Michael Zhang, Rusheel Dasari, Joe Ciminski

ECE552 Computer Architecture Final Report

Overview

Our project consists of making a working CPU. Our CPU for Phase 1 was a single cycle CPU that ran one instruction for every cycle. For Phase 2, our CPU was a pipelined CPU which broke the operations to run each instruction into the Fetch, Decode, Execute, Memory, and Memory Writeback stages, and the CPU would run each one of these stages per cycle as well as handling stalls and full forwarding. For Phase 3, we removed the automatic assumption that the memory was instantly fast and added both an instruction cache and data cache that was designed as a 2 way set associative cache with both caches hooked up to the same 4 cycle memory.

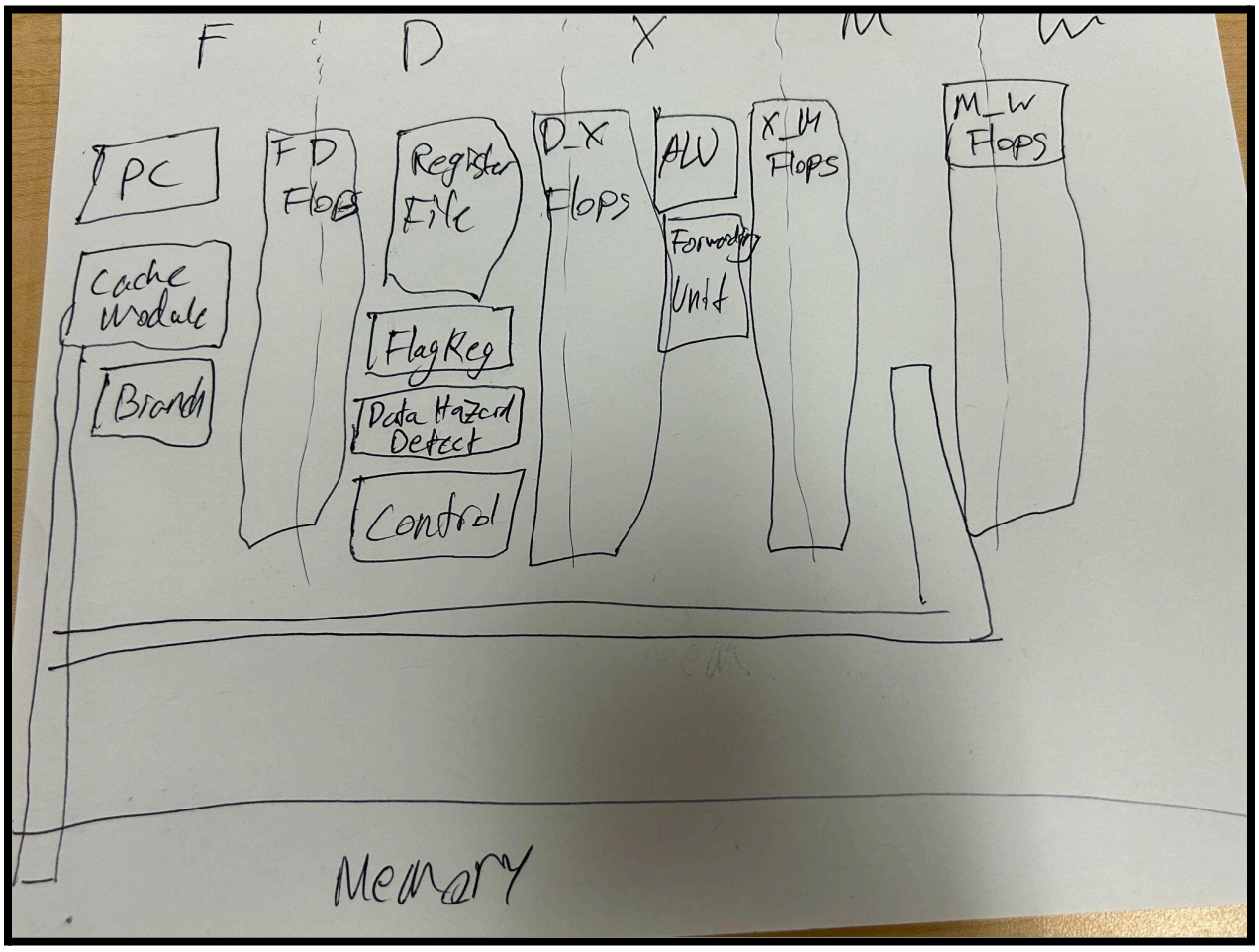


Diagram of our CPU

Responsibility and Task Breakdown

Our team was an interesting one, where we would initially work on various work ourselves and then debugging was pretty much a collective work.

	Phase 1		Phase 2		Phase 3	
	Desc	%	Desc	%	Desc	%
Rusheel	Control Unit & Debugging	33	Decode, Debugging	33	Report, Debugging	33
Joe	Memory Units	33	Fetch, Debugging	33	Report, Debugging	33
Michael	ALU and Sub-Components	33	ALU, Memory, Writeback, Debugging	33	Report, Debugging	33

Special Features

Our Own Testbenches: We created multiple mini-testbenches to assist in debugging. During Phase III, for example, we created an FSM testbench to test the FSM's correctness before the Cache had been implemented.

Improvement on Existing Testbenches: In addition to our creation of additional testbenches, we improved the output of the existing testbenches by making them print the values of key variables at each cycles inside the verilogsim.trace file.

Exceptional Debugging Effort: While working on this project, we ran into a number of unique and significant problems, including infinite loops in Phase I and issues with the Branch and HLT instructions in Phase 2. During this period, our debugging progress in Phase 1 was extremely slow due to the quantity of unforeseen infinite loops and various problems as explained below. For Phase 3 our debugging process started relatively early, however work on debugging continued the longest out of all the 3 phases.

Phase 1: For phase 1 we had multiple issues. Firstly we had some time management issues, so we ended up submitting the code one day late. Secondly we had multiple issues of moving files around. So we had a few faulty systems in place. One was where we had to use the site codeshare.io to share code like a google doc. This worked fine initially because we were all creating the CPU.sv, we ended up implementing the various parts that we filled out. But after we had debugged, we had multiple issues stemming from it.

First we were using the same codeshare doc to debug so that multiple people would be able to debug the file at once. However, at the same time we would also edit our individual code base we stored locally since that's where we were running the

model/questa sim files at. Due to this there were multiple times where we would end up changing our code locally, run it on modelsim, and then copy and paste the CPU.sv code back to codeshare and vice versa. This made some changes that worked overwritten over and over, and this methodology of using codeshare was eventually scrapped near the beginning of Phase 2 due to this conflict.

We also had issues because the location of the git clone was different from the location of the machine running it so there were some hiccups where we might've made the change on the machine running questasim, and we would post the results onto our group chat for progress, but end up forgetting to update our local git clone for that matter.

