

---

UM–SJTU JOINT INSTITUTE

**System-on-Chip Design**  
**(ECE4810J)**

---

LABORATORY REPORT

**Lab 6. Getting Started with the OpenLane  
ASIC Design Flow and Skywater 130nm  
Technology**

**Group 2**

Name: Haochen Wu	ID: 518021910558
Name: Siyuan Zhang	ID: 518370910180
Name: Yihua Liu	ID: 518021910998

Date: November 19, 2021

# Contents

<b>1</b>	<b>Overview</b>	<b>2</b>
<b>2</b>	<b>TCL</b>	<b>2</b>
2.1	Problem 1 . . . . .	2
2.2	Problem 2 . . . . .	3
<b>3</b>	<b>Setting up the OpenLane flow</b>	<b>3</b>
3.1	Displaying the Final Layout(3.4 in lab manual) . . . . .	3
3.2	Adding a Design(3.5 in lab manual) . . . . .	4
<b>4</b>	<b>Questions</b>	<b>10</b>

# 1 Overview

In this lab, we will get familiarised with the TCL(“Tool Command Language”) script language grammar and OpenLane VLSI design flow together with Skywater 130nm PDK. OpenLane is an open-source VLSI flow built around open-source tools, where a collection of scripts that run these tools gather in the right order and transform their inputs and outputs appropriately.

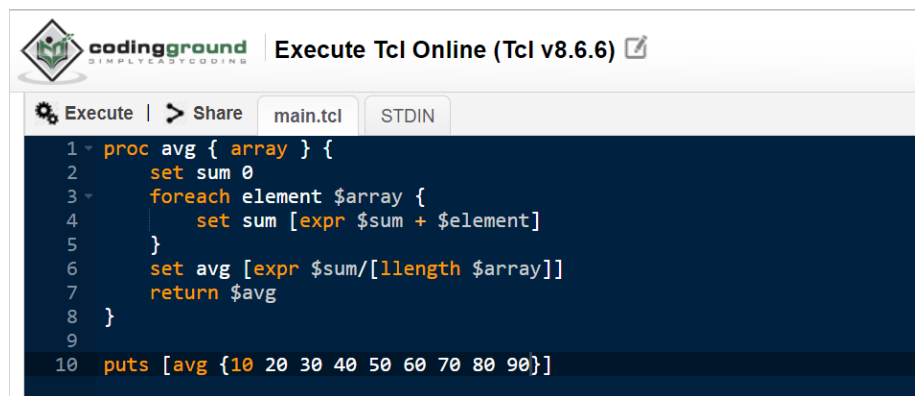
## 2 TCL

TCL ( “Tool Command Language”) is a shell script language that is widely used in most digital design flows nowadays.

### 2.1 Problem 1

Write a procedure (proc) called “avg” that takes an array of numbers with any length and return the average of it.

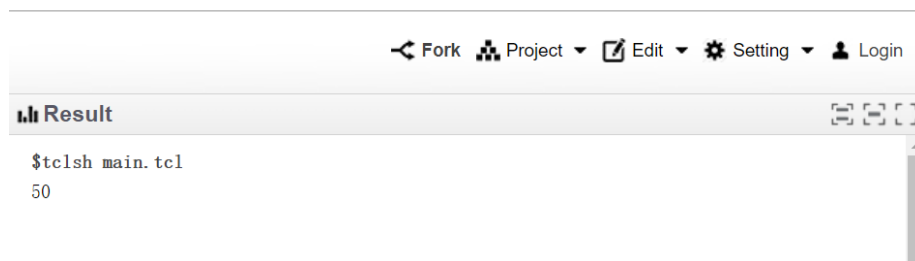
Code:



```
1 proc avg { array } {
2     set sum 0
3     foreach element $array {
4         set sum [expr $sum + $element]
5     }
6     set avg [expr $sum/[llength $array]]
7     return $avg
8 }
9
10 puts [avg {10 20 30 40 50 60 70 80 90}]
```

Figure 1. Code screenshot of problem 1.

