

Digital Techniques (313303)

Model Question Paper — Set A · Model Answers

Q.1

a) Base and symbols of number systems.

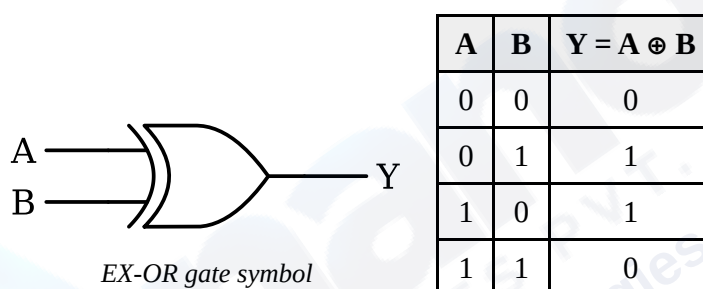
2 M

Number system	Base (radix)	Symbols used	Example
Binary	2	0, 1	$(1011)_2$
Octal	8	0 to 7	$(257)_8$
Decimal	10	0 to 9	$(479)_{10}$
Hexadecimal	16	0 to 9, A, B, C, D, E, F	$(3AF)_{16}$

(½ M each)

b) EX-OR gate: symbol and truth table.

2 M



$Y = A \oplus B = \bar{A}B + A\bar{B}$ — output is 1 when the inputs are different.

(Symbol 1 M, truth table 1 M)

c) Commutative and associative laws.

2 M

Commutative laws: the order of variables does not change the result.

i) $A + B = B + A$ ii) $A \cdot B = B \cdot A$

Associative laws: the grouping of variables does not change the result.

i) $A + (B + C) = (A + B) + C$ ii) $A \cdot (B \cdot C) = (A \cdot B) \cdot C$

(1 M each law)

d) Excitation table of J-K flip-flop.

2 M

Q_n	Q_{n+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

X = don't care (0 or 1).

e) Applications of multiplexer.

2 M

1. Data selector / data routing (many inputs to one line).
2. Parallel-to-serial data conversion.
3. Logic function generator (implementing Boolean functions).
4. Time-division multiplexing in communication systems (sending many signals on one line).
5. Selecting one of many sensor/ADC inputs in data acquisition systems.

(Any four, $\frac{1}{2}$ M each)

f) Modulus of counter; flip-flops for MOD-12.

2 M

Modulus (MOD number) of a counter is the number of distinct states the counter goes through before it returns to its initial state. An n-bit counter can have maximum 2^n states.

For MOD-12: choose the smallest n such that $2^n \geq 12 \rightarrow 2^3 = 8 < 12$, $2^4 = 16 \geq 12$. $\therefore$ **4 flip-flops are required.**

(Definition 1 M, answer 1 M)

g) SRAM vs DRAM.

2 M

SRAM	DRAM
Stores a bit in a flip-flop (latch) made of transistors (about 6 per cell).	Stores a bit as charge on a capacitor (1 transistor + 1 capacitor per cell).
No refreshing is required.	Must be refreshed every few milliseconds.
Faster (smaller access time).	Slower than SRAM.

(Any two points, 1 M each)

Q.2

a) Conversions.

4 M

i) $(157.625)_{10} \rightarrow$ binary

Integer part (157): divide repeatedly by 2, write remainders from bottom to top.

Division	Quotient	Remainder
$157 \div 2$	78	1
$78 \div 2$	39	0
$39 \div 2$	19	1
$19 \div 2$	9	1
$9 \div 2$	4	1
$4 \div 2$	2	0
$2 \div 2$	1	0
$1 \div 2$	0	1

Fraction part (0.625): multiply repeatedly by 2, take the integer parts from top to bottom.

Fraction $\times 2$	Product	Integer part
0.625×2	1.25	1
0.25×2	0.5	0
0.5×2	1	1

$$\therefore (157.625)_{10} = (10011101.101)_2$$

ii) $(3B9)_{16} \rightarrow$ octal (through binary)

Hex $\rightarrow$ Binary: replace each hex digit by its 4-bit binary equivalent.

