Lab 3: Floorplan with IC Compiler II

Objective

To be familiar with IC Compiler II and the floorplan steps:

- Physical synthesis data preparation
- Floorplanning
- Power network synthesis.

Reports

Please include the floorplan results and any problems and findings in the report, especially:

1. Screenshot of design after each step
2. IR drop map.

Introduction

IC Compiler II is a single, convergent netlist-to-GDSII synthesis design tool for chip designers developing very deep submicron designs. It takes as input a gate-level netlist, a detailed floorplan, timing constraints, physical and timing libraries, and foundry-process data, and it generates as output either a GDSII-format file of the layout.

IC Compiler II provides two user interfaces:

- Shell interface (icc2_shell)

The IC Compiler II command-line interface is used for scripts, batch mode, and push-button operations.

- Graphical user interface (GUI)

The IC Compiler II graphical user interface is an advanced analysis and physical editing tool.

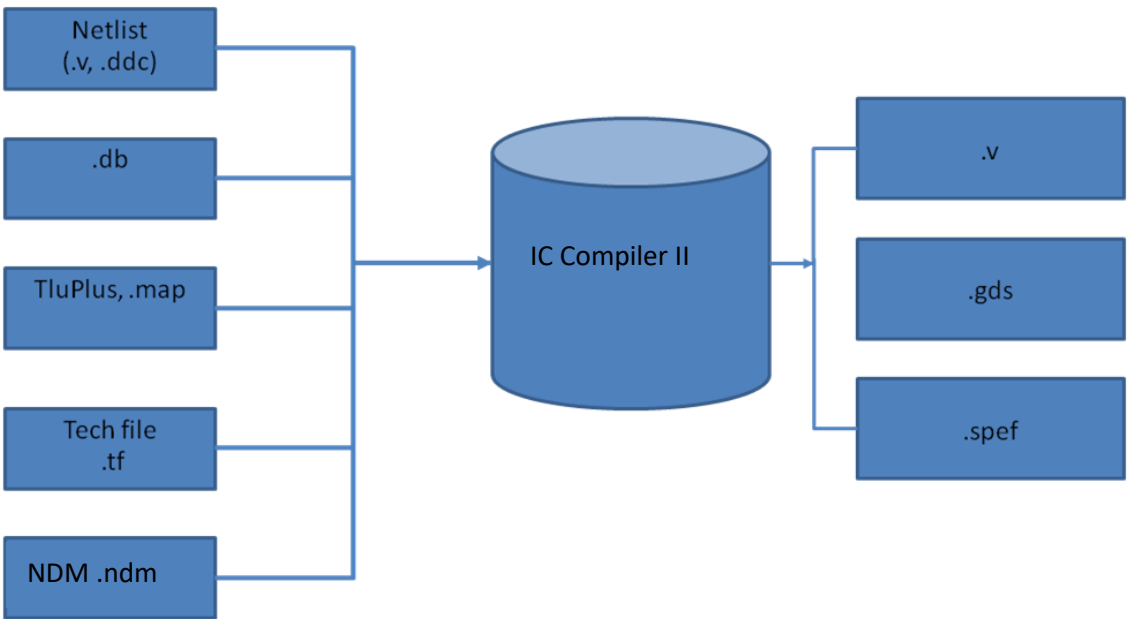

Fig. 1. Input and output files of IC compiler II

The logic, timing, and power information of the cell is typically contained in a set of Synopsys database (.**db**).

Tech file

A technology file provides technology-specific information, such as names and physical and electrical characteristics of each metal layer and routing design rules. IC Compiler II uses the Milkyway design library to access the technology information.

TluPlus

The parasitic attributes define the metal layer parasitics. In general, IC Compiler II gets the parasitic information from the TLUPlus files rather than from the technology file.

Theoretical part

IC Compiler || (ICC2) uses logic libraries to provide timing and functionality information for all standard cells. In addition, logic libraries can provide timing information for hard macros, such as RAMs.

ICC2 uses NDM databases and technology files to provide physical library information. NDM databases contain physical information about standard cells and macro cells in the library. In addition, these libraries define placement unit tile. The technology files provide technology specific information such as the names and characteristics for each metal layer. The physical library information is stored in the NDM design library. If NDM library is not created, it is necessary to create one before starting the design.

TLUPlus is a binary table format that stores the RC coefficients. The TLUPlus models enable accurate RC extraction results by including the effects of width, space, density, and temperature on the resistance coefficients.

Practical part

1. Move to the working directory lab03

```
$ cd $HOME/icdesign/m3/lab03
$ source /home/tools/synopsys/env.sh
```

2. Synthesis the design using Design Compiler

```
$ cd $HOME/icdesign/m3/lab03/syn
$ dc_shell -f scripts/compile.tcl 2>&1 | tee run.log
```

Check the synthesis report to see if there is any problem and include your QoR report in the lab report.

3. Invoke IC Compiler II

```
$ cd $HOME/icdesign/m3/lab03/pnr
$ icc2_shell -gui
```

A new window will open as in Fig. 2.

