CSE260 Practice Sheet

Q: Are all of these enough to get full marks in the exam?

A:

NO. NO. NO.

This is a practice sheet intended to help you reinforce your understanding of the material. You are encouraged to use it as much as needed for practice. However, success in exams ultimately depends on your ability to comprehend questions, formulate appropriate responses, and articulate your answers clearly. These are essential human skills that cannot be fully developed through practice sheet alone. Nonetheless, best of luck with your preparation.

Number System Conversions & Arithmetic Operations [Lecture 1, Lecture 2]

- Convert the following binary numbers to equivalent decimal numbers.
  (a) $(101110001001)_2$
  (b) $(11011.101)_2$
- Convert the following decimal number to equivalent binary numbers: $(4195)_{10}$
- Convert the following decimal number to equivalent binary numbers: $(3785.65625)_{10}$
- Convert the following decimal number to equivalent binary numbers: $(4785.150263)_{10}$ [for infinite fractional part, just do 6-7 steps and use dots for the rest]
- Convert the following decimal number to equivalent base 5 numbers: $(4123)_{10}$
- Convert the following decimal number to equivalent hexadecimal numbers: $(513)_{10}$
- Convert the following decimal number to equivalent base 9 numbers: $(813)_{10}$
- Perform the following base conversions
  a) $(29)_{12} = (?)_7$
  b) $(10110111)_5 = (?)_4$
- Perform addition, subtraction and multiplication for the pair of following base-9 numbers. Verify your results by converting the problem into decimal.

  412

  134
- Subtract 13 from 27 in 7 bits using 2's complement number system and justify whether there is an overflow or not.
- Subtract 45 from 98 in 12 bits using 2's complement number system and justify whether there is an overflow or not.
- Add 13 with 27 in 6 bits using 2's complement number system and justify whether there is an overflow or not.

- Add -57 with 63 in 11 bits using 2’s complement number system and justify whether there is an overflow or not.
- Perform the following arithmetic operations using 13-bit two’s complement and one’s complement systems . State if there is an overflow in each case.

  a) 91 - 499
  b) 379 + 98

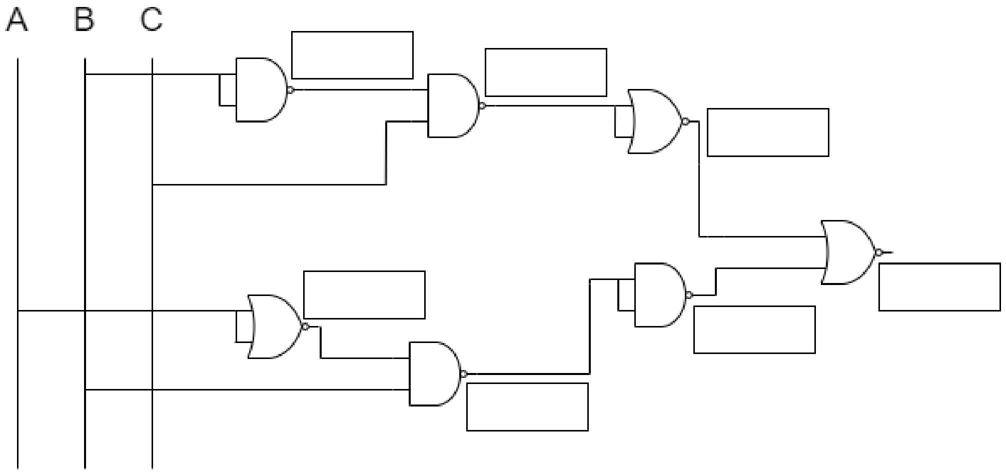
- Divide $101101_2 \div 11_2$
- Divide $725_8 \div 5_8$
- Divide $A2_{16} \div 4_{16}$
- Jack has 721_8 colored pencils. He wants to pack them into small boxes each holding 14_8 pencils. How many boxes can be made?
- A warehouse received $1C_{16}$ toy boxes in the morning. Each box contained 23_8 toys. Later, it was discovered that 45_{10} of those toys were damaged. The remaining toys were then equally divided among 12_3 toy stores. How many toys did each store receive?
- You are a computer engineer and you want to buy two 8 GB DDR4 RAMs. Each RAM costs $(1C2)_{16}$ dollars. You also want to buy a graphics card RTX4070Ti which costs $(10010110000)_2$ dollars. However, you don’t have that much money with you and you are afraid to ask your parents about it. Suddenly, one of your generous friends agreed to give you the money you need. He decided to give you $(4064)_8$ dollars. How much will you have left after buying those components? (Show the answer in decimal)
- A construction project has three phases. Phase 1 costs $(A3)_{16}$ dollars, while Phase 2 costs $(120)_8$ dollars. Phase 3 has been operating for $(30)_5$ days, each day costs $(101.01)_2$ dollars. How much more money will be needed for Phase 3 to match the total cost of Phases 1 and 2 combined? (Show the answer in decimal)
- Bangladesh is facing up against Australia in a cricket match. They have scored $(154)_{16}$ runs in total. Australia has currently played $(24)_8$ overs with a run rate of $(110.1)_2$. Calculate how many more runs they need to score to win the game. (Show your answer in decimal.)

