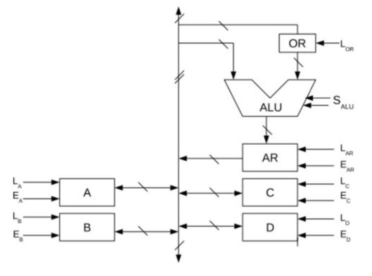
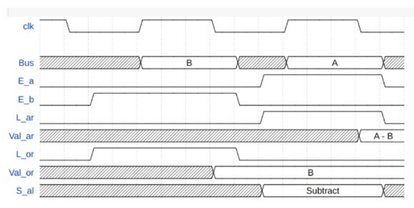
DSM Tutorial Quiz Questions (Post Midsems)

November 14th

Q3: **Given the following ALU design, Give the timing diagram for performing the operation (A-B) and storing it in AR: (Dont use registers C and D)**



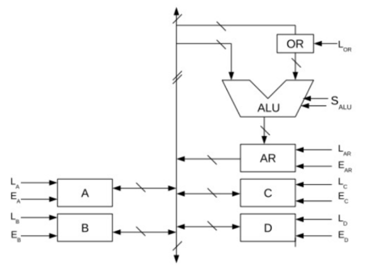
Ans



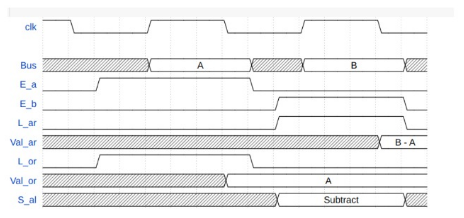
Clock Cycle 1: E_b, L_or
Clock Cycle 2: E_a, L_ar, S_al = SUBTRACTS

Q4:

Given the following ALU design, Give the timing diagram for performing the operation (B-A) and storing it in AR: (Dont use registers C and D)

