Physical Synthesis

ECE 481/581: Final Project

Winter 2022

Leo Garcia: lwg@pdx.edu

1. Introduction:

This project is about physical synthesis and to get familiar with synopsys DC and IC tools. In the first half of the project, is to run the rtl and it's constraints file in DC_shell and icc2_shell; Comparing the report timing between logical and physical synthesis.

2. Main Objectives or outcomes of the Project:

Fix the broken config files provided by the instructor. Compare the worst slack from logical to physical synthesis. Fix the macros and voltage domain.

Task 1a

- **For fifo1_sram, find the worst timing path in the design for INPUTS, OUTPUTS, rclk, wclk in the logical synthesis.**

I used the report_path_group to get the groups then I did report_timing for those respective groups. I then found the slack for the highest one in the logical synthesis and looked for the same path in the physical synthesis.

group	command	Slack logical	Cap logical	Slack physical	Cap physical	Area phys	Area logical
inputs	report_timing -from wdata_in[0] -through io_r_wdata_in_0_/DOUT -to wdata_reg_0_/D -input -capacitance -transition	-0.07	0.41	-0.08	6.56	371815.213893	370819.985982
outputs	report_timing -from fifomem/genblk1_2__U/CE	-0.01	1433.81	-0.23	1433.81		

	2 -through io_l_rdata_0_/PADIO -to rdata[0] -capacitance -input -transition						
rclk	report_timing -from rptr_empty/rbin_reg_3_/CLK -through rptr_empty/U95/Y -to rptr_empty/rempty_reg/D -capacitance -input -transition	0.05	0.61	-0.02	13.70		
wclk	report_timing -from wptr_full/wfull_reg/CLK -through wptr_full/U13/Y -to wptr_full/wfull_reg/D -capacitance -input -transition	0.05	0.59	-0.10	14.22		

Include the two timing reports side by side in your report.

It's very long. I attached a concatenated version of the timing reports for all groups, these are technically side by side.

logical	physical
https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/fifo1_sram.dc.max.all.rpt	https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/fifo1_sram.dct.max.all.rpt

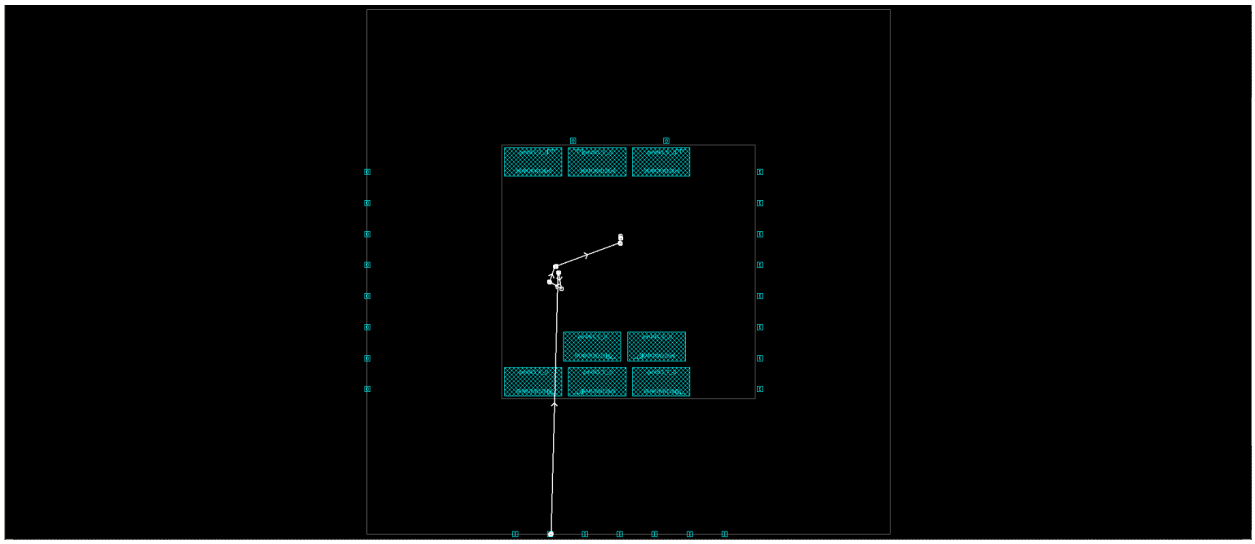
- **Include your textual analysis of the difference. Difference in delays, slack, drive strengths used.**

  The logical slack is better for all cases. The physical slack is always worse. This is to be expected. As for capacitance the physical capacitance is generally larger. This is also to be expected. The reasons are because “parasitic effects due to the proximity of certain devices cause the “physical” version of the circuit to behave differently than the “logical” version of the circuit.” from synopsys.com/glossary/what-is-physical-synthesis.html

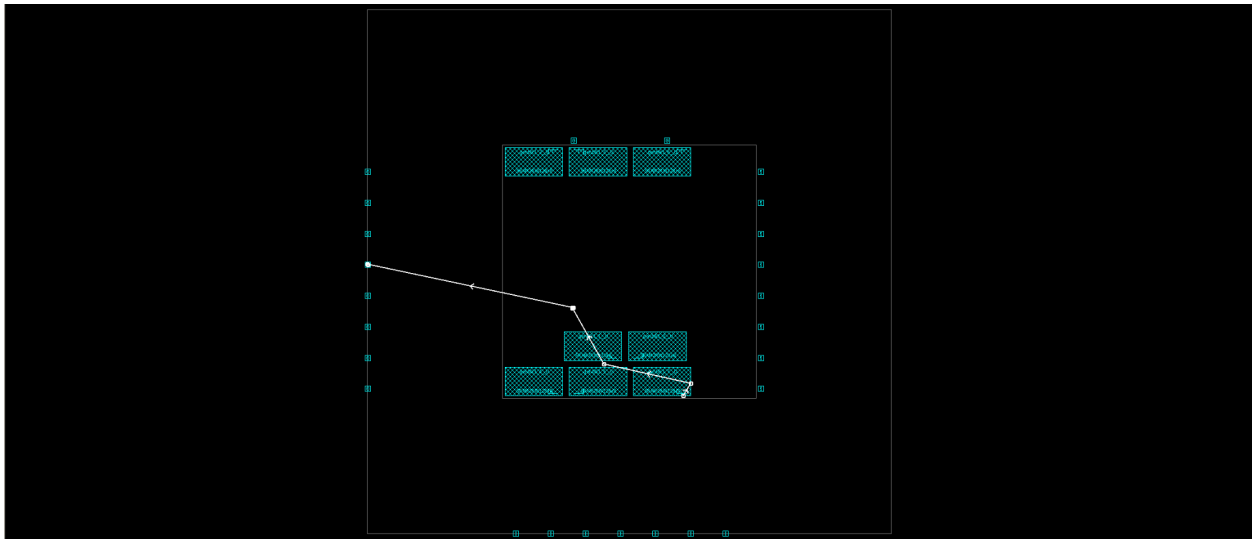
  The physical cell area is larger than the logical cell area this is to be expected.

