Computer Architecture: Lab 0

Student 1: Steven Howell **CWID:** A20236687

Student 2: Seth **CWID:** A20271944

•Section 1: Introduction – The lab was a review of Digital Logic Design concepts, specifically a review of how finite state machines work and how to implement registrars. The finite state machine was built for us and we used it to test the setup of ModelSim and its commands. The register refreshed us on how to implement System Verilog and test benches. We had to set up a register with 32 registrars numbered 0 to 31 that stored 32 bits. These locations were read and written with a 5-bit input state and the data channel was 32-bits. Register zero was automatically filed with all zeros and could not be overwritten. The register can only be written if the write enable is set to 1 and the write location is set. Then the ports can be read based on the read value.

•Section 2: Baseline Design - We had to set up a register that stored 32 registrars numbered 0 to 31 that had 32 bits.. These locations were read and written to with a 5-bit input and the data channel was 32-bits. Register zero was automatically filed with all zeros and could not be overwritten. The register can only be written if the write enable is set to 1 and if the location is set. Then the ports can be read based on the read value.

•Section 3: Detailed Design - We established a register that would store 32 registrars numbered from 0 to 31. Each registrar is capable of storing 32 bits. These locations were read and written to with a 5-bit input and the data channel was 32-bits(Each location had a read and write data channel that was able to go up to 5 inputs. Register zero was automatically filed with all zeros and could not be overwritten. The register can

only be written if the write enable is set to 1 and the write location is set. Then the ports can be read based on the read value.

- Section 4: Testing Strategy - Our testing strategy was to display the write enable read 1 and read 2 the write address and the wide data port as the inputs and the register 0 and register 1 as the outputs. We turned on the write enable then wrote data to multiple different register addresses then turned off the write enable. Next, we read and confirmed that the data was properly stored and that the two registrars were separate from each other. Next, we took that data and checked the clock cycles to ensure that it switched on the rising clock edge. After that, we reran the test and attempted to break the registrar by placing data in the 0 port and reading it. Once everything was confirmed to work we saved the test output.
- Section 5: Evaluation - Our code ultimately worked when given an input. One of the hardest parts about this lab is to make the register read file work combinatorially with the register. There were issues with setting the register value to zero since under the right combination of write commands you could override the register value. We added an extra conditional to the write statement preventing registar zero to be written too outside of the initial write that sets it to 32’b0. The cycle states worked properly overall, with it skipping 2 lines due to cycle timing. This lab helped us understand how registrars work and how to implement and utilize registrars as a method of information storage and retrieval.