Advance SOC Lab3 - Synopsys Flow

Group3/Name	Github link
陳冠晰/Vic	Vic Chen (https://github.com/vic9112/)
王彥智/Kenny	Yanzhi Wang (https://github.com/kenny0915)
陳柏翰/Lukas	Lukas Chen (https://github.com/lukaschen1010)

Table of Contents:

- Advance SOC Lab3 - Synopsys Flow
  - Baseline of Cell-Based Design Flow:
  - Front-End (Design Compiler)
    - 1. Synthesis
    - 2. Floorplan
  - Back-End (IC Compiler 2, StarRC, PrimeTime)
    - 3. Placement & Route
    - 4. StarRC
    - 5. PrimeTime
    - 6. Chip Finishing
  - Verificaion (IC Validator, Formality)
    - 7. IC Validator - DRC
    - 8. IC Validator - LVS
    - 9. Formality
  - Discussion and Observation

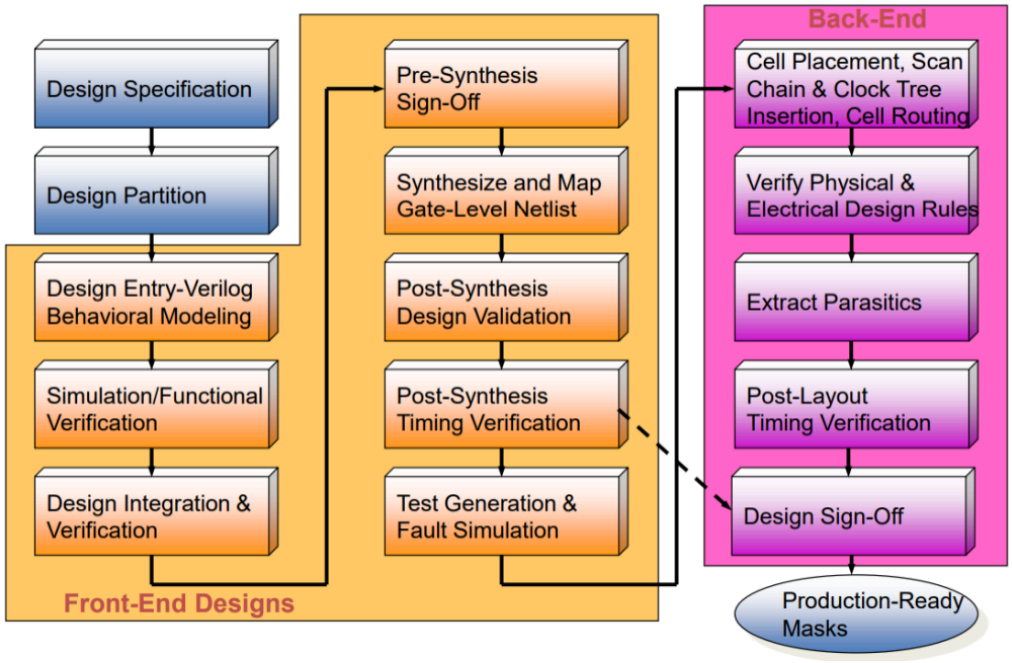
Reference:

Synopsys User Manual

(https://drive.google.com/drive/u/1/folders/1tm9_EBIIaClZw2gbKs5kXTk8pARAAcHy)

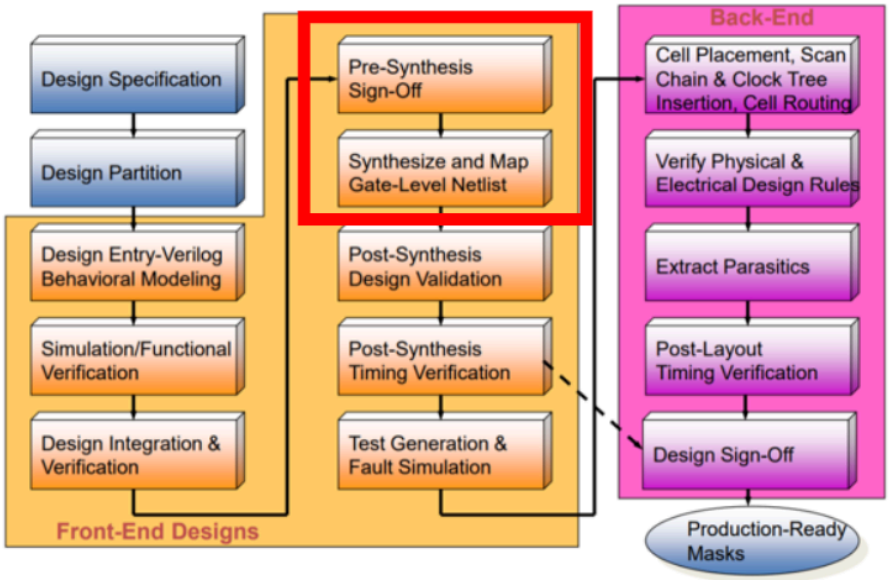
for description of the script

Baseline of Cell-Based Design Flow:

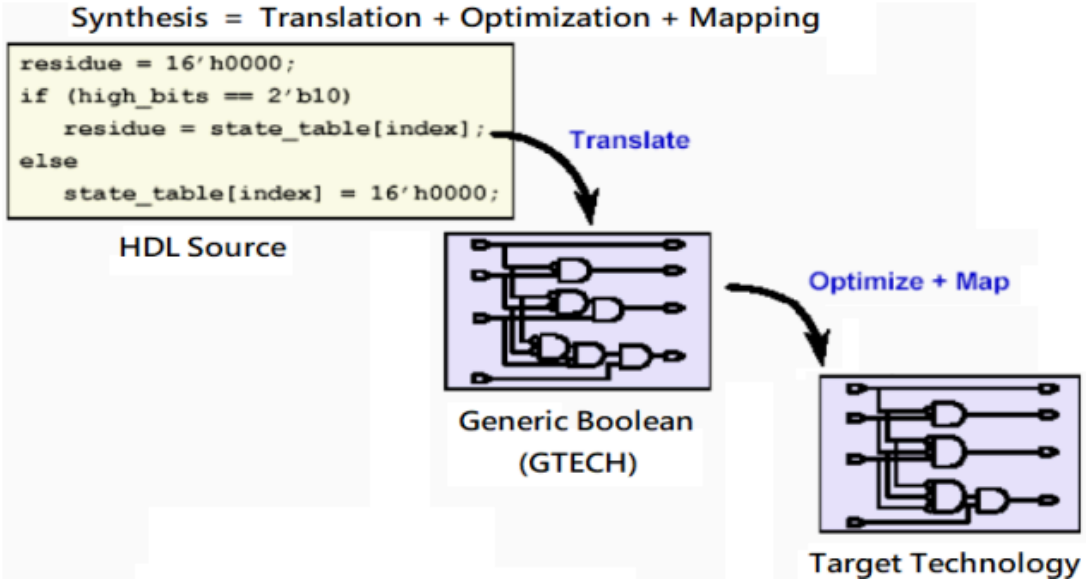


Front-End (Design Compiler)

1. Synthesis



- **Goal:** Analyze and synthesis HDL code `ic2_master_top.v` from RTL to gate-netlist



Description of command in scripts (tcl file)

`compile_for_timing.tcl`

1. Common Script Source:

```
1 source ../../common/common.tcl
```

- This command loads common settings or procedures from the common.tcl file located in the common directory. It's used to maintain consistency in environment settings across various scripts .

2. Library Settings:

```
3 set link_library "$Std_cell_lib $Ram_lib"
4 set target_library "$Std_cell_lib $Ram_lib"
```

- These settings define the libraries used for linking and targeting during synthesis, ensuring that the tool uses the correct cell libraries for optimization and mapping. (refer to `dcug.pdf`)

3. RTL Power Gating:

```
6 set dc_allow_rtl_pg true
```

- This enables power gating at the RTL level, allowing the tool to consider power-saving techniques early in the design proces. (refer to `dcug.pdf`)

4. Design Analysis and Elaboration:

```
source $analyze_script
elaborate ${DESIGN_NAME} -architecture verilog -library WORK
```

- Loads and executes an analysis script, then elaborates the specified design using Verilog in a working library. This prepares the design for further synthesis by building an internal representation. (refer to `pwcug.pdf`)

5. Current Design Setting:

```
current_design ${DESIGN_NAME}
```

- Specifies the current design context for all subsequent commands to operate on. (refer to `dcug.pdf`)

6. Link and Constraints Application:

```
link
source $Constraints_file
```

- Resolves all external references and loads timing or other constraints from a specified constraints file, critical for correct synthesis behavior. (refer to `preug.pdf`)

7. Fix Multiple Port Nets:

```
set_fix_multiple_port_nets -outputs -feedthroughs
```

- Adjusts settings to handle nets with multiple port connections effectively, ensuring signal integrity and logical correctness. Also, loads a file containing configuration for handling synthesis warnings. (refer to `dcug.pdf`)

8. Warning Management and Don't Use Settings:

```
source $Warning_file
source ../scripts/set_dont_use.tcl
```

- Sources a file that likely contains configurations for managing synthesis warnings and another script to specify cells that should not be used in the synthesis (`set_dont_use.tcl`).

9. Design Check and Re-linking:

```
check_design
link
```

- Performs a check to ensure the design is correct and consistent post-modifications, and then performs a linking process again to resolve any new or outstanding references. (refer to tcoug.pdf)

10. Compile Design:

```
compile -exact_map -map_effort high -area_effort medium -power_effort non
```

- This command compiles the design with specific focus on mapping accuracy, effort levels for mapping and area optimization, and no specific efforts on power optimization.

11. Reporting and Output Generation:

```
report_timing > ../../reports/timing_${DESIGN_NAME}_timing_reports.log
report_qor > ../../reports/timing_${DESIGN_NAME}_qor_reports.log
report_area -hierarchy  > ../../reports/timing_${DESIGN_NAME}_area_report
report_power -hierarchy > ../../reports/timing_${DESIGN_NAME}_power_repor
```

- Detailed reporting commands generate logs and reports about timing, quality of results (QoR), area, and power, which are directed to specified log files in the `reports` directory. This helps in evaluating the synthesis outcome and making necessary adjustments. (refer to `dcug.pdf`)

12. File and Design Management:

```
change_names -rules verilog
write_file -format verilog -hierarchy -pg -output ../../input/${DESIGN_NA
quit
```

- This script changes names according to Verilog rules, writes out the hierarchical Verilog file with power gating (`-pg`) enabled, and then exits the tool. (refer to `dcug.pdf`)

Outputs on Terminal

- Beginning Mapping Optimizations

```
                                  TOTAL
ELAPSED            WORST NEG   SETUP    DESIGN
 TIME      AREA      SLACK     COST    RULE COST         ENDPOINT
--------- --------- --------- --------- --------- -------------------------
  0:00:06     412.7      0.00       0.0       9.5
  0:00:06     412.7      0.00       0.0       9.5
  0:00:06     412.7      0.00       0.0       9.5
  0:00:06     412.7      0.00       0.0       9.5
  0:00:27     412.7      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
```

- Area-Recovery Phase (cleanup)

```
                                  TOTAL
ELAPSED            WORST NEG   SETUP    DESIGN
 TIME      AREA      SLACK     COST    RULE COST         ENDPOINT
--------- --------- --------- --------- --------- -------------------------
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     331.5      0.00       0.0       9.5
  0:00:27     330.4      0.00       0.0       9.5
  0:00:27     330.1      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     329.8      0.00       0.0       9.5
  0:00:27     326.0      0.00       0.0       9.5
```

Reports

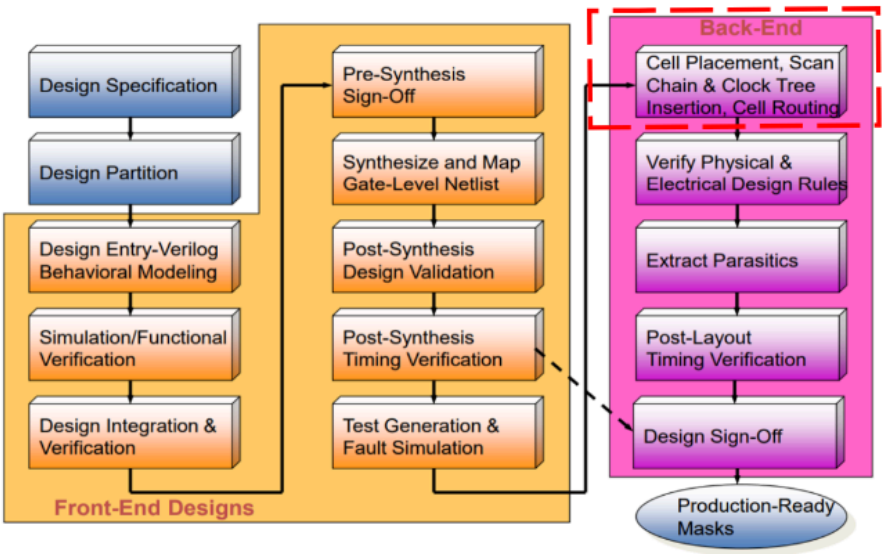
- Area Report

```
Number of ports:                          185
Number of nets:                           882
Number of cells:                          719
Number of combinational cells:            562
Number of sequential cells:               153
Number of macros/black boxes:               0
Number of buf/inv:                        175
Number of references:                      20

Combinational area:                161.482800
Buf/Inv area:                       32.500800
Noncombinational area:             164.546405
Macro/Black Box area:                0.000000
Net Interconnect area:             485.972168

Total cell area:                   326.029204
Total area:                        812.001372
```

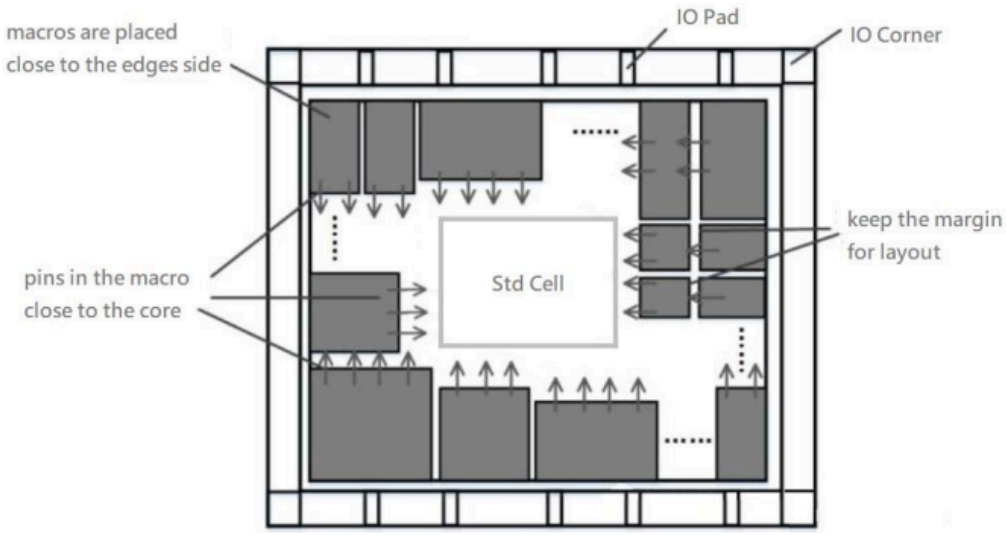
- Timing Report

```
---------------------------------------------------------------------
data required time                                               1.69
data arrival time                                               -0.38
---------------------------------------------------------------------
slack (MET)                                                      1.31
```

2. Floorplan



- **Goal:** Design initiation and chip floor plan



Description of command in scripts (tcl file)

`step1_data_setup.tcl`

1. Common Script Source:

```
source ../../common/common.tcl
```

- This loads common configurations or procedures that are shared across multiple scripts within the project. This practice is typical in design environments to ensure consistency in settings and variables.

