OpenROAD Flow Improvement

Neural Semiconductor Limited

Seamless Technology Service

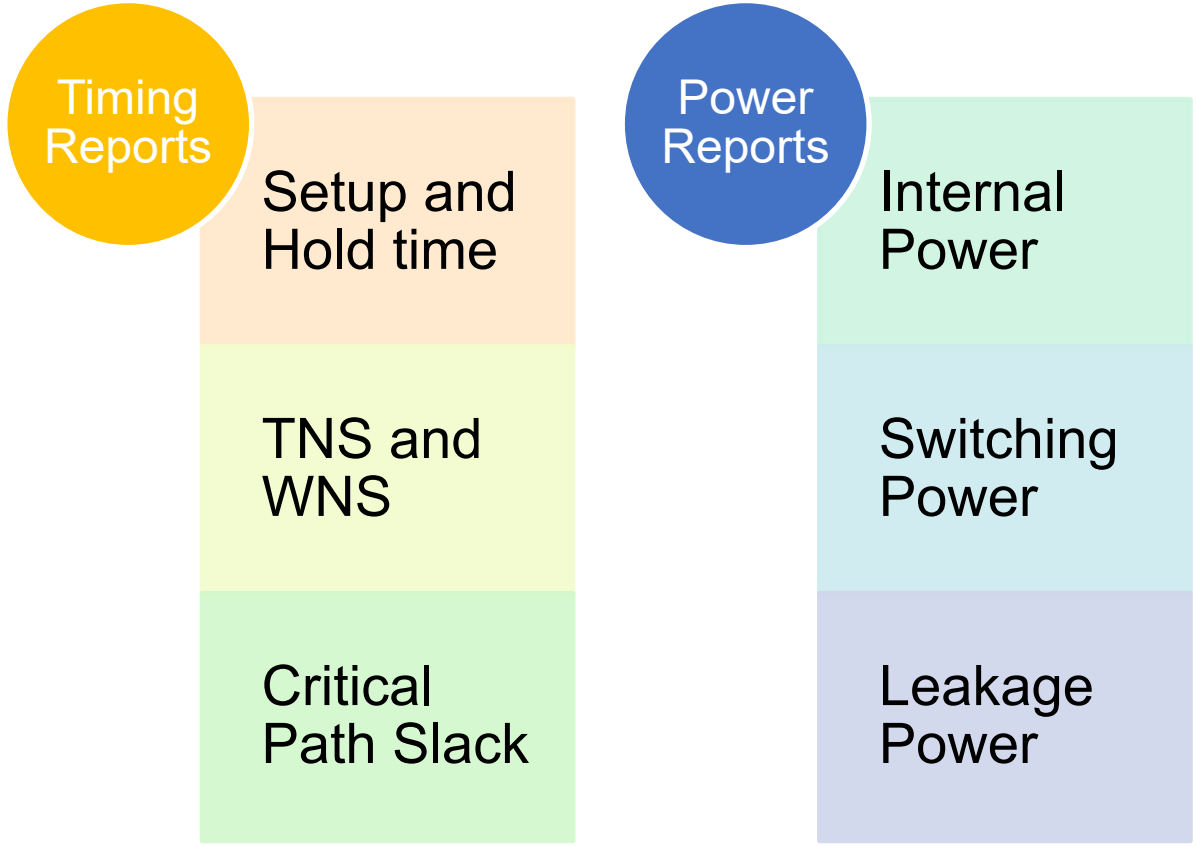
Presented by
Fahim Faisal

NEURAL SEMICONDUCTOR

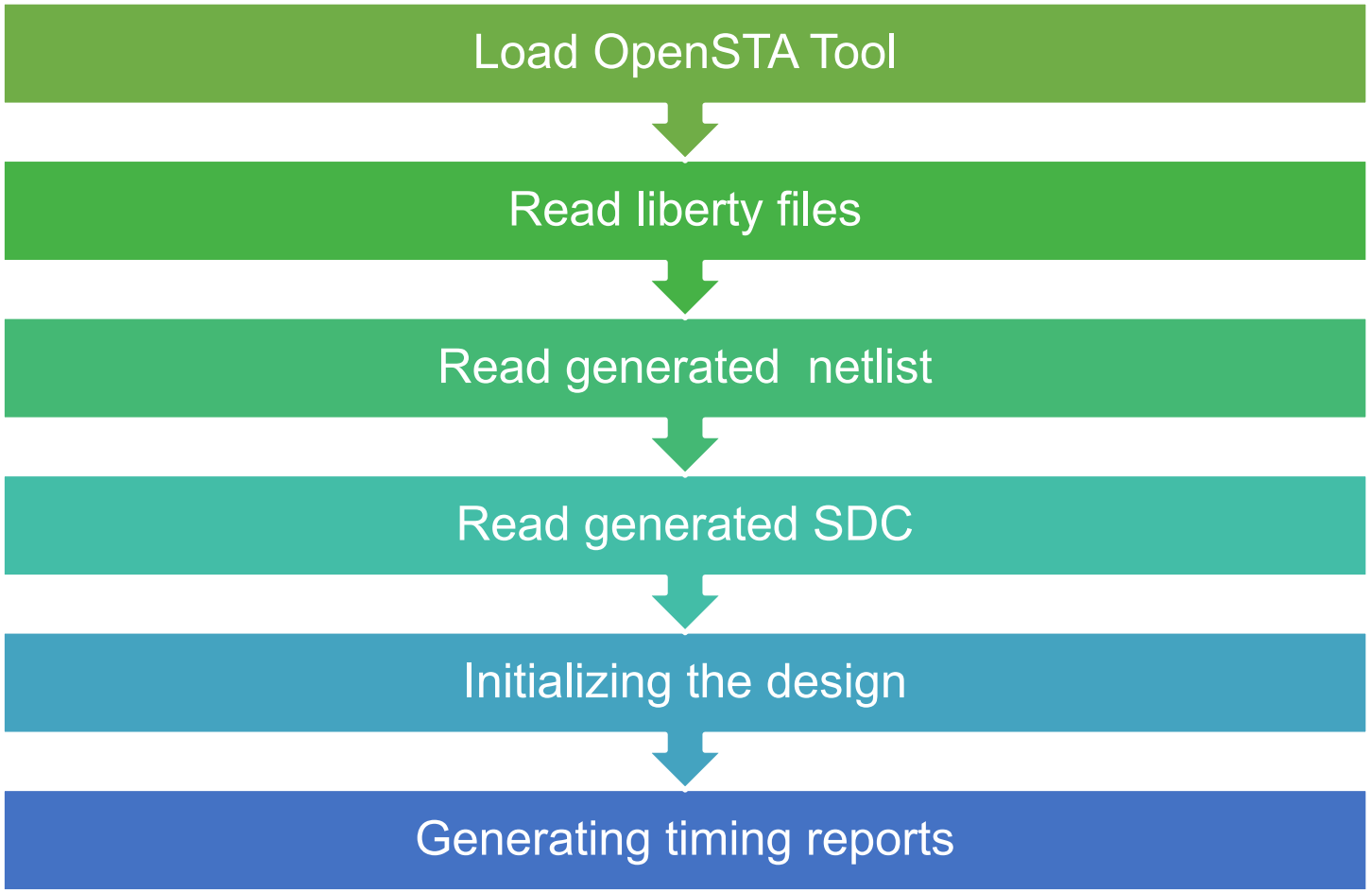
Limitation of the OpenROAD synthesis flow

Most EDA tools that perform synthesis

- perform multiple checks.
- allow the designer to get an early estimate. about the feasibility of the design.
- However, the OpenROAD flow lacked this feature.
- It was explored how to add these necessary checks after synthesizing a design and generate timing reports successfully.



