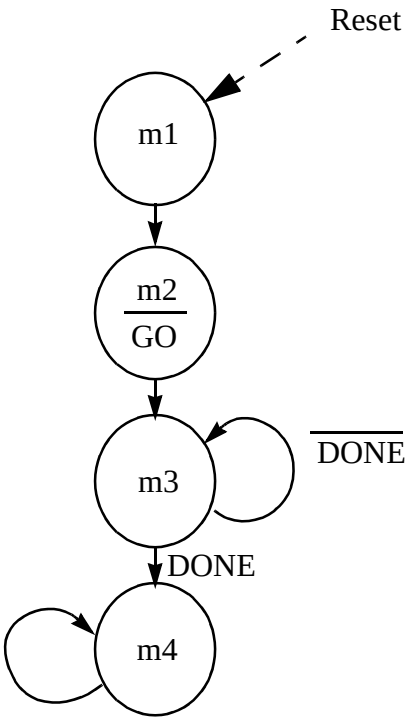
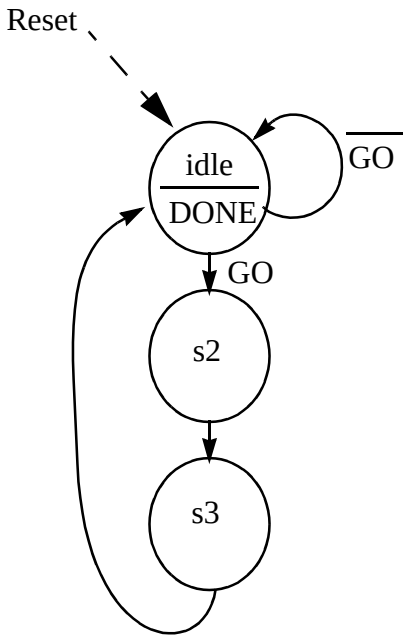
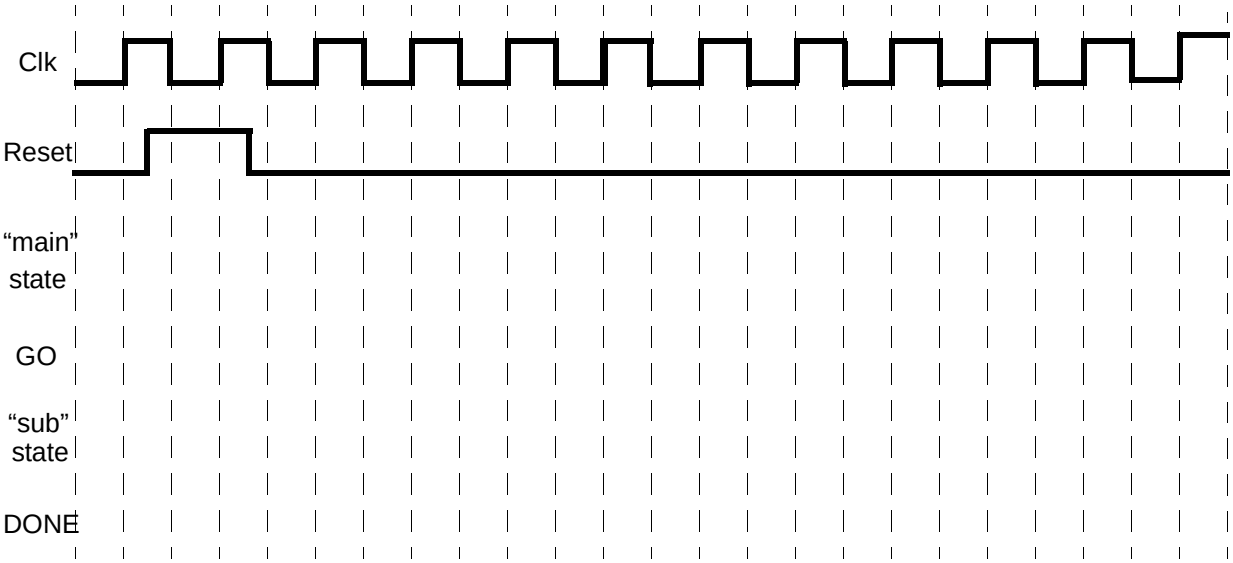
Homework 3 - Digital Systems Design

Problem 1

The following two Moore state machines interact with one another. Complete the timing diagram below for the two state machines. Assume the flip-flops trigger on the positive edge and there exists a small amount of delay for each transition. If an output is not shown on the state diagram, assume it is deasserted. Show the timing information for GO and DONE, however just write the state name for “main state” and “sub state”.