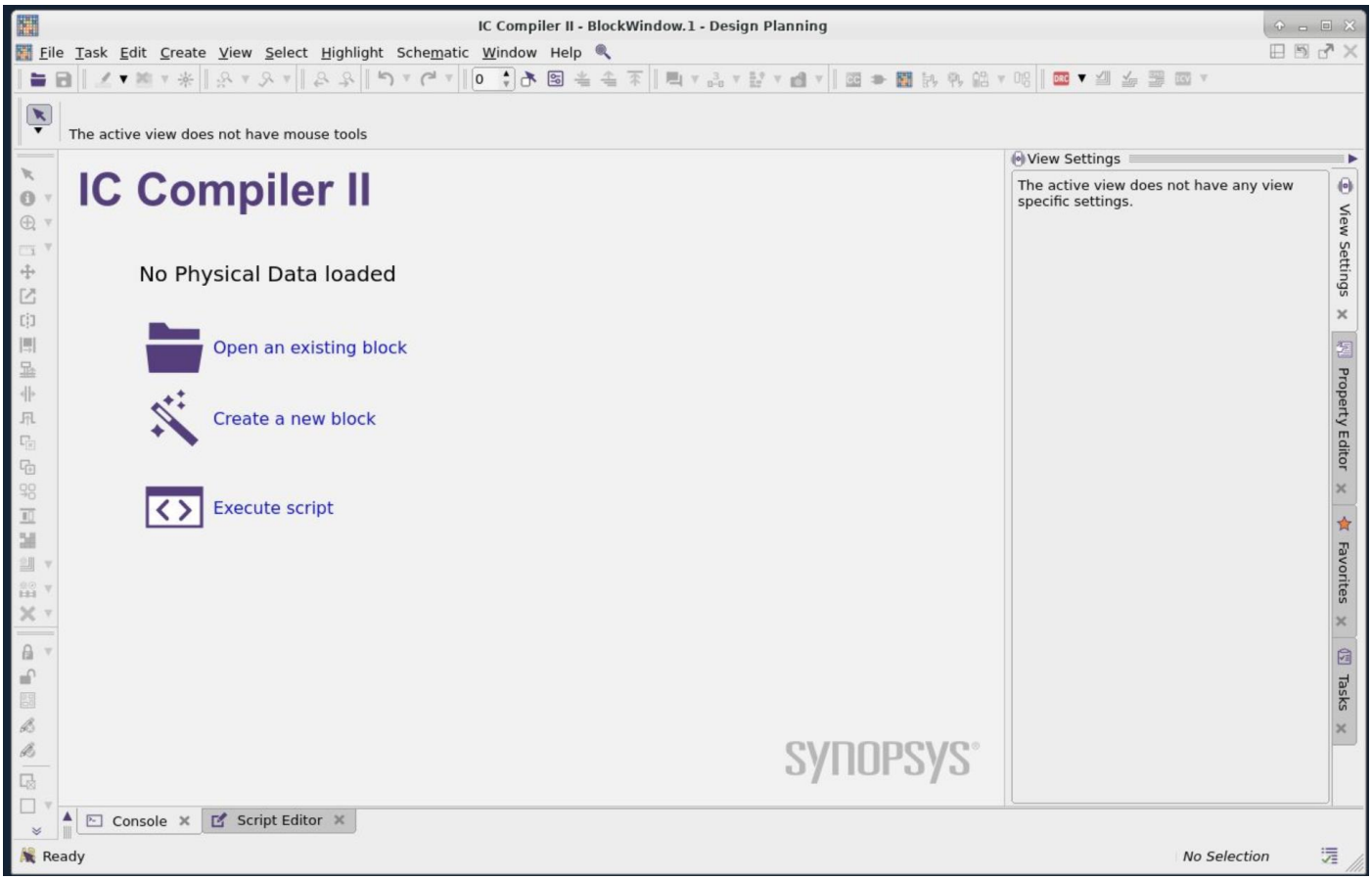

Fig. 2. IC Compiler II main window.

4. Setup target and link libraries.

Use the following commands to set the `link_library` and `target_library`:

```
icc2_shell> source ../common/common.tcl
icc2_shell> set link_library          "* saed14rvt_tt0p8v25c.db"
icc2_shell> set target_library        "saed14rvt_tt0p8v25c.db"
```

It is also possible to set the link_library and target_library through the GUI by using *File->Setup-> Application Setup*. The dialog box is shown in Fig. 3.

Fig. 3. Libary setup.

5. Create NDM library

```
icc2_shell> create_lib i2c_master_top.dlib \
                -technology saed14nm_1p9m.tf  \
                -ref_libs saed14rvt_frame_only.ndm
```

For GUI, use *File->Create Library*. The dialog box is shown in **Error! Reference source not found.**.

