

**HOMEWORK IV**

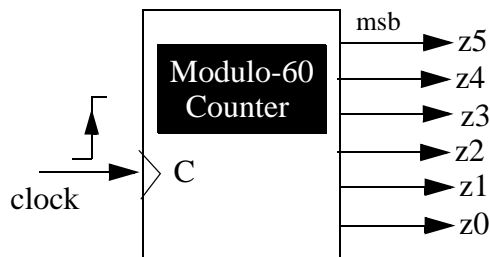
**DUE** : November 3, 2011

**READ** : Related portions of Chapters III, VII and VIII

**ASSIGNMENT** : There are five questions, one of which are developed from Chapter III and VIII of the textbook.

**Solve all homework and exam problems as shown in class and past exam solutions.**

**1)** Design a modulo-60 (divide-by-60) **synchronous up** counter circuit by using 4-bit counters studied in class and a few gates :

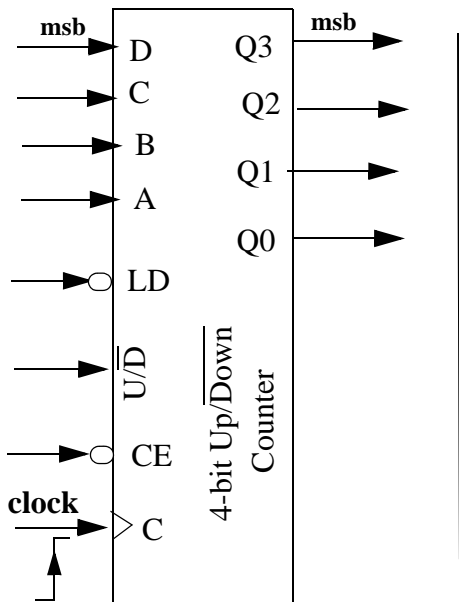


The count range is 0 through 59.

Clearly show the connections of inputs and outputs of the chips and gates.

Label all the chip pins. Show what is connected to all inputs and outputs.

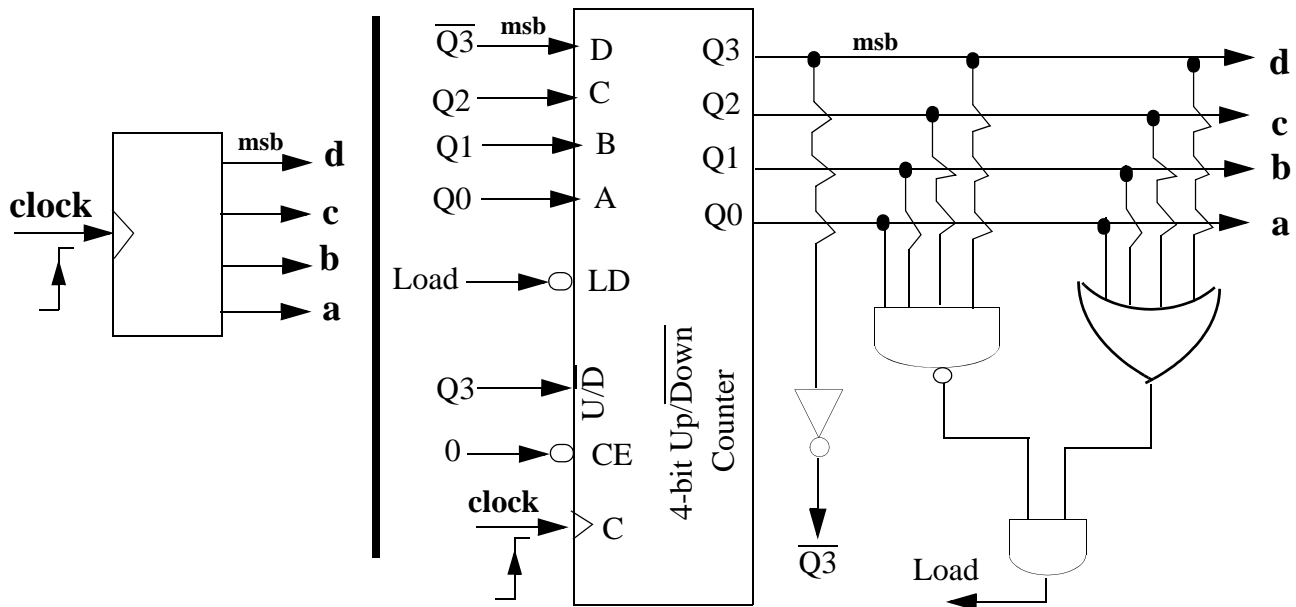
Note that the 4-bit counter studied **in class** has the following black box view and operation table :



4-bit Up/ $\overline{\text{D}}$ own Counter Operation Table

$\overline{\text{LD}}$	$\text{U}/\overline{\text{D}}$	$\overline{\text{CE}}$	C	Operation
0	x	0	↑	Store (DCBA) (Next count is DCBA)
1	0	0	↑	Count Down (Next count is 1 down)
1	1	0	↑	Count Up (Next count is 1 up)
x	x	1	x	Not stored
x	x	x	0	Not stored

2) Consider the following circuit whose black box view and internal circuitry are given :



What is the counting sequence of the above circuit ? That is, determine the purpose of a sequential circuit, i.e. what the circuit does.

We do not have to step through the six analysis steps since we know the sequential circuit is a counter with four flip-flops, wiring and gating. All we have to do is the timing analysis and the specification of the purpose : just the 6<sup>th</sup> step ! To do the timing analysis, we will start with an arbitrary count on the 4-bit counter, such as, Q<sub>3</sub>,Q<sub>2</sub>,Q<sub>1</sub>,Q<sub>0</sub> at **0, 1, 0, 1**, respectively.

Then observe the inputs and outputs of the counter until we figure out a pattern which then leads us to the purpose of the sequential circuit. For that, we work on a table that starts with initial input and output values, the values at t<sub>0</sub> :

Q<sub>3</sub> is connected to U/ $\bar{D}$ , so at time t<sub>0</sub> the U/ $\bar{D}$  value is 0

time	Q <sub>3</sub>	Load	$\bar{Q}_3$	Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>	Initial values									
	U/ $\bar{D}$	$\bar{LD}$	D	C	B	A	Q <sub>3</sub>	Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>	Load	$\bar{Q}_3$	d	c	b	a
t <sub>0</sub>	0	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1

Then, get the outputs at t<sub>1</sub> from the input values at t<sub>0</sub> :

time	Q <sub>3</sub>	Load	$\bar{Q}_3$	Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>					Initial values					
	U/ $\bar{D}$	$\bar{LD}$	D	C	B	A	Q <sub>3</sub>	Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>	Load	$\bar{Q}_3$	d	c	b	a
t <sub>0</sub>	0	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1
t <sub>1</sub>							0	1	0	0	1	1	0	1	0	0

Then, get the input values at t1 from the output values at t1 :

Q3 is connected to  $U/\overline{D}$ , so at time t1 the  $U/\overline{D}$  value is 0

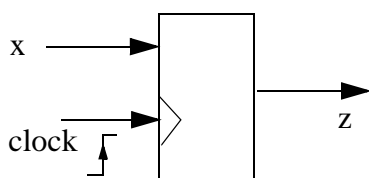
time	Q3	Load	$\overline{Q3}$	Q2	Q1	Q0						d	c	b	a	
	$\downarrow$ U/ $\overline{D}$	$\downarrow$ $\overline{LD}$	$\downarrow$ D	$\downarrow$ C	$\downarrow$ B	$\downarrow$ A	Q3	Q2	Q1	Q0	Load					$\overline{Q3}$
t0	0	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1
t1	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	0

Then, get the output values at t2, from the input values at t1 :

time	Q3	Load	$\overline{Q3}$	Q2	Q1	Q0						d	c	b	a	
	$\downarrow$ U/ $\overline{D}$	$\downarrow$ $\overline{LD}$	$\downarrow$ D	$\downarrow$ C	$\downarrow$ B	$\downarrow$ A	Q3	Q2	Q1	Q0	Load					$\overline{Q3}$
t0	0	1	1	1	0	1	0	1	0	1	1	1	0	1	0	1
t1	0	1	1	1	0	0	0	1	0	0	1	1	0	1	0	0
t2	.... Continue .....						0	0	1	1	1	1	0	0	1	1

Continue in this fashion to complete the table.