Flow


Load OpenSTA Tool
Read liberty files
Read generated netlist
Read generated SDC
Initializing the design
Generating timing reports

Version 1

Use the OpenSTA tool to take the Verilog and SDC file generated by Yosys and perform timing and power checks

This was hard coded as a tcl file and sourced to verify the feasibility of the idea

```
########################## synthesis_timing.tcl ###################################
set lib {platforms/asap7/lib/asap7sc7p5t_AO_RVT_FF_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_INVBUF_RVT_FF_nldm_220122.lib.gz \
platforms/asap7/lib/asap7sc7p5t_OA_RVT_FF_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_SIMPLE_RVT_FF_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_SEQ_RVT_FF_nldm_220123.lib \
platforms/asap7/lib/fakeram7_256x32.lib \
platforms/asap7/lib/asap7sc7p5t_SEQ_RVT_FF_nldm_220123.lib \
platforms/asap7/lib/asap7sc7p5t_AO_RVT_SS_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_INVBUF_RVT_SS_nldm_220122.lib.gz \
platforms/asap7/lib/asap7sc7p5t_OA_RVT_SS_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_SEQ_RVT_SS_nldm_220123.lib \
platforms/asap7/lib/asap7sc7p5t_SIMPLE_RVT_SS_nldm_211120.lib.gz \
platforms/asap7/lib/fakeram7_256x32.lib \
platforms/asap7/lib/asap7sc7p5t_SEQ_RVT_SS_nldm_220123.lib \
platforms/asap7/lib/asap7sc7p5t_AO_RVT_TT_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_INVBUF_RVT_TT_nldm_220122.lib.gz \
platforms/asap7/lib/asap7sc7p5t_OA_RVT_TT_nldm_211120.lib.gz \
platforms/asap7/lib/asap7sc7p5t_SEQ_RVT_TT_nldm_220123.lib \
platforms/asap7/lib/asap7sc7p5t_SIMPLE_RVT_TT_nldm_211120.lib.gz \
platforms/asap7/lib/fakeram7_256x32.lib \
platforms/asap7/lib/asap7sc7p5t_SEQ_RVT_TT_nldm_220123.lib} ;#all required .lib files
foreach library $lib {
    read_liberty $library
}
read_verilog results/asap7/riscv32i/base/1_1_yosys.v
link_design riscv_top
read_sdc results/asap7/riscv32i/base/1_synth.sdc
source scripts/report_metrics_synthesis.tcl ;#modified version of report_metrics.tcl for synthesis
report_metrics synthesis
```

Read the library of the design manually

Read the Verilog and SDC file and source the report generation file from the flow

```
######################### report_metrics_synthesis.tcl ##############################
proc report_metrics { when {include_erc true} {include_clock_skew true} } {
  puts "\n=========================================================================="
  puts "$when check_setup"
  puts "--------------------------------------------------------------------------"
  check_setup

  puts "\n=========================================================================="
  puts "$when report_tns"
  puts "--------------------------------------------------------------------------"
  report_tns

  puts "\n=========================================================================="
  puts "$when report_wns"
  puts "--------------------------------------------------------------------------"
  report_wns

  puts "\n=========================================================================="
  puts "$when report_worst_slack"
  puts "--------------------------------------------------------------------------"
  report_worst_slack

  if {$include_clock_skew} {
    puts "\n=========================================================================="
    puts "$when report_clock_skew"
    puts "--------------------------------------------------------------------------"
    report_clock_skew
  }
  puts "\n=========================================================================="
  puts "$when report_checks -path_delay min"
  puts "--------------------------------------------------------------------------"
  report_checks -path_delay min -fields {slew cap input nets fanout} -format full_clock_expanded

  puts "\n=========================================================================="
  puts "$when report_checks -path_delay max"
  puts "--------------------------------------------------------------------------"
  report_checks -path_delay max -fields {slew cap input nets fanout} -format full_clock_expanded
```

In the “report_metrics_synthesis.tcl” file

- report_tns_metric
- report_worst_slack_metric
- report_clock_skew_metric
- report_clock_skew_metric -hold
- report_erc_metrics
- report_power_metric
- reporting design area