Fig. 4. Dialog box of creating a library

6. Setup TLU+ files

```
icc2_shell> read_parasitic_tech -tlup "saed14nm_1p9m_Cmax.tlup saed14nm_1p9m_Cmin.tlup" \
                    -layermap saed14nm_tf_itf_tluplus.map
```

For GUI mode choose *Task->Task Assistant->Create Block->Read Parasitic Model* (Fig. 5)

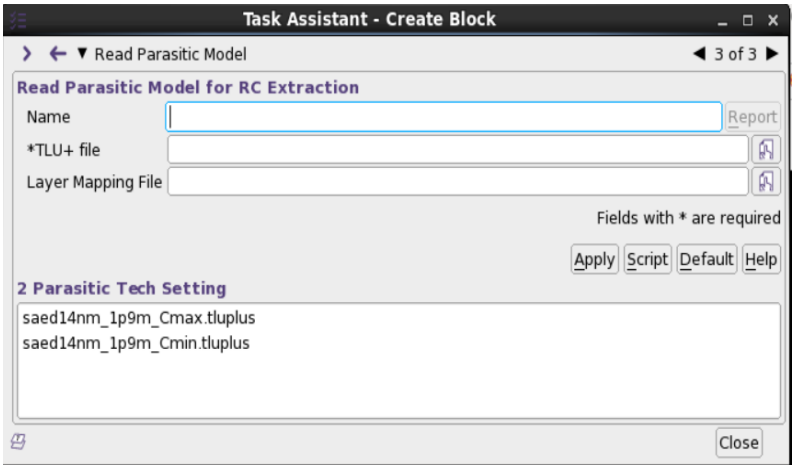

Fig. 5. Set TLU+ window.

Physical Synthesis: Floorplanning

The core can be drawn with 4 parameters: aspect ratio, width and height, row number, boundary.

- Aspect ratio - ratio of height divided by width.
- Width & height - the exact width and height.
- Row number - the number of rows.
- Boundary - a fixed size according to the boundary defined in design planning.

Core utilization factor indicates the amount of core area used for cell placement. The number is calculated as a ratio of the total cell area to the core area.

Aspect ratio (H/W):	the aspect ratio must be entered for the core area.
Row/Core ratio:	numbers from 0 to 1.0 indicate the amount of channel space provided for routing between the cell rows. The smaller is the number, the more space is left for routing.
Core width:	the width of the core area must be entered in the user units.
Num rows:	the number of cell rows must be specified.

Core height: the height of the core area in the user units must be specified.

Data preparation is performed. Next, floorplan and floorplan placement must be created.

7. Read the design

```
icc2_shell> read_verilog ../results/i2c_master_top.v
icc2_shell> current_design i2c_master_top
```

Design view after importing is shown in Fig. 6. After design importing, constraints' file can be sourced by the following command:

```
icc2_shell> source -echo -verbose ../inputs/i2c_master_top.sdc
```