3) Consider the following 1-input, 1-output sequential circuit :



The **input/output relationship** of the sequential circuit is that the circuit checks for three successive **1s**.

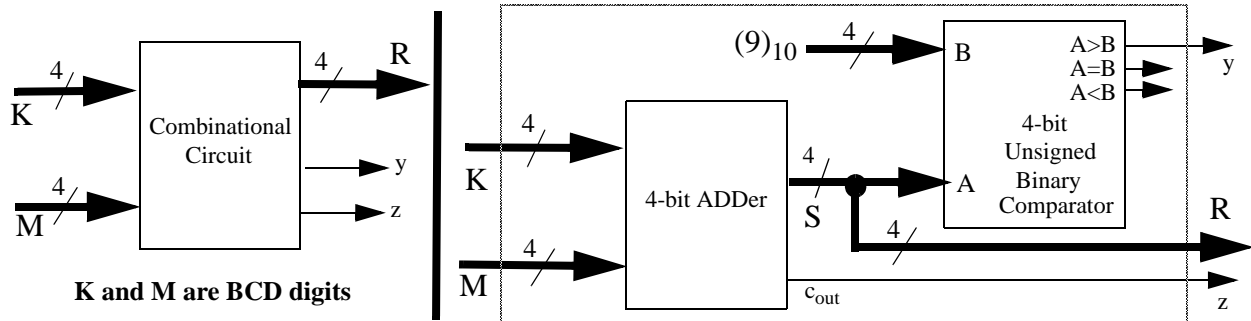
After receiving three 1s in a row, the following clock period, it outputs a 1 for **two** clock periods. Then, it starts checking for three 1s again. Hence, its cycle of checking bits is 5 clock periods.

If it does not receive a 1 when it is waiting for a 1, the next clock period, it starts checking for three 1s again.

Obtain the state diagram of this sequential circuit as discussed in class. Is this a finite memory or non-finite memory sequential circuit ? Why ?

4) Consider the combinational circuit below with 8 inputs and 3 sets of outputs.

**Analyze** the circuit to obtain its **purpose**. The purpose must include the meaning of each output (R, y and z).



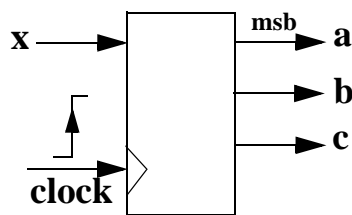
Note that there are 8 inputs therefore a truth table is impractical. You are suggested that you obtain an **operation table** that classifies input combinations into a few possibilities.

5) Solve Problem 3.31.

The question is asking this : “Your friend tells you to hold a CMOS chip. Would you hold the chip ? Why ?”

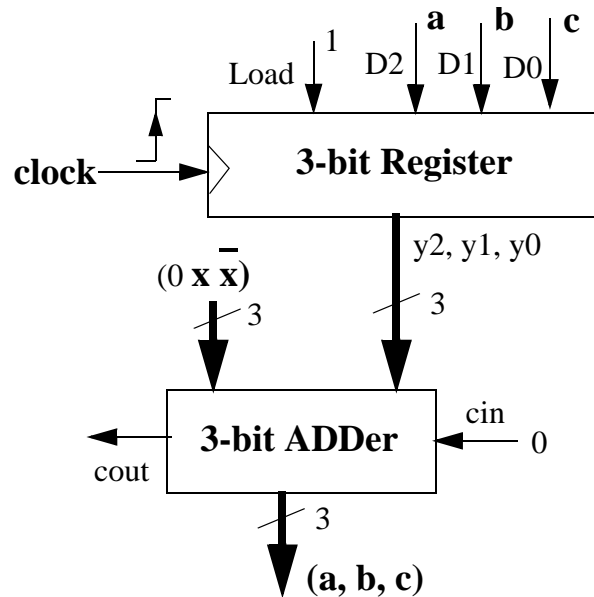
## RELEVANT QUESTIONS AND ANSWERS

**Q1)** Consider a **sequential** circuit with a **single** input and **three** outputs. The black-box view and implementation of this sequential circuit are shown below. The sequential circuit uses a 3-bit register whose operation table is also given below.



3-bit Register Operation Table

Load	C	Operation
1	↑	Load (D2, D1, D0)
0	↑	Not stored
X	0	Not stored



Note that one input of the ADDer is  $(0x\bar{x})$  where “ $x$ ” is input  $x$ . Therefore, this input of the ADDer is either “001”

when  $x$  is 0 or “010” when  $x$  is 1. Determine the purpose of this sequential circuit by continuing with the following table and showing the values for **11** clock periods :

Time	$x$	$y_2, y_1, y_0$	$(0 \ x \ \bar{x})$	$a \ b \ c$
t0	1	1 0 1	0 1 0	1 1 1
t1	1			
t2	1			
t3	1			
t4	0			
t5	0			

Time	$x$	$y_2, y_1, y_0$	$(0 \ x \ \bar{x})$	$a \ b \ c$
t6	0			
t7	0			
t8	0			
t9	1			
t10	1			

**A1)** The table is continued below :

Time	$x$	$y_2, y_1, y_0$	$(0 \ x \ \bar{x})$	$a \ b \ c$
t0	1	1 0 1	0 1 0	1 1 1
t1	1	1 1 1	0 1 0	0 0 1
t2	1	0 0 1	0 1 0	0 1 1
t3	1	0 1 1	0 1 0	1 0 1
t4	0	1 0 1	0 0 1	1 1 0
t5	0	1 1 0	0 0 1	1 1 1

Time	$x$	$y_2, y_1, y_0$	$(0 \ x \ \bar{x})$	$a \ b \ c$
t6	0	1 1 1	0 0 1	0 0 0
t7	0	0 0 0	0 0 1	0 0 1
t8	0	0 0 1	0 0 1	0 1 0
t9	1	0 1 0	0 1 0	1 0 0
t10	1	1 0 0	0 1 0	1 1 0

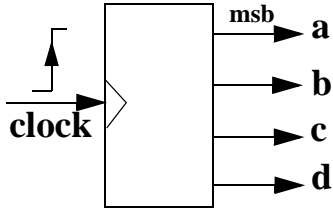
The purpose of the circuit is almost identical to the purpose of the circuit in Question 1 above. It is a 3-bit Up counter that count up

- by 1 when  $x$  is 0 : 3, 4, 5, 6, 7, 0, 1, 2,...
- by 2 when  $x$  is 1 : 7, 1, 3, 5, 7, 1, 3, 5,....

The only **difference** is that when the value of “ $x$ ” is changed, the new value takes effect immediately in this circuit. In the Question 1 circuit, it affects in the following clock period. For example, in both circuits, “ $x$ ” is changed from 1 to 0 in t4. The Question 2 circuit counts up by 1 in t4. But, the Question 1 circuit counts up by 2 in t4. This difference can be eliminated by delaying “ $x$ ” by one clock period. This can be done by storing it on a FF. This question confirms lab and classroom discussions that a simple sequential circuit can be converted to a circuit with registers/counters/shift registers.

