# [CS M51A F14] Solution to Assignment 5

Due: 12/05/14

TA: Yang Lu (yangluphil@gmail.com)

---

## Problems (40 points total)

### Problem 1 (6 points)

1. Create a D flip-flop using a JK flip-flop. Use the excitation tables on page 221 of the textbook.

   ***Solution*** First, we know that a D flip-flop works like this:

| $Q(t)$ | $D(t)=0$ | $D(t)=1$ |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 1 |
| | $Q(t+1)$ | |

   To make the transition from $Q(t)$ to $Q(t+1)$, we need to figure out the inputs to our JK flip-flop. The inputs should be a combinational circuit with inputs $Q(t)$ and $D(t)$.

| $Q(t)$ | $Q(t+1)=0$ | $Q(t+1)=1$ |
|---|---|---|
| 0 | 0- | 1- |
| 1 | -1 | -0 |
| | $J(t)K(t)$ | |

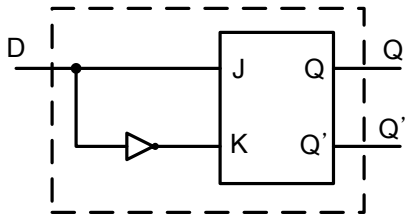
   As we know the excitation table of a JK flip-flop is as shown in the above table, we can write:

| $Q(t)$ | $D(t)=0$ | $D(t)=1$ | $D(t)=0$ | $D(t)=1$ |
|---|---|---|---|---|
| 0 | 0 | 1 | 0- | 1- |
| 1 | 0 | 1 | -1 | -0 |
| | $Q(t+1)$ | | $J(t)K(t)$ | |

   Looking at this table we can write:

$$\begin{aligned} J(t) &= D(t) \\ K(t) &= D(t)' \end{aligned}$$

   The final D flip-flop would look like this.




2. Create a T flip-flop using an SR flip-flop.

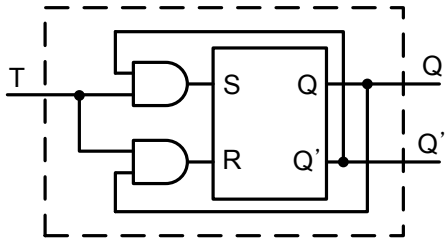
   ***Solution*** The excitation table of an SR flip-flop is as shown:

| $Q(t)$ | $Q(t+1)=0$ | $Q(t+1)=1$ |
|---|---|---|
| 0 | 0- | 10 |
| 1 | 01 | -0 |
| | $S(t)R(t)$ | |

Following a similar procedure as above, we get the following:

| $Q(t)$ | $T(t)=0$ | $T(t)=1$ | $T(t)=0$ | $T(t)=1$ |
|---|---|---|---|---|
| 0 | 0 | 1 | 0- | 10 |
| 1 | 1 | 0 | -0 | 01 |
| | $Q(t+1)$ | | $S(t)R(t)$ | |

$$\begin{aligned} S(t) &= T(t)Q'(t) \\ R(t) &= T(t)Q(t) \end{aligned}$$



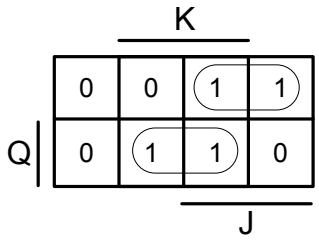

3. Create a JK flip-flop using a T flip-flop.

***Solution*** The excitation table of a T flip-flop is:

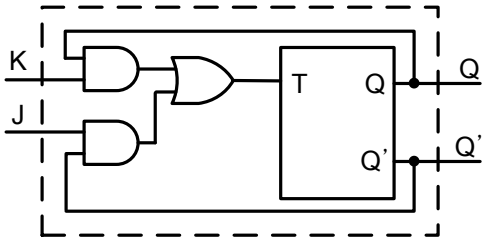
| $Q(t)$ | $Q(t+1)=0$ | $Q(t+1)=1$ |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |
| | $T(t)$ | |