Boolean Simplification [Lecture 3, Lecture 4]

- **Simplify** the following boolean equation using the laws of boolean algebra:
  - F(x,y,z) = x’(yz’+xy)’ + xyz + z’y(x+x’y’)’
  - F(a,b,c) = c’b(a+a’b’)’ + a’(bc’+ab)’ + abc
  - F(A,B,C,D) = (A⊕B)(C+D)(A⊙C)+(A+B)(C′+D)(A′+CC′)
  - A.B'.C+(A.B.C⊙C')'+A.B'.C'.(A.B'+C')+(A+B⊙C)'
  - $(A + AB)(A + \overline{B}C)\ (\overline{A} + ABC)$
  - A’B’ + B(C’⊙B’C’)’
  - P’Q’+PQ(PQZ+Q’Z’) + 1
  - CD’+A’C’D’(A⊕D’)’
  - A+AB’+(A+1)’B’+(1+B)’
  - (A’(B’)’ + AB’)’+B’C’+(BC’)’+(B⊕C)’

- Simplify the following boolean expression to minimum number of literals:
  $(A + B)(A + \overline{B})(\overline{A} + C)$
- Find the complement of the following expression:
  (x' + y + z')(x' + y')(x + z')
- Draw an XOR gate using NOR gates only
- Draw an XNOR gate using NAND gates only.

- Determine the logic function of the below circuit. Use the boxes in the diagram to write your answer step by step.



- Draw the following functions using NAND gates only:
  F(A,B,C,D) = (A'B'CD' + A'D + (B+D'))
  F(X, Y, Z) = X' + Y ⊕ Z
  NB: Please draw horizontally on your script.
  NB: You can't simplify the above functions and then draw using NAND gate. You have to draw based on the function given in question
- Draw the following functions using NOR gates only:
  F(A,B,C,D) = (AB'C'D' + AD + (B+D'))
  F(A,B,C) = A' + B ⊙ C
  NB: Please draw horizontally on your script.
  NB: You can't simplify the above functions and then draw using NAND gate. You have to draw based on the function given in question
- Find out SOP and POS for the following:
  1. F(A,B,C) = AB+BC'
  2. F(A,B,C,D) = A+ B'CD'
  3. F(A,B,C,D,E) = AB+CDE
- Given that the max terms of the function F(A, B, C) are M(0,2,4), complete the remaining

POS expression by filling in the gaps:
F = (A+B+C)(A+__+C)(?)

- Given that the min terms of the function F(A, B, C) are m(1,3,5), complete the remaining SOP expression by filling in the gaps:
  F = A’.__. C + A’. B. C + ?
- **Simplify** the following boolean equation using the laws of boolean algebra and implement the simplified function using only NOR gates:
  F(a,b,c,d) = Σ(8, 9, 10, 11, 13, 15)

K-MAP [Lecture 5]