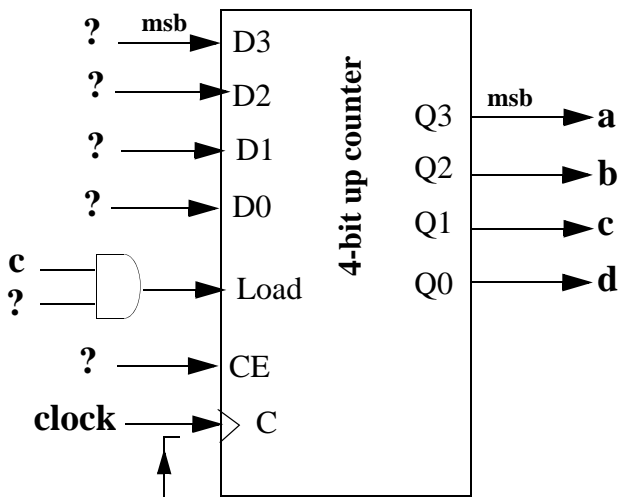
**Q2)** Consider a **sequential** circuit with **four** outputs. The black-box view, the partial implementation, the operation table of the 4-bit up counter used by the circuit and output values for 18 clock periods are given below.

The partial implementation of the sequential circuit contains a 4-bit up counter and an AND gate for its Load input.



4-bit Up Counter Operation Table

Load	CE	C	Operation
1	1	↑	Load (D3, D2, D1, D0)
0	1	↑	Count up
0	0	X	Not stored
0	X	0	Not stored



Time	Load	D3	D2	D1	D0	a	b	c	d
t0	0	1	1	1	1	0	0	0	0
t1	0	1	1	1	0	0	0	0	1
t2	0	1	1	0	1	0	0	1	0
t3	1	1	1	0	0	0	0	1	1
t4	0	0	1	1	1	1	1	0	0
t5	0	0	1	1	0	1	1	0	1
t6	0	0	1	0	1	1	1	1	0
t7	1	0	1	0	0	1	1	1	1
t8	0	1	0	1	1	0	1	0	0
t9	0	1	0	1	0	0	1	0	1
t10	0	1	0	0	1	0	1	1	0
t11	1	1	0	0	0	0	1	1	1
t12	0	0	0	1	1	1	0	0	0
t13	0	0	0	1	0	1	0	0	1
t14	0	0	0	0	1	1	0	1	0
t15	1	0	0	0	0	1	0	1	1
t16	0	1	1	1	1	0	0	0	0
t17	0	1	1	1	0	0	0	0	1

Obtain the **full** implementation of the circuit which must contain (i) the above 4-bit up counter, (ii) the above AND gate and (iii) a few more gates. **No** other component can be used !

**Explain** your decision for each **counter input** and the **AND gate input**. **Draw the circuit in your blue book**.

**A2)** The explanation of the inputs and the circuit are below :

**CE :**

We see that the counter counts up or loads every clock period. Therefore, its clock input must be enabled all the time. CE must be permanently connected to 1 : **CE = 1**

**Load :**

We see that the counter loads when “c” is 1 and “d” is 1. Therefore, **Load = cd**

**D0:**

We see that D0 is the complement of “d”. Therefore, **D0 =  $\bar{d}$**

**D1 :**

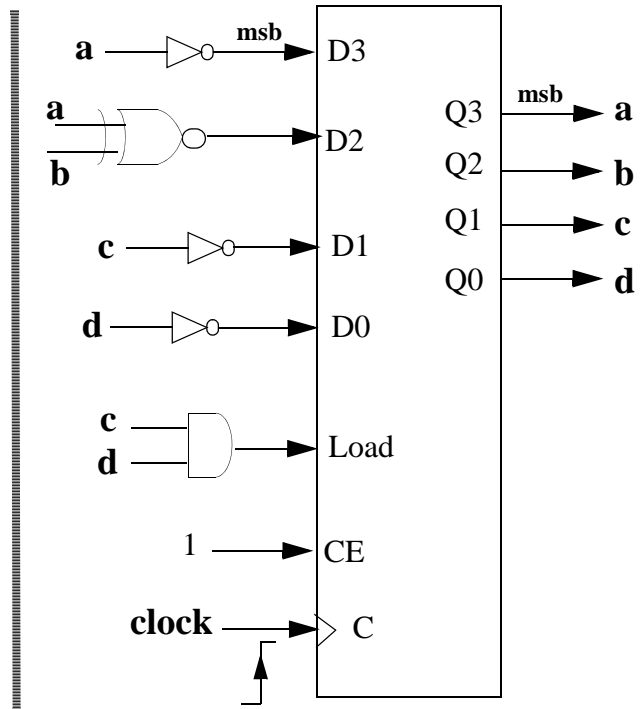
We see that D1 is the complement of “c”. Therefore, **D1 =  $\bar{c}$**

**D2 :**

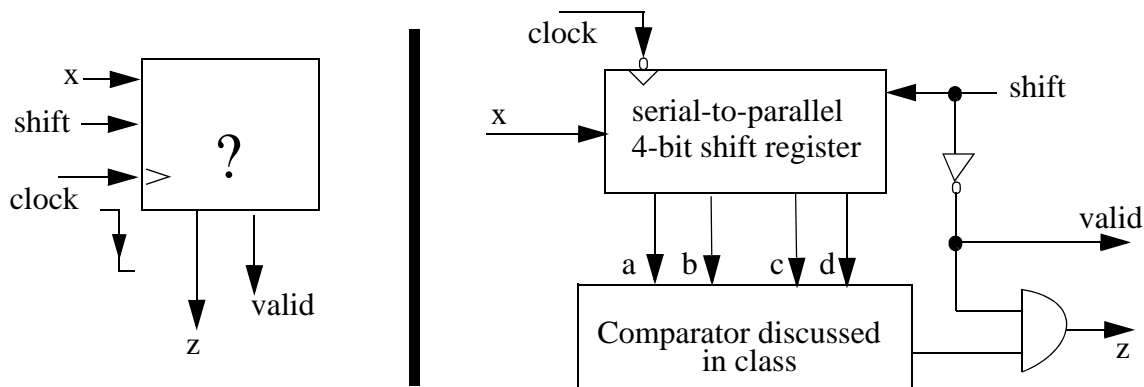
We see that D2 is 1 when “a” and “b” are the same at the same time. Therefore, **D2 =  $a \oplus b$**

**D3 :**

We see that D3 is the complement of “a”. Therefore, **D3 =  $\bar{a}$**



**Q3)** Consider the following sequential circuit below. The sequential circuit has 2 inputs (x and shift) and 2 outputs (z and valid). The shift input is active when the x input has valid data so we can shift it in. Determine what the circuit does, i.e. its purpose.



**A3)** We are asked to perform a sequential circuit analysis where there are less than 5 flip-flops. But, we are not shown those individual flip-flops, so we cannot do the formal sequential circuit analysis for this circuit from step 1 through step 6. But, we are told what these four flip-flops do as a whole : a serial-to-parallel 4-bit shift register. Thus, we can simply continue with the last step of the analysis : Step 6, the functional description step. First we will do a timing analysis and then state the purpose of the circuit.