Running result:



```
$tclsh main.tcl
50
```

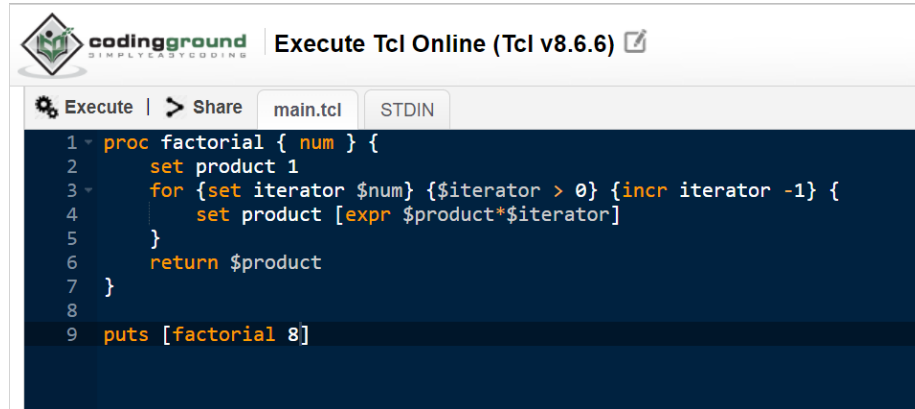
Figure 2. Result screenshot of problem 1.

Here we can see this function gives the output of 50, which is the average of set 10, 20, 30, 40, 50, 60, 70, 80, 90. This TCL script works properly.

## 2.2 Problem 2

Write a procedure (proc) that does the factorial function.

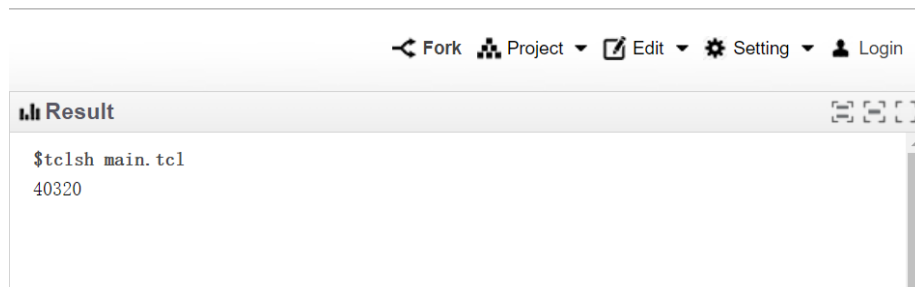
Code:



```
1 ~ proc factorial { num } {  
2     set product 1  
3 ~     for {set iterator $num} {$iterator > 0} {incr iterator -1} {  
4         set product [expr $product*$iterator]  
5     }  
6     return $product  
7 }  
8  
9 puts [factorial 8]
```

Figure 3. Code screenshot of problem 2.

Running result:



```
$tclsh main.tcl  
40320
```

Figure 4. Result screenshot of problem 2.

This function gives the output of 40320, which is the factorial product of  $8*7*6*5*4*3*2*1 = 40320$ . This TCL script works properly.

## 3 Setting up the OpenLane flow

### 3.1 Displaying the Final Layout(3.4 in lab manual)

Here we run the design of "spm" through the flow and obtain the results in corresponding "magic" and "klayout" folder. We use "Klayout" to see these generated layout file.

For "spm.gdc" in "magic" folder:

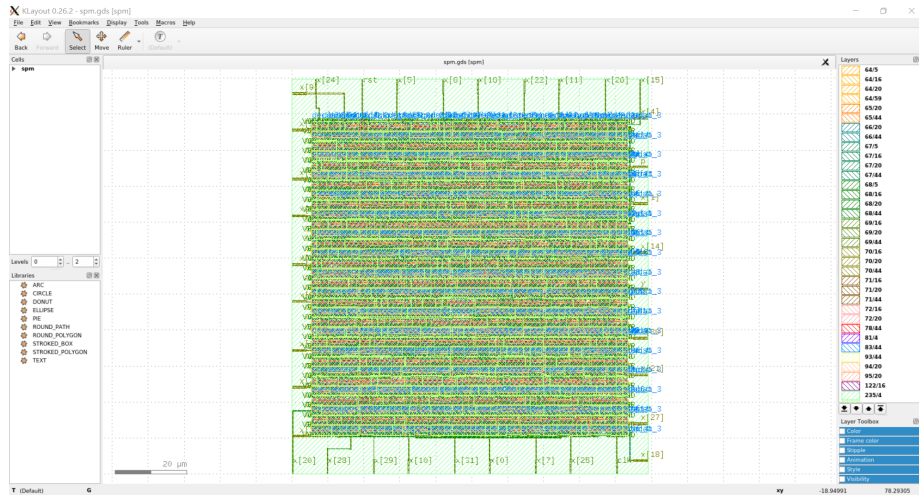


Figure 5. Generated layout in Klayout editor.

