Lab 4: Placement and Route with IC Compiler II

Objective

To be familiar with IC Compiler II and the place and route steps:

- Placement
- Clock tree synthesis
- Routing.

Reports

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

1. Screenshot of design after each step

2. Generated scripts & reports

Placement

After completing design planning, floorplanning and power planning, optimization is performed. The place_opt command performs coarse placement, high-fanout net synthesis, physical optimization and legalization.

After performing placement, do legalize cell placement.

Legalize cell placement: Legal placement of cells is not required for analyzing routing congestion at an early stage.

After performing placement and “legalize cell placement” the following can be analyzed:
- Placement congestion (report_congestion)
- Power (report_power)
- Timing (report_timing).

Note that placement, routing and timing optimizations are performed in the multi_corner_multi_mode bases and for that, additional file is used, where appropriate corners and modes are created. That file is mcmm.tcl, which will be sourced before performing optimizations.

Practical part

- Move to directory lab6/work.
- Invoke ICC ||.
- Setup logic libraries.
- Set TLU+ files.
- Create NDM library.

Content: This lab includes the following scripts:
- flow.tcl
- mcmm.tcl
- step4_place.tcl
- step5_clock_tree_synthesis.tcl
- step6_route.tcl

Either source flow.tcl to complete this lab or go through steps. Note that in each script for this lab there are parts for reporting. View all reports after each completed step.

1. Move to the working directory lab04/pnr

```
$ cd $HOME/icdesign/m3/lab04/pnr
$ source /home/tools/synopsys/env.sh
```

2. Run the floorplan

The following command will run the floorplan that we did from the previous lab.

```
$ icc2_shell -f scripts/floorplan.tcl 2>&1 | tee run.log
```

3. Open the floorplaned design

Run ICC2 with GUI:

```
$ icc2_shell -gui
```

Open the library and the floorplaned design

```
icc2_shell> source ../common/common.tcl
icc2_shell> open_lib $ARC_TOP
icc2_shell> copy_block -from_block 06_${DESIGN_NAME}_floorplan_pns_finish \
                -to 07_${DESIGN_NAME}_place_start
icc2_shell> open_block  07_${DESIGN_NAME}_place_start
```

Run the checking:

```
icc2_shell> report_lib $ARC_TOP
```

4. Placement

To do this, use *Task->Task Assistant->Placement->place_opt*. The dialog box is shown in Fig. 1.
For proper placement optimization some application options must be updated, which are available in the below mentioned script.

```
icc2_shell> set_app_options -name time.disable_recovery_removal_checks -value false
icc2_shell> set_app_options -name time.disable_case_analysis -value false
icc2_shell> set_app_options -name place.coarse.continue_on_missing_scandef -value true
icc2_shell> set_app_options -name opt.common.user_instance_name_prefix -value place

icc2_shell> source scripts/mcmm.tcl
icc2_shell> place_opt
```

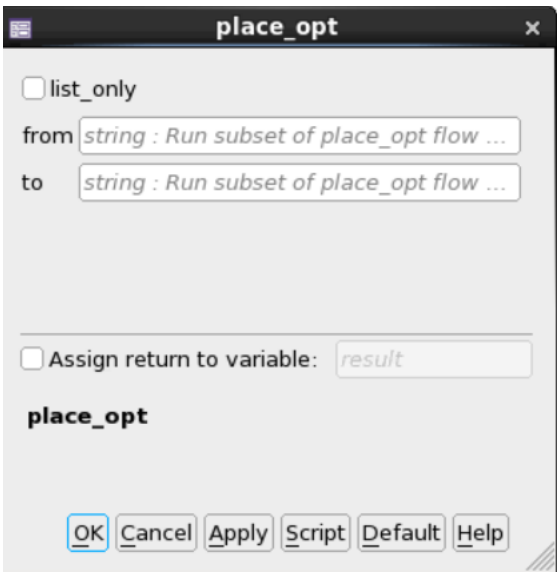

Fig. 1. Dialog box of placement

5. Legalize cell placement

To do this, use *Task->Task Assistant->Placement->Legalize_Placement*. The dialog box is shown in Fig. 2.

```
icc2_shell> legalize_placement
```