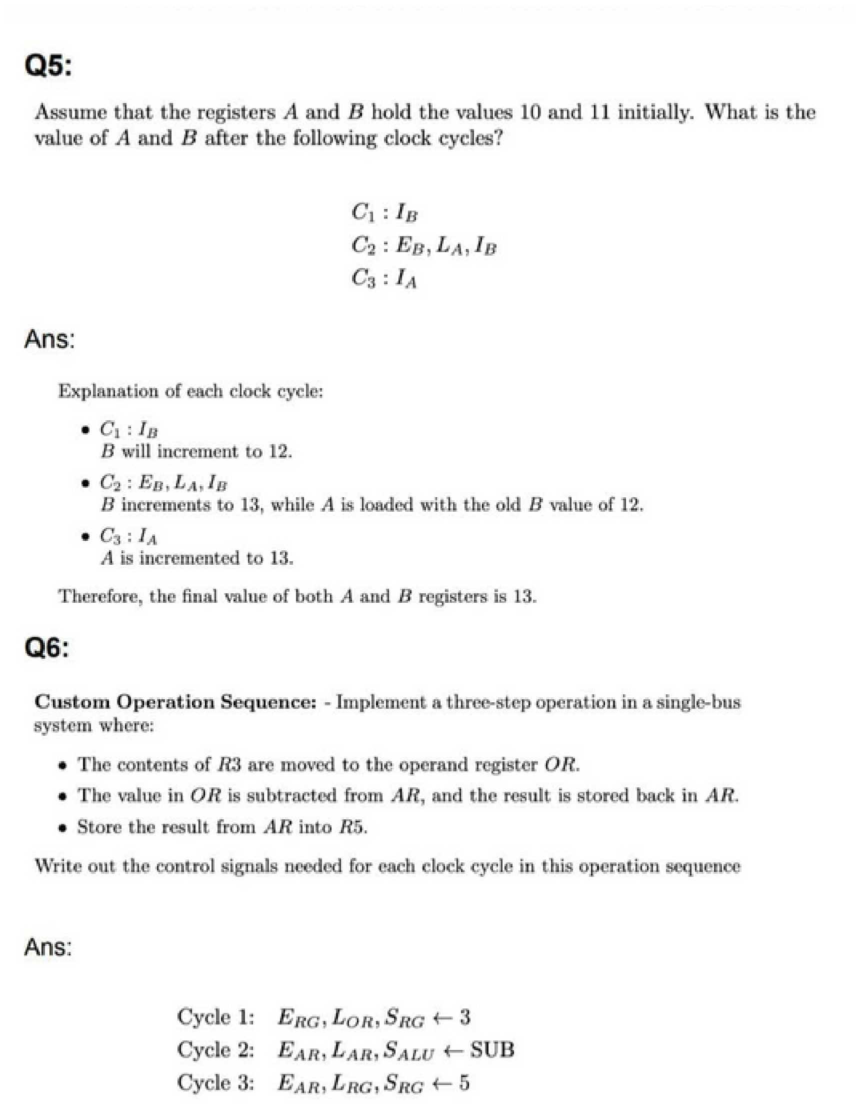


Ans



Clock cycle 1: E_a, L_or
Clock cycle 2: E_b, L_ar, S_al = SUBTRACT

November 13th



Q5:

Assume that the registers A and B hold the values 10 and 11 initially. What is the value of A and B after the following clock cycles?

$$C_1 : I_B$$
$$C_2 : E_B, L_A, I_B$$
$$C_3 : I_A$$

Ans:

Explanation of each clock cycle:

- $C_1 : I_B$
  B will increment to 12.
- $C_2 : E_B, L_A, I_B$
  B increments to 13, while A is loaded with the old B value of 12.
- $C_3 : I_A$
  A is incremented to 13.

Therefore, the final value of both A and B registers is 13.

Q6:

Custom Operation Sequence: - Implement a three-step operation in a single-bus system where:

- The contents of $R3$ are moved to the operand register OR.
- The value in OR is subtracted from AR, and the result is stored back in AR.
- Store the result from AR into $R5$.

Write out the control signals needed for each clock cycle in this operation sequence

Ans:

Cycle 1: $E_{RG}, L_{OR}, S_{RG} \leftarrow 3$
Cycle 2: $E_{AR}, L_{AR}, S_{ALU} \leftarrow \text{SUB}$
Cycle 3: $E_{AR}, L_{RG}, S_{RG} \leftarrow 5$

November 12th

Q1

In an accumulator-based architecture, the arithmetic unit relies heavily on the accumulator register to perform operations. If an 8-bit accumulator initially holds the value 0x2F (in hexadecimal), and it receives an instruction to add 0x3A to the accumulator's value, what will be the final value in the accumulator after the operation? Assume no carry-in from previous operations.

Ans

To solve this, we perform an 8-bit addition of the hexadecimal values in the accumulator.

Convert the hexadecimal values to binary or decimal for clarity:

0x2F in binary: 0010 1111
0x3A in binary: 0011 1010

Adding them will give 0110 1001 ⇒ 0x69

Thus, the final value in the accumulator is 0x69.

Q2

In a single-bus architecture, data transfer between components takes place over a single shared bus, which can introduce delays due to the sequential nature of transfers. Consider a simple addition operation that follows these steps in a single-bus system:

Step 1: Load operand A from memory to a register.
Step 2: Load operand B from memory to a register.
Step 3: Perform addition in the arithmetic unit using operand A and operand B.
Step 4: Store the result back to memory.
If each data transfer over the bus (either loading from or storing to memory) takes 2 cycles and the arithmetic operation itself takes 1 cycle, calculate the total number of cycles required to complete this addition operation.

Ans

To determine the total cycles, add the cycles required for each step:

Step 1: Load operand A from memory to the register — 2 cycles.
Step 2: Load operand B from memory to the register — 2 cycles.
Step 3: Perform addition in the arithmetic unit — 1 cycle.
Step 4: Store the result back to memory — 2 cycles.

Total cycles required:
2+2+1+2=7 cycles

Answer: The addition operation requires a total of 7 cycles to complete.