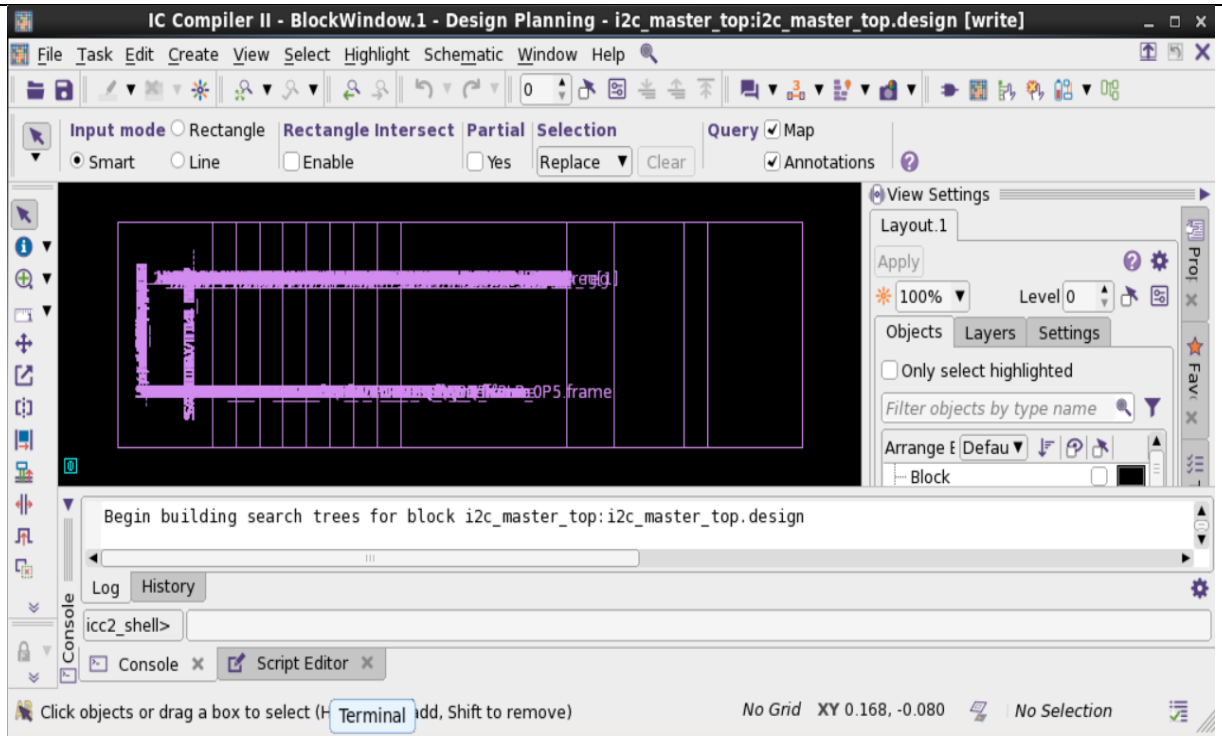

Fig. 6. Design view after importing

8. Floorplan the design

Before floorplan initialization the preferred routing directions for metals must be specified:

```
icc2_shell> set_ignored_layers -min_routing_layer M2 -max_routing_layer M7
icc2_shell> set_attribute [get_layers M1] routing_direction horizontal
icc2_shell> set_attribute [get_layers M2] routing_direction horizontal
icc2_shell> set_attribute [get_layers M3] routing_direction vertical
icc2_shell> set_attribute [get_layers M4] routing_direction horizontal
icc2_shell> set_attribute [get_layers M5] routing_direction vertical
icc2_shell> set_attribute [get_layers M6] routing_direction horizontal
icc2_shell> set_attribute [get_layers M7] routing_direction vertical
```

After that, use the following command to initialize the floorplan:

```
icc2_shell> initialize_floorplan -core_utilization 0.7 \
                        -core_offset {10 10 10 10}
```

For GUI, Use Task->Task Assistant->Floorplan Preparation->Floorplan Initialization. The dialog box is shown in Fig. 7.

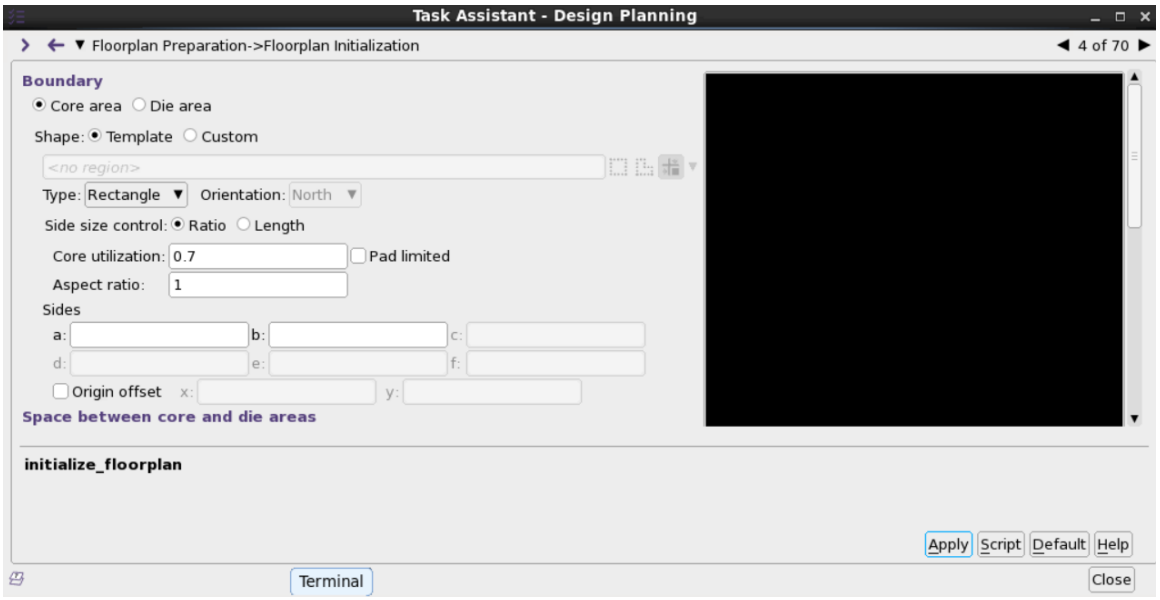

Fig. 7. Dialog box of floorplanning

The floorplanned design view is shown in Fig. 8.