2. Set Library:

```
3 set link_library "$Std_cell_lib $Ram_lib"
4 set target_library "$Std_cell_lib $Ram_lib"
```

- These commands define the libraries used for linking and targeting during the synthesis process. The link_library is crucial for resolving references during synthesis, while target_library specifies the library against which the design will be mapped. (refer to `dcug.pdf`)

3. Library Creation:

```
6 create_lib $ARC_TOP -technology $Tech_file -ref_libs $REFERENCE_LIB
```

- This script segment creates a new library ($ARC_TOP) with specified technology and reference libraries. It sets up the necessary library framework for the floorplan and further design stages. (refer to `dcug.pdf`)

4. Set Tluplus Files:

```
8 read_parasitic_tech -tlup "$Tlup_max_file $Tlup_min_file" -layermap $Map_
```

- Reads and sets up the technology files for handling parasitics in the design, crucial for accurate delay and power analysis during later synthesis stages. (refer to `tcoug.pdf`)

5. Import Design:

```
10 read_verilog "$Core_compile"
11 current_design ${DESIGN_NAME}
12 source $Constraints_file
```

- This portion loads the Verilog design, sets it as the current design context for operations, and applies constraints, which are essential for guiding the synthesis process according to design specifications. (refer to `preug.pdf`)

6. Save Block:

```
14 save_block -as ${DESIGN_NAME}_1_data_setup
15 save_lib
16 close_block
17 close_lib
18 exit
```

- Saves the configured design block under a specific setup, ensuring that all changes are preserved. It also closes the library and block, securing all modifications before exiting the session. This step is critical for maintaining design integrity and preventing data loss. (refer to `dcug.pdf`)

`step2_floorplan.tcl`

1. Common Script Source:

```
1  source ../../common/common.tcl
```

2. Set Library:

```
3  set link_library "$Std_cell_lib $Ram_lib"
4  set target_library "$Std_cell_lib $Ram_lib"
```

- These commands configure the synthesis tool to use standard cell libraries and RAM libraries as both link and target libraries, essential for resolving logical to physical mappings. (refer to `dcug.pdf`)

3. Open Library and Copy Block:

```
6  open_lib $ARC_TOP
7  copy_block -from_block ${DESIGN_NAME}_1_data_setup -to temp_data_setup
8  open_block temp_data_setup
```

- Opens a library for operations, copies a block from one name to another, and opens it for further modifications, part of setting up the environment for subsequent design edits. (refer to `dcug.pdf`)

4. Reports:

```
11 report_clocks -skew -attributes
12 report_exceptions
13 report_disable_timing
```

- These commands generate reports on clock skew, attributes, design exceptions like setup/hold violations, and disables timing checks, providing insights into timing issues and exceptions within the design. (refer to `dcug.pdf`)

5. Set Power/Ground Nets and Pins:

```
15 set power "VDD"
16 set ground "VSS"
17 set powerPort "VDD"
18 set groundPort "VSS"
19 set ndm_logic0_net "VSS"
20 set ndm_logic1_net "VDD"
```

- Configures the design's power and ground nets and assigns ports, which are critical for the proper electrical functionality and power integrity of the chip. (refer to `dcug.pdf`)

6. Set Options and Group Paths:

```
22 set_app_option -name time.disable_recovery_removal_checks -value false
23 set_app_option -name time.disable_case_analysis -value false
24 group_path -name INPUT  -from [all_inputs]
25 group_path -name OUTPUT -to   [all_outputs]
26 group_path -name COMBO -from [all_inputs] -to [all_outputs]
```

- Adjusts specific tool options and groups paths for more focused timing analysis, which helps in optimizing the synthesis process and managing design timing more efficiently. (refer to `smvfug.pdf`)

7. Save Block:

```
29 save_block -as temp_floorplan_init
```

- Saves the current state of the block under a new name to preserve any changes made during the session for later retrieval or further modifications. (refer to `pwcug.pdf`)

8. Defining Preferred Routing Directions:

```
31 set_ignored_layers -min_routing_layer ${route_min_layer} -max_routing_lay
32 set_attribute [get_layers M1] routing_direction vertical
33 set_attribute [get_layers M2] routing_direction horizontal
34 set_attribute [get_layers M3] routing_direction vertical
35 set_attribute [get_layers M4] routing_direction horizontal
36 set_attribute [get_layers M5] routing_direction vertical
37 set_attribute [get_layers M6] routing_direction horizontal
38 set_attribute [get_layers M7] routing_direction vertical
39 set_attribute [get_layers {M1}] track_offset 0.037
```

- Specifies routing preferences for different metal layers to optimize the physical layout for electrical performance and manufacturability, including setting track offsets for specific technology libraries. (refer to `tcoug.pdf`)

9. Set Track Offset

```
42 set_attribute [get_layers {M1}] track_offset 0.037
```

- Adjusts the track offset for the first metal layer (`M1`), a parameter critical for ensuring correct alignment and spacing of routing tracks within the specified technology constraints. (refer to tcoug.pdf)

10. Create Floorplan:

```
44 initialize_floorplan -core_utilization 0.2 -core_offset {10 10 10 10}
```

- Initializes the floorplan with specified core utilization and offsets, setting up the foundational layout parameters for the design, which affects subsequent placement and routing stages(refer to `smvfug.pdf`)

11. Ports Placement:

```
46 place_pins -ports [get_ports *]
```

- Places pins according to the port definitions in the design, crucial for ensuring optimal connectivity and access for signals entering and exiting the chip. (refer to `smvfug.pdf`)

12. Defining Power/Ground Nets and Pins:

```
48 set_attribute -objects [get_nets VDD] -name net_type -value power
49 set_attribute -objects [get_nets VSS] -name net_type -value ground
50 check_mv_design
```

- Assigns net types to power (VDD) and ground (VSS) nets and checks the multi-voltage design setup to ensure correct power distribution and integrity across different voltage domains. (refer to `smvfug.pdf`)

13. Save Block:

```
53 save_block -as temp_floorplane
```

- Saves the current state of the design under a new name after floorplanning adjustments, preserving all changes made during this phase. (refer to `smvfug.pdf`)

14. Create Floorplan Placement:

```
55 create_placement -floorplan -effort high -timing_driven
56 legalize_placement
57 route_global -congestion_map_only true -effort high
```

- Initiates the placement process with high effort and timing-driven considerations, legalizes the placement to meet design rules, and performs a preliminary global routing to identify potential congestion areas, all crucial for optimizing the design for performance and manufacturability. (refer to `smvfug.pdf`)

15. Report Placement:

```
59 report_placement
```

- Generates a report detailing the results of the placement process, providing insights into the effectiveness of the placement strategy and potential areas for improvement. (refer to `decug.pdf`)

16. Final Save Blocks:

```
61 save_block -as temp_floorplane_placed
62 save_block -as ${DESIGN_NAME}_2_floorplan_ends
```

- Saves the design under multiple names at different stages, ensuring that each stage of the floorplanning and placement process is captured and can be revisited or rolled back if necessary. (refer to `pwcug.pdf`)

17. Finalize and Exit:

```
64 save_lib
65 close_block
66 close_lib
67 exit
```

- Completes the session by saving the library, closing the current block and library, and exiting the tool, ensuring that all data is properly stored and the environment is cleanly terminated. (refer to `decug.pdf`)

`step3_powerplan.tcl`

1. Common Script Source and Library Operations:

```
source ../../common/common.tcl
open_lib $ARC_TOP
copy_block -from_block ${DESIGN_NAME}_2_floorplan_ends -to temp_floorplan
open_block temp_floorplan_ends
```

- This script sources a common configuration file, opens a library (**$ARC_TOP**), copies a block from ${DESIGN_NAME}_2_floorplan_ends to temp_floorplan_ends, and opens the new block for modification. (refer to `dcug.pdf`)

2. Power Planning:

```
set_attribute -objects [get_nets VDD] -name net_type -value power
set_attribute -objects [get_nets VSS] -name net_type -value ground
connect_pg_net -net VDD [get_pins -physical_context */VDD]
connect_pg_net -net VSS [get_pins -physical_context */VSS]
```

- Sets the net attributes for VDD and VSS to power and ground respectively, and connects power and ground nets to their corresponding pins in the physical context. (refer to `dcug.pdf`)

3. Create Power Plan:

```
remove_pg_via_master_rules -all
remove_pg_patterns -all
remove_pg_strategies -all
remove_pg_strategy_via_rules -all
```

- Removes all existing power grid (PG) via master rules, patterns, strategies, and strategy via rules to prepare for a fresh setup of power strategies. (refer to `dcug.pdf`)

4. Create Standard Cells Rail:

```
create_pg_std_cell_conn_pattern M1_rail -layers {M1} -rail_width {@wtop @
set_pg_strategy M1_rail_strategy_pwr -core -pattern {{name: M1_rail} {net
set_pg_strategy M1_rail_strategy_gnd -core -pattern {{name: M1_rail} {net
compile_pg -strategies M1_rail_strategy_pwr -ignore_drc
compile_pg -strategies M1_rail_strategy_gnd -ignore_drc
```

- Creates a connection pattern for standard cells using M1 layer rails, sets power and ground strategies for these rails, and compiles the PG strategies, ignoring

design rule checks (DRC). (refer to `dcug.pdf`)

5. Create Top Vertical and Horizontal Mesh:

```
create_pg_mesh_pattern TOP_MESH_VERTICAL ...
set_pg_strategy VDDVSS_TOP_MESH_VERTICAL ...
compile_pg -strategies {VDDVSS_TOP_MESH_VERTICAL}
create_pg_mesh_pattern TOP_MESH_HORIZONTAL ...
set_pg_strategy VDDVSS_TOP_MESH_HORIZONTAL ...
compile_pg -strategies {VDDVSS_TOP_MESH_HORIZONTAL}
```

- Defines vertical and horizontal mesh patterns for the top layers of the design and sets strategies to manage power and ground distribution effectively across these meshes. (refer to `dcug.pdf`)

6. Create Rectangular Rings:

```
create_pg_ring_pattern ...
set_pg_strategy RING -core -pattern {{ name: ring_pattern} { nets: "VDD V
compile_pg -strategies RING
check_pg_connectivity -nets "VDD VSS"
```

- Creates a ring pattern for power and ground distribution, sets up a strategy for the rings, compiles the PG strategy, and checks connectivity for power integrity. (refer to `dcug.pdf`)

7. Finalize and Save:

```
save_block -as ${DESIGN_NAME}_3_powerplan_ends
save_lib
close_block
close_lib
exit
```

- Saves the block with the final power planning configurations, saves the library, closes the block and library, and exits the session to ensure all changes are preserved. (refer to `dcug.pdf`)

Outputs on Terminal

- Step 1

```
################################################################################
Set_Tluplus_Files
read_parasitic_tech -tlup "$Tlup_max_file $Tlup_min_file"                    -l
ayermap $Map_file
1
################################################################################
Import_Design
read_verilog "$Core_compile"
Loading verilog file '/home/course/u110590022/SOC/lab_snps_flow/lab_formal_relea
se/input/i2c_master_top.v'
Information: Reading Verilog into new design 'i2c_master_top' in library 'i2c_ma
ster_top'. (VR-012)
Number of modules read: 5
Top level ports: 35
Total ports in all modules: 185
Total nets in all modules: 890
Total instances in all modules: 719
Elapsed = 00:00:00.02, CPU = 00:00:00.01
1
current_design ${DESIGN_NAME}
{i2c_master_top:i2c_master_top.design}
source $Constraints_file
Using libraries: i2c_master_top saed14rvt_tt0p8v25c saed14rvt_tt0p8v25c_physical
_only EXPLORE_physical_only
Linking block i2c_master_top:i2c_master_top.design
Information: User units loaded from library 'saed14rvt_tt0p8v25c' (LNK-040)
Design 'i2c_master_top' was successfully linked.
1
################################################################################
Save_Block
save_block -as ${DESIGN_NAME}_1_data_setup
Information: Saving 'i2c_master_top:i2c_master_top.design' to 'i2c_master_top:i2
c_master_top_1_data_setup.design'. (DES-028)
1
save_lib
Saving library 'i2c_master_top'
```

- Step 2

```
****************************************
Report : report_placement
Design : i2c_master_top
Version: R-2020.09-SP3
Date   : Fri Apr 26 15:17:19 2024
****************************************

  Wire length report (all)
  ==================
  wire length in design temp_data_setup: 4354.517 microns.
  wire length in design temp_data_setup (see through blk pins): 4354.517 microns
.

  Physical hierarchy violations report
  ====================================
  Violations in design temp_data_setup:
     0 cells have placement violation.

  Voltage area violations report
  ==================================
  Voltage area placement violations in design temp_data_setup:
     0 cells placed outside the voltage area which they belong to.
```

- Step 3

```
check_pg_connectivity -nets "VDD VSS"
Loading cell instances...
Number of Standard Cells: 715
Number of Macro Cells: 0
Number of IO Pad Cells: 0
Number of Blocks: 0
Loading P/G wires and vias...
Number of VDD Wires: 58
Number of VDD Vias: 144
Number of VDD Terminals: 41
Number of VSS Wires: 58
Number of VSS Vias: 144
Number of VSS Terminals: 41
**************Verify net VDD connectivity*****************
  Number of floating wires: 34
  Number of floating vias: 0
  Number of floating std cells: 715
  Number of floating hard macros: 0
  Number of floating I/O pads: 0
  Number of floating terminals: 1
  Number of floating hierarchical blocks: 0
************************************************************
**************Verify net VSS connectivity*****************
  Number of floating wires: 34
  Number of floating vias: 0
  Number of floating std cells: 715
  Number of floating hard macros: 0
  Number of floating I/O pads: 0
  Number of floating terminals: 1
  Number of floating hierarchical blocks: 0
************************************************************
```

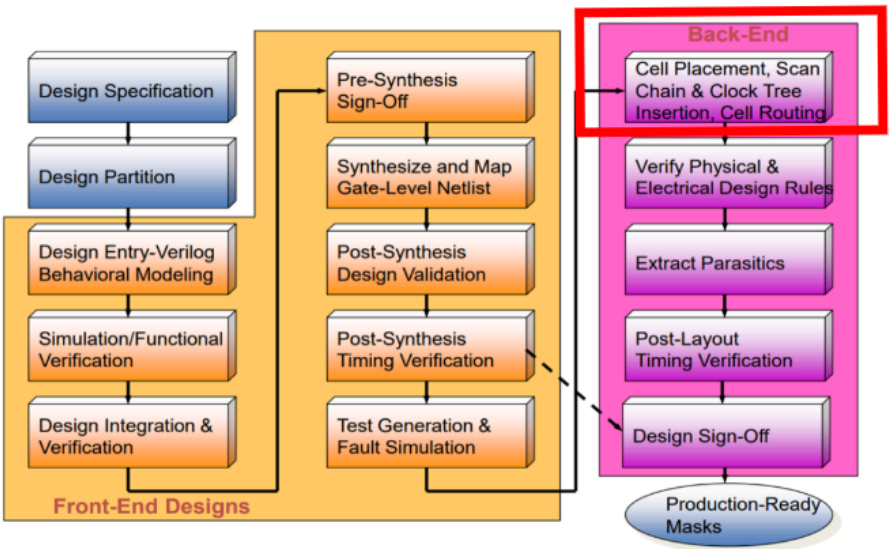
Report

- Placement Report

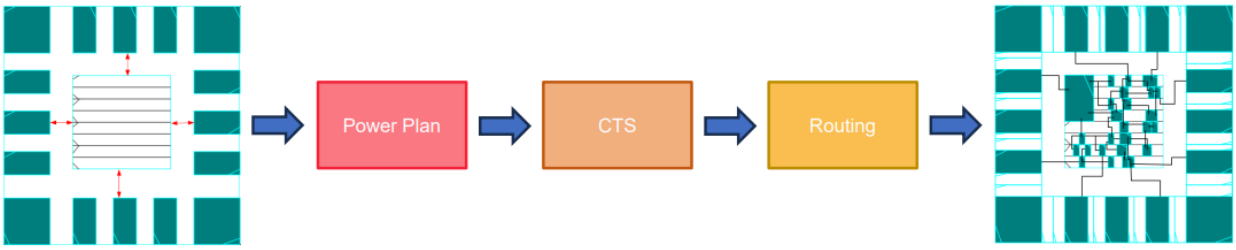
```
TOTAL 0 Violations.

VIOLATIONS BY SUBCATEGORY:
     MOVABLE  APP-FIXED USER-FIXED  DESCRIPTION

         0          0          0    Two objects overlap.
           0          0          0    Two cells overlap.
           0          0          0    Two cells have overlapping keepout margins.
           0          0          0    A cell overlaps a blockage.
           0          0          0    A cell keepout margin overlaps a blockage.

         0          0          0    A cell violates a pnet.

         0          0          0    A cell is illegal at a site.
           0          0          0    A cell violates pin-track alignment rules.
           0          0          0    A cell is illegal at a site.
           0          0          0    A cell violates legal index rule.
           0          0          0    A cell has the wrong variant for its location.

         0          0          0    A cell is not aligned with a site.
           0          0          0    A cell is not aligned with the base site.
           0          0          0    A cell is not aligned with an overlaid site.

         0          0          0    A cell has an illegal orientation.

         0          0          0    A cell spacing rule is violated.
           0          0          0    A spacing rule is violated in a row.
           0          0          0    A spacing rule is violated between adjacent rows.
           0          0          0    A cell violates vertical abutment rule.
           0          0          0    A cell violates metal spacing rule.

         0          0          0    A layer rule is violated.
           0          0          0    A layer VTH rule is violated.
           0          0          0    A layer OD rule is violated.
           0          0          0    A layer OD max-width rule is violated.
           0          0          0    A layer ALL_OD corner rule is violated.
           0          0          0    A layer max-vertical-length rule is violated.
           0          0          0    A layer TPO rule is violated.
           0          0          0    Filler cell insertion cannot satisfy layer rules.

         0          0          0    A cell is in the wrong region.
           0          0          0    A cell is outside its hard bound.
           0          0          0    A cell is in the wrong voltage area.
           0          0          0    A cell violates an exclusive movebound.

         0          0          0    Two cells violate cts margins.

         0          0          0    Two cells violate coloring.
```

Back-End (IC Compiler 2, StarRC, PrimeTime)

3. Placement & Route



- **Goal:** `macro/std-cell` placement and route



Description of command in scripts (tcl file)

`step4_place.tcl`

1. Common Script Source:

```
source ../../common/common.tcl
```

2. Library Settings:

```
set link_library "$Std_cell_lib $Ram_lib"
set target_library "$Std_cell_lib $Ram_lib"
```

- These commands specify the libraries to be used by the compiler for linking and targeting, essential for the synthesis process to know where to fetch standard cell and RAM definitions.
- Reference: Refer to the DCUG section on defining libraries for synthesis

3. Script for Lab #8 Initialization:

```
open_lib $ARC_TOP
copy_block -from_block ${DESIGN_NAME}_3_powerplan_ends -to temp_powerplan
open_block temp_powerplan_ends
```

- These commands open the specified library, copy a design block from a previous stage for modifications, and open this block for subsequent operations, setting up the environment for lab-specific tasks.
- Reference: Refer to the user guide section on library management and block operations

4. General Optimization Settings:

```
set_app_options -name time.disable_recovery_removal_checks -value false
set_app_options -name time.disable_case_analysis -value false
set_app_options -name place.coarse.continue_on_missing_scandef -value tru
```

- These settings configure application options for the synthesis tool to enhance the optimization process, including enabling recovery removal checks and case analysis, and handling missing scan definitions during placement.
- Reference: IC Compiler II Timing Analysis User Guide (U-2022.12-SP4)

5. Place Optimization Settings:

```
set_app_options -name opt.common.user_instance_name_prefix -value place
set_app_options -name place.coarse.congestion_driven_max_util -value 0.6
set_app_options -name place.coarse.max_density -value 0.2
```

- Sets various placement optimization parameters to manage naming conventions, congestion, and density during the placement process, aimed at improving layout and performance.
- Reference: IC Compiler II Design Planning User Guide for information on placement settings and optimization techniques.

6. Additional Scripts and Placement Operations:

```
source ./../scripts/mcmm.tcl
source ./../scripts/set_dont_use.tcl
place_opt
legalize_placement
```

- Sources additional scripts for Multi-Corner Multi-Mode (MCMM) setup and cell exclusion, followed by placement optimization and legalization to ensure the design meets all physical and performance constraints.
- Reference: IC Compiler II Timing Analysis User Guide for sections detailing the creation and interpretation of quality of results and timing reports.

7. Reports and Quality of Results:

```
create_qor_snapshot -name place_qor_snp -significant_digits 4
report_qor_snapshot -name place_qor_snp > ../../reports/place.qor_snapsho
report_qor > ../../reports/place.qor
report_constraints -all_violators > ../../reports/place.con
report_timing -capacitance -transition_time -input_pins -nets -delay_type
report_timing -capacitance -transition_time -input_pins -nets -delay_type
```

- Generates various reports detailing the quality of results, constraint violations, and timing for maximum and minimum delay scenarios, providing comprehensive feedback on the optimization process.
- Reference: Refer to the user guide section on saving and managing design checkpoints.

