

**ECE 120 Third Midterm Exam  
Fall 2016**

Tuesday, November 15, 2016

Name: _____	NetID: _____	
Discussion Section:		
9:00 AM		
10:00 AM		
11:00 AM	<input type="checkbox"/> AB1	<input type="checkbox"/> AB8
12:00 PM	<input type="checkbox"/> AB2	<input type="checkbox"/> AB9
1:00 PM	<input type="checkbox"/> AB3	<input type="checkbox"/> ABA
2:00 PM	<input type="checkbox"/> AB4	<input type="checkbox"/> ABB
3:00 PM	<input type="checkbox"/> AB5	
4:00 PM	<input type="checkbox"/> AB6	<input type="checkbox"/> ABC
5:00 PM	<input type="checkbox"/> AB7	<input type="checkbox"/> ABD

- **Be sure that your exam booklet has 8 pages.**
- **Write your name, netid and check discussion section on the title page.**
- **Do not tear the exam booklet apart, except for the last two pages.**
- **Use backs of pages for scratch work if needed.**
- **This is a closed book exam. You may not use a calculator.**
- **You are allowed one handwritten 8.5 x 11" sheet of notes (both sides).**
- **Absolutely no interaction between students is allowed.**
- **Clearly indicate any assumptions that you make.**
- **The questions are not weighted equally. Budget your time accordingly.**
- **Show your work.**

Problem 1     16 points     \_\_\_\_\_

Problem 2     23 points     \_\_\_\_\_

Problem 3     15 points     \_\_\_\_\_

Problem 4     10 points     \_\_\_\_\_

Problem 5     16 points     \_\_\_\_\_

---

Total            80 points     \_\_\_\_\_

### Problem 1 (16 points): FSM Design

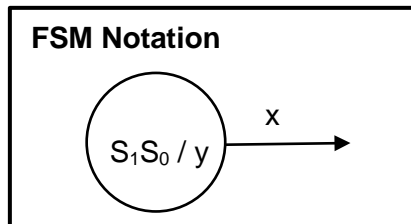
In this problem you will implement a **011** sequence recognizer. The circuit has one input  $x$ , one output  $y$ , and the output is 1 if and only if the pattern **011** has been detected in the input stream.

Example:

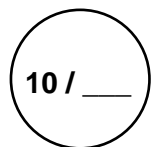
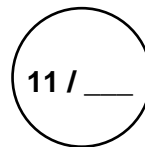
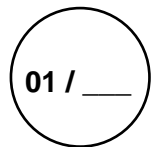
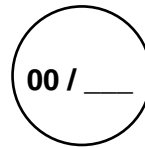
Input:  $x =$  0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 0 1 1 1 . . .  
 Output:  $y =$  0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 1 0 . . .

Note that the output sequence is delayed by 1 clock cycle compared to the input sequence because the output is a function of the flip-flop outputs.

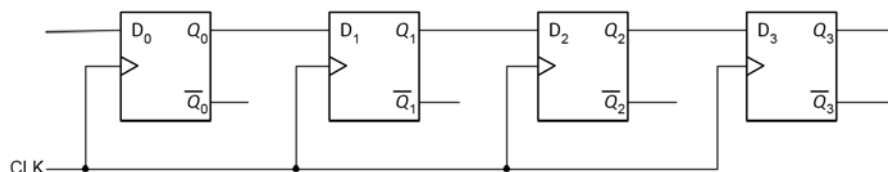
1. (11 points) Draw the *Moore* state diagram, labeling the output and inputs. Give the meaning of each state.



State	Meaning
"Start" 00	
01	
10	
11	



2. (5 points) Shown below is a 4-bit **shift register**, constructed with 4 positive-edge-triggered D flip-flops. Use this shift register and **only one gate** (NOT, AND, OR, NAND, NOR, XOR, or XNOR) to implement a circuit which recognizes **011** just like the example at the top of the page. Be sure to label input  $x$  and output  $y$ .

