1. Design a 3-bit Flash ADC converter which encodes an analog input voltage from 0 to 1.6 Volts into eight possible states encoded on 3 output lines ($d = d_2d_1d_0$) as a binary number. Show the resistor chain with reference voltages, relative resistance values, and the necessary comparators. For the priority encoder, you must specify the logic necessary to obtain $d$, but you do not need to draw out the gates. Make any reasonable assumptions or design choices that are needed and explain the choices you make.

2. Draw the state diagram for a 3-bit successive approximation register as shown to the right. This device takes a single input $D$ from a comparator which should be set HI if the input voltage is greater than the current test value (indicated by the three bits of $Q$) or LO if the input voltage is less-than or equal-to the current test value. At the end of the successive approximation cycle, a conversion complete CC line should go HI. How many states does your SAR state machine have? Make sure it can arrive at all possible states with CC set HI. Hint: The SAR should start with $Q = 011$ on the first cycle. You do not need to show the truth table for the combinatoric logic, just the state diagram. Also, assume that your SAR is reset by some asynchronous reset line (so that you don’t need to worry about your logic synchronously resetting itself after the conversion is complete).

3. Design a control circuit for a simple vending machine. The product costs 15 cents ($0.15). Either dimes ($0.10) or nickels ($0.05) may be input. Design the control circuit as a Moore-style state machine. Each valid input coin represents a “clock” cycle. An input bit $C$ determines whether the coin was a nickel ($C = 0$) or a dime ($C = 1$). When 15 cents have been input, your circuit should set an output bit $V$ HIGH, which vend the product and collects the coins. A 2nd output bit $R$ refunds all input coins if too much money has been entered. For simplicity, assume there is an external RESET triggered by the completion of either the coin return or vend operation that will take the state machine back to a starting state. In other words, your state machine does not have to synchronously reset itself.

Indicate the input and output lines into and out of your state machine. Draw a complete state diagram and specify the output values for each state. Specify clearly the combinatoric logic needed, both for the state transitions and for the output logic. You do not need to draw the final circuit.

4. Design a traffic signal controller for a farm road crossing a highway. Detectors $C$ detect cars waiting on the farm road. With no cars on the farm road, the highway lights stay green (output $HG$). If a car arrives on the farm road, the highway lights go from green to yellow (output $HY$) then change to red (output $HR$) while the farm road light goes green (output $FG$). This light stays green for as long as cars are detected on the farm road, but never longer than some set interval. The farm road
lights then transition through yellow to red (outputs $FY, FR$) and the highway lights return to green. If more cars are waiting on the farm road, the highway green is also maintained for the same set interval.

To make your life easier, you have an interval timer that will receive a start signal ($ST$) will first return a signal after a short time ($TS$) and second return a signal after a long time ($TL$). $TS$ can be used to time the duration of the yellow lights, while $TL$ can be used to time the duration of the green lights.

Write down and identify all input and output signals to your state machine and identify the unique states. Hint: this can be done with only four unique states. Draw out a complete state diagram with transitions labeled clearly using the input names I have given above. Make sure that for each state you have specified all needed transitions. You may want to describe your thinking for each state in words as well. Neatness counts!

Write down the truth table logic giving the next state as a function of the current state and the input lines. Write down a separate table giving the state machine outputs for each of the four unique states.

Solve the combinatoric logic for the output signals as a function of the internal state bits. There are too many inputs to easily be able to solve the state transition logic, so don’t waste your time on this.