Problem Set 2 Solutions

Problem 1.

(a) F = (A*B) + (A+D)*(B xor C)* [/ (B*D)]

Look at the simplest terms first, i.e. if both A and B are 1, then F is 1 independently of the rest of the circuit. Similarly, if A and D are both 0, then the second expression is 0 independently of B and C.

(b)

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

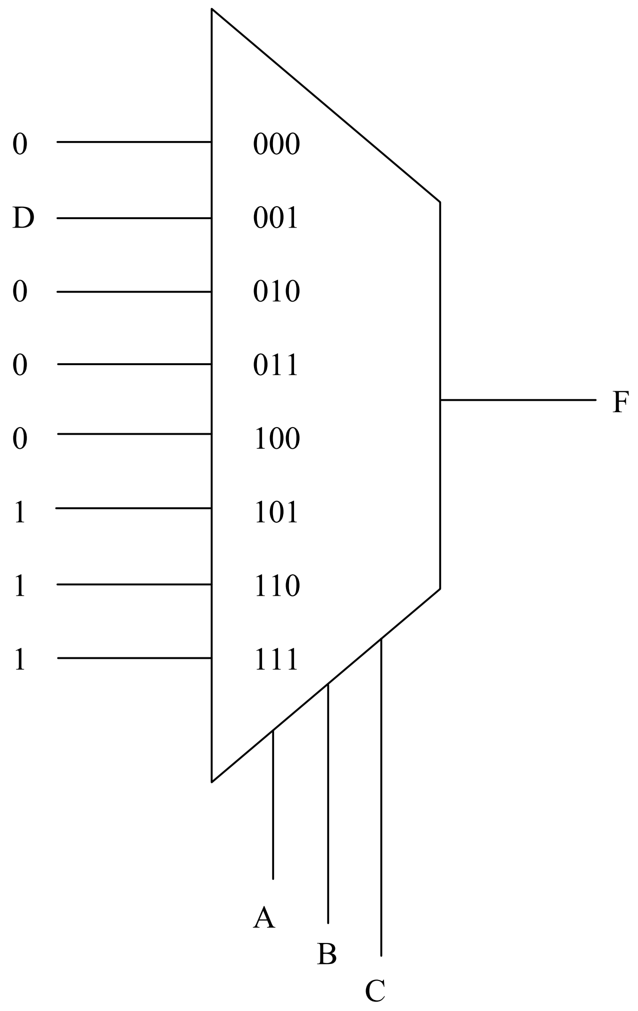
(c)

AB CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	1	0	1	1
10	0	0	1	1

MSP: F = A*B + A*C + (¬B)*C*D

(d) The expression above is free from static hazards.

(e)



Problem 2.

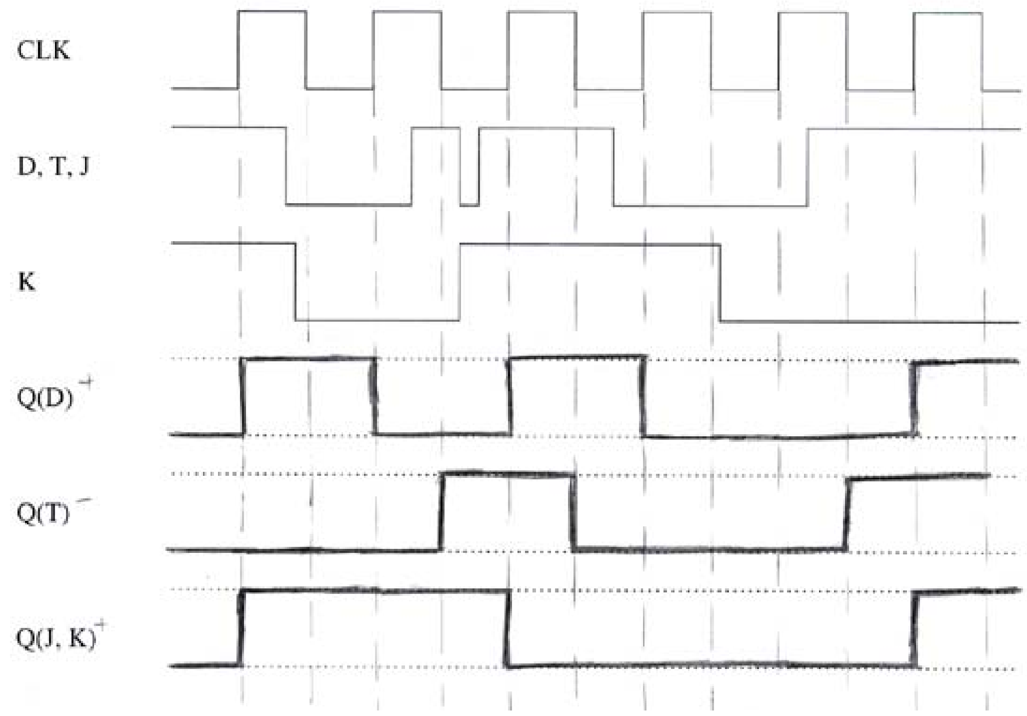
The timing diagram below assumes that flip-flops have no propagation delay, that is $P_d = 0$ ns, and consequently all state changes occur on the clock edge. If some non-zero propagation delay was assumed, the timing diagram would be shifted to the right by P_d.