Are omitted as these do not work with the OpenSTA tool.

```
puts "\n==================================
puts "$when report_power"
puts "----------------------------------
if {[info exists ::env(CORNERS)]} {
  foreach corner $::env(CORNERS) {
    puts "Corner: $corner"
    report_power -corner $corner
    report_power_metric -corner $corner
  }
  unset corner
} else {
  report_power
}

puts ""
```

```
source ../setup_env.sh
sta
source scripts/synthesis_timing.tcl
```

Version 2

The “report_metrics.tcl script has been modified to work with both openSTA and “synthesis_timing.tcl” was integrated with ORFS.

```
puts "\n========================================================================="
puts "$when report_tns"
puts "-------------------------------------------------------------------------"
report_tns
if {$when != "synthesis"} {
report_tns_metric
}
```

```
if {$include_erc} {
  puts "\n========================================================================="
  puts "$when report_check_types -max_slew -max_cap -max_fanout -violators"
  puts "-------------------------------------------------------------------------"
  report_check_types -max_slew -max_capacitance -max_fanout -violators
  if {$when != "synthesis"} {
  report_erc_metrics
  }
```

```
puts "\n========================================================================="
puts "$when report_worst_slack"
puts "-------------------------------------------------------------------------"
report_worst_slack
if {$when != "synthesis"} {
report_worst_slack_metric
}

if {$include_clock_skew} {
  puts "\n========================================================================="
  puts "$when report_clock_skew"
  puts "-------------------------------------------------------------------------"
  report_clock_skew
  if {$when != "synthesis"} {
  report_clock_skew_metric
  report_clock_skew_metric -hold
  }
}
```

```
if {$when != "synthesis"} {
puts "\n========================================================================="
puts "$when report_design_area"
puts "-------------------------------------------------------------------------"

report_design_area
report_design_area_metrics
```

the original file could not be read properly so this file was modified accordingly

```
$(RESULTS_DIR)/1_synth.sdc: $(SDC_FILE)
        mkdir -p $(RESULTS_DIR) $(LOG_DIR) $(REPORTS_DIR)
        cp $< $@
        # report timing for synthesis
        sta synthesis_timing.tcl -exit | tee $(LOG_DIR)/1_1_synth_timing_report.log

clean_synth:
        rm -f  $(RESULTS_DIR)/1_*.v $(RESULTS_DIR)/1_synth.sdc
        rm -f  $(REPORTS_DIR)/synth_*
        rm -f  $(LOG_DIR)/1_*
        rm -f  $(SYNTH_STOP_MODULE_SCRIPT)
        rm -rf _tmp_yosys-abc-*
```

the log file that houses the timing report data is named “1_1_synth_timing_report.log” so that when “make clean_synthesis” or “make clean_all” is run, it will clear all logs with “1_*”.

```
source ../setup_env.sh
```

Here, there was an issue with openSTA tool not being setup through flow. so, before using flow source “setup_env.sh” in the flow directory.

Version 3

Now, the “synthesis_timing.tcl” script was made dynamic and also group path was integrated to ”report_metrics.tcl”.

It is able to perform the timing and power checks for any design after synthesis.

```
########################## synthesis_timing.tcl ###################################
source $::env(SCRIPTS_DIR)/read_liberty.tcl
read_verilog $::env(RESULTS_DIR)/1_synth.v
link_design $::env(DESIGN_NAME)
read_sdc $::env(RESULTS_DIR)/1_synth.sdc
source $::env(SCRIPTS_DIR)/report_metrics.tcl
report_metrics synthesis
```

```
$(RESULTS_DIR)/1_synth.sdc: $(SDC_FILE)
	mkdir -p $(RESULTS_DIR) $(LOG_DIR) $(REPORTS_DIR)
	cp $< $@
	sta $(SCRIPTS_DIR)/synthesis_timing.tcl -exit | tee $(LOG_DIR)/1_1_synth_timing_report.log

clean_synth:
	rm -f  $(RESULTS_DIR)/1_*.v $(RESULTS_DIR)/1_synth.sdc
	rm -f  $(REPORTS_DIR)/synth_*
	rm -f  $(LOG_DIR)/1_*
	rm -f  $(SYNTH_STOP_MODULE_SCRIPT)
	rm -rf _tmp_yosys-abc-*
```

in make flow, using sta tool with “synthesis_timing.tcl” is also made dynamic for ease of use.

```
################################### .bashrc ###################################
source_tool=$(find /home/fahim/ -name setup_env.sh) >/dev/null
source $source_tool >/dev/null
```