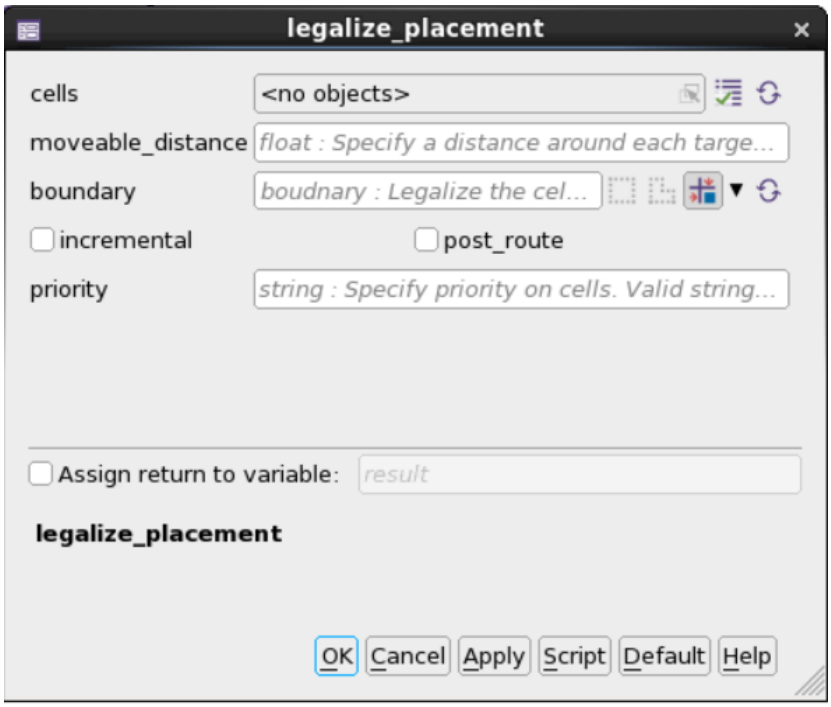

Fig. 2. Dialog box of legalize placement

6. Check the placement reports

```
icc2_shell> set_app_option -name time.snapshot_storage_location -value "./"
icc2_shell> create_qor_snapshot -name place_qor_snp -significant_digits 4
icc2_shell> file mkdir reports
icc2_shell> report_qor_snapshot -name place_qor_snp > reports/place.qor_snapshot.rpt
icc2_shell> report_qor > reports/place.qor
icc2_shell> report_constraints -all_violators > reports/place.con
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets
                  -delay_type max > reports/place.max.tim
icc2_shell> report_timing -capacitance -transition_time -input_pins \
                  -nets -delay_type min > reports/place.min.tim
```

7. Clock Tree Synthesis (CTS)

Before running CTS in IC Compiler || (ICC2), the design must meet the following requirements:

- The design is placed and optimized. To verify that the placement is legal **check_legality –verbose** command is used. The estimated QoR for the design must meet the requirements before starting clock tree synthesis.
- Logical, physical libraries and TLU+ files must be provided.

In this phase one must verify that the clock sources are correctly defined using **check_clock_trees** command.

Note that if the clock root is an input port without I/O pad cell, driving cell must be specified for port.

```
set_driving_cell -lib_cell lib_name/lib_cell [get_ports CLK]
```

If the clock root is an input port with I/O pad cell, input transition delay must be specified.

```
set_input_transition -rise 0.2 [get_ports CLK]
set_input_transition -fall 0.1 [get_ports CLK]
```

CTS in ICC2 has some exceptions. To mention the exceptions, **set_clock_balance_points set_dont_touch, set_size_only, set_dont_touch_network** commands with the necessary options are used.

To perform clock tree synthesis, clock tree optimization, and incremental physical optimization, clock_opt command is used. Some application options related to clock tree synthesis are updated, which are available in **step5_clock_tree_syntesis.tcl** script.