November 8th

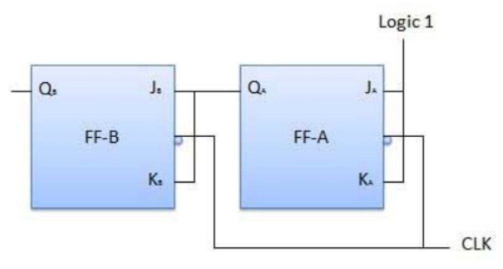
1. Design a 2-bit ripple counter using JK flipflops.

State Table

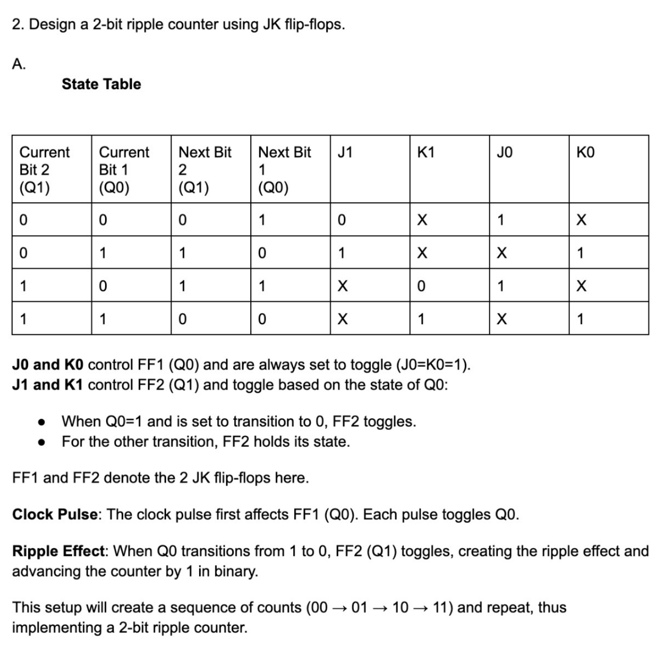
Current Bit 2 (s(i))	Current Bit 1 (t(i))	Next Bit 2 (s(i+1))	Next Bit 1 (t(i+1))	J2	K2	J1	K1
0	0	0	1	0	X	1	X
0	1	1	0	1	X	X	1
1	0	1	1	X	0	1	X
1	1	0	0	X	1	X	1

Using K-maps (or by directly analyzing the table), we get,

J2 = t
K2 = t
J1 = 1
K1 = 1



FF-A is the flip-flop required for Bit 1 and the other one is for Bit 2. Qa and Qs depict the states.



2. Design a 2-bit ripple counter using JK flip-flops.

A.

State Table

Current Bit 2 (Q1)	Current Bit 1 (Q0)	Next Bit 2 (Q1)	Next Bit 1 (Q0)	J1	K1	J0	K0
0	0	0	1	0	X	1	X
0	1	1	0	1	X	X	1
1	0	1	1	X	0	1	X
1	1	0	0	X	1	X	1

J0 and K0 control FF1 (Q0) and are always set to toggle (J0=K0=1).
J1 and K1 control FF2 (Q1) and toggle based on the state of Q0:

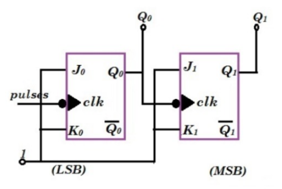
- When Q0=1 and is set to transition to 0, FF2 toggles.
- For the other transition, FF2 holds its state.

FF1 and FF2 denote the 2 JK flip-flops here.

Clock Pulse: The clock pulse first affects FF1 (Q0). Each pulse toggles Q0.

Ripple Effect: When Q0 transitions from 1 to 0, FF2 (Q1) toggles, creating the ripple effect and advancing the counter by 1 in binary.

This setup will create a sequence of counts (00 → 01 → 10 → 11) and repeat, thus implementing a 2-bit ripple counter.



November 7th

Q1

In a 1K-word memory with 16-bit words, calculate the number of address lines required. Also detail how the word size affects the number of address lines required

10 address lines are required to uniquely address each word in a 1K-word memory. Word size should not affect the address line size because the address lines only determine the number of unique locations (words) we can address in memory.