- **Include a graphical picture highlighting the timing path physically.**

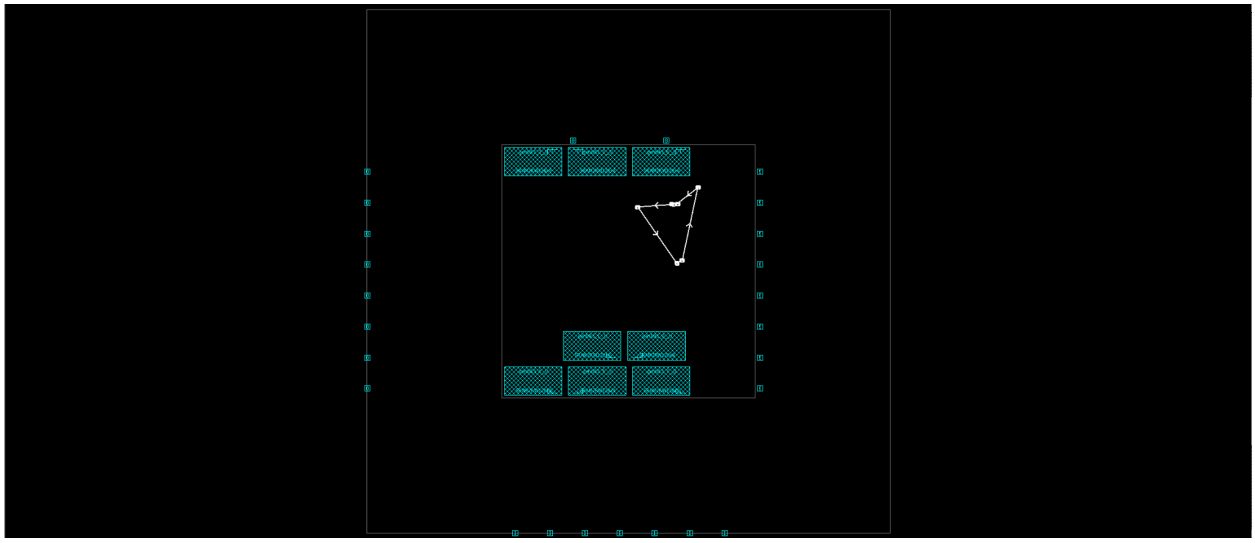
Inputs



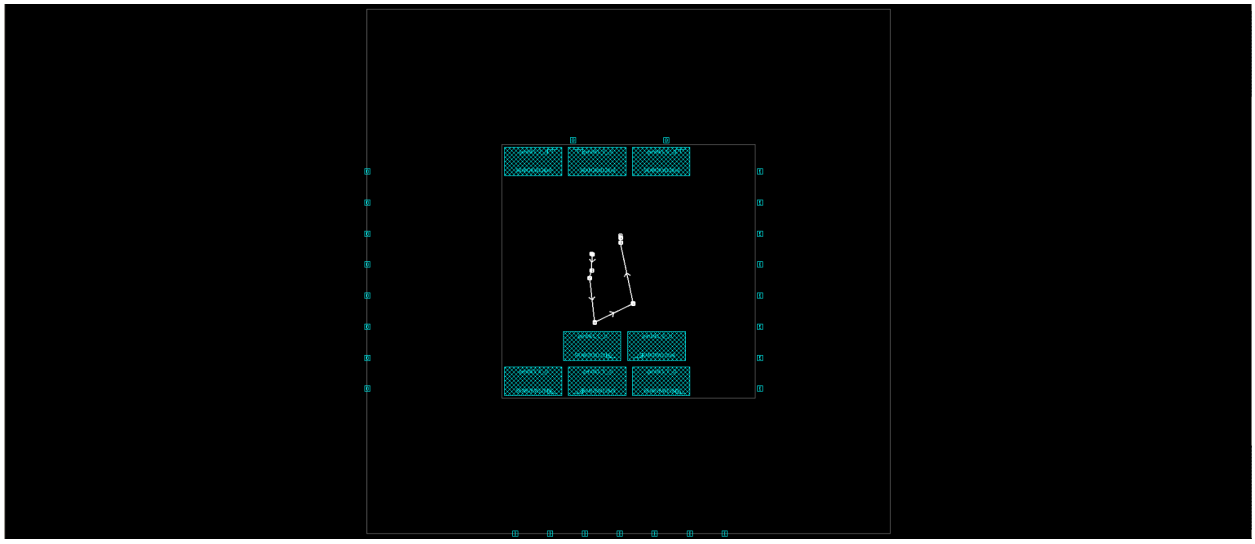
outputs



rclk



wclk



Task_1B:

Finding Worst Paths in Physical and comparing to Logical. Synopsys and/or Cadence.

- **For fifo_sram, find the worst timing path in the design for INPUTS, OUTPUTS, rclk, wclk in the physical synthesis.In parallel, report_timing the exact same startpoint/endpoint in logical synthesis**

group	command	Slack physical	Cap physical	Slack logical	Cap logical	Area phys	Area logical
inputs	report_timing -from winc -through io_b_winc/DOUT -to wptr_full/wfull_reg/D -capacitance -input -transition	-0.18	66.11	-0.04	3.78	371815.213893	370819.985982
outputs	report_timing -from fifomem/genblk1_7__U/CE2 -through fifomem/genblk1_7__U/O2[4] -to rdata[4] -capacitance -input -transition	-0.29	18.34	0.00	0.59		
rclk	report_timing -from rptr_empty/rbin_reg_3_/CLK -through rptr_empty/rbin_reg_3_/Q -to rptr_empty/rempty_reg/D -capacitance -input -transition	-0.06	1.81	0.05	2.25		
wclk	report_timing -from wptr_full/wbin_reg_0_/CLK -through wptr_full/wbin_reg_0_/Q -to wptr_full/wfull_reg/D -capacitance -input -transition	-0.10	4.13	0.05	1.25		

- **Include the two timing reports side by side in your report.**

physical	logical
https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/fifo1_sram.dct.max.phy.all.rpt	https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/fifo1_sram.dc.max.phy.all.rpt

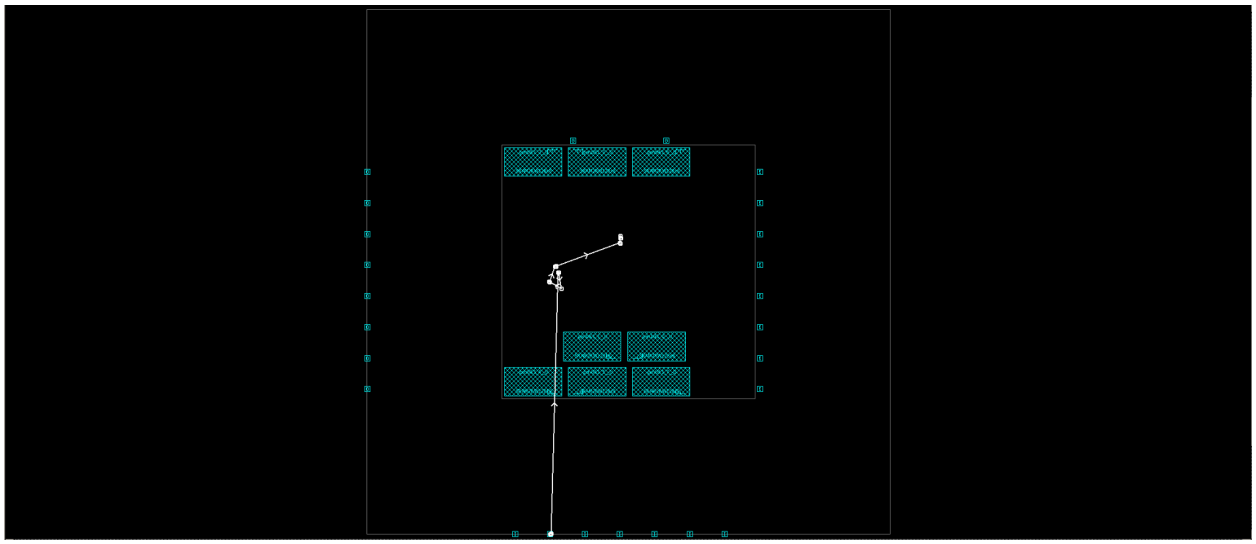
- **Include your textual analysis of the difference. Difference in delays, slack, drive strengths use.**

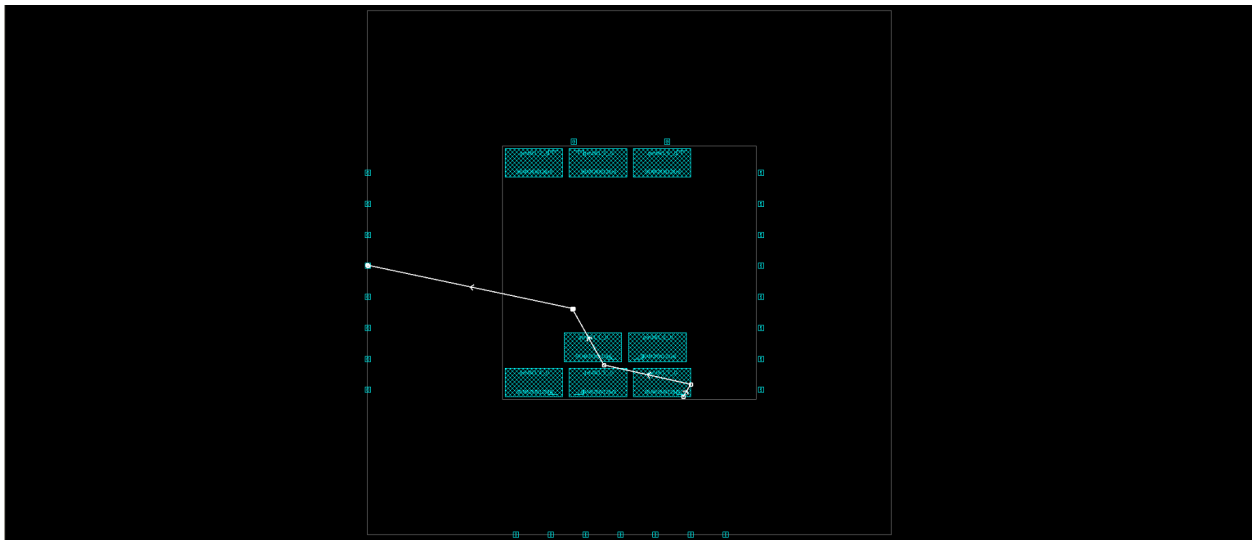
The longest path compared from physical to logical is different. This is expected. Due to the actual physical layout the behavior can be very different.