Similarly, we can get:

| $Q(t)$ | $J(t)K(t)$ | | | | $J(t)K(t)$ | | | |
|---|---|---|---|---|---|---|---|---|
| | 00 | 01 | 10 | 11 | 00 | 01 | 10 | 11 |
| 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| | $Q(t+1)$ | | | | $T(t)$ | | | |

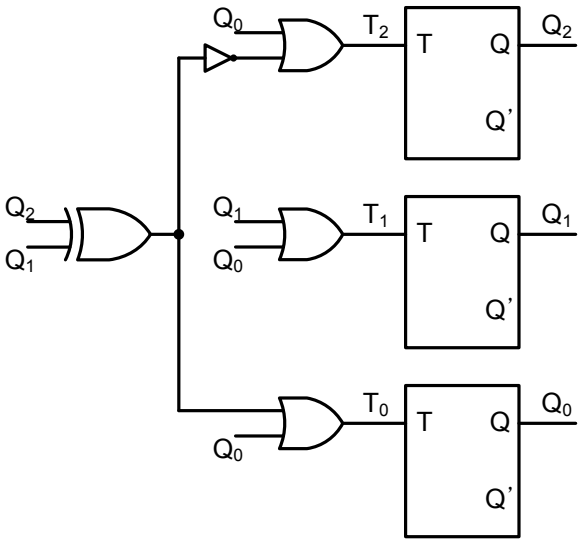



$$T(t) = Q(t)K(t) + Q'(t)J(t)$$




**Problem 2 (6 points)**

We would like to analyze the sequential system shown below. It is an autonomous counter which outputs a fixed string of numbers. The output changes at every clock cycle.




1. Write the expressions for $T_2$, $T_1$, and $T_0$.

   ***Solution*** Looking at the gate network we can write:

$$\begin{aligned} T_2 &= Q_0 + (Q_2 \oplus Q_1)' \\ &= Q_0 + Q_2' Q_1' + Q_2 Q_1 \\ T_1 &= Q_1 + Q_0 \\ T_0 &= Q_0 + Q_2 \oplus Q_1 \\ &= Q_0 + Q_2' Q_1 + Q_2 Q_1' \end{aligned}$$

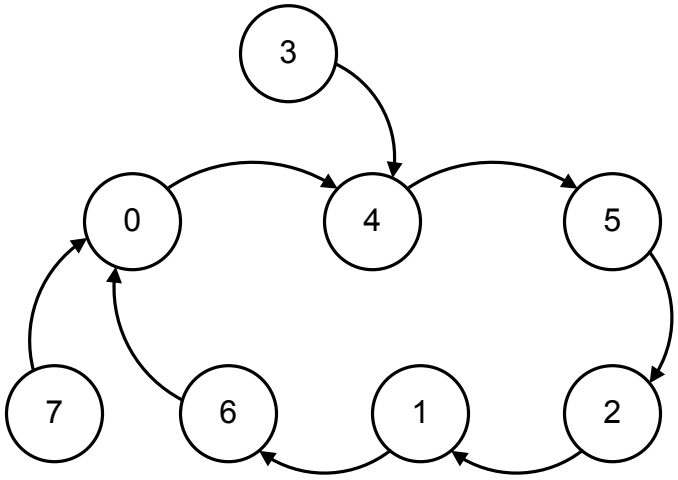
2. Write the table of $T_2$, $T_1$ and $T_0$. Using this table, show the state transition table.

   ***Solution***

| $PS$ | T | | | $NS$ | | |
|---|---|---|---|---|---|---|
| $Q_2Q_1Q_0$ | $T_2$ | $T_1$ | $T_0$ | $Q_2$ | $Q_1$ | $Q_0$ |
| 000 | 1 | 0 | 0 | 1 | 0 | 0 |
| 001 | 1 | 1 | 1 | 1 | 1 | 0 |
| 010 | 0 | 1 | 1 | 0 | 0 | 1 |
| 011 | 1 | 1 | 1 | 1 | 0 | 0 |
| 100 | 0 | 0 | 1 | 1 | 0 | 1 |
| 101 | 1 | 1 | 1 | 0 | 1 | 0 |
| 110 | 1 | 1 | 0 | 0 | 0 | 0 |
| 111 | 1 | 1 | 1 | 0 | 0 | 0 |