For "spm.gdc" in "Klayout" folder:

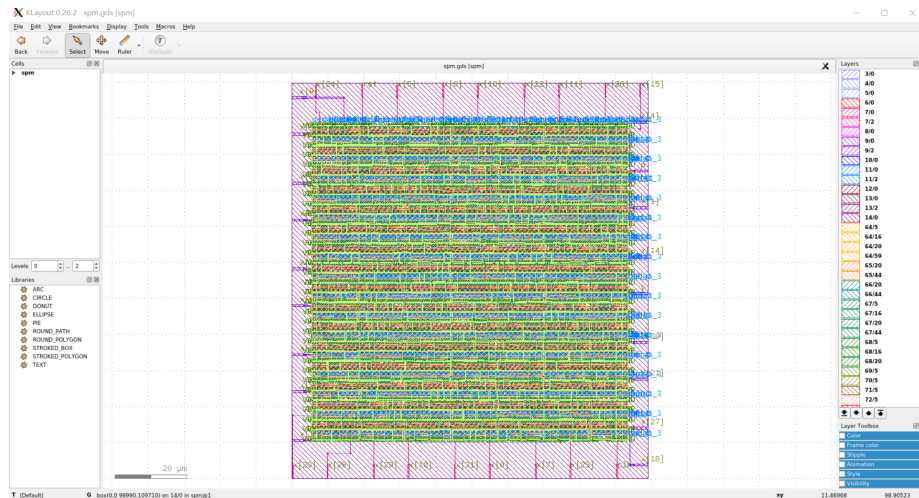


Figure 6. Generated layout in Klayout editor.

### 3.2 Adding a Design(3.5 in lab manual)

In this section, Verilog files provided on Canvas are added into the design in OpenLane and

Exercises about timing after synthesis and route:

1. Please modify the sdc file and create a case where setup time didn't pass, then use the critical as an example and describe how you can fix the timing for that path. Please upload the full timing path and also your answers for fixing the timing violation. Setup time failure:

anchorwu@LAPTOP-GEBO5300: ~/OpenLane/designs/gcd\_test/runs/RUN\_2021.11.21\_13.26.04/reports/routing

