Design Compiler Synthesis Assignment Report

Tai Duc Nguyen (tdn47) 02/01/2020 Instructor name: Baris Taskin

Table of Content:

1. Report Summary

This report consists of the result and analysis of 5 different benchmark circuits, all of which are bounded by different timing & performance constraints. In this assignment, the 2 performance constraints are: 100Mhz and 500Mhz; and the … timing/clock constraints are:

- Clock latency 0.4 ns
- Clock uncertainty 0.05 ns
- Clock transition 0.1ns
- Input delay 2 ns (*)
- Output delay 2ns (*)
- Load of 0.3 on all the outputs

Using these constraints, the 5 benchmark circuits (s1238, s15850, s35932, s386, s9234) described in Verilog will be synthesized into a netlist of logic gates using the Synopsis's Design Compiler Software. The results (shown in later sections) indicate that all 5 circuits can perform well under 100Mhz but cannot run under 500Mhz due to having paths that violate the slack requirements. All the circuits' timing reports (produced by Synopsis's PrimeTime Software) are then analyzed, and it is apparent that the input/output delay of the data paths (the paths inside the combinational circuits) are much larger than the time needed for the data to be computed. Hence, if the input and output delays are forgiven, then 4 out of 5 circuits can perform under a 500Mhz clock. If the input and output delays are mandatory, then the highest clock frequencies that the circuits can withstand are shown in the sections below.

2. Synthesized Design with Full Constraints

In this section, the area, power and timing reports for the synthesized design of the 5 benchmark circuits are detailed in Table 1 and 2 below:

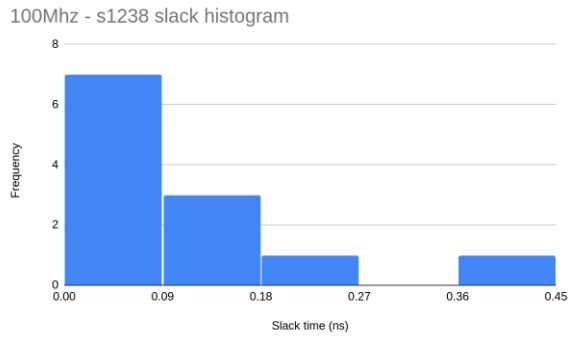
Design	Area (μm^2)	Power (μW)	WNS (ns)	TNS (ns)	Cell Count	No. of violating paths
s1238	921.601008	28.3344	0	0	340	0
s15850	8631.1469	198.3502	0	0	2178	0
s35932	30263.54327	639.5707	0	0	5972	0
s386	198.590684	6.5227	0	0	79	0
s9234	3845.636578	95.0375	0	0	1062	0

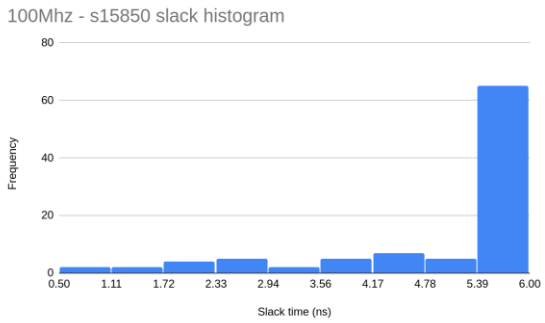
Table 1: Area, power, worst negative slack, total negative slack, cell count and number of violating paths are shown for 5 benchmark circuits operating at 100Mhz

Design	Area (μm^2)	Power (μW)	WNS (ns)	TNS (ns)	Cell Count	No. of violating paths
s1238	1496.501803	58.7079	-3.39	-62.1	632	32
s15850	9721.486871	259.9825	-4.11	-1230.64	2594	621
s35932	27730.74094	772.4252	-2.87	-3553.87	5734	2048
s386	296.939745	11.5519	-2.58	-23.27	119	13
s9234	4232.049392	118.7659	-2.52	-275.96	1244	200

Table 2: Area, power, worst negative slack, total negative slack, cell count and number of violating paths are shown for 5 benchmark circuits operating at 500Mhz

The histograms for the slack timings of 5 benchmark circuits at 100Mhz and 500Mhz are shown below:





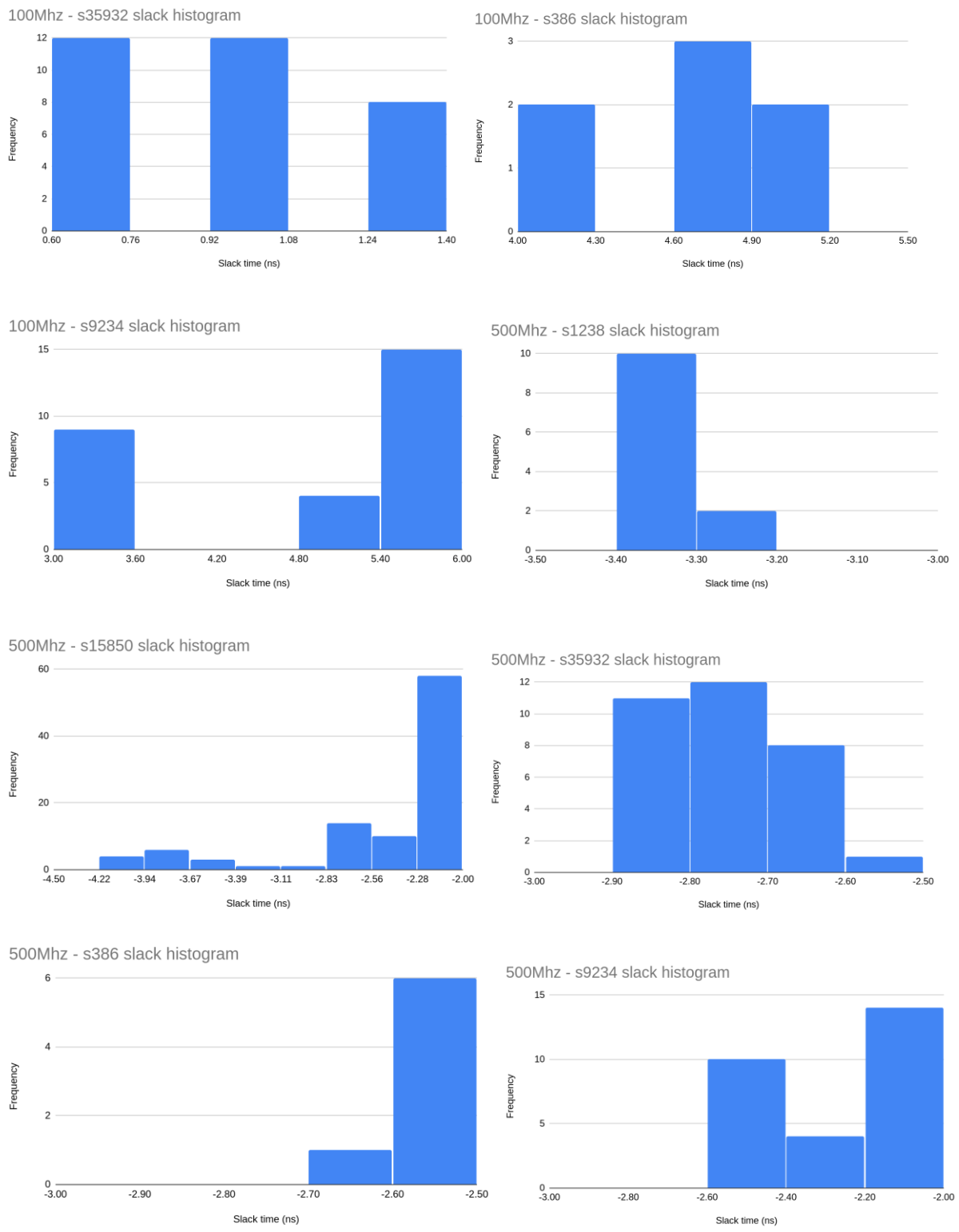


Figure 1: Histograms for the slack timings of 5 benchmark circuits at 100Mhz and 500Mhz

3. Power, Performance and Area Analysis

From the results available in the section above, it is apparent that higher clock frequency increases both power and area. However, it is noted that a 5-time-increase in clock frequency only induces on average 1.5 times increase in power and 1.25 times increase in area. This could be a result of the fact that the synthesized benchmark circuits have the amount of leakage power >> amount of dynamic power. Particularly, in benchmark s15850, at 100Mhz and 500Mhz, the total leakage power is 27.7 and 5.8 times larger than the total dynamic power, respectively. Also, since the area constraint is set at "maximum", the Design Compiler Software tries to minimize the area "as much as heuristically possible." This cause the area to only grow to an average of 1.25 times, at the cost of high localized heat dissipation.

4. Slack Timings Analysis

From the design, it is known that increasing clock frequency causes the design to include more registers (D-flip-flops) so that they can hold value computed different stages. However, with a clock frequency of 500Mhz, even after putting in these registers after every stage, the clock period is still too small for the data to be able to come out of each stage and into the DFFs. Hence, the slacks are all negative. Looking carefully into the constraints, the clock period is 2 (ns) but the input and output delay are 2 (ns) each. Therefore, if the time for one stage to finish computation is X ns, then the total time needed for the data to travel from the DFF of the previous stage to the DFF of the next stage is 2 + data_arrive_time = 2 + clock_period.

With this knowledge in mind, if the input and output delay constraints are removed (set to 0.0), then all stages which take less than 2 (ns) to compute should fit the slack requirement. This is verified by running the Design Compiler again with the 2 constraints command commented out. The results are expected: Using PrimeTime, all benchmark circuits pass the slack requirement (at 500Mhz), except for s15850, whose WNS is -0.37 ns.

