CompArch: Lab 2

Brenna Manning, William Saulnier, Ziyu (Selina) Wang

11-08-15

1 Input Conditioner

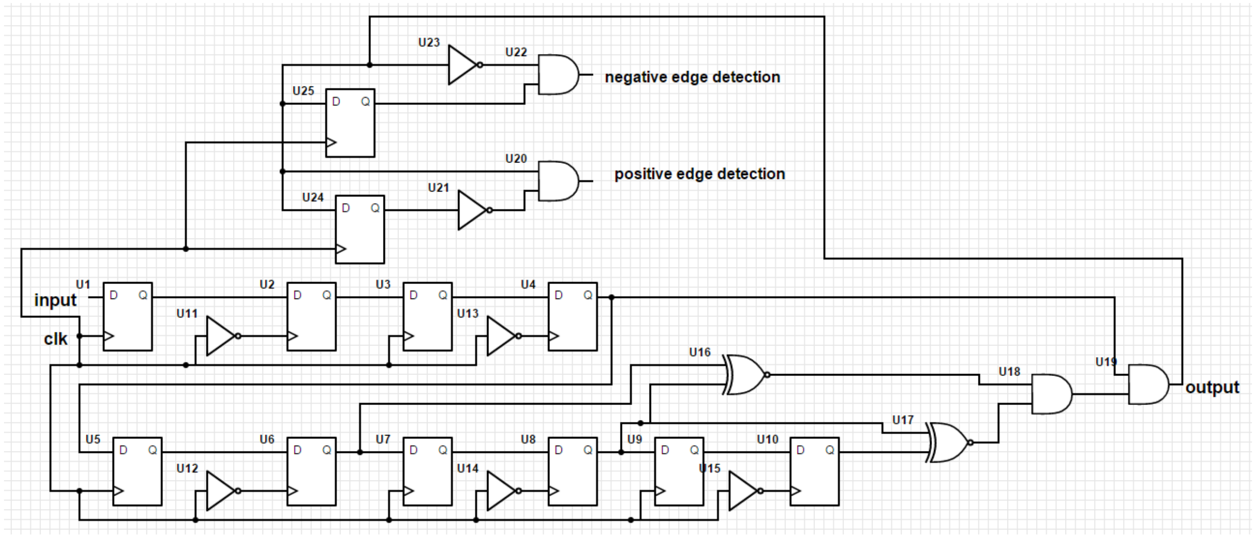

Figure 1: Structural circuit for the input conditioner

For a system with no delay, we would expect that a system with a waittime of 10 would change it's output if a value is sustained for a total of 10 clock cycles. This means that our maximum number of clock cycles a signal could be sustained for is 9. In an ideal world, to fully maximize this number, the glitch would turn off right before the next high clock edge. This would make the total maximum length input glitch sustainable by the system to be equal to 199 nanoseconds.

2 Shift Register

The goal with our test bench was to identify perticular failure modes that we belived to be common to the Shift Register. We first check to ensure that data can properly load into the system. Then, we proceed to check to make sure the serial loading process works, and we can turn off the paralell loading system. We also verify that the system can hold its state properly. This short bench is able to verify all of the behaviors of the system in a quick way.

3 Test Strategy

Our goal with the testing strategy for the full SPI system was two fold. One, to comprehensively test for any faults, and to ensure that the entire memory could be addressed and was functioning. By using a for loop, we're able to loop through all 2^7 memory addresses allowed by the system, writing to each their memory address, and then reading off the result. Concluding the test bench are three random tests. This catches any and all addressing faults as well as writing faults, on top of being relatively quick to execute in software.

4 Fault Injection

Our injectable failure mode is that the most significant bit of the address bus is permanently 1. This could occur if this D flip-flop that stores that address bit was broken during manufacturing. We simulate this by modifying our latch to accept the fault_pin signal and if it's true tying that most significant bit high. This can be identified during test by noticing that adjacent memory locations have been merged. For example, writing to address 0011000 affects address 1011000 as well.

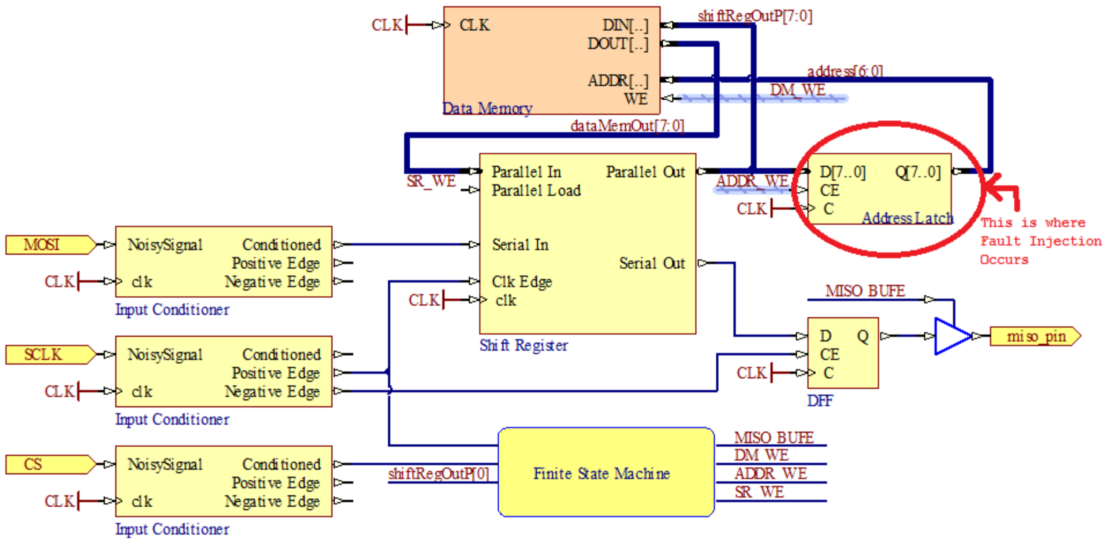

Figure 2: Fault Injection SPI Memory Schematic

5 Work Plan Reflection

The first half of this lab progressed very well. In fact, we came in under time in all of the coding portions of the lab. Total time spent on the actual coding writing came in under 3-4 hours. However, where we ran into most of our issues was getting the hardware to work. Where we budgeted much less time (about an hour), we ended up spending almost five times that. It's a clear indication that we should budget more time for resolving hardware issues that are unrelated to the actual work for the project, as it appears that this step is where the primary sink of time occurs, despite the fact that the learning during that part is much lower than prior. Instead, we decided to refocus on writing good Verilog test benches rather than worrying about hardware.