It was also at this point that we realized that our questasim configurations were weird. For some people in the group, but not others, the questasim would simulate with optimization even explicitly making sure optimization was not checked. Because of this there was a while in which debugging was particularly difficult because it would not reveal the deeper layers and waves from that. It was only until deeper analysis on commands, that now I use this command to run questasim, which I believe was found about some time after submitting phase 1. Here is the command used to run the simulation

```
vsim work.cpu_ptb -novopt -suppress 12110 -suppress 8891
```

Command used to run simulation

We also misinterpreted what the late policy was. We assumed it was a -20%, so hypothetically if we got 80/100 it would be bumped down to 64/100. We didn't realize it was a -20 points and since we got 30/100 for phase 1, it meant that without the 1 day late nerf it would be 50/100 and so the late penalty was a -40% debuff. Due to this extreme realization, we made sure not to risk any late work at all, and made sure to plan ahead of time for the projects. The worst part about this 1 day extra work was that we still suffered from the conflicts faced from the previous day, as well as merging conflicts that wiped some very important features due to our panicky nature of trying to get it done instead of carefully looking at merge issues. Our methodology changes are more detailed in phase 2.

Phase 2: For Phase 2, in our opinion this part of the project felt very long to build, however relatively more easy to debug. This time we made some changes into our work. First we started a bit earlier, and we divided the CPU into the respective comment slots (we had very thick bordered comments to separate where each pipeline stage should go) and we did the slow of the Phase 1 stuff over to Phase 2. Building the code took long

enough that we almost only had 2 days before the deadline to debug our code and we were not going to risk a 1 day delay like we mistakenly did previously in Phase 1.

Again, we started using the codeshare.io document to real time edit our parts, however this time we quickie dropped it and instead relied more solely on github push/pulls as well as regularly notifying each other on Discord about any tiny changes we were going to make, instead of relying on the realtime codeshare.io and randomly assuming whether or not someone else saw those changes. This new system worked better, because we could easily just go on github to see the exact line changes (usually for debugging it was only a few lines changed) and directly notifying each other on discord increased both our productivity and compatibility we had for each other.

In the end, we solved most of the debugging for Phase 2 within the last 2 days, and although we submitted our project about 10 mins before the deadline with broken Branching, this was a proud moment for us to celebrate our work and productivity as a team.

Phase 3: Phase 3 was actually much harder than we thought. Compared to Phase 2, building it was pretty simple, but the debugging was much much harder. Building it we finished about 4-5 days before the deadline, however we were stuck debugging it right up until the deadline. I think most of the issue with doing Phase 3 is that there is a lot of hierarchical files and now there's a much more serious consideration to timing. A lot of the debugging was maybe adding a flop here and there or adding a stack of flops to delay the signal by a bit, etc.

Extra Credit Synthesis: For Synthesis we initially started to work off the provided github by Zhewen404 initially we had a few major roadblocks here. The first issue was that the shell script seemed to hang/freeze for over 30 mins (actually over 40 mins) while sitting on DataArray. We initially were unsure about why this was and asking on piazza yielded little results (nearing the deadline, we haven't had a response on the Synthesis question on Piazza). When we eventually decided to read the shell script's contents itself, it looked like it was trying to convert all .v files into .sv. So then we converted the names of all the files into .sv and then we hit another roadblock where it was giving weird results. Since it was midnight we 2 days before the deadline, we made sure to document as many changes and analyses of what we were seeing on Piazza, however we couldn't find out why it wasn't working.

After that, we tried using the synthesis tutorial found from Hoffman's ECE551 course. This file was actually buried in an old usb that we luckily found. We quickly created the .dc synthesis script (which is mirrored very similar to the ECE551 final project script, with some changes) as shown below.

```
##########################################Read in
file#################################################
##########################################################################
################################
read_file -format sverilog {ALU.sv, Forwarding_Unit.sv X_M_Flops.sv,
Branch.sv, M_W_Flops.sv, cacheControl_cache_fill_FSM.sv,
CLA_16bit.sv, CLA_4bit.sv, MetaDataArray.sv, CLA_4bit_tb.sv,
PADDSB.sv, CLA_8bit.sv, PC.sv, multicycle_memory.sv, CLA_8bit_tb.sv,
CPU.sv, RED.sv, Cache.sv, RED_tb.sv, CacheModule.sv, ROR.sv,
Control.sv, Register.sv, RegisterFile.sv, D_X_Flops.sv, DataArray.sv,
SATADDSUB_16bit.sv, Data_Hazard_Detect.sv, SATADDSUB_16bit_tb.sv,
FLAG_Reg.sv, Shifter.sv}


##################################Set current design, create
clock###################################
##########################################################################
################################
set current_design CPU

# create_clock -name "clk" -period 2.5 -waveform {0 1} {clk}
create_clock -name "clk" -period 50 -waveform {0 1} {clk}

# set don't touch on clock network
set_dont_touch_network [find port clk]



#############################Set input delay and drive
strengths#####################################
#########################################################################
################################
set prim_inputs [remove_from_collection [all_inputs]\
 [find port clk,rst_n]]
```

```
 set_input_delay -clock clk 0.25 $prim_inputs

set_driving_cell -lib_cell NAND2X2_LVT -library saed32lvt_tt0p85v25c
$prim_inputs

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



#####################################Set output delay and
loads#####################################
#########################################################################
##############################
# tell Synopsys w eneed outputs valid 0.55ns before the next rising
edge of clock
set_output_delay -clock clk 0.35 [all_outputs]
set_load 50 [all_outputs]
#########################################################################
##############################

#################################Set max transition time and wire load
model#########################
#########################################################################
##############################
set_max_transition 0.10 [current_design]


set_wire_load_model -name 16000 \
  -library saed32lvt_tt0p85v25c

set_clock_uncertainty 0.125 clk
#########################################################################
##############################
```

```
compile -map_effort medium
ungroup -all -flatten
compile -map_effort high


set_multicycle_path 2 -setup -from [find pin iNEMO/ptch_*_reg*/CLK]
set_multicycle_path 2 -setup -from [find pin iNEMO/AZ*_reg*/CLK]
set_multicycle_path 2 -setup -from [find pin
iBAL/iCNTRL/ss_tmr_reg*/CLK]
set_multicycle_path 2 -hold -from [find pin iNEMO/ptch_*_reg*/CLK]
set_multicycle_path 2 -hold -from [find pin iNEMO/AZ*_reg*/CLK]
set_multicycle_path 2 -hold -from [find pin
iBAL/iCNTRL/ss_tmr_reg*/CLK]

set_fix_hold clk

compile -map_effort medium
ungroup -all -flatten
compile -map_effort high


########################Log area & times and write out gate level
verilog netlist###################
#########################################################################
################################
report_timing -delay max
report_timing -delay min
report_area > area.txt
report_timing > timing.txt

# write -format verilog CPU -output CPU.vg
```

CPU.dc script

After fixing up the errors (which was much easier to do since now that we know we created the script ourselves, we knew any errors that were produced on us entirely). We ran it and then got the area.txt and timing.txt which is shown in the Completeness section.

Completeness

Phase 1: For Phase 1 our submission was mostly ineligible since we accidentally overwritten the removal of the data. Later we found that we accidentally overwrote the disabling of the writeback to read an infinite loop, and we ended up having a lot of Xs on the result. We actually weren't sure if the Xs was required or not and we developed a lot of thrown away features later on because those features were based on dealing with the Xs. It wasn't until we got feedback for our Phase 1 that we realized that there should be little to almost no Xs at all. Later on we made changes as described in the results section of Phase 1. But at the same time we had obvious incompleteness like forgetting to implement the PC instructions and properly implementing branching.