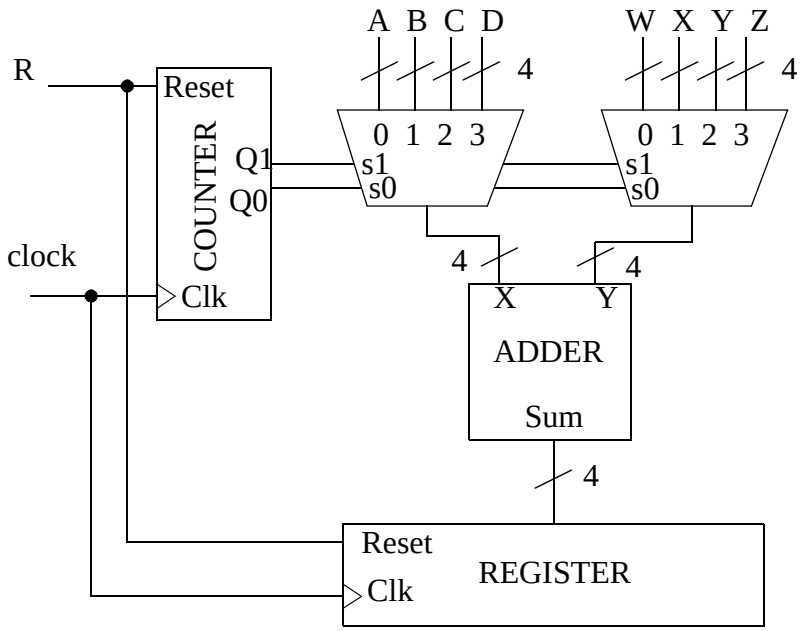






Problem 2

Develop a muti-module verilog description for the following schematic. Describe each unique block in the system using a unique .v file. Then develop a top-level module that calls each of the lower level modules and connects them appropriately. Reuse your verilog descriptions whenever possible. Design a test bench that will verify your design. Be sure to turn in a printout of each of your .v files, your testbench file, and an annotated waveform proving that your solution works correctly.



Problem 3

Write the verilog description of the state diagram for the lock problem in Homework 1. Use the Meely solution and encode your states using a binary representation. (Please start with our solution posted on the web site rather than your own solution to the state diagram).Verify that your description is correct using a testbench file.

Problem 4

A system has an 8-bit input that changes at the rising edge of every clock cycle. The input stream can contain either 8-bit instructions or 8-bit numbers (data), depending on the previous instruction. The goal of this problem is to design a circuit that processes this input stream and correctly outputs the processed data values.

- **Inputs:**
  - clk – the system clock.
  - R – the system reset. This should reset any stored information to zero. (Note: The input stream will also reset when R is asserted, guaranteeing that the 8-bit value provided at the next rising edge is an instruction, not data. You do not have to provide this function).
  - I [7:0] – the 8-bit input stream (see below).

- **Outputs:**
  - Out [7:0] – the 8-bit output data values. These values should come directly from the internal "A" register (see below).

The instructions in the input stream can be any of the following:

ID	Value	Description
LdA	'hFF	Loads the next value in the input stream to an internal register labeled "A"
LdB	'hFE	Loads the next value in the input stream to an internal register labeled "B"
ClA	'hFD	Clears the internal "A" register
ClB	'hFC	Clears the internal "B" register
AddA	'hFB	Adds the contents of the "A" and "B" registers and places the result into the "A" register
SubA	'hFA	Subtracts the contents of the "A" register from the "B" register and places the result into the "A" register
AndA	'hF9	Places the logical AND of the "A" and "B" registers into the "A" register
OrA	'hF8	Places the logical OR of the "A" and "B" registers into the "A" register
NotA	'hF7	Inverts the contents of the "A" register
NotB	'hF6	Inverts the contents of the "B" register

For example, the input stream might be:

…, FC FF 03 F6 FB

This stream would translate to the following instructions:

…, ClB, LdA, 'h03, NotB, AddA

After the first clock cycle, register “B” should be zero.
After the third clock cycle, register “A” should be ‘h03.
After the fourth clock cycle, register “B” should be ‘hFF.
After the fifth clock cycle, register “A” should be ‘h02.

(a) Step through the following input stream, like the example above, showing the contents of both registers at the end of each clock cycle. Assume that R was asserted in the clock period prior to the beginning of the stream.

R asserted, then: FF FF FE FF FD FF FC FB F7 F9 …

(b) One way to implement this circuit is with a control unit (CU) that communicates with a data unit (DU). List the responsibilities of each unit for accomplishing the overall task. *Explain why you chose to place each responsibility in each unit*.

(c) Draw a state diagram for the CU and a schematic using LSI blocks for the DU.

(d) Implement the CU and the DU as separate Verilog modules in separate files. Attach a copy of your code.

(e) Write a test bench for your implementation. Attach a copy of the testbench and *explain why the testbench is adequate for testing your design.*

(f) Run your testbench and produce waveform output that demonstrates a successful test sequence. Annotate your plot to show what is happening.