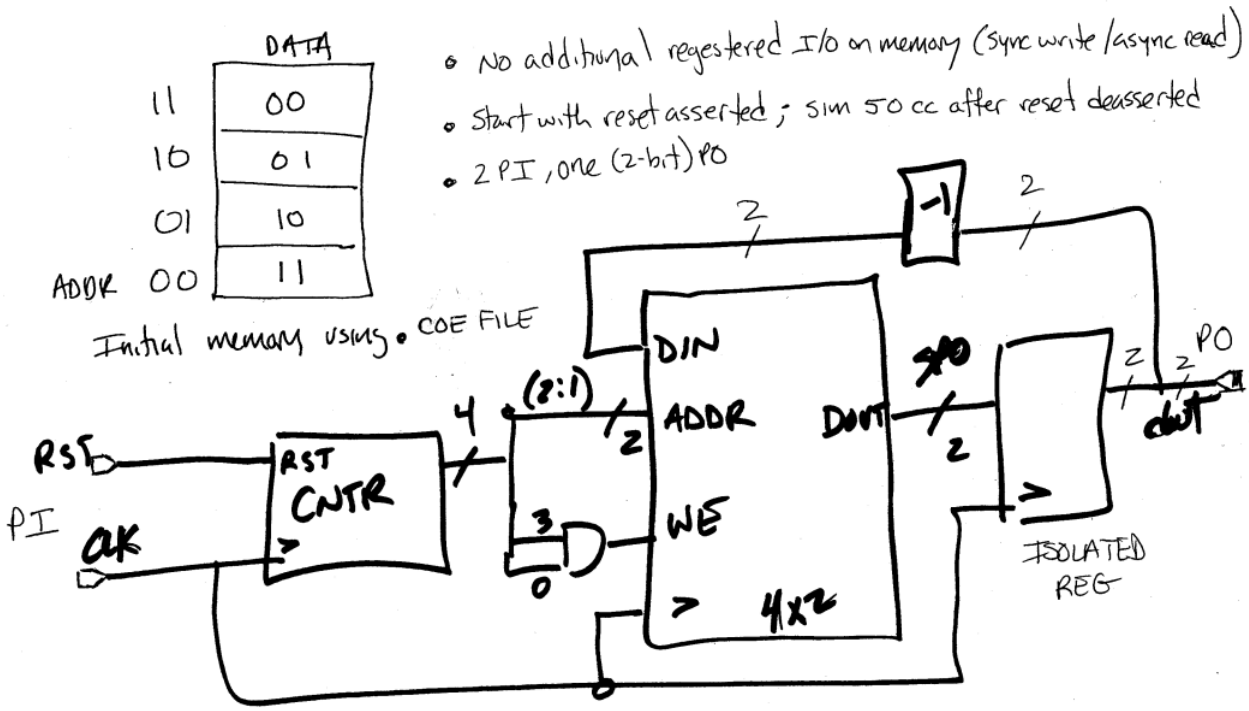
Cory Holt

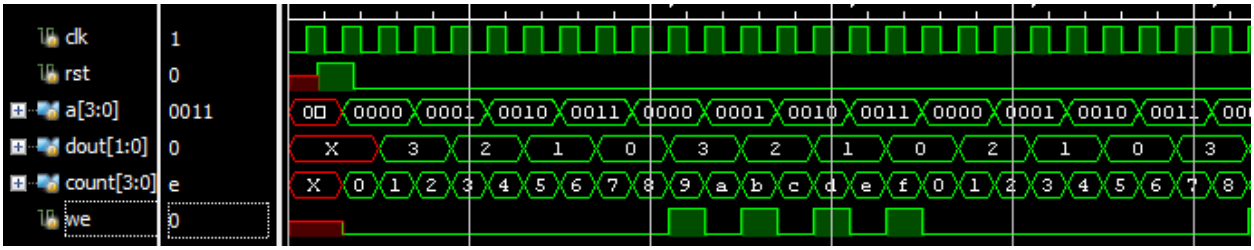
ECE 440

HW5

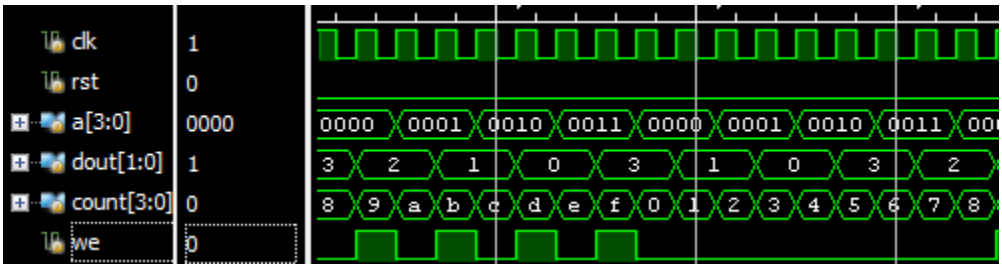
2-12-20

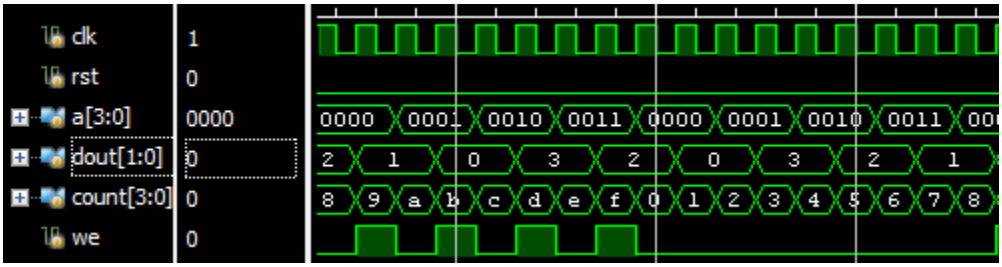


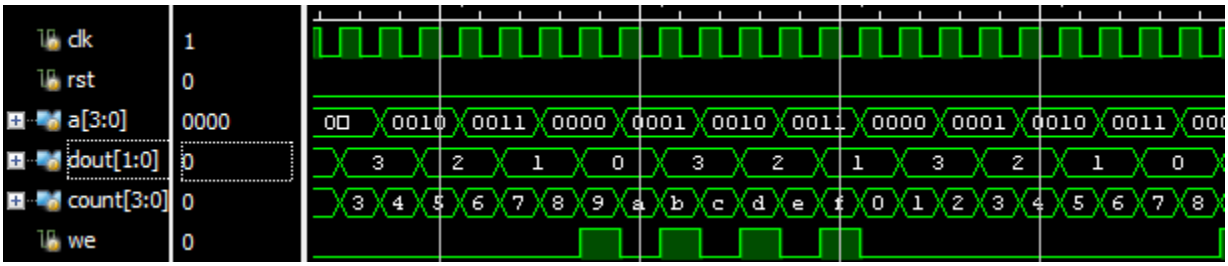
This “amusing little circuit” is centered around the block memory. The main output is looped back to the input through a subtractor, so it modifies its own contents. The key to *when* the contents are modified lies in the WE signal. The counter is a 4 bit counter, so it has 16 values. The WE signal only looks at the MSB and the LSB. This tells us that it will be active for 4 out of the 16 counts, as it is using 2 bits. Looking at the binary values, we see that WE will be active for b1001, b1011, b1101, and b1111, or 9, 11, 13, and 15 in decimal. The address uses the middle two bits, and looking at those bits, we see that the address increments fairly normally, from b00 to b11. Since the address uses the middle bits, the output is held for two bits. Due to the isolated register, the outputs are delayed by a clock cycle. What I presume will happen is that all the data contents will get subtracted by one (from the subtractor), as the address increments normally, and the dout is delayed a cycle. Looking at the memory as a block, as these are 2-bit numbers, it will appear as if they are shifting around. So after the first sequence of 4 WE signals, the memory will contain b10, b01, b00, and b11, in that order. To verify this hypothesis, I ran a simulation.



As expected, the change is only apparent after the write sequence finishes. The sequence of digits appearing on dout changes from 3,2,1,0 to 2,1,0,3. Subtracting from zero on a 2 bit number rolls the value back around to 3. Observing the next few write sequences shows that this pattern continues.







By the end of four sequences, the memory pattern is back to the initial condition of 3,2,1,0.