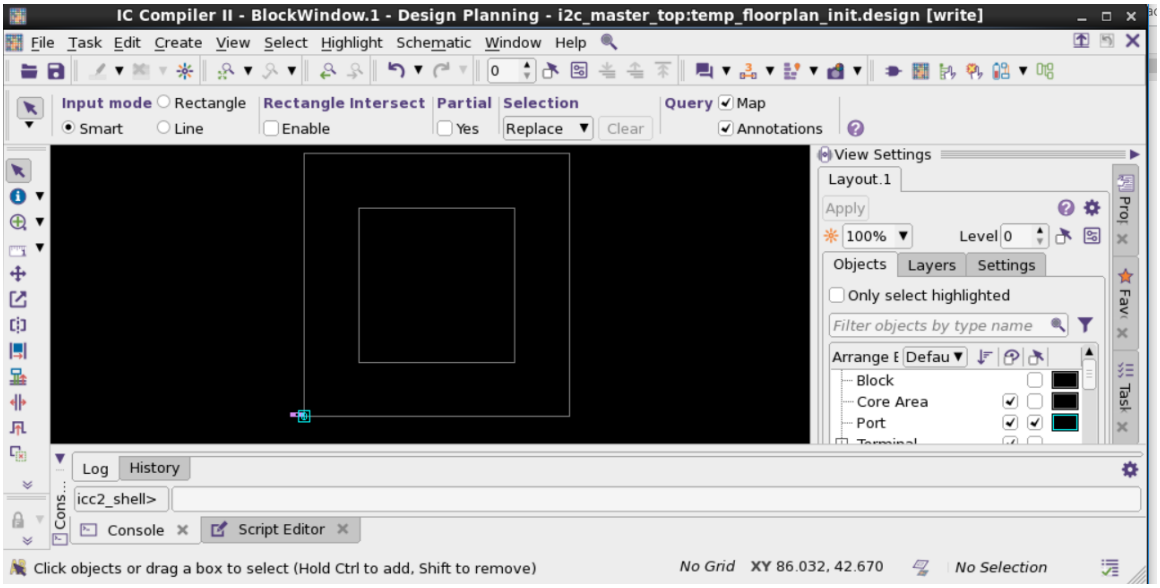

Fig. 8. Floorplanned design view

The design view after FloorPlan placement including pin placement and standard cell initial placement is shown in Fig. 9.

```
icc2_shell> place_pins -ports [get_ports *]
icc2_shell> create_placement -floorplan -effort high -timing_driven
icc2_shell> legalize_placement
```

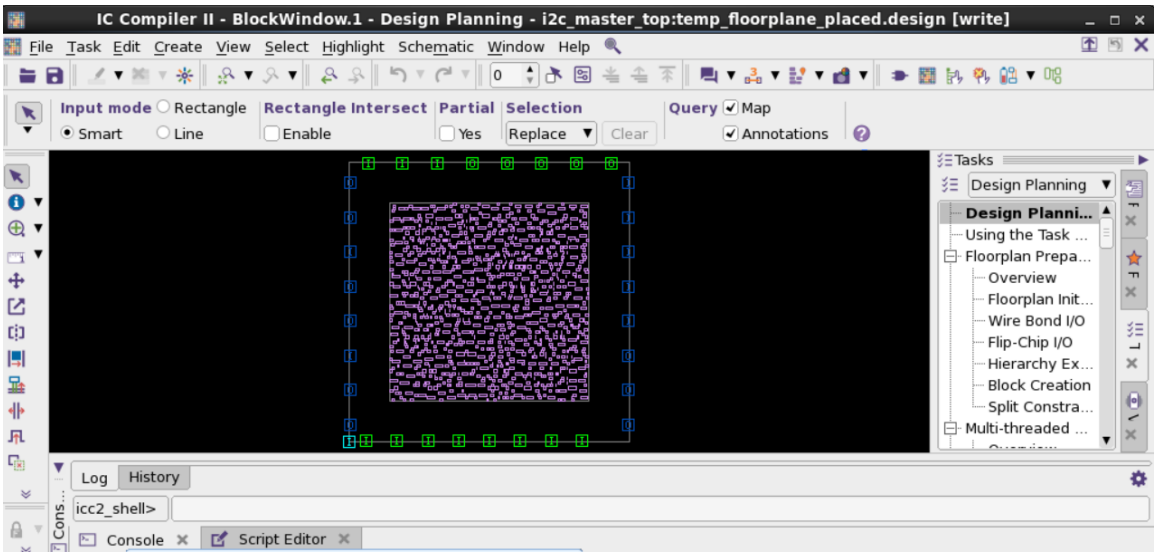

Fig. 9. Design view after FloorPlan placement

After floorplanning and floorplan placement, global route is performed by the following command:

```
icc2_shell> route_global -congestion_map_only true -effort high
```

Physical Synthesis: Power Net Synthesis (PNS)

Creation of rectangular rings and power straps is performed, after completing data preparation, floorplanning and floorplan placement.

9. Synthesize power rails

PNS (Power Net Synthesis) is a very important stage in design. In this lab, it will be done by the following steps:

At first all patterns, strategies and rules must be removed.

```
icc2_shell> remove_pg_via_master_rules -all
icc2_shell> remove_pg_patterns -all
icc2_shell> remove_pg_strategies -all
icc2_shell> remove_pg_strategy_via_rules -all
icc2_shell> create_net -power VDD
icc2_shell> create_net -ground VSS
icc2_shell> connect_pg_net -net VDD [get_pins -physical_context */VDD]
icc2_shell> connect_pg_net -net VSS [get_pins -physical_context */VSS]
```

After removing existing patterns, new pattern must be created for std_cells' rails connection.

```
icc2_shell> create_pg_std_cell_conn_pattern M1_rail -layers {M1} \
                  -rail_width {@wtop @wbottom} -parameters {wtop wbottom}
```

Then strategies for both power and ground must be created using the new pattern.

```
icc2_shell> set_pg_strategy M1_rail_strategy_pwr -core \
                 -pattern {{name: M1_rail} {nets: VDD} {parameters: {0.094 0.094}}}
icc2_shell> set_pg_strategy M1_rail_strategy_gnd -core \
                 -pattern {{name: M1_rail} {nets: VSS} {parameters: {0.094 0.094}}}
```

Compile_pg command is used to compile the rails given by the strategy.

```
icc2_shell> compile_pg -strategies M1_rail_strategy_pwr
icc2_shell> compile_pg -strategies M1_rail_strategy_gnd
```

The design view after compiling pg for std_cells' rails is shown in Fig.8.

