EEE 3104 and ETI 3104: Digital Electronics I

CAT II – make-up (model solution)

1. Explain the **differences** between ***bistable and astable*** *logic devices*. **(1)**
   - Bistable: two stable states (0.5)
   - Astable: no stable state, oscillator (0.5)
2. Give two applications of digital counters (1)
   - Frequency division, frequency measurement, timing, data storage, event counting, data sequence generation, digital arithmetic, measurement of pulse widths. (any two x 0.5)
3. Use a block diagram to explain the operation of sequential logic circuits. **(3)**
   - See the notes
4. Design a full adder (inputs: X, Y, C_{in} , outputs: S, C_{out}) and implement the design using 2-1 multiplexer. **(6)**

To use a multiplexer with two inputs, partition table until only a single input remains unused. Since there are two outputs, we have to partition tables for the two of them separately.

Truth table (for S): (2 mks)

X	Y	C_{in}	S	$S(C_{in})$	$S(Y, C_{in})$
0	0	0	0	C_{in}	$Y \oplus C_{in}$
0	0	1	1		
0	1	0	1	$\overline{C_{in}}$	
0	1	1	0		
1	0	0	1	$\overline{C_{in}}$	$\overline{Y \oplus C_{in}}$
1	0	1	0		
1	1	0	0	C_{in}	
1	1	1	1		

Truth table (for C_{out}): (2 mks)

X	Y	C_{in}	C_{out}	$C_{out}(C_{in})$	$C_{out}(Y, C_{in})$
0	0	0	0	0	YC_{in}
0	0	1	0		
0	1	0	0	C_{in}	
0	1	1	1		
1	0	0	0	C_{in}	$Y + C_{in}$
1	0	1	1		
1	1	0	1	1	
1	1	1	1		

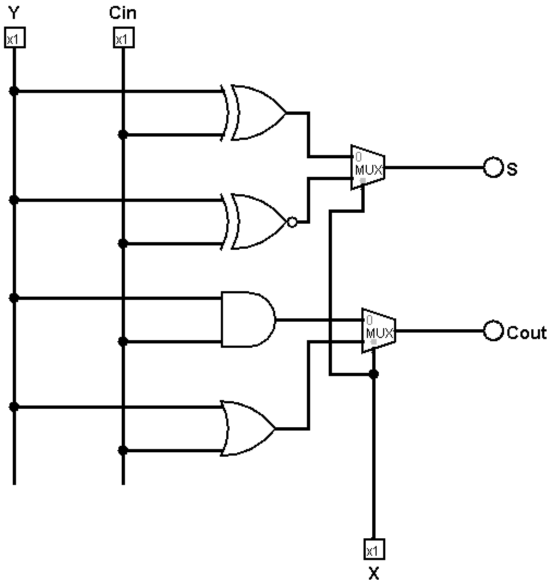


Figure 1: q4 (2 mks)

5. Implement a JK flip-flop using the SR flip-flop. **(5)**

JK flip-flop truth table and required SR inputs (3 marks for Q_{n+1}, S, and R)

JK truth table				SR inputs	
J	K	Q_n	Q_{n+1}	S	R
0	0	0	0	0	x
0	0	1	1	x	0
0	1	0	0	0	x
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	x	0
1	1	0	1	1	0
1	1	1	0	0	1

Now, with J, K and Q_n as inputs, use K-maps to get minimized expressions for S and R. (1 mark)

$$S = J\,\overline{Q_n}$$
$$R = K\,Q_n$$

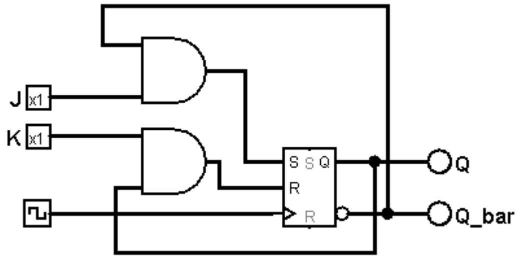


Figure 2: JK flip-flop using SR flip-flop (1 mark)