Q2

A Random Access Memory (RAM) module has 64KB capacity with 8-bit word length. Determine the number of address lines required for this RAM.

Solution:
Calculating Address Lines:

A 64KB RAM means it has 64×1024 =65,536 locations.
Since each location stores an 8-bit word, there are 65,536 unique addresses.
The number of address lines n required to access these locations can be determined using
2^n = 65,536
⇒ n = 16
Therefore, 16 address lines are required for the 64KB RAM.

November 6th

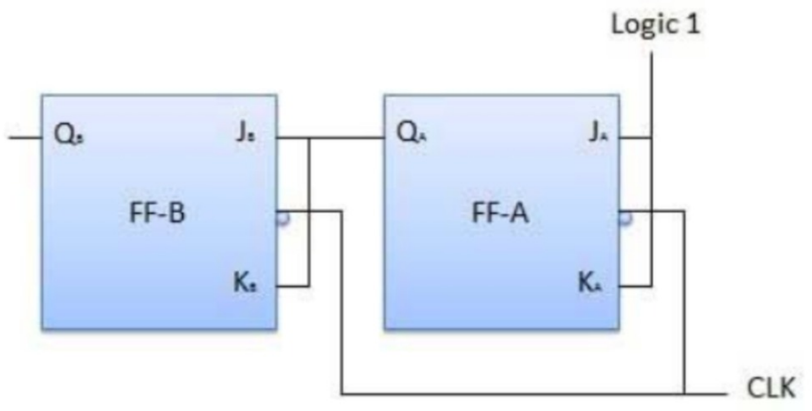
1. Design a 2-bit ripple counter using JK flipflops.

State Table

Current Bit 2 (s(i))	Current Bit 1 (t(i))	Next Bit 2 (s(i+1))	Next Bit 1 (t(i+1))	J2	K2	J1	K1
0	0	0	1	0	X	1	X
0	1	1	0	1	X	X	1
1	0	1	1	X	0	1	X
1	1	0	0	X	1	X	1

Using K-maps (or by directly analyzing the table), we get,

J2 = t
K2 = t
J1 = 1
K1 = 1



FF-A is the flip-flop required for Bit 1 and the other one is for Bit 2. Qa and Qs depict the states.

November 5th

Q1
How many address and data lines will be there for a memory of size, 1K X 8?

Ans:
$1K = 2^{10}$, Number of address lines = 10
Number of data lines = 8

Q2
How many number of 16X8 size memories are needed to obtain a memory of size 256X16?

Ans:
256/16 = 16, 16X8 memories are sufficient to get a memory of size 256 X 8. But to get 256 X 16, we need twice of that.
So, the required number of 16 X 8 memories = 16 * 2 = 32

October 16th

Q1

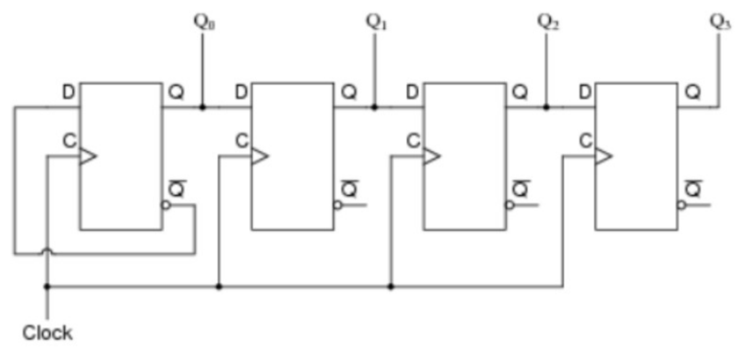
You have an 8-bit bidirectional shift register with the initial value 11001100. The register can shift both left and right, and it has an additional control input (C) that determines the direction:
C = 0 shifts the register to the left.
C = 1 shifts the register to the right.