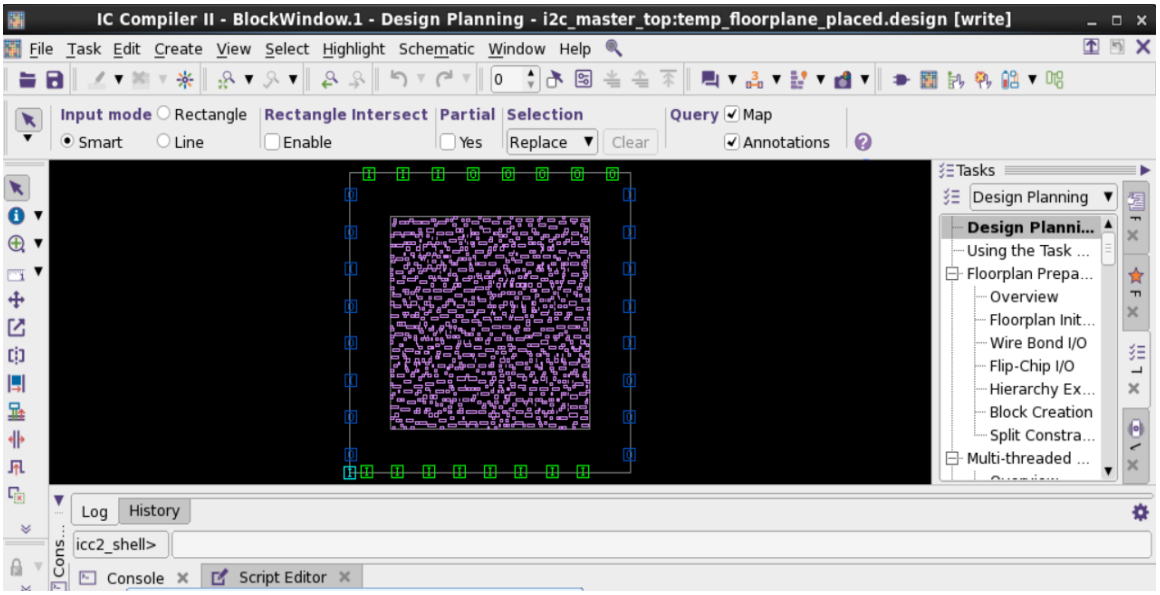

Fig.3.The design view after placement

The clock tree options can be specified by the mentioned command.

```
icc2_shell> check_legality  -verbose > reports/place_report.rpt
icc2_shell> set_ignored_layers -min_routing_layer ${MIN_ROUTING_LAYER} \
                  -max_routing_layer ${MAX_ROUTING_LAYER}
icc2_shell> set_app_options -name cts.compile.enable_cell_relocation -value all
icc2_shell> set_app_options -name cts.compile.size_pre_existing_cell_to_cts_references \
                -value true
icc2_shell> set_clock_tree_options    -clocks [all_clocks] \
                         -target_skew 0.1
```

The reference cells for clock tree synthesis can be specified by the following command.
The dialog box is shown in Fig.4.

```
icc2_shell> set_lib_cell_purpose -include cts {*/SAEDRVT14_INV_1 */SAEDRVT14_INV_2 \
                */SAEDRVT14_INV_4 */SAEDRVT14_INV_8 */SAEDRVT14_INV_16 */SAEDRVT14_INV_20 \
                */SAEDRVT14_BUF_2 */SAEDRVT14_BUF_4 */SAEDRVT14_BUF_6 */SAEDRVT14_BUF_8 \
                */SAEDRVT14_BUF_16 */SAEDRVT14_BUF_20 }
```