8. Finalize and Save Block:

```
save_block -as ${DESIGN_NAME}_4_place_ends
save_lib
close_block
close_lib
exit
```

- Saves the current state of the block and library, ensuring all modifications are preserved, and closes the session cleanly.
- Reference: Look for sections in the user guide discussing the proper exit or cleanup procedures post-synthesis or optimization.

step5_clock_tree_synthesis.tcl

1. Common Script Source:

```
source ../../common/common.tcl
```

2. Set Library:

```
set link_library "$Std_cell_lib $Ram_lib"
set target_library "$Std_cell_lib $Ram_lib"
```

- Defines the libraries used by the synthesis tool for linking and targeting. This step is critical for setting up the reference libraries that the synthesizer will use to map the logical design into physical implementations.
- Reference: See the section on library configuration in the ICC2 Data Model User Guide .

3. Open Library and Copy Block:

```
open_lib $ARC_TOP
copy_block -from_block ${DESIGN_NAME}_4_place_ends -to temp_place_ends
open_block temp_place_ends
```

- Opens a design library and manages design data within a session. This script copies a previously completed block for further modification and opens it for subsequent operations, which is a common practice in iterative design processes.
- Reference: Described in the chapters related to libraries and blocks management in the user guides.

4. Set Clock Tree Options:

```
check_legality -verbose > ../../reports/place_report.rpt
set_ignored_layers -min_routing_layer ${route_min_layer} -max_routing_lay
set_app_options -name cts.compile.enable_cell_relocation -value all
set_app_options -name cts.compile.size_pre_existing_cell_to_cts_reference
set_clock_tree_options -clocks [all_clocks] -target_skew 0.1
```

- Configures the clock tree synthesis options, such as setting target skew and enabling cell relocation for clock tree optimization. This also includes checking the legality of the placement and configuring layer usage for routing, which are essential for ensuring that the design meets timing and manufacturability requirements.
- Reference: Details on clock tree synthesis settings can be found in the ICC2 Clock Tree Synthesis documentation.

5. Set Clock Tree References:

```
15 set_lib_cell_purpose -include cts {list of cells}
```

- Specifies which library cells are to be used in the clock tree, ensuring that the synthesizer uses only specified cells for constructing the clock network, which is crucial for controlling clock skew and insertion delay.
- Reference: For instructions on constraining the use of library cells, see the appropriate user guide section.

6. Create Routing Rule and Set Clock Routing Rules:

```
17 set_clock_uncertainty 0.1 [all_clocks]
18 create_routing_rule CLK_SPACING -spacings {M2 0.3 M3 0.5 M4 0.7}
19 set_clock_routing_rules -rules CLK_SPACING -min_routing_layer M2 -max_rou
```

- These commands define custom routing rules and assign them to clock nets before CTS.
- Reference: Refer to the section on custom routing rules in the user guides for details

7. Report Clock Settings:

```
20 report_clock_settings
```

- Outputs the current configuration settings used by the clock synthesis processes.
- Reference: Information on reporting clock settings can be found in the user guides

8. Clock Optimization:

```
21 clock_opt -from build_clock -to build_clock
```

- Optimizes the clock network post-CTS to meet the design requirements.
- Reference: Described in the Clock Tree Synthesis documentation

9. Quality of Results (QoR) Snapshots and Reports:

```
create_qor_snapshot -name clock_pre_route -significant_digits 4
report_qor_snapshot -name clock_pre_route > ../../reports/clock_pre_route
report_qor > ../../reports/clock_pre_route.qor
report_constraints -all_violators > ../../reports/clock_pre_route.con
report_timing -capacitance -transition_time -input_pins -nets -delay_type
report_timing -capacitance -transition_time -input_pins -nets -delay_type
```

- Generates a snapshot of the design's quality of results and reports various design metrics.
- Reference: Refer to the QoR snapshot creation and reporting sections in the user guides

10. Connecting Power/Ground Nets and Finalizing the Design:

```
connect_pg_net -net VDD [get_pins -physical_context */VDD]
connect_pg_net -net VSS [get_pins -physical_context */VSS]
save_block -as ${DESIGN_NAME}_5_clock_ends
save_lib
close_block
close_lib
exit
```

- Connects the power and ground nets to their respective pins, which is crucial for power integrity.
- Reference: The connect_pg_net command and its usage are detailed in the user guides

step6_route.tcl

1. Common Script Source:

```
source ../../common/common.tcl
```

2. Set Library:

```
set link_library          "$Std_cell_lib $Ram_lib"
set target_library        "$Std_cell_lib $Ram_lib"
```

- Sets the link and target libraries to the specified standard cell and RAM libraries, which are essential for the synthesis process.
- Reference: DCUG Section on "Specifying Logic Libraries".

3. Open Library and Copy Block:

```
open_lib $ARC_TOP
copy_block -from_block  ${DESIGN_NAME}_5_clock_ends -to temp_clock_ends
open_block temp_clock_ends
```

- The `open_lib` command opens a library and `copy_block` duplicates a block for subsequent modifications. The open_block command then opens the specified block for editing.
- Reference: Refer to IC Compiler II Implementation User Guide for details on library management .

4. Set Ignored Layers for Routing:

```
set_ignored_layers -min_routing_layer  ${route_min_layer} -max_routing_la
report_ignored_layers
```

- This section involves setting the ignored layers for routing and reporting them, which could affect the RC extraction and congestion analysis.
- Reference: For setting and reporting ignored layers, see the IC Compiler II Implementation User Guide .

5. Route Optimization:

```
source ./../scripts/mcmm.tcl

set_app_option -name route.detail.ignore_drc -value { }

#track assignment & initial route optimize
route_auto

#route optimize
route_opt
```

- Involves running Multi-Corner Multi-Mode (MCMM) analysis and configuring route optimization settings to ignore certain design rule checks (DRCs) during automatic routing.
- Reference: For route optimization techniques, refer to IC Compiler II Design Planning User Guide .

6. Connecting Power/Ground Nets and Pins:

```
connect_pg_net -net VDD [get_pins -physical_context */VDD]
connect_pg_net -net VSS [get_pins -physical_context */VSS]

optimize_routes -max_detail_route_iterations 5

check_lvs -max_errors 2000
```

- This step involves connecting power (VDD) and ground (VSS) nets to their respective pins and optimizing the routes for these connections.
- Reference: See IC Compiler II Implementation User Guide for details on connecting power and ground nets .

7. Reports:

```
set_app_option -name time.snapshot_storage_location -value "./"
create_qor_snapshot -name route -significant_digits 4
report_congestion
write_verilog -include {pg_netlist} "$Top_design_pt"

report_qor_snapshot > ../../reports/route.qor_snapshot.rpt
report_qor > ../../reports/route.qor
report_constraints -all_violators > ../../reports/route.con
report_timing -capacitance -transition_time -input_pins -nets -delay_type
report_timing -capacitance -transition_time -input_pins -nets -delay_type
```

- Commands to create Quality of Results (QoR) snapshots and generate various reports on design congestion, QoR, and constraints.
- Reference: Refer to the Timing Analysis User Guide for information on QoR snapshots and reporting congestion .

8. Save Cell:

```
save_block -as ${DESIGN_NAME}_6_complete
save_lib
```

- Saves the current state of the block and library, preserving all modifications, constraints, and settings.
- Reference: DCUG Section on "Saving the Design and ASCII Export".

9. GDS Output:

```
write_verilog -exclude {physical_only_cells} ../../results/${DESIGN_NAME}

write_sdc -output ../../results/${DESIGN_NAME}.out.wo_filler.sdc

write_def ../../results/${DESIGN_NAME}.out.wo_filler.def

write_parasitics -format SPEF -output ../../results/${DESIGN_NAME}.out.wo

write_gds -design ${DESIGN_NAME}_6_complete \
          -layer_map $Gds_map_file \
          -keep_data_type \
          -fill include \
          -output_pin all \
          -merge_files $Std_cell_gds \
          -long_names \
          ../../results/${DESIGN_NAME}.out.wo_filler.gds
```

- These commands generate various output files like Verilog, SDC, DEF, SPEF, and GDS for the design, which are important for post-synthesis validation and preparing the design for further steps like physical implementation.
- Reference: The user guides provide insights into the purpose of each file type and the commands to generate them .

10. Close Operations:

```
close_block
close_lib
exit
```

- Closes the block and the library, ensuring the environment is cleanly exited.

Outputs on Terminal

- Step 4

```
****************************************
  Report : Placement Attempts
  Site   : unit
****************************************

number of cells:                    702
number of references:                75
number of site rows:                 67
number of locations attempted:    15419
number of locations failed:           0  (0.0%)
```

```
PATHGROUP QOR
Scene  PG      WNS        TNS    NSV      WHV        THV    NHV
    1   1   0.0000     0.0000      0   0.0000     0.0000      0
    1   2   0.0000     0.0000      0   0.0000     0.0000      0
    1   3   0.0000     0.0000      0   0.0000     0.0000      0
    1   4   0.0000     0.0000      0   0.0000     0.0000      0
    1   5   0.0000     0.0000      0   0.0000     0.0000      0
    1   6   0.0000     0.0000      0   0.0000     0.0000      0
    1   7   0.0000     0.0000      0   0.0000     0.0000      0
    2   1   0.0000     0.0000      0   0.0000     0.0000      0
    2   2   0.0000     0.0000      0   0.0000     0.0000      0
    2   3   0.0000     0.0000      0   0.0000     0.0000      0
    2   4   0.0000     0.0000      0   0.0000     0.0000      0
    2   5   0.0000     0.0000      0   0.0000     0.0000      0
    2   6   0.0000     0.0000      0   0.0000     0.0000      0
    2   7   0.0000     0.0000      0   0.0000     0.0000      0

SCENARIO QOR
Scene  PG      WNS        TNS   R2RTNS    NSV      WHV        THV    NHV  MaxTrnV   MaxTranC  MaxCapV     Leakage
    1   *   0.0000     0.0000   0.0000      0   0.0000     0.0000      0        0     0.0000        0 103504.695
    2   *   0.0000     0.0000   0.0000      0   0.0000     0.0000      0        0     0.0000        0 103504.695

DESIGN QOR
Scene  PG      WNS        TNS   R2RTNS    NSV      WHV        THV    NHV  MaxTrnV   MaxTranC  MaxCapV     Leakage         Area     InstCnt     BufCnt      InvCnt
    *   *   0.0000     0.0000   0.0000      0   0.0000     0.0000      0        0     0.0000        0 103504.695       323.37         702          5         155

Place-opt final QoR Summary         WNS        TNS   R2RTNS    NSV      WHV        THV    NHV  MaxTrnV  MaxCapV     Leakage         Area     InstCnt
Place-opt final QoR Summary      0.0000     0.0000   0.0000      0   0.0000     0.0000      0        0        0 103504.695       323.37         702
```

```
WNS of each timing group:                          s1         s2
wb_clk_i                                        1.3766     1.3631

Setup WNS:                                      1.3766     1.3631       1.3631
Setup TNS:                                      0.0000     0.0000       0.0000
Number of setup violations:                          0          0            0
Hold WNS:                                       0.0242     0.0247       0.0242
Hold TNS:                                       0.0000     0.0000       0.0000
Number of hold violations:                           0          0            0
Number of max trans violations:                      0          0            0
Number of max cap violations:                        0          0            0
Number of min pulse width violations:                0          0            0

Area:                                                                  323.365
Cell count:                                                                702
Buf/inv cell count:                                                        160
Std cell utilization:                                                   0.1995
CPU(s):                                                                    57
Mem(Mb):                                                                  686
Host name:                                                               ws27

Histogram:                  s1   s2
Max violations:              0    0
    above ~ -0.7   ---       0    0
     -0.6 ~ -0.7   ---       0    0
     -0.5 ~ -0.6   ---       0    0
     -0.4 ~ -0.5   ---       0    0
     -0.3 ~ -0.4   ---       0    0
     -0.2 ~ -0.3   ---       0    0
     -0.1 ~ -0.2   ---       0    0
        0 ~ -0.1   ---       0    0
Min violations:              0    0
   -0.06 ~ above   ---       0    0
   -0.05 ~ -0.06   ---       0    0
   -0.04 ~ -0.05   ---       0    0
   -0.03 ~ -0.04   ---       0    0
   -0.02 ~ -0.03   ---       0    0
   -0.01 ~ -0.02   ---       0    0
       0 ~ -0.01   ---       0    0
```

- Step 5

```
************************************************************
* CTS STEP: Pre-Optimization DRC Fixing
************************************************************
Started Initial DRC Fixing at Fri Apr 26 16:16:02 2024
Scenario func:fast
Clock                 Latency        Skew     DRCViols    WorstTran     WorstCap        BufCt      BufArea     CellArea   ClkWireLen
--------------------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
wb_clk_i               0.0085     0.0065           0     0.0000     0.0000           0     0.0000     0.0000     438.6970

Scenario func:slow
Clock                 Latency        Skew     DRCViols    WorstTran     WorstCap        BufCt      BufArea     CellArea   ClkWireLen
--------------------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
wb_clk_i               0.0090     0.0069           0     0.0000     0.0000           0     0.0000     0.0000     438.6970

------------------------------------------------------------
Fixing clock: wb_clk_i mode: func root: wb_clk_i

Clock QoR Before Fixing Clock Tree DRC Violation ...
Clock: wb_clk_i, Mode: func, Root: wb_clk_i
Information: CTS QoR Pre Initial DRC Fixing: GlobalSkew = 0.0065; ID = 0.0085; NetsWithDRC = 0; Worst Tran/Cap cost = 0.0000/0.0000; ClockBufCount = 0; ClockBufArea = 0.0000;
 ClockCellArea = 0.0000; ClockWireLen = 438.6970; Clock = wb_clk_i; Mode = func; Corner = fast; ClockRoot = wb_clk_i. (CTS-037)
Information: CTS QoR Pre Initial DRC Fixing: GlobalSkew = 0.0069; ID = 0.0090; NetsWithDRC = 0; Worst Tran/Cap cost = 0.0000/0.0000; ClockBufCount = 0; ClockBufArea = 0.0000;
 ClockCellArea = 0.0000; ClockWireLen = 438.6970; Clock = wb_clk_i; Mode = func; Corner = slow; ClockRoot = wb_clk_i. (CTS-037)

Starting fixing clock tree DRC Violation ...
Resized 0 cell(s), relocated 0 cell(s), cloned 0 repeater(s) and inserted 0 buffer(s)/inverter(s)
Cloned 0 repeater gate(s)
Ran incremental ZGR 0 time(s) for 0 net(s) and restored ZGR 0 time(s) for 0 net(s)
Clock QoR After Fixing Pre-Opt DRC Violation:
Clock: wb_clk_i, Mode: func, Root: wb_clk_i
Information: CTS QoR Post Initial DRC Fixing: GlobalSkew = 0.0065; ID = 0.0085; NetsWithDRC = 0; Worst Tran/Cap cost = 0.0000/0.0000; ClockBufCount = 0; ClockBufArea = 0.0000;
 ClockCellArea = 0.0000; ClockWireLen = 438.6970; Clock = wb_clk_i; Mode = func; Corner = fast; ClockRoot = wb_clk_i. (CTS-037)
Information: CTS QoR Post Initial DRC Fixing: GlobalSkew = 0.0069; ID = 0.0090; NetsWithDRC = 0; Worst Tran/Cap cost = 0.0000/0.0000; ClockBufCount = 0; ClockBufArea = 0.0000;
 ClockCellArea = 0.0000; ClockWireLen = 438.6970; Clock = wb_clk_i; Mode = func; Corner = slow; ClockRoot = wb_clk_i. (CTS-037)
The elapsed time for fixing DRC violation is 0 hr : 0 min : 0.00 sec, cpu time is 0 hr : 0 min : 0.00 sec.
Finished Initial DRC Fixing at Fri Apr 26 16:16:02 2024 (elapsed: 0:00:00)
Scenario func:fast
Clock                 Latency        Skew     DRCViols    WorstTran     WorstCap        BufCt      BufArea     CellArea   ClkWireLen
--------------------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
wb_clk_i               0.0085     0.0065           0     0.0000     0.0000           0     0.0000     0.0000     438.6970
                      +0.0000    +0.0000          +0    +0.0000    +0.0000          +0    +0.0000    +0.0000      +0.0000
Scenario func:slow
Clock                 Latency        Skew     DRCViols    WorstTran     WorstCap        BufCt      BufArea     CellArea   ClkWireLen
--------------------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ---------- ----------
wb_clk_i               0.0090     0.0069           0     0.0000     0.0000           0     0.0000     0.0000     438.6970
                      +0.0000    +0.0000          +0    +0.0000    +0.0000          +0    +0.0000    +0.0000      +0.0000
Information: The run time for pre-optimization DRC fixing is 0 hr : 0 min : 0.01 sec, cpu time is 0 hr : 0 min : 0.01 sec. (CTS-104)
************************************************************
```

```
--------------------------------------------------------------------------------
WNS of each timing group:                          s1        s2
--------------------------------------------------------------------------------
wb_clk_i                                        1.3226    1.2999
--------------------------------------------------------------------------------
Setup WNS:                                      1.3226    1.2999      1.2999
Setup TNS:                                      0.0000    0.0000      0.0000
Number of setup violations:                          0         0           0
Hold WNS:                                       0.0265    0.0272      0.0265
Hold TNS:                                       0.0000    0.0000      0.0000
Number of hold violations:                           0         0           0
Number of max trans violations:                      0         0           0
Number of max cap violations:                        0         0           0
Number of min pulse width violations:                0         0           0
--------------------------------------------------------------------------------
Area:                                                                  331.668
Cell count:                                                               713
Buf/inv cell count:                                                       150
Std cell utilization:                                                  0.2046
CPU(s):                                                                   86
Mem(Mb):                                                                1186
Host name:                                                              ws27
--------------------------------------------------------------------------------
Histogram:              s1    s2
--------------------------------------------------------------------------------
Max violations:          0     0
   above ~ -0.7   ---    0     0
    -0.6 ~ -0.7   ---    0     0
    -0.5 ~ -0.6   ---    0     0
    -0.4 ~ -0.5   ---    0     0
    -0.3 ~ -0.4   ---    0     0
    -0.2 ~ -0.3   ---    0     0
    -0.1 ~ -0.2   ---    0     0
       0 ~ -0.1   ---    0     0
--------------------------------------------------------------------------------
Min violations:          0     0
  -0.06 ~ above   ---    0     0
  -0.05 ~ -0.06   ---    0     0
  -0.04 ~ -0.05   ---    0     0
  -0.03 ~ -0.04   ---    0     0
  -0.02 ~ -0.03   ---    0     0
  -0.01 ~ -0.02   ---    0     0
     0 ~ -0.01    ---    0     0
--------------------------------------------------------------------------------
```

