Switching Circuits & Logic Design, Fall 2010

Final Examination (01/14/2011, 3:30pm~5:20pm)

Problem 1: (20 points)

Consider the S-R FF implemented by two S-R latches as shown in Fig. 1.

A. (10 points) Please complete P and Q values in the timing diagram (Initial values: P=1 and Q=0 at t=0).

B. (10 points) Please identify fault output(s) (if any) and how to remove the fault output(s)?

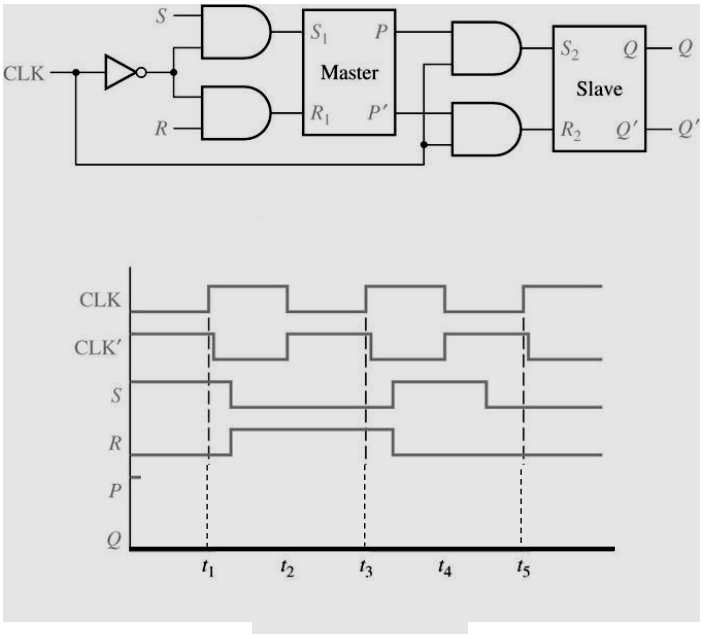
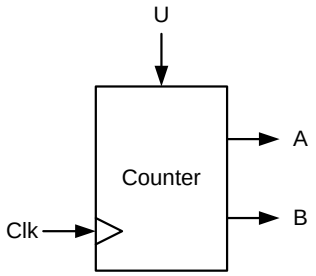


Fig. 1 SR FF implemented by two SR latches and the timing diagram.

Problem 2: (20 points)

A counter has one control pin, U. The counting sequences are as follows:



U	Sequence (AB)
0	00, 01, 10, 11, 00, 01, 10, 11, 00, …
1	00, 11, 01, 10, 00, 11,01, 10, 00, …

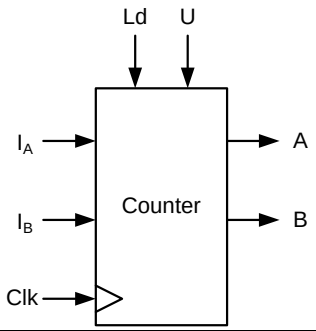
A. (10 points) Please complete the following State table and Excitation table by using an S-R flip-flop for A and a J-K flip-flop for B:

(Please write your answer with the same form as the following table)

AB	U=0		U=1		U=0		U=1		U=0		U=1	
	A^+	B^+	A^+	B^+	S_A	R_A	S_A	R_A	J_B	K_B	J_B	K_B
00												
01												
11												
10												

B. (5 points) Realize the counter with one positive-edge triggered S-R flip-flop, one positive-edge triggered J-K flip-flop, and basic logic gates (AND, OR, NOT).

C. (5 points) Modify the circuit you designed to have a new control pin Load (**Ld**) which function is described in the following table.



Ld	U	A^+B^+
1	X	I_AI_B
0	0/1	As described in the above table

<u>Problem 3</u>: (20 points)

The following block diagram (Fig 3) shows how a ROM and two D flip-flops are connected to realize the state transitions listed in Table 1. Let S_0=00, S_1=01, S_2=10, and S_3=11. Specify the ROM contents by filling up the binary numbers in Table 2 (where ? = 1 or 0).

Present State	Next State X_1X_2 = 00	01	10	11	Present Output (Z_1Z_2) X_1X_2 = 00	01	10	11
S_0	S_3	S_2	S_1	S_0	00	10	11	01
S_1	S_0	S_1	S_2	S_3	10	10	11	11
S_2	S_3	S_0	S_1	S_1	00	10	11	01
S_3	S_2	S_2	S_1	S_0	00	00	01	01

Table 1.