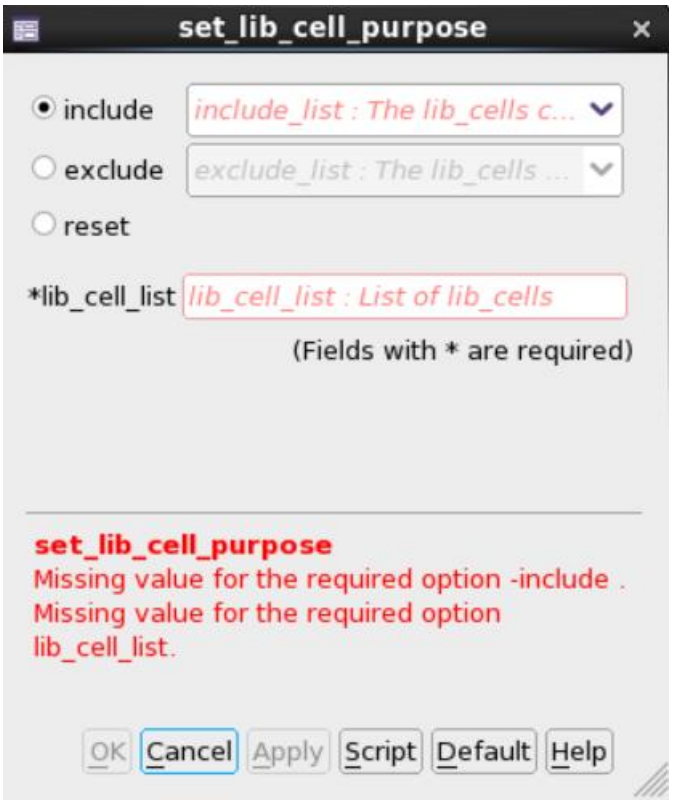

Fig. 4. Dialog box of Clock Tree References

Apply the clock tree settings

```
icc2_shell> set_clock_uncertainty 0.1 [all_clocks]
icc2_shell> create_routing_rule CLK_SPACING -spacings {M2 0.3 M3 0.5 M4 0.7}
icc2_shell> set_clock_routing_rules -rules CLK_SPACING -min_routing_layer M2 \
                -max_routing_layer M4
icc2_shell> report_clock_settings
```

Perform clock tree synthesis. Use *Task->Task Assistant->Clock Tree->clock_opt*. The dialog box is shown in Fig. 5.

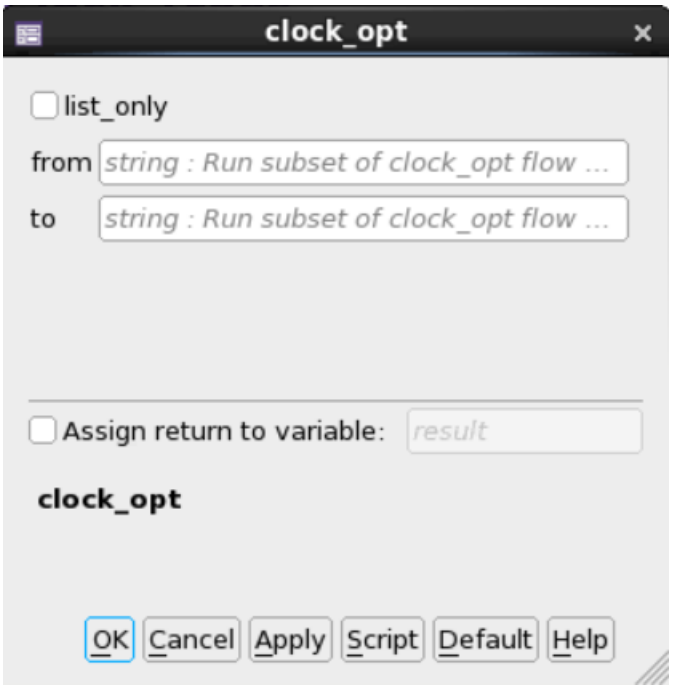

Fig. 5. Dialog box of CTS

Note that clock_opt command has -from -to options, which can be used to control clock tree synthesis process.

```
icc2_shell> set_app_options -name opt.common.user_instance_name_prefix -value clock
icc2_shell> source scripts/mcmm.tcl
icc2_shell> clock_opt -from build_clock -to build_clock
```

The synthesized clock tree is shown in Fig. 6. Choose Highlight->Color by->Clock Tree to view the synthesized clock tree.