Hex digit	3	B	9
Binary	0011	1011	1001

$$(3B9)_{16} = (1110111001)_2$$

Binary $\rightarrow$ Octal: make groups of 3 bits from the binary point (add 0s at the ends if needed).

3-bit group	001	110	111	001
Octal	1	6	7	1

$$\therefore (3B9)_{16} = (1110111001)_2 = (1671)_8$$

$$\text{Check: } (3B9)_{16} = 3 \times 256 + 11 \times 16 + 9 = 953 \text{ and } (1671)_8 = 953 \checkmark$$

iii) $(725)_{10} \rightarrow$ BCD

Decimal digit	7	2	5
BCD (8421)	0111	0010	0101

$$\therefore (725)_{10} = (0111 \ 0010 \ 0101)_{BCD}$$

iv) $(110101)_2 \rightarrow$ Gray code

Binary $\rightarrow$ Gray: MSB is copied as it is; every next Gray bit = XOR of the present and previous binary bits.

Binary	1	1	0	1	0	1
XOR step (copy)	(copy)	$1 \oplus 1$	$1 \oplus 0$	$0 \oplus 1$	$1 \oplus 0$	$0 \oplus 1$
Gray	1	0	1	1	1	1

$$\therefore (110101)_2 = (101111)_{\text{Gray}}$$

(1 M each)

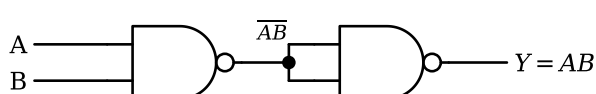
b) Basic gates using only NAND gates.

4 M

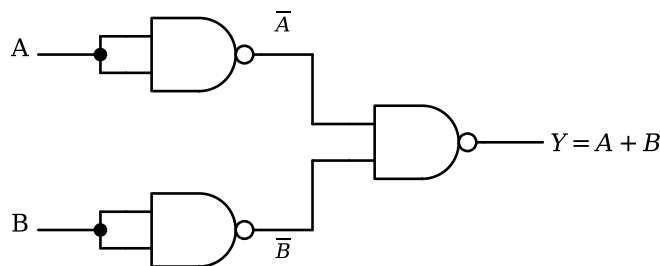
NAND is a **universal gate**: any logic function can be built using only NAND gates. $Y = \overline{AB}$.



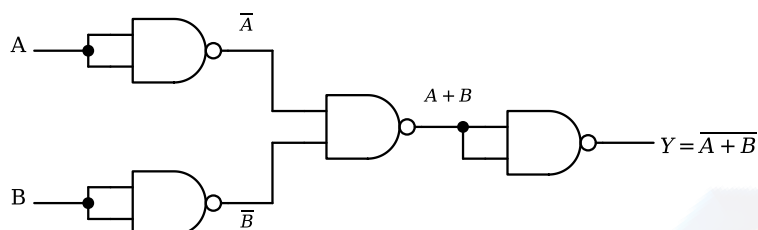
i) NOT gate: inputs tied together



ii) AND gate: NAND followed by NAND-inverter



iii) OR gate (De Morgan: NOT A NAND NOT B = A + B)



iv) NOR gate: OR followed by NAND-inverter

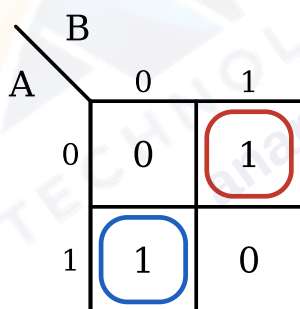
(Each gate 1 M)

c) Half adder design.

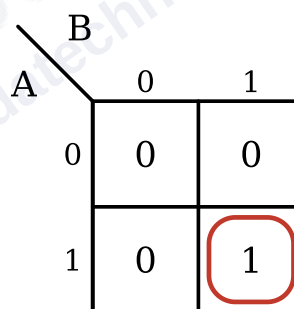
4 M

A **half adder** adds two 1-bit numbers A and B and gives Sum (S) and Carry (C).