- Step 6

```
Total Wire Length =                    7139 micron
Total Number of Contacts =             4948
Total Number of Wires =                6015
Total Number of PtConns =              640
Total Number of Routed Wires =       6015
Total Routed Wire Length =           6653 micron
Total Number of Routed Contacts =       4948
  Layer               M1 :         14 micron
  Layer               M2 :       2908 micron
  Layer               M3 :       3809 micron
  Layer               M4 :        366 micron
  Layer               M5 :         42 micron
  Layer               M6 :          0 micron
  Layer               M7 :          0 micron
  Layer               M8 :          0 micron
  Layer               M9 :          0 micron
  Layer             MRDL :          0 micron
  Via           VIA45SQ :         10
  Via     VIA34SQ_C(rot) :        223
  Via       VIA34SQ(rot) :          3
  Via          VIA23SQ_C :       2793
  Via            VIA23SQ :          1
  Via        VIA2_33SQ_C :          1
  Via     VIA12SQ_C(rot) :       1848
  Via       VIA12SQ(rot) :         59
  Via   VIA1_32SQ_C(rot) :         10
```

```
--------------------------------------------------------------------------------
WNS of each timing group:                          s1        s2
--------------------------------------------------------------------------------
wb_clk_i                                        1.3403    1.3200
--------------------------------------------------------------------------------
Setup WNS:                                      1.3403    1.3200      1.3200
Setup TNS:                                      0.0000    0.0000      0.0000
Number of setup violations:                          0         0           0
Hold WNS:                                       0.0277    0.0294      0.0277
Hold TNS:                                       0.0000    0.0000      0.0000
Number of hold violations:                           0         0           0
Number of max trans violations:                      0         0           0
Number of max cap violations:                        0         0           0
Number of min pulse width violations:                0         0           0
--------------------------------------------------------------------------------
Area:                                                                 331.624
Cell count:                                                               713
Buf/inv cell count:                                                       150
Std cell utilization:                                                  0.2045
CPU(s):                                                                   33
Mem(Mb):                                                                 653
Host name:                                                              ws27
--------------------------------------------------------------------------------
Histogram:                 s1   s2
--------------------------------------------------------------------------------
Max violations:             0    0
   above ~ -0.7  ---        0    0
    -0.6 ~ -0.7  ---        0    0
    -0.5 ~ -0.6  ---        0    0
    -0.4 ~ -0.5  ---        0    0
    -0.3 ~ -0.4  ---        0    0
    -0.2 ~ -0.3  ---        0    0
    -0.1 ~ -0.2  ---        0    0
       0 ~ -0.1  ---        0    0
--------------------------------------------------------------------------------
Min violations:             0    0
  -0.06 ~ above  ---        0    0
  -0.05 ~ -0.06  ---        0    0
  -0.04 ~ -0.05  ---        0    0
  -0.03 ~ -0.04  ---        0    0
  -0.02 ~ -0.03  ---        0    0
  -0.01 ~ -0.02  ---        0    0
     0 ~ -0.01   ---        0    0
--------------------------------------------------------------------------------
```

```
----------------------------------------
Number of Written DEF Constructs
----------------------------------------
VERSION                         : 1
DIVIDERCHAR                     : 1
BUSBITCHARS                     : 1
DESIGN                          : 1
UNITS                           : 1
PROPERTYDEFINITIONS             : 1
DIEAREA                         : 1
ROW                             : 67
TRACKS                          : 20
VIAS                            : 4
NONDEFAULTRULES                 : 1
COMPONENTS                      : 713
PINS                            : 35
PINPROPERTIES                   : 35
SPECIALNETS                     : 7
NETS                            : 734
```

Reports

- Clock QoR Report

```
****************************************
No. of scenario = 2
s1 = func_fast
s2 = func_slow
--------------------------------------------------------------------------------
WNS of each timing group:                        s1        s2
--------------------------------------------------------------------------------
wb_clk_i                                     1.3226    1.2999
--------------------------------------------------------------------------------
Setup WNS:                                   1.3226    1.2999      1.2999
Setup TNS:                                   0.0000    0.0000      0.0000
Number of setup violations:                       0         0           0
Hold WNS:                                    0.0265    0.0272      0.0265
Hold TNS:                                    0.0000    0.0000      0.0000
Number of hold violations:                        0         0           0
Number of max trans violations:                   0         0           0
Number of max cap violations:                     0         0           0
Number of min pulse width violations:             0         0           0
--------------------------------------------------------------------------------
Area:                                                               331.668
Cell count:                                                             713
Buf/inv cell count:                                                     150
Std cell utilization:                                                0.2046
CPU(s):                                                                 86
Mem(Mb):                                                              1186
Host name:                                                            ws27
--------------------------------------------------------------------------------
Histogram:              s1   s2
--------------------------------------------------------------------------------
Max violations:          0    0
   above ~ -0.7  ---     0    0
    -0.6 ~ -0.7  ---     0    0
    -0.5 ~ -0.6  ---     0    0
    -0.4 ~ -0.5  ---     0    0
    -0.3 ~ -0.4  ---     0    0
    -0.2 ~ -0.3  ---     0    0
    -0.1 ~ -0.2  ---     0    0
       0 ~ -0.1  ---     0    0
--------------------------------------------------------------------------------
Min violations:          0    0
  -0.06 ~ above  ---     0    0
  -0.05 ~ -0.06  ---     0    0
  -0.04 ~ -0.05  ---     0    0
  -0.03 ~ -0.04  ---     0    0
  -0.02 ~ -0.03  ---     0    0
  -0.01 ~ -0.02  ---     0    0
     0 ~ -0.01   ---     0    0
--------------------------------------------------------------------------------
```

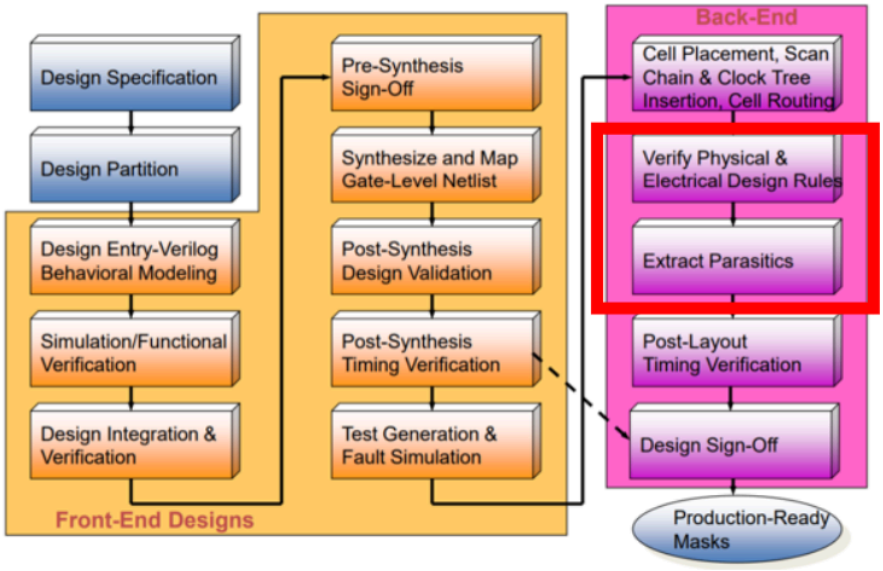
- Clock Pre-Route QoR Report

```
No. of scenario = 2
s1 = func_fast
s2 = func_slow
------------------------------------------------------------------------------
WNS of each timing group:                    s1        s2
------------------------------------------------------------------------------
wb_clk_i                                  1.3953    1.3814
------------------------------------------------------------------------------
Setup WNS:                                1.3953    1.3814      1.3814
Setup TNS:                                0.0000    0.0000      0.0000
Number of setup violations:                    0         0           0
Hold WNS:                                 0.0244    0.0249      0.0244
Hold TNS:                                 0.0000    0.0000      0.0000
Number of hold violations:                     0         0           0
Number of max trans violations:                0         0           0
Number of max cap violations:                  0         0           0
Number of min pulse width violations:          0         0           0
------------------------------------------------------------------------------
Area:                                                            323.365
Cell count:                                                          702
Buf/inv cell count:                                                  160
Std cell utilization:                                             0.1995
CPU(s):                                                              38
Mem(Mb):                                                            731
Host name:                                                         ws27
------------------------------------------------------------------------------
Histogram:             s1   s2
------------------------------------------------------------------------------
Max violations:         0    0
   above ~ -0.7  ---    0    0
    -0.6 ~ -0.7  ---    0    0
    -0.5 ~ -0.6  ---    0    0
    -0.4 ~ -0.5  ---    0    0
    -0.3 ~ -0.4  ---    0    0
    -0.2 ~ -0.3  ---    0    0
    -0.1 ~ -0.2  ---    0    0
       0 ~ -0.1  ---    0    0
------------------------------------------------------------------------------
Min violations:         0    0
  -0.06 ~ above  ---    0    0
  -0.05 ~ -0.06  ---    0    0
  -0.04 ~ -0.05  ---    0    0
  -0.03 ~ -0.04  ---    0    0
  -0.02 ~ -0.03  ---    0    0
  -0.01 ~ -0.02  ---    0    0
     0 ~ -0.01   ---    0    0
------------------------------------------------------------------------------
```

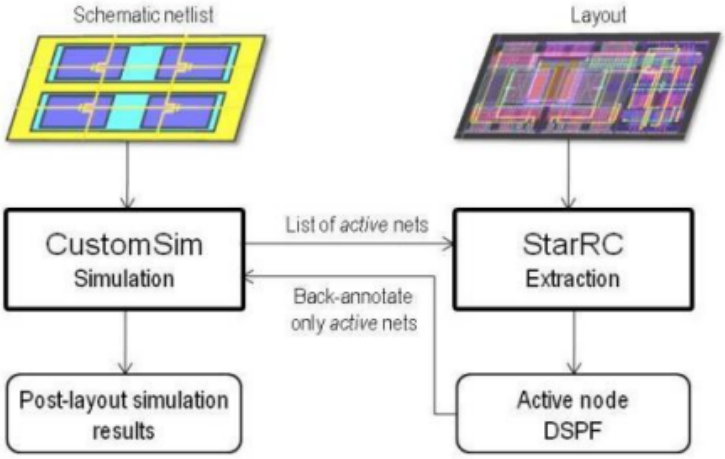
- Clock Timing Report

```
 Mode: func
 Clock: wb_clk_i

 Clock Pin                                          Latency      Skew            Corner
-------------------------------------------------------------------------------------------
 byte_controller/bit_controller/busy_reg/CK          0.01                rp-+       slow
 wb_dat_o_reg_6_/CK                                  0.00       0.01     rp-+       slow

-------------------------------------------------------------------------------------------
```

4. StarRC



- **Goal:** Extract the information of parasitic devices for simulation and analysis



Description of command in scripts (tcl file)

`gen_StarRC_smc.tcl`

1. Common script source

```
source ../../common/common.tcl
```

2. Open and write to the StarRC.smc file

```
set fp [open "../script/StarRC.smc" w+]
```

- Here, the script is creating or opening the `StarRC.smc` file for writing. This file is intended to define settings for different corners in StarRC simulations.

3. Constructing directory paths

```
set cur_path_split [split [pwd] "/"]
set abs_dir ""
for {set index 0} {$index < [expr [llength $cur_path_split] - 1]} {incr i
  set abs_dir "${abs_dir}/[lindex $cur_path_split $index]"
}
```

- This loop builds the path up to the last directory. It's commonly used in automation scripts to adapt to different execution environments dynamically.

4. Defining corner-specific configurations

```
foreach corner ${STARRC_SELECTED_CORNERS} {
  if { $corner == "slow" } {
    puts $fp "CORNER_NAME: slow"
    puts $fp "OPERATING_TEMPERATURE: 25"
    puts $fp "TCAD_GRD_FILE: $abs_dir_slow"
    puts $fp ""
  }
  if { $corner == "fast" } {
    puts $fp "CORNER_NAME: fast"
    puts $fp "OPERATING_TEMPERATURE: 25"
    puts $fp "TCAD_GRD_FILE: $abs_dir_fast"
    puts $fp ""
  }
}
```

- The script loops over selected corners. For each corner, it writes specific settings such as corner name, operating temperature, and the grid file (TCAD_GRD_FILE) path into the StarRC.smc file. For example, if corner is `slow`, it configures a specific simulaiton corner with a name(`slow`), set the temperature to 25, and configure a corresponding grid file. The StarRC.smc file format, used here, is pivotal for defining how StarRC handles multi-corner simulations, allowing it to simulate circuit behavior under different manufacturing variances or operating conditions.

gen_star_cmd_gate_DEFLEF.tcl

1. File and directory setup

```
set fp [open "../script/star_cmd_gate" w+]
```

- This opens (and creates if not existing) a file named star_cmd_gate for writing. The script then writes a series of configurations into this file.

2. Basic configuration

```
puts $fp "BLOCK: ${DESIGN_NAME}"
puts $fp "TOP_DEF_FILE: ../../results/${DESIGN_NAME}.out.wo_filler.def"
puts $fp "MAPPING_FILE:  ${Map_file}"
```

- `BLOCK` : Specifies the block name for which parasitic extraction is to be performed.
- `TOP_DEF_FILE` : Defines the path to the DEF (Design Exchange Format) file of the design without filler cells, used as input for the extraction.
- `MAPPING_FILE` : Indicates the file that maps design layers to process technology layers, critical for accurate parasitic extraction.

3. LEF files

```
foreach lef [list ${Std_cell_lef} ${Ram_lef}] {
  if {$lef != "" && $lef != " "} {
    puts $fp "LEF_FILE: $lef"
  }
}
```

- This loop adds all specified LEF (Library Exchange Format) files to the configuration, ensuring that all standard cell and RAM library data are included for extraction processes.

4. Extraction and simulation setting

```
puts $fp "REDUCTION: NO_EXTRA_LOOPS"
puts $fp "NETS: *"
puts $fp "EXTRACTION: RC"
puts $fp "DPT: YES"
puts $fp "NUM_CORES: 4"
puts $fp "STARRC_DP_STRING:"
puts $fp "CORNERS_FILE: ../script/StarRC.smc"
puts $fp "DENSITY_BASED_THICKNESS: YES"
puts $fp "SELECTED_CORNERS: $STARRC_SELECTED_CORNERS"
puts $fp "SIMULTANEOUS_MULTI_CORNER: YES"
```

- `REDUCTION` : Configures the reduction settings for static timing analysis, optimizing the netlist for performance.
- `EXTRACTION` : Sets the type of extraction, RC indicating resistance and capacitance.
- `DPT` : Deep pattern technology, likely a setting related to advanced node processing.
- `NUM_CORES` and `STARRC_DP_STRING` : Specifies the number of processing cores and a custom distributed processing string for parallel processing.
- `CORNERS_FILE` , `SELECTED_CORNERS` , `SIMULTANEOUS_MULTI_CORNER` : Configure multi-corner analysis, crucial for simulating different process variations and operating conditions.

5. Output configuaration

```
puts $fp "COUPLE_TO_GROUND: NO"
puts $fp "COUPLING_ABS_THRESHOLD: 3e-15"
puts $fp "COUPLING_REL_THRESHOLD: 0.03"
puts $fp "REDUCTION_MAX_DELAY_ERROR: 1e-14"
puts $fp "NETLIST_FORMAT: SPEF"
puts $fp "NETLIST_FILE: ./../../results/${DESIGN_NAME}.star.wo_filler.spe
puts $fp "SUMMARY_FILE: ./../../results/${DESIGN_NAME}.star_sum"
puts $fp "STAR_DIRECTORY: ./star_directory"
```

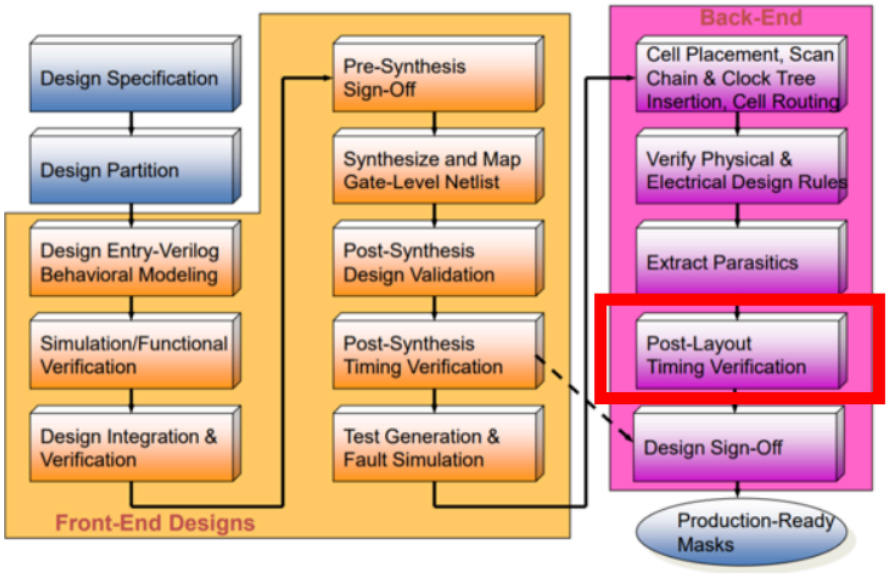
- These settings configure how the parasitic effects are reported and the thresholds for including them in the output.
- `NETLIST_FORMAT` : SPEF indicates the use of Standard Parasitic Exchange Format for output.
- `SUMMARY_FILE` : and 'STAR_DIRECTORY' define where the summary and StarRC's internal data will be stored.