For Cloud machine there was no problem using OpenSTA with this modifications but, for local machine sta tool could not be found thus needed sourcing.
“setup_env.sh” script. this is added to “.bashrc” so that creating a new terminal will automatically source “setup_env.sh”

```
########################## 1_1_synth_timing_report.log ########################

.........
.........
.........
.........

============================================================================
synthesis check_setup
----------------------------------------------------------------------------

============================================================================
synthesis report_tns
----------------------------------------------------------------------------
tns 0.00

============================================================================
synthesis report_wns
----------------------------------------------------------------------------
wns 0.00

============================================================================
synthesis report_worst_slack
----------------------------------------------------------------------------
worst slack 494.97

.........
.........
.........
.........
```

a new log file named “1_1_synth_timing_report.log” is generated which contains all necessary timing information:

```
============================================================================
synthesis report_checks -path_delay min
----------------------------------------------------------------------------
Startpoint: reset (input port clocked by clk)
Endpoint: riscv/dp/pcreg/_67_ (removal check against rising-edge clock clk)
Path Group: **async_default**
Path Type: min

Fanout     Cap    Slew   Delay    Time   Description
---------------------------------------------------------------------------
                  0.00    0.00    0.00   clock clk (rise edge)
                          0.00    0.00   clock network delay (ideal)
                        200.00  200.00 v input external delay
                  0.00    0.00  200.00 v reset (in)
     1    1.46                          reset (net)
                  0.00    0.00  200.00 v riscv/dp/pcreg/_34_/A (INVx3_ASAP7_75t_R)
                 59.70   27.19  227.19 ^ riscv/dp/pcreg/_34_/Y (INVx3_ASAP7_75t_R)
    32   29.08                          riscv/dp/pcreg/_32_ (net)
                 59.70    0.00  227.19 ^ riscv/dp/pcreg/_67_/SETN (DFFASRHQNx1_ASAP7_75t_R)
                                 227.19   data arrival time

                  0.00    0.00    0.00   clock clk (rise edge)
                          0.00    0.00   clock network delay (ideal)
                          0.00    0.00   clock reconvergence pessimism
                                  0.00 ^ riscv/dp/pcreg/_67_/CLK (DFFASRHQNx1_ASAP7_75t_R)
                         11.41   11.41   library removal time
                                 11.41   data required time
---------------------------------------------------------------------------
                                 11.41   data required time
                               -227.19   data arrival time
---------------------------------------------------------------------------
                                 215.77   slack (MET)
.........
.........
.........
.........

============================================================================
synthesis hold_violation_count
----------------------------------------------------------------------------
hold violation count 0

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

Path Delays

```
synthesis critical path delay
--------------------------------------------------------------------------------
1095.5405

================================================================================
synthesis critical path slack
--------------------------------------------------------------------------------
494.9744

================================================================================
synthesis slack div critical path delay
--------------------------------------------------------------------------------
45.180840

================================================================================
synthesis report_power
--------------------------------------------------------------------------------
Group                  Internal  Switching    Leakage      Total
                          Power      Power      Power      Power (Watts)
----------------------------------------------------------------
Sequential             1.01e-03   4.82e-05   1.58e-07   1.06e-03  12.2%
Combinational          6.81e-03   2.41e-04   5.17e-04   7.57e-03  87.8%
Macro                  0.00e+00   0.00e+00   0.00e+00   0.00e+00   0.0%
Pad                    0.00e+00   0.00e+00   0.00e+00   0.00e+00   0.0%
----------------------------------------------------------------
Total                  7.82e-03   2.89e-04   5.17e-04   8.63e-03 100.0%
                          90.7%       3.3%       6.0%
```

Power Checks