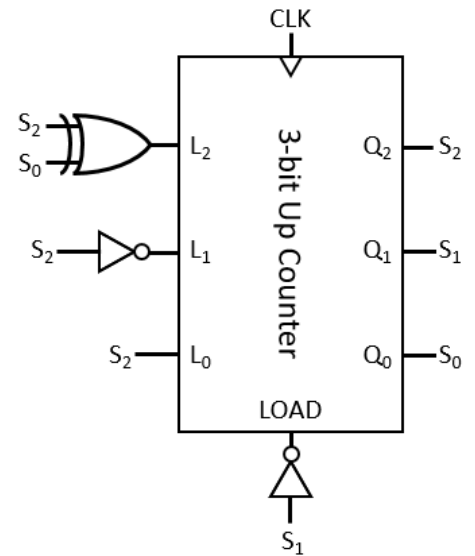


**Problem 2 (23 points): FSM Analysis**

1. **(13 points)** The 3-bit binary up counter shown to the right has parallel load. If  $\text{LOAD}=1$ , the counter loads the input value  $L_2L_1L_0$  into  $Q_2Q_1Q_0$  in the next clock cycle; if  $\text{LOAD}=0$ ,  $Q_2Q_1Q_0$  counts up.

Fill out the state-transition table below for state  $S_2S_1S_0$ . Start by filling out the  $\text{LOAD}$  column.

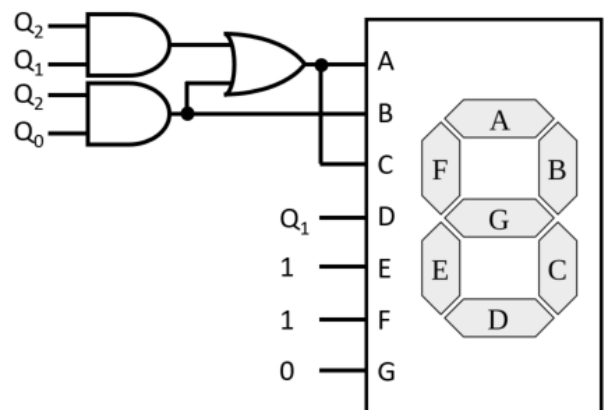
$S_2$	$S_1$	$S_0$	LOAD	$S_2^+$	$S_1^+$	$S_0^+$
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				



2. **(10 points)** In this part, we use a 3-bit modulo-6 binary up counter  $Q_2Q_1Q_0$  (not shown in the figure) to light up the 7-segment LED display shown to the right. For example, the segment marked 'A' lights up if  $A=1$ .

In the table below, the sequence of states  $Q_2Q_1Q_0$  generated by the counter is given to you, starting with the state 001 in clock cycle #1.

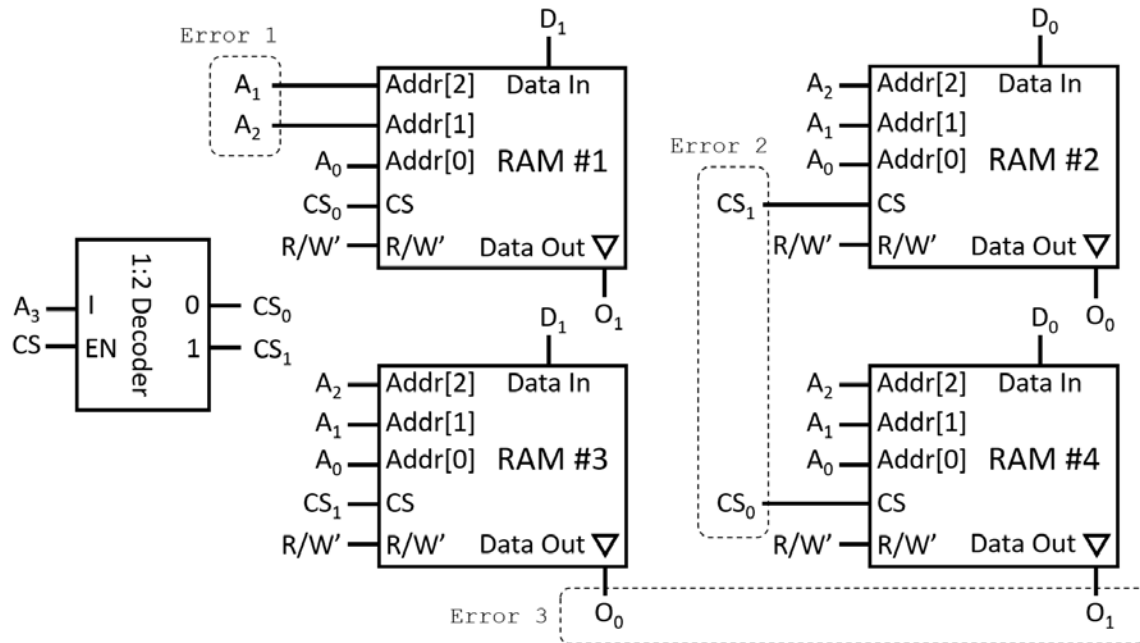
Fill in the LED segments that light up in each clock cycle. The first one has been done for you.