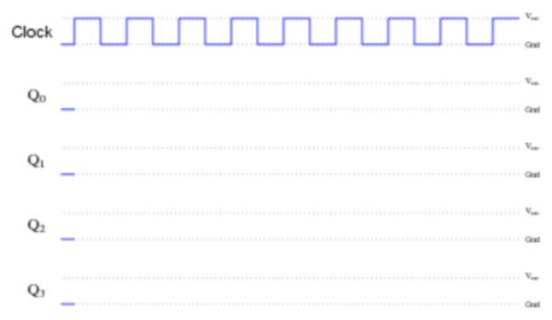
Each shift operation also takes in an external input bit (D_in) that is inserted into the register from the direction of the shift. The external input bits provided are: 1, 0, 1, 1 (used in the order of each shift operation).
The sequence of operations is as follows:

1. Shift left (C = 0) with D_in = 1
2. Shift right (C = 1) with D_in = 0
3. Shift left (C = 0) with D_in = 1
4. Shift left (C = 0) with D_in = 1
   What will be the final value in the shift register after these 4 operations?

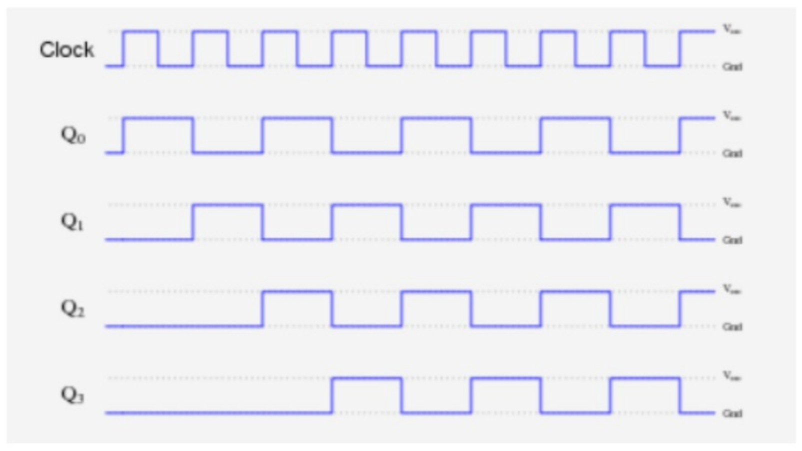
Answer : 00110011

2. Complete the timing diagram for this circuit, assuming all Q outputs begin in the low state:





A.



The input to each consecutive flip-flop keeps alternating at their first clock pulse. If k is the output at flip-flop 1, k will be the input for flip-flop 2 and k' will be the input for flip-flop 1. If we keep on repeating this, we get k as the input for flip-flop i whenever i is even and k' as the input whenever i is odd.

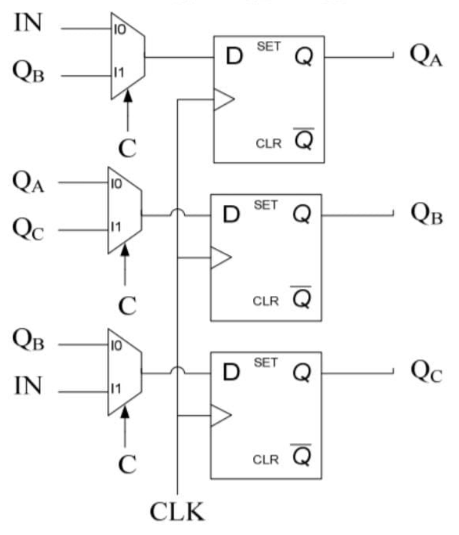
October 15th

Q1

Design a 3-bit shift register using 2:1 Mux and D Flip Flops which shifts right if the control input, C = 0 and shifts left if C = 1?

