GIT Department of Computer Engineering
CSE 232 - Spring 2020
Homework 2

BARAN HASAN BOZDUMAN
171044036

1. **Compute the clock period for the following clock frequencies.**

Frequency = 1/clock period

1 kHz = 1000 cycles/second

1 MHz= 1000 000 cycles/second

1 GHz = 1000 000 000 cycles/second

1 THz = 1000 000 000 000 cycles/second

a. 50 kHz (early computers)

1/50 000 = 0,00002s =>20000ns

b. 300 MHz (Sony Playstation 2 processor)

1/300 000 000 = 3,33*10^-9s =>3.33ns

c. 3.4 GHz (Intel Pentium 4 processor)

1/3 400 000 000 = 2.94*10^-10s=>0.294ns

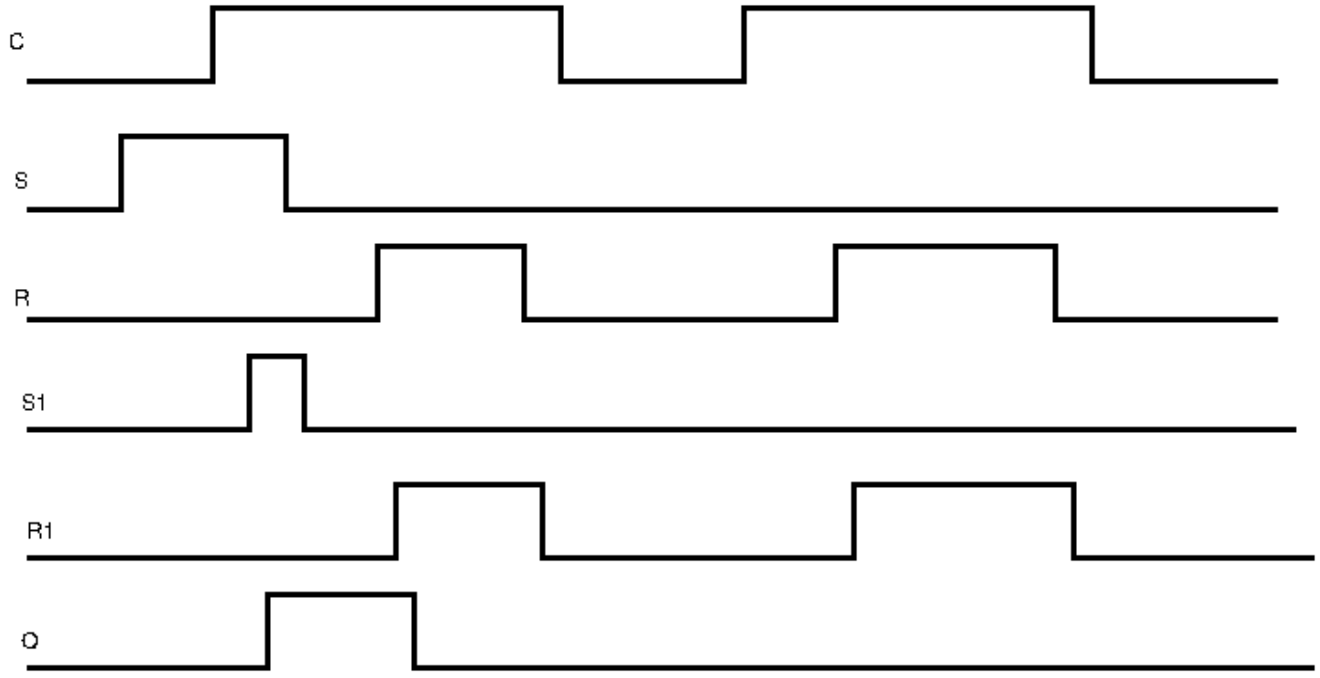
d. 10 GHz (PCs of the early 2010s)