- Find the simplified function using K-map:
  1. F(a,b,c,d) = Σ(8, 9, 10, 11, 13, 15)
  2. F(a,b,c,d) = Σ(0, 1, 2, 11, 13, 15)
  3. F(a,b,c) = Σ(0,3,5,7)
  4. F(a,b,c,d) = Σ(8, 9, 10) + d(0,1,2)
  5. F(a,b,c,d) = Σ(1, 9, 10) + d(0,8,11,12)
  6. F(a,b,c,d) = π(1, 9, 10) + d(0,8,11,12)
  7. F(a,b,c,d) = Σ(0, 1, 2,3,4,5,6,7,8,9) + d(10,11,12,13,14)
  8. F(a,b,c,d) = Σ(0, 1, 2,3,4,5,6,7,8,9) + d(10,11,12,13,14,15)
  9. F(a,b,c,d) = Σ( ) + d(4,5,9)
  10. F(a,b,c,d) = Σ( ) + d( )

- **K-Maps Scenario Based Problems:**
  https://docs.google.com/document/d/1eJ2ItIDALeGNAVn1iUGjXkjEJCbmhoYA0LH2RHIP8rc/edit

Combinational Circuits [Lecture 6]

For all the following: Make sure that your circuit is efficient, meaning you should use the lowest number of components. You may use external gates if required.

- Build an adder cum subtractor (4 bits)
- Draw the block diagram of a 20 bits parallel adder.
- Build a 13 person voting system using full and parallel adders.

- Build a 15 person voting system using full and parallel adders.
- Consider A is a 4 bit number. Design A-3 using a 4 bit parallel adder. Use external gates if required.
- Consider A is a 4 bit number. Design A+ 3 using a 4 bit parallel adder. Use external gates if required.
- Consider two numbers: 7 and 5. You can only calculate addition and subtraction between those two numbers. Design a circuit that can perform the above calculations based upon the user's intention.
- Design a full adder using two half adders. You must use two NOR gates and no OR gates.
- Design a full adder using two half adders. You must use three NAND gates and no OR gates.
- Design a circuit diagram for the following system that takes two 4-bit binary numbers [A and B] as inputs and outputs in the following fashion:
  - If both numbers are divisible by 10 the output should be **A+B**
  - **A-B** otherwise
- Design a circuit that takes two 4-bit numbers A and B performs A+2B using parallel adders.
- Design a circuit diagram for the following system that takes two 4-bit binary numbers [A and B] as inputs and outputs in the following fashion:
  - If **A** is divisible by 4 and **B** is greater than 7 then the output should be **A-B**
  - **A+B** otherwise
- Design a circuit diagram for the following system that takes two 4-bit binary numbers [A and B] as inputs and outputs in the following fashion:
  - If C0 is given zero, it will perform normal addition and returns **A+B**
  - If C0 is given as 1, it will perform addition in 1’s complement and returns **A-B**
- Design a circuit diagram for the following system that takes two 4-bit binary numbers [A and B] as inputs and outputs in the following fashion:
  - If one input is odd and the other input is even, the output should be **A-B**
  - If both inputs are odd or both inputs are even, the output should be **A+B**

- Design a circuit that takes two 4-bit numbers A and B performs 2A-B using parallel adders.
- Design a circuit that takes two 4-bit numbers A and B and performs A-B if B is divisible by 4 and performs A+B otherwise.

Combinational Circuits II [Lecture 7]

- Design a Half Adder using two 4:1 Mux(s).
- Design a Full Adder using two 8:1 Mux(s).
- Design a 3-bit Parallel Adder using 8:1 Mux(s).
- Build a 8 person voting system using full and parallel adders; however, implement the full adders using 3:8 decoders and 4:2 encoders.
- Build a circuit that implements the 2’s complement number system (3 bits) using encoder(s) and decoder(s).
- Design an Octal to Binary encoder.
- Design a 4x2 priority encoder prioritizing MSB.
- Design a 4x2 priority encoder prioritizing LSB.

