lm.lef	LEF file of IO PAD
	tsmc090lk_9lm_2thick_tech.lef	LEF file of standard cells
	tsmc090nvt_macros.lef	LEF file for antenna rules
SOC/lib	fast.lib	Fast timing library for standard cells
	slow.lib	Slow timing library for standard cells
	typical.lib	Common timing library for standard cells
	tpzn90gv3bc.lib	Fast timing library for IO PAD
	tpzn90gv3tc.lib	Common timing library for IO PAD
	tpzn90gv3wc.lib	Slow timing library for IO PAD
SOC/ library.cl	….	QX libraries
SOC	addIoFiller.cmd	Add IO filler script
	addbonding_v3.6.pl	Add bonding PAD script
	icecaps.tch	FireIce capacitance file
	t90g_rcb.CapTbl t90g_rct.CapTbl t90g_rcw.CapTbl	Capacitance table

	io.list	IOPAD list file
ADD_LVS	0.18um_Virtuoso4.4.tf	Technology file
	tsmc18_core.gds	Layout of standard cells
	tsmc18_io_final.gds3	Layout of IO PADs
	t18abs2lay.ile	-
	display.drf	Display file for TSMC 18
	T18drc_13a25a.drc	DRC command file
DRC	T18drc_13a25a.drc	DRC command file
LVS	Caliber-lvs-cur_soce	LVS command file
	tsmc18_lvs.spi	SPICE model for standard cells
	tsmc18_lvs.v	Verilog model for standard cells
POSIM	CHIP.cfg	Nanosim configuration
	CHIP.vec	Po-sim Test pattern
PATTERN _GEN	-	-

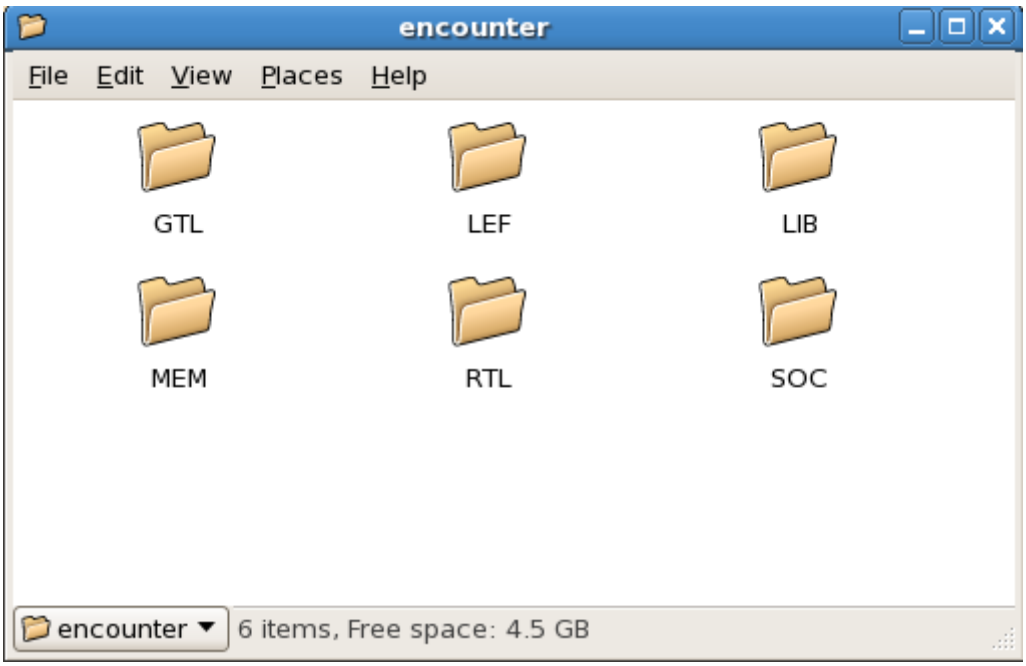


Fig. LAB Folders

- Preparations before APR:
  - Goto [SOC]

- Copy <core.vg> and <core.sdc> from [GTL] to [SOC]
- Goto [SOC/lef]
- Copy <SRAM_SP_ADV.vclef> from [MEM] to [SOC/LEF]
- Goto [SOC/LIB]
- Copy <*.lib> from [MEM] to [SOC/LIB]
- Edit the file <CHIP.vg> → chip-level netlist
- Edit the file <CHIP.sdc> → chip-level timing constraint file
- Edit the file <CHIP.ioc> → IO pad assignment

- Open <core.vg>, then save as <CHIP.vg>. Then, add the following content in the last of file:

==========

TSMC 90: IO pad module names are as follow
input IO: PDIDGZ_33
output IO: PDO24CDG_33
|
是英文

For example:
PDIDGZ_33 PAD_clk (.PAD(I_clk), .C(clk));
//"I_clk" is from external signal to IO PAD
//"clk" is from IO PAD to internal signal

PDO24CDG_33 PAD_out (.PAD(I_out), .I(out));
//"I_out" is from IO PAD to external signal
//"out" is from internal signal to IO PAD

==========

```
module CPU_CHIP(
I_clk,
I_rst,
I_control,
I_memaddr,
I_in,
O_out,
I_se,
I_si,
I_scantest,
O_so,
```

```
I_tpclk,
I_test_si2,
O_test_so2);

  //INPUT 27; OUTPUT 10 (TOTAL:37)
  input [3:0] I_control;
  input [7:0] I_memaddr;
  input [7:0] I_in;
  output [7:0] O_out;
  input I_clk, I_rst, I_se, I_si, I_scantest, I_tpclk, I_test_si2;
  output O_so, O_test_so2;

  //INTERCONNECTIONS
  wire [3:0] control;
  wire [7:0] memaddr;
  wire [7:0] in;
  wire [7:0] out;
  wire clk, rst, se, si, scantest, tpclk, test_si2;
  wire so, test_so2;

  //CORE: note that the reference module in the TOP MODULE should be
  //"CALL BY REFERENCE"
  cpu cpu(
  .clk(clk),
  .rst(rst),
  .control(control),
  .memaddr(memaddr),
  .in(in),
  .out(out),
  .se(se),
  .si(si),
  .scantest(scantest),
  .so(so),
  .tpclk(tpclk),
  .test_si2(test_si2),
  .test_so2(test_so2)
  );
```

```
PDIDGZ_33   PAD_clk          (.PAD(I_clk),          .C(clk));   //27 pads
PDIDGZ_33   PAD_rst          (.PAD(I_rst),          .C(rst));
PDIDGZ_33   PAD_control0     (.PAD(I_control[0]),   .C(control[0]));
PDIDGZ_33   PAD_control1     (.PAD(I_control[1]),   .C(control[1]));
PDIDGZ_33   PAD_control2     (.PAD(I_control[2]),   .C(control[2]));
PDIDGZ_33   PAD_control3     (.PAD(I_control[3]),   .C(control[3]));
PDIDGZ_33   PAD_memaddr0     (.PAD(I_memaddr[0]),   .C(memaddr[0]));
PDIDGZ_33   PAD_memaddr1     (.PAD(I_memaddr[1]),   .C(memaddr[1]));
PDIDGZ_33   PAD_memaddr2     (.PAD(I_memaddr[2]),   .C(memaddr[2]));
PDIDGZ_33   PAD_memaddr3     (.PAD(I_memaddr[3]),   .C(memaddr[3]));
PDIDGZ_33   PAD_memaddr4     (.PAD(I_memaddr[4]),   .C(memaddr[4]));
PDIDGZ_33   PAD_memaddr5     (.PAD(I_memaddr[5]),   .C(memaddr[5]));
PDIDGZ_33   PAD_memaddr6     (.PAD(I_memaddr[6]),   .C(memaddr[6]));
PDIDGZ_33   PAD_memaddr7     (.PAD(I_memaddr[7]),   .C(memaddr[7]));
PDIDGZ_33   PAD_in0          (.PAD(I_in[0]),        .C(in[0]));
PDIDGZ_33   PAD_in1          (.PAD(I_in[1]),        .C(in[1]));
PDIDGZ_33   PAD_in2          (.PAD(I_in[2]),        .C(in[2]));
PDIDGZ_33   PAD_in3          (.PAD(I_in[3]),        .C(in[3]));
PDIDGZ_33   PAD_in4          (.PAD(I_in[4]),        .C(in[4]));
PDIDGZ_33   PAD_in5          (.PAD(I_in[5]),        .C(in[5]));
PDIDGZ_33   PAD_in6          (.PAD(I_in[6]),        .C(in[6]));
PDIDGZ_33   PAD_in7          (.PAD(I_in[7]),        .C(in[7]));
PDIDGZ_33   PAD_se           (.PAD(I_se),           .C(se));
PDIDGZ_33   PAD_si           (.PAD(I_si),           .C(si));
PDIDGZ_33   PAD_scantest     (.PAD(I_scantest),     .C(scantest));
PDIDGZ_33   PAD_tpclk        (.PAD(I_tpclk),        .C(tpclk));
PDIDGZ_33   PAD_test_si2     (.PAD(I_test_si2),     .C(test_si2));

PDO24CDG_33 PAD_out0         (.PAD(O_out[0]),       .I(out[0]));   //10 pads
PDO24CDG_33 PAD_out1         (.PAD(O_out[1]),       .I(out[1]));
PDO24CDG_33 PAD_out2         (.PAD(O_out[2]),       .I(out[2]));
PDO24CDG_33 PAD_out3         (.PAD(O_out[3]),       .I(out[3]));
PDO24CDG_33 PAD_out4         (.PAD(O_out[4]),       .I(out[4]));
PDO24CDG_33 PAD_out5         (.PAD(O_out[5]),       .I(out[5]));
PDO24CDG_33 PAD_out6         (.PAD(O_out[6]),       .I(out[6]));
PDO24CDG_33 PAD_out7         (.PAD(O_out[7]),       .I(out[7]));
```

```
PDO24CDG_33 PAD_so          (.PAD(O_so),          .I(so));
PDO24CDG_33 PAD_test_so2    (.PAD(O_test_so2),    .I(test_so2));

endmodule
```