1/10 000 000 000 = 1*10^(-10)s=>0.1ns

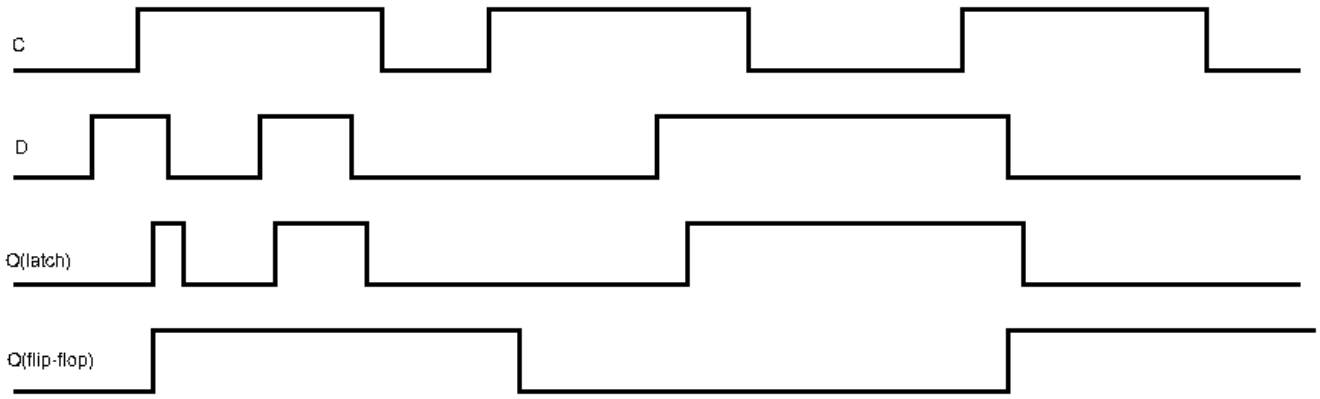
e. 1 THz (1 terahertz) (PCs of the future?)

1/1 000 000 000 0001*10^(-12)s=>0.001ns

2. Trace the behavior of a level-sensitive SR latch for the input pattern in below figure. Assume S1, R1, and Q are initially 0. Complete the timing diagram for S1, R1 and Q, assuming logic gates have a tiny but non-zero delay.



3.Compare the behavior of D latch and D flip-flop devices by completing the timing diagram adding Q (latch) and Q (flip-flop) in below figure. Provide a brief explanation of the behavior of each device. Assume each device initially stores a 0.



4. FSMs with the following numbers of states, indicate the smallest possible number of bits for a state register representing those states:

a. 4 =>2

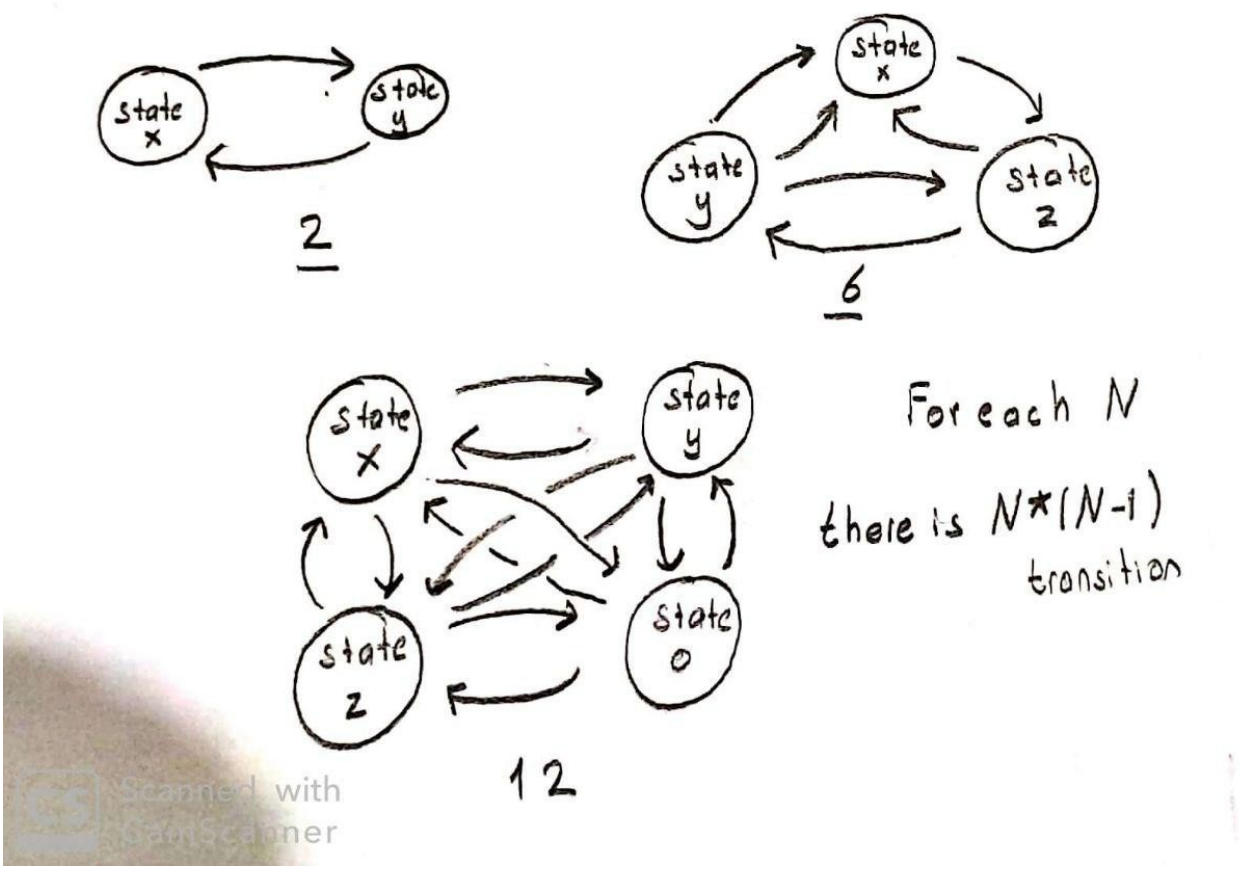
b. 8=>3

c. 9=>4

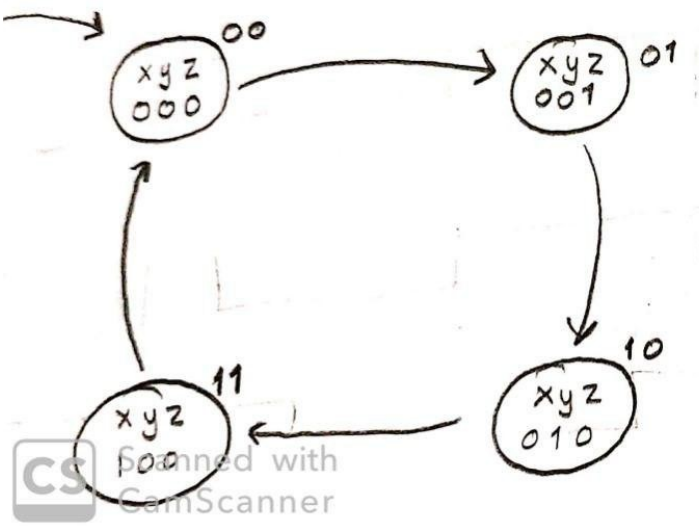
d. 23=>5

e. 900=>10

5. If an FSM has N states, what is the maximum number of possible transitions that could exist in the FSM?Assume that no pair of states has more than one transition in the same direction, and that no state has a transition point back to itself. Assuming there are a large number of inputs, meaning the number of transitions is not limited by the number of inputs? Hint: try for small N, and then generalize.



6. Draw a state diagram for an FSM with no inputs and three outputs x, y, and z. xyz should always exhibit the following sequence: 000, 001, 010, 100, repeat. The output should change only on a rising clock edge. Make 000 the initial state. Using the process for designing a controller, convert the FSM to a controller, stopping once you have created the truth table and derive the Boolean expressions.