cycle	#1	#2	#3	#4	#5	#6
$Q_2Q_1Q_0$	001	010	011	100	101	000
seven segment display						

### Problem 3 (15 points): Memory

A careless engineer attempted to build a 16x2 RAM out of a 1:2 decoder and four 8x1 RAMs. The 16x2 RAM is intended to have two Data-In bits  $D_1D_0$ , four address bits  $A_3A_2A_1A_0$ , chip select CS, read/not-write  $R/W'$ , and two Data-Out bits  $O_1O_0$ . Unfortunately, the engineer switched 3 pairs of connections compared to the conventional design. The mistakes are shown in the diagram as Errors 1 to 3.



1. (12 points) Suppose a user attempts to write ( $CS=1$  and  $R/W'=0$ ) into the faulty 16x2 RAM with data  $D_1D_0 = 01$  at address  $A_3A_2A_1A_0 = 0101$ . Indicate in the table below which of the 8x1 RAMs are accessed by **circling YES or NO for each RAM**. If an 8x1 RAM is accessed, also write down the 1-bit Data-In and the 3-bit address  $Addr[2:0]$  for that 8x1 RAM.

RAM #1	Accessed? YES / NO	RAM #2	Accessed? YES / NO
Data-In _____	Addr[2:0] _____	Data-In _____	Addr[2:0] _____
RAM #3	Accessed? YES / NO	RAM #4	Accessed? YES / NO
Data-In _____	Addr[2:0] _____	Data-In _____	Addr[2:0] _____

2. (3 points) If you want to make a functional 16x2 RAM by correcting the **minimal number of errors**, which error(s) must you fix?

Circle the error(s) that MUST be fixed:      Error 1      Error 2      Error 3

**Problem 4 (10 points): von Neumann Model**

For each question, **CIRCLE EXACTLY ONE ANSWER.**

In the von Neumann model, where is the program stored?

processing unit      output      MAR      PC      memory

What is the first step of instruction processing in a computer based on the von Neumann model?

EXECUTE      READ OPERANDS      DECODE      FETCH      INCREMENT PC

Operands for instructions in a computer based on the von Neumann model **can NOT be found in** which of the following?

memory      tri-state buffers      register file      input      PC

Which of the following contains the address of the next instruction to be executed by the computer?

ALU      MDR      PC      IR      MAR

In the von Neumann model, the memory includes which of the following?

MAR      register file      KBDR      FSM      control unit

### Problem 5 (16 points): LC-3 Interpretation and Assembly

The registers of an LC-3 processor currently have the values shown in the table to the right.

R0	x0111	R4	bits	PC	x3001
R1	x3002	R5	bits	IR	xE622
R2	x1175	R6	bits	MAR	x3000
R3	x3023	R7	bits	MDR	xE622

The table to the right shows some of the contents of the LC-3 processor's memory.