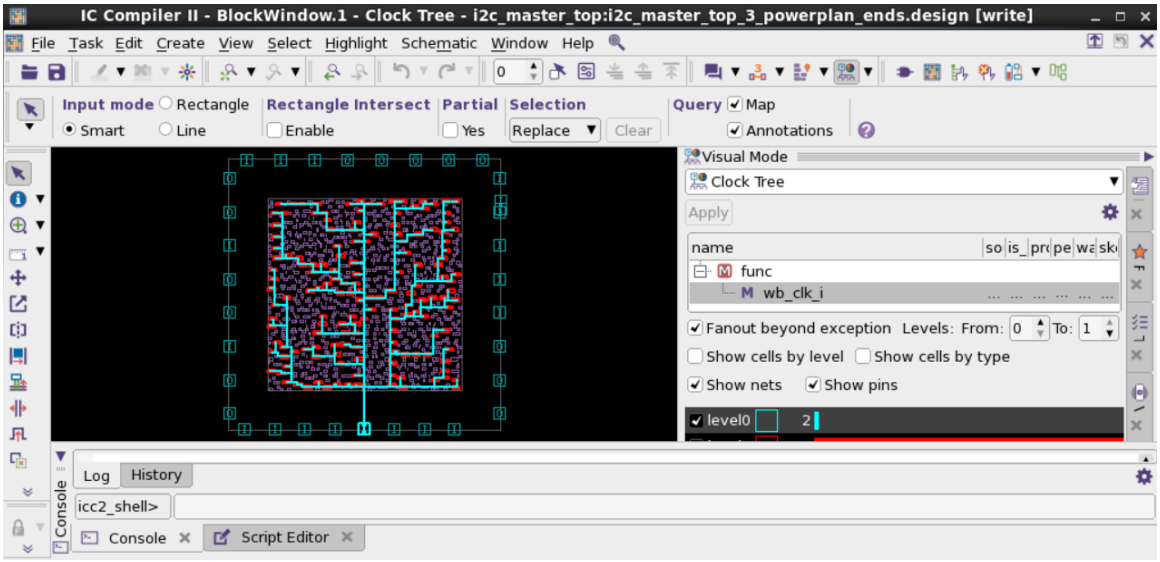

Fig. 6. Design view of clock

8. Clock tree synthesis report

```
icc2_shell> set_app_option -name time.snapshot_storage_location -value "./"
icc2_shell> create_qor_snapshot -name clock_pre_route -significant_digits 4

icc2_shell> report_qor_snapshot \
                -name clock_pre_route > reports/clock_pre_route.qor_snapshot.rpt
icc2_shell> report_qor > reports/clock_pre_route.qor
icc2_shell> report_constraints -all_violators > reports/clock_pre_route.con
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets \
                -delay_type max > reports/clock_pre_route.max.tim
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets \
               -delay_type min > reports/clock_pre_route.min.tim
```

9. Clock tree synthesis optimization and reports

```
icc2_shell> set_app_options -name opt.common.user_instance_name_prefix -value clock
icc2_shell> clock_opt -from route_clock -to final_opto
icc2_shell> report_clock_qor > reports/clock_tree.rpt
icc2_shell> report_clock_timing -type skew > reports/clock_timing.rpt
icc2_shell> create_qor_snapshot -name clock -significant_digits 4
icc2_shell> report_qor_snapshot -name clock > reports/clock.qor_snapshot.rpt
icc2_shell> report_qor > reports/clock.qor
icc2_shell> report_constraints -all_violators > reports/clock_route.con
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets \
                -delay_type max > reports/clock.max.tim
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets \
                -delay_type min > reports/clock.min.tim
```

10. Routing

First verify that the design is ready for detailed routing. For this, select *Task -> Task Assistant -> Routing -> check_routability.*

In ICC2 routing guide can be chosen by selecting *Task->Task Assistant->Routing-> create_routing_guide*.

To specify routing direction, *routing_direction* attribute is used. The syntax is:

set_attribute [get_layers metal] routing_direction horizontal | vertical

In ICC2 routing layers can be mentioned. For this, **set_ignored_layers** command with the necessary option is used.

-min_routing_layer layer_name – with this option the mentioned layer and all the layers above it are used for routing.

-max_routing_layer layer_name- with this option the mentioned layer and all the layers below are used for routing.