Outputs on Terminal

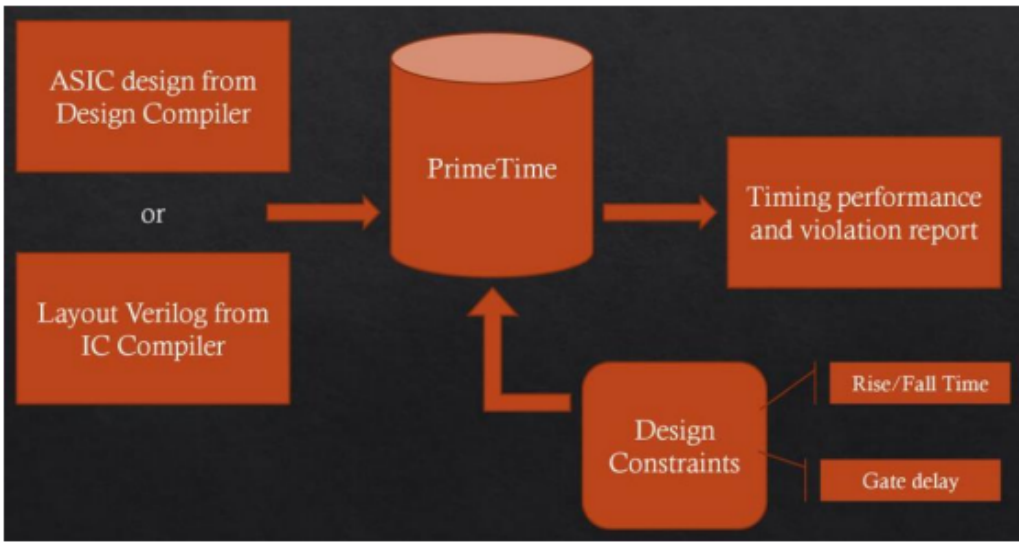
- Finish

```
Setup                 Elp=00:00:23 Cpu=00:00:01 Usr=1.7    Sys=0.0    Mem=513.0
Layers                Elp=00:00:00 Cpu=00:00:00 Usr=0.1    Sys=0.0    Mem=513.1
HN                    Elp=00:00:00 Cpu=00:00:00 Usr=0.1    Sys=0.0    Mem=518.9
Cells                 Elp=00:00:01 Cpu=00:00:00 Usr=0.1    Sys=0.1    Mem=521.9
Translate             Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=513.4
NetlistSetup          Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=483.6
GPD_XtractSetup       Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=513.2
GPD_NameMap           Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=483.8
xTract                Elp=00:00:01 Cpu=00:00:01 Usr=1.4    Sys=0.1    Mem=522.8
xTractPP              Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=338.7
ReportViolations      Elp=00:00:01 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=483.5
ReportOpens           Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=483.4
GPD_PostProcess       Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=483.8
GPD_Converter1        Elp=00:00:00 Cpu=00:00:00 Usr=0.1    Sys=0.0    Mem=333.9
GPD_Converter2        Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=332.5
GPD_Converter_merge_c1 Elp=00:00:00 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=332.4
GPD_Converter_merge_c2 Elp=00:00:01 Cpu=00:00:00 Usr=0.0    Sys=0.0    Mem=332.4

Done          Elp=00:00:27 Cpu=00:00:03 Usr=3.5    Sys=0.2    Mem=522.8
date >  run_StarRC_cmd
date >  all
[u110061217@ws27 work]$
```

5. PrimeTime



- **Goal:** Static Timing Analysis and Power analysis



Description of command in scripts (tcl file)

1. Script header

```
source ../../common/common.tcl
```

- This line sources or includes a common TCL file that likely contains predefined variables, procedures, or configurations used across multiple scripts within the project.

2. Loop over scenarios

```
foreach scenario $PT_SELECTED_SCENARIO {
```

- This `foreach` loop iterates over a list of scenarios stored in the variable 'PT_SELECTED_SCENARIO'. Each iteration processes one scenario at a time, setting up and configuring analysis accordingly.

3. File handling

```
set fp [open "./pt_workspace/${scenario}/run_pt_cmd.${scenario}.tcl" w+]
```

- Open a file or writing(`w+` mode, which means open for reading and writing; if the file doesn't exist, it will be created) in the specitied directory. The file name is dynamically generated based on the current scenario.

4. Writing comments and metadata

```
puts $fp "##########################################################################"
puts $fp "# PrimeTime Reference Methodology Script"
puts $fp "# Script: pt.tcl"
puts $fp "# Version: U-2022.12-SP4 (June 30, 2023)"
puts $fp "# Copyright (C) 2007-2023 Synopsys, Inc. All rights reserved."
puts $fp "##########################################################################"
puts $fp "# Please do not modify the sdir variable."
puts $fp "# Doing so may cause script to fail."
puts $fp "set sdir \".\" "
puts $fp "\n"
```

- Writes a series of comment lines and meta information into the script file, including a warning about not modifying `sdir` variable.

5. Source common files

```
puts $fp "##########################################################################"
puts $fp "#    Source common and pt_setup.tcl File"
puts $fp "##########################################################################"
puts $fp "source ../../common/common.tcl"
```

- Writes commands into the file to source common files, which are likely needed for every run, ensuring that all necessary configuation and common procedures are loaded.

6. Directory creation

```
puts $fp "# make PT_REPORTS_DIR"
puts $fp "file mkdir ./pt_workspace/${scenario}/\$PT_REPORTS_DIR"
puts $fp "# make RESULTS_DIR"
puts $fp "file mkdir ./pt_workspace/${scenario}/\$PT_RESULTS_DIR"
puts $fp "\n"
```

- These lines write commands to create directories for storing reports and result, ensuring that the output from each scenario es organized and separated.

7. Configuration and environment setup

```
puts $fp "#########################################################################"
puts $fp "#    Search Path, Library and Operating Condition Section"
puts $fp "#########################################################################"
puts $fp "set report_default_significant_digits 3 "
puts $fp "set sh_source_uses_search_path true "
puts $fp "set search_path \". \$search_path \""
puts $fp "\n"
puts $fp "#########################################################################"
puts $fp "#    Netlist Reading Section"
puts $fp "#########################################################################"
puts $fp "set link_path \"* \$PT_link_path\""
puts $fp "read_verilog \$PT_NETLIST_FILES"
puts $fp "current_design \$DESIGN_NAME"
puts $fp "link"
puts $fp "\n"
```

- These lines set up the environment by defining paths and libraries, and handle the reading and linking of the netlist files specific to the current design. The setting ensure that all necessary files are accessible and that the design is correctly set up fot analysis.

8. Netlist reading seciton

```
puts $fp "#########################################################################"
puts $fp "#    Netlist Reading Section"
puts $fp "#########################################################################"
puts $fp "set link_path \"* \$PT_link_path\""
puts $fp "read_verilog \$PT_NETLIST_FILES"
puts $fp "current_design \$DESIGN_NAME"
puts $fp "link"
puts $fp "\n"
```

- Setting the Link Path: set link_path "* $PT_link_path" - This command sets the variable link_path to a value that likely includes a wildcard (*) appended with the

content of another variable, PT_link_path. This variable (link_path) is used to define search paths for library files or other design files necessary for the netlist link process. The wildcard indicates a non-specific or global setting applicable across multiple directories or scopes.

- Reading the Verilog Netlist: read_verilog $PT_NETLIST_FILES - This command instructs PrimeTime to read the Verilog netlist files specified by the variable PT_NETLIST_FILES. Netlist files contain descriptions of the electronic components and their interconnections in the design, and reading these files is essential for the tool to perform timing analysis.
- Setting the Current Design: current_design $DESIGN_NAME - Sets the focus of subsequent operations to the design specified by the DESIGN_NAME variable. This makes sure that all operations and analyses are targeted at the correct design entity.

9. Back annotation section

```
puts $fp "#########################################################################
puts $fp "#    Back Annotation Section
puts $fp "#########################################################################
puts $fp "if { [info exists PT_PARASITIC_PATHS] && [info exists PT_PARASI
puts $fp "  foreach para_path \$PT_PARASITIC_PATHS para_file \$PT_PARASIT
puts $fp "    if {[string compare \$para_path \$DESIGN_NAME] == 0} {"
puts $fp "      read_parasitics \$para_file"
puts $fp "    } else {"
puts $fp "      read_parasitics -path \$para_path \$para_file"
puts $fp "    }"
puts $fp "  }"
puts $fp "}"
puts $fp "\n"

```

- Back Annotation: This section is used to annotate the design with parasitic data, which typically includes resistance and capacitance values extracted from the physical layout. This data is crucial for accurate post-layout timing analysis.
- Conditional Loading: The script checks if the parasitic paths and files exist, then iterates over each file. If the path matches the current design name, it reads the parasitic file directly; otherwise, it specifies the path explicitly.
- Purpose: Parasitics significantly affect signal integrity and timing, making this step essential for achieving reliable simulation results.

10. Reading contraint section

```
puts $fp "#########################################################################
puts $fp "#    Reading Constraints Section
puts $fp "#########################################################################
puts $fp "if {[info exists PT_CONSTRAINT_FILES]} {"
puts $fp "        foreach constraint_file \$PT_CONSTRAINT_FILES {"
puts $fp "                if {[file extension \$constraint_file] eq \".sd
puts $fp "                        read_sdc -echo \$constraint_file"
puts $fp "                } else {"
puts $fp "                        source -echo \$constraint_file"
puts $fp "                }"
puts $fp "        }"
puts $fp "}"
puts $fp "\n"
```

- Reading Constraints: This script section loads timing constraints from files. Timing constraints might include setup and hold times, maximum delays, etc., which are crucial for timing verification.
- Support for Multiple Formats: It supports different types of files, distinguishing between Synopsys Design Constraints (`.sdc`) files and potentially other script or constraint files.
- Conditional Execution: Only loads these constraints if they are specified, ensuring flexibility in simulation configurations.

11. Update timing and check timing section

```
puts $fp "#########################################################################
puts $fp "#    Update_timing and check_timing Section
puts $fp "#########################################################################
puts $fp "update_timing -full"
puts $fp "check_timing -verbose > ./pt_workspace/${scenario}/\$PT_REPORTS
puts $fp "\n"
```

- Update Timing: This command recalculates timing paths, taking into account any changes in the design or setup, ensuring that the timing data is up-to-date.
- Check Timing: Performs a timing analysis, checking for any violations against the loaded constraints, and outputs a detailed report. The -verbose option provides extended details, which are helpful for debugging.

12. Report timing seciton

```
puts $fp "########################################################################
puts $fp "#    Report_timing Section
puts $fp "########################################################################
puts $fp "report_global_timing > ./pt_workspace/${scenario}/\$PT_REPORTS_
puts $fp "report_clock -skew -attribute > ./pt_workspace/${scenario}/\$PT
puts $fp "report_analysis_coverage > ./pt_workspace/${scenario}/\$PT_REPO
puts $fp "report_timing -slack_lesser_than 10 -delay min_max -nosplit -in
puts $fp "report_timing -slack_lesser_than 0.0 -delay min_max -nosplit -i
puts $fp "\n"
```

- Reporting: This section generates various timing reports that include global timing, clock skew, analysis coverage, and specific timing paths with potential issues.
- Customizable Reports: The script uses various options to filter and customize the output, such as specifying slack thresholds and delay types.

13. Fix eco power -pba_mode none -verbose

```
puts $fp "########################################################################
puts $fp "#    Fix ECO Power Cell Downsize Section
puts $fp "########################################################################
puts $fp "fix_eco_power -pba_mode none -verbose"
puts $fp "\n"
```

- Fix ECO Power: This command adjusts the design for power optimization as part of an Engineering Change Order (ECO). ECOs are modifications made to the design post initial implementation to correct errors or optimize performance.
- Parameters:
  - `-pba_mode none` : This option specifies that power-based analysis (PBA) should not be used in the ECO process. PBA can sometimes be used to make more power-efficient changes by analyzing the power implications of potential modifications.
  - `-verbose` : This enables verbose output, providing detailed logs of the changes made, which can be useful for debugging and verification purposes.

14. Write SDF for Post-sim

```
puts $fp "#    Write SDF for Post-sim
puts $fp "#############################################################
puts $fp "#   write_sdf ./${DESIGN_NAME}.${scenario}.sdf"
puts $fp "write_sdf ./../../results/${DESIGN_NAME}.${scenario}.sdf"
puts $fp "\n"
```

- Write SDF: This command writes out a Standard Delay Format (SDF) file for the current design scenario. SDF files contain timing information for all cells in the design and are used for back-annotation in gate-level simulation to ensure timing simulations match timing analyses.
- File Path Customization: The script comments out a line for generating an SDF in one location and uses another line to specify a different path, showing flexibility in file management based on different use cases or directory structures.

15. Save PT Session

```
puts $fp "#############################################################
puts $fp "#    Save PT Session
puts $fp "#############################################################
puts $fp "save_session ./pt_workspace/${scenario}/\$PT_SESSION_DIR"
puts $fp "\n"
```

- Save PT Session: This command saves the current PrimeTime session, including all setup information, loaded designs, and state, to a specified directory. This is crucial for preserving the session state for later review or for continuing work in a subsequent session.
- Directory Customization: It specifies a path that includes the scenario name, ensuring that each scenario's session data is kept separate and organized.

16. Copy the parasitics_command.log and pt_shell_command.log

```
puts $fp "#############################################################
puts $fp "#    Copy the parasitics_command.log and pt_shell_command.log
puts $fp "#############################################################
puts $fp "exec /bin/cp -r parasitics_command.log ./pt_workspace/${scenari
puts $fp "exit"
```

- Copy Logs: This part of the script uses the system `cp` command to copy log files from a default location to a scenario-specific directory. This is useful for archiving and later analyzing the logs specific to each scenario.

- Exit: The script issues an `exit` command to terminate the session after copying the logs, ensuring a clean termination of the script processing.

Outputs on Terminal

- Netlist reading

```
#############################################################
#    Netlist Reading Section                                #
#############################################################
set link_path "* $PT_link_path"
* * /home/course/ee5252/lab_snps_flow/SAED14_EDK_LAB/SAED14_EDK_LAB/lib/stdcell_rvt/db_nldm/saed14rvt_tt0p8v25c.db
read_verilog $PT_NETLIST_FILES
1
current_design $DESIGN_NAME
Information: current_design won't return any data before link (DES-071)
link
Loading verilog file '/home/u110/u110061217/2024ASoC/lab_formal_release/lab_formal_release/results/i2c_master_top.out.wo_filler.v'
Loading db file '/home/course/ee5252/lab_snps_flow/SAED14_EDK_LAB/SAED14_EDK_LAB/lib/stdcell_rvt/db_nldm/saed14rvt_tt0p8v25c.db'
Linking design i2c_master_top...
Information: Removing 4 unneeded designs..... (LNK-034)
Information: 868 (94.14%) library cells are unused in library saed14rvt_tt0p8v25c..... (LNK-045)
Information: total 868 library cells are unused (LNK-046)
Design 'i2c_master_top' was successfully linked.
Information: There are 753 leaf cells, ports, hiers and 920 nets in the design (LNK-047)
1
```

- Reading contraint

```
#####################################################################
#    Reading Constraints Section                                    #
#####################################################################
if  {[info exists PT_CONSTRAINT_FILES]} {
        foreach constraint_file $PT_CONSTRAINT_FILES {
                if {[file extension $constraint_file] eq ".sdc"} {
                        read_sdc -echo $constraint_file
                } else {
                        source -echo $constraint_file
                }
        }
}

Reading SDC version 2.1...
```

- Back notation section

```
#####################################################################
#    Back Annotation Section                                        #
#####################################################################
if { [info exists PT_PARASITIC_PATHS] && [info exists PT_PARASITIC_FILES] } {
  foreach para_path $PT_PARASITIC_PATHS para_file $PT_PARASITIC_FILES {
    if {[string compare $para_path $DESIGN_NAME] == 0} {
      read_parasitics $para_file
    } else {
      read_parasitics -path $para_path $para_file
    }
  }
}
Information: Log for 'read_parasitics command' will be generated in 'parasitics_command.log'. (PARA-107)
```

- Update timing and check timing

```
#############################################################
#    Update_timing and check_timing Section                 #
#############################################################
update_timing -full
Information: Using automatic min wire load selection group 'predcaps'. (ENV-003)
Information: Using automatic max wire load selection group 'predcaps'. (ENV-003)
Information: Using automatic min wire load selection group 'predcaps'. (ENV-003)
Information: Building multi voltage information for entire design. (MV-022)
Warning: Some timing arcs have been disabled for breaking timing loops or because of constant propagation. Use the 'report_disable_timing' command to get the
list of these disabled timing arcs. (PTE-003)
1
check_timing -verbose > ./pt_workspace/fast/$PT_REPORTS_DIR/${DESIGN_NAME}_check_timing.report
```

- Report timing

```
#############################################################
#    Report_timing Section                                  #
#############################################################
report_global_timing > ./pt_workspace/fast/$PT_REPORTS_DIR/${DESIGN_NAME}_report_global_timing.report
report_clock -skew -attribute > ./pt_workspace/fast/$PT_REPORTS_DIR/${DESIGN_NAME}_report_clock.report
report_analysis_coverage > ./pt_workspace/fast/$PT_REPORTS_DIR/${DESIGN_NAME}_report_analysis_coverage.report
report_timing -slack_lesser_than 10 -delay min_max -nosplit -input -net  > ./pt_workspace/fast/$PT_REPORTS_DIR/${DESIGN_NAME}_report_timing.report
report_timing -slack_lesser_than 0.0 -delay min_max -nosplit -input -net  > ./pt_workspace/fast/$PT_REPORTS_DIR/${DESIGN_NAME}_report_timing_violation.report
```

- Fix eco power -pba_mode none -verbose

```
#############################################################
#    Fix ECO Variable Setup                                 #
#############################################################
set timing_save_pin_arrival_and_slack true
true
```

Reports

- Singal wiring statistics

```
Signal Wiring Statistics

Metal layer     Num wires  % of total#  Wire length % of total length
PO                      0        0.00%         0.00             0.00%
M1                    120        2.00%        13.62             0.21%
M2                   3475       57.89%      2889.29            43.97%
M3                   2371       39.50%      3483.35            53.01%
M4                     36        0.60%       159.02             2.42%
M5                      1        0.02%        25.83             0.39%
M6                      0        0.00%         0.00             0.00%
M7                      0        0.00%         0.00             0.00%
M8                      0        0.00%         0.00             0.00%
M9                      0        0.00%         0.00             0.00%
MRDL                    0        0.00%         0.00             0.00%
```