- Open <cpu.sdc>, then save as <CHIP.sdc>. Then, modify it as follows:

```
#######################################################

#   Created by Design Compiler write_sdc on Sun Jul 20 17:09:22 2008

#######################################################
set sdc_version 1.4
current_design CPU_CHIP
create_clock [get_pins {PAD_clk/C}] -name CLK1 -period 20 -waveform {0 10}

set_false_path   -from [get_ports {I_rst}]

set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_scantest}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_si}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_se}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[0]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[1]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[2]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[3]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[4]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[5]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[6]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_in[7]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[0]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[1]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[2]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[3]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[4]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[5]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[6]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_memaddr[7]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_control[0]}]
```

```
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_control[1]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_control[2]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_control[3]}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_rst}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_clk}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_tpclk}]
set_input_delay 0.34 -clock [get_clocks {CLK1}] [get_ports {I_test_si2}]

set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_so}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[0]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[1]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[2]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[3]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[4]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[5]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[6]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_out[7]}]
set_output_delay 1.6 -clock [get_clocks {CLK1}] [get_ports {O_test_so2}]

set_drive 0.288001 [get_ports {I_clk}]
set_drive 0.288001 [get_ports {I_rst}]
set_drive 0.288001 [get_ports {I_control[3]}]
set_drive 0.288001 [get_ports {I_control[2]}]
set_drive 0.288001 [get_ports {I_control[1]}]
set_drive 0.288001 [get_ports {I_control[0]}]
set_drive 0.288001 [get_ports {I_memaddr[7]}]
set_drive 0.288001 [get_ports {I_memaddr[6]}]
set_drive 0.288001 [get_ports {I_memaddr[5]}]
set_drive 0.288001 [get_ports {I_memaddr[4]}]
set_drive 0.288001 [get_ports {I_memaddr[3]}]
set_drive 0.288001 [get_ports {I_memaddr[2]}]
set_drive 0.288001 [get_ports {I_memaddr[1]}]
set_drive 0.288001 [get_ports {I_memaddr[0]}]
set_drive 0.288001 [get_ports {I_in[7]}]
set_drive 0.288001 [get_ports {I_in[6]}]
set_drive 0.288001 [get_ports {I_in[5]}]
set_drive 0.288001 [get_ports {I_in[4]}]
```

```
set_drive 0.288001 [get_ports {I_in[3]}]
set_drive 0.288001 [get_ports {I_in[2]}]
set_drive 0.288001 [get_ports {I_in[1]}]
set_drive 0.288001 [get_ports {I_in[0]}]
set_drive 0.288001 [get_ports {I_se}]
set_drive 0.288001 [get_ports {I_si}]
set_drive 0.288001 [get_ports {I_scantest}]
set_drive 0.288001 [get_ports {I_tpclk}]
set_drive 0.288001 [get_ports {I_test_si2}]

set_load -pin_load 0.06553 [get_ports {O_out[7]}]
set_load -pin_load 0.06553 [get_ports {O_out[6]}]
set_load -pin_load 0.06553 [get_ports {O_out[5]}]
set_load -pin_load 0.06553 [get_ports {O_out[4]}]
set_load -pin_load 0.06553 [get_ports {O_out[3]}]
set_load -pin_load 0.06553 [get_ports {O_out[2]}]
set_load -pin_load 0.06553 [get_ports {O_out[1]}]
set_load -pin_load 0.06553 [get_ports {O_out[0]}]
set_load -pin_load 0.06553 [get_ports {O_so}]
set_load -pin_load 0.06553 [get_ports {O_test_so2}]
```

- Why the modification?
  - A clock source CLK1 is declared representing the external tester clock
  - Only the input delay, input drive, output delay, and output load are remained, since other settings (i.e. wire load model) will be ignored in the SOC Encounter.

- Edit <CHIP.ioc> as follows: (Here a 48-pin package is assumed)

```
Version: 1
Pad: CORNER0          NW   PCORNERDG
Pad: PAD_clk          N
Pad: PAD_rst          N
Pad: PAD_CoreVDD1     N    PVDD1DGZ_33
Pad: PAD_CoreVSS1     N    PVSS1DGZ_33
Pad: PAD_control0     N
Pad: PAD_control1     N
Pad: PAD_control2     N
Pad: PAD_control3     N
```

Pad: PAD_memaddr0	N	
Pad: PAD_IOVDD1	N	PVDD2DGZ_33
Pad: PAD_memaddr1	N	
Pad: PAD_memaddr2	N	
Pad: CORNER1	NE	PCORNERDG
Pad: PAD_memaddr3	E	
Pad: PAD_memaddr4	E	
Pad: PAD_CoreVDD2	E	PVDD1DGZ_33
Pad: PAD_CoreVSS2	E	PVSS1DGZ_33
Pad: PAD_memaddr5	E	
Pad: PAD_memaddr6	E	
Pad: PAD_memaddr7	E	
Pad: PAD_in0	E	
Pad: PAD_in1	E	
Pad: PAD_IOVSS1	E	PVSS2DGZ_33
Pad: PAD_in2	E	
Pad: PAD_in3	E	
Pad: CORNER2	SE	PCORNERDG
Pad: PAD_in4	S	
Pad: PAD_in5	S	
Pad: PAD_CoreVDD3	S	PVDD1DGZ_33
Pad: PAD_CoreVSS3	S	PVSS1DGZ_33
Pad: PAD_in6	S	
Pad: PAD_in7	S	
Pad: PAD_se	S	
Pad: PAD_si	S	
Pad: PAD_scantest	S	
Pad: PAD_IOVDD2	S	PVDD2DGZ_33
Pad: PAD_tpclk	S	
Pad: PAD_test_si2	S	
Pad: CORNER3	SW	PCORNERDG
Pad: PAD_out0	W	
Pad: PAD_out1	W	
Pad: PAD_CoreVDD4	W	PVDD1DGZ_33

Pad: PAD_out2	**W**	
Pad: PAD_out3	**W**	
Pad: PAD_out4	**W**	
Pad: PAD_out5	**W**	
Pad: PAD_out6	**W**	
Pad: PAD_out7	**W**	
Pad: PAD_IOVSS2	**W**	**PVSS2DGZ_33**
Pad: PAD_so	**W**	
Pad: PAD_test_so2	**W**	

Placement and Routing Using <SOC Encounter>

- File preparations:
  - CPU_CHIP.vg – gate-level netlist (i.e. from Design Compiler)
  - CHIP.sdc – definitions of I/O driving, delays, and loadings, …etc (i.e. from Design Compiler, needed to be modified)
  - CHIP.ioc – rearrangement of I/O positions (i.e. user edit)
  - Folder “lib” – library files of the memory, standard cells, and PADs, which define the electronic parameters of the cells (i.e. from Artisan and CIC Design Kit)
  - Folder “lef” – LEF files of the memory, standard cells, which define the technology design rules for P&R (i.e. from Artisan and CIC Design Kit. If you have full-custom design, its lef file is generated by Abstractor)
  - Folder “library.cl” – QX library (i.e. from CIC Design Kit)
  - Others: tsmc018.capTbl (capacitance values for timing evaluation), icecaps_5lm.tch (QX technology file), addIoFiller.cmd (IO filler execution file), addbonding.pl (bonding PAD execution file), ioPad.list (i.e. from CIC Design Kit), streamOut.map (i.e. user edit)

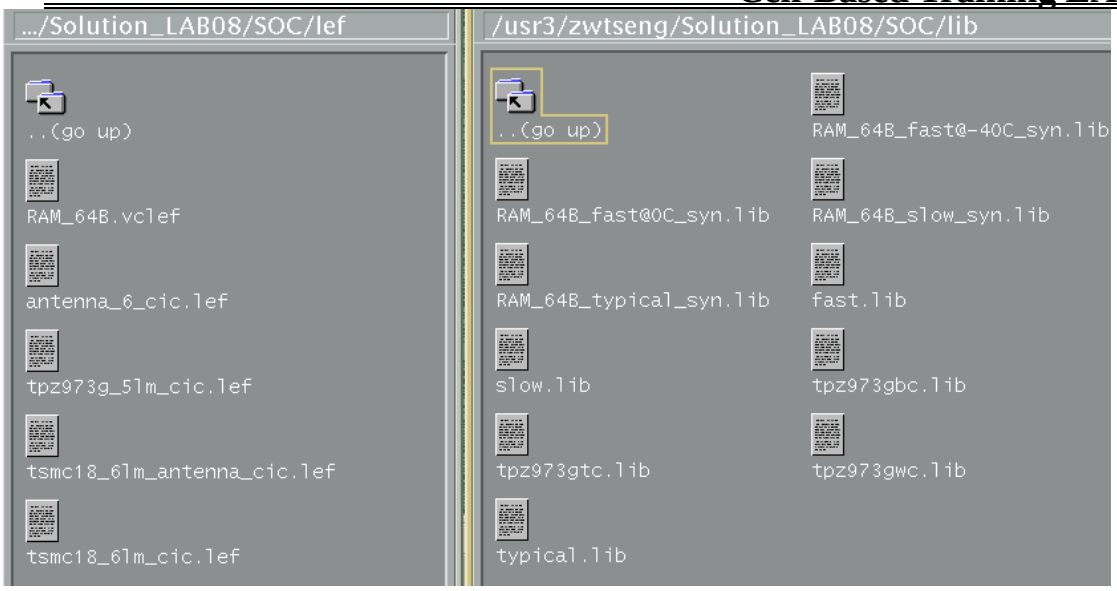


Fig. LIB and LEF files

- encounter ↵ (In the " SOC/ " folder)
- Design → Import Design
  - **<Basic> Tag**
  - Files: CPU_CHIP.vg
  - Top Cell: CPU_CHIP (By User)
  - LEF Files:
    - select the following files and press "Add" button (In " SOC/lef/ ")
      - tsmc090lk_9lm_2thick_tech.lef (should be selected first)
        - tsmc090lk_9lm_2thick_tech.lef
        - tsmc090nvt_macros.lef
      - tpzn90gv3_9lm.lef
        - tpzn90gv3_9lm.lef
        - antenna.lef
      - tpbn90gv_9lm.lef

    - Then, revise the "*.lef" into "all files" in the blank of Filter↵

    - Then, select the following files in the same way:
      - SRAM_SP_ADV.vclef

  - IO Assignment File: CHIP.ioc
  - **< Advanced/Power> Tag**
    - Power Nets: VDD

- Ground Nets: VSS
- Save… → CHIP.conf↵
- Exit from soc encounter GUI interface

==========Write CHIP.conf==========

0. vi CHIP.conf

>>>Please find the following line and modify<<<

1. ~~set rda_Input(ui_timingcon_file,full) “”~~ Not this line

   set rda_Input(ui_timingcon_file) “”

- Timing Constraint File: CHIP.sdc
- set rda_Input(ui_timingcon_file) “CHIP.sdc”

2. set rda_Input(ui_timelib,max) “”

*Please check the library names of the timing libraries of SRAM in tt ss ff corners. If the library names are the same please rename these timing libraries.