When the bits represent instructions, an interpretation has been provided for you in RTL.

address	contents	RTL interpretation
x3000	1110 0110 0010 0010	$R3 \leftarrow PC + x0022$
x3001	0110 0010 1100 0001	$R1 \leftarrow M[R3 + 1]$
x3002	1001 0100 0111 1111	$R2 \leftarrow \text{NOT } R1$
x3003	0111 0100 1100 0010	$M[R3 + 2] \leftarrow R2$
	...	...
x3023	0000 1111 0000 0101	$PC \leftarrow PC + xFF05$
x3024	0001 0011 0001 0100	(data: x1314)
x3025	1101 1010 1011 1100	(data: xDABC)
x3026	0000 0000 0000 0000	(no operation)

Here's the question:

The LC-3 processor **PROCESSES THREE INSTRUCTIONS.**

1. (7 points) Write a complete list of the sequence of values taken by the MAR register as the LC-3 processes these instructions. Use only as many lines as are necessary.

#1: x3000 (initial value)      #5: \_\_\_\_\_  
 #2: \_\_\_\_\_      #6: \_\_\_\_\_  
 #3: \_\_\_\_\_      #7: \_\_\_\_\_  
 #4: \_\_\_\_\_      #8: \_\_\_\_\_

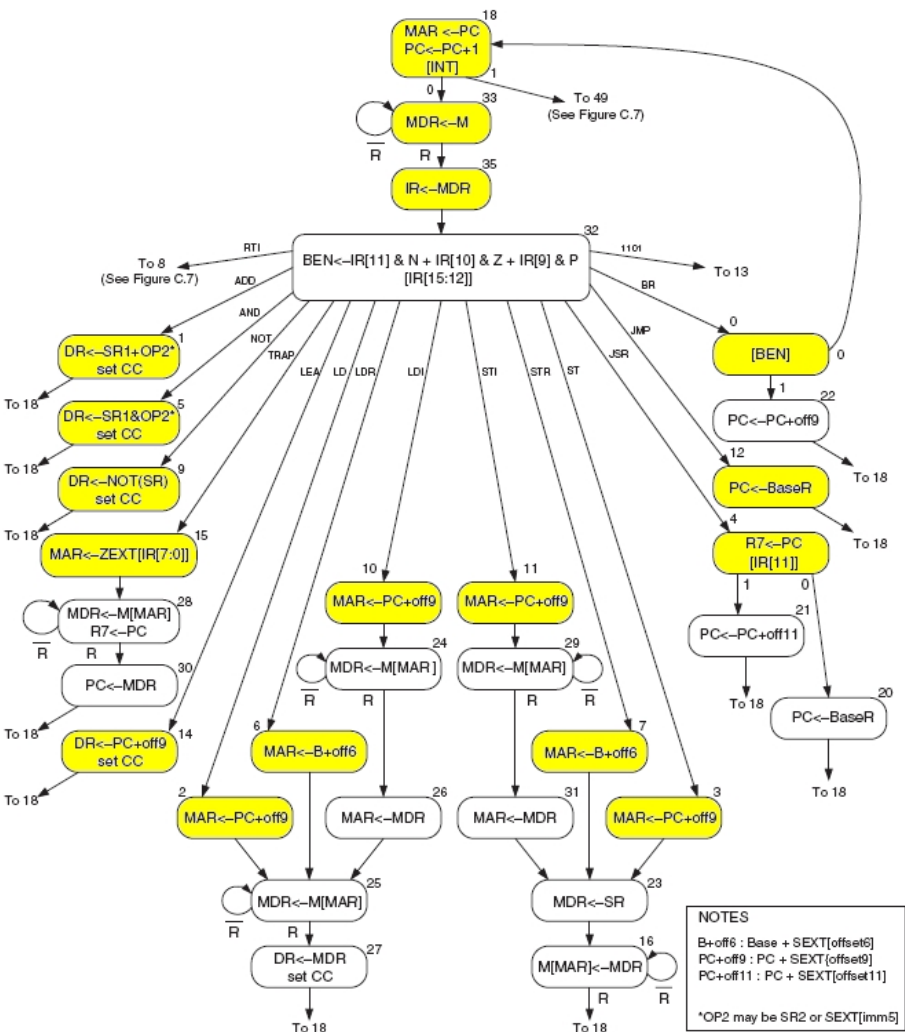
2. (9 points) Complete the tables below with the **FINAL** values (after processing of three instructions) of each register and memory location. If you cannot know a particular value, write "bits." Note: You **must write your answers in hexadecimal**.