- **Include a graphical picture highlighting the timing path physically.**

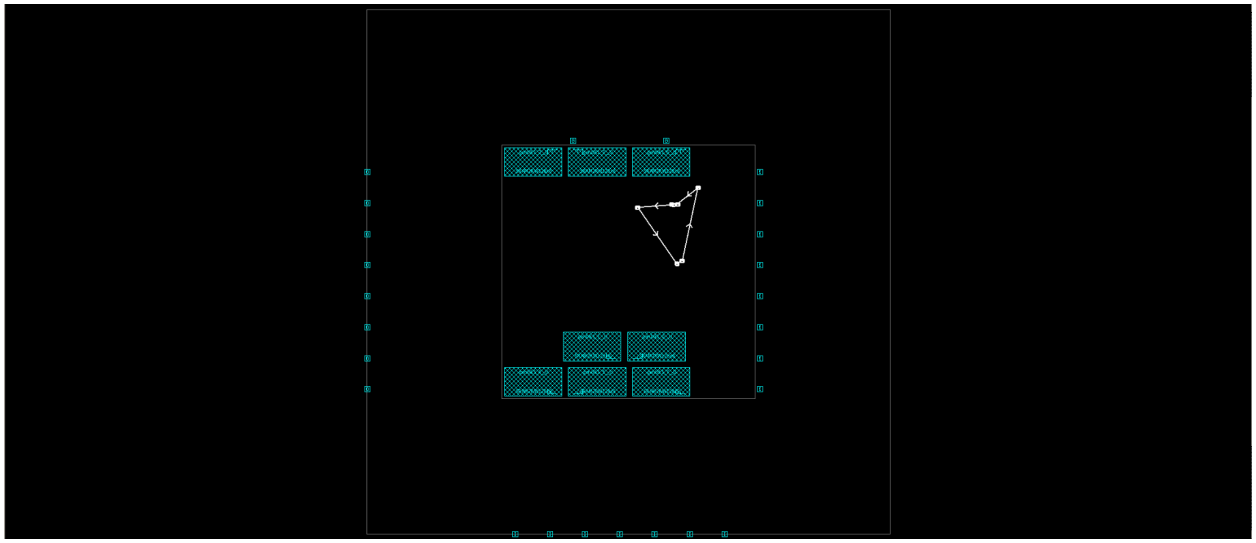
  Inputs

  outputs

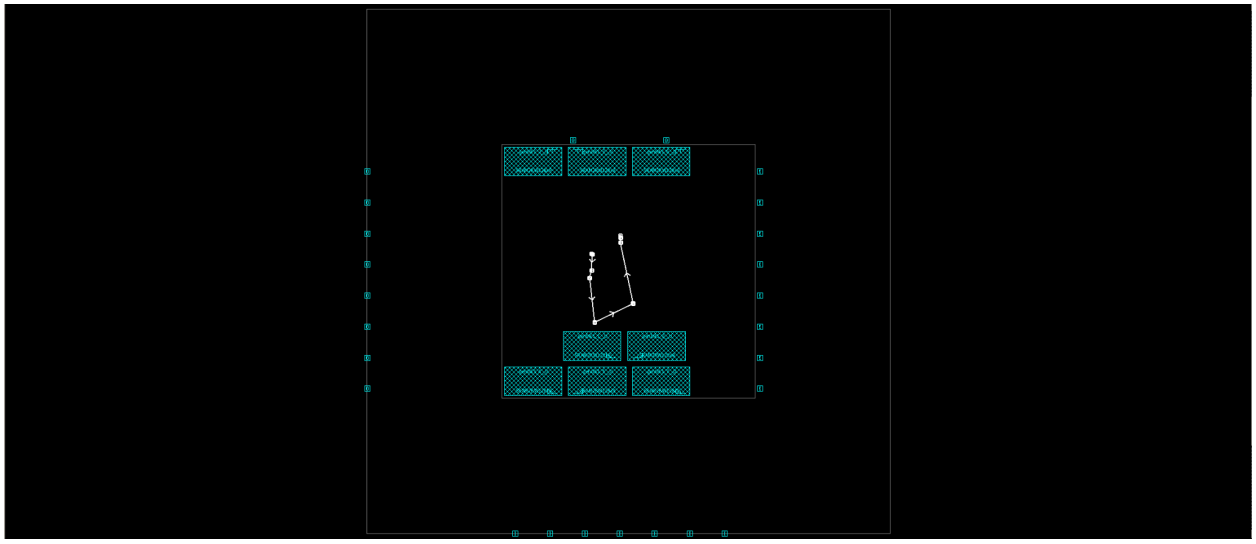




rclk



wclk



Task_2A:

Finding Worst Paths in Logical and comparing to Physical. Synopsys and/or Cadence.

- **For ExampleRocketSystem, find the worst timing path in the design for INPUTS, OUTPUTS, clock in the logical synthesis.**
- **In parallel, report_timing the exact same startpoint/endpoint in physical synthesis.**

group	command	Slack logic	Cap logical	Slack physi	Cap physical	Area phys	Area logic

		al		cal			
inputs	report_timing -from mem_axi4_0_b_bits_id[2] -through sbus/system_bus_xbar/auto_out_2_d_bits_source[4] -to sbus/coupler_to_port_named_mmio_port_axi4/axi4deint/u_T_14_reg_2_/D -capacitance -input -transition	0.18	1.71	0.24	1.55	1541598.929109	1490824.767540
outputs	report_timing -from cbus/buffer/Queue_1/value_1_reg/CLK -through sbus/system_bus_xbar/auto_out_1_d_valid -to mem_axi4_0_b_ready -capacitance -input -transition	1.33	1.66	0.89	6.01		
rclk	report_timing -from tile/fpuOpt/sfma/in_in2_reg_27_/CLK -through tile/fpuOpt/sfma/fma/mulAddRecFNToRaw_preMul/io_b[27] -to tile/fpuOpt/sfma/fma/u_T_5_reg_48_/D -capacitance -input -transition	0.00	0.04	0.09	0.05		

- **Include the two timing reports side by side in your report.**

https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/ExampleRocketSystem.dc.max.rpt	https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/ExampleRocketSystem.dct.max.rpt

- **Include your textual analysis of the difference. Difference in delays, slack, drive strengths used.**

  Something peculiar happened the first command I did was

  report_timing -from mem_axi4_0_b_bits_id[2] -through bh/U115/Y -to sbus/coupler_to_port_named_mmio_port_axi4/axi4deint/u_T_14_reg_2_/D -capacitance -input -transition

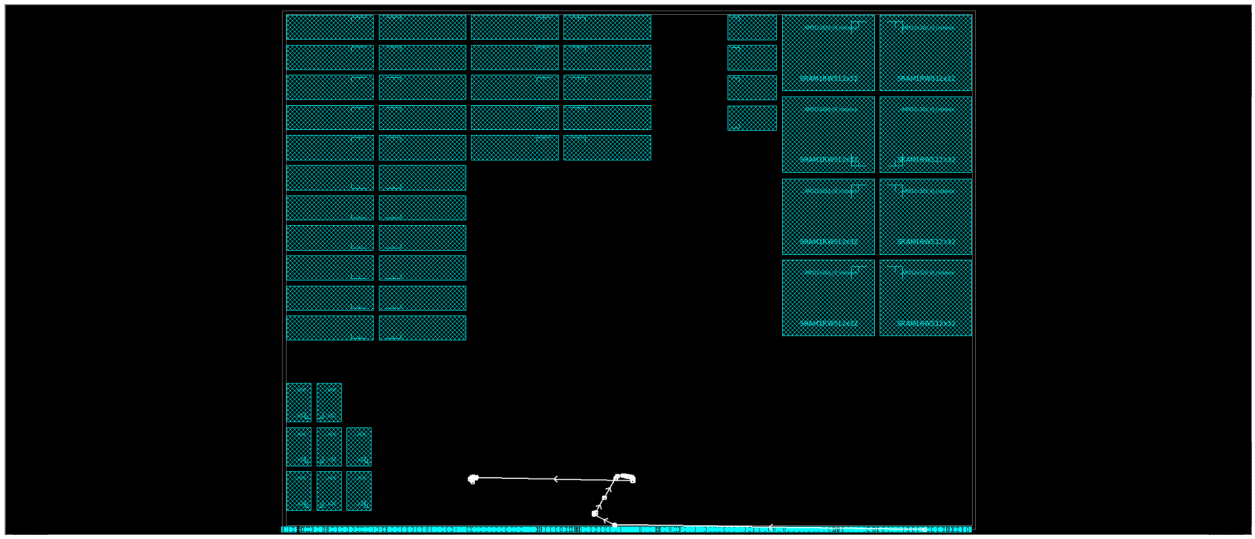
However this path did not show up on the physical side. What happened is that the cell used between logical synthesis and physical is different

It Is surprising that the logical paths are worse than the physical paths. I guess it depends on the particular path. The physical optimization could have found a better way to place this.