We just need to remember our flip-flops to create the timing diagram, that is

D	Q_{n-1}	Q_n
0	0	0
0	1	0
1	0	1
1	1	1

T	Q_{n-1}	Q_n
0	0	0
0	1	1
1	0	1
1	1	0

J	K	Q_{n-1}	Q_n
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0



Problem 3.

(a) The ‘393 is called a ripple counter because the effect of a clock edge has to “ripple” through the counter to cause a change in the value. Only the least significant bit is triggered by the clock; the rest have to be triggered by the bit that precedes it. A ‘163 is a synchronous counter because ALL bits of the counter are triggered directly by the clock.

(b) Wiring the RCO of the first counter to the clock input of the second counter breaks the synchronicity of the counter. The counting of the second counter is no longer synchronized to the clock, but triggered by the RCO of the first counter. A better way to do this is to wire up the RCO of the first counter to enable the second counter. This way both counters are synchronized to the clock.

(c) The difference between /ENT and /ENP is that /ENT is an enable for both counting and the RCO, whereas /ENP is just a count enable.

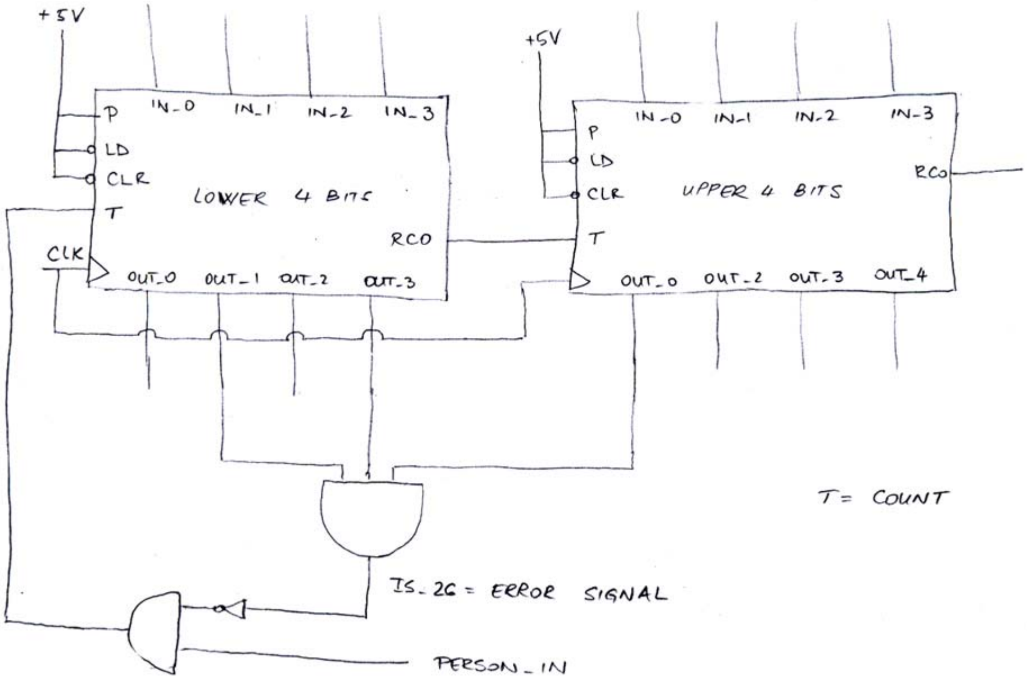
(d) Let’s assume that signal PERSON_IN is 1 for one clock cycle every time a person walks into the classroom. Also assume that signal IS_26 asserts whether the counter reached a value of 26, that is, IS_26 is 1 when the counter has reached 26 and is 0 when the counter has reached a value less than 26.

Then we require our design to exhibit the following behavior:

SIGNAL_IN	IS_26?	Count	Error
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	1

The counter does not count either when the counter has reached the value of 26 or when no new person enters the classroom. That is, when a person walks in and the counter hasn’t reached the value of 26, the counter just counts up by one (that is Count = 1). If the counter has reached the value of 26, then the counter does not count up by 1 when the next person walks in and asserts the Error signal.

So the following circuit will satisfy the above constraints:



$26 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 11010$ IN BINARY