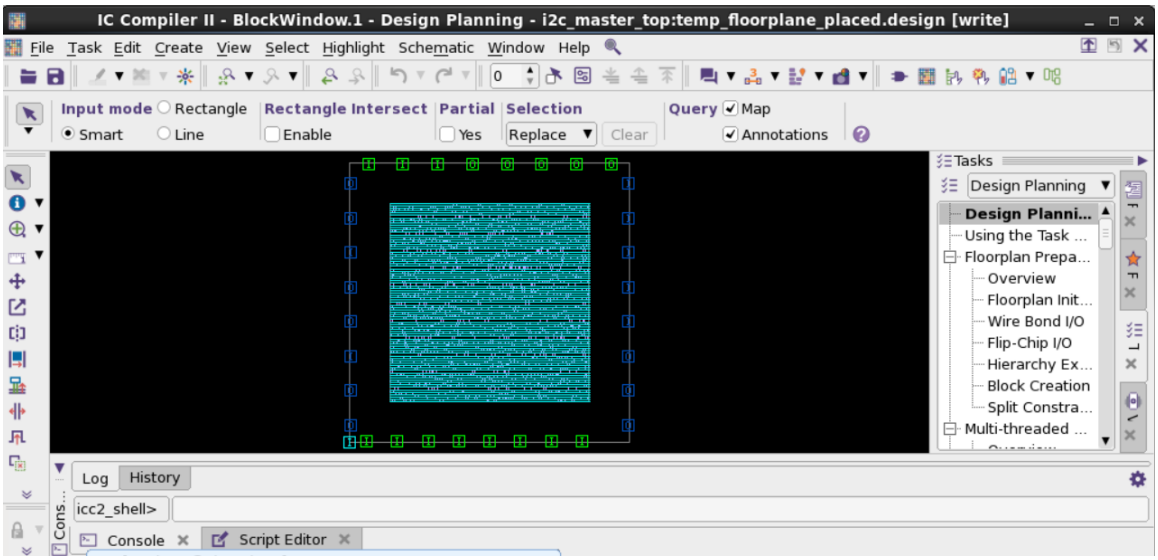

Fig. 8. Design view after std_cells' rail creation

Top pg mesh must be created using the following commands:

See that the approach is the same as for the std_cells' rail creation.

```
icc2_shell> create_pg_mesh_pattern TOP_MESH_VERTICAL \
                -layers "{{vertical_layer: M6} {width: 0.3} {spacing: interleaving} {pitch: 4}
{offset: 0.5} {trim: true}}"
icc2_shell> set_pg_strategy VDDVSS_TOP_MESH_VERTICAL \
                -core \
                -pattern {{name: TOP_MESH_VERTICAL} {nets: {VSS VDD}}} \
                -extension {{{stop:design_boundary_and_generate_pin}}}

icc2_shell> compile_pg -strategies {VDDVSS_TOP_MESH_VERTICAL} -show_phantom -ignore_drc
icc2_shell> create_pg_mesh_pattern TOP_MESH_HORIZONTAL \
                -layers " \
                {{horizontal_layer: M7} {width: 0.3} {spacing: interleaving} {pitch: 4}
{offset: 0.5} {trim: true}} \
                "

icc2_shell> set_pg_strategy VDDVSS_TOP_MESH_HORIZONTAL \
                -core \
                -pattern {{name: TOP_MESH_HORIZONTAL} {nets: {VSS VDD}}} \
                -extension {{{stop:design_boundary_and_generate_pin}}}

icc2_shell> compile_pg -strategies {VDDVSS_TOP_MESH_HORIZONTAL}
```

The design view after top pg synthesis is shown in Fig.9.