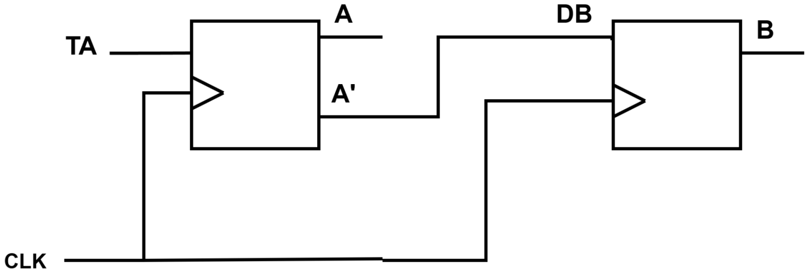
- Construct the circuit of 4x2 Priority Encoder. If the number of 1s in the binary input is even, the priority is given to the MSB 1. If the number of 1s in the binary input is odd, the priority is given to the LSB 1.
- In a futuristic city named "**GG**", public protests are monitored and managed by an **autonomous digital protest control unit**. This system ensures peaceful protests are conducted efficiently. The control unit uses a component called **protest signal prioritizer** to ensure that.

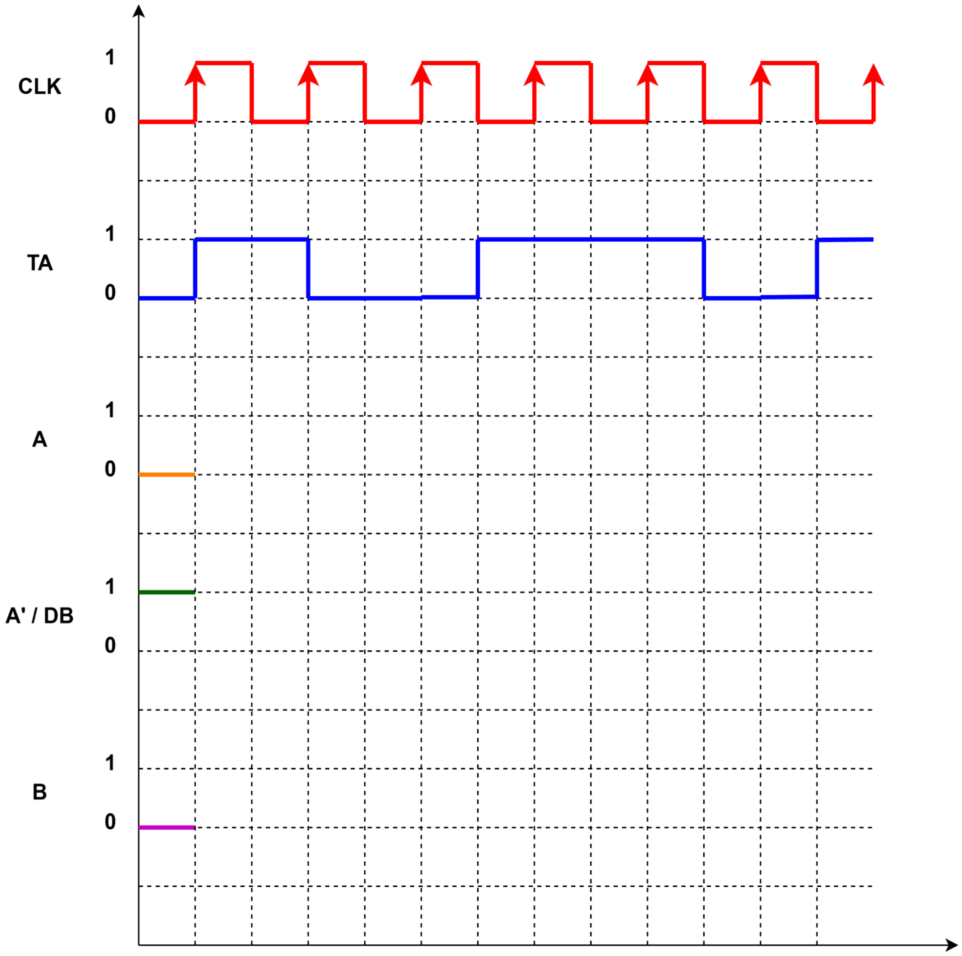
  Protest Signal Prioritizer: The protesters use four digital flags (bits) to send signals to the authorities. The flags work as below: **1 =** indicates a request or emergency, **0 =** indicates no signal from that line.
  The priority of handling the signal by the authority is given below:
  - If there is no **1** signal, the output should be **Don't Care**.
  - If the combined signal number is divisible by **4**, it's assumed to be a high-level organizational message. So, priority goes to the MSB (Most Significant Bit).
  - If the number of raised flags (**1s**) is exactly three, then the signal likely comes from a middle-layer protest coordinator. So, the priority is given to the middle bit (not the MSB or LSB).
  - Otherwise, it is assumed to be individual protesters. Therefore, the priority is given to the LSB (Least Significant Bit), who might be in distress or need help.