- Max Timing Libraries:
  - (In “ SOC/LIB/ “)
    - LEF/slow.lib
    - LEF/tpzn90gv3wc.lib
    - LEF/SRAM_SP_AVD_ss_0.9_125.0_syn.lib
- set rda_Input(ui_timelib,max) “LEF/slow.lib LEF/tpzn90gv3wc.lib LEF/SRAM_SP_AVD_ss_0.9_125.0_syn.lib ”

3. set rda_Input(ui_timelib,min) “”

- Min Timing Libraries:
  - (In “ SOC/LIB/ “)
    - LEF/fast.lib
    - LEF/tpzn90gv3bc.lib
    - LEF/SRAM_SP_AVD_ff_1.1_-40.0_syn.lib
    - LEF/SRAM_SP_AVD_ff_1.1_.0_syn.lib
- set rda_Input(ui_timelib,min) “LEF/fast.lib LEF/tpzn90gv3bc.lib LEF/SRAM_SP_AVD_ss_1.1_-40.0_syn.lib LEF/SRAM_SP_AVD_ss_1.1_.0_syn.lib”

4. set rda_Input(ui_timelib) “”

- Common Timing Libraries:
  - (In “ SOC/LIB/ ”)

- LEF/typical.lib
- LEF/tpzn90gv3tc.lib
- LEF/SRAM_SP_AVD_t t_1.0_25.0_syn.lib

➢ set rda_Input(ui_timelib) “LEF/typical.lib LEF/tpzn90gv3tc.lib LEF/SRAM_SP_AVD_tt_1.0_25.0_syn.lib”

5. set rda_Input(ui_captbl_cap) “”

■ capacitor libraries:

◆ (In “SOC”)

- t90g_rct.CapTbl
- t90g_rcb.CapTbl
- t90g_rcw.CapTbl

➢ set rda_Input(ui_captbl_cap) “ –typical t90g_rct.CapTbl -best t90g_rcb.CapTbl -worst t90g_rcw.CapTbl”

6. set rda_Input(ui_qxtech_file) ””

■ QX Technique file

◆ (In “SOC”)

- icecaps.tch

➢ set rda_Input(ui_qxtech_file) ”icecaps.tch”

7. set rda_Input(ui_qxlib_file) ””

- QX Technique Direction

➢ set rda_Input(ui_qxlib_file) ”$cwd/library”

■ OK↵

- After a period of parsing time, you can see the initial window as following Fig. .
- Open <encounter.log>, search the key word “skipped”. Make sure that all the skipped counts of time constraints are 0.

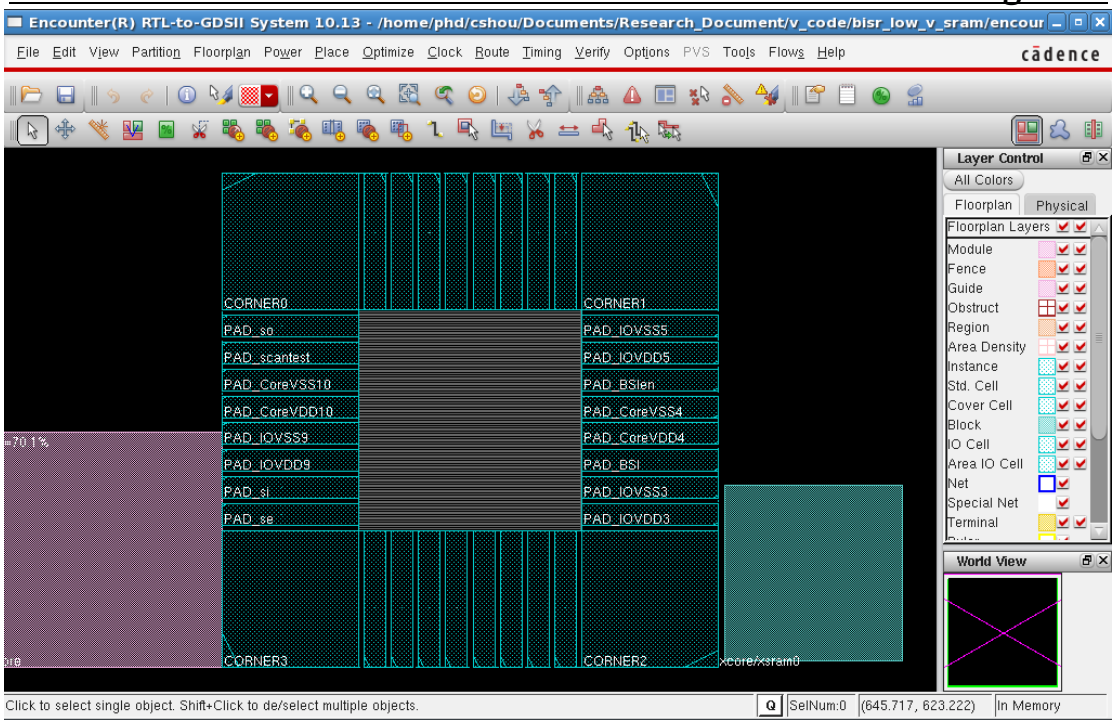
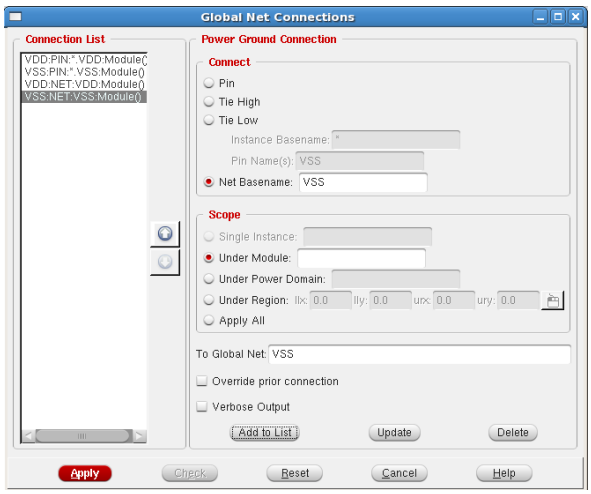


Fig. Initial wondow

- Power → Connect Global Nets



- Select “Pin” and type “VDD” in “Pin Name(s)”, then type “VDD” in “To Global Net” and press “Add to List”
- Select “Pin” and type “VSS” in “Pin Name(s)”, then type “VSS” in “To Global Net” and press “Add to List”

- Select "**Net Basename**" and type in "VDD", then type "VDD" in "To Global Net" and press "Add to List"
- Select "**Net Basename**" and type in "VSS", then type "VSS" in "To Global Net" and press "Add to List"
- Apply → Check → Close (☒)
- Since the Tie High and Tie Low cells will be handled later, the following warning messages are shown.

```
encounter 1> Warning: term EMA[2] of inst xcore/xsram0 is not connect to global special net.
Warning: term EMA[1] of inst xcore/xsram0 is not connect to global special net.
Warning: term EMA[0] of inst xcore/xsram0 is not connect to global special net.
```

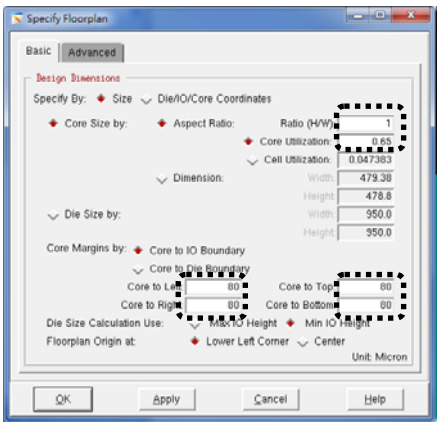
Fig. Warning messages

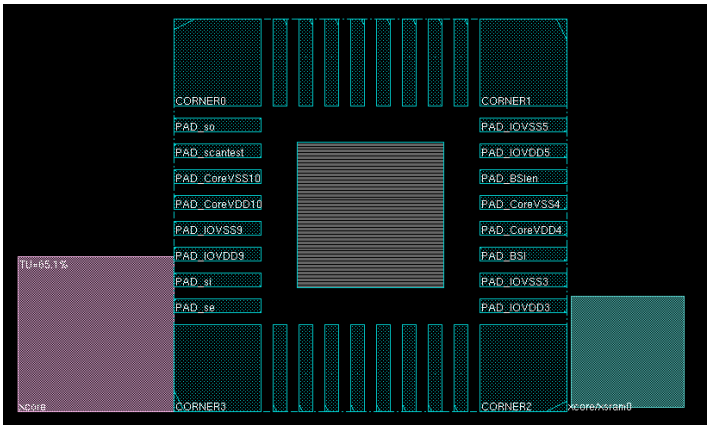
- **In the "consol"**
  - encounter> specifyScanChain scan1 –start PAD_si/C –stop PAD_so/I
  - encounter> specifyScanChain scan2 –start PAD_test_si2/C –stop PAD_test_so2/I (i.e. There are 2 scan chain in my design)
  - encounter> scantrace↵ (i.e. scan chain tracing)

```
encounter 3> scantrace
Tracing scan chain: scan1
Successfully traced scan group scan1 (25 elements; 24 scan bits).
Tracing scan chain: scan2
Successfully traced scan group scan2 (29 elements; 28 scan bits).
*** Scan Trace Summary:
Successfully traced scan group scan1 (25 elements; 24 scan bits).
Successfully traced scan group scan2 (29 elements; 28 scan bits).
Successfully traced 2 scan groups (total 54 elements; 52 scan bits).
INFO: Passed sanity check on scan group scan1.
INFO: Passed sanity check on scan group scan2.
*** Scan Sanity Check Summary:
*** 2 scan groups passed sanity check.
```

Fig. Scan chain tracing

- Floorplan → Specify Floorplan
  - Ratio (H/W): 1
  - Core Utilization: 0.65
  - Core to Left: 80
  - Core to Top: 80
  - Core to Right: 80
  - Core to Bottom: 80
  - OK





- The IOs of SRAM are along the bottom of SRAM. Now, we want to change the direction of SRAM IOs.
  - Press  and draw the hard-block "xsram0" to the right-up corner
    - i.e. In another way, you can type "setObjFPlanBox Instance xcore/xsram0 324 327 587.66 588.285." in the consol
  - Select the memory "xsram0" and press the hot key "R".
    - Or using: Floorplan→ Edit Floorplan→ Flip/Rotate Instances…
  - Select R270 → Apply → Cancel

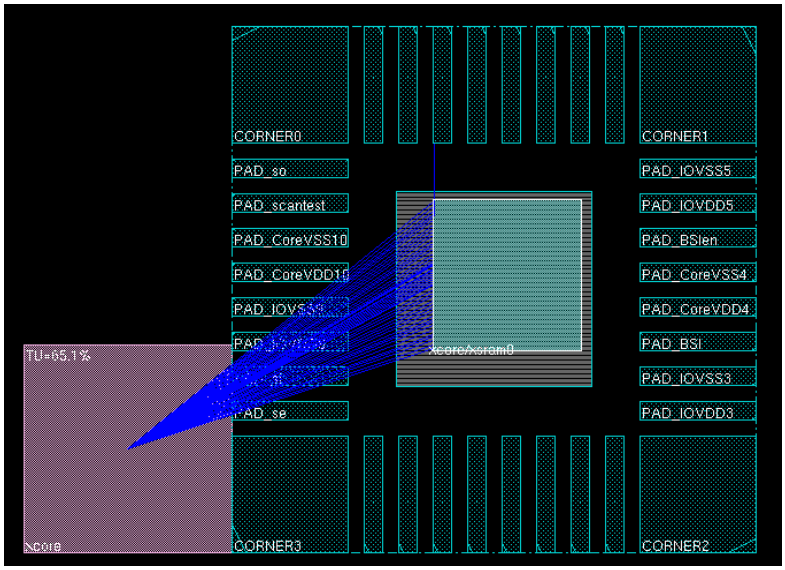


Fig. RAM placement

- Press  and select the SRAM. Then, Floorplan → Edit Floorplan → Set Instance Placement Status…
  - Selected → Apply → OK
  - i.e. this step keeps the RAM fixed in the selected position avoiding any

changes.