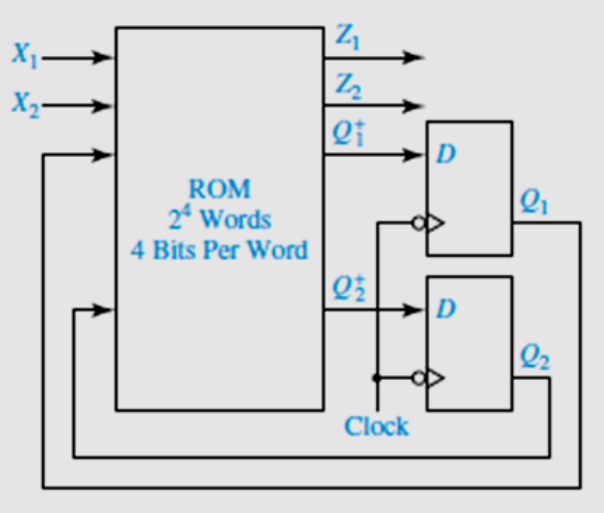


Fig 3.

X_1	X_2	Q_1	Q_2	Q_1^+	Q_2^+	Z_1	Z_2
0	0	0	0	?	?	?	?
0	0	0	1	?	?	?	?
0	0	1	0	?	?	?	?
0	0	1	1	?	?	?	?
0	1	0	0	?	?	?	?
0	1	0	1	?	?	?	?
0	1	1	0	?	?	?	?
0	1	1	1	?	?	?	?
1	0	0	0	?	?	?	?
1	0	0	1	?	?	?	?
1	0	1	0	?	?	?	?
1	0	1	1	?	?	?	?
1	1	0	0	?	?	?	?
1	1	0	1	?	?	?	?
1	1	1	0	?	?	?	?
1	1	1	1	?	?	?	?

Table 2. Assign binary numbers for the ROM contents (where ? = 0 or 1)

Problem 4: (20 points)

The (**8,4,-2,-1)** BCD code is similar to the **8-4-2-1** BCD code, except that the weights are negative for the two least significant bit positions. For example, **0111** in **(8, 4,-2,-1)** code represents: 8×**0** + 4×**1** + (–2)×**1** + (–1)×**1** = 1.

Design a Mealy sequential circuit to convert (8,4,-2,-1) code to 8-4-2-1 code. The input and output should be serial with the least significant bit (LSB) first. The input **X** represents an (8,4,-2,-1) coded decimal digit, and the output **Z** represents the

corresponding 8-4-2-1 BCD code. ***After four time steps, the sequential circuit should be RESET to the starting state regardless of the input sequence***.

Answer the following three questions.

A. (5 points) Fill up the **?** in the conversion table (Table 3) with 1 or 0.

B. (5 points) Fill up the **?** in State Table (Table 4) with ***state notations*** or ***a don't care state (d)***. You must use a '*d*' in your answer if the *Next State* is ***don't care*** as specified and/or indicated in Table 3.

C. (5 points) Fill up the **#** in State Table (Table 4) with ***0, 1 or d (don't care)***. You must use a '*d*' in your answer if the *Present Output* is ***don't care*** as specified and/or indicated in Table 3.

D. (5 points) Write down the sets of equivalent states in $\mathbf{t_2}$ and $\mathbf{t_3}$ in the State Table (Table 4). For example, write down your answer as $H \equiv I \equiv J$, $M \equiv N \equiv P$, …

X Input (8,4,-2,-1)	Z Output (8,4,2,1)
t_3 t_2 t_1 t_0	t_3 t_2 t_1 t_0
0 0 0 0	? ? ? ?
0 0 0 1	d d d d
0 0 1 0	d d d d
0 0 1 1	d d d d
0 1 0 0	? ? ? ?
0 1 0 1	? ? ? ?
0 1 1 0	? ? ? ?
0 1 1 1	? ? ? ?
1 0 0 0	? ? ? ?
1 0 0 1	? ? ? ?
1 0 1 0	? ? ? ?
1 0 1 1	? ? ? ?
1 1 0 0	d d d d
1 1 0 1	d d d d
1 1 1 0	d d d d
1 1 1 1	? ? ? ?