Phase 2: For Phase 2 we finished most of the functionality needed in the pipelined processor. We split our processor into the Fetch, Decode, Execute, Memory, and Memory Writeback stage. We implemented almost everything, with only the branch not working, which was a problem we faced in Phase 1 as well.

Phase 3: Should be progressing

Phase 3 Extra Credit Synthesis: For Phase 3 Extra Credit Synthesis, we were able to generate the area.txt and timings.txt file.

Testing

We ran all three test cases that were given for each. We had an automated test bench for most of our individual components.
By using a combination of the test cases that were provided to us and supplementary tests we designed ourselves, we were able to compare the CPU's behavior to logically-correct behavior. For the tests that were provided to us, we manually calculated the operations performed by each instruction, as well as any register changes they would affect. By using ModelSim's Wave tool and the ptrace.log file, we were then able to examine the values of the various registers and wires at each clock cycle, using these values to locate errors in our designs.

Results

Phase 1 Results and Waveforms: For Phase 1, we had a major problem where we ended up going 1 day late as well as overwriting some fixes that we made before. We later made a resubmit by email, but due to some misunderstanding it was too late, so here were the fixes after the demo:

- Error: Long lines of xxxx over and over
    - We realized that the previous project directory was corrupted and when we ran it again in a new project directory everything worked fine. Also made minor changes to the testbench file to correctly reference the signals

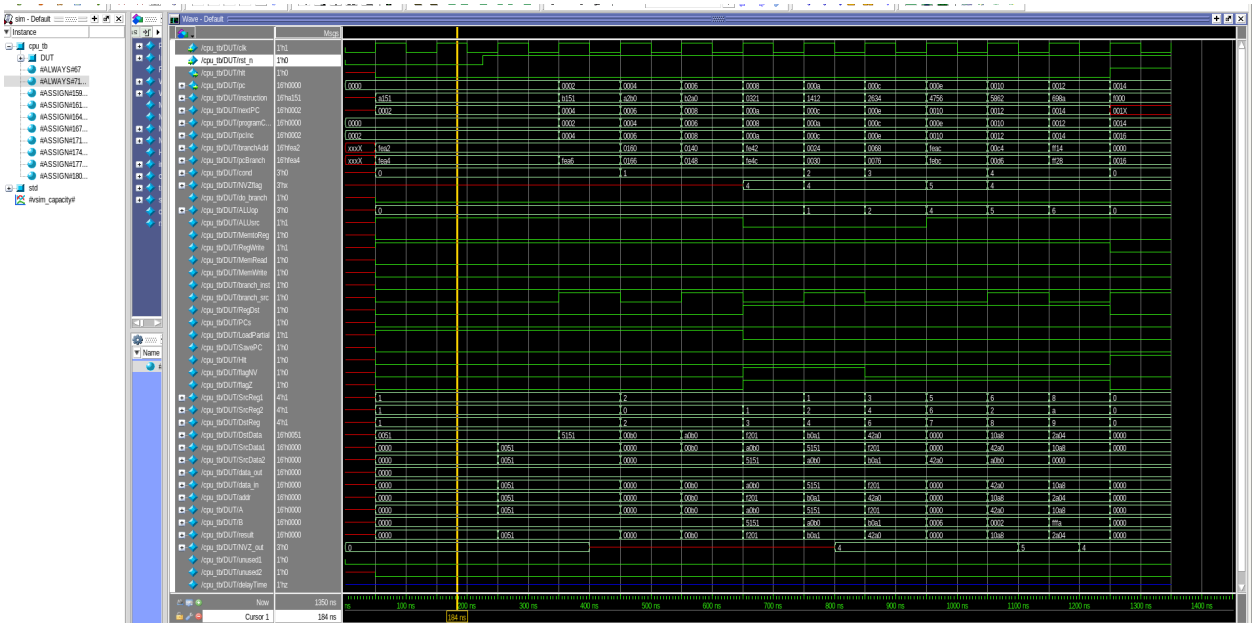
- Error with xxxx at the beginning and during write on read instructions
    - We had made the change previously, however due to merge/file transfers conflicts it reverted removing the write over read infinite loop, and fixing this fixes all those errors
- PADDSB errors
    - This was an interesting one where the fix was very simple, it was trying to reference bits 15, 11, and 7, on a 4 bit number. The interesting thing is, is that QuestaSim never threw compilation errors or warnings and it would run completely fine until we found it by coincidence.

These fixes above were what was submitted by email later. After that most of the errors on phase 1 was that we forgot to even implement the PC instruction at all and our branching seemed to not work correctly.

```
Log:
SIMLOG:: Cycle           3 PC: 00000000 I: 0000a151 R: 1   1 00000051
M: 0 0 00000000 00000000
SIMLOG:: Cycle           4 PC: 00000000 I: 0000a151 R: 1   1 00000051
M: 0 0 00000051 00000000
SIMLOG:: Cycle           5 PC: 00000002 I: 0000b151 R: 1   1 00005151
M: 0 0 00000051 00000000
SIMLOG:: Cycle           6 PC: 00000004 I: 0000a2b0 R: 1   2 000000b0
M: 0 0 00000000 00000000
SIMLOG:: Cycle           7 PC: 00000006 I: 0000b2a0 R: 1   2 0000a0b0
M: 0 0 000000b0 00000000
SIMLOG:: Cycle           8 PC: 00000008 I: 00000321 R: 1   3 0000f201
M: 0 0 0000f201 00000000
SIMLOG:: Cycle           9 PC: 0000000a I: 00001412 R: 1   4 0000b0a1
M: 0 0 0000b0a1 00000000
SIMLOG:: Cycle          10 PC: 0000000c I: 00002634 R: 1   6 000042a0
M: 0 0 000042a0 00000000
SIMLOG:: Cycle          11 PC: 0000000e I: 00004756 R: 1   7 00000000
M: 0 0 00000000 00000000
SIMLOG:: Cycle          12 PC: 00000010 I: 00005862 R: 1   8 000010a8
M: 0 0 000010a8 00000000
SIMLOG:: Cycle          13 PC: 00000012 I: 0000698a R: 1   9 00001504
M: 0 0 00001504 00000000
SIMLOG:: Cycle          14 PC: 00000014 I: 0000f000 R: 0   0 00000000
M: 0 0 00000000 00000000
SIMLOG:: Processor halted
```

```
SIMLOG:: sim_cycles          14

SIMLOG:: inst_count          12



Trace:
INUM:         0 PC: 0x0000 REG:  1 VALUE: 0x0051
INUM:         1 PC: 0x0000 REG:  1 VALUE: 0x0051
INUM:         2 PC: 0x0002 REG:  1 VALUE: 0x5151
INUM:         3 PC: 0x0004 REG:  2 VALUE: 0x00b0
INUM:         4 PC: 0x0006 REG:  2 VALUE: 0xa0b0
INUM:         5 PC: 0x0008 REG:  3 VALUE: 0xf201
INUM:         6 PC: 0x000a REG:  4 VALUE: 0xb0a1
INUM:         7 PC: 0x000c REG:  6 VALUE: 0x42a0
INUM:         8 PC: 0x000e REG:  7 VALUE: 0x0000
INUM:         9 PC: 0x0010 REG:  8 VALUE: 0x10a8
INUM:        10 PC: 0x0012 REG:  9 VALUE: 0x1504
INUM:        11 PC: 0x0014
```