6	0.09	0.10	0.29	10.12	v	repeater390/X (sky130_fd_sc_hd_buf_12)
						net390 (net)
		0.10	0.00	10.13	v	split850/A (sky130_fd_sc_hd_buf_12)
		0.05	0.25	10.37	v	split850/X (sky130_fd_sc_hd_buf_12)
2	0.02					net1247 (net)
		0.05	0.00	10.37	v	rebuffer12/A (sky130_fd_sc_hd_buf_8)
		0.13	0.29	10.66	v	rebuffer12/X (sky130_fd_sc_hd_buf_8)
8	0.09					net1269 (net)
		0.13	0.01	10.67	v	_4034_/S (sky130_fd_sc_hd_mux2_1)
		0.11	0.74	11.41	v	_4034_/X (sky130_fd_sc_hd_mux2_1)
1	0.00					_0198_ (net)
		0.11	0.00	11.41	v	_4035_/A0 (sky130_fd_sc_hd_mux2_1)
		0.13	0.70	12.11	v	_4035_/X (sky130_fd_sc_hd_mux2_1)
1	0.00					GCDdpath0.A_next[96] (net)
		0.13	0.00	12.11	v	_2734_/A0 (sky130_fd_sc_hd_mux2_1)
		0.12	0.70	12.81	v	_2734_/X (sky130_fd_sc_hd_mux2_1)
1	0.00					_1489_ (net)
		0.12	0.00	12.81	v	_2735_/A (sky130_fd_sc_hd_clkbuf_1)
		0.05	0.19	13.00	v	_2735_/X (sky130_fd_sc_hd_clkbuf_1)
1	0.00					_0724_ (net)
		0.05	0.00	13.00	v	_4308_/D (sky130_fd_sc_hd_dfrtp_4)
				13.00		data arrival time
			5.50	5.50		clock clk (rise edge)
			0.00	5.50		clock source latency
		0.63	0.42	5.92		clk (in)
1	0.05					clk (net)
		0.63	0.00	5.92		clkbuf_0_clk/A (sky130_fd_sc_hd_clkbuf_16)
		0.08	0.44	6.36		clkbuf_0_clk/X (sky130_fd_sc_hd_clkbuf_16)
2	0.02					clknet_0_clk (net)
		0.08	0.00	6.36		clkbuf_1_1_0_clk/A (sky130_fd_sc_hd_clkbuf_2)
		0.25	0.31	6.66		clkbuf_1_1_0_clk/X (sky130_fd_sc_hd_clkbuf_2)
2	0.03					clknet_1_1_0_clk (net)
		0.25	0.00	6.67		clkbuf_2_3_0_clk/A (sky130_fd_sc_hd_clkbuf_2)
		1.12	0.95	7.62		clkbuf_2_3_0_clk/X (sky130_fd_sc_hd_clkbuf_2)
9	0.14					clknet_2_3_0_clk (net)
		1.12	0.01	7.63		clkbuf_leaf_15_clk/A (sky130_fd_sc_hd_clkbuf_16)
		0.10	0.58	8.20		clkbuf_leaf_15_clk/X (sky130_fd_sc_hd_clkbuf_16)
9	0.03					clknet_leaf_15_clk (net)
		0.10	0.00	8.21		_4308_/CLK (sky130_fd_sc_hd_dfrtp_4)
			-0.25	7.96		clock uncertainty
			0.09	8.05		clock reconvergence pessimism
			-0.26	7.79		library setup time
				7.79		data required time
				7.79		data required time
				-13.00		data arrival time
				-5.21		slack (VIOLATED)

Figure 7. Failure timing path.

Ways to fix timing violation:

The most simple way to fix setup timing violation is to reduce the clock frequency(increase clock period), but this way also deteriorates the performance as well. Other ways including improving the design to add some buffers.

2. Please modify the sdc file and find the maximum frequency that passes the timing check after route. Please upload the screenshot of the layout after place and route, and report area and power.

When the clock period is set to 5.5ns, the design can still go through all

flows and can pass the timing check after route.  
Layout:

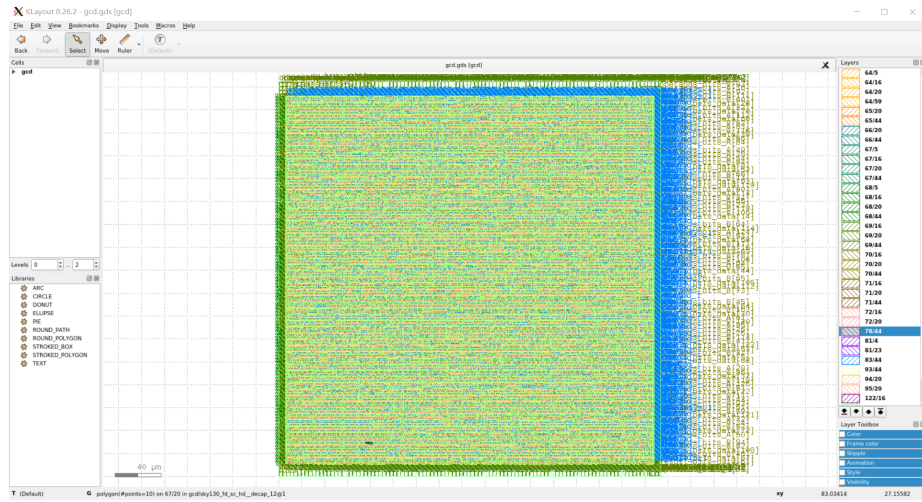


Figure 8. Layout of gcd design.

