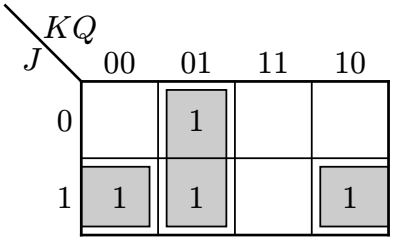
# CS 4341 — Digital Logic and Computer Design, Homework 2

https://iniyanjoseph.com | iniyanijoseph@gmail.com

## 3.10(a) A JK flip-flop receives a clock and two inputs, J and K. On the rising edge of the clock, it updates the output, Q. If J and K are both 0, Q retains its old value. If only J is 1, Q becomes 1. If only K is 1, Q becomes 0. If both J and K are 1, Q becomes the opposite of its present state. Construct a JK flip-flop using a D flip-flop and some combinational logic.

| J | K | Q | Q' |
|---|---|---|---|
| 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 |

$$Q' = Q\overline{K} + J\overline{Q}$$



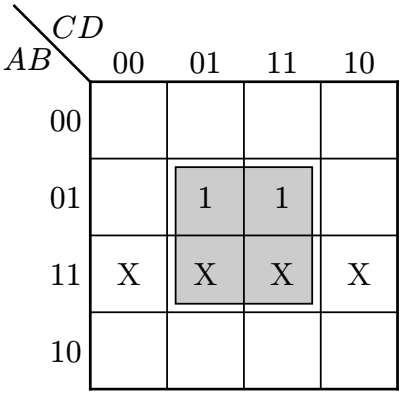

## 3.22 Describe in words what the state machine in Figure 3.69 does. Using binary state encodings, complete a state transition table and output table for the FSM. Write Boolean equations for the next state and output and sketch a schematic of the FSM

If you recieve inputs $A = 1$ and then $B = 1$ in order, then Q becomes 1 and then becomes 0. Otherwise, Q is 0.

| S1 | S0 | A | B | S'1 | S'0 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | X | 0 | 1 |
| 0 | 0 | 0 | X | 0 | 0 |
| 0 | 1 | X | 1 | 1 | 0 |
| 0 | 1 | X | 0 | 0 | 0 |
| 1 | 0 | X | X | 0 | 0 |

| S1 | S0 | Q |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |

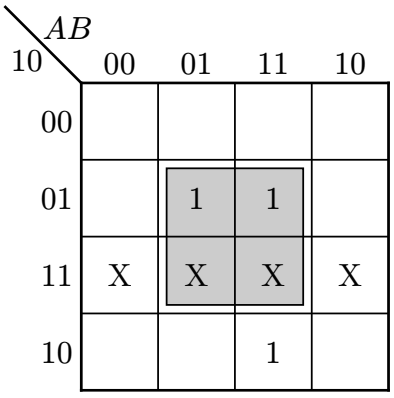
$$S_1' = \overline{S_0} \cdot \overline{S_1} \cdot A$$

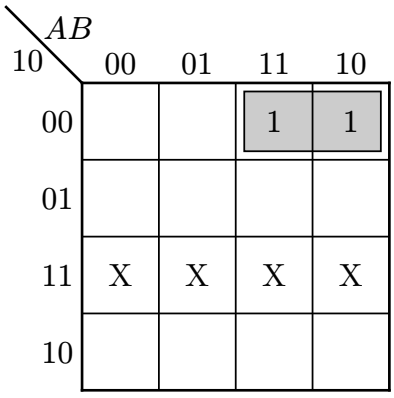



$$S_0' = S_1 \cdot B$$
$$Q = S_1$$

## 3.23 Describe in words what the state machine in Figure 3.70 does. Using binary state encodings, complete a state transition table and output table for the FSM. Write Boolean equations for the next state and output and sketch a schematic of the FSM

| S1 | S0 | A | B | S'1 | S'0 | Output |
|---|---|---|---|---|---|---|
| 0 | 0 | 1 | X | 0 | 1 | 0 |
| 0 | 0 | 0 | X | 0 | 0 | 0 |
| 0 | 1 | X | 1 | 1 | 0 | 0 |
| 0 | 1 | X | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 |




$$S_1' = S_1 \cdot B + S_0 \cdot \overline{S_1} \cdot A \cdot B$$
$$S_0' = \overline{S_0} \cdot \overline{S_1} \cdot A$$

$$Q = S_0 \cdot \overline{S_1} \cdot A \cdot B$$

Although this may at first glance seem more complicated (by not taking the don't cares), it allows for simplification of the circuit by observing that the terms match up with those of the output.