Phase 1 Test 1 Output



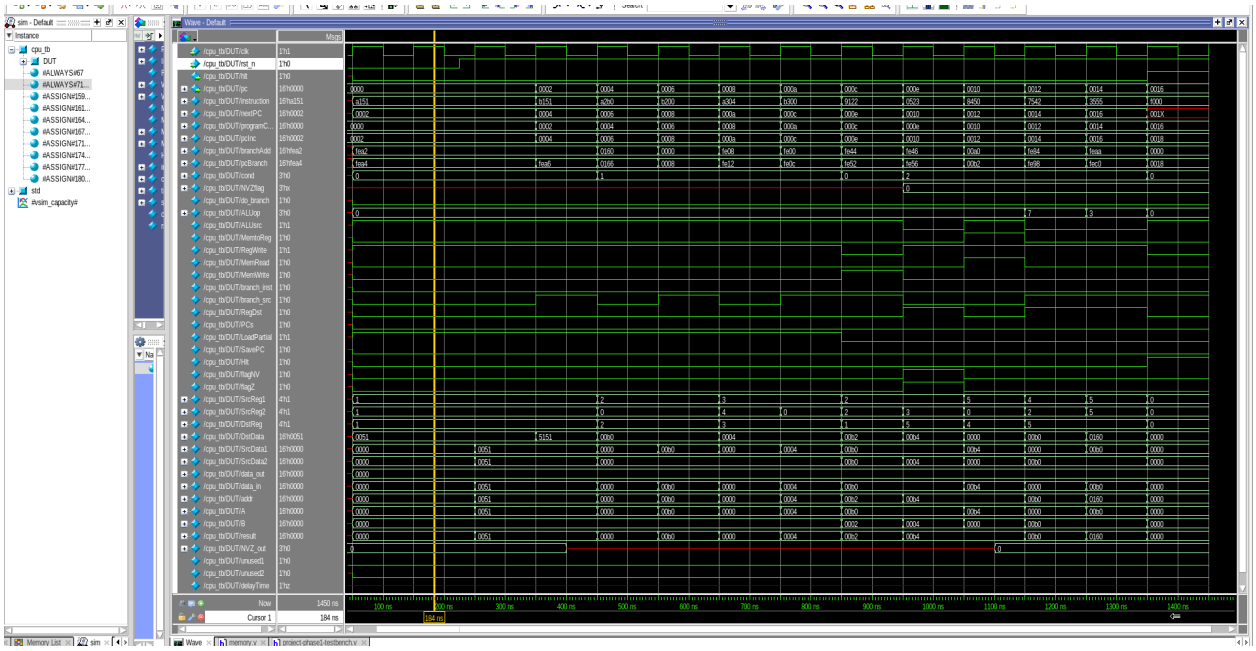
Phase 1 Test 1 Waveform

```
Log:
```

```
SIMLOG:: Cycle           3 PC: 00000000 I: 0000a151 R: 1   1 00000051
M: 0 0 00000000 00000000
SIMLOG:: Cycle           4 PC: 00000000 I: 0000a151 R: 1   1 00000051
M: 0 0 00000051 00000000
SIMLOG:: Cycle           5 PC: 00000002 I: 0000b151 R: 1   1 00005151
M: 0 0 00000051 00000000
SIMLOG:: Cycle           6 PC: 00000004 I: 0000a2b0 R: 1   2 000000b0
M: 0 0 00000000 00000000
SIMLOG:: Cycle           7 PC: 00000006 I: 0000b200 R: 1   2 000000b0
M: 0 0 000000b0 00000000
SIMLOG:: Cycle           8 PC: 00000008 I: 0000a304 R: 1   3 00000004
M: 0 0 00000000 00000000
SIMLOG:: Cycle           9 PC: 0000000a I: 0000b300 R: 1   3 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          10 PC: 0000000c I: 00009122 R: 0   1 000000b2
M: 0 1 000000b2 00000000
SIMLOG:: Cycle          11 PC: 0000000e I: 00000523 R: 1   5 000000b4
M: 0 0 000000b4 00000000
SIMLOG:: Cycle          12 PC: 00000010 I: 00008450 R: 1   4 00000000
M: 1 0 000000b4 00000000
SIMLOG:: Cycle          13 PC: 00000012 I: 00007542 R: 1   5 000000b0
M: 0 0 000000b0 00000000
SIMLOG:: Cycle          14 PC: 00000014 I: 00003555 R: 1   5 00000160
M: 0 0 00000160 00000000
SIMLOG:: Cycle          15 PC: 00000016 I: 0000f000 R: 0   0 00000000
M: 0 0 00000000 00000000
SIMLOG:: Processor halted

SIMLOG:: sim_cycles          15

SIMLOG:: inst_count          13



Trace:
INUM:        0 PC: 0x0000 REG:  1 VALUE: 0x0051
INUM:        1 PC: 0x0000 REG:  1 VALUE: 0x0051
INUM:        2 PC: 0x0002 REG:  1 VALUE: 0x5151
INUM:        3 PC: 0x0004 REG:  2 VALUE: 0x00b0
```

```
INUM:         4 PC: 0x0006 REG:  2 VALUE: 0x00b0
INUM:         5 PC: 0x0008 REG:  3 VALUE: 0x0004
INUM:         6 PC: 0x000a REG:  3 VALUE: 0x0004
INUM:         7 PC: 0x000c ADDR: 0x00b2 VALUE: 0x0000
INUM:         8 PC: 0x000e REG:  5 VALUE: 0x00b4
INUM:         9 PC: 0x0010 REG:  4 VALUE: 0x0000 ADDR: 0x00b4
INUM:        10 PC: 0x0012 REG:  5 VALUE: 0x00b0
INUM:        11 PC: 0x0014 REG:  5 VALUE: 0x0160
INUM:        12 PC: 0x0016
```