-min_routing_layer min_layer_name

-max_routing_layer max_layer_name- with this option all the layers between the mentioned layers are used for routing.

-rc_congestion_ignored_layers

layer_name...– the mentioned layers are ignored for RC and congestion estimation.

Route type, routing rules, net aggressors are specified in *Task->Task Assistant-> Routing Setup*.

To set routing and post route optimization strategy appropriate application options for zroute engine are selected.

Commands which are used for signal nets routing are shown in Fig.7. To see congestion map *View->map->Global Route Congestion* is selected. To view the place and route summary report ***check_routes, signoff_check_drc, check_lvs, check_legality*** commands can be used. For more commands see the appropriate section in *Task->Task Assistant*.

Fig. 7. Commands for routing and optimization

Dialog box of route options is shown in Fig. 8.

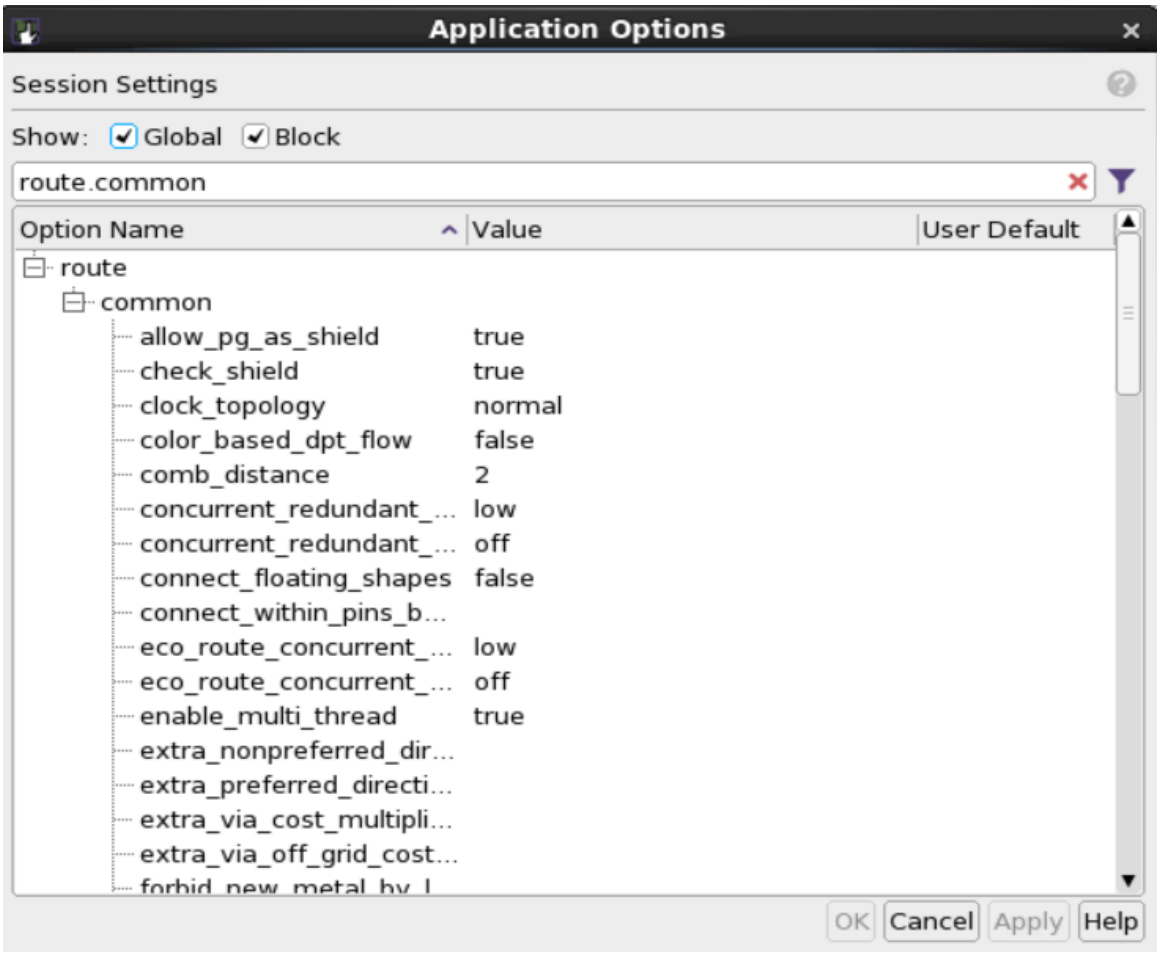

Fig.8. Dialog box of route options

Dialog box of checking routeability is shown in Fig. 9

```
icc2_shell> check_routeability
```