A	B	Sum S	Carry C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

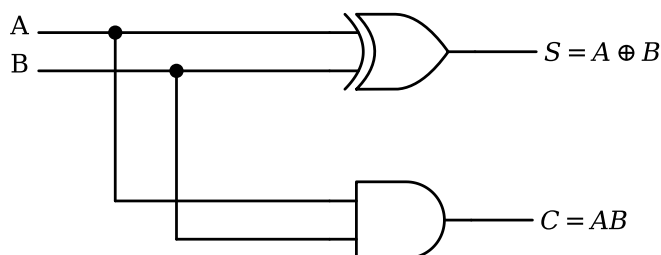


K-map for S (no adjacent 1s)



K-map for C

$$S = \bar{A}B + A\bar{B} = A \oplus B \quad C = AB$$

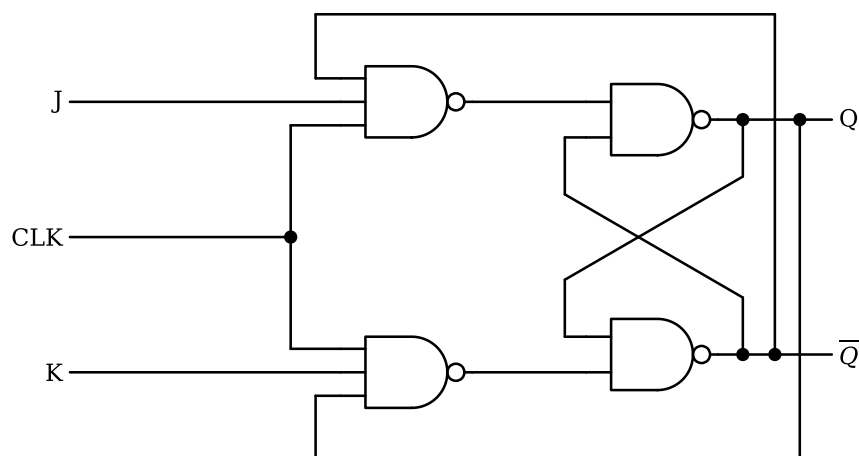


Logic diagram of half adder

(Truth table 1 M, K-map/expressions 1 M, logic diagram 2 M)

d) Clocked J-K flip-flop.

4 M



Clocked J-K flip-flop using NAND gates

Working: When CLK = 0, the input NAND gates are disabled (outputs = 1), so the latch holds its state. When CLK = 1:

- J = 0, K = 0: both input gates give 1 → no change ($Q_{n+1} = Q_n$).
- J = 0, K = 1: lower gate is enabled (through $\bar{Q}$ feedback) → Q is reset to 0.
- J = 1, K = 0: upper gate is enabled (through Q feedback) → Q is set to 1.
- J = 1, K = 1: only the gate whose feedback is 1 is enabled, so the output changes to its complement → **toggle**.

CLK	J	K	Q_{n+1}	Action
0	X	X	Q_n	No change
1	0	0	Q_n	No change
1	0	1	0	Reset
1	1	0	1	Set
1	1	1	$\bar{Q}_n$	Toggle

(Diagram 2 M, working + truth table 2 M)

Q.3

a) K-map: $F = \sum m(1, 3, 4, 5, 9, 11, 12, 13)$.