Phase 1 Test 1 Output



Phase 1 Test 2 Waveform

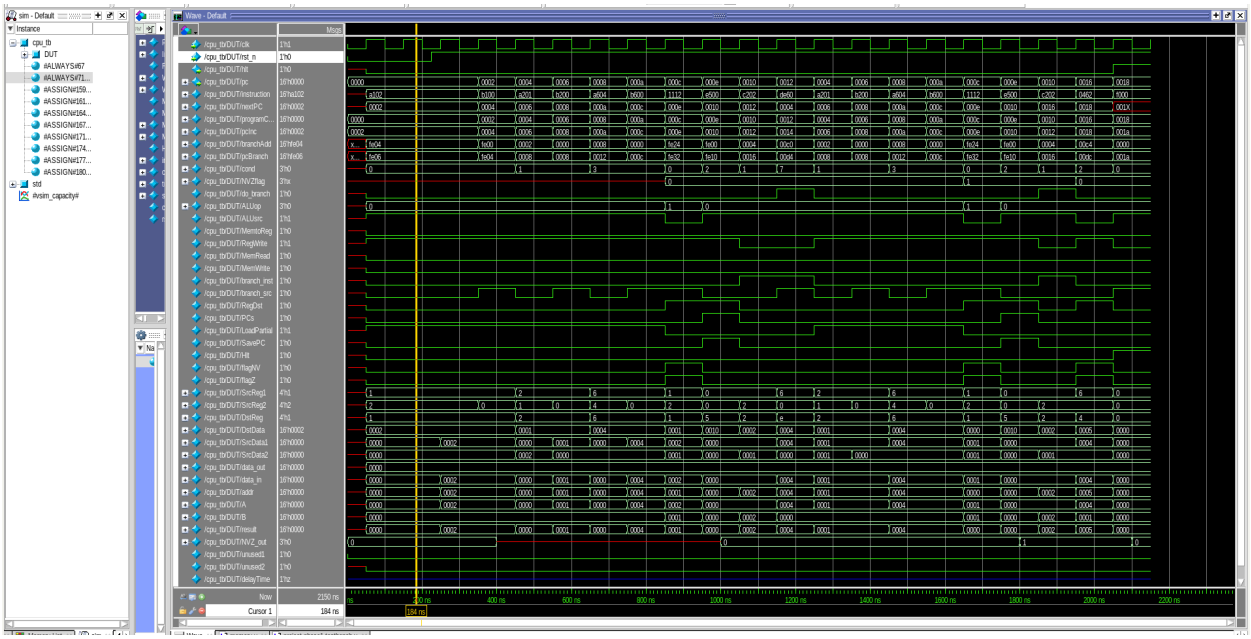
```
Log:
SIMLOG:: Cycle           3 PC: 00000000 I: 0000a102 R: 1   1 00000002
M: 0 0 00000000 00000000
SIMLOG:: Cycle           4 PC: 00000000 I: 0000a102 R: 1   1 00000002
M: 0 0 00000002 00000000
SIMLOG:: Cycle           5 PC: 00000002 I: 0000b100 R: 1   1 00000002
M: 0 0 00000002 00000000
SIMLOG:: Cycle           6 PC: 00000004 I: 0000a201 R: 1   2 00000001
M: 0 0 00000000 00000000
SIMLOG:: Cycle           7 PC: 00000006 I: 0000b200 R: 1   2 00000001
```

```
M: 0 0 00000001 00000000
SIMLOG:: Cycle           8 PC: 00000008 I: 0000a604 R: 1   6 00000004
M: 0 0 00000000 00000000
SIMLOG:: Cycle           9 PC: 0000000a I: 0000b600 R: 1   6 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          10 PC: 0000000c I: 00001112 R: 1   1 00000001
M: 0 0 00000001 00000000
SIMLOG:: Cycle          11 PC: 0000000e I: 0000e500 R: 1   5 00000000
M: 0 0 00000000 00000000
SIMLOG:: Cycle          12 PC: 00000010 I: 0000c202 R: 0   2 00000002
M: 0 0 00000002 00000000
SIMLOG:: Cycle          13 PC: 00000012 I: 0000de60 R: 0  14 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          14 PC: 00000004 I: 0000a201 R: 1   2 00000001
M: 0 0 00000001 00000000
SIMLOG:: Cycle          15 PC: 00000006 I: 0000b200 R: 1   2 00000001
M: 0 0 00000001 00000000
SIMLOG:: Cycle          16 PC: 00000008 I: 0000a604 R: 1   6 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          17 PC: 0000000a I: 0000b600 R: 1   6 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          18 PC: 0000000c I: 00001112 R: 1   1 00000000
M: 0 0 00000000 00000000
SIMLOG:: Cycle          19 PC: 0000000e I: 0000e500 R: 1   5 00000000
M: 0 0 00000000 00000000
SIMLOG:: Cycle          20 PC: 00000010 I: 0000c202 R: 0   2 00000002
M: 0 0 00000002 00000000
SIMLOG:: Cycle          21 PC: 00000012 I: 0000de60 R: 0  14 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          22 PC: 00000004 I: 0000a201 R: 1   2 00000001
M: 0 0 00000001 00000000
SIMLOG:: Cycle          23 PC: 00000006 I: 0000b200 R: 1   2 00000001
M: 0 0 00000001 00000000
SIMLOG:: Cycle          24 PC: 00000008 I: 0000a604 R: 1   6 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          25 PC: 0000000a I: 0000b600 R: 1   6 00000004
M: 0 0 00000004 00000000
SIMLOG:: Cycle          26 PC: 0000000c I: 00001112 R: 1   1 00007fff
M: 0 0 00007fff 00000000
```

```
SIMLOG:: Cycle          27 PC: 0000000e I: 0000e500 R: 1   5 00000000
M: 0 0 00000000 00000000
SIMLOG:: Cycle          28 PC: 00000010 I: 0000c202 R: 0   2 00000002
M: 0 0 00000002 00000000
SIMLOG:: Cycle          29 PC: 00000016 I: 00000462 R: 1   4 00000005
M: 0 0 00000005 00000000
SIMLOG:: Cycle          30 PC: 00000018 I: 0000f000 R: 0   0 00000000
M: 0 0 00000000 00000000
SIMLOG:: Processor halted

SIMLOG:: sim_cycles          30

SIMLOG:: inst_count          28

Trace:
INUM:        0 PC: 0x0000 REG:  1 VALUE: 0x0002
INUM:        1 PC: 0x0000 REG:  1 VALUE: 0x0002
INUM:        2 PC: 0x0002 REG:  1 VALUE: 0x0002
INUM:        3 PC: 0x0004 REG:  2 VALUE: 0x0001
INUM:        4 PC: 0x0006 REG:  2 VALUE: 0x0001
INUM:        5 PC: 0x0008 REG:  6 VALUE: 0x0004
INUM:        6 PC: 0x000a REG:  6 VALUE: 0x0004
INUM:        7 PC: 0x000c REG:  1 VALUE: 0x0001
INUM:        8 PC: 0x000e REG:  5 VALUE: 0x0000
INUM:        9 PC: 0x0010
INUM:       10 PC: 0x0012
INUM:       11 PC: 0x0004 REG:  2 VALUE: 0x0001
INUM:       12 PC: 0x0006 REG:  2 VALUE: 0x0001
INUM:       13 PC: 0x0008 REG:  6 VALUE: 0x0004
INUM:       14 PC: 0x000a REG:  6 VALUE: 0x0004
INUM:       15 PC: 0x000c REG:  1 VALUE: 0x0000
INUM:       16 PC: 0x000e REG:  5 VALUE: 0x0000
INUM:       17 PC: 0x0010
INUM:       18 PC: 0x0012
INUM:       19 PC: 0x0004 REG:  2 VALUE: 0x0001
INUM:       20 PC: 0x0006 REG:  2 VALUE: 0x0001
INUM:       21 PC: 0x0008 REG:  6 VALUE: 0x0004
INUM:       22 PC: 0x000a REG:  6 VALUE: 0x0004
INUM:       23 PC: 0x000c REG:  1 VALUE: 0x7fff
```

```
INUM:        24 PC: 0x000e REG:  5 VALUE: 0x0000
INUM:        25 PC: 0x0010
INUM:        26 PC: 0x0016 REG:  4 VALUE: 0x0005
INUM:        27 PC: 0x0018
```