## 3.24 Accidents are still occurring at the intersection of Academic Avenue and Bravado Boulevard. The football team is rushing into the intersection the moment light B turns green. They are colliding with sleep-deprived CS majors who stagger into the intersection just before light A turns red. Extend the traffic light controller from Section 3.4.1 so that both lights are red for 5 seconds before either light turns green again. Sketch your improved Moore machine state transition diagram, state encodings, state transition table, output table, next state and output equations, and your FSM schematic

| S2 | S1 | S0 | TA | TB | S'2 | S'1 | S'0 | LA1 | LA0 | LB1 | LB0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | x | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | x | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | x | x | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | x | x | 0 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | x | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | x | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | x | x | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | x | x | 0 | 0 | 0 | 1 | 0 | 1 | 0 |

State encodings are same as in textbook. The SX is given by its numerical value and 0 is green, 1 is yellow, and 2 is red.

$$S_2' = \overline{S_2} \cdot S_1 \cdot S_0 \cdot T_B + S_2 \cdot \overline{S_1} \cdot \overline{S_0}$$

$$S_1' = \overline{S_2} \cdot \overline{S_1} \cdot S_0 + \overline{S_2} \cdot S_1 \cdot \overline{S_0} + \overline{S_2} \cdot S_1 \cdot S_0 \cdot T_B$$

$$S_0' = \overline{S_2} \cdot \overline{S_1} \cdot \overline{S_0} \cdot \overline{T_A} + \overline{S_2} \cdot S_1 \cdot \overline{S_0} + \overline{S_2} \cdot S_1 \cdot S_0 \cdot T_B + S_2 \cdot \overline{S_1} \cdot \overline{S_0}$$

$$L_1^A = S_1 + S_2$$

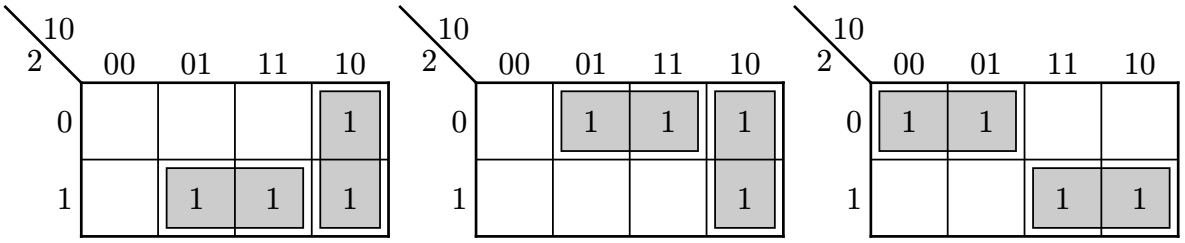
$$L_0^A = \overline{S_2} \cdot \overline{S_1} \cdot S_0$$

$$L_1^B = \overline{S_2} \cdot \overline{S_0} + \overline{S_1} \cdot S_0$$

$$L_0^B = S_2 \cdot \overline{S_1} \cdot \overline{S_0}$$

**3.27 Gray codes have a useful property in that consecutive numbers differ in only a single bit position. Table 3.23 lists a 3-bit Gray code representing the numbers 0 to 7. Design a 3-bit modulo 8 Gray code counter FSM with no inputs and three outputs. (A modulo N counter counts from 0 to N − 1, then repeats. For example, a watch uses a modulo 60 counter for the minutes and seconds that counts from 0 to 59.) When reset, the output should be 000. On each clock edge, the output should advance to the next Gray code. After reaching 100, it should repeat with 000.**

| # | S2 | S1 | S0 | S'2 | S'1 | S'0 |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 2 | 0 | 1 | 1 | 0 | 1 | 0 |
| 3 | 0 | 1 | 0 | 1 | 1 | 0 |
| 4 | 1 | 1 | 0 | 1 | 1 | 1 |
| 5 | 1 | 1 | 1 | 1 | 0 | 1 |
| 6 | 1 | 0 | 1 | 1 | 0 | 0 |
| 7 | 1 | 0 | 0 | 0 | 0 | 0 |




$$S_2' = S_2 \cdot S_0 + S_1 \cdot \overline{S_0}$$
$$S_1' = \overline{S_2} \cdot S_0 + S_1 \cdot \overline{S_0}$$
$$S_0' = \overline{S_2} \cdot \overline{S_1} + S_2 \cdot S_1$$