  Design and **Draw** the circuit of a Priority Encoder that will take the 4 bit flags(**F_3 F_2 F_1 F_0**) as input and provide the output based on the above rules.
- Implement the following boolean function using a single 16:1 mux .
  F(A,B,C,D) =∑(0,1,2,7,8,10,11,13, 15). Use external gates if required.
- Implement the following boolean function using a single 8:1 mux.
  F(A,B,C,D) =∑(0,1,2,7,8,10,11,13, 15). Use external gates if required.
- Implement the following boolean function using a single 4:1 mux.
  F(A,B,C,D) =∑(0,1,2,7,8,10,11,13, 15). Use external gates if required.
- Implement the following boolean function using both 4:1 and 2:1 mux in a single circuit
  F(A,B,C,D) =∑(0,1,2,7,8,10,11,13, 15). Use external gates if required.
- Implement the following boolean function using **a)** 4x16 decoder(s) only **b)** 2x4 decoder(s) only
  F(A,B,C,D,E) =∑(0,1,2,7,8,10,11,13, 15,18,21,24,25). Use external gates if required.
- Build a full adder using encoder(s) and decoder(s).
- Build a BCD to Excess-3 code converter using encoder(s) and decoder(s).
- Build a BCD to Excess-5 code converter using encoder(s) and decoder(s).
- Design the circuit diagram for a 3x8 decoder.
- Design the circuit diagram for a 2x4 decoder.
- Design the circuit diagram for a 4:1 mux.
- Design a 8:1 mux using both 4:1 mux and 2:1 mux in a single circuit.
- Design AND, OR, NOT gate using 4:1 mux.
- Design AND, OR, NOT gate using 2:1 mux.
- Design a circuit that can demonstrate the usage of ENABLE pin in decoders.

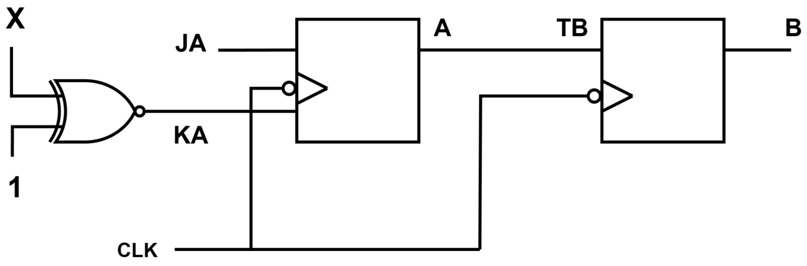
Sequential Circuits [Lecture 8, Lecture 9]