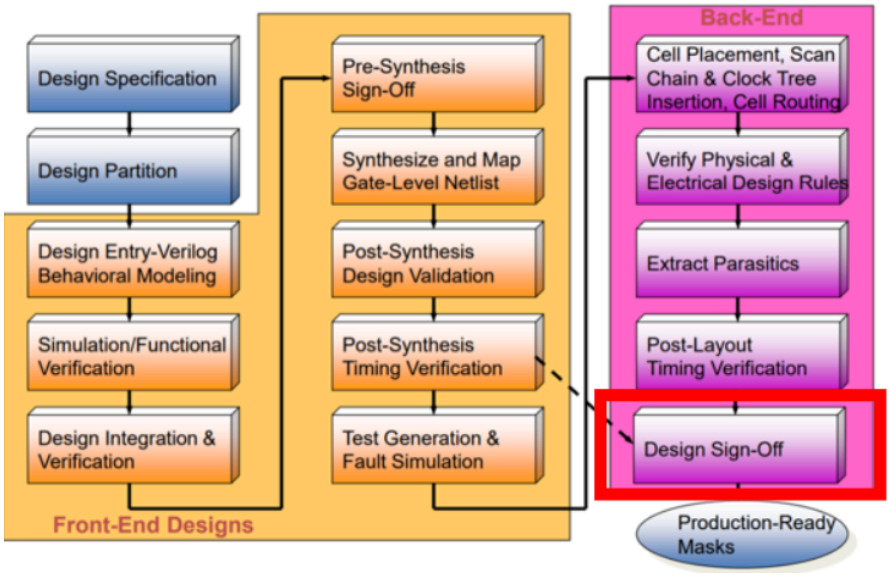
- Clock wiring statistics

```
Clock Wiring Statistics

Metal layer     Num wires  % of total#  Wire length % of total length
PO                      0        0.00%         0.00             0.00%
M1                      0        0.00%         0.00             0.00%
M2                     49       10.79%         7.40             1.45%
M3                    272       59.91%       278.66            54.53%
M4                    126       27.75%       209.16            40.93%
M5                      7        1.54%        15.81             3.09%
M6                      0        0.00%         0.00             0.00%
M7                      0        0.00%         0.00             0.00%
M8                      0        0.00%         0.00             0.00%
M9                      0        0.00%         0.00             0.00%
MRDL                    0        0.00%         0.00             0.00%
```

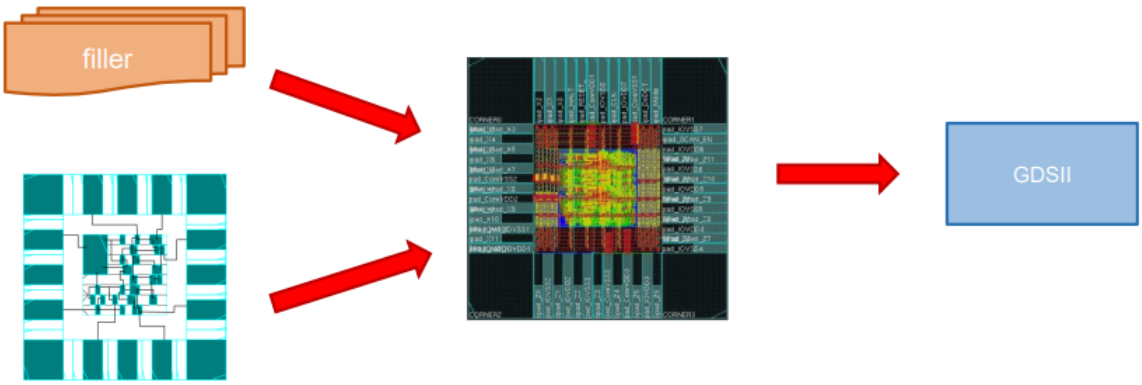
- P/G wiring statistics

```
P/G Wiring Statistics

Metal layer     Num wires  % of total#  Wire length % of total length
PO                      0        0.00%         0.00             0.00%
M1                     68       58.62%      2742.44            49.65%
M2                      0        0.00%         0.00             0.00%
M3                      0        0.00%         0.00             0.00%
M4                      0        0.00%         0.00             0.00%
M5                      0        0.00%         0.00             0.00%
M6                     24       20.69%      1388.80            25.15%
M7                     24       20.69%      1391.92            25.20%
M8                      0        0.00%         0.00             0.00%
M9                      0        0.00%         0.00             0.00%
MRDL                    0        0.00%         0.00             0.00%
```

6. Chip Finishing



- **Goal:** Insert filler and export GDSII file for chip tap out



Description of command in scripts (tcl file)

1. Set Library

```
set link_library "$Std_cell_lib $Ram_lib"
set target_library "$Std_cell_lib $Ram_lib"
```

- These lines set the libraries to be used for linking and targeting during the design process. `$Std_cell_lib` and `$Ram_lib` are placeholders for standard cell libraries and RAM libraries, respectively. THese are crucial for the tool to understand what components it can use to synthesis the design.

2. Open Library and Copy Block

```
open_lib $ARC_TOP
copy_block -from_block ${DESIGN_NAME}_6_complete -to temp_route_ends
open_block temp_route_ends
```

- Open Library: Opens a top-level library, which could be a reference to the highest hierarchy in the design, specified by the variable `$ARC_TOP`.
- Copy Block: This operation copies a design block (probably a version of the design that has completed a certain stage of routing) to a new temporary block for further processing.
- Open Block: Opens the newly created block for subsequent operations. This step is typically used to modify or analyze the newly copied version without affecting the original.

3. Insert Redundant Vias

```
add_redundant_vias
```

- Adds redundant vias to the design, which is a common practice to ensure manufacturing reliability. Redundent vias reduce the risk of via failure due to manufacturing defects.

4. Insert Filler

```
set pnr_std_fillers "SAEDRVT14_FILL*"
set std_fillers ""
foreach filler $pnr_std_fillers { lappend std_fillers "*/${filler}" }
create_stdcell_fillers -lib_cells $std_fillers
```

- Setup Fillers: Defines a list of standard cell fillers (used to occupy empty spaces in the design to improve manufacturability and performance).
- Append Paths: The `foreach loop` formats the filler names to include wildcards for paths, preparing them for a function or procedure that likely expects a certain path format.
- Create Fillers: This command uses the prepared list of filler cells to actually populate the design with these cells.

5. Use Correct ICV Version and Create Metal Fill

```
set_app_options -name signoff.create_metal_fill.runset -value "${icv_mfil
set create_metal_fill_cmd "signoff_create_metal_fill"
if {$METAL_FILL_SELECT_LAYER != ""} {
  lappend create_metal_fill_cmd -select_layers $METAL_FILL_SELECT_LAYER
}
eval $create_metal_fill_cmd
```

- Set Application Options: Configures application-specific options, probably for a tool used in the signoff process. This sets the runset configuration for metal fill

creation to a specific ICV (Integrated Circuit Validator) version.

- Dynamic Command Construction: Prepares a command for creating metal fills, with an optional parameter for selecting specific layers if specified.
- Execute Command: The `eval` command is used to execute the dynamically constructed command string, allowing for flexibility in command execution based on conditions.

6. Preroute Standard Cells

```
set_attribute -objects [get_nets VDD] -name net_type -value power
set_attribute -objects [get_nets VSS] -name net_type -value ground

connect_pg_net -net VDD [get_pins -physical_context */VDD]
connect_pg_net -net VSS [get_pins -physical_context */VSS]

check_mv_design
```

- Set Attributes: This part of the script sets attributes for specific nets in the design. The nets `VDD` and `VSS` are identified as power and ground nets, respectively. This classification helps the tool in handling these nets appropriately during further processing.
- Connect PG Nets: Power and ground nets are connected to their respective pins across the design. This step ensures that all components in the design have the correct power connections, which is critical for both simulation and physical implementation.
- Check Multi-Voltage Design: This command (`check_mv_design`) likely verifies aspects of the design related to its operation at multiple voltage levels, which is important in modern low-power IC designs.

7. Verify DRC/LVS

```
check_lvs > ../../reports/${DESIGN_NAME}.lvs.rpt
```

- Runs layout versus schematic (LVS) checks to ensure that the layout accurately represents the schematic. The results are directed to a report file. LVS checks are crucial for confirming that the physical layout of the IC matches the schematic or circuit diagram.

8. Reports

```
report_design -all > ../../reports/${DESIGN_NAME}.PR_summary.rpt

report_timing -capacitance -transition_time -input_pins -nets -delay_type
report_timing -capacitance -transition_time -input_pins -nets -delay_type
```

- Report Design: Generates a comprehensive report of the design, which includes various parameters and states of the design elements, useful for review and documentation.
- Report Timing: Generates detailed timing reports. There are two reports:
  - One for the maximum delay conditions, which helps identify the worst-case scenarios in terms of signal propagation delays.
  - One for the minimum delay conditions, useful for analyzing the best-case scenario.

9. Write Verilog

```
write_verilog -include {pg_netlist unconnected_ports} ../../results/${DES
write_verilog -exclude {physical_only_cells} ../../results/${DESIGN_NAME}

## write_verilog for comparison with a DC netlist (no pg, no physical onl
set write_verilog_dc_cmd_root "write_verilog -compress gzip -exclude {sca
set write_verilog_dc_cmd "$write_verilog_dc_cmd_root ../../results/${DESI
eval ${write_verilog_dc_cmd}
```

- Write Verilog Files: Outputs Verilog files for the design, with specific exclusions and inclusions:
  - The first command writes a Verilog netlist that includes power grids and unconnected ports.
  - The second command excludes physical-only cells (like fillers, which don't affect logical functionality) from the Verilog output.
- Write Verilog for DC Netlist: A more complex command sequence is set up to write a Verilog file specifically for comparison with a DC (Design Compiler) netlist. This version excludes various elements that are not crucial for synthesis comparisons and compresses the output using gzip for efficiency. The command is constructed dynamically and executed using `eval`, which allows for command customization based on potentially varying requirements or conditions.

10. Write Verilig Netlist

- **For Prime Time**

```
set write_verilog_pt_cmd_root "write_verilog -compress gzip -exclude {sca
set write_verilog_pt_cmd "$write_verilog_pt_cmd_root ../../results/${DESI
eval ${write_verilog_pt_cmd}
```

- This command configures and runs a script to generate a Verilog netlist excluding specific physical elements like power grids and various filler types, optimized for PrimeTime analysis which focuses on timing without needing these elements. The netlist includes diodes and decoupling capacitors (DCAP) necessary for accurate leakage power analysis.
- **For Formality**

```
set write_verilog_fm_cmd_root "write_verilog -compress gzip -exclude {sca
set write_verilog_fm_cmd "$write_verilog_fm_cmd_root ../../results/${DESI
eval ${write_verilog_fm_cmd}
```

- Generates a Verilog netlist for use in Formality, which is a tool for equivalence checking between RTL code and synthesized netlists. It includes power grids but excludes supply statements and physical-only cells, focusing on the functional aspects required for logical checks.
- **For LVS**
- Prepares a Verilog netlist tailored for LVS, which checks the correspondence between the schematic and the layout. This script excludes non-functional elements like filler cells, as these do not impact the electrical behavior checked during LVS.

11. Generate Design Files

- **SDC Output**

```
write_sdc -output ../../results/${DESIGN_NAME}.out.sdc
```

- Outputs the Synopsys Design Constraints (SDC) file, essential for timing analysis by specifying constraints like clock frequencies, input and output delays, etc.
- **SPEF Output**

```
report_timing -crosstalk_delta
write_parasitics -format SPEF -output ../../results/${DESIGN_NAME}.out.sp
```

- Generates a Standard Parasitic Exchange Format (SPEF) file, which details the parasitic capacitance and resistance of the design, vital for accurate post-layout timing simulation.
- **DEF Output**

```
write_def ../../results/${DESIGN_NAME}.out.def
```

- Writes out the Design Exchange Format (DEF), which is a standard format for representing physical placement of a design's components and routing in EDA tools.
- **GDS Output**

```
save_block -as ${DESIGN_NAME}_7_finished

write_gds -design ${DESIGN_NAME}_7_finished \
    -layer_map $Gds_map_file \
    -keep_data_type \
    -fill include \
    -output_pin all \
    -merge_files $Std_cell_gds \
    -long_names \
    ../../results/${DESIGN_NAME}.gds
```

- Saves the current block of the design and writes a GDSII file, which is a binary file format representing the layout of the physical design used for fabrication. It includes various configurations like layer mapping and inclusion of fills.

Outputs on Terminal

- Insert reduntant vias

```
################################################################################Insert_Redundant_Vias
add_redundant_vias



Using libraries: i2c_master_top saed14rvt_tt0p8v25c saed14rvt_tt0p8v25c_physical_only EXPLORE_physical_only
Visiting block i2c_master_top:temp_route_ends.design
Design 'i2c_master_top' was successfully linked.
Warning: Layer M8 does not have a preferred direction, assigning to horizontal. (ZRT-025)
Warning: Layer M9 does not have a preferred direction, assigning to vertical. (ZRT-025)
Warning: Layer MRDL does not have a preferred direction, assigning to horizontal. (ZRT-025)
Cell Min-Routing-Layer = M1
Cell Max-Routing-Layer = M9
Turn off antenna since no rule is specified
Info: number of net_type_blockage: 0
Via on layer (VIA1) needs more than one tracks
Warning: Layer M1 pitch 0.074 may be too small: wire/via-down 0.074, wire/via-up 0.098. (ZRT-026)
Via on layer (VIA1) needs more than one tracks
Warning: Layer M2 pitch 0.060 may be too small: wire/via-down 0.098, wire/via-up 0.060. (ZRT-026)
Via on layer (VIA3) needs more than one tracks
Warning: Layer M3 pitch 0.074 may be too small: wire/via-down 0.060, wire/via-up 0.080. (ZRT-026)
Wire on layer (M4) needs more than one tracks
Via on layer (VIA3) needs more than one tracks
Via on layer (VIA4) needs more than one tracks
Warning: Layer M4 pitch 0.074 may be too small: wire/via-down 0.107, wire/via-up 0.105. (ZRT-026)
Via on layer (VIA4) needs more than one tracks
Warning: Layer M5 pitch 0.120 may be too small: wire/via-down 0.135, wire/via-up 0.107. (ZRT-026)
Via on layer (VIA8) needs more than one tracks
Warning: Layer M8 pitch 0.120 may be too small: wire/via-down 0.107, wire/via-up 0.135. (ZRT-026)
Wire on layer (M9) needs more than one tracks
Via on layer (VIA8) needs more than one tracks
Via on layer (VIARDL) needs more than one tracks
Warning: Layer M9 pitch 0.120 may be too small: wire/via-down 0.135, wire/via-up 1.570. (ZRT-026)
Via on layer (VIARDL) needs more than one tracks
Warning: Layer MRDL pitch 0.600 may be too small: wire/via-down 4.500, wire/via-up 0.600. (ZRT-026)
Transition layer name: M3(2)
When applicable Min-max layer allow_pin_connection mode will allow paths of length 0.98 outside the layer range.
Warning: Standard cell pin SAEDRVT14_AO221_0P5/B1 has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_BUF_ECO_1/A has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_MUXI2_U_0P5/D1 has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_EO2_1/A2 has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_EO2_0P5/A2 has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_EO2_0P5/X has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_AO221_2/A1 has no valid via regions. (ZRT-044)
Warning: Standard cell pin SAEDRVT14_AO221_2/B1 has no valid via regions. (ZRT-044)
```

- Print auto via ladder

```
Printing options for 'route.auto_via_ladder.*'

[Dr init] Elapsed real time: 0:00:00
[Dr init] Elapsed cpu  time: sys=0:00:00 usr=0:00:00 total=0:00:00
[Dr init] Stage (MB): Used   10  Alloctr   10  Proc    9
[Dr init] Total (MB): Used   35  Alloctr   36  Proc 2465

Redundant via optimization will attempt to replace the following vias:

   VIA12SQ_C     -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1

   VIA12SQ_C(r) -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1

  VIA12BAR1_C     -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1

  VIA12BAR1_C(r) -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1

  VIA12BAR2_C     -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1

  VIA12BAR2_C(r) -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1

   VIA12LG_C     -> VIA12SQ_C_1x2(r) VIA12BAR2_C_1x2(r) VIA12SQ_1x2(r) VIA12LG_C_1x2
                    VIA12BAR1_C_1x2    VIA12SQ_C_2x1(r) VIA12SQ_2x1(r) VIA12SQ_C_1x2
                    VIA12BAR1_C_2x1(r) VIA12SQ_1x2    VIA12BAR2_C_1x2    VIA12BAR1_C_2x1
                    VIA12LG_C_2x1(r) VIA12SQ_C_2x1    VIA12SQ_2x1    VIA12BAR2_C_2x1
```

- run ICV

```
Start to run ICV ...
.....................................
... 2% complete Elapsed Time=0:00:42
..... 15% complete Elapsed Time=0:01:30
..... 25% complete Elapsed Time=0:01:31
..... 35% complete Elapsed Time=0:01:31
..... 45% complete Elapsed Time=0:01:32
..... 55% complete Elapsed Time=0:01:32
..... 65% complete Elapsed Time=0:01:33
..... 75% complete Elapsed Time=0:01:33
..... 90% complete Elapsed Time=0:01:52
..... 95% complete Elapsed Time=0:01:52
..... 95% complete Elapsed Time=0:01:52
.....Updated tech file has been written at: /home/u110/u110061217/2024ASoC/lab_formal_release/lab_formal_release/lab4_finishing/work/signoff_fill_run/icv_run_1/tech.tf

ICV is done!
Run status:
Fills created successfully. For more details, see /home/u110/u110061217/2024ASoC/lab_formal_release/lab_formal_release/lab4_finishing/work/signoff_fill_run/icv_run_1/signoff_create_metal_fill.log
Overall engine Time=0:01:59 Highest command Mem=0.505 GB
1
################################################################################Preroute_Standard_Cells
```

- Finish

```
################################################################################Save_Block
save_block
Information: Saving block 'i2c_master_top:temp_route_ends.design'
1
save_lib
Saving library 'i2c_master_top'
1
close_block
Information: Decrementing open_count of block 'i2c_master_top:temp_route_ends.design' to 1. (DES-022)
1
close_lib
Closing library 'i2c_master_top'
Information: The net parasitics of block i2c_master_top are cleared. (TIM-123)
1
exit
Maximum memory usage for this session: 519.24 MB
Maximum memory usage for this session including child processes: 3109.68 MB
CPU usage for this session:     47 seconds (  0.01 hours)
Elapsed time for this session:    706 seconds (  0.20 hours)
Thank you for using IC Compiler II.
date >  step7_finishing
date > all
```

Reports

- Data mismatch

```
Report : Data Mismatches
Version: R-2020.09-SP3
Date   : Sat Apr 27 14:19:54 2024
****************************************
No mismatches exist on the design.
----------------------------------------
Number of Written DEF Constructs
----------------------------------------
VERSION                       : 1
DIVIDERCHAR                   : 1
BUSBITCHARS                   : 1
DESIGN                        : 1
UNITS                         : 1
PROPERTYDEFINITIONS           : 1
DIEAREA                       : 1
ROW                           : 67
TRACKS                        : 20
VIAS                          : 27
NONDEFAULTRULES               : 1
COMPONENTS                    : 4427
PINS                          : 35
PINPROPERTIES                 : 35
FILLS                         : 2
SPECIALNETS                   : 98
NETS                          : 734
1
################################################################################GDS_OUT
```