			memory address	contents
R0	<input type="text"/>	PC	<input type="text"/>	
R1	<input type="text"/>	IR	x3002	<input type="text"/>
...			...	
R4	<input type="text"/>	MDR	x3024	<input type="text"/>
			x3025	<input type="text"/>

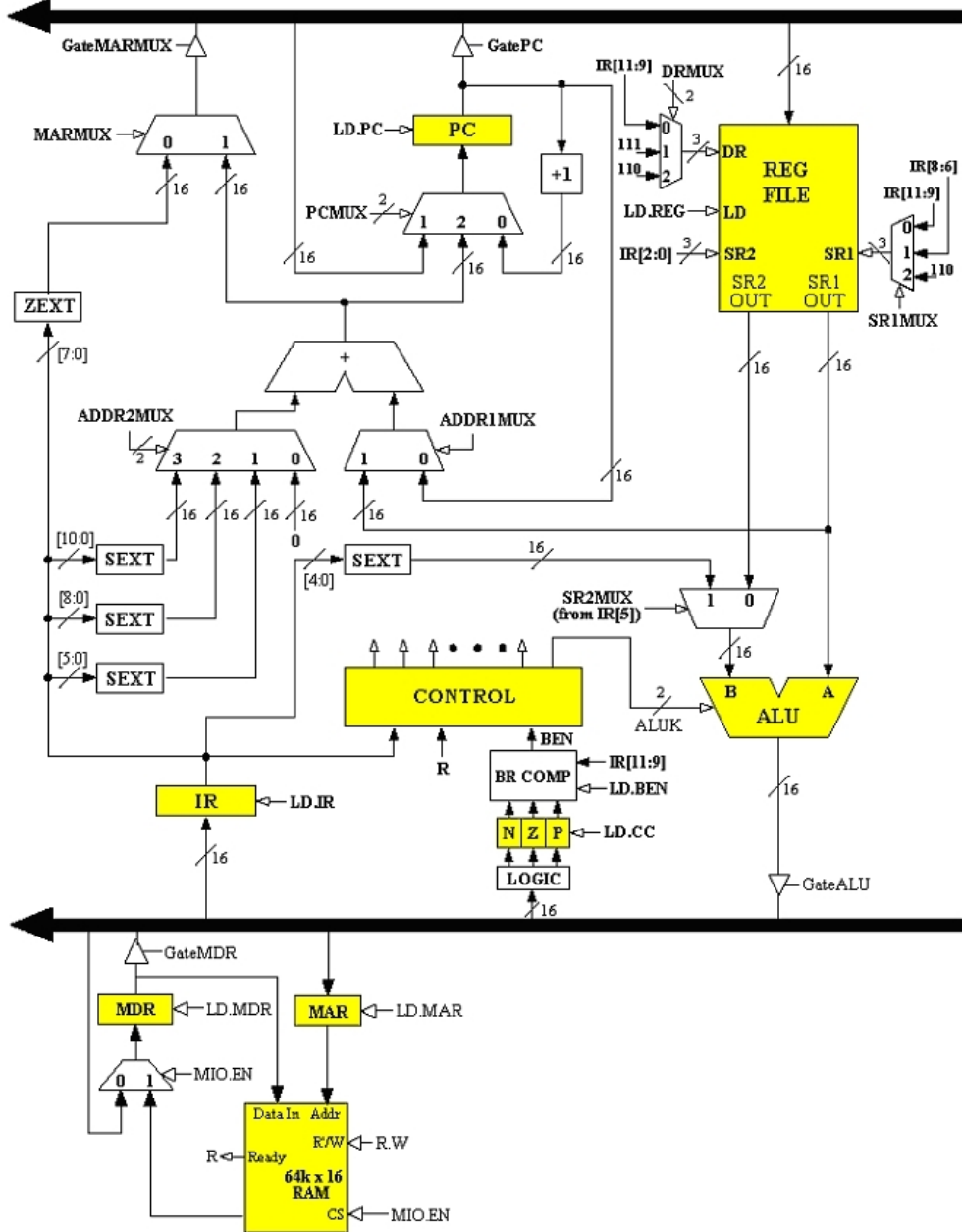
LC-3 Instructions

ADD	0001	DR	SR1	0	00	SR2	ADD DR, SR1, SR2	LD	0010	DR		PCoffset9	LD DR, PCoffset9
							DR ← SR1 + SR2, Setcc						DR ← M[PC + SEXT(PCoffset9)], Setcc
ADD	0001	DR	SR1	1		imm5	ADD DR, SR1, imm5	LDI	1010	DR		PCoffset9	LDI DR, PCoffset9
							DR ← SR1 + SEXT(imm5), Setcc						DR ← M[M[PC + SEXT(PCoffset9)]], Setcc
AND	0101	DR	SR1	0	00	SR2	AND DR, SR1, SR2	LDR	0110	DR	BaseR	offset6	LDR DR, BaseR, offset6
							DR ← SR1 AND SR2, Setcc						DR ← M[BaseR + SEXT(offset6)], Setcc
AND	0101	DR	SR1	1		imm5	AND DR, SR1, imm5	LEA	1110	DR		PCoffset9	LEA DR, PCoffset9
							DR ← SR1 AND SEXT(imm5), Setcc						DR ← PC + SEXT(PCoffset9), Setcc
BR	0000	n	z	p		PCoffset9	BR(nzp) PCoffset9	NOT	1001	DR	SR	111111	NOT DR, SR
							(n AND N) OR (z AND Z) OR (p AND P): PC ← PC + SEXT(PCoffset9)						DR ← NOT SR, Setcc
JMP	1100	000	BaseR	000000			JMP BaseR	ST	0011	SR		PCoffset9	ST SR, PCoffset9
							PC ← BaseR						M[PC + SEXT(PCoffset9)] ← SR
JSR	0100	1		PCoffset11			JSR PCoffset11	STI	1011	SR		PCoffset9	STI SR, PCoffset9
							R7 ← PC, PC ← PC + SEXT(PCoffset11)						M[M[PC + SEXT(PCoffset9)]] ← SR