Phase 1 Test 3 Output



Phase 3 Test 3 Waveform

Phase 2 Results and Waveforms:

```
SIMLOG:: Cycle            2 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Cycle            4 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Cycle            4 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Cycle            6 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0051     0051     0000
SIMLOG:: Cycle            6 PC:     0000 I:     a151 R: 1   1     0051
M: 0 0     5151     5100     0000
SIMLOG:: Cycle            8 PC:     0002 I:     b151 R: 1   1     5151
M: 0 0     00b0     00b0     0000
SIMLOG:: Cycle            8 PC:     0004 I:     a2b0 R: 1   2     00b0
M: 0 0     a0b0     a000     0000
SIMLOG:: Cycle           10 PC:     0006 I:     b2a0 R: 1   2     a0b0
```

```
M: 0 0     f201     5151     0000
SIMLOG:: Cycle          10 PC:     0008 I:     0321 R: 1   4     f201
M: 0 0     b0a1     a0b0     0000
SIMLOG:: Cycle          12 PC:     000a I:     1412 R: 1   4     b0a1
M: 0 0     42a0     b0a1     0000
SIMLOG:: Cycle          12 PC:     000c I:     2634 R: 1   7     42a0
M: 0 0     0000     0006     0000
SIMLOG:: Cycle          14 PC:     000e I:     4756 R: 1   7     0000
M: 0 0     10a8     0002     0000
SIMLOG:: Cycle          14 PC:     0010 I:     5862 R: 0   9     10a8
M: 0 0     2a04     000a     0000
SIMLOG:: Cycle          16 PC:     0012 I:     698a R: 0   9     2a04
M: 0 0     0000     0000     0000
SIMLOG:: Cycle          16 PC:     0014 I:     f000 R: 0   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Processor halted

SIMLOG:: sim_cycles          16

SIMLOG:: inst_count          13

REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
REG:  1 VALUE: 0x0051
REG:  1 VALUE: 0x5151
REG:  2 VALUE: 0x00b0
REG:  2 VALUE: 0xa0b0
REG:  4 VALUE: 0xf201
REG:  4 VALUE: 0xb0a1
REG:  7 VALUE: 0x42a0
REG:  7 VALUE: 0x0000
```

Phase 2 Test 1 Output Should Be Correct

```
SIMLOG:: Cycle           2 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Cycle           4 PC:     0000 I:     0000 R: 1   0     0000
```

```
M: 0 0     0000     0000     0000
SIMLOG:: Cycle           4 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Cycle           6 PC:     0000 I:     0000 R: 1   0     0000
M: 0 0     0051     0051     0000
SIMLOG:: Cycle           6 PC:     0000 I:     a151 R: 1   1     0051
M: 0 0     5151     5100     0000
SIMLOG:: Cycle           8 PC:     0002 I:     b151 R: 1   1     5151
M: 0 0     00b0     00b0     0000
SIMLOG:: Cycle           8 PC:     0004 I:     a2b0 R: 1   2     00b0
M: 0 0     00b0     0000     0000
SIMLOG:: Cycle          10 PC:     0006 I:     b200 R: 1   2     00b0
M: 0 0     0004     0004     0000
SIMLOG:: Cycle          10 PC:     0008 I:     a304 R: 1   3     0004
M: 0 0     0004     0000     0000
SIMLOG:: Cycle          12 PC:     000a I:     b300 R: 1   3     0004
M: 0 0     00b4     0004     0000
SIMLOG:: Cycle          12 PC:     000c I:     9122 R: 0   5     00b4
M: 0 1     00b4     0004     0000
SIMLOG:: Cycle          14 PC:     000e I:     0523 R: 0   5     00b4
M: 0 1     00b4     0000     0000
SIMLOG:: Cycle          14 PC:     0010 I:     8450 R: 1   4     00b4
M: 0 0     0004     0004     0000
SIMLOG:: Cycle          16 PC:     0012 I:     9452 R: 1   4     0004
M: 0 0     0004     0004     0000
SIMLOG:: Cycle          16 PC:     0012 I:     9452 R: 0   5     0004
M: 0 0     00b4     00b0     0000
SIMLOG:: Cycle          18 PC:     0014 I:     7542 R: 0   5     00b4
M: 0 0     0168     00b4     0000
SIMLOG:: Cycle          18 PC:     0016 I:     3555 R: 0   0     0168
M: 0 0     0000     0000     0000
SIMLOG:: Cycle          20 PC:     0018 I:     f000 R: 0   0     0000
M: 0 0     0000     0000     0000
SIMLOG:: Processor halted

SIMLOG:: sim_cycles          20

SIMLOG:: inst_count          15
```

```
REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
REG:  1 VALUE: 0x0051
REG:  1 VALUE: 0x5151
REG:  2 VALUE: 0x00b0
REG:  2 VALUE: 0x00b0
REG:  3 VALUE: 0x0004
REG:  3 VALUE: 0x0004
STORE: ADDR: 0x00b4 VALUE: 0x0004
STORE: ADDR: 0x00b4 VALUE: 0x0000
REG:  4 VALUE: 0x00b4
REG:  4 VALUE: 0x0004
```