3. Draw the state transition diagram of the system.

   ***Solution*** Writing the state values in decimal numbers, the state diagram is as shown:

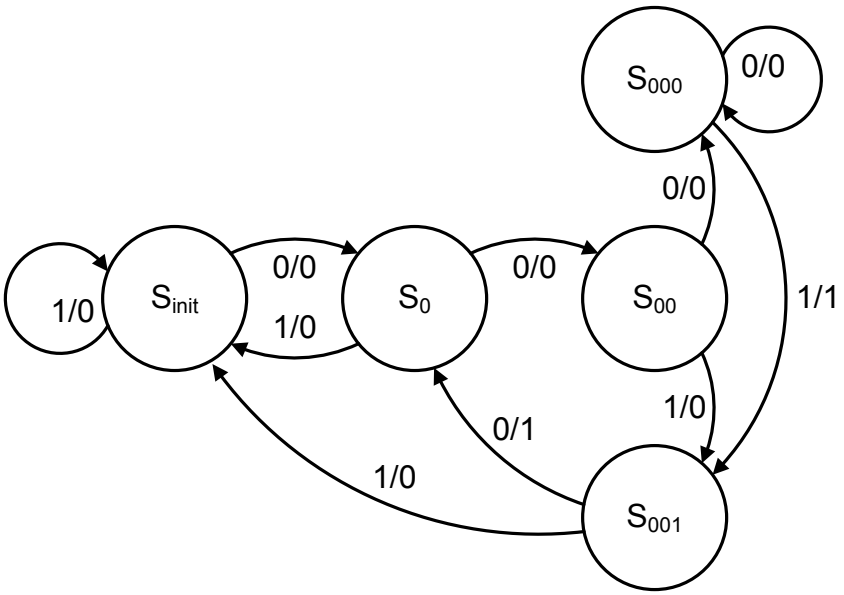



## Problem 3 (8 points)

We would like to design a pattern recognizer with a binary stream as input. The system has a binary output bit, which is 1 whenever $x(t-3,t) = 0010$ or 0001 and 0 otherwise.

1. Draw the state transition diagram. Try to minimize the number of states. (*Hint: The system can be designed using only 5 states.*)

   ***Solution*** We can create a pattern recognizer by only creating states for the states necessary for identifying the patterns, and simply moving to one of the previous states when it becomes certain that the current string will not match any valid patterns. This gives us the following state diagram.




   Alternatively, it is also possible to write a loose state description and do state minimization. It should also give you 5 states.

2. Encode the states into binary bits. From the encoding, write the state transition table and table of JK flip-flop inputs.

***Solution*** Since we have 5 states, we need 3 state bits to represent them all. We shall use the following encoding.

| | $s_2$ | $s_1$ | $s_0$ |
|---|---|---|---|
| $S_{init}$ | 0 | 0 | 0 |
| $S_0$ | 0 | 0 | 1 |
| $S_{00}$ | 0 | 1 | 0 |
| $S_{000}$ | 0 | 1 | 1 |
| $S_{001}$ | 1 | 0 | 0 |