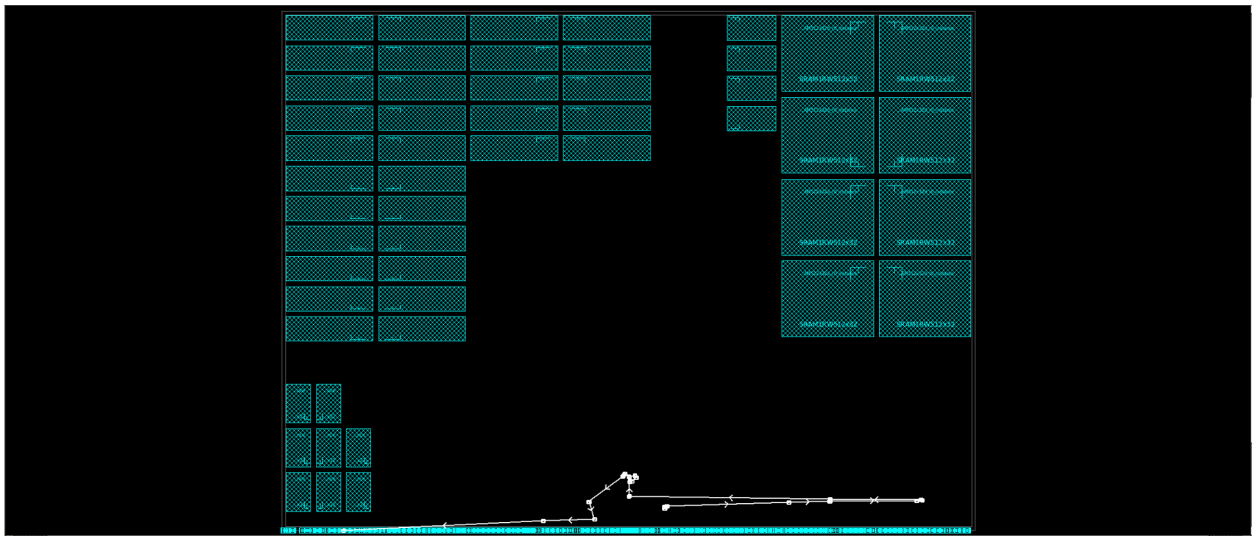
The capacitance is usually better with physical as well. The area is larger in the physical design as well. This is expected.

- **Include a graphical picture highlighting the timing path physically.**

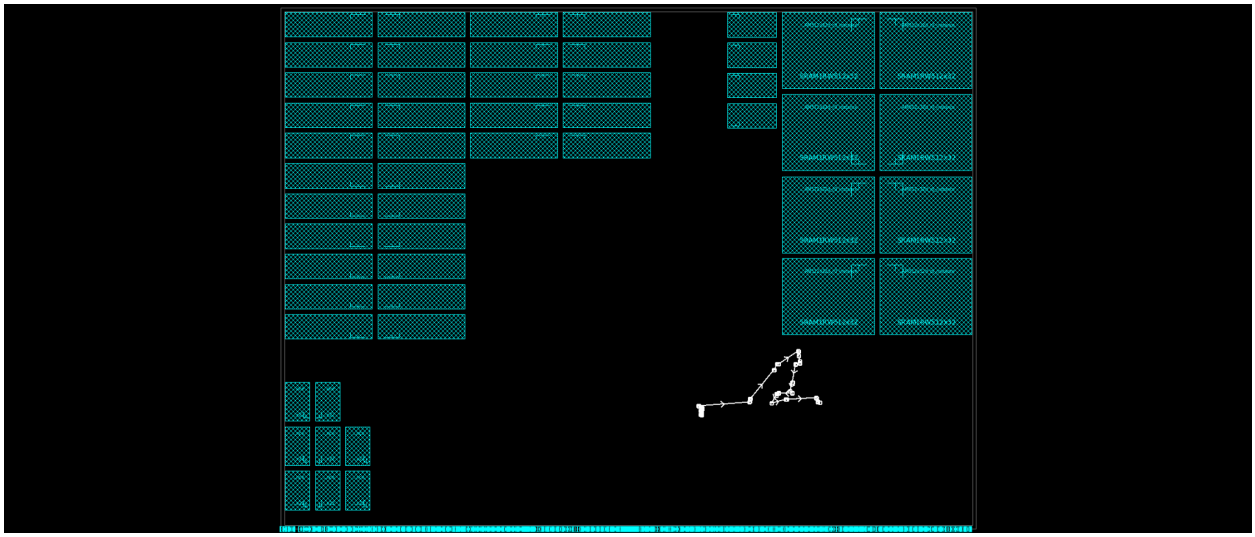
Inputs



outputs



clock



Task_2B:

Finding Worst Paths in Physical and Comparison to Logical. Synopsys and/or Cadence.

- **For ExampleRocketSystem, find the worst timing path in the design for INPUTS, OUTPUTS, rclk, wclk in the physical synthesis.**

group	command	Slack physical	Cap physical	Slack logical	Cap logical
inputs	report_timing -from mem_axi4_0_b_bits_id[3] -through sbus/auto_system_bus_xbar_out_d_bits_source[4] -to sbus/coupler_to_port_named_mmio_port_axi4/axi4deint/u_T_14_reg_2_/D -capacitance -input -transition	0.09	1.55	0.19	1.71
outputs	report_timing -from sbus/coupler_to_port_named_mmio_port_axi4/axi4buf/Queue_2/value_1_reg/CLK -through sbus/system_bus_xbar/auto_out_0_d_bits_source[2] -to mbus/coupler_to_memory_controller_named_axi4/axi4yank/Queue_4/u_T_1_reg/D -capacitance -transition -input	0.14	2.13	1.17	1.10

clock	report_timing -from tile/core/csr/reg_pmp_7_addr_reg_0_/CLK -through tile/ptw/io_requestor_1_pmp_7_mask[29] -to tile/frontend/tlb/sectored_entries_3_data_0_reg_6_/D -capacitance -input -transition > ../reports/ExampleRocketSystem.dc.phy.clock.rpt	-0.03	0.61	1.13	0.47

- **Include the two timing reports side by side in your report.**

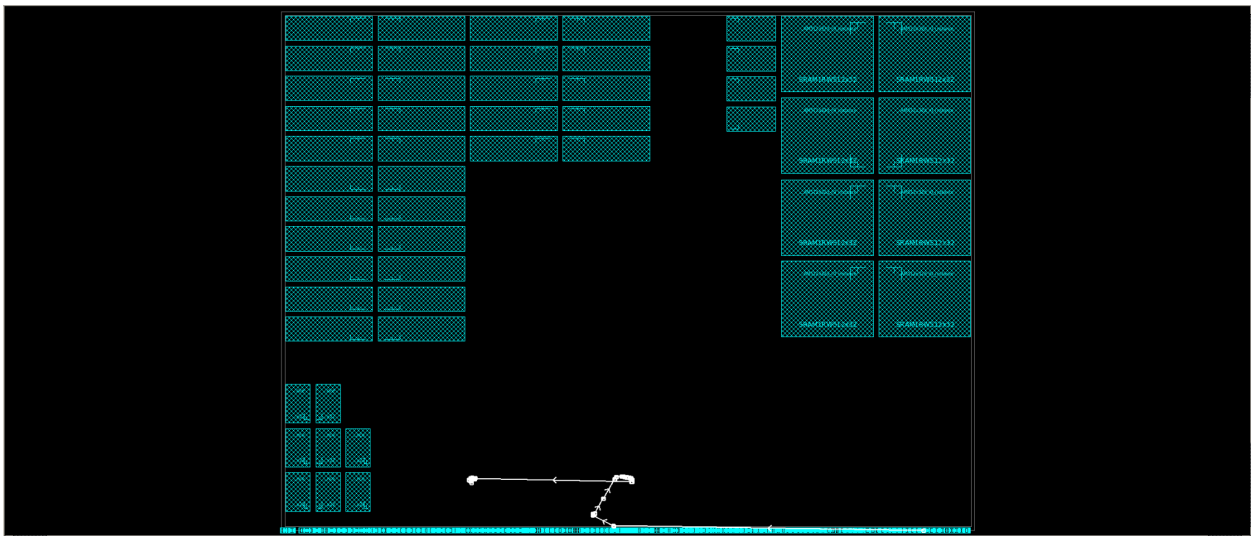
https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/ExampleRocketSystem.dct.phy.max.rpt	https://github.com/DrAtomic/final_project_asic_design/blob/master/syn/reports/ExampleRocketSystem.dc.phy.max.rpt