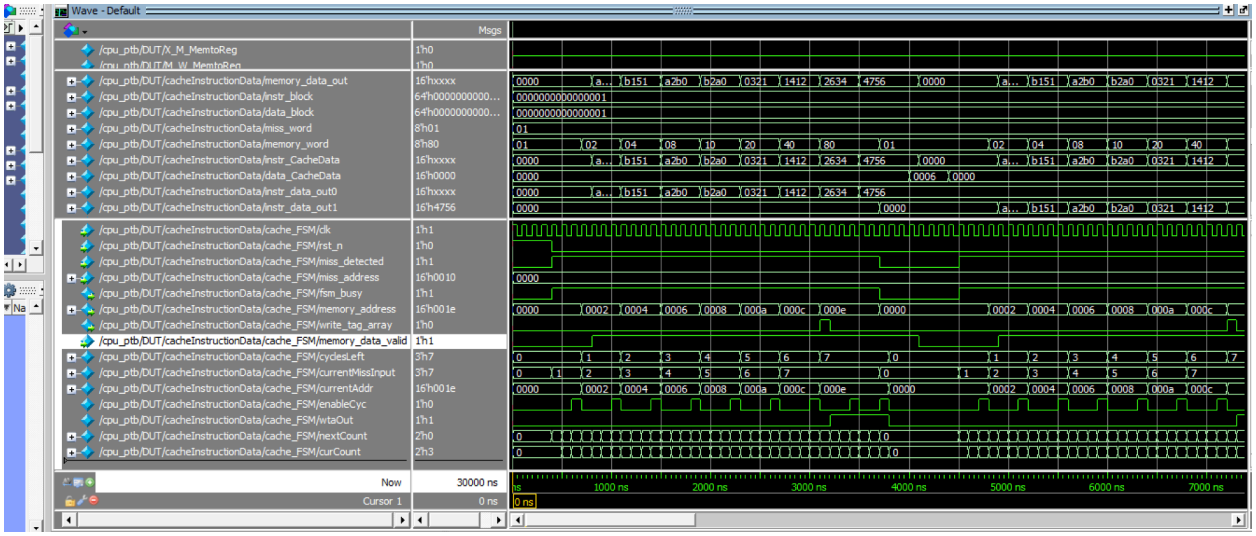
Phase 2 Test 2 Output Should Be Correct

```
SIMLOG:: Cycle           4 PC: 00000000 I: 00000000 R: 0   0 00000000
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle           5 PC: 00000000 I: 00000000 R: 0   0 00000000
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle           6 PC: 00000000 I: 00000000 R: 0   0 00000000
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle           7 PC: 00000000 I: 00000000 R: 1   0 00000000
M: 0 0 00000002 00000002 00000000
SIMLOG:: Cycle           8 PC: 00000000 I: 0000a102 R: 1   1 00000002
M: 0 0 00000002 00000000 00000000
SIMLOG:: Cycle           9 PC: 00000002 I: 0000b100 R: 1   1 00000002
M: 0 0 00000001 00000001 00000000
SIMLOG:: Cycle          10 PC: 00000004 I: 0000a201 R: 1   2 00000001
M: 0 0 00000001 00000000 00000000
SIMLOG:: Cycle          11 PC: 00000006 I: 0000b200 R: 1   2 00000001
M: 0 0 00000004 00000004 00000000
SIMLOG:: Cycle          12 PC: 00000008 I: 0000a604 R: 1   6 00000004
M: 0 0 00000004 00000000 00000000
SIMLOG:: Cycle          13 PC: 0000000a I: 0000b600 R: 1   6 00000004
M: 0 0 00000001 00000001 00000000
SIMLOG:: Cycle          14 PC: 0000000c I: 00001112 R: 1   1 00000001
```

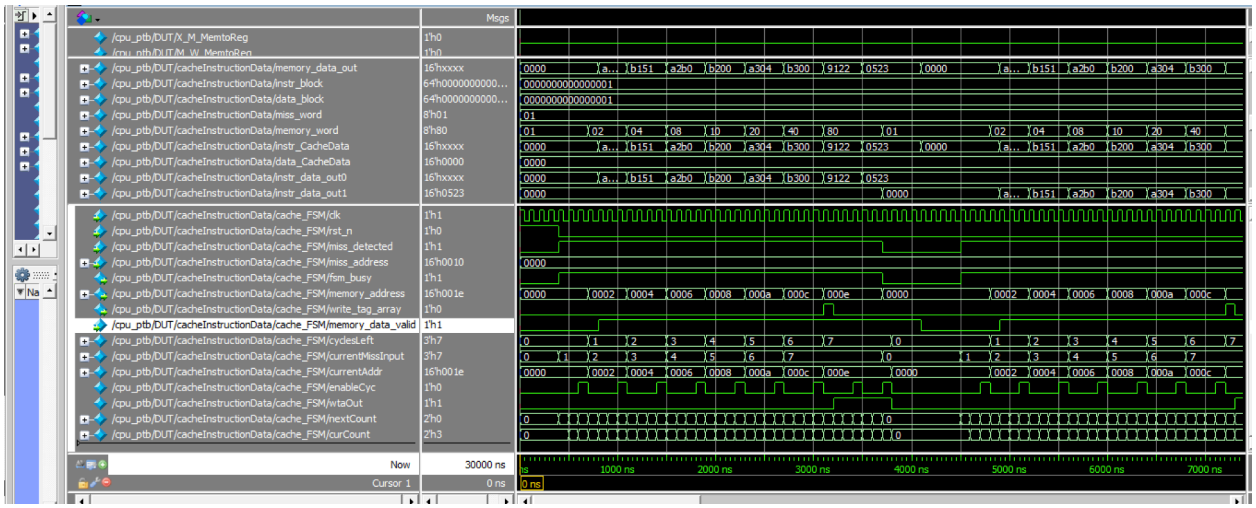
```
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle          15 PC: 0000000e I: 0000e500 R: 1   5 00000010
M: 0 0 00000000 00000002 00000000
SIMLOG:: Cycle          16 PC: 00000010 I: 0000c202 R: 0   2 00000000
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle          17 PC: 00000000 I: 00000000 R: 1   0 00000000
M: 0 0 00000004 00000000 00000000
SIMLOG:: Cycle          18 PC: 00000012 I: 0000de60 R: 0  14 00000004
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle          19 PC: 00000000 I: 00000000 R: 1   0 00000000
M: 0 0 00000000 00000000 00000000
SIMLOG:: Cycle          20 PC: 00000014 I: 0000f000 R: 0   0 00000000
M: 0 0 00000000 00000000 00000000
SIMLOG:: Processor halted

SIMLOG:: sim_cycles          20

SIMLOG:: inst_count          12

REG:  0 VALUE: 0x0000
REG:  1 VALUE: 0x0002
REG:  1 VALUE: 0x0002
REG:  2 VALUE: 0x0001
REG:  2 VALUE: 0x0001
REG:  6 VALUE: 0x0004
REG:  6 VALUE: 0x0004
REG:  1 VALUE: 0x0001
REG:  5 VALUE: 0x0010
REG:  0 VALUE: 0x0000
REG:  0 VALUE: 0x0000
```

Phase 2 Test 3 Result Should Not be Correct Due to Faulty Branch

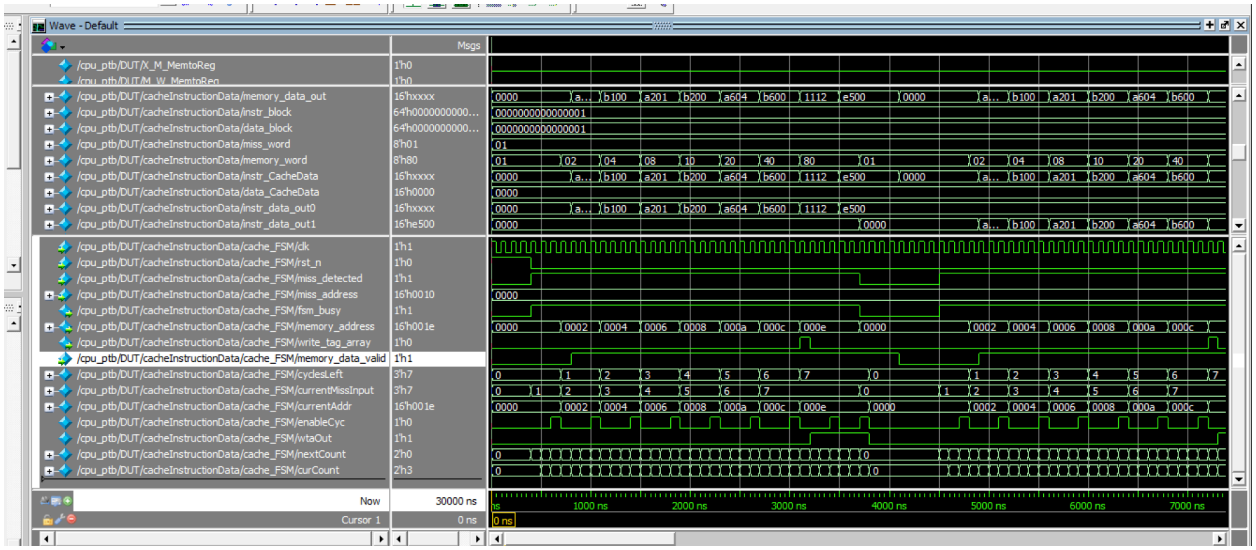
Phase 3 Results and Waveforms:



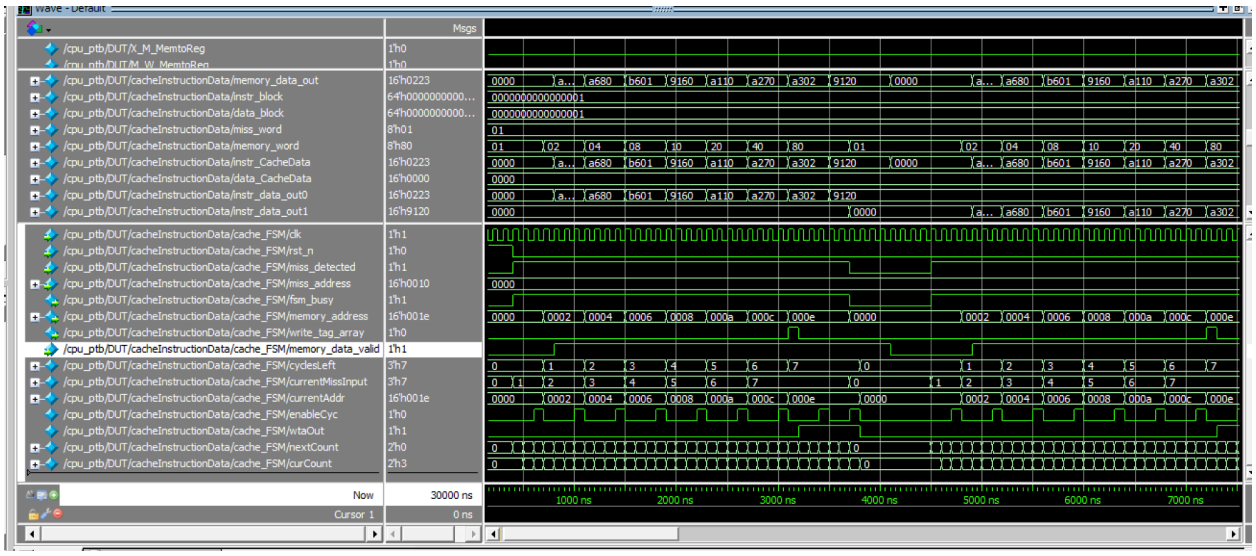
Phase 3 Test 1 Waveform



Phase 3 Test 2 Waveform



Phase 3 Test 3 Waveform



Phase 3 Test 4 Waveform

Phase 3 Extra Credit Synthesis: We made a few changes, where one we made sure that all the files that were being synthesized had to be .sv files. The other thing was that we were not successful in using the bash script in the given github and analyzing the contents of the bash script helped to some degree. So currently with synthesis we were only able to synthesize it and retrieve the area.txt and timings.txt

```
****************************************
Report : area
Design : ALU
```

```
Version: U-2022.12-SP4
Date   : Fri May  3 20:50:14 2024
****************************************

Library(s) Used:

    saed32lvt_tt0p85v25c (File:
/cae/apps/data/saed32_edk-2022/lib/stdcell_lvt/db_nldm/saed32lvt_tt0p
85v25c.db)

Number of ports:                           56
Number of nets:                           692
Number of cells:                          655
Number of combinational cells:            655
Number of sequential cells:                 0
Number of macros/black boxes:               0
Number of buf/inv:                        113
Number of references:                      40

Combinational area:               1652.190145
Buf/Inv area:                      173.072063
Noncombinational area:               0.000000
Macro/Black Box area:                0.000000
Net Interconnect area:             385.076424

Total cell area:                  1652.190145
Total area:                       2037.266568
1
```

Area.txt

```
****************************************
Report : timing
        -path full
        -delay max
        -max_paths 1
Design : ALU
Version: U-2022.12-SP4
Date   : Fri May  3 20:50:14 2024
```

```
****************************************

Operating Conditions: tt0p85v25c   Library: saed32lvt_tt0p85v25c
Wire Load Model Mode: enclosed

  Startpoint: opcode[0] (input port)
  Endpoint: nvz_flags[0]
            (output port)
  Path Group: (none)
  Path Type: max

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  ALU                16000                 saed32lvt_tt0p85v25c

  Point                                    Incr       Path
  -----------------------------------------------------------
  input external delay                     0.00       0.00 f
  opcode[0] (in)                           0.02       0.02 f
  U1937/Y (INVX4_LVT)                      0.15       0.18 r
  U2469/Y (MUX21X1_LVT)                    0.15       0.32 f
  U2468/Y (AO21X1_LVT)                     0.04       0.37 f
  U2463/Y (AO21X1_LVT)                     0.06       0.42 f
  U2459/Y (AO22X1_LVT)                     0.06       0.48 f
  U2456/Y (AO21X1_LVT)                     0.06       0.54 f
  U2451/Y (AO21X1_LVT)                     0.06       0.60 f
  U2446/Y (AO21X1_LVT)                     0.06       0.66 f
  U2441/Y (AO21X1_LVT)                     0.06       0.71 f
  U2436/Y (AO21X1_LVT)                     0.06       0.77 f
  U2431/Y (AO21X1_LVT)                     0.06       0.83 f
  U2428/Y (AO21X1_LVT)                     0.06       0.89 f
  U2423/Y (AO21X1_LVT)                     0.06       0.94 f
  U2418/Y (AO21X1_LVT)                     0.06       1.00 f
  U2409/Y (AO21X1_LVT)                     0.06       1.06 f
  U2396/Y (AOI21X1_LVT)                    0.07       1.13 r
  U2393/Y (XNOR3X1_LVT)                    0.14       1.27 r
  U2392/Y (INVX0_LVT)                      0.03       1.30 f
  U2391/Y (AND3X1_LVT)                     0.06       1.35 f
  U2390/Y (AO21X1_LVT)                     0.04       1.40 f
```

```
  U2389/Y (INVX0_LVT)                      0.03       1.43 r
  U1943/Y (NAND2X1_LVT)                    0.08       1.51 f
  U2384/Y (INVX0_LVT)                      0.06       1.57 r
  U2380/Y (NAND4X0_LVT)                    0.04       1.61 f
  U2370/Y (OR4X1_LVT)                      0.09       1.70 f
  U2314/Y (AO21X1_LVT)                     0.05       1.75 f
  U2059/Y (NAND4X0_LVT)                    0.04       1.78 r
  U55/Y (AO22X1_LVT)                       0.26       2.04 r
  nvz_flags[0] (out)                       1.11       3.15 r
  data arrival time                                   3.15
  ----------------------------------------------------------
  (Path is unconstrained)


1
```

Timings.txt