We know that the comparator circuit compares two 2-bit unsigned binary numbers. It outputs 1, if (a, b) is greater than or equal to (c, d). For correct operation, four bits must be shifted into the shift register in four clock periods during which the shift input is 1. As long as the shifting is not complete, the comparator output is not correct, so its out-

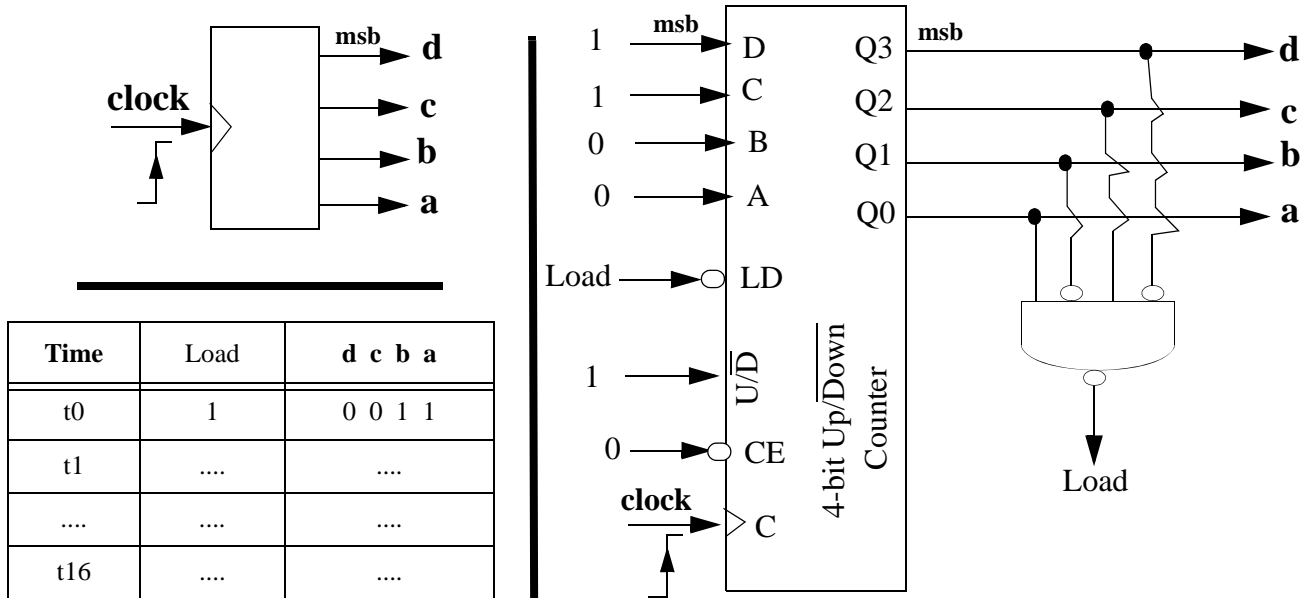
put is not valid. Its output can reflect a correct compare (is valid) when all four bits are shifted in, i.e. when the shift input is 0. That is why, the valid output is the complement of the shift input.

Then, we see that we have an operation period which is at least five clock periods long : four clock periods to shift in x and one clock period to compare. It is important to realize that the circuit that outputs the shift and x signals knows this minimum five clock period duration. We note that number (a, b) is received after number (c, d) is received. We now obtain a table to relate the values of the lines in the circuit. We also pick an arbitrary bit sequence for input x as shown on the table below :

time	shift	x	a b c d	valid	z	Comment
t <sub>0</sub>	1	1	????	0	0	First bit is being received
t <sub>1</sub>	1	0	1???	0	0	Second bit is being received
t <sub>2</sub>	1	1	01??	0	0	Third bit is being received
t <sub>3</sub>	1	1	101?	0	0	Fourth bit is being received
t <sub>4</sub>	0	X	1101	1	1	Compare output is valid.
t <sub>5</sub>	0	X	1101	1	1	Compare output is valid
t <sub>6</sub>	1	0	1101	0	0	First bit is being received
t <sub>7</sub>	1	1	0110	0	0	Second bit is being received
....	.....		....	...	...	.....

Thus, this circuit receives four bits serially, treats them as two 2-bit unsigned numbers and compares them.

**Q4)** Consider a **sequential** circuit with a clock input and four outputs. The black-box view and implementation of this sequential circuit with the 4-bit counter discussed in class are shown below. Determine what this sequential circuit does by continuing with the following table and showing the values for **17** clock periods

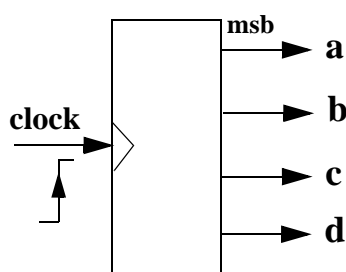


**A4)** The table is continued below. This is a special modulo-10 counter that always counts as ...2, 3, 4, 5, 12, 13, 14, 15, 0, 1, 2, 3,...

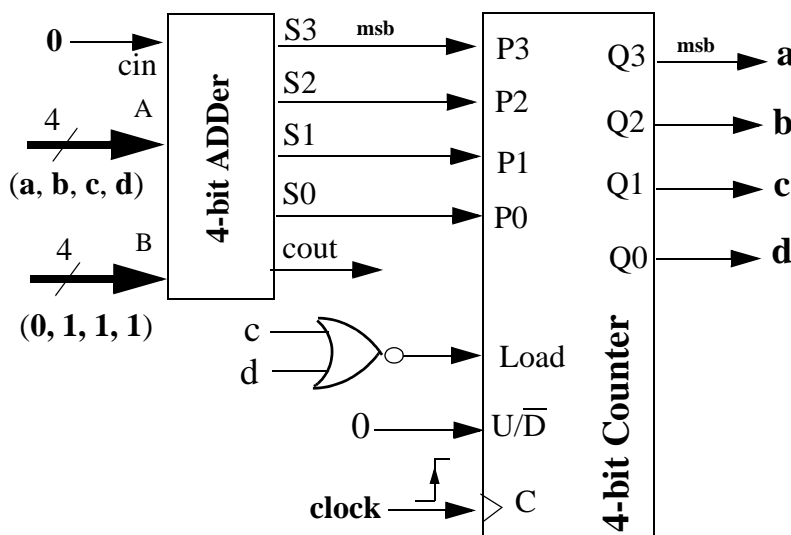
Time	Load	d c b a
t0	1	0 0 1 1
t1	1	0 1 0 0
t2	0	0 1 0 1
t3	1	1 1 0 0
t4	1	1 1 0 1
t5	1	1 1 1 0
t6	1	1 1 1 1
t7	1	0 0 0 0
t8	1	0 0 0 1

Time	Load	d c b a
t9	1	0 0 1 0
t10	1	0 0 1 1
t11	1	0 1 0 0
t12	0	0 1 0 1
t13	1	1 1 0 0
t14	1	1 1 0 1
t15	1	1 1 1 0
t16	1	1 1 1 1

**Q5)** A **sequential** circuit has a clock input and four outputs. It uses a **generic 4-bit Up/Down** counter besides other components. Its black-box view and the operation table of the generic 4-bit counter are shown below :



Load	U/ $\bar{D}$	C	Operation
1	x	↑	Store P inputs (Next count is P)
0	1	↑	Count Up (Next count is 1 up)
0	0	↑	Count down (Next count 1 down)
0	x	0	Not stored



The implementation of the sequential circuit is shown on the left.

**Determine** what this sequential circuit does by continuing with the table below and showing the values for **21** clock periods :

Time	Load	U/ $\bar{D}$	a b c d
t0	0	0	0 0 1 0
....	....	....	....
t20	....	....	....

**A5)** We determine what this sequential circuit does, by completing the table shown below :

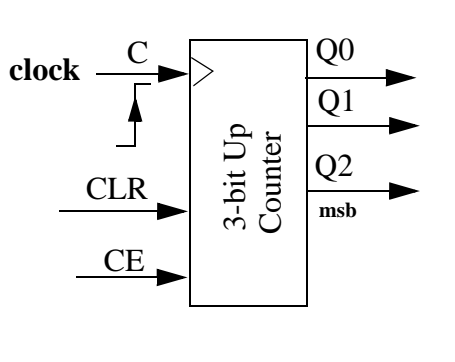
Time	Load	$U/\overline{D}$	a b c d
t0	0	0	0 0 1 0
t1	0	0	0 0 0 1
t2	1	0	0 0 0 0
t3	0	0	0 1 1 1
t4	0	0	0 1 1 0
t5	0	0	0 1 0 1
t6	1	0	0 1 0 0

Time	Load	$U/\overline{D}$	a b c d
t7	0	0	1 0 1 1
t8	0	0	1 0 1 0
t9	0	0	1 0 0 1
t10	1	0	1 0 0 0
t11	0	0	1 1 1 1
t12	0	0	1 1 1 0
t13	0	0	1 1 0 1

Time	Load	$U/\overline{D}$	a b c d
t14	1	0	1 1 0 0
t15	0	0	0 0 1 1
t16	0	0	0 0 1 0
t17	0	0	0 0 0 1
t18	1	0	0 0 0 0
t19	0	0	0 1 1 1
t20	0	0	0 1 1 0

The counter counts always down. Every four counts, it loads a value by adding  $(7)_{10}$  to its current value. The loading happens when the rightmost two outputs of the counter are both 0. In the given example, the counter has initial value 2. It counts down to 0, then loads 7 and counts down to 4, then loads 11 and counts down to 8, then loads 15 and counts down to 12 then loads 3 and counts down to 0. It repeats this continuously.