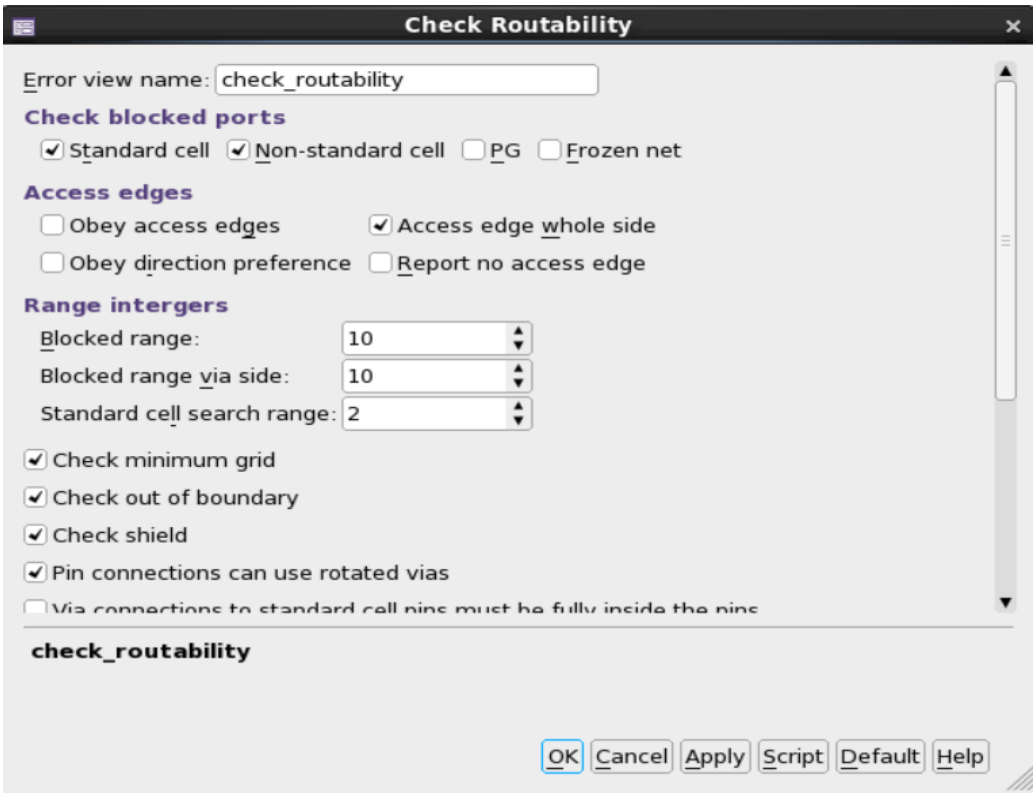

Fig. 9. Dialog box of check routeability

Dialog box of ignored layers is shown in Fig. 10.

```
icc2_shell> set_ignored_layers -min_routing_layer M2 -max_routing_layer M6
```