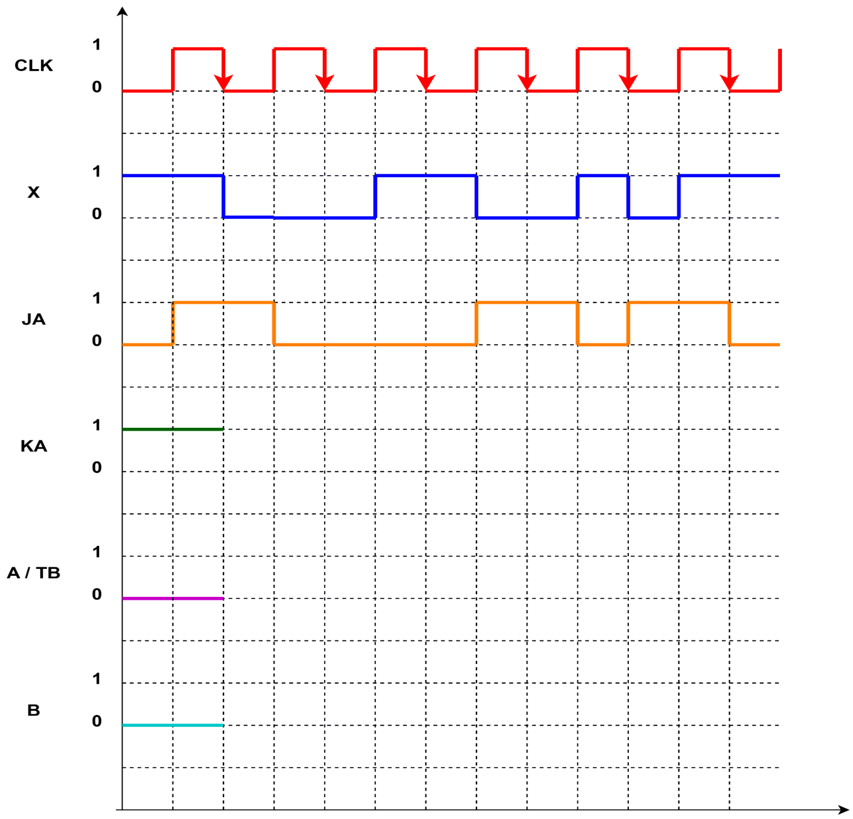
- Design a D FF using SR FF.
- Design a T FF using JK FF.
- Design a D FF using JK FF.
- Design a SR FF.
- Write down the excitation table for SR, D, JK and T FF.
- Write down the characteristics table for SR, D, JK and T FF.
- A digital clock pulse has a frequency of 5 MHz.
    - What is the clock period?
    - How many clock pulses occur in 10 ms?
- A clock signal has a HIGH time of 15ms and a LOW time of 5ms.
    - Find the clock period
    - Find the duty cycle
    - Find the frequency
- The HIGH time of a clock signal is 12ms, and its duty cycle is 60%. Find the clock period.
- Draw and complete the timing diagram for the following circuit, where all the flip-flops are positive-edge-triggered. Clock Pulse(CLK) and TA waveforms are given.



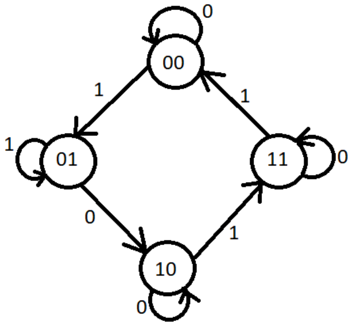


- Draw and complete the timing diagram for the following circuit, where all the flip-flops are negative-edge-triggered. Clock Pulse(CLK), JA, and X waveforms are given.





- Given the state diagram as follows, get the sequential circuit using SR flipflop.



- Implement the following counter using T flip flop
  CSE110 -> CSE111 -> CSE220 -> CSE221 -> CSE331 -> CSE221 -> CSE321 -> CSE110
- 3->4->6->10->12->13->15->3
  1. Implement the given counter using JK flip-flop.
  2. Implement the given counter using T flip-flop.

  NB: For states not given in question, please move to the initial state as per question.
- Implement 4 bit up counter using JK flip flop
- Implement 4 bit down counter using JK flip flop
- Implement 2 bit up/down counter using D flip flop

- Implement 3 bit up/down counter using T flip flop
- Implement 2 bit up/down counter using SR flip flop
- Implement 3 bit up/down counter using JK flip flop
- Implement the following counter using T FF: Green->Orange->Yellow->Red->Yellow->Orange->Yellow->Green
- Implement the following counter using JK FF: Green->Yellow->Red->Yellow->Green
- Implement the following counter using JK FF: 1->2->3->5->7->11->13->1
- Implement the following counter using SR FF: 2->3->6->8->10->12->2
- You are a sports enthusiast whose interests shift with time, making it difficult to predict what you'll be passionate about next. You began with an intense passion for Cricket, spending hours analyzing matches and following every major tournament. As winter approached and Football season peaked, you immersed yourself in the drama of league matches and tactical breakdowns. Later, after watching a few thrilling Formula 1 races with friends, you found yourself drawn to the speed and strategy of motorsport. However, the excitement of Football pulled you back again, especially with a major derby coming up. Then, the Ashes Test series reignited your interest in Cricket, but soon after, the buzz around a high-stakes Football final reeled you back in once more. Then came winter break, and with it, your discovery of competitive Ice Skating. Initially skeptical, you quickly became fascinated by the skill, precision, and artistry involved. Regardless, as the Formula 1 season resumed, your focus shifted to racetracks and championship standings again. Eventually, you went back to where it all began - revisiting Cricket through old match highlights and looking forward to the upcoming season.