- Until now, the floorplan and hard blocks placement are done.
- Select the RAM and then, Floorplan → Edit Floorplan → Edit Halo…
  - Selected Block/Pad
  - Placement Halo → Add/Update Block Halo
  - Type 20 in the blanks Top, Bottom, Left, and Right. Apply → OK
  - i.e. this step keeps the RAM surrounded by the gap in which no other cells can be place in.

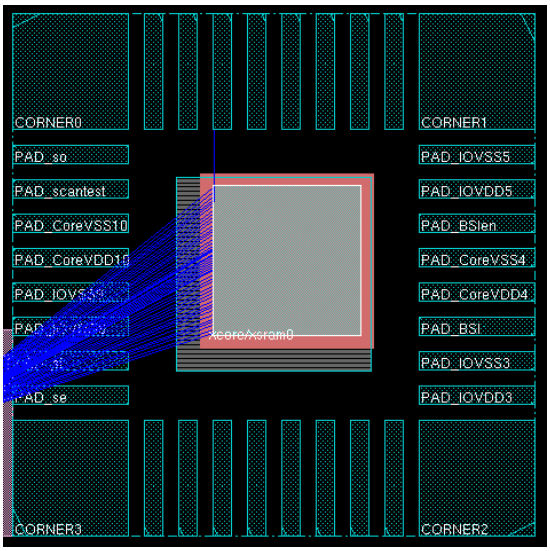


Fig. Hard block halos

- Place →Place Standard Cells…
  - Select "Run Full Placement"
  - De-select "Include Pre-Place Optimization"
  - Select "Include In-Place Optimization"
  - OK

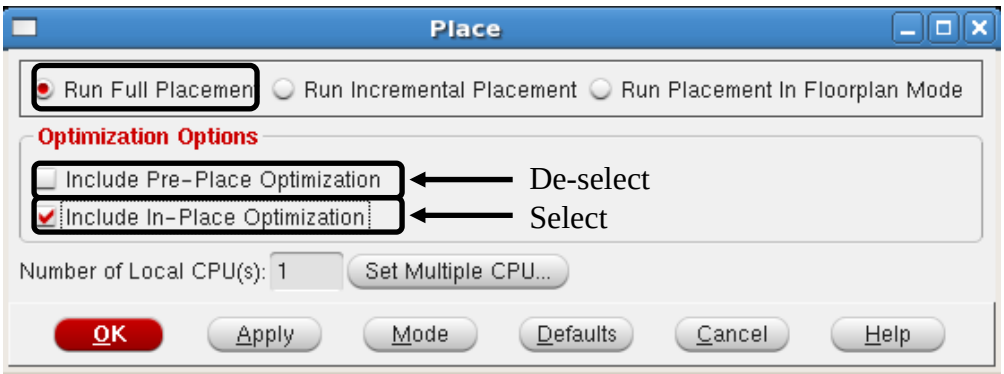


Fig. The standard cell placement setting

- After a period later (i.e. could be very long for a huge design), the placement of standard cells is done. Press  (physical view), you can see all the cells are placed.

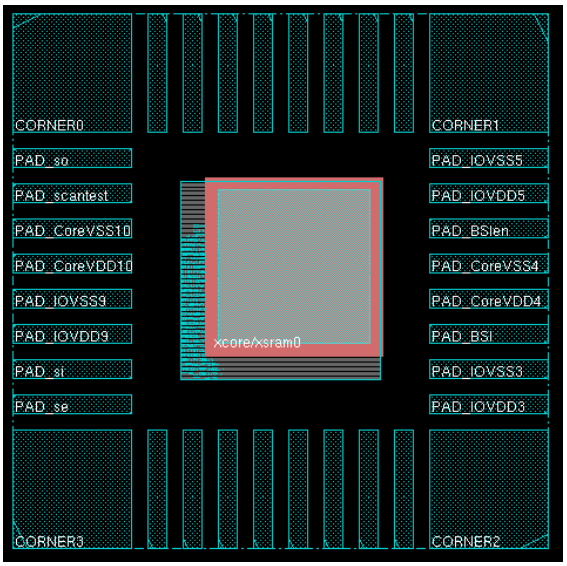


Fig. Placement

- Until now, the hard blocks and standard cells placement are done.
- Timing → Report Timing…
  - Select "Pre-CTS"
  - Select "setup" (i.e. setup time evaluation. "hold" is selected for the hold time evaluation)
  - OK
  - Does there any violation paths exist?

```
-----------------------------------------------------------
          timeDesign Summary
-----------------------------------------------------------

+--------------------+---------+---------+---------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | in2reg  | reg2out | in2out  | clkgate |
+--------------------+---------+---------+---------+---------+---------+---------+
|           WNS (ns):|  4.167  |  4.167  |  6.281  |   N/A   |   N/A   |   N/A   |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |   N/A   |   N/A   |   N/A   |
|    Violating Paths:|    0    |    0    |    0    |   N/A   |   N/A   |   N/A   |
|          All Paths:|   279   |   154   |   235   |   N/A   |   N/A   |   N/A   |
+--------------------+---------+---------+---------+---------+---------+---------+

+----------------+-------------------------------+------------------+
|                |              Real             |       Total      |
|    DRVs        +------------------+------------+------------------|
|                |  Nr nets(terms)  | Worst Vio  |  Nr nets(terms)  |
+----------------+------------------+------------+------------------+
|   max_cap      |      0 (0)       |   0.000    |      1 (1)       |
|   max_tran     |      1 (1)       |   -0.090   |      2 (2)       |
|   max_fanout   |      1 (1)       |    -36     |      2 (2)       |
+----------------+------------------+------------+------------------+
```

Fig. Timing analysis results

- If the timing is failed, the in-place optimization of the timing should be performed. Otherwise, you can ignore this step.
  - Optimize → Optimize Design
    - Select “Pre-CTS”
    - Select “setup”
    - OK

```
------------------------------------------------------------
          timeDesign Summary
------------------------------------------------------------
Setup mode
  Worst Slack: -0.237ns
  TNS: -0.237ns  Violating Paths: 1
  Pathgroup Slacks
    reg2reg: 3.594ns
    in2reg: 2.703ns
    reg2out: -0.237ns
    in2out: 8.586ns
Density: 14.415%
Routing Overflow: 0.00% H and 0.00% V
Real DRV (fanout, cap, tran): (2, 2, 0)
Total DRV (fanout, cap, tran): (3, 2, 0)
------------------------------------------------------------
Reported timing to dir timingReports
Total CPU time: 1.68 sec
Total Real time: 2.0 sec
Total Memory Usage: 247.594208 Mbytes
```

```
------------------------------------------------------------
     optDesign Final Summary
------------------------------------------------------------
Setup mode
  Worst Slack: 0.612ns
  TNS: 0.000ns  Violating Paths: 0
  Pathgroup Slacks
    reg2reg: 3.593ns
    in2reg: 2.895ns
    reg2out: 0.612ns
    in2out: 9.166ns
Density: 14.546%
Routing Overflow: 0.00% H and 0.00% V
Real DRV (fanout, cap, tran): (2, 0, 0)
Total DRV (fanout, cap, tran): (3, 0, 0)
------------------------------------------------------------
**optDesign ... cpu = 0:0:8, real = 0:0:9, mem = 247.6M **
*** Finished optDesign ***
```

Fig. Timing analysis before/after IPO

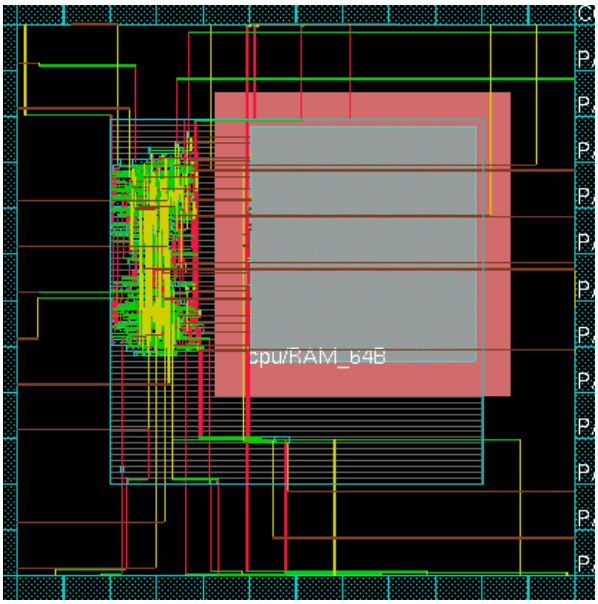


Fig. Trail routing

- **Since the timing analysis performs a simple routing (i.e. trial route)**

operation, they should be removed avoiding the hazard in the next step *"power ring generation"*

- Place → Refine Placement

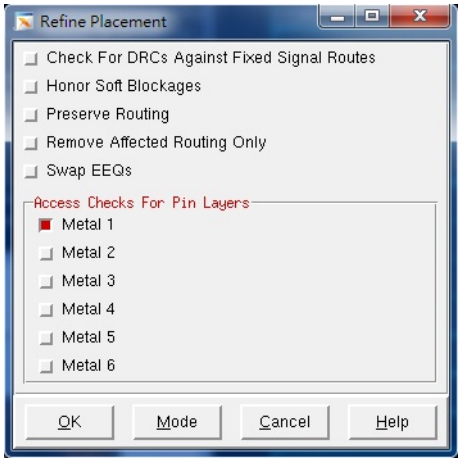
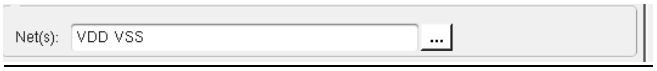