- **Include your textual analysis of the difference. Difference in delays, slack, drive strengths used**

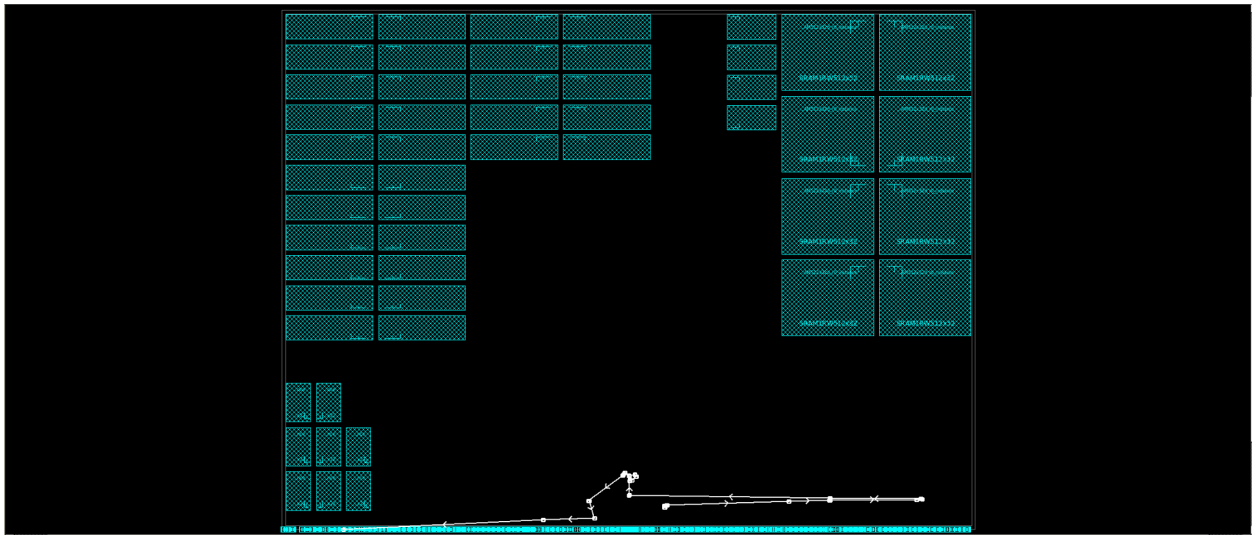
  This is another strange path. This path is worse in logical as well. I suppose that it depends on the physical constraints for the timing and capacitance.

- **Include a graphical picture highlighting the timing path physically.**

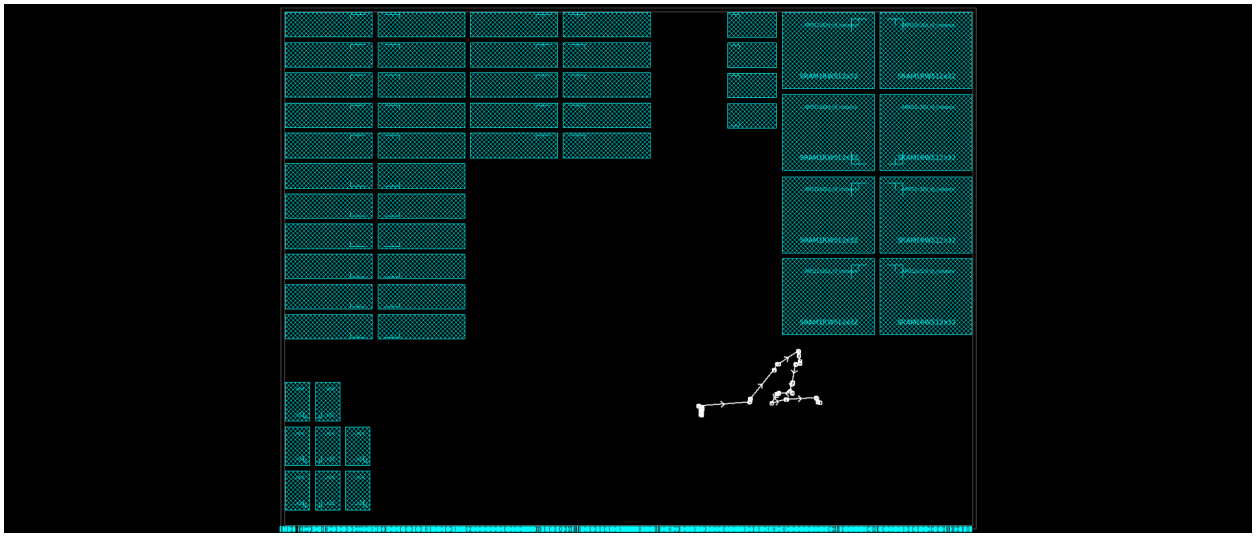
inputs



outputs



clock



Task 3

1. **Git pull and Copy ORCA_TOP files over from the synth_apr_repo template area againGit pull and Copy ORCA_TOP files over from the synth_apr_repo template area again**

   First I rebased the repo to get the updates from the upstream, then I did

   git show --pretty --name-only 72cc48608 -m

   To show the difference in files.

File Name	Description

ORCA_TOP.design_config.tcl	Design configuration
apr/outputs/ORCA_TOP.floorplan.macros.def	Macro placement for dc
constraints/ORCA_TOP.sdc	Constraints file
constraints/ORCA_TOP.upf	Unified power format “UPF is designed to reflect the power intent of a design at a relatively high level”[1]
constraints/ORCA_TOP_func_best.sdc	I think this is the functional design test for best design, notice the ff linked libraries
constraints/ORCA_TOP_func_worst.sdc	functional design test for the worst build. Notice the ss linked libraries
constraints/ORCA_TOP_test_best.sdc	This is probably the timing tests for the best build
constraints/ORCA_TOP_test_worst.sdc	Timing for worst build
syn/rtl/ORCA_TOP.sv.gz	Compressed file

2. Make modifications to design_config for libraries similar to fifo1_sram and ExampleRocket

- **Describe what the modifications are for:**

changes	effect
set lib_dir /pkgs/synopsys/2020/32_28nm/SAED32_EDK/lib	This is the correct lib path
set enable_dft	variable for enable dft
Set synth_corners_slow/fast $slow/fast_corner	these are the corner variables read by other scripts

Set synth_corners_slow/fast $slow/fast_corner, set lib_types "$lib_dir/stdcell_hvt/db_nldm $lib_dir/stdcell_lvt/db_nldm $lib_dir/io_std/db_nldm $lib_dir/sram/db_nldm $lib_dir/pll/db_nldm" set ndm_types "$lib_dir/stdcell_lvt/ndm $lib_dir/stdcell_rvt/ndm $lib_dir/stdcell_hvt/ndm $lib_dir/sram/ndm $lib_dir/io_std/ndm $lib_dir/pll/ndm", These are the paths to the ndm libraries set lib_types_target "$lib_dir/stdcell_rvt/db_nldm" set sub_lib_type "saed32?vt_ saed32sram_ saed32io_wb_ saed32pll_" set sub_lib_type_target "saed32rvt_" set synth_corners_target "ss0p95vn40c ss0p75vn40c" #set synth_corners_target "ss0p95v125c"	sets the variables for the libs
set lef_types [list $lib_dir/stdcell_hvt/lef $lib_dir/stdcell_rvt/lef $lib_dir/stdcell_lvt/lef $lib_dir/sram/lef/ $lib_dir/io_std/lef $lib_dir/pll/lef]	Set the lef libraries
set sub_lef_type "saed32nm_?vt_*.lef saed32_sram_*.lef saed32io_std_wb saed32_PLL.lef"	used for actually linking the library
set mwlib_types [list $lib_dir/stdcell_hvt/milkyway $lib_dir/stdcell_rvt/milkyway $lib_dir/stdcell_lvt/milkyway $lib_dir/io_std/milkyway $lib_dir/sram/milkyway $lib_dir/pll/milkyway]	sets the milkyway libraries

set sub_mwlib_type "saed32nm_?vt_* SRAM32NM saed32io_wb_* SAED32_PLL_FR*" #set lib_types "stdcell_hvt stdcell_rvt stdcell_lvt sram"	

- **What TCL script is using it?**

  I'm assuming *it* is referring to the design_config

  Pretty much every script is using it. dc.tcl dct.tcl genus.tcl and both of the create floor plans scripts.