Ans

If C = 0, the circuit shifts from IN → Q_A → Q_B → Q_C and
If C = 1, the circuit shifts from IN → Q_C → Q_B → Q_A

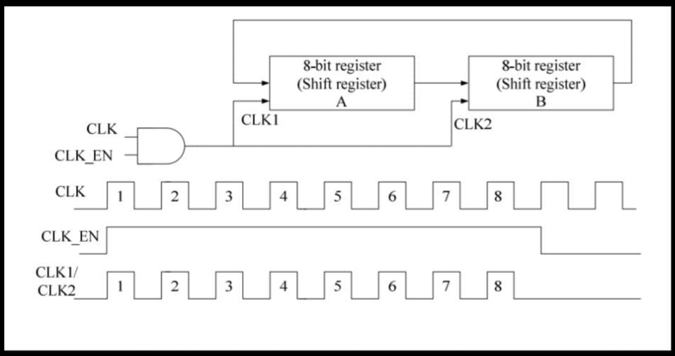


Q2

How to swap the contents of two 8-bit registers without using a third register.

Ans

The complete design using shift registers is shown in the following figure. The main clock is gated with the clock enable so that A and B will be shifted just 8 clocks. After 8 clocks A and B will have their contents swapped.


8-bit register (Shift register) A
8-bit register (Shift register) B
CLK1
CLK2
CLK
CLK_EN
CLK
1 2 3 4 5 6 7 8
CLK_EN
CLK1/ CLK2
1 2 3 4 5 6 7 8

October 11th

Q1

State Reduction: A sequential circuit has the following state table:

Curr State	Next State (I = 0)	Next State (I = 1)	Output (I = 0)	Output (I = 1)
A	A	C	1	0
B	A	D	1	0
C	D	A	0	1
D	D	A	0	1

Perform state reduction to find the minimum equivalent state table.

Solution:

- The state reduction process involves finding equivalent states. In this case, states C and D can be combined because they have the same behavior for all inputs. The reduced state table is:

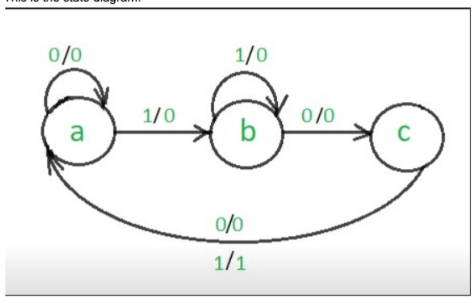
Curr State	Next State (I = 0)	Next State (I = 1)	Output (I = 0)	Output (I = 1)
A	A	C(=D)	1	0
B	A	D	1	0
D	D	A	0	1

- Similarly, since C and D are equivalent, states A and B can be combined because they have the same behavior for all inputs.

Curr State	Next State (I = 0)	Next State (I = 1)	Output (I = 0)	Output (I = 1)
A	A	D	1	0
D	D	A	0	1

Q2

2. Develop a “101” sequence detector using D flip flops
A. This is the state diagram:



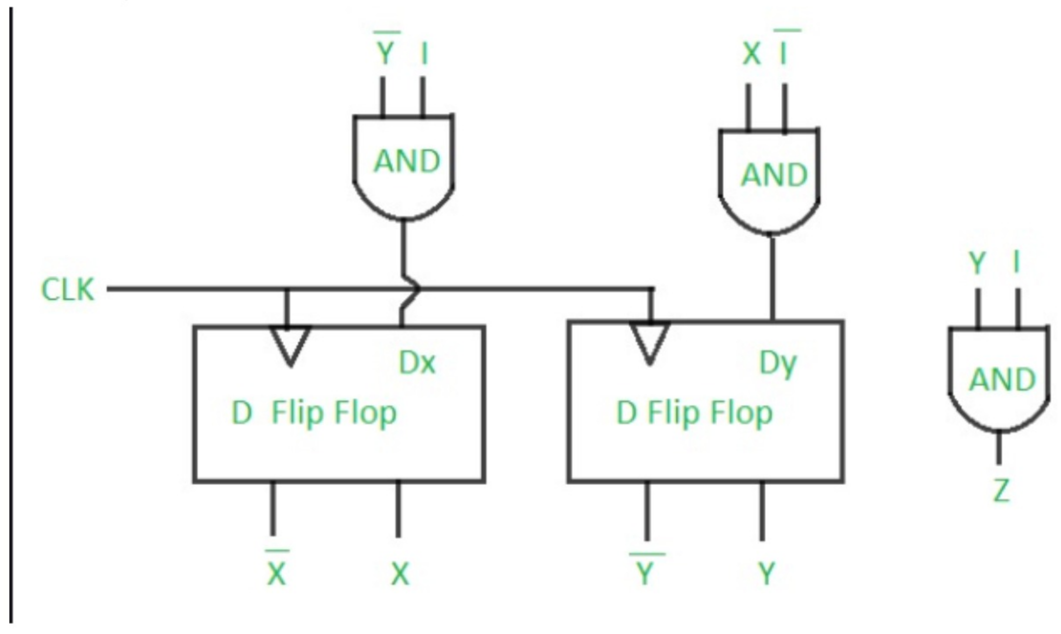
Consider state a to be 00, state b to be 10, state c to be 01