Fig. Refine placement

- File → Save Design
  - Type in "PLACE.enc" → Save
- Power →Power Planning → Add Rings…
  - <Basic> Tag
    - Select Power nets: VDD, VSS



    - Change the layer of "Top" and "Bottom" to "METAL9 H"; "Left" and " Right" to "METAL8 V"
    - Set all "Width" to 3 and p "Spacing" to 0.805 Press "-----"

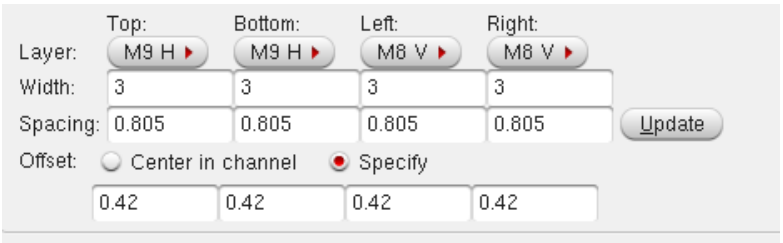


Fig. Ring setting

  - <Advanced> Tag
    - Select "Use wire group"
    - Select "Interleaving" and type in "9" to "Number of bits"
  - OK

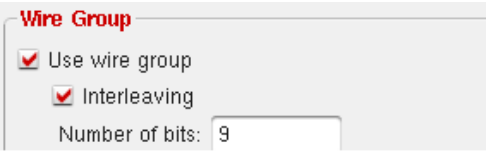


Fig. Wire group setting

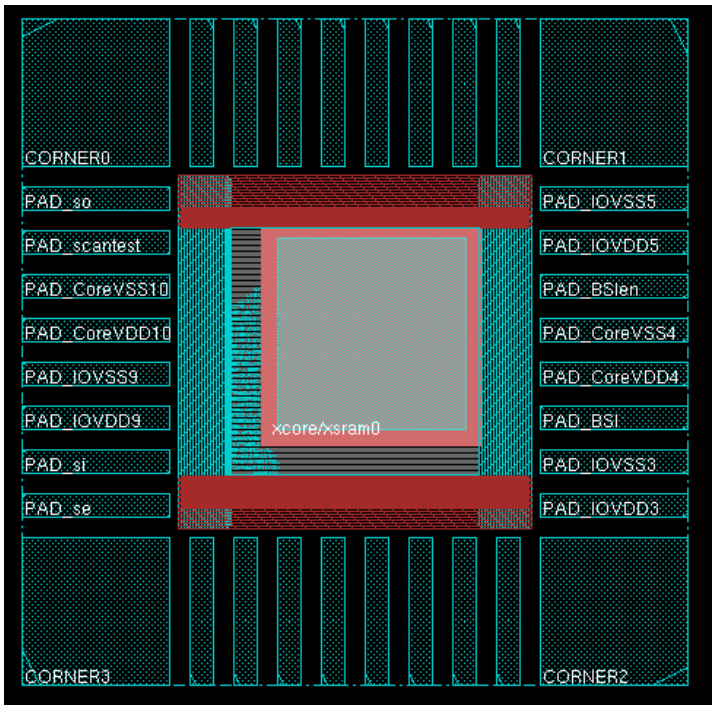


Fig. Power rings

- Route → Special Route…
  - Only "Pad pins" is selected, then press OK

Fig. Select Pad pins only

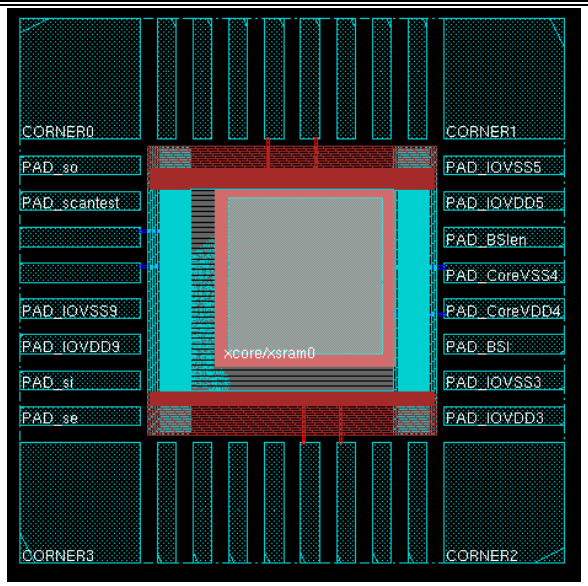
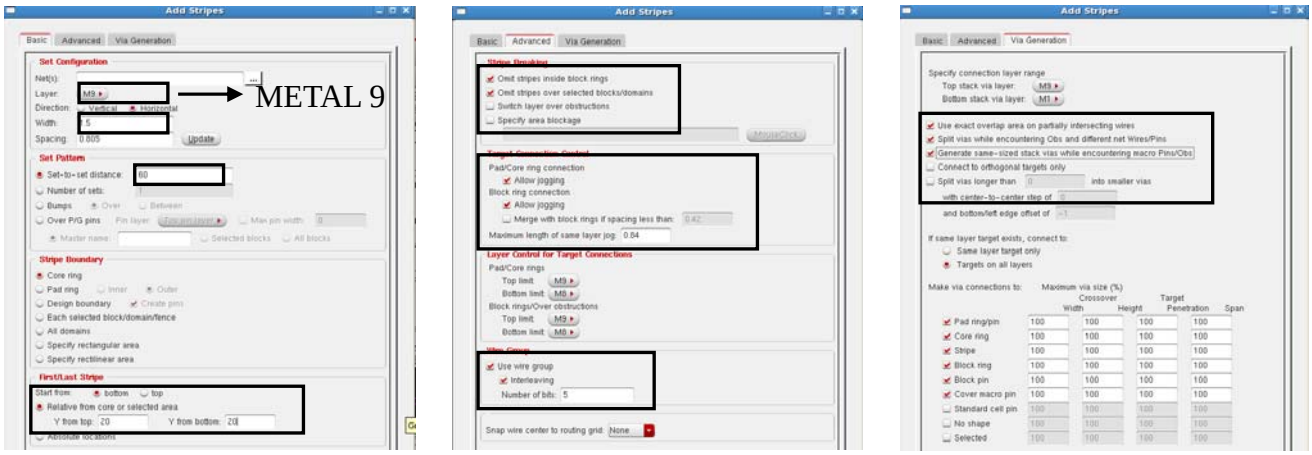


Fig. Power PAD connections

- Design → Save Design As → SoCE
  - Type in "PRE_STRIPE.enc" → Save
- Power →Power Planning → Add Stripes…
  (In METAL 9)
  - <u><Basic> Tag</u>
  - Layer: METAL9
  - Width: 1.5
  - Spacing 0.805
  - Set-to-set distance: 60
  - Start from: bottom
  - X from left: 20
  - X from right: 20
  - <u><Advanced> Tag</u>
  - Select "Omit stripes inside block rings"
  - Select "Omit stripes over selected blocks/domains"
  - Pad/Core ring connection → Select " Allow jogging"
  - Block ring connection → Select " Allow jogging"
  - **De-select "Merge with block rings if spacing less than: 0.56"**
  - Select "Use wire group"
  - Select "Interleaving" and set "Number of bits" to "5"
  - <u><Via Generation> Tag</u>
  - Select "Use exact overlap area on partially intersecting wires"
  - Select "Split vias while encountering Obs and different Wires/Pins"
  - **De-select "Connect to orthogonal targets only"**

- Select “Generate same-sized stack vias while encountering macro Pins/Obs”
- OK



Basic Advanced Via Generation

Fig. Power stripes setting

- Power →Power Planning → Add Stripes…
  (In METAL 8)
  - <Basic> Tag
  - Layer: METAL8
  - Start from: bottom
  - Y from top: 20
  - Y from bottom: 20
  - OK

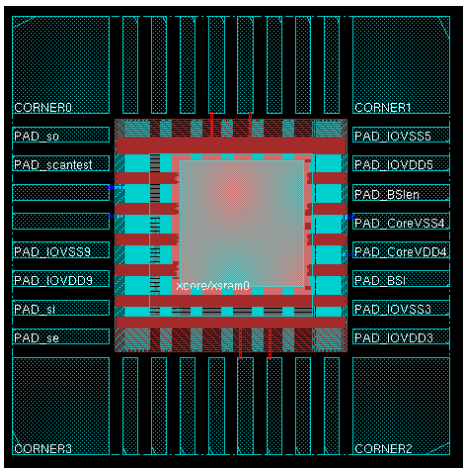


Fig. Power stripe generation

- Power stripes are used to reduce the IR-drop effect. However, some stripes do not connect correctly along the hard blocks.

- Due to GUI remove stripes (unconnected), we need to key in the following command
  - sroute -noBlockPins -noPadRings -noCorePins -noPadPins -jogControl { preferWithChanges differentLayer }
    (上面的指令要好好地注意正確性，在一行裡面輸入完成)
- However, some violations occur. We should figure out what's the problem.

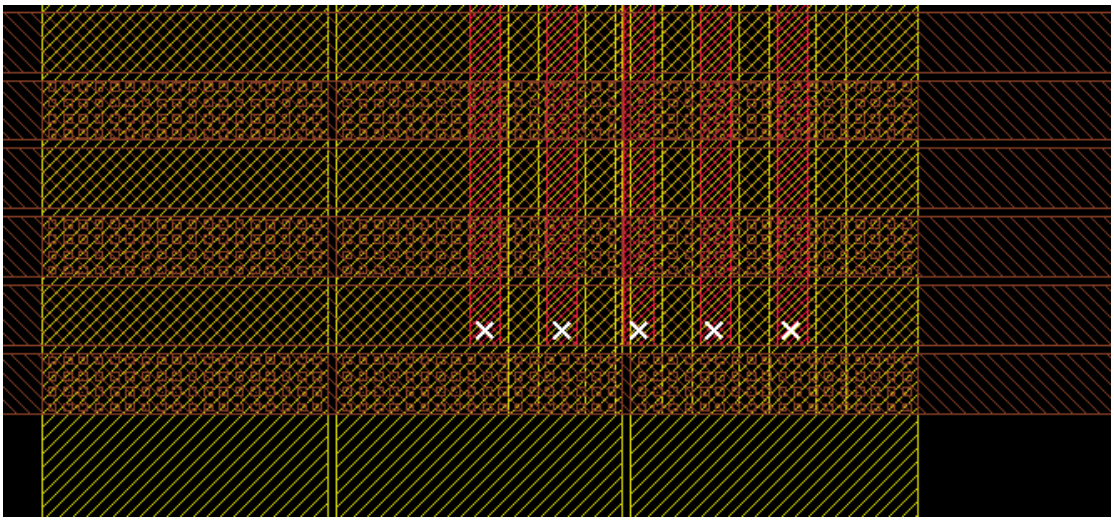


Fig. Open violations

- Verify→Violation Browser
  - There're 5 open nets

```
⊟ Other (5)
  ⊟ Connectivity (5)
    ⊟ Open (5) Net
                M2        (335.955, 249.355) (336.005, 249.40
                M2        (338.515, 249.355) (338.565, 249.40
                M2        (341.075, 249.355) (341.125, 249.40
                M2        (343.635, 249.355) (343.685, 249.40
                M2        (346.195, 249.355) (346.245, 249.40
```