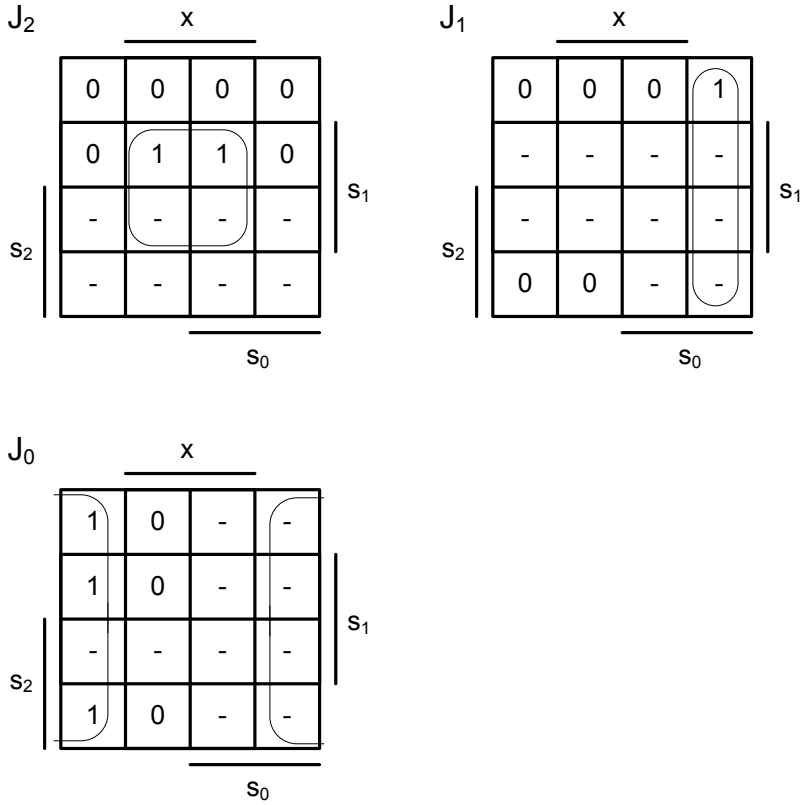
With this encoding we can get the following tables.

| $PS$ | $NS$ | | $JK$ | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $x=0$ | $x=1$ | $x=0$ | | | $x=1$ | | |
| 000 | 001,0 | 000,0 | 0- | 0- | 1- | 0- | 0- | 0- |
| 001 | 010,0 | 000,0 | 0- | 1- | -1 | 0- | 0- | -1 |
| 010 | 011,0 | 100,0 | 0- | -0 | 1- | 1- | -1 | 0- |
| 011 | 011,0 | 100,1 | 0- | -0 | -0 | 1- | -1 | -1 |
| 100 | 001,1 | 000,0 | -1 | 0- | 1- | -1 | 0- | 0- |

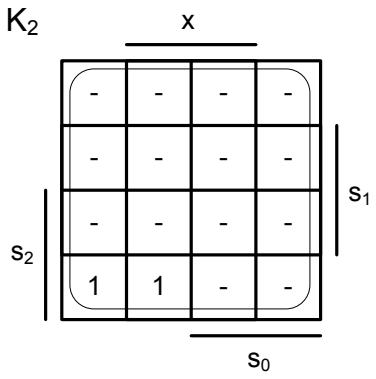
All the items not on the table are don't-cares.

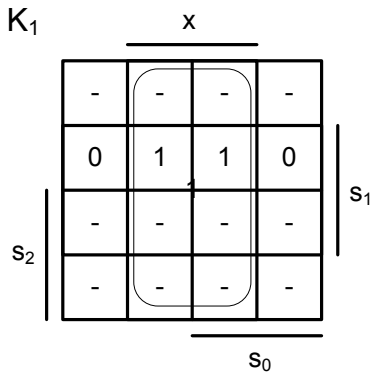
3. Using K-maps, derive the switching expressions for each flip-flop input $J_i$, $K_i$, and the output bit $z$.

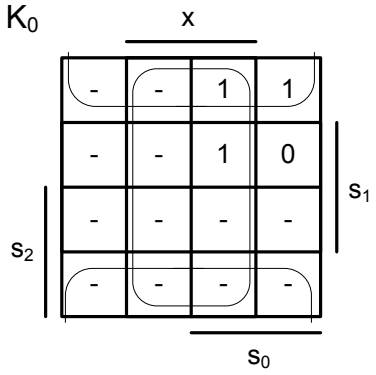
***Solution*** K-maps for Js:



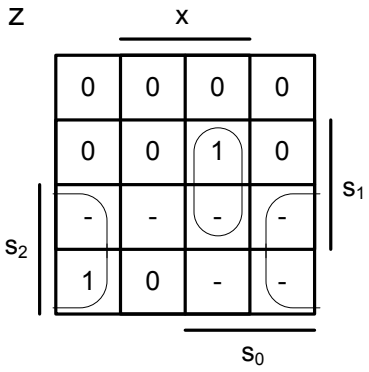

K-maps for Ks:




K-map for $z$:

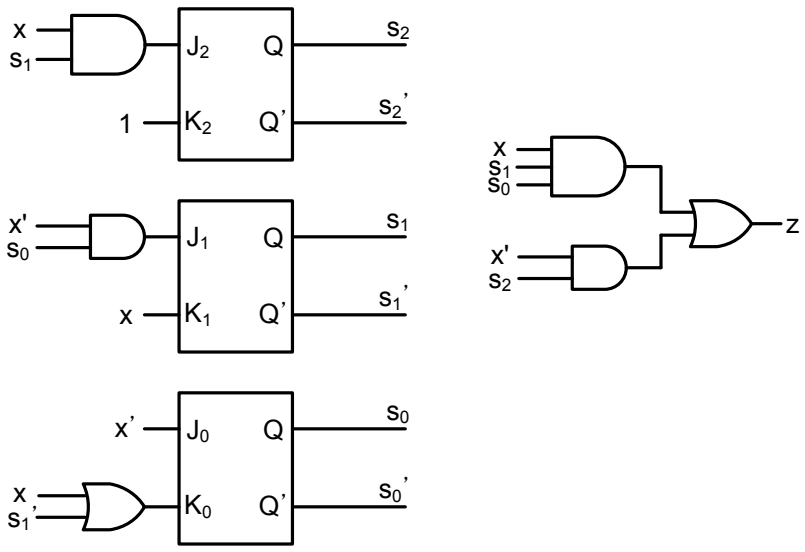



The switching expressions are therefore:

$$\begin{aligned} J_2 &= xs_1 \\ J_1 &= x's_0 \\ J_0 &= x' \\ K_2 &= 1 \\ K_1 &= x \\ K_0 &= x + {s_1}' \\ z &= xs_1s_0 + x's_2 \end{aligned}$$

4. Draw the sequential flip-flop network.

***Solution***




## Problem 4 (6 points)

We would like to put together a 12-input decoder using 4-input decoders. Let us consider the following two methods.

1. Using a tree structure, how many levels would we need? How many 4-input decoders are used in total?

   ***Solution*** At each level, we can deal with 4 inputs since our building block is 4-input decoders. Since we need to deal with 12 inputs total, this leads to $\frac{12}{4} = 3$ levels.

   The total number of 4-input decoders is:

   $$1 + 2^4 + (2^4)^2 = 273$$

2. The textbook has an example of using a coincident structure which divides the $n$-inputs into two groups using $\frac{n}{2}$-input decoders. This can be expanded to the case where we divide the $n$ inputs into $k$ groups, using $r$-input decoders. In this case, what would be the value of $k$ in terms of $n$ and $r$? How many AND gates would we need?

   ***Solution*** The textbook example divides the inputs into 2 groups and combines 2 decoder outputs, one from each, into one 2-input AND gate. Therefore, it is possible to divide the inputs into $k$ groups and combine $k$ decoder inputs into one $k$-input AND gate each. Each divided group now needs to accomodate for $\frac{n}{k}$ inputs, which can be done with $r = \frac{n}{k}$-input decoders.

   As $r = \frac{n}{k}$, $k = \frac{n}{r}$, and AND gates are needed for every combination of $r$-input decoder outputs, which equals $(2^r)^k = 2^{rk} = 2^n$.

   Another way of calculating the number of AND gates would be to consider what the output of the AND gates correspond to. Since each output of the AND gate is an output for the overall $n$-input decoder, the number of AND gates must be equal to the number of outputs, which is $2^n$.

3. Now, for our coincident 12-input decoder, how many 4-input decoders do we need? How many AND gates? How many inputs do we need for the AND gates?

   ***Solution*** We plug in $n = 12$, $r = 4$ and $k = 3$ in the equations we derived in part 2. We need 3 4-input decoders, $2^{12} = 4096$ AND gates, where each AND gate has 3 inputs.

## Problem 5 (4 points)

We want to use multiplexers to implement the function $f(a,b,c,d) = \sum m(1,2,4,7,10,11,14)$ using the following methods.