Using D flip flops, this will be the final truth table:

Present X state (X)	Present Y state (Y)	Input (I)	Next X state (X’)	Next X state (Y’)	Dx	Dy	Output (Z)
0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	1	1	0	1	0	0
1	1	0	X	X	X	X	X
1	1	1	X	X	X	X	X

Using K-maps, we get
Dx = Y’.I
Dy = X.I’
Z = Y.I

Circuit diagram:



October 10th

Q1

1. Implement D flip-flop using JK flipflop

D	Qn	Qn+1	J	K
0	0	0	0	x
0	1	0	x	1
1	0	1	1	x
1	1	1	x	0

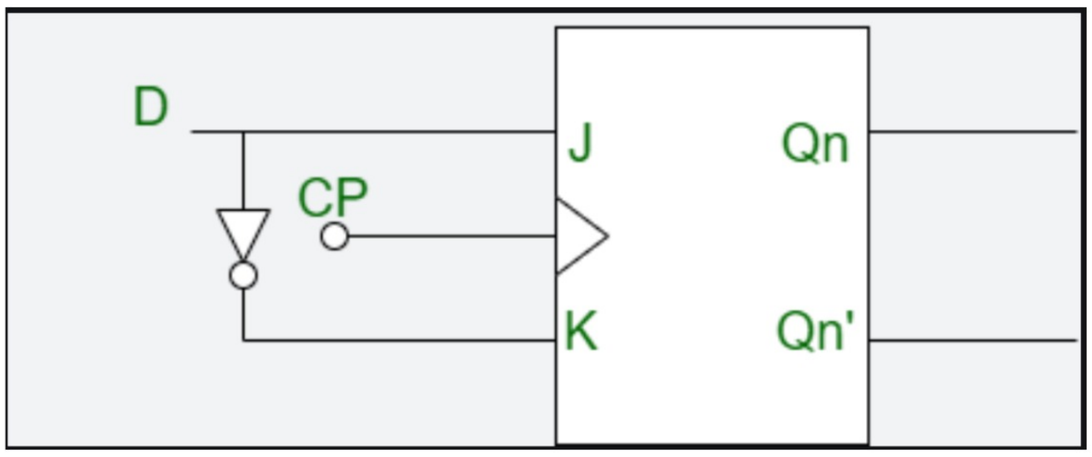
The first 3 columns show all the cases for D flipflop.

For the transition Qn = 0 and Qn+1 = 0, we just need J to be 0 as the state won't change when K=0 and Q will remain 0 when K=1.
For the transition Qn = 0 and Qn+1 = 1, we need J to be 1 as the state toggles when K=1 and Q will change to 1 when K=0.
For the transition Qn = 1 and Qn+1 = 0,we need K to be 1 as the state toggles when J=1 and Q changes to 0 when J=0.
For the transition Qn = 1 and Qn+1 = 1,we need K to be 0 as the state remains the same when J=0 and Q remains 1 when J=1.

For D and Qn as inputs and J as output, we apply a K-map and we'll get the equation J=D

For D and Qn as inputs and K as output, we apply a K-map and we'll get the equation K=D'.