- We can see that the violations are existed on M2, but some M4 wires cover the M2 violations. Therefore, we first block the display of M4. Follow the steps as Fig. below:

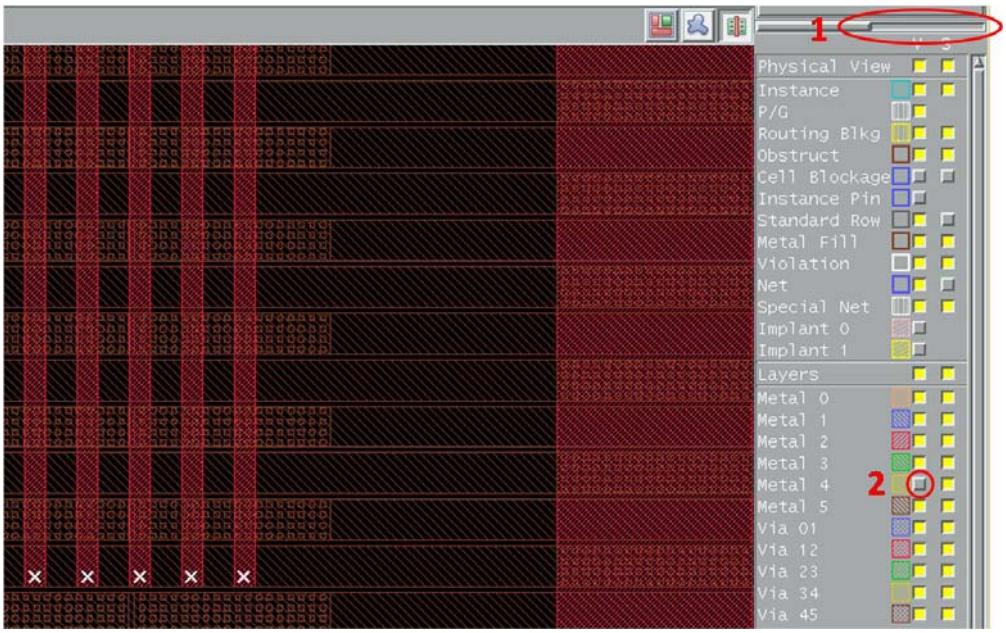


Fig. M4 blocking

- Select each open wire, and press “DEL”

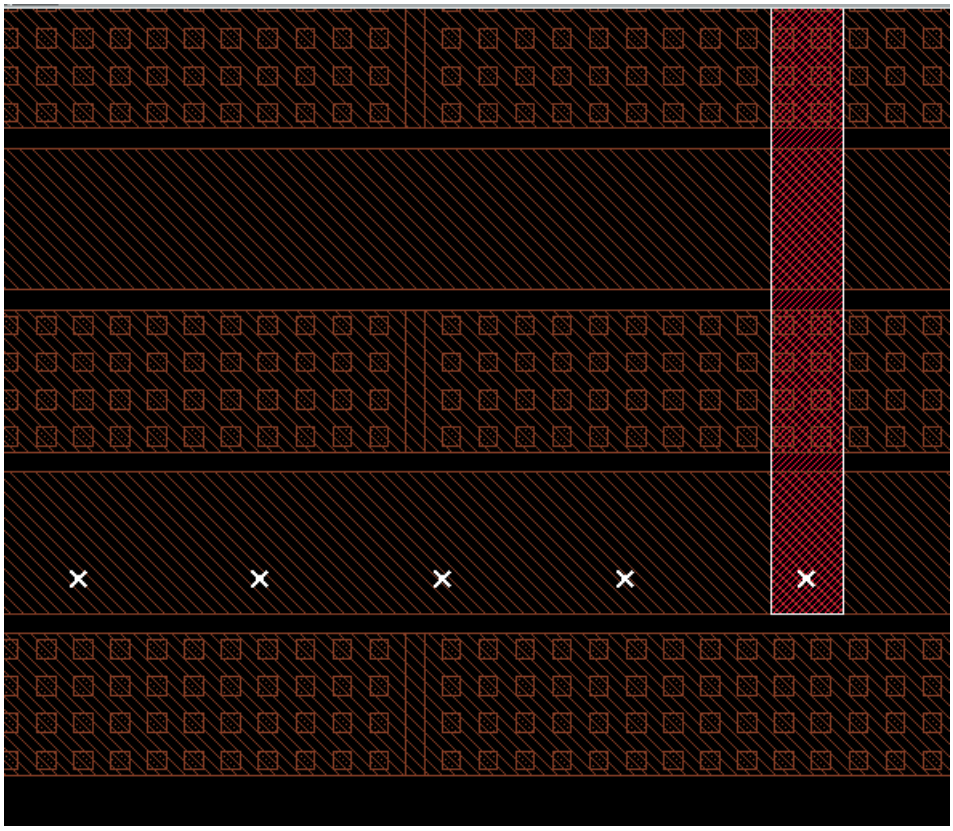


Fig. Delete open wires

- Tool →Violation browser→ Clear Violation→ Yes
- Verify→Verify Geometry…→OK

- Wire-short violations occur.

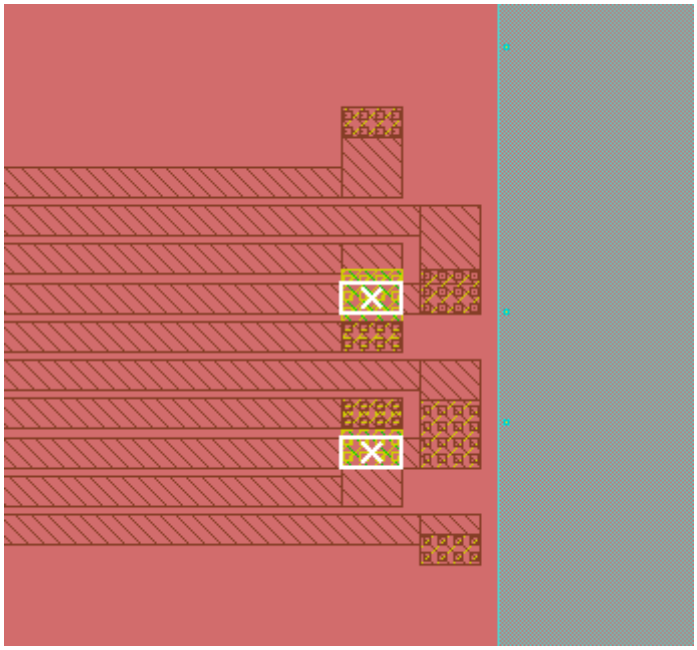


Fig. Short violations

- Tools→Violation Browser
  - There’re 2 short nets

```
⊟ Verify (2)
  ⊟ Short (2)
    ⊟ Short (2)  Layer      Geom        Net/Cell
                 M5         NET         VDD
                 M5         NET         VDD
```

- Select each short wire, and press “DEL”
- Verify→Verify Geometry…→OK
  - Finally, no violations exist

```
Begin Summary ...
  Cells        : 0
  SameNet      : 0
  Wiring       : 0
  Antenna      : 0
  Short        : 0
  Overlap      : 0
End Summary

  Verification Complete : 0 Viols.  0 Wrngs.

**********End: VERIFY GEOMETRY**********
 *** verify geometry (CPU: 0:00:01.7  MEM: 0.0M)
```

- File → Save Design…

- Type “STRIPE.enc” → Save

- Options→Set Mode→Mode Setup…
  - In the list, select “TieHiLo”
  - Press “Select” and select cells “TIEHI TIELO”
  - Specify Maximum Fanout: 10
  - Specify Maximum Distance: 100
  - OK

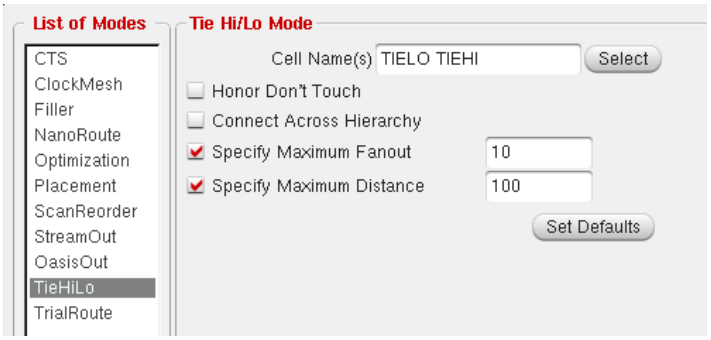


Fig. TIEHI & TIELO cells configurations

- Place→Tie HI/LO Cell→Add→OK
  - We can see that 1 TIELO cell and 1 TIEHI cell are placed

```
encounter 5> Options: Max Distance = 100.000 microns, Max Fan-out = 10.
INFO: Total Number of Tie Cells (TIELO) placed: 1
INFO: Total Number of Tie Cells (TIEHI) placed: 1
```

- Clock → Synthesize Clock Tree
  - Press “Gen Spec…”→ OK
- Open the file “Clock.ctstch” and revise the “MaxDelay” to”1”, since we do not expect that the delay is large as its default value.
- Return to “Synthesize Clock Tree Form”, and press ”OK”
- Does there any violation paths exist?
- If the timing is failed, the in-place optimization of the timing should be performed. Otherwise, you can ignore this step.
  - Optimize→Optimize design
    - Select “Post-CTS”
    - Select “setup”
    - OK
- Clock→Display→Display Clock Tree…
  - Select “Display Clock Tree” and “All Level”→ OK
  - You can see the clock tree as follows:

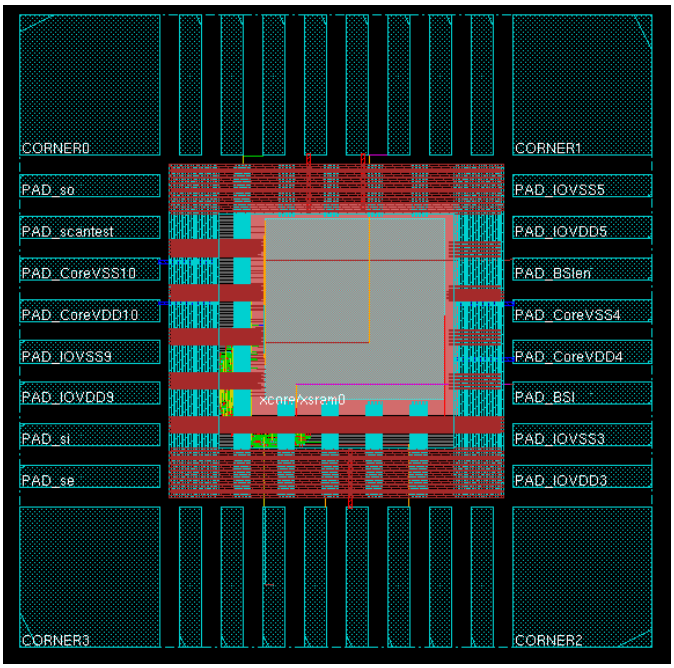


Fig. Clock tree

- The info. of the clock tree is reported in "../clock_report/clock.report"

Text Editor – clock.report

File Edit Format Options

```
# CLOCK: PAD_clk/C
#
# Mode: preRoute
#
##############################################################


Nr. of Subtrees                 : 1
Nr. of Sinks                    : 25
Nr. of Buffer                   : 0
Nr. of Level (including gates)  : 0
Root Rise Input Tran            : 0.1(ps)
Root Fall Input Tran            : 0.1(ps)
Max trig. edge delay at sink(R): cpu/RAM_64B/CLK 40.4(ps)
Min trig. edge delay at sink(R): cpu/regc_reg_5_/CK 4.5(ps)

                                  (Actual)              (Required)
Rise Phase Delay                : 4.5~40.4(ps)          0~1000(ps)
Fall Phase Delay                : 4.5~40.2(ps)          0~1000(ps)
Trig. Edge Skew                 : 35.9(ps)              300(ps)
Rise Skew                       : 35.9(ps)
Fall Skew                       : 35.7(ps)
Max. Rise Buffer Tran           : 0(ps)                 400(ps)
Max. Fall Buffer Tran           : 0(ps)                 400(ps)
Max. Rise Sink Tran             : 221(ps)               400(ps)
Max. Fall Sink Tran             : 205.6(ps)             400(ps)
Min. Rise Buffer Tran           : 0(ps)                 0(ps)
Min. Fall Buffer Tran           : 0(ps)                 0(ps)
Min. Rise Sink Tran             : 219.1(ps)             0(ps)
Min. Fall Sink Tran             : 203(ps)               0(ps)


***** NO Max Transition Time Violation *****

***** NO Min Transition Time Violation *****

***** NO Max_Fanout Violation *****
```

Fig.: clock.report

- Does there any violations exist?
- Design → Save Design
  - Type "CTS.enc" → Save

- Timing → Analyze Timing…
  - Select “Post-CTS”
  - Select “setup” (i.e. setup time evaluation. “hold” is selected for the hold time evaluation)
  - OK

----------(Power Analysis)----------

- The power analysis is not a mandatory step for the designer.
- Power →Analysis→ Edit Pad Location
  - Type “VDD” in “Net” and press “ Auto Fetch”
  - Type “VSS” in “Net” and press “ Auto Fetch”
  - Save… → CPU_CHIP.pp → Save→Cancle
- Power → Analysis→Edit Net Toggle Probability
  - Get Clock
  - Select “CLK1” and press “Edit”
  - Type “0.9” in the blank and press “Add/Replace” (i.e. For strengthen the IR-drop evaluation)
  - Save… → CPU_CHIP.tg →Save→ Cancle
- Power → Analysis→Power Analysis → Statistical…
  - Net Names: VDD
  - Select “post-CTS clock”
  - Net Toggle Probability File: CPU_CHIP.tg
  - Pad Location Files: CPU_CHIP.pp
  - Instance Voltage File: instance.voltage
  - Press “Apply”
- Power analysis results:
  - What is the average power?
  - What is the worst IR-drop?

```
power supply: 1.62 v
average power(default): 3.0797e+00 mw
    average switching power(default): 3.2080e-01 mw
    average internal power(default): 2.7476e+00 mw
    average leakage power(default): 1.1298e-02 mw
    average user specified power(default): 0.0000e+00 mw
average power by clock domain category:
    clock domain(CLK1, 0.9) : 3.0771e+00 mw
       clock tree power : 1.6775e+00 mw
       non clock tree power : 1.3996e+00 mw
          combinational instance power : 8.2722e-01 mw
          sequential instance power : 5.7235e-01 mw
    unclock domain(0.2) : 2.6510e-03 mw
average power by cell category:
     core: 1.2367e+00 mw
    block: 1.8430e+00 mw
       io: 0.0000e+00 mw
average power(considered in rail analysis): 3.0796e+00 mw
worst IR drop average analysis: 4.0125e-04 v
  number of nodes in rail network: 11450
worst EM:
    "M1"  4.2340e-02 mA/u
    "M2"  1.3492e-03 mA/u
    "M3"  1.9139e-02 mA/u
    "M4"  3.9171e-02 mA/u
    "M5"  5.7648e-02 mA/u
    "V12" 6.7296e-03 mA/cut
    "V23" 6.7296e-03 mA/cut
    "V34" 8.3367e-03 mA/cut
    "V45" 8.3367e-03 mA/cut
biggest toggled net: clk
    no. of terminal: 26
    total cap: 4.0055e+02 ff
*** Power analysis (cpu=0:00:01.3 mem=411.4M) ***
```

Fig. power analysis

- Power → Analysis→Display → Display Rail Analysis Results
  - Net Name: VDD
  - Select “IRD (V)”
  - IRD Threshold: 0.003
  - Press “Update filter range”
  - OK
  - The IR-drop evaluation will be shown.
  - You can block the “Net” and “Instance” in the right switch bar

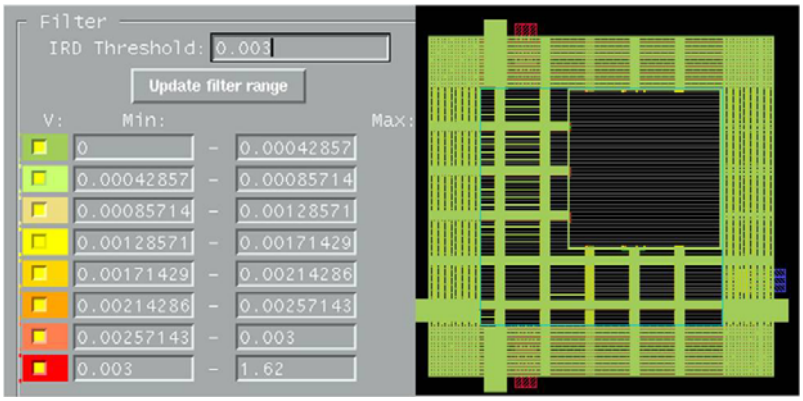


Fig. IR-drop

- Power → Analysis→Display → Display Rail Analysis Results
  - Net Name: VDD
  - Select “EM (J/Jmax)”
  - Press “Update EM limit”
  - OK
- Design → Save Design As → SoCE
  - Type “POWER_ANA.enc” → Save
- Route → Special Route…
  - Only “Follow Pin” is selected, then press OK
- We can see that all the standard cells are connected with horizontal power lines

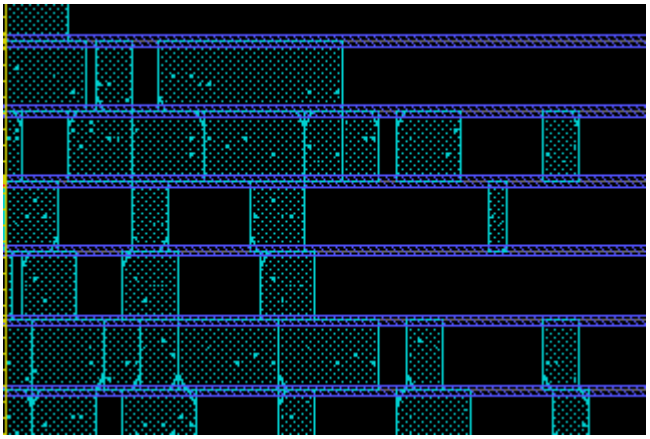


Fig. Power lines connection for standard cells

- Place→Refine Placement.. →OK
- Verify→Verify Geometry…→OK
  - Does there any violation occur?
- Verify→Verify Connectivity…
  - Select “Special Only”→OK
  - Does there any violation occur?
    - 5 antenna violations:

```
⊟ Verify (5)
  ⊟ Connectivity (5)
    ⊟ Antenna (5) Net
                  M4        VSS
                  M4        VSS
                  M4        VSS
                  M4        VSS
                  M4        VSS
```

- Select the wire with violation, and press “Shift+T” to kill the violation

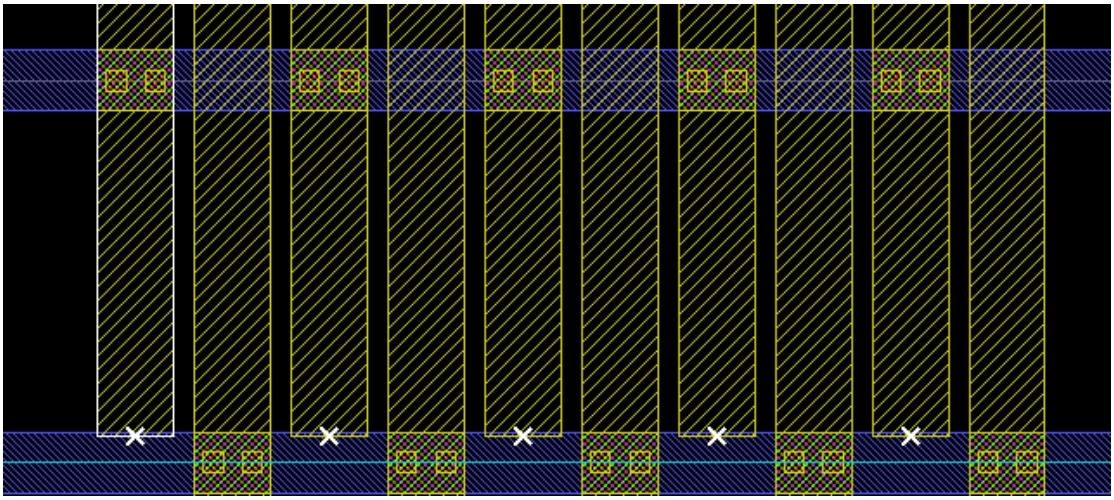
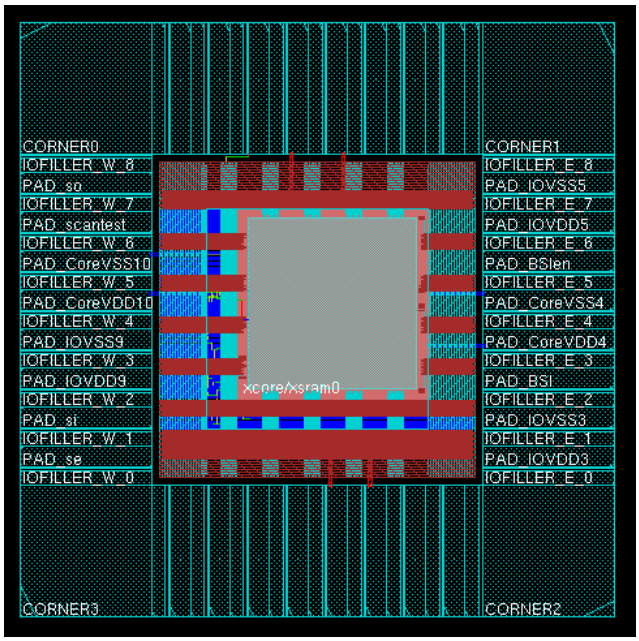
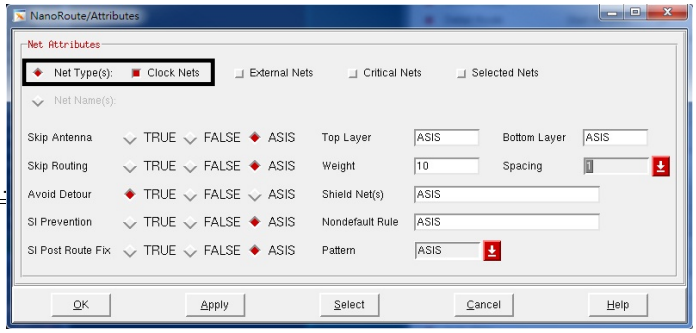