  A lot of scripting languages (languages that use a run-time environment) have this problem where they make it difficult to use local variables (perl, bash, etc). Tcl is no exception! All of the variables used are global and they have the ability to overwrite each other depending on whatever variable is loaded last. There were some variables that were referenced in scripts that were not defined in the script.

  These variables were set in scripts that were run previously. The reason the variables could be used is because they were loaded into the environment. I regularly found myself writing “echo $<some_variable>” after it was set in the script so I could see what the variable was at that particular instance. I could have done it in the repl (read eval print loop) but as I mentioned before, these variables can be overwritten.

  So the question of “how did these variables get used in the tcl file that were sourced inside dc.tcl” well first they were set in the design_config, then in the dc.tcl they were referenced.

  For example the variable $rtl_list referenced in dc.tcl was set in the design config with “set rtl_list [list ../rtl/$top_design.sv]” that variable is in the run-time environment when dc.tcl asks for it. That's why it is able to be used. I personally don’t like variables being set outside the scope of the script that is calling them but I really can’t think of a better way to do it. “Tcl is the tool we have so you better get used to it”

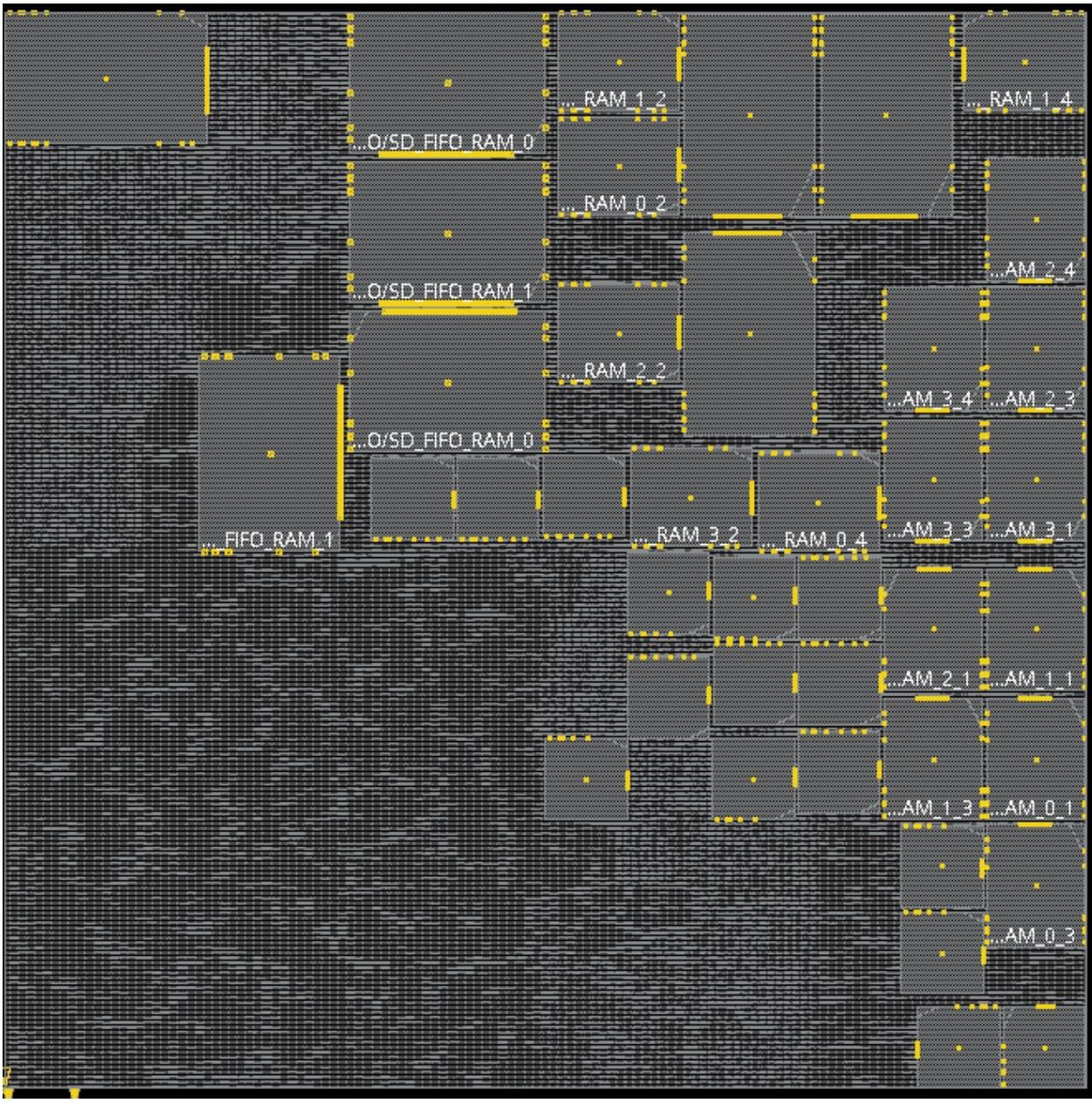
- **What Synopsys/Cadence command uses it and what for?**

  Source -echo -verbose ../../$top_design.design_config.tcl

Anytime the design is loaded is command is run, this is what it sounds like it is the config file for the design it has all the libraries that it links, the lef libraries and the corners used as well as the sub blocks.

3. **Fix the Innovus.macros.def the macros are not all being placed.**

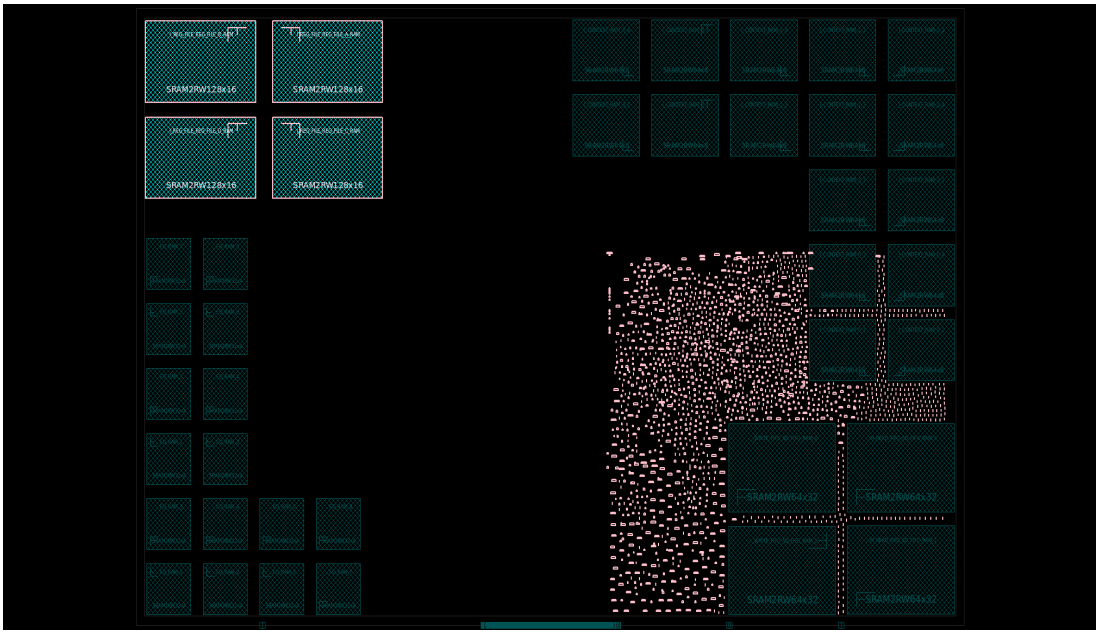
- Explain fixes you had to do.
    - select_obj [get_db insts * -if “.base_cell.name==SRAM*&&.place_status==unplaced”]
        - Selects all the objects that are SRAM and unplaced
    - defOut -selected "../outputs/${top_design}.floorplan.innovus.macros.def"
        - Writes the macros.def
    - placeAIO
        - Automatically places the macros