```
######################### report_metrics.tcl (modified) ##############################
proc report_metrics { when {include_erc true} {include_clock_skew true} } {

  if {[info exists ::env(USE_GROUP_PATH)] && $::env(USE_GROUP_PATH)} {
        set inputs [lsearch -inline -all -not -exact [all_inputs] [all_clocks]]
        group_path -name in2reg -from $inputs -to [all_registers]
        group_path -name reg2out -from [all_registers] -to [all_outputs]
        group_path -name in2out -from $inputs -to  [all_outputs]
        group_path -name reg2reg -from [all_registers] -to [all_registers]
  }
```

Also, added group paths to “report_metrics.tcl” in this version. This helps to check timing paths based on group which is useful when debugging certain paths for causes of timing violation

Timing paths reported for each groups (in2out, rin2reg, reg2out and reg2reg). Here are some snippets of them from 1_1_synth_timing.log

```
========================================================================
synthesis report_checks -path_delay max
------------------------------------------------------------------------
Startpoint: instr[4] (input port clocked by clk)
Endpoint: dataadr[31] (output port clocked by clk)
Path Group: in2out
Path Type: max
```

```
Startpoint: riscv/dp/pcreg/_89_ (rising edge-triggered flip-flop clocked by clk)
Endpoint: dataadr[31] (output port clocked by clk)
Path Group: reg2out
Path Type: max
```

```
Startpoint: instr[4] (input port clocked by clk)
Endpoint: riscv/dp/rf/_13020_ (falling edge-triggered flip-flop clocked by clk')
Path Group: in2reg
Path Type: max
```

```
Startpoint: riscv/dp/pcreg/_89_ (rising edge-triggered flip-flop clocked by clk)
Endpoint: riscv/dp/rf/_13020_ (falling edge-triggered flip-flop clocked by clk')
Path Group: reg2reg
Path Type: max
```

Limitation of DRC check options

Challenges

In the OpenROAD flow, DRC checks work on nangate45 and sky130hd with make drc command.

But for asap7, There was no rule file (.lydrc) to run DRC checks

.lydrc file works for Klayout version 0,27.1 or higher

but the ORFS was using Klayout 0.26.4

Observation

- Required files:
  - 6_final.gds
  - $(PLATFORM).lydrc
- Output Files:
  - 6_drc.log
  - 6_drc_count.rpt
  - 6_drc.lyrdb

This file was missing and had to be added from a Third party

If we had those files we could get these outputs and check DRC

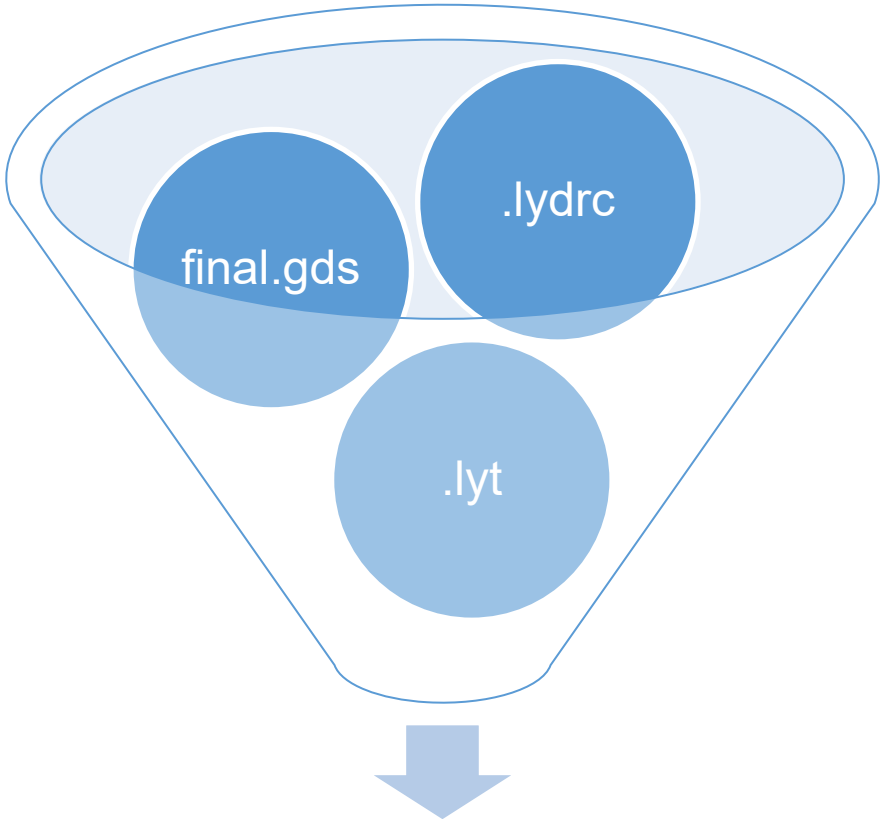
Work Done to resolve the issue

- To add support for DRC on a new platform $(PLATFORM):
  - Ensure that a KLayout tech file (.lyt) exists for the platform. If not, create it.
  - Create (or copy) a KLayout .lydrc file at flow/platforms/$(PLATFORM)/drc/$(PLATFORM).lydrc
  - Add the following line to flow/platforms/$(PLATFORM)/config.mk:

```
################ for lvs drc ######################
export KLAYOUT_DRC_FILE = $(PLATFORM_DIR)/drc/asap7.lydrc
export KLAYOUT_LVS_FILE = $(PLATFORM_DIR)/lvs/asap7.lylvs
```

We should then be able to run make DRC and perform DRC checks

Flow



DRC reports

Outputs

- For Riscv32i
  - design found 3313 DRC violation which shown in drc_count.rpt file and all the violated location and type shown in lyrdb file

Improvements

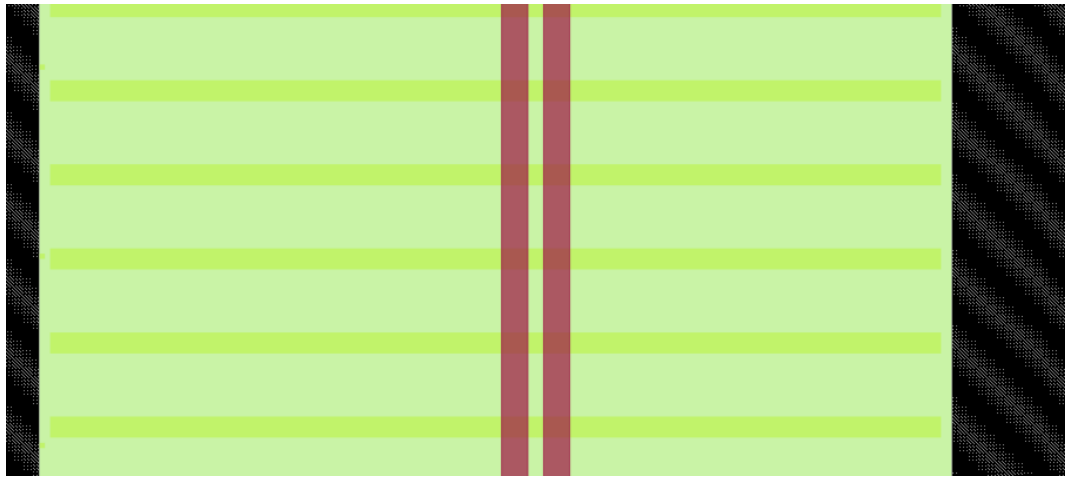
- It would be better if the OpenROAD flow had rule files integrated in the flow for more consistent DRC checking

Lack of Power grid connection to Macro

Challenges

In the OpenROAD flow, for riscv32i design macros were placed but they lacked connection to the design power grid.

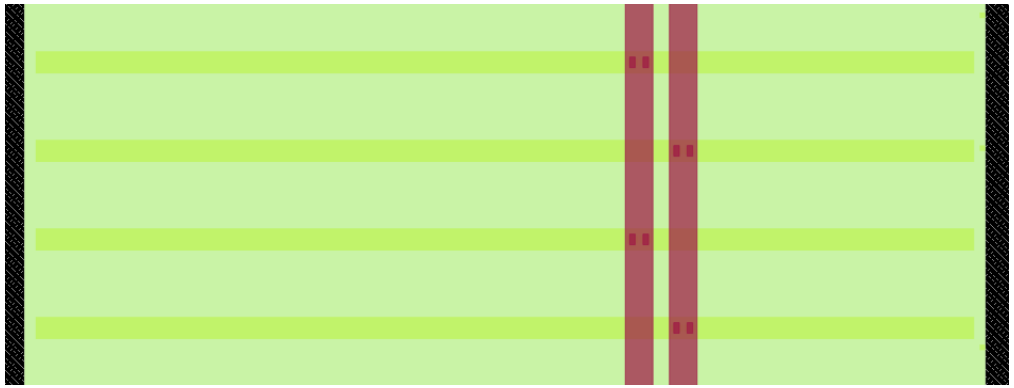
Observation



Analyzing the issue revealed that power stripes for both VDD and VSS at M5 were present above macros but they lacked via connection to macros power grid which is at M4.

Work Done to resolve the issue

- To establish power connection to macros, vias are dropped from M5 to M4.
  - This modification is added to the existing powergrid generating script: “flow/platforms/asap7/openRoad/pdn/grid_strategy-M2-M5-M7.tcl”
  - Thus successfully establishing power connection to macros.



Generation of power via from M5 to M4 to connect power stripe to macros.