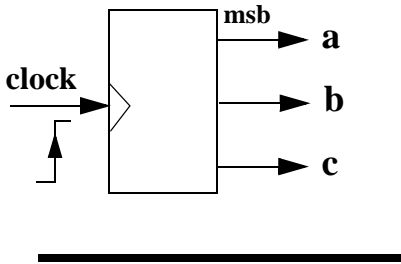
**Q6)** A **sequential** circuit uses two generic 3-bit Up counters whose black box view and operation table are given below :



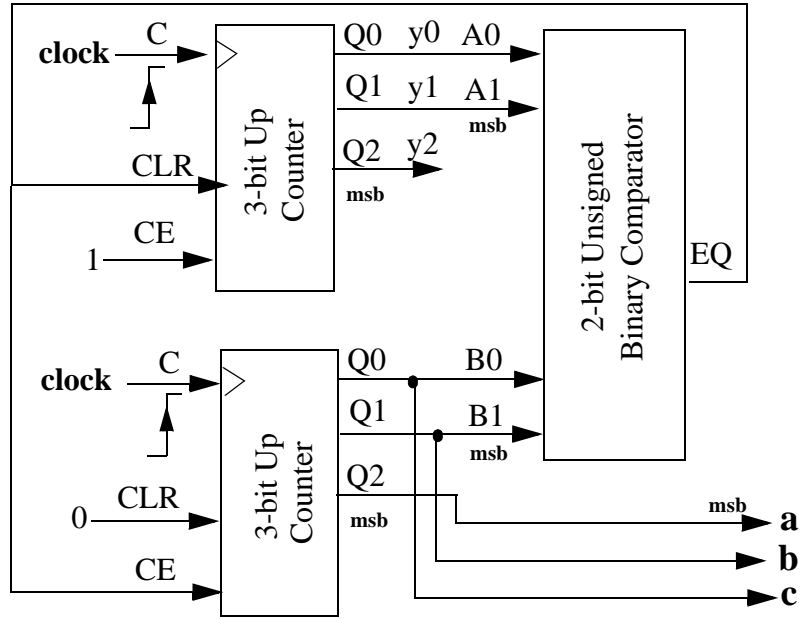
CLR	CE	C	Operation
1	x	↑	Store 0 (Next count is 0)
0	1	↑	Count Up (Next count is 1 up)
0	0	x	Not stored (Do <b>not</b> count up)
0	1	0	Not stored (Do <b>not</b> count up)

The sequential black-box view, the implementation with two **generic** 3-bit Up Counters and a **generic** 2-bit Unsigned Binary Comparator are shown below.

Determine what this sequential circuit does by continuing with the following table and showing the values until a pattern emerges.



Time	EQ	y2	y1	y0	a	b	c
t0	1	0	0	0	0	0	0
t1	0	0	0	0	0	0	1
t2	1	0	0	1	0	0	1
...	...	...	...	...	...	...	...



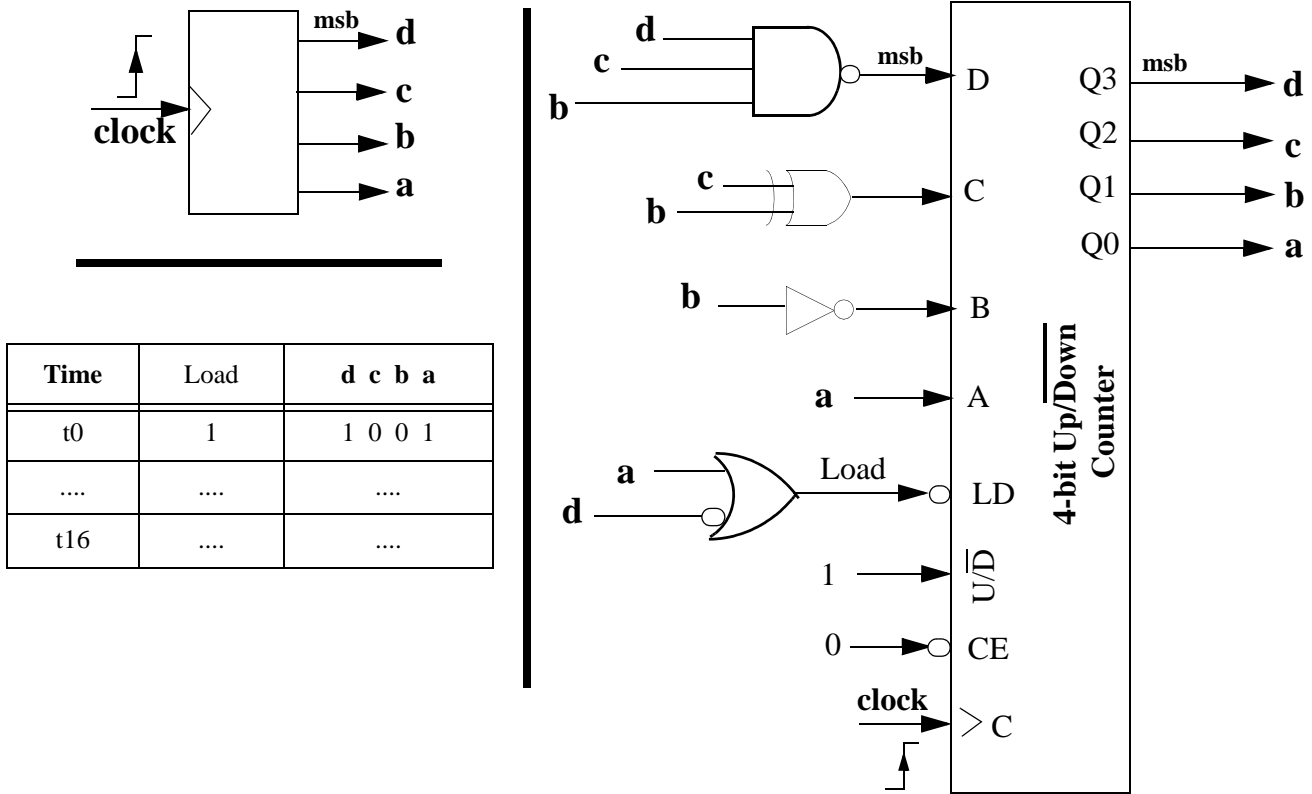
**A6)** We determine what this sequential circuit does, by completing the table below.

The bottom counter (a, b, c) counts up from 0 to 7 and then back to 0. It outputs 0 for one clock period, 1 for two clock periods, 2 for three clock periods, 3 for four clock periods, 4 for one clock period, 5 for two clock periods, 6 for three clock periods and 7 for four clock periods. It repeats this continuously. If the comparator were a 3-bit comparator, then the number of clock periods it would output a value would be one greater than the value all the time.