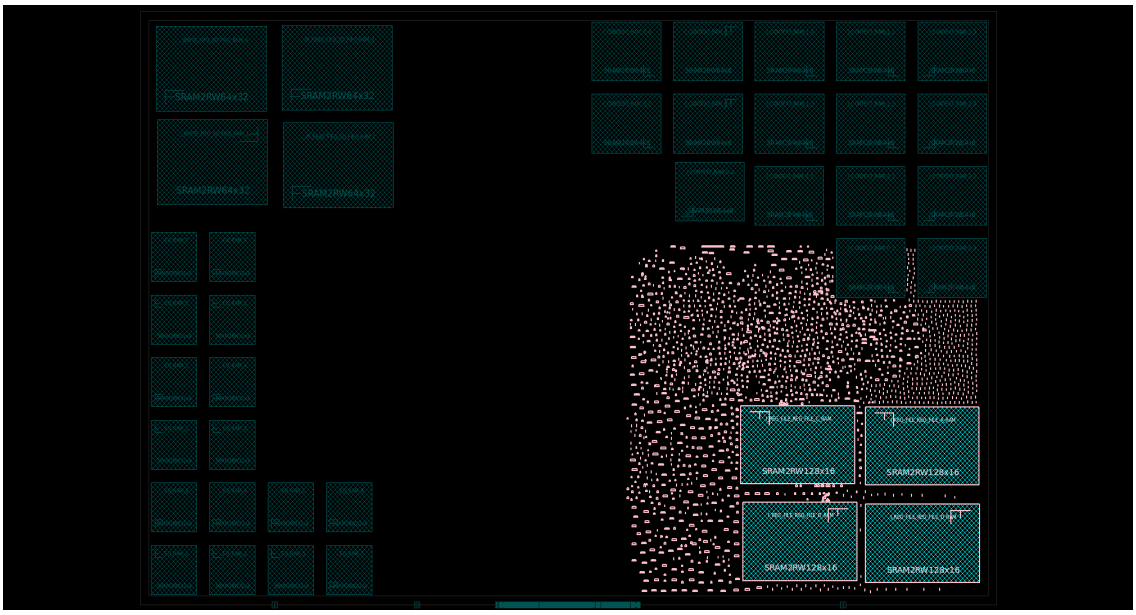
4. **Fix the voltage area for DCT floorplan**

- Are the correct components inside?

No, According to “voltage area buttons” the I_REG_FILE_REG_D_RAM should be swapped with the I_SDRAM_WRITE_FIFO_SD_FIFO_RAM_x and there are a few blocks



To do this I had to physically move the cells and place them in the right spots.



There are still SRAMs that need to be placed out of the region but that can be done by making the voltage domain region smaller.

- What is a voltage domain for?

It is to reduce power consumption. If one domain requires more power than the rest of the chip, it would be impractical to apply that voltage across the whole chip.

General Conclusions or Task4:

Summarize and determine for each of the three designs if logic vs physical synthesis is worse for internal timing, IO timing, and standard cell area.

Based on my data, the physical synthesis was better in timing. However cell area is more while design area being less, with the exception of the fifo_sram

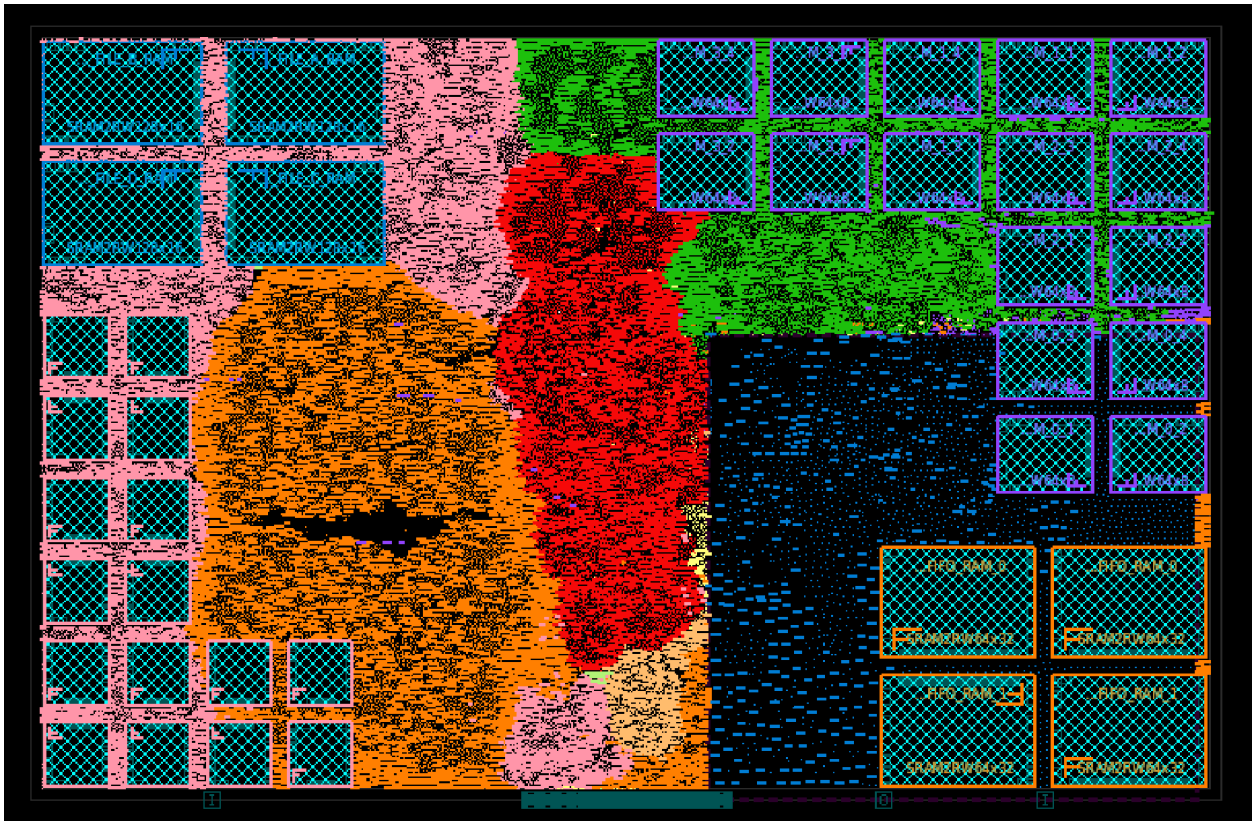
Design	Cell area dc	Cell area dct	Design area dc	Design area dct
fifo_sram	370819.985982	371815.213893	371806.832313	371815.213893
ExampleRocketSystem	1490824.767540	1543354.047520	1888274.919186	1543354.047520
ORCA	379520.115038	394250.809693	458753.721313	394250.809693

Does it match what we learned in class or does it not really confirm our thoughts so well? This could be.lease check and add the Cell Area to your table on logical vs synthesis comparison in addition to WNS/TNS. Is physical synthesis larger or smaller? Does it always track? Which way would you expect and why?