TRAP	1111	0000		trapvec8			TRAP trapvec8	STR	0111	SR	BaseR	offset6	STR SR, BaseR, offset6
							R7 ← PC, PC ← M[ZEXT(trapvec8)]						

LC-3 FSM



## LC-3 Datapath



## LC-3 Datapath Control Signals

Signal	Description	Signal	Description
LD.MAR	= 1, MAR is loaded	LD.CC	= 1, updates status bits from system bus
LD.MDR	= 1, MDR is loaded	GateMARMUX	= 1, MARMUX output is put onto system bus
LD.IR	= 1, IR is loaded	GateMDR	= 1, MDR contents are put onto system bus
LD.PC	= 1, PC is loaded	GateALU	= 1, ALU output is put onto system bus
LD.REG	= 1, register file is loaded	GatePC	= 1, PC contents are put onto system bus
LD.BEN	= 1, updates Branch Enable (BEN) bit		
MARMUX	= 0, chooses ZEXT [IR[7:0]] = 1, chooses address adder output	MIO.EN	= 1, Enables memory, chooses memory output for MDR input = 0, Disables memory, chooses system bus for MDR input
ADDR1MUX	= 0, chooses PC = 1, chooses reg file SR1 OUT	R.W	= 1, M[MAR] < MDR when MIO.EN = 1 = 0, MDR < M[MAR] when MIO.EN = 1
ADDR2MUX	= 00, chooses "0...00" = 01, chooses SEXT [IR[5:0]] = 10, chooses SEXT [IR[8:0]] = 11, chooses SEXT [IR[10:0]]	ALUK	= 00, ADD = 01, AND = 10, NOT A = 11, PASS A
PCMUX	= 00, chooses PC + 1 = 01, chooses system bus = 10, chooses address adder output	DRMUX	= 00, chooses IR[11:9] = 01, chooses IR[8:6] = 10, chooses "110"
SR1MUX	= 00, chooses IR[11:9] = 01, chooses IR[8:6] = 10, chooses "110"		