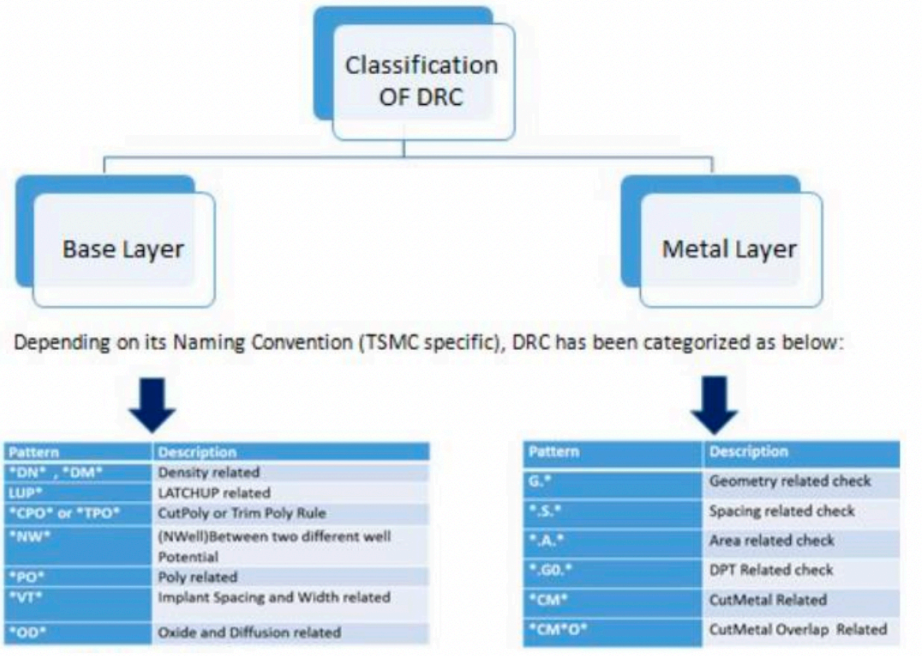
- Reduntant via conversion report

```
Redundant via conversion report:
--------------------------------

  Total optimized via conversion rate = 55.27% (2735 / 4948 vias)

    Layer VIA1       =  8.97% (172    / 1917    vias)
        Weight 1     =  8.97% (172     vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    = 91.03% (1745    vias)
    Layer VIA2       = 83.40% (2331   / 2795    vias)
        Weight 1     = 83.40% (2331    vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    = 16.60% (464     vias)
    Layer VIA3       = 98.23% (222    / 226     vias)
        Weight 1     = 98.23% (222     vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    =  1.77% (4       vias)
    Layer VIA4       = 100.00% (10     / 10      vias)
        Weight 1     = 100.00% (10      vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    =  0.00% (0       vias)

  Total double via conversion rate    = 55.27% (2735 / 4948 vias)

    Layer VIA1       =  8.97% (172    / 1917    vias)
    Layer VIA2       = 83.40% (2331   / 2795    vias)
    Layer VIA3       = 98.23% (222    / 226     vias)
    Layer VIA4       = 100.00% (10     / 10      vias)

  The optimized via conversion rate based on total routed via count = 55.27% (2735 / 4948 vias)

    Layer VIA1       =  8.97% (172    / 1917    vias)
        Weight 1     =  8.97% (172     vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    = 91.03% (1745    vias)
    Layer VIA2       = 83.40% (2331   / 2795    vias)
        Weight 1     = 83.40% (2331    vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    = 16.60% (464     vias)
    Layer VIA3       = 98.23% (222    / 226     vias)
        Weight 1     = 98.23% (222     vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    =  1.77% (4       vias)
    Layer VIA4       = 100.00% (10     / 10      vias)
        Weight 1     = 100.00% (10      vias)
        Un-optimized =  0.00% (0       vias)
        Un-mapped    =  0.00% (0       vias)
```

Verificaion (IC Validator, Formality)

7. IC Validator - DRC

- **Goal:** Verify Design Rule Check for the target technology



Pattern	Description
DN , *DM*	Density related
LUP*	LATCHUP related
CPO or *TPO*	CutPoly or Trim Poly Rule
NW	(NWell)Between two different well Potential
PO	Poly related
VT	Implant Spacing and Width related
OD	Oxide and Diffusion related

Pattern	Description
G.*	Geometry related check
.S.	Spacing related check
.A.	Area related check
.G0.	DPT Related check
CM	CutMetal Related
*CM*O*	CutMetal Overlap Related

Description of command in scripts (tcl file)

- This part doesn't have tcl file since it just do the DRC check.

Description of command in Makefile

`Makefile`

1. Executable Definitions (Commented)

```
#ICC2_EXEC = /global/apps/icc2_2019.12/bin/icc2_shell
#ICC2_EXEC = /global/apps/icc2_2022.12/bin/icc2_shell
#DC_EXEC = /global/apps/syn_2022.12/bin/dc_shell
#ICV_EXEC = /global/apps/icv_2022.12-SP5/bin/LINUX.64/icv
ICV_EXEC = icv
```

- These lines define possible paths to different executables related to the IC design tools like ICC2 and Synopsys Design Compiler. They are commented out, possibly indicating alternatives or historical choices.
- ICV_EXEC is defined as icv, suggesting that the icv command should be directly available in the system's environment path, without specifying a directory.

2. Directory Definitions

```
6  LOGS_DIR = ./LOG
7  #TECH_DIR = ../../../SAED14_EDK_LAB/tech
8  TECH_DIR = /home/course/ee5252/lab_snps_flow/SAED14_EDK_LAB/SAED14_EDK_LA
```

- LOGS_DIR is set to ./LOG, indicating a directory relative to the current directory where logs will be stored.
- TECH_DIR is set to an absolute path that likely contains technology-related data or configuration files essential for the DRC process.

3. GDSII DRC Settings (DRC Configuration (GDSII Format))

```
9  ###################################################################
10 #standalone icv drc setting (GDS)
11 ###################################################################
12 TOP_CELL = i2c_master_top
13 FORMATE = GDSII
14 LAYOUT_FILE = $(INPUT_LIBRARY_PATH)/i2c_master_top.gds
15 LAYER_MAPPING_FILE = $(TECH_DIR)/milkyway/saed14nm_1p9m_gdsout_mw.map
16 INPUT_LIBRARY_PATH = ../../results
17 RUN_DIR = ./signoff_drc_run
18 RUNSET_FILE = $(TECH_DIR)/icv_drc/saed14nm_1p9m_drc_rules.rs
19 #NUMBER_HOST = 4
```

- TOP_CELL specifies the name of the top cell in the layout to be checked. FORMATE is apparently a typo and should be FORMAT, set to GDSII, indicating the format of the layout files.
- LAYOUT_FILE points to the GDSII file of the TOP_CELL, assumed to be located in a directory set by INPUT_LIBRARY_PATH.
- LAYER_MAPPING_FILE points to a file that defines how different layers in the GDS file map to the technology's layer definitions.
- INPUT_LIBRARY_PATH is set relative to the current directory, pointing to where input files like the GDSII layout are expected to be found.
- RUN_DIR specifies a directory for running the ICV process.
- RUNSET_FILE points to a file containing the rules for the DRC process.

4. Makefile Targets

```
console:
	$(ICV_EXEC)
setup:
	test -d $(LOGS_DIR) || mkdir $(LOGS_DIR)
	test -d $(RUN_DIR) || mkdir $(RUN_DIR)
	date > setup
run_icv_DRC: setup
	$(ICV_EXEC) -vue -c $(TOP_CELL) -f $(FORMATE) -lf $(LAYER_MAPPING_
	date >  run_icv_DRC
all: setup run_icv_DRC
	date >  all
clean:
	ls | grep -v "Makefile" | xargs rm -r
```

- **`console`** Target
  - Simply runs the ICV executable without any arguments, useful for manual control or testing.
- **`setup`** Target
  - Checks if LOGS_DIR and RUN_DIR exist and creates them if they don't. Records the current date to a file named setup, possibly to log when setup was last run.
- **`run_icv_DRC`** Target
  - Depends on the setup target. It runs the ICV with various flags set for performing a DRC:
    - vue: Likely triggers a verbose or UI mode.
    - -c, -f, -lf, -p, -i, -l: Specify the cell name, format, layer mapping file, input path, layout file, and run directory respectively.
    - Outputs of the ICV run are piped into tee to log both to the console and to a log file in LOGS_DIR. It captures both standard output and errors.
    - Marks the completion date in a file run_icv_DRC.
- **`all`** Target
  - Runs both setup and run_icv_DRC, then logs the date to a file named all. This target is for doing a full setup and run sequence in one command.
- **`clean`** Target
  - Cleans up the directory by deleting all files except the Makefile. This is done using ls, grep, and xargs rm -r, which lists all files, filters out "Makefile", and removes everything else.

Outputs on Terminal

- IC Validation

```
IC Validator Machine Memory Report
ws29         : Average = 1.256 GB, Peak = 2.906 GB

Overall Disk Usage Disk=0.006 GB
Network Disk Usage Peak=0.006 GB (no group)
Group File Disk Usage Peak=0.000 GB
Overall engine Time=0:02:12 Highest command Mem=0.099 GB

Overall Master Mem=3.279 GB
IC Validator is done.
date >  run_icv_DRC
date >  all
```

- Execution

```
Host startup begins.
Host startup done: 2 successes, 0 failures.
       ws29: 2 total cores, maximum 2 commands in parallel.
Selecting "ws29" as Primary host

System Startup Time=0:00:42 User=0.01 Sys=0.06 Mem=0.876 GB


Command execution begins.  Details recorded in the summary and log files:
    /home/u110/u110000107/lab_formal_release/lab_icv_drc/work/run_details/i2c_master_top.sum
    /home/u110/u110000107/lab_formal_release/lab_icv_drc/work/run_details/saed14nm_1p9m_drc_rules.dp.log

Running ...
  ICV_Engine run is   1% complete.    Elapsed Time=0:01:31
  ICV_Engine run is   2% complete.    Elapsed Time=0:01:49
  ICV_Engine run is   3% complete.    Elapsed Time=0:01:49
  ICV_Engine run is   4% complete.    Elapsed Time=0:01:50
  ICV_Engine run is   5% complete.    Elapsed Time=0:01:50
  ICV_Engine run is  10% complete.    Elapsed Time=0:01:51
  ICV_Engine run is  15% complete.    Elapsed Time=0:01:51
  ICV_Engine run is  20% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  25% complete.    Elapsed Time=0:01:53
  ICV_Engine run is  30% complete.    Elapsed Time=0:01:53
  ICV_Engine run is  35% complete.    Elapsed Time=0:01:54
  ICV_Engine run is  40% complete.    Elapsed Time=0:01:55
  ICV_Engine run is  45% complete.    Elapsed Time=0:01:56
  ICV_Engine run is  50% complete.    Elapsed Time=0:01:57
  ICV_Engine run is  55% complete.    Elapsed Time=0:01:57
  ICV_Engine run is  60% complete.    Elapsed Time=0:01:58
  ICV_Engine run is  65% complete.    Elapsed Time=0:01:59
  ICV_Engine run is  70% complete.    Elapsed Time=0:01:59
  ICV_Engine run is  75% complete.    Elapsed Time=0:02:00
  ICV_Engine run is  80% complete.    Elapsed Time=0:02:01
  ICV_Engine run is  85% complete.    Elapsed Time=0:02:01
  ICV_Engine run is  90% complete.    Elapsed Time=0:02:02
  ICV_Engine run is  95% complete.    Elapsed Time=0:02:03
  ICV_Engine run is 100% complete.    Elapsed Time=0:02:05

Completing error storage...
  Overall error storage time: User=1.67 Sys=0.24 Mem=0.026 GB

Generating i2c_master_top.LAYOUT_ERRORS...
  Generation Time=0:00:01  User=1.04 Sys=0.09 Mem=0.001 GB

   Check Time=0:00:01  User=0.01 Sys=0.00 Mem=0.004 GB
```

- Memory Machine Usage Report

```
IC Validator Machine Memory Report
ws29         : Average = 1.256 GB, Peak = 2.906 GB

Overall Disk Usage Disk=0.006 GB
Network Disk Usage Peak=0.006 GB (no group)
Group File Disk Usage Peak=0.000 GB
Overall engine Time=0:02:12 Highest command Mem=0.099 GB

Overall Master Mem=3.279 GB
IC Validator is done.
date >  run_icv_DRC
date >  all
```

Reports

`i2c_master_top.RESULTS`

```
                                RESULTS: NOT CLEAN

        #   #   ###   #####      ####  #      #####   ###   #   #
        ##  #  #   #    #       #      #      #      #   #  ##  #
        # # #  #   #    #       #      #      ####   #####  # # #
        #  ##  #   #    #       #      #      #      #   #  #  ##
        #   #   ###     #        ####  #####  #####  #   #  #   #

================================================================================
```

```
--------------------------------------------------------------------------------
Results Summary
--------------------------------------------------------------------------------

Rule and DRC Error Summary

372 total rules were run.
0 rules NOT EXECUTED.
52 rules have violations.
There are 424523 total violations.
Refer to i2c_master_top.LAYOUT_ERRORS

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

Rule                                                    Violations Found
DIFF.S.1 Minimum spacing must be 0.074                  v = 3
M1.A.1  M1 minimum area must be 0.005                   v = 167
M1.I.1                                                  v = 286
M1.S.1.2 Minimum notch must be 0.055                    v = 91
M1.S.1                                                  v = 975
M1.S.2.1                                                v = 314
M1.S.2                                                  v = 89
M1.S.3                                                  v = 113
M1.S.6                                                  v = 1
M1.S.6_1                                                v = 5
M2.I.1                                                  v = 514
M2.S.1.1                                                v = 19
M2.S.1.2 Minimum notch must be 0.055                    v = 1317
M2.S.1                                                  v = 8899
```

```
M2.S.2.1                                                    v = 633
M2.S.2                                                      v = 351949
M2.S.3                                                      v = 345
M3.S.1.1                                                    v = 34
M3.S.1.2 Minimum notch must be 0.055                        v = 1676
M3.S.1                                                      v = 794
M3.S.2                                                      v = 579
M3.S.3                                                      v = 589
M3.S.6                                                      v = 3
M4.S.1.2 Minimum notch must be 0.04                         v = 387
M4.S.1                                                      v = 3
M4.S.2                                                      v = 7
M4.W.1                                                      v = 479
M5.S.1.2 Minimum notch must be 0.04                         v = 14
M5.S.2                                                      v = 3
M5.W.1                                                      v = 12
NIMP.O.2 Overlap of NIMP and PIMP is not allowed            v = 112
NWELL.S.3 N+Active minimum space to N_Well must be 0.05     v = 160
PIMP.O.2 Overlap of NIMP and PIMP is not allowed            v = 112
PIMP.S.2                                                    v = 160
PIMP.S.3 Minimum spacing to N+Active in P_Well must be      v = 160
0.05
PO.S.4                                                      v = 504
PO.W.1                                                      v = 504
SBLK.C.2                                                    v = 182
VIA0.C.1                                                    v = 182
VIA0.E.1                                                    v = 10943
```

```
VIA0.S.1                                                    v = 9
VIA0.W.1                                                    v = 6703
VIA0.W.2 and VIA0.W.2a                                      v = 6703
VIA0_3.E.2                                                  v = 25236
VIA1.E.1                                                    v = 1
VIA1.S.1                                                    v = 17
VIA1.S.2                                                    v = 1
VIA1.S.3                                                    v = 47
VIA2.S.1                                                    v = 1527
VIA2.S.2                                                    v = 131
VIA2.S.3                                                    v = 827
VIA3.S.1                                                    v = 2
```

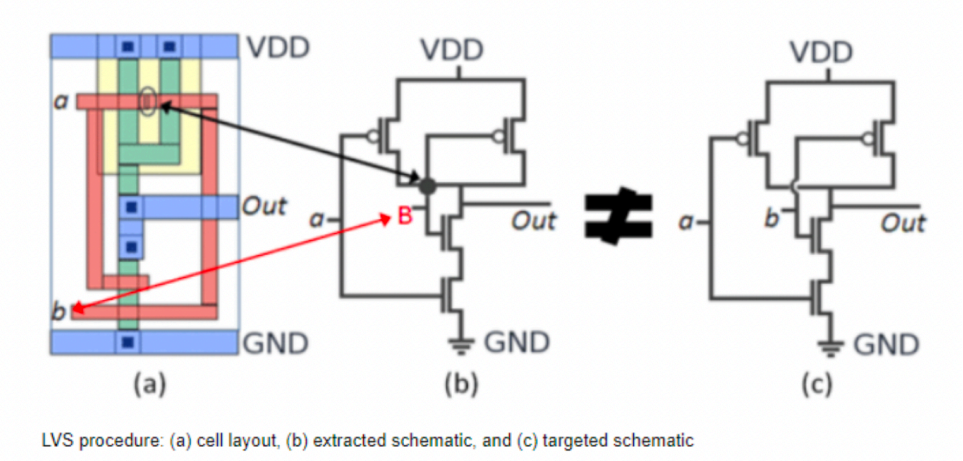
i2c_master_top.ERRORS

```
         LAYOUT ERRORS RESULTS: ERRORS

      ##### ####  ####   ###  ####   ###
      #     #   # #   # #   # #   # #
      ####  ####  ####  #   # ####   ###
      #     #  #  #  #  #   # #  #      #
      ##### #   # #   #  ###  #   # ####


 ============================================================================
```

- Further discussion about above error is posted in the last section.

8. IC Validator - LVS

- **Goal:** Verify Layout Versus Schematic matching the design



LVS procedure: (a) cell layout, (b) extracted schematic, and (c) targeted schematic

Description of command in scripts (Makefile or tcl file)

`.CDL files`

- In the script folder, .cdl files contains the subckts for checking the LVS.