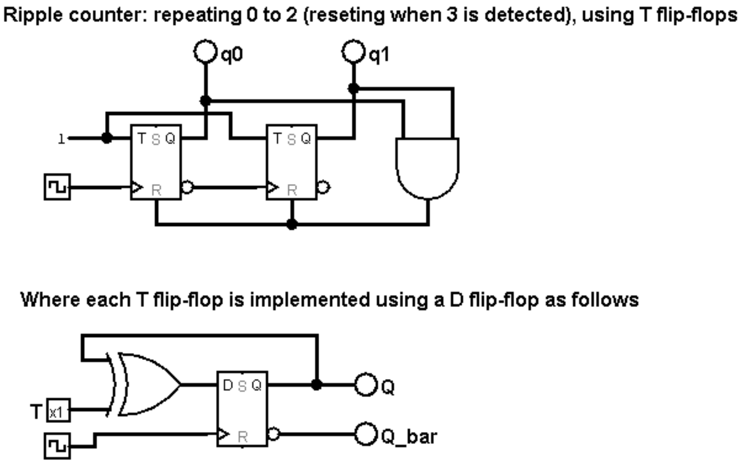
6. Design an asynchronous counter that counts from 0 to 2 using D flip-flops. **(5)**

Asynchronous counters require flip-flops that can be used in toggle mode. Therefore, first, we modify the D flip-flop into a T flip-flop then use it in toggle mode. (1 mark)

***T* truth table**			**input**
T	Q_n	Q_{n+1}	D
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

Minimizing for D using T and Q_n (1 mark)

$$D = T \oplus Q_n$$


Ripple counter: repeating 0 to 2 (reseting when 3 is detected), using T flip-flops

Where each T flip-flop is implemented using a D flip-flop as follows

Figure 3: 0 to 2 ripple counter using D flip-flops (2 marks – counter, 1 mark D to T)

7. You need a 4-bit shift right register right register for an application.

a) Suggest a flip-flop that is convenient to use for shift-register implementation. **(1)**

- D flip-flop (also known as Delay flip-flop) (1 mark)

b) Use the flip-flop suggested in 7(a) to implement the 3-bit shift right register. **(3)**

- The D-flip flop transfers current input to output on the next clock trigger
- A 3-bit shift-left register: (1 mark)
  - Uses 3 D flip-flops
  - Output of each flip-flop input to the next flip-flops

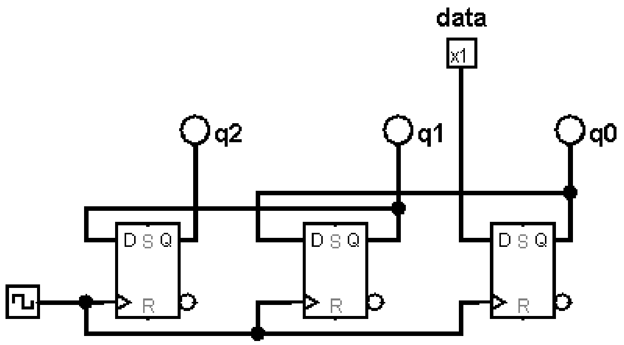


Figure 4: 3-bit shift-left register. You can draw the flip-flops facing left to make it neat. (2 marks)

8. You have the following Boolean expression: $F = (A + B + CD)E$

a) Implement the function using NAND gates only. **(3)**

- Start from an SOP (1 mark)

$$(A + B + CD)E = AE + BE + CDE = \overline{\overline{AE + BE + CDE}}$$
$$\overline{\overline{AE + BE + CDE}} = \overline{\overline{(AE)} \cdot \overline{(BE)} \cdot \overline{(CDE)}}$$

Which can be implemented: (1 mark)

- Level 1: two 2-input NAND gates (for $\overline{(AE)}$ and $\overline{(BE)}$) and 1 3-input NAND gate (for $\overline{(CDE)}$)
- Level 2: one 3-input NAND gate (for $\overline{\overline{(AE)} \cdot \overline{(BE)} \cdot \overline{(CDE)}}$)