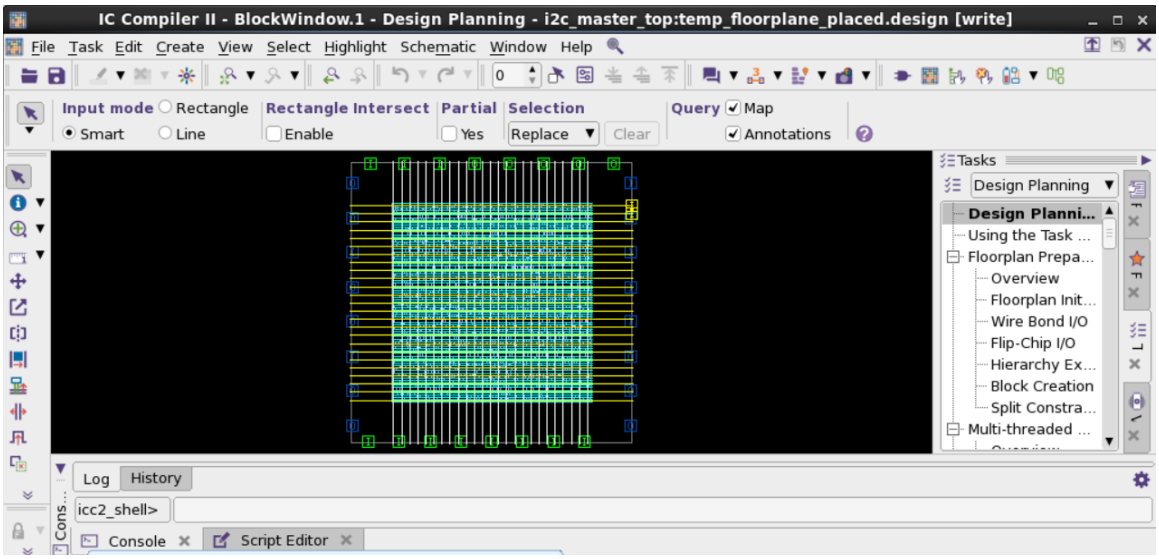

Fig. 9. Design view after top pg synthesis

In this lab power ring is also created. For power ring creation the following commands are used:

```
icc2_shell> create_pg_ring_pattern \
                ring_pattern \
                -horizontal_layer M7 -vertical_layer M6 \
                -horizontal_width 1 -vertical_width 1 \
                -horizontal_spacing 5 -vertical_spacing 5

icc2_shell> set_pg_strategy RING -core -pattern {{name: ring_pattern} {nets: "VDD VSS"}}

icc2_shell> compile_pg -strategies RING
```

The design view after complete PNS is shown in Fig.10.

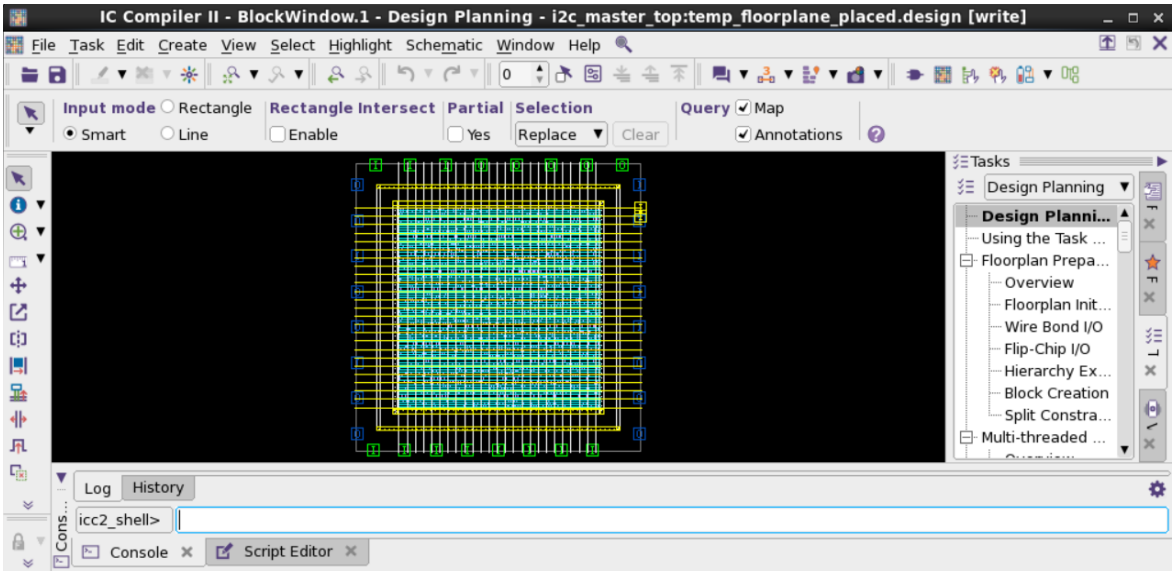

Fig.10. Design view after creating rectangular rings and power straps

10. Check the Power/Ground net connectivity

To verify the supply distribution network you can run the following command to check if it is good:

```
icc2_shell> check_pg_connectivity -nets "VDD VSS"
```

Is the floorplan good?

To read the error, you can open the Error Browser by selecting View >> Error Browser. Then in the error browser window (as in Fig. 10. Error browsers.Fig. 10), select the entry with the checker is check_pg_connectivity. Investigating the problem and try to fix them.

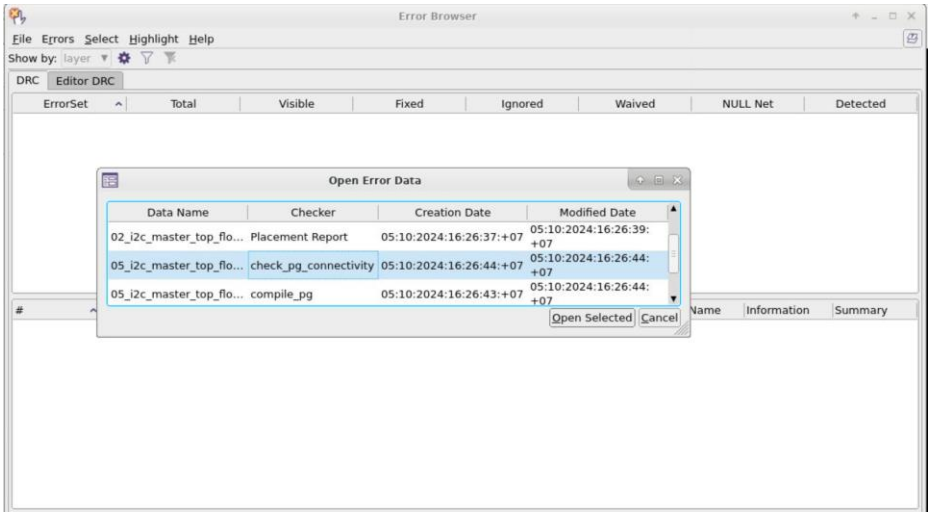

Fig. 10. Error browsers.