Time	EQ	y2	y1	y0	a	b	c
t0	1	0	0	0	0	0	0
t1	0	0	0	0	0	0	1
t2	1	0	0	1	0	0	1
t3	0	0	0	0	0	1	0
t4	0	0	0	1	0	1	0
t5	1	0	1	0	0	1	0
t6	0	0	0	0	0	1	1
t7	0	0	0	1	0	1	1
t8	0	0	1	0	0	1	1
t9	1	0	1	1	0	1	1

Time	EQ	y2	y1	y0	a	b	c
t10	1	0	0	0	1	0	0
t11	0	0	0	0	1	0	1
t12	1	0	0	1	1	0	1
t13	0	0	0	0	1	1	0
t14	0	0	0	1	1	1	0
t15	1	0	1	0	1	1	0
t16	0	0	0	0	1	1	1
t17	0	0	0	1	1	1	1
t18	0	0	1	0	1	1	1
t19	1	0	1	1	1	1	1
t20	1	0	0	0	0	0	0

**Q7)** Consider a **sequential** circuit with a clock input and **four** outputs. The black-box view and implementation of this sequential circuit with the 4-bit counter discussed **in class** are shown below.



Determine what this sequential circuit does (the **purpose**) by continuing with the above table and showing the values for **17** clock periods. Note again that this counter is the 4-bit counter studied **in class**.

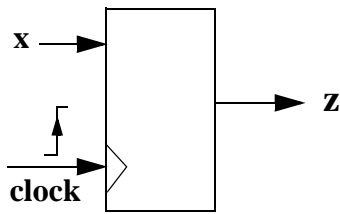
**A7)** The table is continued below :

Time	Load	d c b a
t0	1	1 0 0 1
t1	0	1 0 1 0
t2	0	1 1 0 0
t3	0	1 1 1 0
t4	1	0 0 0 0
t5	1	0 0 0 1
t6	1	0 0 1 0
t7	1	0 0 1 1
t8	1	0 1 0 0

Time	Load	d c b a
t9	1	0 1 0 1
t10	1	0 1 1 0
t11	1	0 1 1 1
t12	0	1 0 0 0
t13	0	1 0 1 0
t14	0	1 1 0 0
t15	0	1 1 1 0
t16	1	0 0 0 0

The sequential circuit counts up by 1 if the current count is odd or below 8. Otherwise, it counts up by 2.

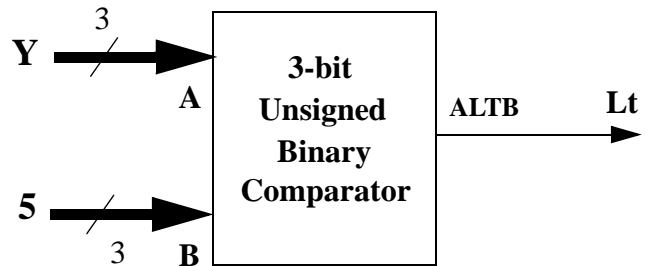
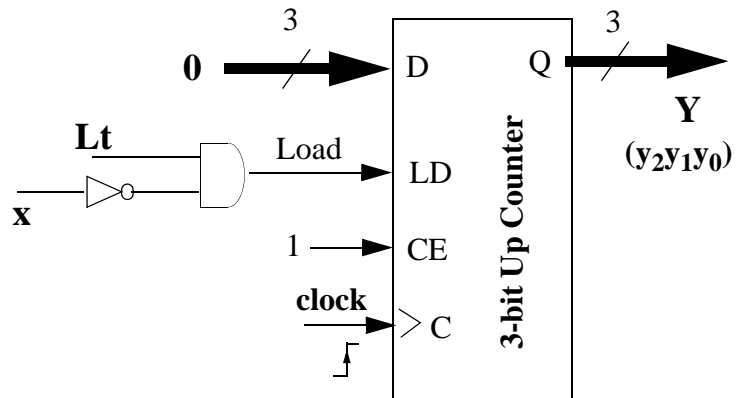
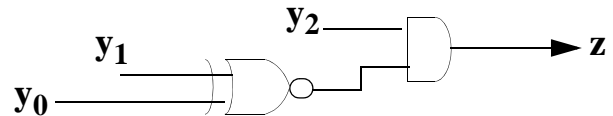
**Q8)** Consider a **sequential** circuit with one input and one output. The black-box view, implementation of this sequential circuit with a 3-bit counter and the operation table of the counter are shown below :



Time	x	Y	Load	z
t0	0	0	1	0
t1	1			
t2	0			
t3	1			
t4	1			
t5	1			
t6	1			
t7	1			
t8	1			
t9	1			
t10	1			
t11	1			
t12	1			
t13	0			
t14	0			
t15	1			
t16	1			

3-bit Up Counter Operation Table

Load	CE	C	Operation
1	1	↑	Load
0	1	↑	Count up
0	0	X	Not stored
0	X	0	Not stored



Determine what this sequential circuit does (the **purpose**) by continuing with the above table and showing the values for **17** clock periods. If you **cannot** figure out the purpose, just write "I cannot determine the purpose."

**A8)** The table is continued below.

Time	x	Y	Load	z
t0	0	0	1	0
t1	1	0	0	0
t2	0	1	1	0
t3	1	0	0	0
t4	1	1	0	0
t5	1	2	0	0
t6	1	3	0	0
t7	1	4	0	1
t8	1	5	0	0

Time	x	Y	Load	z
t9	1	6	0	0
t10	1	7	0	1
t11	1	0	0	0
t12	1	1	0	0
t13	0	2	1	0
t14	0	0	1	0
t15	1	0	0	0
t16	1	1	0	0

The purpose of this sequential circuit is that it is a sequence detector that checks for 5 (five) consecutive 1s :

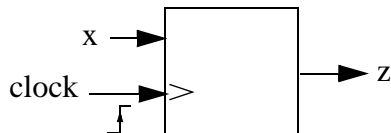
$\xrightarrow{\text{Time}}$   
 x 1 1 1 1 1 X X X  
 z 0 0 0 0 1 0 0 1

It outputs a 1 for one clock period when it receives the fifth consecutive 1. Then, it outputs two 0s and a 1. Then, it starts checking again. Thus, it has a cycle of eight (8) clock periods, meaning it checks 8-bit sequences. If it receives an incorrect bit, it waits until the current 8-bit sequence is over and then starts checking for a new 8-bit sequence. There are **no** overlapped sequences. Note that this circuit with a counter and a comparator has the identical purpose of the sequential circuit in Question Q11 of Homework III with two exceptions :

$\xrightarrow{\text{Time}}$   
 x 1 1 1 1 1 X X X  
 z 0 0 0 0 0 1 1 0

**This problem shows how one can convert a sequential circuit with a few flip-flops to a sequential circuit with registers, counters and shift registers as mentioned in class and in the lab.**

**Q9)** Consider the following 1-input, 1-output sequential circuit and its textual input/output relationship :

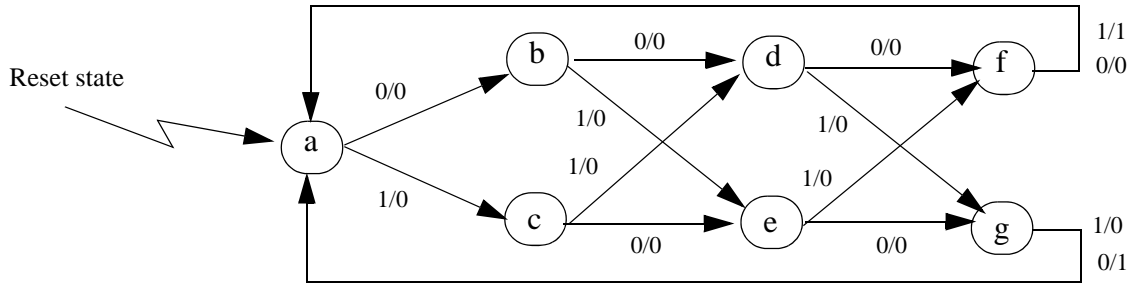


The sequential circuit checks every **nonoverlapping** 4-bit sequence. If a 4-bit sequence has odd number of 1s, it outputs a 1 for one clock period when it receives the last bit of the current sequence.

Obtain the state diagram of this sequential circuit as discussed in class. Is this a Mealy or Moore circuit ? Why ? Is this a finite memory or non-finite memory sequential circuit ? Why ?