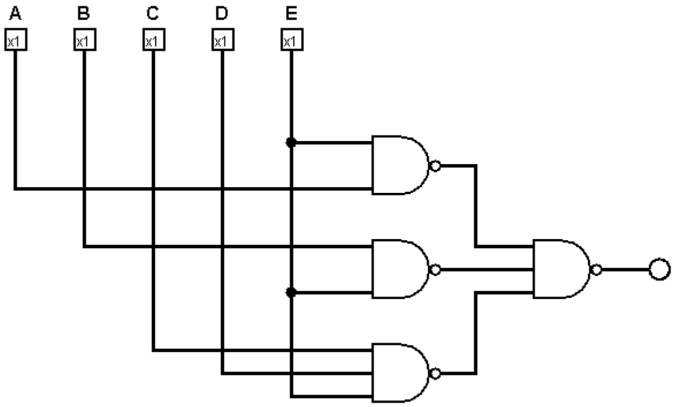


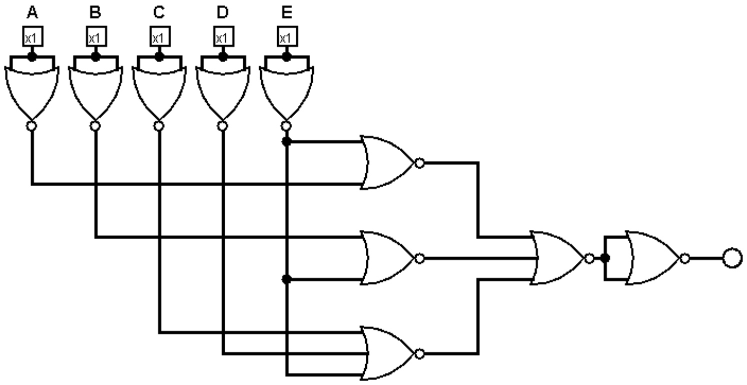
Figure 5: NAND gates-only circuit (1 mark)

b) Implement the function using NOR gates only. **(3)**

- From an SOP go to POS and get a form that is convenient for NOR gates implementation.(1 mark)
- $\overline{\overline{AE + BE + CDE}} = \overline{\overline{(AE)} \cdot \overline{(BE)} \cdot \overline{(CDE)}} = \overline{(\bar{A} + \bar{E})} + \overline{(\bar{B} + \bar{E})} + \overline{(\bar{C} + \bar{D} + \bar{E})}$
- $\overline{(\bar{A} + \bar{E})} + \overline{(\bar{B} + \bar{E})} + \overline{(\bar{C} + \bar{D} + \bar{E})} = \overline{\overline{\overline{(\bar{A} + \bar{E})} + \overline{(\bar{B} + \bar{E})} + \overline{(\bar{C} + \bar{D} + \bar{E})}}}$
- This one can be implemented in three levels (1 mark)

  - ○ Level 1: 2 2-input NOR gates (for $\overline{(\bar{A} + \bar{E})}$ and $\overline{(\bar{B} + \bar{E})}$) and 1 3-input NOR gate (for $\overline{(\bar{C} + \bar{D} + \bar{E})}$)
  - ○ Level 2: 1 3 input NOR gate (for $\overline{\overline{(\bar{A} + \bar{E})} + \overline{(\bar{B} + \bar{E})} + \overline{(\bar{C} + \bar{D} + \bar{E})}}$)
  - ○ Level 3: 1 2 input NOR gate (for $\overline{\overline{\overline{(\bar{A} + \bar{E})} + \overline{(\bar{B} + \bar{E})} + \overline{(\bar{C} + \bar{D} + \bar{E})}}}$)

- In addition, the literals have to be inverted using 2 input nor gates.



-

Figure 6: NOR gates-only implementation (1 mark).

For Question 8 a and b, you do not need to do all that writing, just write the expressions in the desired format and draw the circuits. The explanations here are just to help with your understanding.

This applies to all questions with extra explanations.

Ignore the horizontal lines in q8, it is just a formatting problem.