11. Analyze the power plan

To analyze the power plan run the following command:

```
icc2_shell> source scripts/mcmm.tcl
icc2_shell> analyze_power_plan -nets "VDD VSS" -power_budget 500 -voltage 0.8 \
                -use_terminals_as_pads
```

The result should be the following:

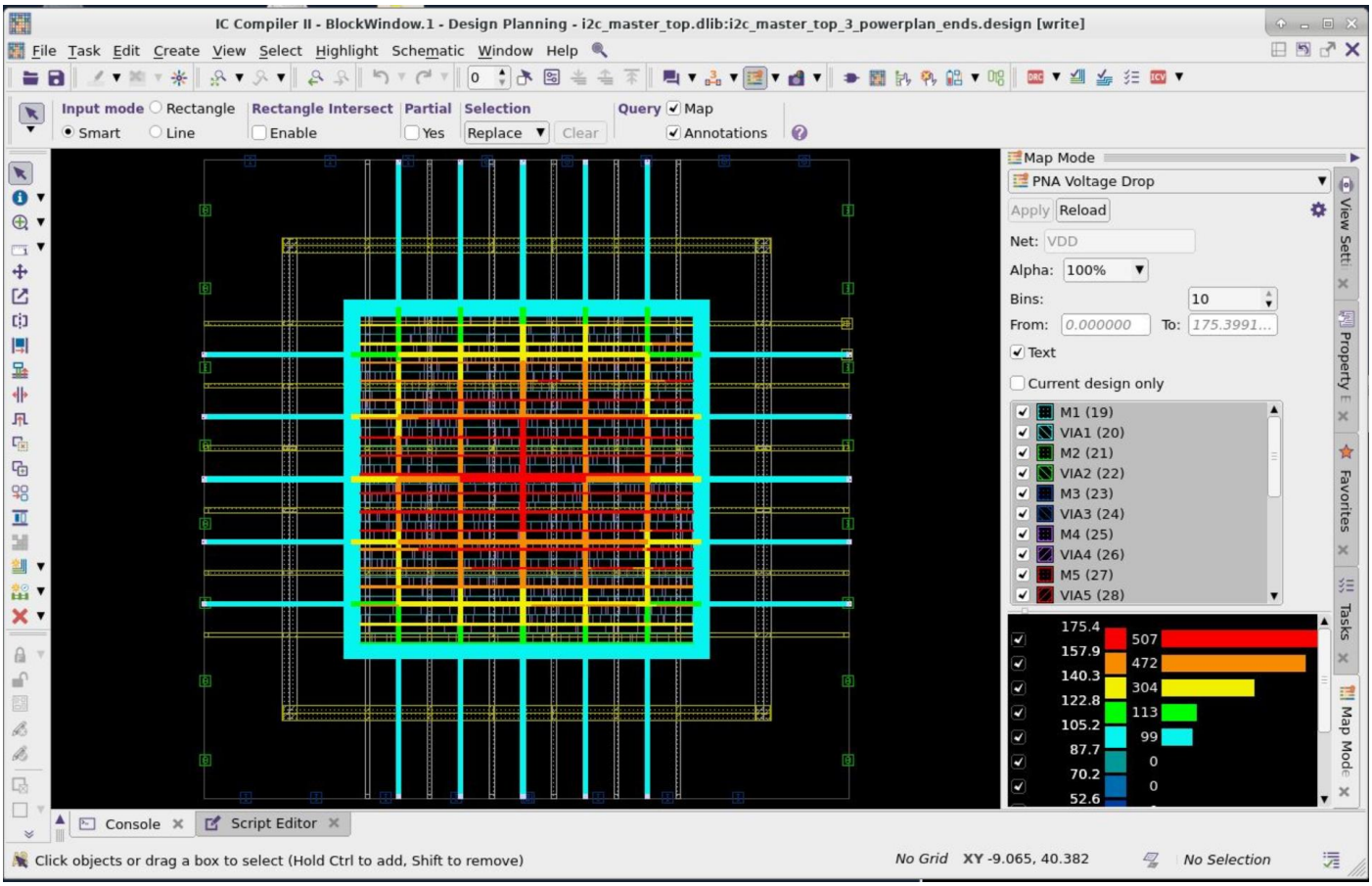

Fig. 11. Power plan analysis.

You can try to change the power budget and the supply voltage to see the difference. Put these results into your reports.

12. Change the utilization ratio and rerun the floorplan

You can try to change the core utilization ratio and rerun the floorplan steps and report the results in your report.