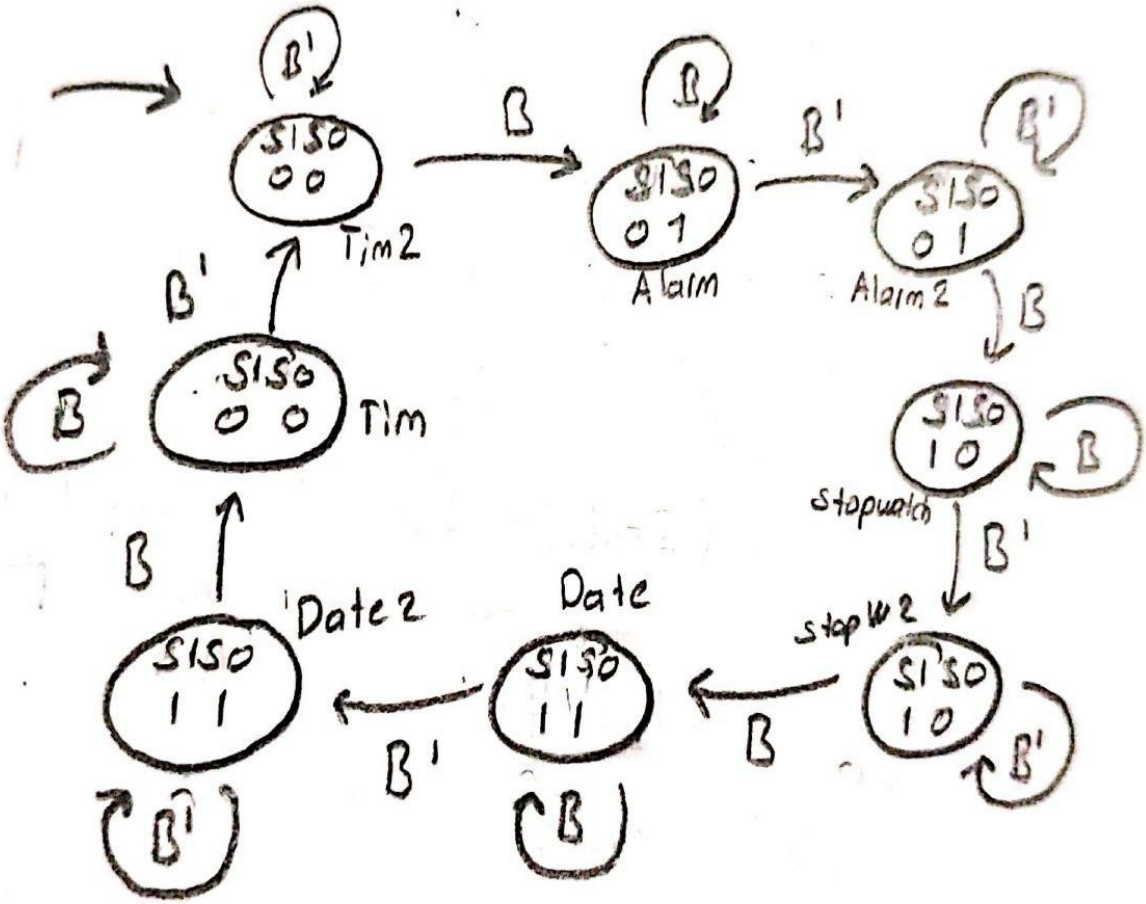


S1	S0	n1	n0	x	y	z
0	0	0	1	0	0	0
0	1	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0

X = s1s0
Y = s1s0'
Z =s1's0
n1= s1 XOR s0
n0= s1's0' + s1s0'= s0'

7. A wristwatch display can show one of four items: the time, the alarm, the stopwatch, or the date, controlled by two signals s1 and s0 (00 displays the time, 01 the alarm, 10 the stopwatch, and 11 the date âassume s1s0 control an N-bit mux that passes through the appropriate register). Pressing a button B (which sets B = 1) sequences the display to the next item. For example, if the presently displayed item is the date, the next item is the current time. Create a state diagram for an FSM describing this sequencing behavior, having an input bit B, and two output bits s1 and s0. Be sure to only sequence forward by one item each time the button is pressed, regardless of how long the button is pressedâin other words, be sure to wait for the button to be released after sequencing forward one item. Use short but descriptive names for each state. Make displaying the time be the initial state. Using the process for designing a controller, convert the FSM to a controller, stopping once you have created the truth table and derive the Boolean expressions.

To block after user press and it needs to change other output we put two states for each of feature



Starting from time2 and it indicates x2x1x0 which is 000 and for each state it increases 1 until end of the loop.

Inputs				**Outputs**				
x2	**x1**	**x0**	**B**	**n2**	**n1**	**n0**	**s1**	**s0**
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	0	0
0	0	1	0	0	1	0	0	1
0	0	1	1	0	0	1	0	1
0	1	0	0	0	1	0	0	1
0	1	0	1	0	1	1	0	1
0	1	1	0	1	0	0	1	0
0	1	1	1	0	1	1	1	0
1	0	0	0	1	0	0	1	0
1	0	0	1	1	0	1	1	0
1	0	1	0	1	0	1	1	1
1	0	1	1	1	1	0	1	1
1	1	0	0	1	1	0	1	1
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0
1	1	1	1	1	1	1	0	0

n2=x2’x1x0 B’ + x2x1’x0’B’ + x2x1’x0’B+ x2x1’x0B’ +x2x1’x0B +x2x1x0’B’+ x2x1x0’B+x2x1x0B

n1=x2’x1’x0B’+ x2’x1x0’B’+ x2’x1x0’B+ x2’x1x0B+ x2x1’x0B+ x2x1x0’B+ x2x1x0’B’+x2x1x0B

n0=x2’x1’x0’B+ x2’x1’x0B+ x2’x1x0’B+ x2’x1x0B+ x2x1’x0’B+ x2x1’x0B’+ x2x1x0’B+x2x1x0B