So, our final circuit will be:



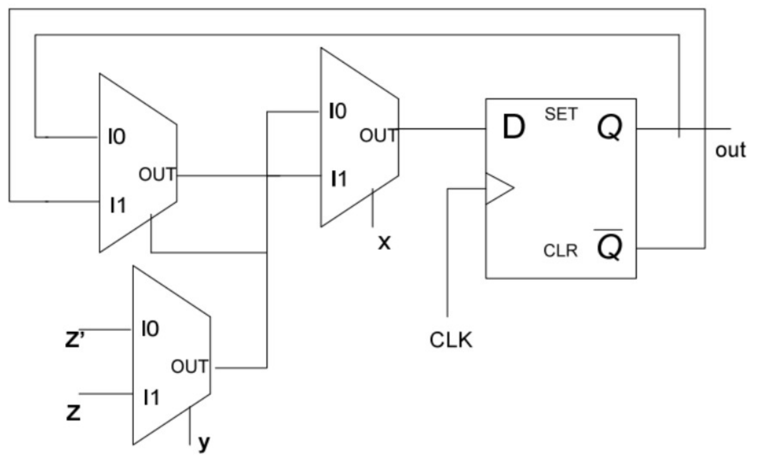
Q2

Using DFFs and minimum no. of 2×1 Mux, implement the following XYZ flip-flop. (Assume you have the inverse f the variables also)

X	Y	Z	Q(t+1)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	Q(t)
1	0	1	Q(t)`
1	1	0	Q(t)`
1	1	1	Q(t)

Ans

It is clear that if X=0, Q(t+1) = Y XNOR Z. If X =1 and if (Y XNOR Z), Q(t+1) = Q(t), else Q(t)'. So we need one 2:1 mux to generate Y XNOR Z. One to select Q(t) and Q(t)' and one more to select between X=0 case and X=1 case. Total we need 3 2:1 mux.



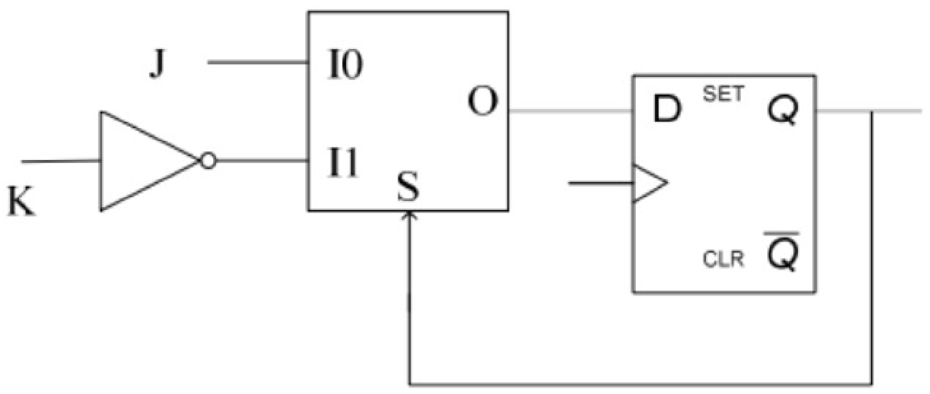
October 9th

Q1

Construct a J-K flip flop using a DFF, 2:1 Mux and an-inverter

Ans

The catch here is to use Q as select line. You can observe the cofactors of Q(t+1) with respect to J,K and Q(t). Using J or K as the select line with 2:1 mux will not do.



Q2

2. A sequential circuit with two D flip-flops A and B, inputs x and y, and output z is specified by the following next-state and output equations:

$$A(t+1) = x'y + xB$$

$$B(t+1) = xA + yB$$

$$z = A'$$

Construct the state table for this circuit.

2. • The resultant state table would be:

Present State		**Input**		**Next State**		**Output**
A	B	x	y	A(t+1)	B(t+1)	z
0	0	0	0	0	0	1
0	0	0	1	1	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	0	0	1
0	1	0	1	1	1	1
0	1	1	0	1	0	1
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	1	0
1	1	0	0	0	0	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	0