

CSE370 Final Exam (11 December 2006)

Please read through the entire examination first! This exam was designed to be completed in 110 minutes (one hour and 50 minutes) and, hopefully, this estimate will be reasonable.

There are 3 problems for a total of 100 points. The point value of each problem is indicated in the table below. Each problem and sub-problem is on a separate sheet of paper. Write your answer neatly in the space provided. If you need more space (you shouldn't), you can write on the back of the sheet where the question is posed, but please make sure that you indicate clearly the problem to which the comments apply. Do NOT use any other paper to hand in your answers. If you have difficulty with part of a problem, move on to the next one. They are mostly independent of each other.

The exam is CLOSED book and CLOSED notes. Please do not ask or provide anything to anyone else in the class during the exam. Make sure to ask clarification questions early so that both you and the others may benefit as much as possible from the answers.

Good luck and have a great holiday break.

Name:

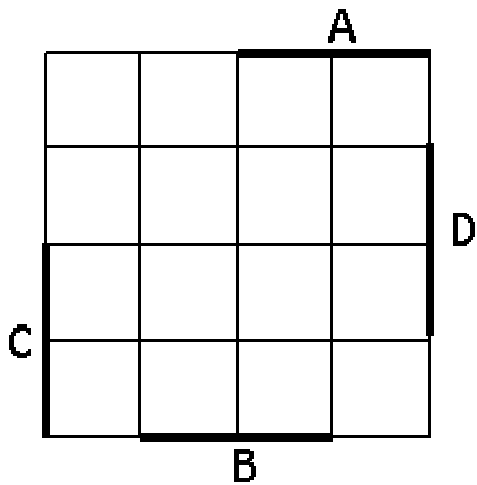
ID#:

Problem	Max Score	Score
1	25	
2	50	
3	25	
TOTAL	100	

1. Combinational Logic (25 points)

(a – 5 pts) Map the following function on the Karnaugh-map provided below and re-write it in canonical sum-of-products form. Please use minterm notation (e.g., m_3) instead of the algebraic form (e.g., $A'B'CD$).

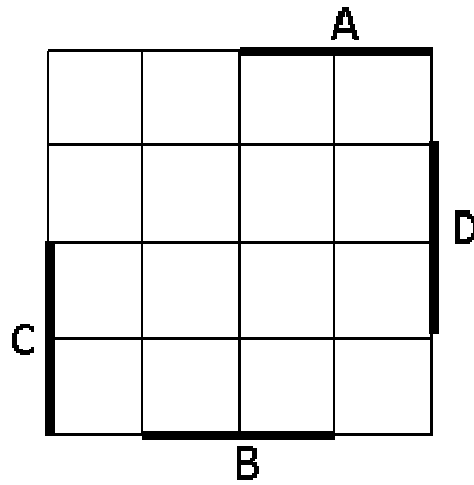
$$Z = ((A' \text{ xor } B) + C) (CD)'$$



(b – 5 pts) Re-write the function in minimized sum-of-products form and highlight each term that represents an essential prime implicant by underlining it. Make sure to circle each term on the K-map above.

$$Z =$$

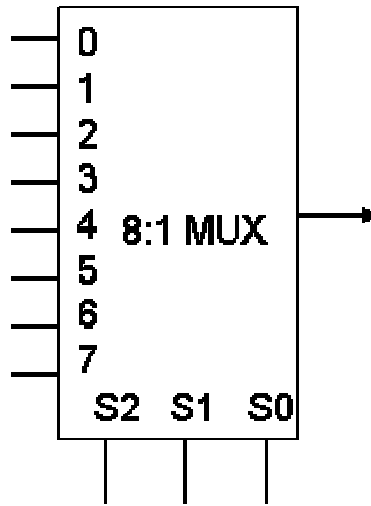
(c – 5 pts) Add don't cares to the K-map you derived in part (a) for the cells covered by $A'C$. Redraw your new K-map below, circle all prime implicants, and then list the essential and non-essential prime implicants for this new function.



Essential:

Non-essential:

(d – 10pts) Implement the function of part (c) using an 8:1 multiplexer (provided below) and at MOST one inverter. S2 is the most significant bit of the control signals, S0 is the least significant (e.g., S2=1, S1=1, S0=0 selects input #6).