Fig. Antenna violations

- Re-do the verification procedure. Check whether any violation occurs.
- encounter> source addIoFiller.cmd
  - i.e. this step place the IO fillers between the IOs. But this design is "PAD-limit", the space for place the IO filler is limited.



- File → Save Design
  - Type "POWER_ROUTE.enc" → Save
- Route → NanoRoute→Route…
  - Select "Timing Driven"
  - Select "SI Driven"
  - Press "Attribute"
    - Select "Net Type(s)"



- Select “Clock Nets”
- Weight: 10
- Spacing: 1
- Avoid Detour: TRUE
- OK

- OK
- Does there any violation occur?

- Timing → Analyze Timing…
  - Select “Post-Route”
  - Select “setup” (i.e. setup time evaluation. “hold” is selected for the hold time evaluation)
  - OK

- Design → Save Design As → SoCE
  - Type “ROUTE.enc” → Save
- Place → Physical Cells → Add Filler…
  - Cell Name(s) → Select
  - Add all fillers to the left
  - Close
  - OK

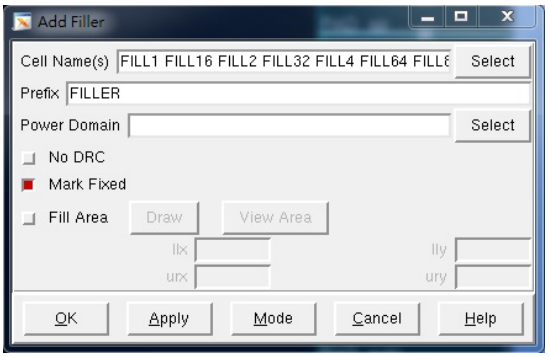


Fig. Add Filler

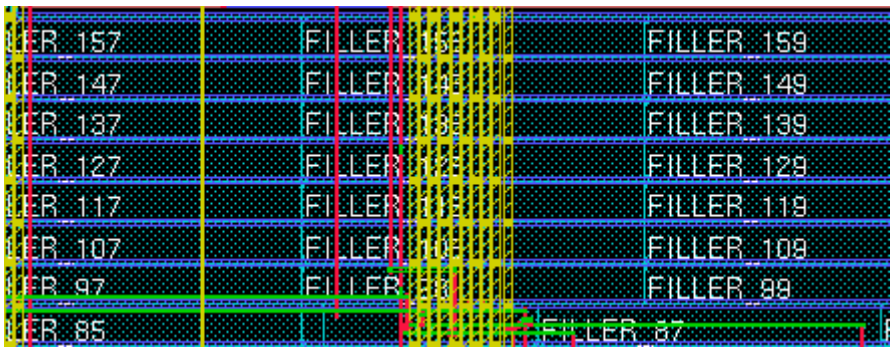


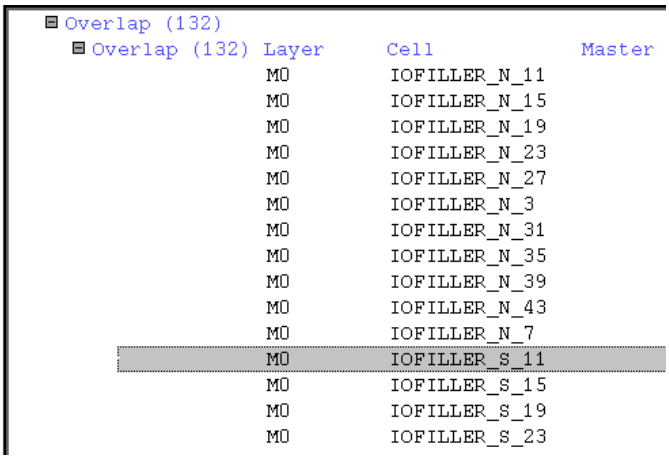
Fig. Core fillers

- How many core fillers are added?

```
*INFO: Iteration 0-#1, Found 3068 DRC violations (real: 0:00:01.0).
For 1395 new insts, *** Applied 2 GNC rules (cpu = 0:00:00.0)
*INFO: Iteration 0-#2, Found 1866 DRC violations (real: 0:00:01.0).
For 903 new insts, *** Applied 2 GNC rules (cpu = 0:00:00.0)
*INFO: Iteration 0-#3, Found 834 DRC violations (real: 0:00:00.0).
For 422 new insts, *** Applied 2 GNC rules (cpu = 0:00:00.0)
*INFO: Iteration 0-#4, Found 229 DRC violations (real: 0:00:01.0).
For 123 new insts, *** Applied 2 GNC rules (cpu = 0:00:00.0)
*INFO: Iteration 0-#5, Found 42 DRC violations (real: 0:00:00.0).
For 27 new insts, *** Applied 2 GNC rules (cpu = 0:00:00.0)
*INFO: Iteration 0-#6, Found 25 DRC violations (real: 0:00:00.0).
For 15 new insts, *** Applied 2 GNC rules (cpu = 0:00:00.0)
*INFO: Iteration 0-#7, Found 0 DRC violation  (real: 0:00:01.0).
*INFO: Adding fillers to top-module.
*INFO:   Added 0 filler inst of any cell-type.
For 0 new insts, *** Applied 0 GNC rules.
*INFO: End DRC Checks. (real: 0:00:04.0 ).
*INFO: Replaced 1330 fillers which had DRC vio's, with 2885 new fillers.
```

Fig. Filler info.

- Verify→Verify Geometry…→OK
  - Does there any violation occur?



Overlap (132)			
Overlap (132)	Layer	Cell	Master
	M0	IOFILLER_N_11	
	M0	IOFILLER_N_15	
	M0	IOFILLER_N_19	
	M0	IOFILLER_N_23	
	M0	IOFILLER_N_27	
	M0	IOFILLER_N_3	
	M0	IOFILLER_N_31	
	M0	IOFILLER_N_35	
	M0	IOFILLER_N_39	
	M0	IOFILLER_N_43	
	M0	IOFILLER_N_7	
	M0	IOFILLER_S_11	
	M0	IOFILLER_S_15	
	M0	IOFILLER_S_19	
	M0	IOFILLER_S_23	

  - However, the overlap violations on IO filler can be ignored.
  - Tools→Clear Violation→ Yes
- Verify→Verify Connectivity…
  - Select "All"→OK
- Does there any violation occur?
- File → Save Design
  - Type "CORE_FILLER.enc" → Save
- File → Save → Netlist… → CHIP_FINAL.v
- Timing → Calculate Delay… → CHIP_FINAL.sdf
  - If it is the first time to calculate delay, you need to extract RC parameters first.
  - Timing→ Extract RC… →OK.
- File → Save → DEF…

- Select “Save Scan”
- CHIP_FINAL.def
- OK

- unix> chmod 755 addbonding_v3.6.pl↵
- unix> /usr/bin/perl addbonding_v3.6.pl CHIP_FINAL.def↵
- encounter> source addbond.cmd↵
- File → Save Design
  - Type “FINISH.enc” → Save
- Options→Set Mode→Mode Setup…
  - Select “StreamOut” tag
  - Un-select the “Virtual Connection→OK”
- File → Save → GDS… → CPU_CHIP.gds → OK (i.e. Map File: streamOut.map)
- Design → Exit → Yes

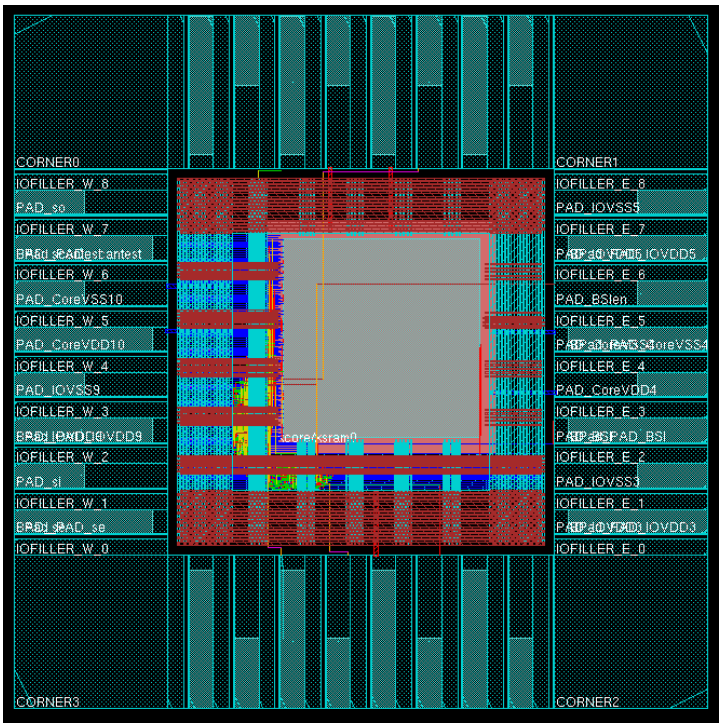


Fig. Bonding PADs