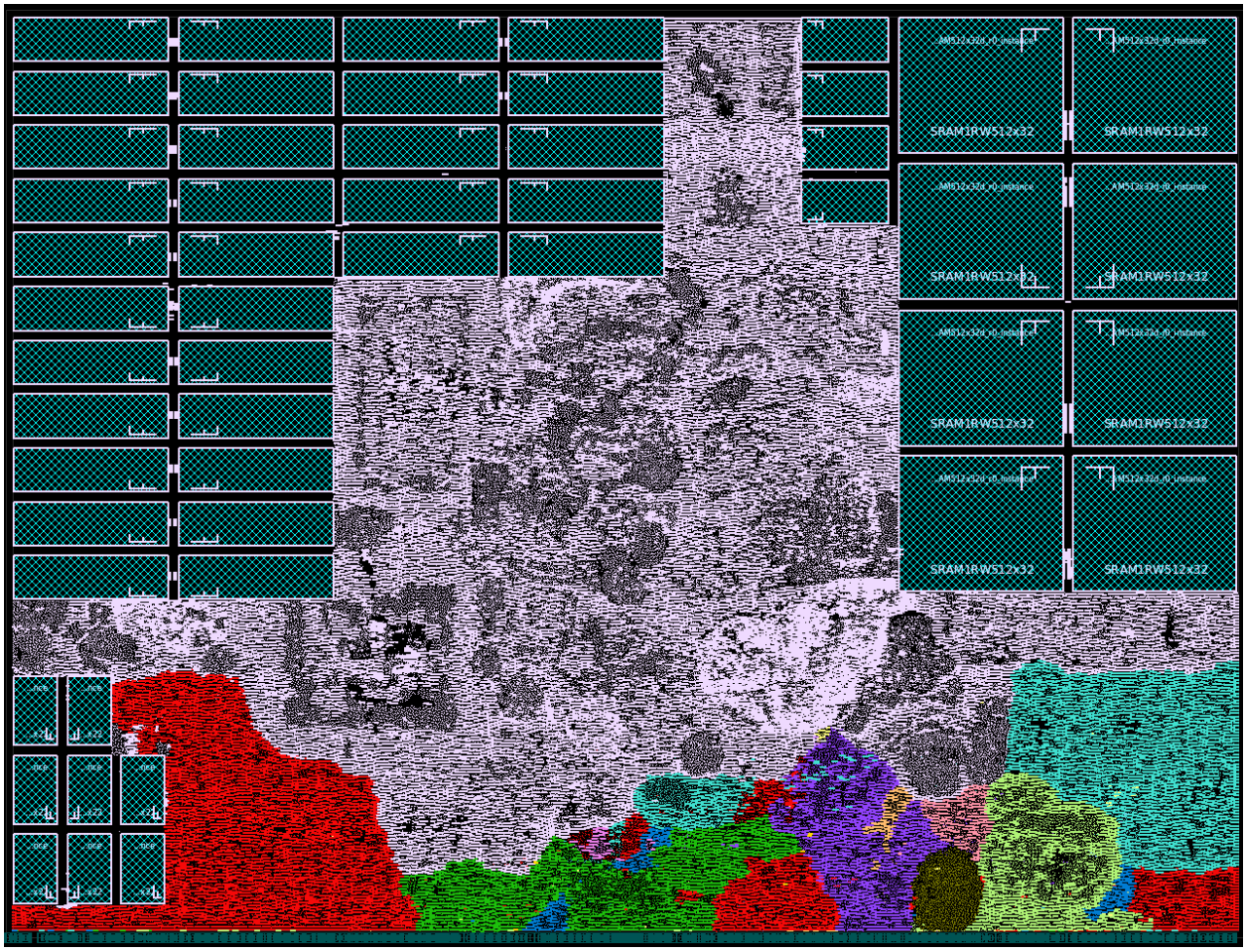
It does not match what I expected after looking into the timing for ORCA I noticed that the paths for the physical are better than the paths for the logical as well. The way that I can see things making sense is that the physical compiler does further optimization or has an idea of what a fake path is.

I would have liked to do more with the actual placing and routing. The best part of the lab was figuring out how to fix the voltage domain. It would of been nice to analyze the placement of

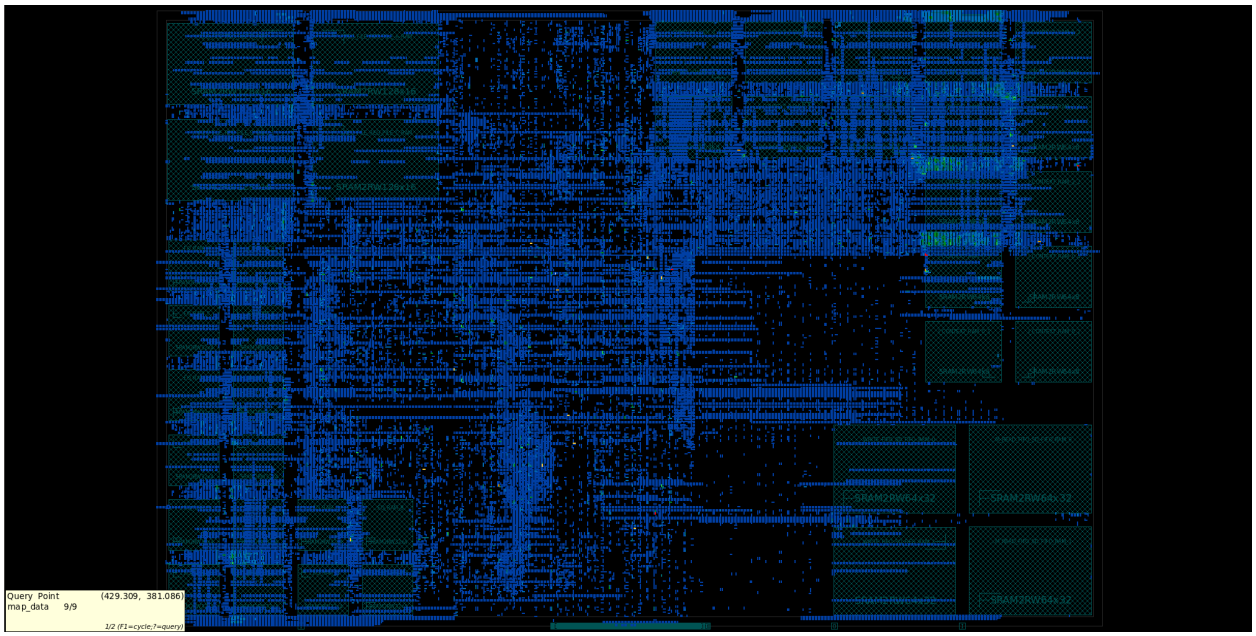
orca,



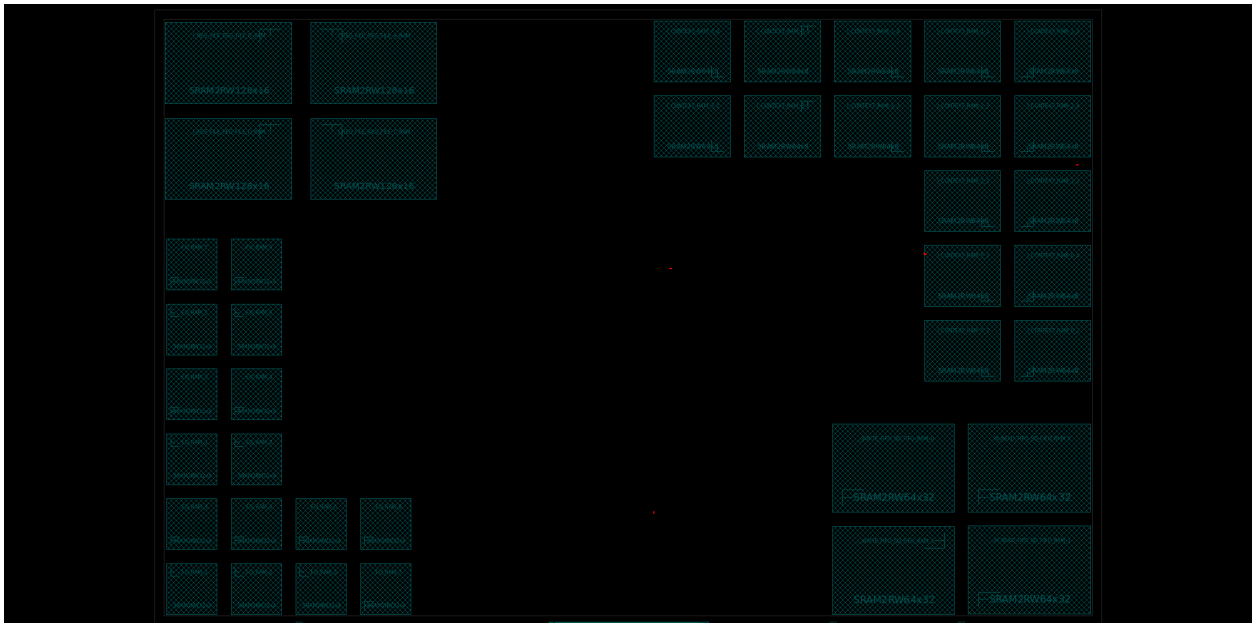
This placement looks really bad, judging from the lecture, there are cells in between the macros. Compared to the rocket hierarchy which looks much cleaner



I also had a good time looking at the congestion



It could even show where the congestion is really bad



Overall the project was fun and working with the synopsis tools was the best part. Fixing the configs was the worst part and it felt a little outside the scope of what the lab was trying to teach.

[1] https://www.techdesignforums.com/practice/guides/unified-power-format-upf/