3DQ5 Lab 4 Exercise Report

Exercise 1

In this exercise, we were required to modify some code from experiment 3. This exercise involved invoking the DP-Rams to preform a series of arithmetic within the W and X 8-bit signed integer arrays. The first thing we did was set all the write enables to low due to them being active high. We started with the write enables to be off in the read state. We asserted them so that on transition to the write state they would be high making this operation readily available within this state. Within the write state we updated the read addresses and put all the write enables low and transitioned back o the read state. This process was repeated to handle the logic for the rest of the equations. The RAMS were true Dual Port so we could read/write simultaneously for faster operations than what a Single Port RAM would provide us with functionality of only being to do these operations one at a time. On the last write state we reset the Read address and flips the state back to IDLE.

Exercise 2

This exercise modifying the built in self test engine from experiment 4. The functionality was split into 2 sessions. The first would involve the initial 128k locations to be verified by writing the value of the 16 LSB of the address. The second would be the same but for the last 128k locations, but with the difference of when writing and reading data, the lines must change in decreasing order. We started by writing to the first 128k locations and then switched to the read cycle to read them. Then we switched to writing them but in a top-down order (256k -1 to 128k). In each read cycle we checked to see if the data being read was equivalent to the expected data, and if it was not we asserted a mismatch flag. To ensure that every location was written/read to exactly once we added a read and write increment, and when we counted down, we changed read increment to be +3, so it would stop reading 3 addresses higher then 128k. A problem we faced was, when counting up, it would go 2 past 128k, so by setting this limit to 3, we ensured to have no overlap. The write increment flag was also set at a limit to 1, to ensure that this operation stopped on time as well for no overlap. We also had a mode flag, to track if we were counting up or down which was altered at the end of states. The BIST finish flag would assert at the end to return to idle if no mismatches were discovered.