**A9)** The state diagram is below.

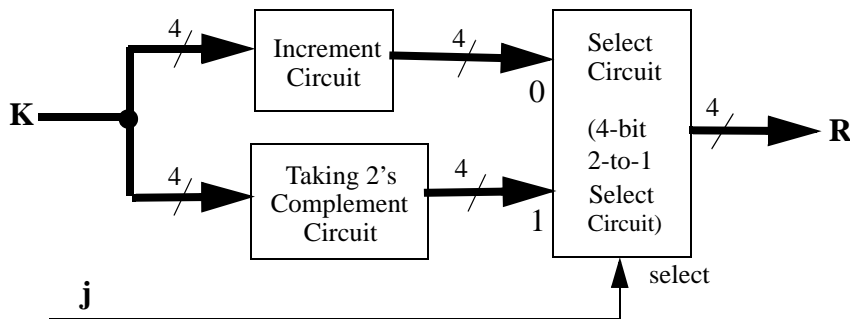
This is a Mealy circuit since the output depends on the input as seen when the state changes from “f” or “g” to “a.” This is a non-finite memory circuit since it does not have the standard state table and state diagram.



**Q10)** Consider the following 5-input, 4-output circuit :



Its block partitioning is shown below :



Analyze the circuit to obtain its purpose. In order to do that first, obtain the operation table based on the given blocks above. For each block, there is a major operation on the operation table. Based on the major operations derive the textual input-output relationship and then the purpose of the circuit.

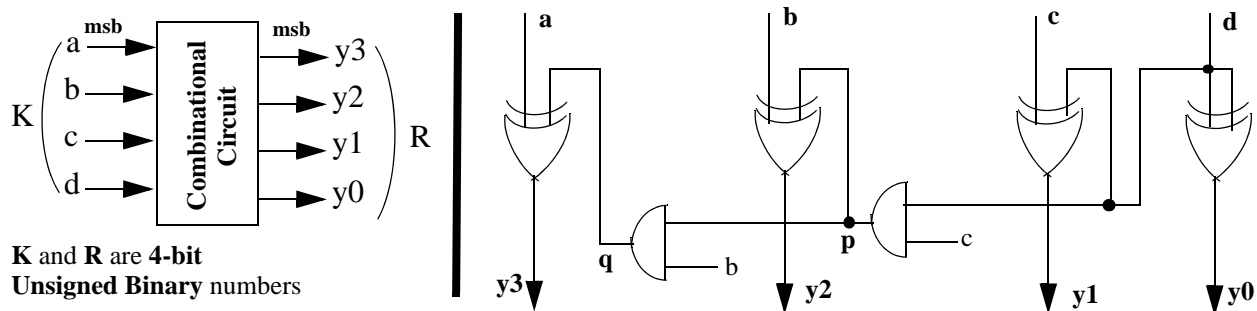
**A10)** The operation table and the blocks are as follows :

Input	Operation
$j = 0$	$R = (K + 1)$
$j = 1$	$R = \overline{K}^2$

From the operation table we see that the major operations are (1) incrementing  $K$  (that is,  $K + 1$ ), (2) taking the 2's complement of  $K$  ( $\overline{K}^2$ ) and (3) selecting the output of one of these two operations. Hence, we have one block for each major operation

This circuit is an Arithmetic Unit that increments or negates a 4-bit number input.

**Q11)** Consider the following 4-input, 4-output circuit and its gate network :



Analyze the gate network to obtain the **operation table** and then determine its **purpose**. In order to do that continue with the truth table below from which the purpose can be obtained :

<b>K</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>p</b>	<b>q</b>	<b>y3</b>	<b>y2</b>	<b>y1</b>	<b>y0</b>	<b>R</b>
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	1	0	2
...	...	...	...	...	...	...	...	...	...	...	...

**A11)** We continue with the truth table below :

<b>K</b>	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>p</b>	<b>q</b>	<b>y3</b>	<b>y2</b>	<b>y1</b>	<b>y0</b>	<b>R</b>
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	1	0	2
2	0	0	1	0	0	0	0	0	1	0	2
3	0	0	1	1	1	0	0	1	0	0	4
4	0	1	0	0	0	0	0	1	0	0	4
5	0	1	0	1	0	0	0	1	1	0	6
6	0	1	1	0	0	0	0	1	1	0	6
7	0	1	1	1	1	1	1	0	0	0	8
8	1	0	0	0	0	0	1	0	0	0	8
9	1	0	0	1	0	0	1	0	1	0	10
10	1	0	1	0	0	0	1	0	1	0	10
11	1	0	1	1	1	0	1	1	0	0	12
12	1	1	0	0	0	0	1	1	0	0	12
13	1	1	0	1	0	0	1	1	1	0	14
14	1	1	1	0	0	0	1	1	1	0	14
15	1	1	1	1	1	1	0	0	0	0	0

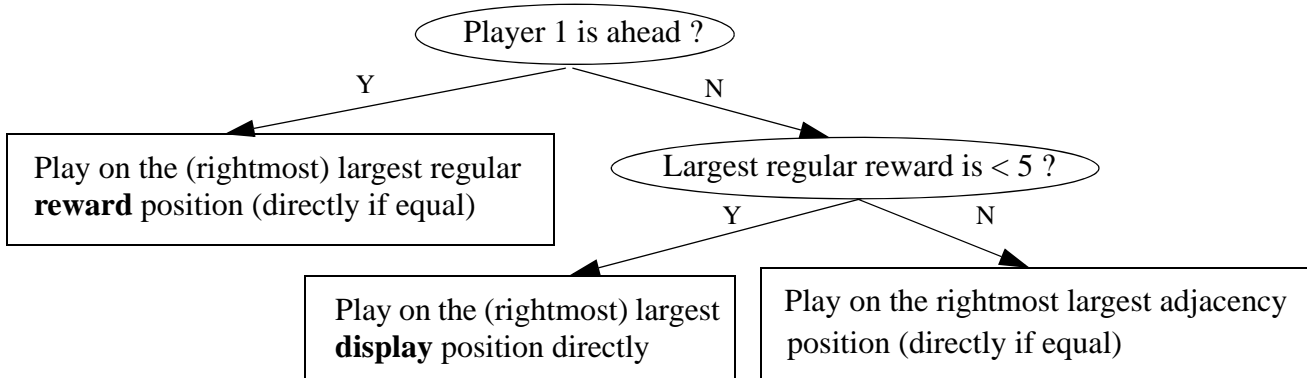
Operation Table :

Situation	Operation
K is even	$R = K$
K is odd	$R = K + 1$

The purpose is that when K is even, the output is equal to K. Otherwise, the output is equal to K + 1.

In the case of K = 15, the output is 0 which is normal since  $15 + 1 = 16$  that requires 5 bits and its rightmost four bits are zero.

Q12) Consider the Ppm term project. Assume that the playing strategy of the machine player is as follows :



Consider the table below that shows the random digit, position displays before and after the machine player plays, whether Player 1 is ahead or not, whether the random digit is played directly or added, the number of adjacencies, the points earned by the machine player and whether the machine player plays again.