**3.31 Analyze the FSM shown in Figure 3.72. Write the state transition and output tables and sketch the state transition diagram. Describe in words what the FSM does.**

| S1 | S0 | X | S'1 | S'0 | Q |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 |

This represents a finite state machine where if the current state is $> 0$ (i.e. one of the two current state bits is true), Q is 1. The next state behaviour can best be seen using 2 bits to represent the current state $S_1, S_0$ where $S_1$ is the combinational logic circuit in the circuit diagram provided. If $S_1$ is 0, $S_1$ will be 1 in the next state. If $S_1$ is 1, $S_1$ is set to the current value of $S_0$. For $S_0$, the next state is 1 if $S_0 = 0$ and the input $X = 1$

## 3.33 Ben Bitdiddle has designed the circuit in Figure 3.74 to compute a registered four-input XOR function. Each two-input XOR gate has a propagation delay of 100 ps and a contamination delay of 55 ps. Each flip-flop has a setup time of 60 ps, a hold time of 20 ps, a clock-to-Q maximum delay of 70 ps, and a clock-to-Q minimum delay of 50 ps.

### If there is no clock skew, what is the maximum operating frequency of the circuit?

The maximum operating frequency of the circuit can be computed by first finding the minimum cycle time. In the worst case, the clock to Q delay is 70ps (i.e. its minimum, or the clock-to-Q contamination delay). Additionally, each gate must finish evaluating. This means that it can take up to 300ps for the XOR gates to finish evaluating. Finally, the setup time is 60ps. This implies that $\min T_c = 70\text{ps} + 300\text{ps} + 60\text{ps} = 430\text{ps}$.
Taking the reciprocal, the maximum operating frequency of the circuit is $\frac{1}{430}\text{ps} \approx 2.33\text{GHz}$

### How much clock skew can the circuit tolerate if it must operate at 2 GHz?

If the circuit must operate at 2GHz, then we may solve the following equation. $T_c = t_{\text{pcq}} + t_{\text{pd}} +$ t_setup + t_skew Setting $T_c$ to 500ps, since we know that $t_{\text{pcq}} + t_{\text{pd}} +$ t_setup $= 430\text{ps}$, we can subtract this to find that the maximum clock skew the circuit can tolerate is 70ps

### How much clock skew can the circuit tolerate before it might experience a hold time violation?

In order to prevent a hold time violation, this inequality must hold: $t_{\text{ccq}} + t_{\text{cd}} \geq$ t_hold $+ t_{\text{skew}}$ In other words, the time that the the output may start changing plus the time the output will actually change (based on the specific gates involved) minus how much sooner the output can change (depending on clock skew) must be after the hold time.
Rearranging, we get $t_{\text{ccq}} + t_{\text{cd}} -$ t_hold $\geq t_{\text{skew}}$. Substituting, $50\text{ps} + 3 * 55\text{ps} - 20\text{ps} = 195\text{ps}$ of clock skew.

### Alyssa P. Hacker points out that she can redesign the combinational logic between the registers to be faster and tolerate more clock skew. Her improved circuit also uses three two-input XORs, but they are arranged differently. What is her circuit? What is its maximum frequency if there is no clock skew? How much clock skew can the circuit tolerate before it might experience a hold time violation?

The only thing which changes is the $t_{\text{pd}} \to 200$ and $t_{\text{cd}} \to 110$. This can be used in the same calculations as before $\min T_c = 70\text{ps} + 200\text{ps} + 60\text{ps} = 330\text{ps}$, giving a maximum frequency of $\approx$ 3.01GHz. The circuit can tolerate $50\text{ps} + 2 * 55\text{ps} - 20\text{ps} = 140\text{ps}$ of clock skew.
Used https://www.circuit-diagram.org/editorb/ for circuit diagrams.