Professor Fearing EECS150/Problem Set 2 Fall 2013

Due at 10 am, Thu. Sep. 19 (homework box under stairs)

Reading Harris and Harris: 4.1-4.6, 4.8

1. (20 pts) Verilog FSM

```
module fsm1(input clk, input reset, input [1:0] A, output reg q);
  reg [1:0] state, nextstate;
  parameter S0 = 2'b00;
  parameter S1 = 2'b01;
  parameter S2 = 2'b10;
  always @(posedge clk) // State Register
     if (reset) state <= S0;
        else state <= nextstate;
  always @ (*) // Next State Logic
    case (state)
       S0: if (A[1]) nextstate = S1;
           else nextstate = S2;
       S1: if (A[0]) nextstate = S1;
           else nextstate = S0;
       S2: if (& A) nextstate = S2;
           else nextstate = S0;
       default: nextstate = S0;
    endcase
   always @ (*) // Output Logic
      case (state)
        S0: q = 0;
        S1: q = 0;
        S2: if (&A) q = 1;
            else q = 0;
        default: q = 0;
      endcase
endmodule
```

a. Draw the circuit corresponding to this code. (You may use muxes if desired for Next State and Output logic.)

b. Sketch the state transition diagram corresponding to the above code.

c. Let the input to the module `test = {reset, A[1:0]}`. `test` is applied before the rising edge of each `clk` in the sequence $100_2, 000_2, 000_2, 001_2, 011_2, 111_2$. Draw a timing diagram showing `clk, reset, A[1:0], q`, assuming a unit delay for the combinational logic blocks, a unit delay for register output (t_{CKO}), and clock period 10 unit delays.

2. (25 pts) FSM design

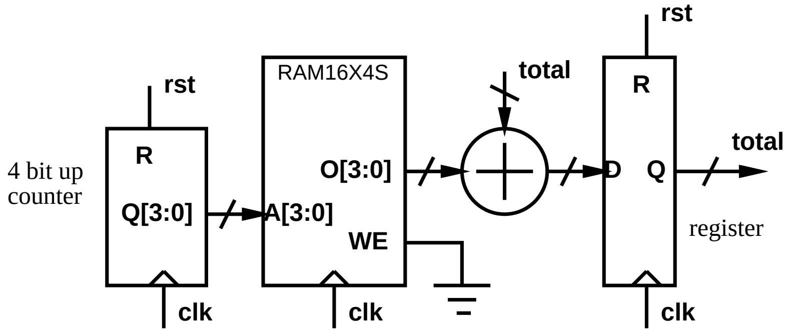
Design a fully-synchronous, loadable, 3 bit, modulo 3 up/down counter. In *up* mode, sequence should be 0,3,6,1, ... In $\overline{up}$ mode, sequence should be 6,3,0,5,2, ... The counter has inputs `clk, reset, up, load, data[2:0]` and output `count`. `reset` (to 0) has priority over `load` which has priority over `count`. On `load == 1` the counter should be loaded with the `data` value on the next rising edge of the clock.

a. Draw the state transition diagram for the counter.

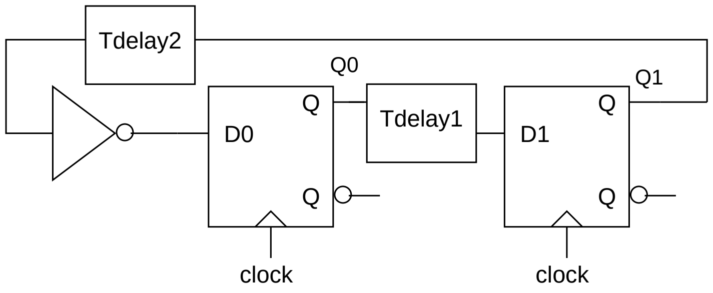
b. Draw a high level block diagram of the counter, with the blocks of state register, and next state CL.

c. Write the behavioral Verilog code that will implement this counter.

3. (35 pts) Using Xilinx Library
The circuit below shows a simple data processor which calculates the summation of the contents of 16 four bit words stored in a memory. (For this problem you can assume that the memory contents have already been initialized). The RAM16X4S is a Xilinx primitive described in *Libraries Guide for HDL Designs,* p. 256. In operation, `rst` synchronously sets the register and counter to zero. The summation operation runs continuously. The module has output `total[7:0]` and output `done` which is True when the counter reaches 1111_2. Write Verilog code for `module summer(...)`. (Note that part of the module will need to use structural Verilog to use the built-in primitive.)



4. Timing Analysis (20 pts)
For the circuit below, $T_{setup} = 1.0$ ns min, clock-to-Q delay $T_{CKO} = 1.5$ ns, $T_{delay1} = 1.0$ ns, $T_{delay2} = 0.5ns$, and inverter propagation $t_p = 0.8$ ns.
a. Sketch a timing diagram, showing $D_0, Q_0, D_1, Q_1, clock$ for 4 clock periods, assuming a 100 MHz clock.
b. Determine the minimum clock period for this circuit.


Tdelay2
Q0
Q1
Q
D0
Q
Tdelay1
D1
Q
Q
clock
clock