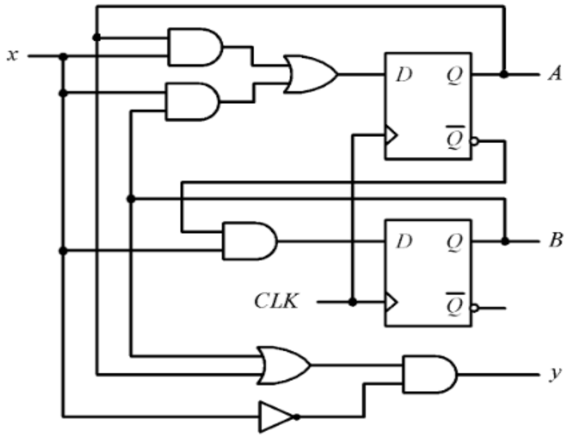
  Design and draw the circuit diagram of your sports preference change throughout the year using

    - JK Flip-Flop(s).
    - T Flip-Flop(s)

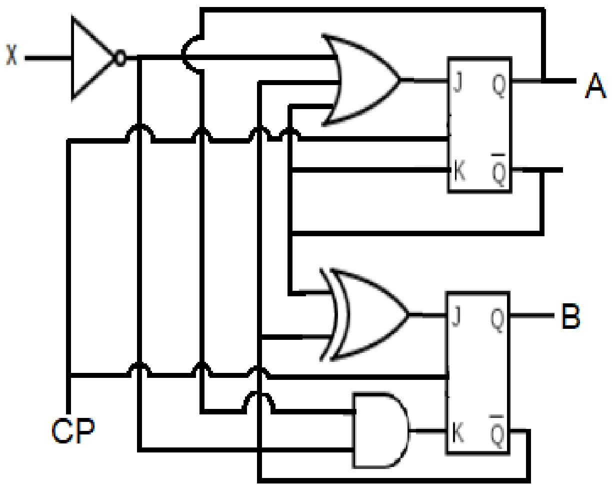
  In case of missing transition(s), use Don't Cares for the next state.

  NB: Certain transitions may occur more than once. Follow the sequence as described without confusion or assumption.

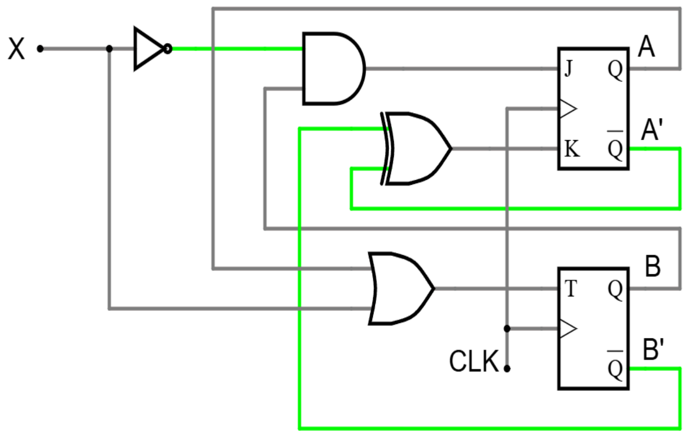
- Draw the state diagram for the given circuit.



- Draw the state diagram for the given circuit.



- Draw the state diagram for the given circuit.



Memory [Lecture 10]

- How many address lines do we need for a 64 MB RAM with 16 bit/words?
- Design a 2048x32 RAM showing internal details.
- Design a 4096x16 RAM showing internal details.
- What is the capacity for a 2^16 x 16 RAM in Gigabytes?
- Draw the block diagram of a binary cell.

- Draw and explain the functionalities of MAR and MBR. Can you find the address lines, bits per words, inputs and outputs, capacity of the RAM from the number of flip flops included in MAR and MBR?
- What type of information can you get from the “2048x32” part of a 2048x32 RAM?
- How many types of RAM are there?
- What is static RAM?
- What is dynamic RAM?
- Convert 8,796,093,022,208 Bits to Gigabytes.
- Convert 3 Terabytes to bits.