```
####################################
# global connections
####################################
add_global_connection -net {VDD} -inst_pattern {.*} -pin_pattern {^VDD$} -power
add_global_connection -net {VDD} -inst_pattern {.*} -pin_pattern {^VDDPE$}
add_global_connection -net {VDD} -inst_pattern {.*} -pin_pattern {^VDDCE$}
add_global_connection -net {VSS} -inst_pattern {.*} -pin_pattern {^VSS$} -ground
add_global_connection -net {VSS} -inst_pattern {.*} -pin_pattern {^VSSE$}
global_connect
####################################
# voltage domains
####################################
set_voltage_domain -name {CORE} -power {VDD} -ground {VSS}
####################################
# standard cell grid
####################################
define_pdn_grid -name {top} -voltage_domains {CORE}
add_pdn_stripe -grid {top} -layer {M1} -width {0.018} -pitch {0.54} -offset {0} -followpins
add_pdn_stripe -grid {top} -layer {M2} -width {0.018} -pitch {0.54} -offset {0} -followpins
#add_pdn_stripe -grid {top} -layer {M5} -width {0.12} -spacing {0.072} -pitch {11.88} -offset {0.300
add_pdn_stripe -grid {top} -layer {M5} -width {0.12} -spacing {0.072} -pitch {14.88} -offset {7.600}
add_pdn_stripe -grid {top} -layer {M6} -width {0.288} -spacing {0.096} -pitch {12} -offset {0.513}
add_pdn_connect -grid {top} -layers {M1 M2}
add_pdn_connect -grid {top} -layers {M2 M5}
add_pdn_connect -grid {top} -layers {M5 M6}

####################################
# macro via connection
####################################
 define_pdn_grid -name {core_macro} -voltage_domains {CORE} -macro -orient {R0 R180 MX MY} -halo {2.0 2.0 2.0 2.0} -default -grid_over_boundary
 add_pdn_connect -grid {core_macro} -layers {M4 M5}
```

The modified “grid_strategy-M2-M5-M7.tcl” with the suggested macro power via connection.

Thank You