Area and power:

```
ubuntu@AP106-08063002: /OpenLane/designs/gcd_test/runs/RCN_2021.11.21_08.24.15/reports/routing$ cat 23-spf_extraction_multi_corner_sta.area.rpt
report_design_area
=====
Design area 100735 u^2 97% utilization.
```

Figure 9. Area report of gcd design.

```
anchorwu@LAPTOP-GE80S300: /OpenLane/designs/gcd_test/runs/RUN_2021.11.21.08.24.15/reports/routing$ cat 23-spef_extraction_multi_corner_sta.power.rpt
```

```
report power
```

```
===== Slowest Corner =====
```

Group	Internal Power	Switching Power	Leakage Power	Total Power
Sequential	8.90e-04	2.73e-04	3.98e-06	1.17e-03
Combinational	1.64e-03	1.43e-03	1.50e-05	3.08e-03
Macro	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Pad	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Total	2.53e-03	1.70e-03	1.90e-05	4.24e-03
	59.5%	40.0%	0.4%	100.0%

```
===== Typical Corner =====
```

Group	Internal Power	Switching Power	Leakage Power	Total Power
Sequential	1.15e-03	3.48e-04	2.09e-09	1.50e-03
Combinational	2.04e-03	1.82e-03	3.38e-08	3.86e-03
Macro	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Pad	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Total	3.19e-03	2.17e-03	3.59e-08	5.36e-03
	59.5%	40.5%	0.0%	100.0%

```
===== Fastest Corner =====
```

Group	Internal Power	Switching Power	Leakage Power	Total Power
Sequential	1.34e-03	4.12e-04	4.55e-09	1.75e-03
Combinational	2.37e-03	2.17e-03	4.33e-08	4.54e-03
Macro	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Pad	0.00e+00	0.00e+00	0.00e+00	0.00e+00
Total	3.71e-03	2.58e-03	4.79e-08	6.29e-03
	59.0%	41.0%	0.0%	100.0%

Figure 10. Power report of gcd design.

The area is  $100735\mu m^2$  with 97% utilization. The total power is about  $4.24 \times 10^{-3}$  w at the slowest corner, with around 59% internal power and around 40% switching power.

3. Please add 10% of rise and fall time in your sdc for the clock and check the results again. Find the maximum frequency that passes the timing check. Any differences from the first experiment?

On previous conditions, clock period of 5.5 ns can pass through all flows. After adding 10% of rise and fall time, 5.5ns cannot pass through all the flows and clock period of 6.0ns can pass through flows. Adding rise and fall time indeed make the timing condition harder to fulfill the requirements.

Exercises about modifying parameters:  
(Unless specified, all tests below are with 5.5ns clock period.)

1. Run a test where you set the core utilization target to 55%, what did you observe? Is the flow running ok? If not, why? If so, what is the total area? How about timing?

In this section, I set the core utilization target to 55% in file "sky130A\_sky130\_fd\_sc\_hd.config.tcl", instead of original value of 25%. I find that after this modification, the design cannot pass through the flow, because the area for floorplan is too small and there are not enough pins(as shown in figure below).



```

[ERROR PPL-0072] Number of pins (390) exceed number of valid positions (284).
Error: ioplacer.tcl: 49 PPL-0072
[ERROR] During executing, "openroad -exit /openlane/scripts/openroad/ioplacer.tcl & tee /dev/null >/openlane/designs/gcd_test/runs/RUN_2021.11.21.13.04.05/logs/floorplan/4-ioplacer.log"
[ERROR] Exit code: 1
[ERROR] test 10 lines:
child process exited abnormally
[ERROR]: Please check openroad_log file
[ERROR]: Dumping to /openlane/designs/gcd_test/runs/RUN_2021.11.21.13.04.05/error.log
[INFO]: Calculating Runtime from the Start...
[INFO]: flow failed for gcd/2021.11.21.13.04.05 in 0h0m17s
[INFO]: Generating Final Summary Report...
[INFO]: Design Name: gcd
Run Directory: /openlane/designs/gcd_test/runs/RUN_2021.11.21.13.04.05
Source not found.

LVS Summary:
Source: /openlane/designs/gcd_test/runs/RUN_2021.11.21.13.04.05/results/lvs/gcd.lvs_parsed.gds.log
Source not found.

Antenna Summary:
No antenna report found.
[INFO]: check full report here: /openlane/designs/gcd_test/runs/RUN_2021.11.21.13.04.05/reports/final_summary_report.csv
[INFO]: Saving Runtime Environment
[ERROR]: Flow Failure

```

Figure 11. Flow failed when core utilization is 55%.