RD	Displays Before Play				Displays After Play				P1 Ahead	D/A	The Adjacency	Points Earned (Decimal)	Machine player plays again
	PD3	PD2	PD1	PD0	PD3	PD2	PD1	PD0					
3	F	F	F	F	F	F	F	3	No	D	0	3	No
5	7	C	7	7					Yes				
2	0	2	2	D					No				
9	E	6	F	C					No				
4	A	E	5	9					Yes				
1	0	F	F	F					Yes				

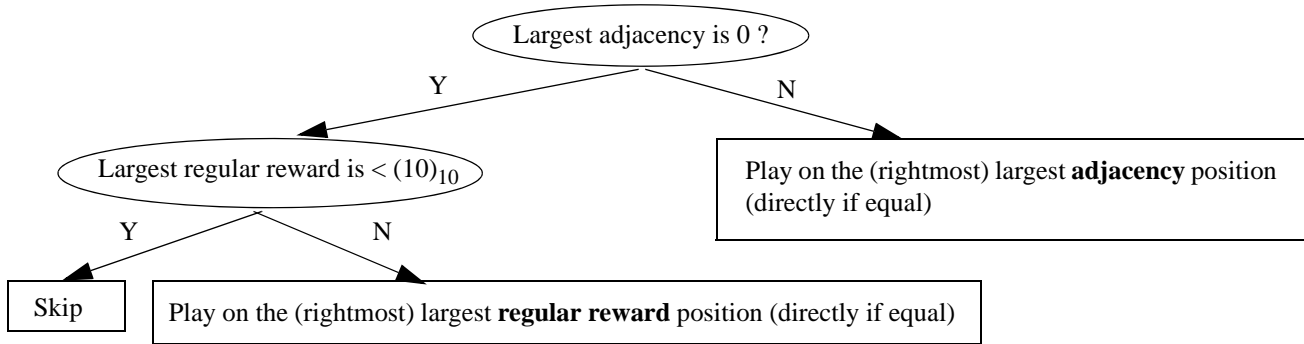
Assume that the code is C1. The first row shows how the random digit is played by the machine player. A circle is drawn on a position if it is played on. The meaning of D/A is Direct/Add which is whether the player plays the random digit directly on a position or by adding to a position.

Note that the cases are independent of each other. That is, they do not necessarily follow each other with respect to time. Complete the table.

A12) The table is completed below. Those entries to be filled out are shown in bold. The positions played on are shown by circles :

RD	Displays <b>Before</b> Play PD3 PD2 PD1 PD0	Displays <b>After</b> Play PD3 PD2 PD1 PD0	P1 Ahead	D/A	The Adjacency	Points Earned (Decimal)	Machine player plays again
3	F F F F	F F F (3)	No	D	0	3	No
5	7 C 7 7	7 C (C) 7	Yes	A	1	120	Yes
2	0 2 2 D	0 2 2 (2)	No	D	2	8	Yes
9	E 6 F C	E (F) F C	No	A	1	30	Yes
4	A E 5 9	(E) E 5 9	Yes	A	1	28	Yes
1	0 F F F	0 F F (1)	Yes	D	0	9	No

**Q13)** Consider the **Ppm** term project. Assume that the playing strategy of the **machine** player is as follows :



Consider the following table that shows the random digit, position displays **before** and **after** the **machine** player plays, whether the random digit is played directly or added, the number of adjacencies, the points earned by the **machine** player and whether the machine player plays again :

RD	Displays <b>Before</b> Play PD3 PD2 PD1 PD0	Displays <b>After</b> Play PD3 PD2 PD1 PD0	D/A	The Adjacency	Points Earned (Decimal)	Machine player plays again
1	E F E F	E F (F) F	A	2	60	Yes
0	0 0 0 0					
7	0 0 0 0					
2	1 F 1 1					
7	F 8 E 7					
6	C 6 6 6					

Assume that the code is **C7**. The first row shows how the random digit is played by the **machine** player. A circle is drawn on a position if it is played on. The meaning of **D/A** is Direct/Add which is whether the machine player plays the random digit **directly** on a position or by **adding** to a position. Note that the cases are **independent** of each other.

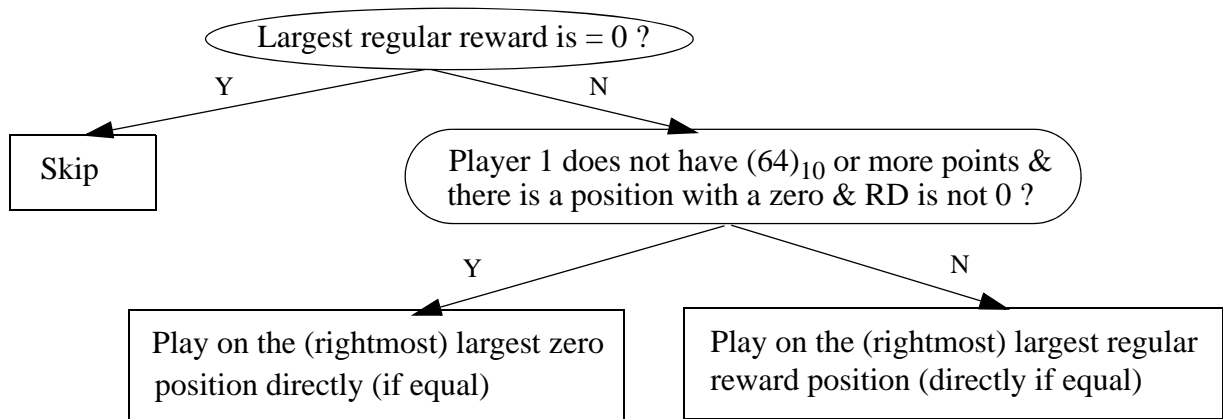
That is, they do **not** necessarily follow each other with respect to time. Complete the table.

**A13)** The table is completed below. Those entries to be filled out are shown in bold. The positions played on are shown by circles :

RD	Displays <b>Before</b> Play PD3 PD2 PD1 PD0	Displays <b>After</b> Play PD3 PD2 PD1 PD0	D/A	The Adjacency	Points Earned (Decimal)	Machine player plays again
1	E F E F	E F (F) F	A	2	60	Yes
0	0 0 0 0	0 0 0 (0)	D	3	0	Yes
7	0 0 0 0	0 0 0 0	Skip	Skip	Skip	Skip
2	1 F 1 1	1 (1) 1 1	A	3	8	Yes
7	F 8 E 7	F 8 (7) 7	D	1	14	Yes
6	C 6 6 6	(6) 6 6 6	D	3	48	Yes

Note that the machine player does not check for code digits and so misses large reward points on rows 3, 5 and 6. The random digits on these rows enable it to play the code digits.

**Q14)** Consider the **Ppm** term project. Assume that the playing strategy of the **machine** player is as shown below.



Consider also the table below that shows the random digit, Player 1 points in decimal, position displays **before** and **after** the **machine** player plays, whether the random digit is played directly or added, the number of adjacencies, the points earned by the **machine** player and whether the machine player plays again.

Assume that the code is 79. A circle is drawn on a position if it is played on. The meaning of **D/A** is Direct/Add which is whether the player plays the random digit **directly** on a position or by **adding** to a position. Note that the cases are **independent** of each other. That is, they do not necessarily follow each other with respect to time. Complete the table.

RD	P1PT (Decimal)	Displays <b>Before</b> Play PD3 PD2 PD1 PD0	Displays <b>After</b> Play PD3 PD2 PD1 PD0	D/A	The Adjacency	Points Earned (Decimal)	Machine player plays again
3	72	0 C 0 9					
7	29	F 8 7 7					
1	53	F E 0 E					
4	61	7 7 3 9					
0	238	F E 7 9					

**A14)** The table is completed below. The displays played on are shown in bold and circled :

RD	P1PT (Decimal)	Displays <b>Before</b> Play PD3 PD2 PD1 PD0	Displays <b>After</b> Play PD3 PD2 PD1 PD0	D/A	The Adjacency	Points Earned (Decimal)	Machine player plays again
3	72	0 C 0 9	0 <b>(F)</b> 0 9	A	0	15	N
7	29	F 8 7 7	F <b>(F)</b> 7 7	A	1	30	Y
1	53	F E 0 E	F E <b>(1)</b> E	D	0	1	N
4	61	7 7 3 9	7 7 <b>(7)</b> 9	A	2	149	Y
0	238	F E 7 9	<b>(F)</b> E 7 9	A	0	15	N

We observe that the machine player misses a chance to earn code reward points when RD is 7 and 0. But, by chance, it earns code reward points when RD is 4.