2. Finite State Machines (50 points)

The following Verilog was found among old papers in a dusty drawer of a now defunct dot-com company. Unfortunately, there were no comments in the code.

```
module Mystery (In, Clk, Reset, A, B, Out);
    input      In, Clk, Reset;
    output     A, B, Out;

    reg [5:0]  state;
    reg [5:0]  next_state;
    wire [2:0] count;

    parameter S0   = 6'b000000;
    parameter S1   = 6'b000001;
    parameter S2   = 6'b000010;
    parameter S3a  = 6'b100011;
    parameter S3b  = 6'b001011;
    parameter S4aa = 6'b100100;
    parameter S4ab = 6'b010100;
    parameter S4bb = 6'b001100;

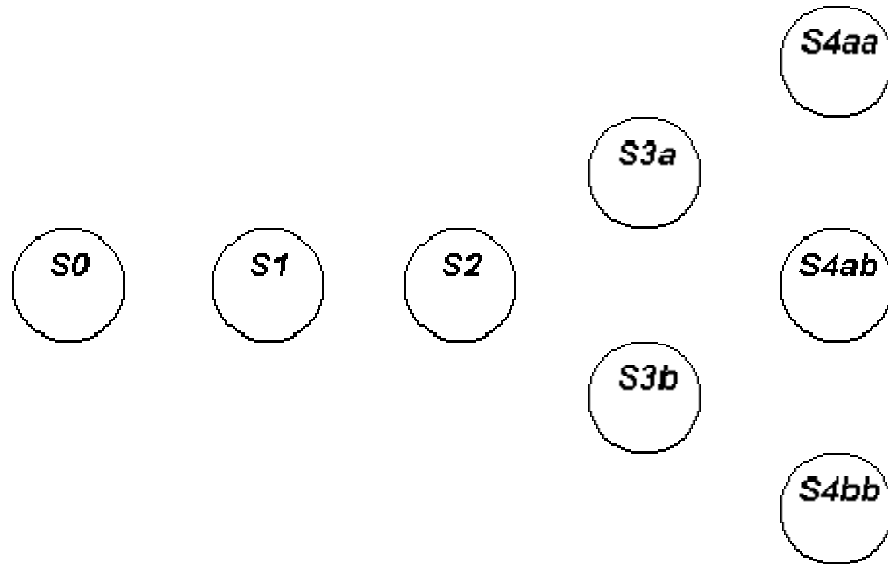
    always @(posedge Clk) begin
        if (Reset) begin state = `S0;          end
        else          begin state = next_state; end
    end

    always @(In or state) begin
        case(state)
            `S0:  next_state = `S1;
            `S1:  next_state = `S2;
            `S2:  if (In) next_state = `S3b; else next_state = `S3a;
            `S3a: if (In) next_state = `S4ab; else next_state = `S4aa;
            `S3b: if (In) next_state = `S4bb; else next_state = `S4ab;
            `S4aa: next_state = `S0;
            `S4ab: next_state = `S0;
            `S4bb: next_state = `S0;
        endcase
    end

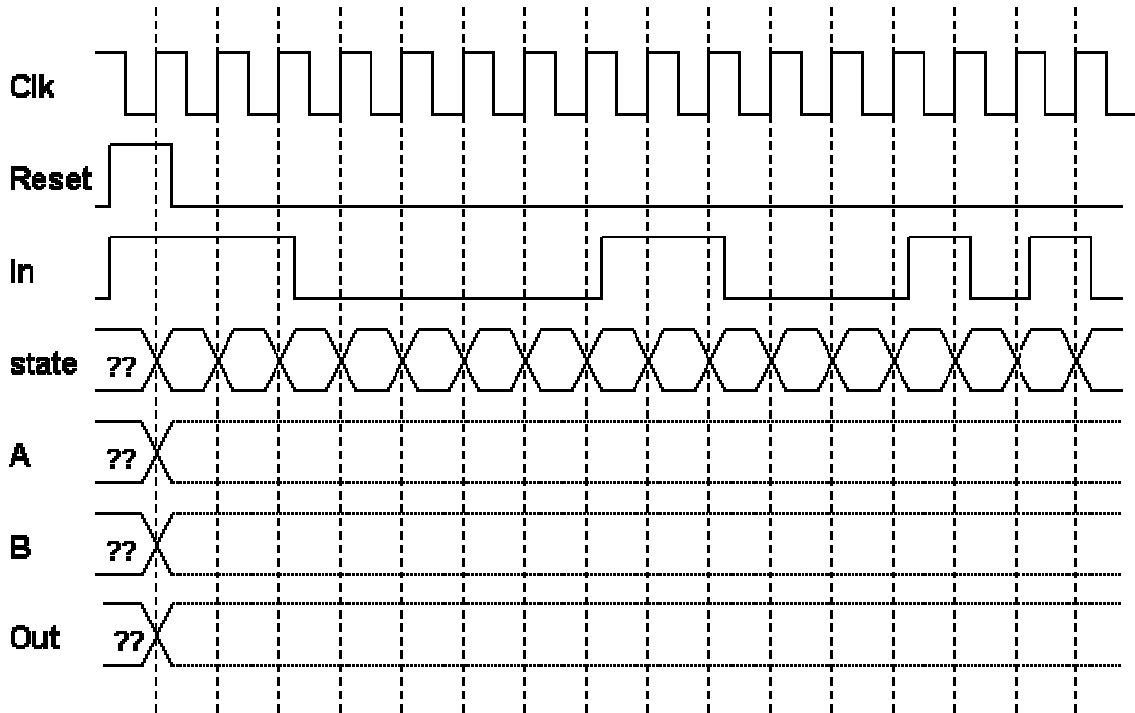
    assign count = state[2:0];
    assign A     = state[5];
    assign Out   = state[4];
    assign B     = state[3];

endmodule
```