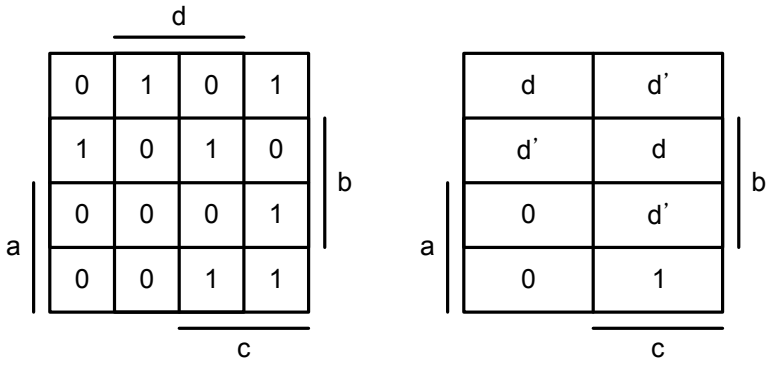
For both cases, show your work, either the minterm expressions or K-map implementations. Assume that complements to inputs are available.

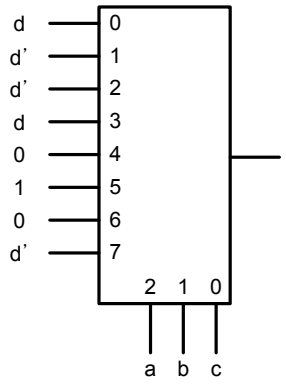
1. Implement the function using 8-to-1 multiplexers. Use $a, b, c$ as the selection bits.

   ***Solution***

$$\begin{aligned} f(a,b,c,d) &= \sum(1,2,4,7,10,11,14) \\ &= a'b'c'd + a'b'cd' + a'bc'd' + a'bcd + ab'cd' + ab'cd + abcd' \\ &= m_0(a,b,c)d + m_1(a,b,c)d' + m_2(a,b,c)d' + m_3(a,b,c)d + m_5(a,b,c)(d'+d) + m_7(a,b,c)d' \end{aligned}$$

   Or if we use K-maps, we get:



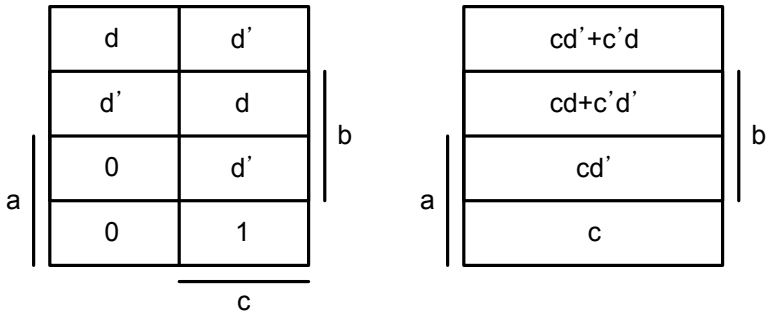

8 to 1 MUX

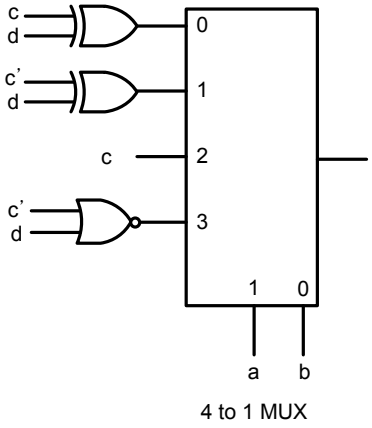
2. Repeat the same process using 4-to-1 multiplexers. Use $a, b$ as the selection bits, and for the additional logic needed, only use XOR gates and NOR gates.

   ***Solution***

$$\begin{aligned} f(a,b,c,d) &= \sum(1,2,4,7,10,11,14) \\ &= a'b'c'd + a'b'cd' + a'bc'd' + a'bcd + ab'cd' + ab'cd + abcd' \\ &= m_0(a,b)(c'd + cd') + m_1(a,b)(c'd' + cd) + m_2(a,b)(cd' + cd) + m_3(a,b)cd' \\ &= m_0(a,b)(c \oplus d) + m_1(a,b)(c' \oplus d) + m_2(a,b)c + m_3(a,b)(c'+d)' \end{aligned}$$

With K-maps, we get:




Note that for input at 1, the input can also be $c \oplus d'$.