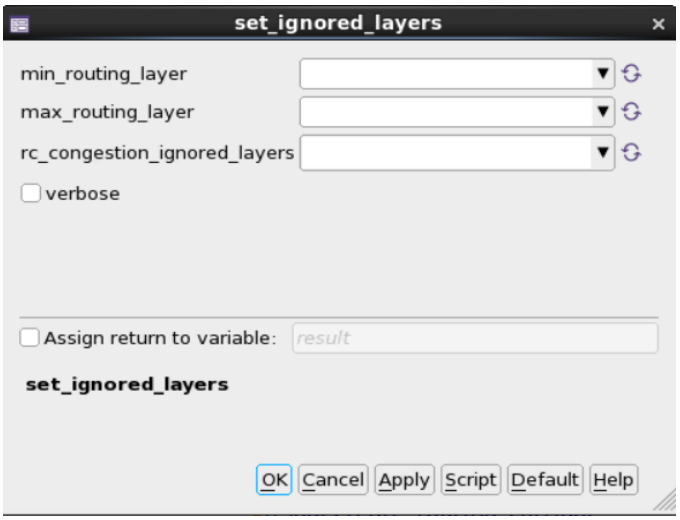

Fig. 10. Dialog box of ignored layers

Check the ignored layout by using the following command:

```
icc2_shell> report_ignored_layers
```

In this lab for signal nets routing and optimization *route_opt* command is used.

```
icc2_shell> route_opt
```

Design after routing and optimization is shown in Fig. 11.

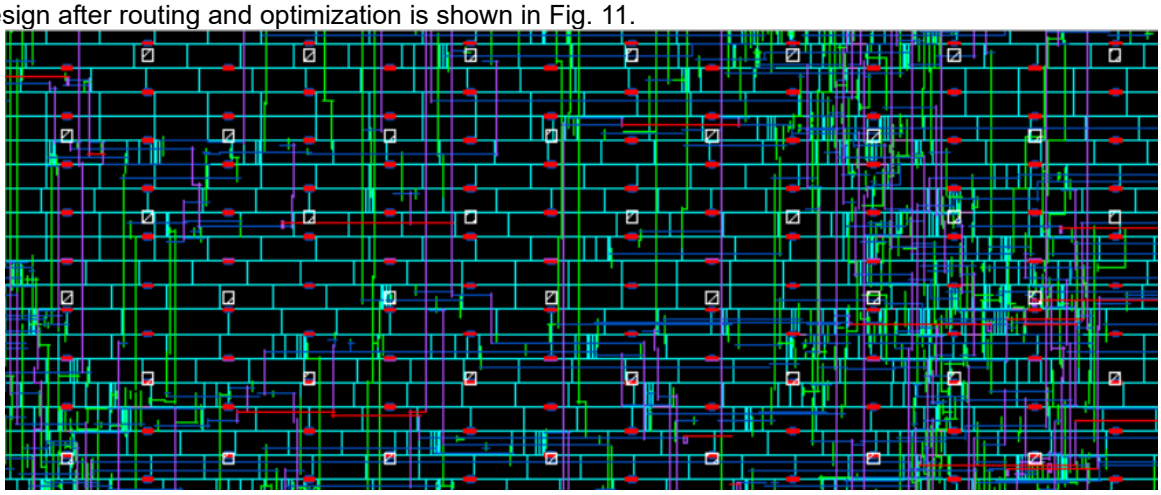

Fig. 11. Design after routing and optimization

11. Routing optimization

The following command will optimize the routing after the initial routing

```
icc2_shell> connect_pg_net -net VDD [get_pins -physical_context */VDD]
icc2_shell> connect_pg_net -net VSS [get_pins -physical_context */VSS]
icc2_shell> optimize_routes -max_detail_route_iterations 5
```

12. Checking for problems and reports

Run the check layout-versus-schematic to see if there is any problem after the routing:

```
icc2_shell> check_lvs -max_errors 2000
```

After that, we can write out the design for the next step and the reports after routing:

```
icc2_shell> set_app_option -name time.snapshot_storage_location -value "./"
icc2_shell> create_qor_snapshot -name route -significant_digits 4
icc2_shell> report_congestion
icc2_shell> write_verilog -include {pg_netlist} "${DESIGN_NAME}_routed.v"
icc2_shell> report_qor_snapshot > reports/route.qor_snapshot.rpt
icc2_shell> report_qor > reports/route.qor
icc2_shell> report_constraints -all_violators > reports/route.con
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets \
                  -delay_type max > reports/route.max.tim
icc2_shell> report_timing -capacitance -transition_time -input_pins -nets \
                  -delay_type min > reports/route.min.tim
```