(a – 10 pts) What type of state machine is this? Circle: Mealy, Moore, or synchronous-Mealy. Derive its state diagram and clearly label all transitions and the values for the 3 outputs in each state. Use the template below. Clearly label all transitions and outputs.



(b – 15 pts) Simulate the state machine for the following sample input waveforms. You are provided the values for the input signals, Reset and In. Fill in the details for the signals A, B, and Out. Also, please indicate the state the FSM is in for each clock cycle (i.e., write the symbolic name of the state in each cycle). Assume that the FSM is initially in an unknown state. Assume that all FFs are positive edge-triggered.



(c – 15pts) When we look to see how this state machine was actually implemented, we find that there is a 3-bit binary counter in the circuit and 2 individual D flip-flops with some very simple logic around them. We need to figure out if we have an up to date description for this module. The Verilog to describe the actual circuit implementation is provided below.

```
module MysteryImplementation (In, Clk, Reset, A, B, Out);
  input      In, Clk, Reset;
  output     A, B, Out;

  reg [2:0] counter;
  reg      A, B;

  always @(posedge Clk) begin
    if (Reset | counter[2]) counter = 0;
    else counter = counter + 1;
  end

  always @(posedge Clk) begin
    if (Reset | counter[2]) A = 0;
    else A = A | (counter[1] & ~In);
  end

  always @(posedge Clk) begin
    if (Reset | counter[2]) B = 0;
    else B = B | (counter[1] & In);
  end

  assign Out = A & B;

endmodule
```


(c – 15 pts continued) Is this an `MysteryImplementation` a correct implementation of the `Mystery` module? It turns out it is. In fact, `Mystery` and `MysteryImplementation` describe the same state diagram. What is the correspondence of states between them? For each of the states of the `Mystery` module list the states for the sequential elements of the `MysteryImplementation` module – note that there are 3 state elements in the `MysteryImplementation`: the counter (3 bits), the A FF (1 bit), and the B FF (1 bit).

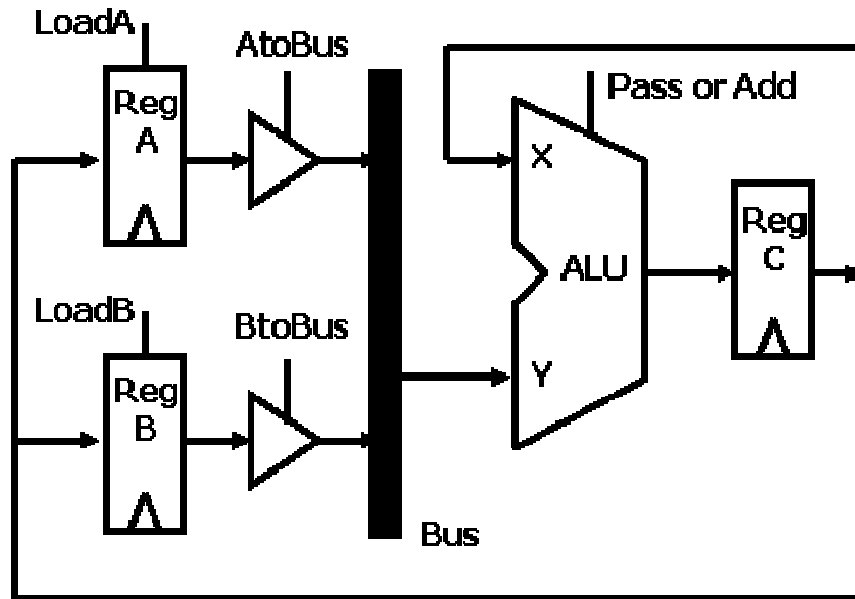
Fill in the table below.