Table 3 Conversion table from (8,4,-2,-1) code to 8-4-2-1 code

State table for code converter

Time	Input Sequence Received (Least Significant Bit First)	Present State	Next State X=0	Next State 1	Present Output (Z) X=0	Present Output (Z) 1
t_0	Reset	A	B	C	#	#
t_1	0	B	D	F	#	#
	1	C	E	G	#	#
t_2	00	D	?	?	#	#
	01	E	?	?	#	#
	10	F	?	?	#	#
	11	G	?	?	#	#
t_3	000	H	?	?	#	#
	001	I	?	?	#	#
	010	J	?	?	#	#
	011	K	?	?	#	#
	100	L	?	?	#	#
	101	M	?	?	#	#
	110	N	?	?	#	#
	111	P	?	?	#	#

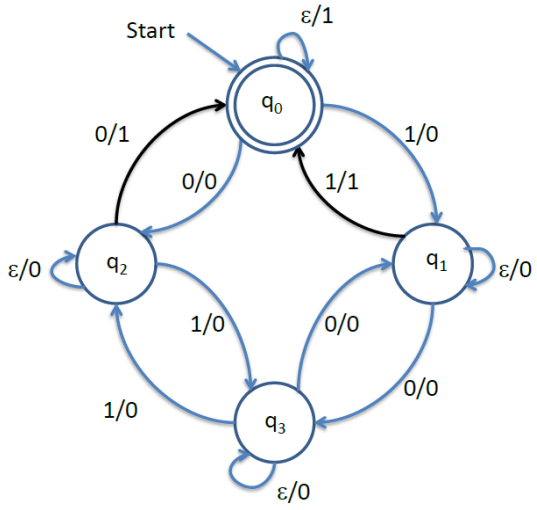
? = H, I, J, K, L M, N, P, A, d(don't care) ***# = 0, 1, d***

Table 4 State Table for the (8,4,-2,-1) to 8-4-2-1 code conversion

<u>Problem 5</u>: (25 points)

(Ch.14, Sequence Detector) You are asked to design a Sequence Detector with the following state graph, where the double-circle state represents a *detected state*(here, it is q_0). This detector has two inputs (input sequence X and flag F) and one output (detected signal D). Assume we always start from state q_0. If and only if the read-in data x in the input sequence X reaches the *detected state* (here, it is q_0) and no more data in the sequence (expressed by an **ending symbol** of ε), we will stop reading

and output a logic "1" to the detected signal D. Otherwise we will continue our detection process until the end of input sequence. To simplify the design, we assume that when we read in the **ending symbol** ε, we will have the flag F=1, otherwise F=0 (that is, you need not to design this part of circuit).



x/Z (F not shown)

For example, when we apply an input sequence X=1010ε to the Detector, we have:

X = 1 0 1 0 ε

Z = 0 0 0 1 1 (note: the function of Z will be given in Part C, basically D = F·Z)

F = 0 0 0 0 1

D = 0 0 0 0 1

When we apply an input sequence X=10010110ε, we have:

X = 1 0 0 1 0 1 1 0 ε

Z = 0 0 0 1 0 0 0 1 1

F = 0 0 0 0 0 0 0 0 1

D = 0 0 0 0 0 0 0 0 1

A. (5 points) What sequence of input (note: most significant bit first) will be detected by this Detector (multiple choices)?

(1) 101011ε; (2) 101111ε; (3) 010001ε; (4) 100110ε; (5) 010100ε.

B. (5 points) What is the function of this Detector (multiple choices)?

(1) Detect an even number of consecutive 1's;

(2) Detect an odd number of 1's;

(3) Detect an even number of consecutive 0's;

(4) Detect an odd number of 0's;

(5) Detect both an even number of 1's and an even number of 0's.

C. (10 points) Given the above state graph, try to design a Mealy finite state machine that reads in an Input x from the input sequence X and generates an

Output Z. And Z=1 indicates that we are entering the accepted state q_0 in the next state or we stop at the state q_0. Write down all the Next State functions δ of this machine (Hint: $\delta(q_0,0)=q_2$, …). Complete the following state table for this Mealy machine.

Present State (PS)	Next State (NS)			Output Z		
	x=0	x=1	x=ε	x=0	x=1	x=ε
q_0	q_2					
q_1						
q_2						
q_3						

D. (**Bonus** 5 points) How do you design this Sequence Detector from the table obtained in Part C by using T flip-flops and other necessary logic gates? (Hint: We need more than one bit to encode x; D is a function of F and Z, i.e., D = F·Z)

Bonus: (5 points)

The following code is a sample of Verilog module. Please find out the **five** errors and **correct** them.

```
============================================================
module my_module (out_display , in_data)
input   in_data;
or o1( w1[0] , in_data[0] , in_data[1] );
or o2( w1[1] , in_data[2] , in_data[3] );
and a1( out_display , w1[0] , w1[1] );
============================================================
```