In terms of area, the die area is as the figure shown below, which is:(0, 0, 228.745, 239.465). In terms of timing, the design can still pass the 1st step of synthesis, but with several slack violations.

```

root@LAPTOP-CEBOS300:/home/anchorwu/OpenLane/designs/gcd_test/runs/RUN_2021.11.21.13.04.05/reports/floorplan# cat 3-init_floorplan.die_area.rpt
0.0 0.0 228.745 239.465

```

Figure 12. Die area when core utilization is 55%.

2. Turn on the placement timing driven and compare against a base case, any changes in timing? Please describe the timing differences.

By setting "set ::env(PL\_TIME\_DRIVEN) 1" in "OpenLane/configuration/placement.tcl", the timing behavior becomes worse than previous cases. That's because timing driven placement will run placement mainly based on timing instead of routing.

3. Try to run synthesis at a different corner (by changing LIB\_SYNTH to a use the same lib defined by LIB\_SLOWEST. Observe if the maximum frequency and describe what you saw. How about area and power?

In this section, I change the "LIB\_SYNTH" into "LIB\_SLOWEST" in "OpenLane/scripts/synth.tcl" and "OpenLane/scripts/synth\_top.tcl" and "OpenLane/pdks/open\_pdks/sky130/openlane/sky130\_fd\_sc\_hd/config.tcl". Then we find that this design cannot pass the flow, due to LVS errors.

```

Circuit 1 contains 4037 devices, Circuit 2 contains 4037 devices.
Circuit 1 contains 4403 nets, Circuit 2 contains 4206 nets. *** MISMATCH ***

Result: Netlists do not match.
Logging to file "/openlane/designs/gcd_test/runs/RUN_2021.11.21_14.35.08/results/lvs/gcd.lvs.lef.log" disabled
LVS Done.
LVS reports:
  net count difference = 197
  device count difference = 0
  unmatched nets = 63
  unmatched devices = 13
  unmatched pins = 0
  property failures = 0

Total errors = 273
[ERROR]: There are LVS errors in the design according to Netgen LVS.
[INFO]: Calculating Runtime From the Start...
[INFO]: flow failed for gcd/2021.11.21_14.35.08 in 0h8m46s
[INFO]: Generating Final Summary Report...
[INFO]: Design Name: gcd
Run Directory: /openlane/designs/gcd_test/runs/RUN_2021.11.21_14.35.08
Source not found.

-----
LVS Summary:
Source: /openlane/designs/gcd_test/runs/RUN_2021.11.21_14.35.08/results/lvs/gcd.lvs_parsed.lef.log
  net count difference = 197
  unmatched nets = 63
Total errors = 273

-----
Antenna Summary:
No antenna report found.
[INFO]: check full report here: /openlane/designs/gcd_test/runs/RUN_2021.11.21_14.35.08/reports/final_summary_report.csv
[INFO]: Saving Runtime Environment
[ERROR]: Flow Failed.

```

Figure 13. Flow failure after changing corners.

4. Change CTS\_TARGET\_SKEW to 150ps (instead of 200ps), observe the timing differences, and describe your observation. Can you explain why?

As explained later, since parameter "CTS\_TARGET\_SKEW" isn't used in TritonCTS 2.0 in new version of OpenLane, and there won't be any change after varying this parameter, we just skip this exercise.

5. Change the core aspect ratio to 2:1 instead of 1:1, what did you observe, any changes about timing, area and power? Please upload the layout after changing the aspect ratio and your answers to the above question

We modify the aspect ratio in "OpenLane/configuration/floorplan.tcl" from 1:1 to 2:1. After running through the flow, the layout becomes:

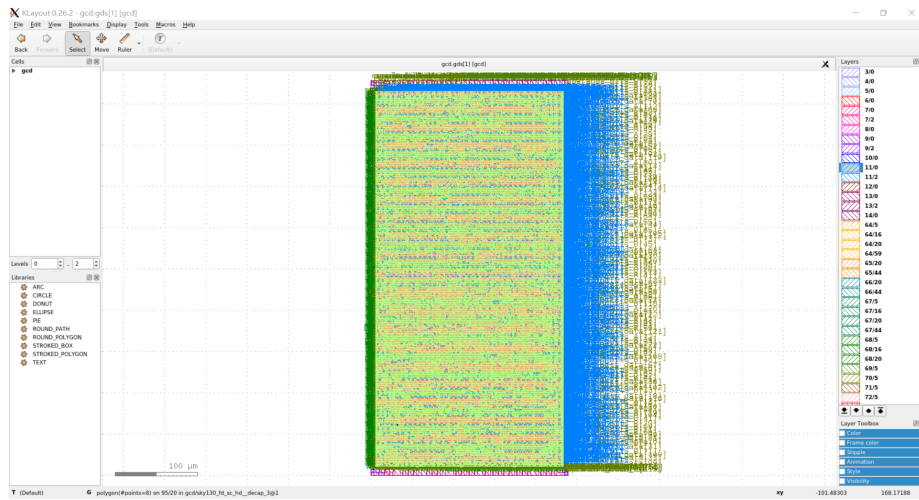


Figure 14. Layout when aspect ratio is 2:1.

Thus, we can find that the chip area becomes a rectangular with length is two times of width. We can also find this by the area report, which becomes (0, 0, 239.37, 478.42):

```
root@LAPTOP-GE80S300: /home/anchorwu/OpenLane/designs/gcd_test/runs/RUN_2021.11.21.13.26.04/reports/floorplan# cat 3-init_floorplan.die_area.rpt
0.0 0.0 239.37 478.42
```

Figure 15. Die area when aspect ratio is 2:1.

## 4 Questions

1. How many corners are in the Skywater 130nm technology libraries? I find 18 ".lib" files under the folder "lib" of "sky130\_fd\_sc\_hd" in "sky130A" PDK, corresponding to 18 corners of fast-fast or slow-slow or typical-typical and also for different temperatures and threshold voltages. There are also going to be other variations of technology, such as "sky130\_fd\_sc\_hvl" etc.

```
root@LAPTOP-GE80S300: /home/anchorwu/OpenLane/pdks/sky130A/libs.ref/sky130_fd_sc_hd/lib# ls
sky130_fd_sc_hd_ff_100C_lv65.lib  sky130_fd_sc_hd_ff_n40C_lv95.lib  sky130_fd_sc_hd_ss_n40C_lv35.lib  sky130_fd_sc_hd_ss_n40C_lv76.lib
sky130_fd_sc_hd_ff_100C_lv85.lib  sky130_fd_sc_hd_ff_n40C_lv95_ccnoise.lib  sky130_fd_sc_hd_ss_n40C_lv40.lib  sky130_fd_sc_hd_tt_025C_lv80.lib
sky130_fd_sc_hd_ff_n40C_lv65.lib  sky130_fd_sc_hd_ss_100C_lv40.lib  sky130_fd_sc_hd_ss_n40C_lv44.lib  sky130_fd_sc_hd_tt_100C_lv80.lib
sky130_fd_sc_hd_ff_n40C_lv65.lib  sky130_fd_sc_hd_ss_100C_lv60.lib  sky130_fd_sc_hd_ss_n40C_lv60.lib
sky130_fd_sc_hd_ff_n40C_lv76.lib  sky130_fd_sc_hd_ss_n40C_lv28.lib  sky130_fd_sc_hd_ss_n40C_lv60_ccnoise.lib
```

Figure 16. Corner files in "sky130\_fd\_sc\_hd".

2. In openLane, what layers are power and ground using for the power grid?

3. Is the placement and route time better or worse than synthesis timing? Why is that?

The placement and route time is worse or stricter than synthesis timing, since when some my designs can indeed pass the synthesis, they can't pass the timing check after placement and route. That's probably because even though

the design can pass synthesis, there's going to be new setup/hold violations after placement and routing. If these violations can't be fixed with margin, then flow failure occurs.

4. Why do we need to use a special buffer for clock tree? Example: in OpenLane, they used `sky130_fd_sc_hd__clkbuf_4`.

Sometimes after synthesis there could still be some timing violations, especially after placement and routing, thus some special buffers should be inserted to fix these violations.

5. Describe how the OpenLane project is constructed into a flow based on your observations and understanding.

OpenLane project consists of a lot of TCL scripts that run different tools. By properly configuring these scripts in a chain that processes the design with the appropriate sequential order, this flow can be generated and works.