Mystery	MysteryImplementation		
state [5:0]	counter [2:0]	A	B
s0 = 6'b000000	000	0	0
s1 = 6'b000001			
s2 = 6'b000010			
s3a = 6'b100011			
s3b = 6'b001011			
s4aa = 6'b100100			
s4ab = 6'b010100			
s4bb = 6'b001100			

(d – 10 pts) Our next task is to verify that the output of the two machines is the same. Note that in `MysteryImplementation`, the output `Out` is specified using an `assign` statement. Describe in words, why this will be asserted in the same way as in the original module and why it is still the same type of FSM (Moore, Mealy, or synchronous-Mealy).

3. Basic Computer Organization (25 points)

You are given the data-path below. Note that there are three registers. Two of these have a load control input (A and B), while the other (C) loads a new value on every clock cycle. There are two tri-state drivers that connect the outputs of registers A and B to a common bus. Finally, there is an ALU that can perform two operations: pass Y, add X and Y. X is always the output of register C, while Y is the value on the bus.



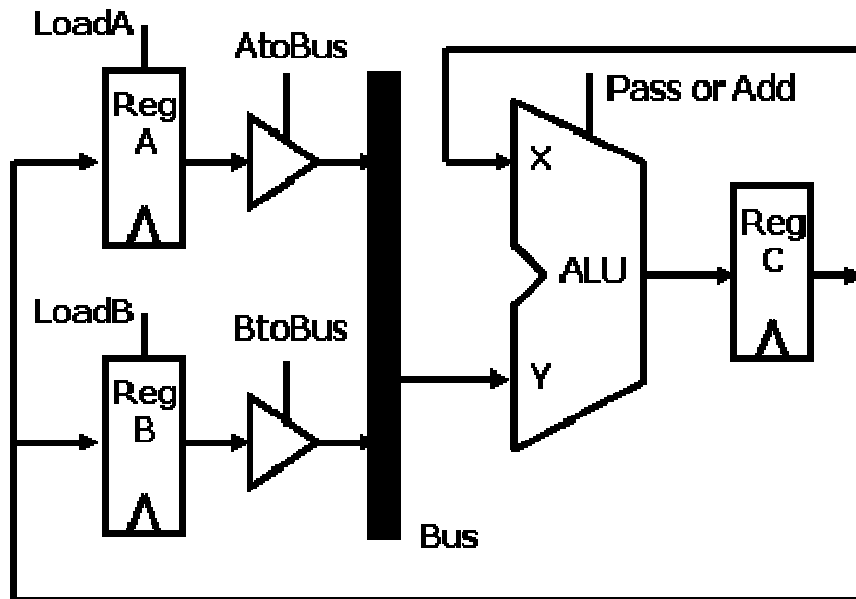
(a – 5 pts) Show the register-transfer operations needed to implement an instruction that swaps the contents of the A and B registers (*SWAP A, B*). Make sure to clearly indicate how many states will be needed to implement the instruction and the value of each control signal in each state. Assume the instruction is already in the instruction register so there is no need to worry about fetching the instruction or incrementing a program counter.

cycle	register-transfer operations	AtoBus	BtoBus	ALU	LoadA	LoadB
1	$C \leftarrow A$	1	0	Pass	0	0
2						
3						

(b – 5 pts) Show the register-transfer operations needed to implement an instruction that doubles the contents of the A register (*TWOX A*). Make sure to clearly indicate how many states will be needed to implement the instruction and the value of each control signal in each state. Assume the instruction is already in the instruction register so there is no need to worry about fetching the instruction or incrementing a program counter.

cycle	register-transfer operations	AtoBus	BtoBus	ALU	LoadA	LoadB
1						
2						
3						

(c – 10 pts) How would you minimally change the architecture above so that you could perform the SWAP instruction in two cycles? You don't need to be concerned with the TWOX instruction for this question. Alter the data path diagram below.



(d – 5 pts) How would you minimally change the architecture above so that you could perform the TWOX instruction in two cycles? You don't need to be concerned with the SWAP instruction for this question. Alter the data path diagram below.