However, if such input & output delay constraints exist, then the minimum clock period that each benchmark can support is max(data_travel_time) (ns) for that benchmark. This is shown theoretically and experimentally in the table below:

Design	Min Clock Period (ns) (theoretical)	Max Clock Period (ns) (synthesized)	No. Violations (synthesized)	Clock Freq (Mhz) (synthesized)
s1238	3.74	3.74	0	267.4
s15850	4.46	4.46	0	224.2
s35932	3.22	3.22	0	310.6

s386	2.95	2.95	0	339.0
s9234	2.87	2.87	0	348.4

Table 2: Theoretical and synthesized clock periods & frequencies that satisfy the slack requirement for each benchmark circuit

5. TCL files used in simulation

The code for the DC synthesis file is below:

```
######################### DC Synthesis Script ################################
## Give the path to the verilog files and define the WORK directory
lappend search_path ../src_assignment/
define_design_lib WORK -path "work"


## Define the library location
set link_library [ list /mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95v125c.db
/mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95v25c.db
/mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95vn40c.db]


set target_library [ list /mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95v25c.db]


## read the verilog files
analyze -library WORK -format verilog [list #bench_mark.v dff.v]


elaborate   -architecture verilog -library WORK #bench_mark


## Check if design is consistent
check_design  > reports/#freq/#bench_mark_synth_check_design.rpt


## Create Constraints
create_clock CK -name ideal_clock -period #clk_period
set_clock_latency -source 0.4 [get_clocks ideal_clock]
set_clock_uncertainty 0.05 [get_clocks ideal_clock]
```

```
set_clock_transition 0.1 ideal_clock
set_input_delay 2.0 [all_inputs] -clock ideal_clock
set_output_delay 2.0 [all_outputs ] -clock ideal_clock
set_max_area 0
set_load 0.3 [ all_outputs ]

## Compilation
## you can change medium to either low or high
compile -area_effort medium -map_effort medium

## Below commands report area , cell, qor, resources, and timing information
needed to analyze the design.
  report_area > reports/#freq/#bench_mark_synth_area.rpt
  report_cell > reports/#freq/#bench_mark_synth_cells.rpt
  report_qor  > reports/#freq/#bench_mark_synth_qor.rpt
  report_resources > reports/#freq/#bench_mark_synth_resources.rpt
  report_timing -max_paths 10 > reports/#freq/#bench_mark_synth_timing.rpt
  report_power > reports/#freq/#bench_mark_synth_power.rpt

## Dump out the constraints in an SDC file
  write_sdc  const/#freq/#bench_mark_dff.sdc

## Dump out the synthesized database and gate-level-netlist
  write -f ddc -hierarchy -output output/#freq/#bench_mark_dff.ddc
  write -hierarchy -format verilog -output  output/#freq/#bench_mark_dff.v

  exit
```

The code for the PT synthesis file is below:

```
######################### Pre-Layout PrimeTime Script ###################

## Define the library location
```

```
set link_library [ list /mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95v125c.db
/mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95v25c.db
/mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95vn40c.db]


set target_library [ list /mnt/class_data/ecec574-
w2019/PDKs/SAED32nm/lib/stdcell_rvt/db_ccs/saed32rvt_ss0p95v25c.db ]

## read the verilog files

read_verilog ../dc_synth/output/#freq/#bench_mark_dff.v


## Set top module name

current_design #bench_mark


## Read in SDC from the synthesis

source ../dc_synth/const/#freq/#bench_mark_dff.sdc


## Analysis reports


report_timing -from [all_inputs] -max_paths 1000 -slack_lesser_than 20 -to
[all_registers -data_pins] > reports/#freq/#bench_mark_timing.rpt

report_timing -from [all_register -clock_pins] -max_paths 1000 -
slack_lesser_than 20 -to [all_registers -data_pins]  >>
reports/#freq/#bench_mark_timing.rpt

report_timing -from [all_registers -clock_pins] -max_paths 1000 -
slack_lesser_than 20 -to [all_outputs] >>
reports/#freq/#bench_mark_timing.rpt

report_timing -from [all_inputs] -to [all_outputs] -max_paths 1000 -
slack_lesser_than 20 >> reports/#freq/#bench_mark_timing.rpt

report_timing -from [all_registers -clock_pins] -to [all_registers -
data_pins] -delay_type max  >> reports/#freq/#bench_mark_timing.rpt

report_timing -from [all_registers -clock_pins] -to [all_registers -
data_pins] -delay_type min  >> reports/#freq/#bench_mark_timing.rpt


report_timing -transition_time -capacitance -nets -input_pins -from
[all_registers -clock_pins] -to [all_registers -data_pins]  >
reports/#freq/#bench_mark_timing.tran.cap.rpt
```

```
## Save outputs
save_session output/#freq/#bench_mark_dff.session

exit
```