PRACTICE PROBLEMS SET 3

Register Transfer Level Design

1. What is the difference between a FSM and a high-level FSM?

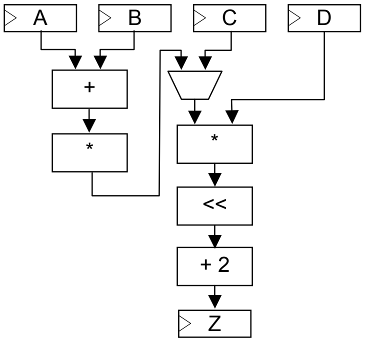
2. Using the RTL design method, create a circuit that calculates the sum of M to N. Implement this circuit using the algorithm provided.

   a) Convert the following C code to a high-level state machine
   b) Create a datapath
   c) Connect the datapath to the controller
   d) Derive the controller's FSM

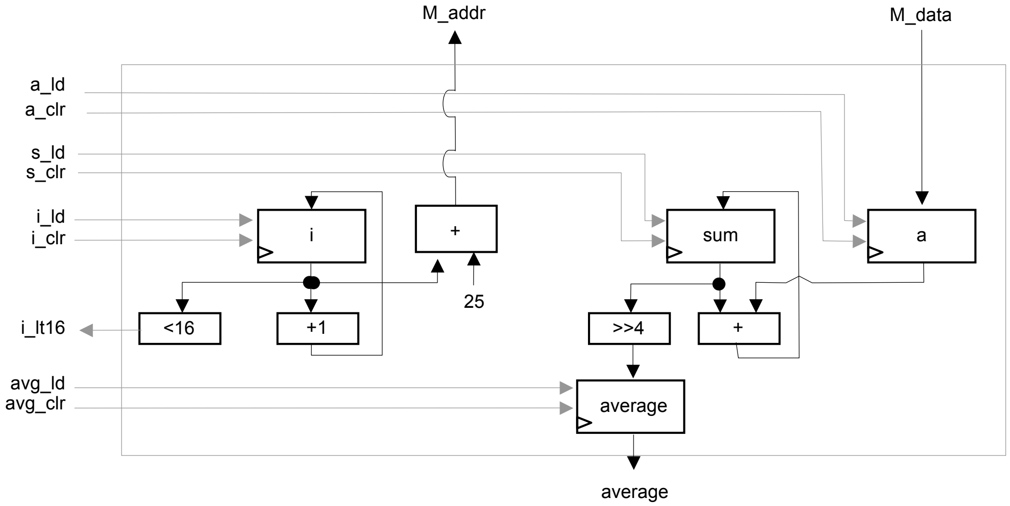
```
Inputs: byte M, byte N, bit start
Output: byte result, bit busy

CIRCUIT_X:
result = 0;

while(1){
   busy = 0;
   while(!start);

   busy = 1;
   result = 0;
   for( int y=M; y<=N; y++ ){
      result = result + y;
   }
   return result;
}
```

3. Assuming muxes have a delay of 5 ns, adders and shifters have 10 ns delay, multipliers have 20 ns delay, and wires have 2 ns delay,

   a) What is the critical path of the datapath and it's corresponding delay?
   b) What is maximum system clock frequency given the current datapath?
   c) What modifications can be made to the datapath to increase the system clock frequency?

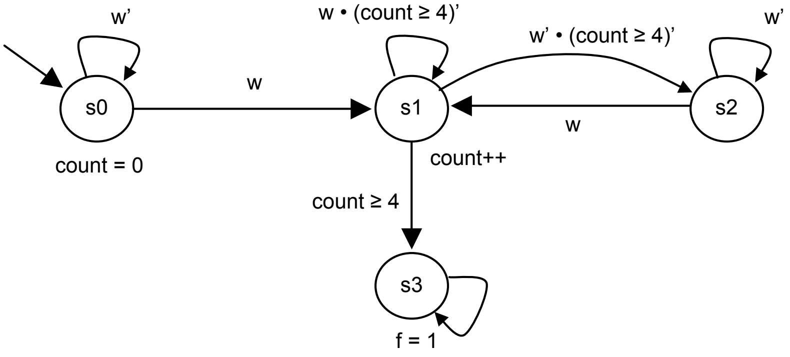


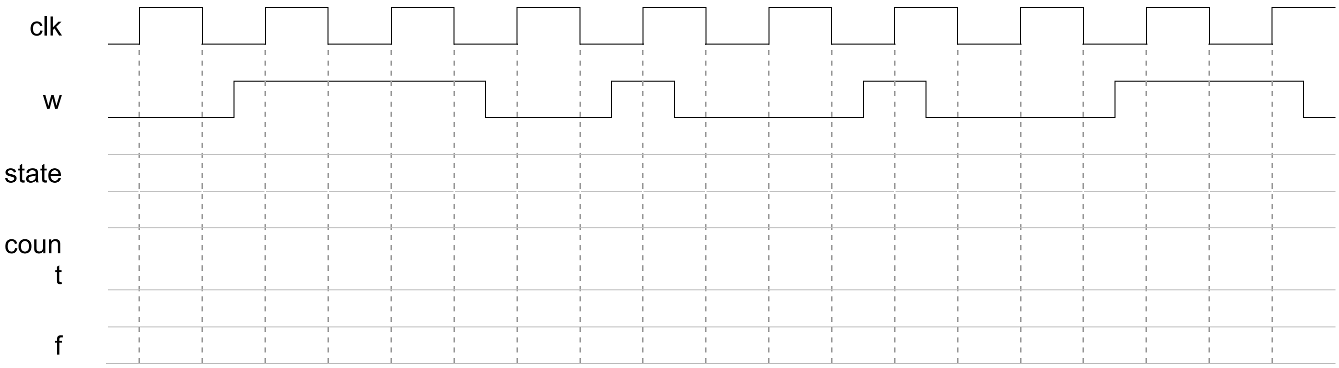
4. Create an FSM that interfaces the datapath shown below. The FSM should use the datapath to compute the average value of the 16 32-bit elements of any array A. Array A is stored in a memory, with the first element at address 25, the second at address 26, and so on. Assume that putting a new value onto the address lines M_addr causes the memory to almost immediately output the read data on the M_data lines. Ignore the possibility of overflow.



5. Consider the following FSM that detects when four 1's appear on an input stream. Fill in the corresponding timing diagram. Is the functionality correct? If not, explain what's wrong and how can it be fixed.

Inputs: w (bit)
Output: f (bit)
Local Reg: count (3 bits)

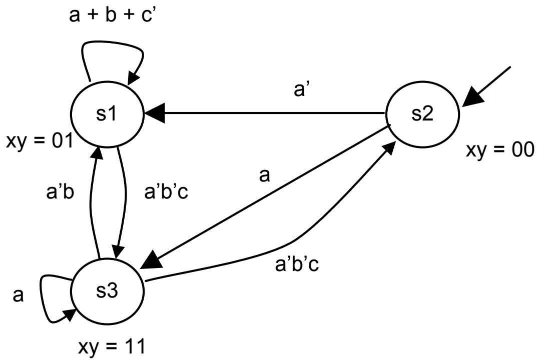




6. Consider the previous FSM that detects when four 1's appear on the input stream. Fill in the timing diagram. Is the functionality correct? If not what's wrong and how can it be fixed?

a) ONLY ONE condition true
b) ONE condition true

Inputs: a, b, c (bit)
Output: x, y (bit)


a + b + c'
a'
s1
s2
xy = 01
xy = 00
a
a'b
a'b'c
a'b'c
a
s3
xy = 11