`run_lvs.csh`

```
1 #!/bin/tcsh -f
```

- This line indicates that the script should be executed using the tcsh shell. The -f option prevents tcsh from reading any startup files, which could affect the script's execution environment.

```
2 #write_gds -design ${DESIGN_NAME}_7_finished -layer_map $Gds_map_file -ke
```

- This is a commented-out command that, if active, would write a GDS file with specific parameters such as design name, layer mapping, and additional flags controlling the output format. It is included for reference or conditional use but is inactive in the current script.

```
3 set GDSII_FILE = "../../results/i2c_master_top_wostd.gds"
```

- Sets the variable GDSII_FILE to the path of a GDSII file. This file is used in the verification process and is located relative to the current directory.

```
4 #set GDSII_FILE =  "/remote/testcases/TC102/042022/4196128/LIB_AJ/ICC2/Fl
5 set GDSII_TOPCELL_NAME = "i2c_master_top"
```

- Provides an alternative (and much longer) path for the GDSII_FILE variable. It is commented out, suggesting it could be used under different circumstances or for testing.

```
5 # icv_nettran -verilog i2c_master_top.v.pg -outType
```

- Defines the variable GDSII_TOPCELL_NAME, which specifies the top cell name of the GDSII file, used in processing.

```
6 SPICE -outName verilog2spice.sp
```

- A commented-out command for converting a Verilog file into a SPICE netlist, indicative of tasks that might be performed in different configurations of this script.

```
7 set SPICE_FILE = "./netlist2spice.sp"
```

- Sets the path to a SPICE file, which will be used in the LVS process. The file is located in the current directory.

```
8 set NETLIST_TOPCELL_NAME = "i2c_master_top"
```

- Assigns the name of the top cell in the netlist to a variable, likely used in the verification process.

```
9 \rm -rf run_details *.vue *.net *LAYOUT_ERROR *LVS_ERRORS *.RESULTS icv.l
```

- This command removes all files and directories related to previous runs that match the specified patterns. This cleanup ensures no residual files affect the current verification process.

```
icv -vue -vueshortALL -create_lvs_short_outputALL \
-i $GDSII_FILE \
-c $GDSII_TOPCELL_NAME \
-s $SPICE_FILE \
-stc $NETLIST_TOPCELL_NAME \
-sf SPICE \
../script/saed14nm_1p9m_lvs_runset.rs.tim
```

- This line executes the IC Validator (icv) with several options:
  - `-vue` : Possibly invokes a graphical user interface or a specific mode in the ICV.
  - `-vueshortALL` : Could specify a detailed summary of results.
  - `-create_lvs_short_outputALL` : Likely directs ICV to create a concise output for all LVS checks.
- The command includes several flags that specify inputs for the verification:
  - `-i $GDSII_FILE` : Specifies the input GDSII file.
  - `-c $GDSII_TOPCELL_NAME` : Specifies the top cell of the GDSII file.
  - `-s $SPICE_FILE` : Specifies the SPICE file used for schematic verification.
  - `-stc $NETLIST_TOPCELL_NAME` : Specifies the top cell of the netlist.
  - `-sf SPICE` : Indicates the format of the schematic file.
- Finally, it specifies a runset file located in ../script/saed14nm_1p9m_lvs_runset.rs.tim, which contains configuration settings for the LVS run.

Outputs on Terminal

```
Command execution begins.  Details recorded in the summary and log files:
    /home/u110/u110000107/lab_formal_release/lab_icv_lvs/work/run_details/i2c_master_top.sum
    /home/u110/u110000107/lab_formal_release/lab_icv_lvs/work/run_details/saed14nm_1p9m_lvs_runset.rs.dp.log

Running ...
  ICV_Engine run is   1% complete.    Elapsed Time=0:01:23
  ICV_Engine run is   2% complete.    Elapsed Time=0:01:31
  ICV_Engine run is   3% complete.    Elapsed Time=0:01:31
  ICV_Engine run is   4% complete.    Elapsed Time=0:01:49
  ICV_Engine run is   5% complete.    Elapsed Time=0:01:49
  ICV_Engine run is  10% complete.    Elapsed Time=0:01:50
  ICV_Engine run is  15% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  20% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  25% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  30% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  35% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  40% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  45% complete.    Elapsed Time=0:01:52
  ICV_Engine run is  50% complete.    Elapsed Time=0:01:53
  ICV_Engine run is  55% complete.    Elapsed Time=0:01:53
  ICV_Engine run is  60% complete.    Elapsed Time=0:01:54
  ICV_Engine run is  65% complete.    Elapsed Time=0:01:55
  ICV_Engine run is  70% complete.    Elapsed Time=0:01:56
  ICV_Engine run is  75% complete.    Elapsed Time=0:01:56
  ICV_Engine run is  80% complete.    Elapsed Time=0:01:57
  ICV_Engine run is  85% complete.    Elapsed Time=0:01:59
  ICV_Engine run is  90% complete.    Elapsed Time=0:02:00
  ICV_Engine run is  95% complete.    Elapsed Time=0:02:30
Comparing schematic and layout netlists.
LVS compare start time    : 2024-04-27 02:35:29
  ICV_Compare run is   0% complete.    Elapsed Time=0:00:41
  ICV_Compare run is   5% complete.    Elapsed Time=0:00:41
  ICV_Compare run is  10% complete.    Elapsed Time=0:00:41
  ICV_Compare run is  15% complete.    Elapsed Time=0:00:41
  ICV_Compare run is  20% complete.    Elapsed Time=0:00:41
  ICV_Compare run is  25% complete.    Elapsed Time=0:00:41
  ICV_Compare run is  30% complete.    Elapsed Time=0:00:41
  ICV_Compare run is 100% complete.    Elapsed Time=0:00:42
LVS compare end time      : 2024-04-27 02:36:11
Total runtime for LVS compare Time=0:00:42  User=1.28 Sys=0.30 Mem=0.028 GB
   Check Time=0:00:42  User=0.02 Sys=0.01 Mem=0.011 GB

  ICV_Engine run is 100% complete.    Elapsed Time=0:03:57
```

Reports

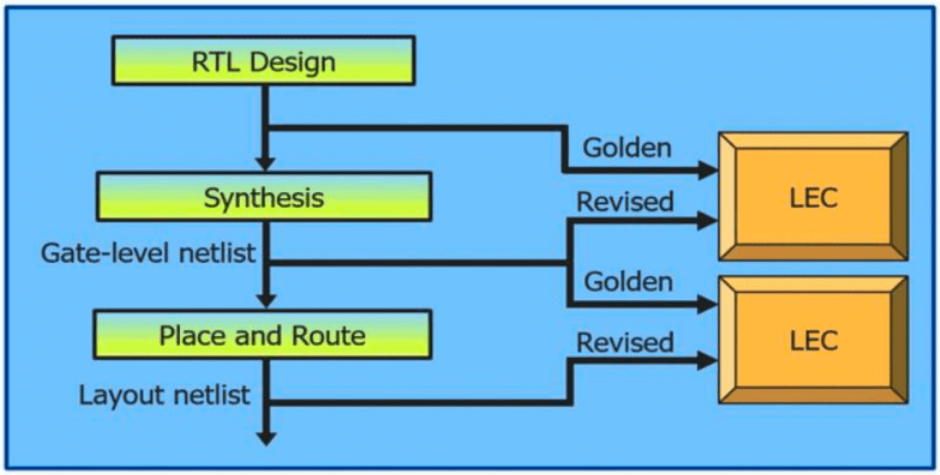
i2c_master_top.RESULTS

```
                    LVS Compare Results: FAIL

                     #####   ###   #   #
                     #      #   #  #   #
                     ####   #####  #   #
                     #      #   #  #   #
                     #      #   #  #   #####

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

                  DRC and Extraction Results: NOT CLEAN

          #   #   ###   #####      ####  #      #####   ###   #   #
          ##  #  #   #    #       #      #      #      #   #  ##  #
          # # #  #   #    #       #      #      ####   #####  # # #
          #  ##  #   #    #       #      #      #      #   #  #  ##
          #   #   ###     #        ####  #####  #####  #   #  #   #

================================================================================
```

- Further discussion about above error is posted in the last section.

9. Formality

- **Goal:** Verify the behavior of circuit and layout



Description of command in scripts (tcl file)

`run_fomality_cmd.tcl`

1. Common Script Source:

```
1 source ../../common/common.tcl
```

- This command sources a TCL file located two directories up and within a folder named common. Sourcing a TCL file imports all the procedures and variables defined in that file into the current script, making them available for use.

2. Variable setting

```
2 set DESIGN_NAME i2c_master_top
```

- Sets a variable DESIGN_NAME with the value i2c_master_top. This is likely the name of the design being processed.

3. Library reading:

```
3 read_db "$Std_cell_lib $Ram_lib"
```

- Reads database files or libraries specified by the variables $Std_cell_lib and $Ram_lib. These variables are expected to be defined in the previously sourced common.tcl file or elsewhere in the environment.

4. Initializes the variable

```
4 set hdlin_interface_only "" ; # set block_box
```

- Initializes the variable hdlin_interface_only to an empty string. The comment # set block_box could indicate a placeholder for future modifications or might be miswritten and intended as black_box. It suggests that this variable might be used to configure interface-only or black-box settings in a design tool.

5. Reads a Verilog file

```
5 read_verilog -r <absolute path>/lab_formal_release/lab_formal/work/../../
```

- Reads a Verilog file located at a specified absolute path, using the -r option which typically implies reading the file recursively or as a reference. The -work_library WORK argument suggests setting or using a specific working library named WORK.

6. command

```
6 set_top -auto
```

- This command typically sets the top-level module for processing automatically based on the context or content of the previously loaded files.

7. Reads another Verilog file

```
7 read_verilog -i <absolute path>/University_LAB_0220/lab_formal_release/la
```

- Reads another Verilog file (compressed with gzip judging by the .gz extension) from a specified path, with the -i flag possibly indicating an import operation.

8. Top module setting

```
8 set_top i:WORK/i2c_master_top
```

- Explicitly sets the top module to i2c_master_top located in a library or namespace WORK, prefixed with i: which might specify a particular instance or context.

9. Configuration setting

```
9 set verification_set_undriven_signals BINARY
```

- Sets a configuration or a flag to handle undriven signals in a verification environment, possibly configuring them to be treated as binary values.

10. Enable Verification

```
10 set verification_check_gate_reserve_gating true
```

- Enables a specific verification check related to gate reservation or gating logic, ensuring it adheres to expected norms or rules.

11. Verify

```
11 verify
```

- Initiates the verification process based on the preceding settings and configurations.

12. Exit

```
12 quit
```

gen_fomality_cmd.tcl

1. Identification

```
1 #!/usr/bin/tclsh
```

- This shebang line tells the system this is a script to be executed with Tcl shell.

2. Source the enviroment

```
2 source ../../common/common.tcl
```

- Sources a common Tcl script that sets up the environment or defines common variables and procedures used across various scripts.

3. Open or create the File

```
3 set fp [open "../script/run_formality_cmd.tcl" w+]
```

- Opens or creates a file named run_formality_cmd.tcl in write-plus mode (which means it will write data, and if the file doesn't exist, it will create it).

4.

```
4 puts $fp "source ../../common/common.tcl"
```

- Writes a line into the file to source the same common Tcl script as the current script does.

5. New line

```
5 puts $fp "\n"
```

- Adds a newline for better readability in the generated script.

6. Variable Setting

```
6 puts $fp "set DESIGN_NAME ${DESIGN_NAME}"
```

- Writes a line to set the DESIGN_NAME variable in the generated script, using the value from the current script's environment.

7. Path and name loading

```
7 puts $fp "read_db \"\$Std_cell_lib \$Ram_lib\""
```

- Commands the generated script to load databases using paths or names stored in the Std_cell_lib and Ram_lib variables. This suggests that these variables should be set in the sourced common.tcl script.

8. Default configuration

```
8 puts $fp "set hdlin_interface_only \"\" ; # set block_box"
```

- Sets hdlin_interface_only to an empty string and adds a comment. This could be a placeholder or default configuration.

9. Indicating Comment

```
9 puts $fp "\#refence design (the golden design)"
```

- Adds a comment indicating the following commands deal with the reference or golden design.

10. Direction

```
10 puts $fp "read_verilog -r ${Core_compile} - work_library WORK"
```

- Directs the generated script to read a Verilog file with options to use recursive reading from a variable Core_compile, which should specify the path or pattern.

11. Top Module Read

```
11 puts $fp "set_top -auto"
```

- Automatically sets the top module of the design being read.

12. Comment

```
12 puts $fp "\#implement design (the target design)"
```

- Adds a comment to clarify that the following commands in the generated script pertain to the implementation design.

13. Verilog Design Reading

```
13 puts $fp "read_verilog -i ${FM_IMPLEMENTED_DESIGN}"
```

- Reads the implemented Verilog design from the path or file specified by FM_IMPLEMENTED_DESIGN.

14. Specifying

```
14 puts $fp "set_top i:WORK/${DESIGN_NAME}"
```

- Specifies explicitly setting the top module for the implemented design, using a namespace or library WORK.

15. Undriven Sinals Setting

```
15 puts $fp "set verification_set_undriven_signals BINARY"
```

- Sets the treatment of undriven signals in the verification environment to be binary.

16. Checking Enable

```
16 puts $fp "set verification_check_gate_reserve_gating true"
```

- Enables checking for reserve gating as part of the verification process.

17. Initiation

```
17 puts $fp "verify"
```

Initiates the verification process in the generated script.

18. Exit

```
18 puts $fp "quit"
```

- Commands the generated script to exit the tool or environment after completing the verification.

19. Done

```
19 close $fp
```

- Closes the file pointer, effectively saving the generated script.

Outputs on Terminal

```
                              Formality (R)

                Version S-2021.06 for linux64 - May 28, 2021

                    Copyright (c) 1988 - 2021 Synopsys, Inc.
   This software and the associated documentation are proprietary to Synopsys,
 Inc. This software may only be used in accordance with the terms and conditions
 of a written license agreement with Synopsys, Inc. All other use, reproduction,
   or distribution of this software is strictly prohibited.  Licensed Products
     communicate with Synopsys servers for the purpose of providing software
    updates, detecting software piracy and verifying that customers are using
    Licensed Products in conformity with the applicable License Key for such
  Licensed Products. Synopsys will use information gathered in connection with
    this process to deliver software updates and pursue software pirates and
                                  infringers.

 Inclusivity & Diversity - Visit SolvNetPlus to read the "Synopsys Statement on
            Inclusivity and Diversity" (Refer to article 000036315 at
                        https://solvnetplus.synopsys.com)

Build: 7085620
Hostname: ws24
Current time: Sat Apr 27 13:27:40 2024

****************************************************************************************
****

Status:  Verifying...

********************************** Matching Results ******************************
****
 167 Compare points matched by name
 0 Compare points matched by signature analysis
 0 Compare points matched by topology
 19 Matched primary inputs, black-box outputs
 0(0) Unmatched reference(implementation) compare points
 0(0) Unmatched reference(implementation) primary inputs, black-box outputs
 2(0) Unmatched reference(implementation) unread points
****************************************************************************************
****


******************************** Verification Results ****************************
****
Verification SUCCEEDED
----------------------
 Reference design: r:/WORK/i2c_master_top
 Implementation design: i:/WORK/i2c_master_top
 167 Passing compare points
----------------------------------------------------------------------------------------
----
Matched Compare Points     BBPin    Loop   BBNet     Cut    Port     DFF     LAT   T
OTAL
----------------------------------------------------------------------------------------
----
Passing (equivalent)           0       0       0       0      14     153       0
 167
Failing (not equivalent)       0       0       0       0       0       0       0
   0
****************************************************************************************
****
1
quit

Maximum memory usage for this session: 634 MB
CPU usage for this session: 3.28 seconds ( 0.00 hours )
Current time: Sat Apr 27 13:28:25 2024
Elapsed time: 108 seconds ( 0.03 hours )

Thank you for using Formality (R)!
date > run_formality_cmd
date >  all
```

Discussion and Observation

- Environment Settings
  Setting up the EDA environment is the first critical step. It includes configuring paths and sourcing scripts for the necessary Synopsys and other EDA tools. Proper setup is vital for ensuring that the tools operate correctly and have access to the necessary technology libraries and licenses. The workbook emphasizes the

importance of having the right environment settings, which could be an indication of how sensitive the tools are to configuration and paths.

- Lab Synthesis
  The synthesis lab aims to transform HDL code from RTL to a gate-level netlist. This is the step where the design is first converted from an abstract description into a representation that can be physically implemented. This step is critical as it sets the tone for all subsequent stages. The gate-level netlist will be used in later labs for floorplanning, placement, routing, and more.
- Lab1 Floorplan
  Floorplanning is about defining the chip's physical structure, which includes the placement of I/O pads and macro cells, as well as the creation of the power and ground structure. It sets the stage for the physical implementation and is key for achieving an optimal layout that meets area, power, and performance objectives.
- Lab2 Placement & Route
  This stage focuses on the detailed placement of standard cells and the routing of wires that connect these cells. Successful completion of this stage results in a design that is ready for extraction and verification.
- Lab StarRC
  StarRC is used for extracting parasitic information from the design. This data is critical for accurate post-layout simulation and timing analysis, as parasitics significantly affect the performance of the IC at higher frequencies. This lab indicates the importance of considering parasitic effects early in the design to prevent late-stage design failures.
- Lab PrimeTime
  Static Timing Analysis (STA) and power analysis are performed in this lab using PrimeTime. STA ensures that the design meets its timing requirements, which is crucial for functionality. Power analysis, on the other hand, ensures that the power consumption is within acceptable limits, which is increasingly important in modern IC design.
- Lab4 Chip Finishing
  Here, the process involves inserting filler cells and exporting the GDSII file, which is the final design file used for manufacturing the chip. This step signifies the transition from design to production.
- Lab IC Validator - DRC (**Not clean**)
  Design Rule Checking (DRC) is performed to ensure the design adheres to specific fabrication constraints. This step is critical as non-compliance to these rules can lead to manufacturing issues or chip failure.
  However, DRC check is not clean in this lab. The error in the i2c_master_top.RESULTS shows there are minimum width violations, spacing, enclosure violations, and so on and so forth. The reason is probably because the different manufacturing process results in different layout style, which is not corresponding to the DRC and LVS validator.

- Lab IC Validator - LVS (**Failed**)
  Layout Versus Schematic (LVS) verification ensures that the physical layout matches the original schematic. If not, it can lead to functional errors, thus this step is crucial for ensuring design integrity.
  DRC checks have revealed some errors, which in turn have caused issues in LVS verification.
  In our workbook, environment setting were adjusted, which we suspect led to variations of manufacture process files, consequently resulting in DRC not being clean. Moreover, the differences in manufacturing process also resulted in LVS failure with the different layout style.
  Some other reasons might be missing or extra devices, connection errors, parameter mismatches...
- Lab Formality
  Formality checks the logical equivalence between the RTL code and the synthesized gate-level netlist or the layout. It ensures that throughout the various transformations the design has undergone, its behavior remains consistent with the original specification.
- General Observations:
  The document stresses the importance of a structured approach to IC design, with a strong emphasis on verification at each stage. There's an underlying implication that the Synopsys tools are interdependent, with each tool building on the output of the previous one. It also highlights the complexity of the design flow, where each stage plays a critical role in ensuring the final product meets the original design specifications. The labs seem to offer hands-on experience with real-world scenarios, including dealing with errors and troubleshooting, which is essential for students or practitioners in the field. The workflow presented ensures that students are not just focused on design, but also on the preparation for manufacturing, emphasizing the industry-relevant aspects of SOC&ASIC design.