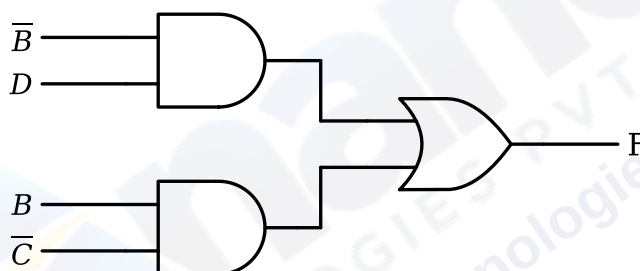
4 M

CD \ AB	00	01	11	10
	00	1	1	0
01	1	1	0	0
11	1	1	0	0
10	0	1	1	0

K-map for F (grouping 1s)

Group	Cells (minterms)	Term
Group 1	1, 3, 9, 11	$\overline{B}D$
Group 2	4, 5, 12, 13	$B\overline{C}$

Minimal SOP: $F = \overline{B}D + B\overline{C}$



Logic diagram (AND-OR)

(K-map 1 M, grouping 1 M, expression 1 M, logic diagram 1 M)

b) $(46)_{10} - (75)_{10}$ by 2's complement.

4 M

$(46)_{10} - (75)_{10} = (46)_{10} + (2\text{'s complement of } (75)_{10})$. Use 8-bit numbers.

$(46)_{10} = 00101110$ $(75)_{10} = 01001011$

1's complement of $01001011 = 10110100$

2's complement = $10110100 + 1 = \mathbf{10110101}$

```

00101110   (46)
+ 10110101 (2's complement of 75)
-----
11100011

```

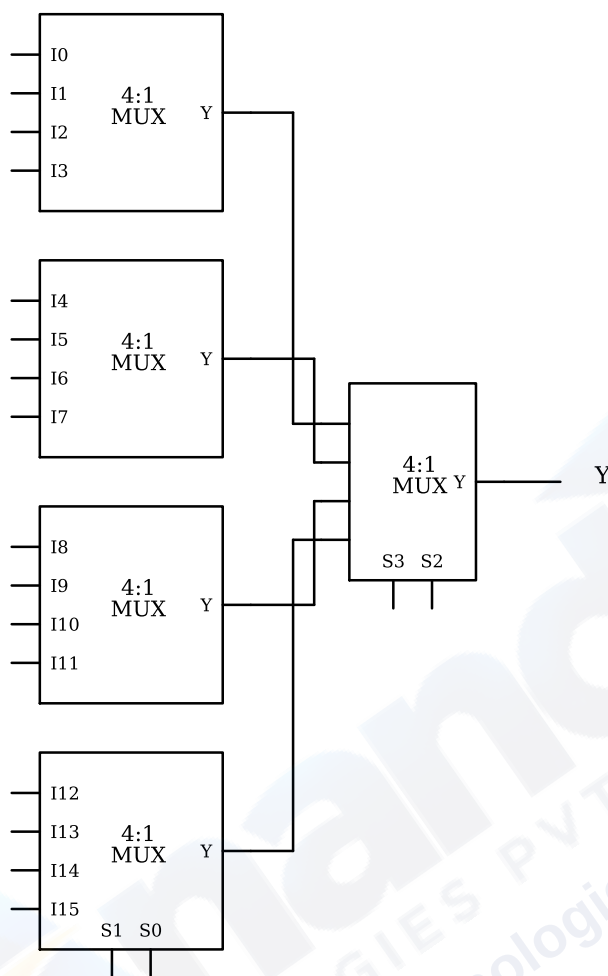
No final carry $\rightarrow$ the result is **negative** and is in 2's complement form.

Take 2's complement of 11100011 : 1's complement = 00011100 , $+ 1 = 00011101 = 29$

$\therefore (46)_{10} - (75)_{10} = -29$ (check: $46 - 75 = -29$)

c) 16 : 1 MUX using 4 : 1 MUX.

A 16:1 MUX has 16 data inputs (I0–I15) and 4 select lines (S3 S2 S1 S0). Using 4:1 MUXes: **four** 4:1 MUXes in the first level select one input out of each group of four using S1 S0, and **one** 4:1 MUX in the second level selects one of these four outputs using S3 S2. Total = 5 MUXes.



(S1, S0 common to all)

16:1 MUX using five 4:1 MUX (MUX tree)

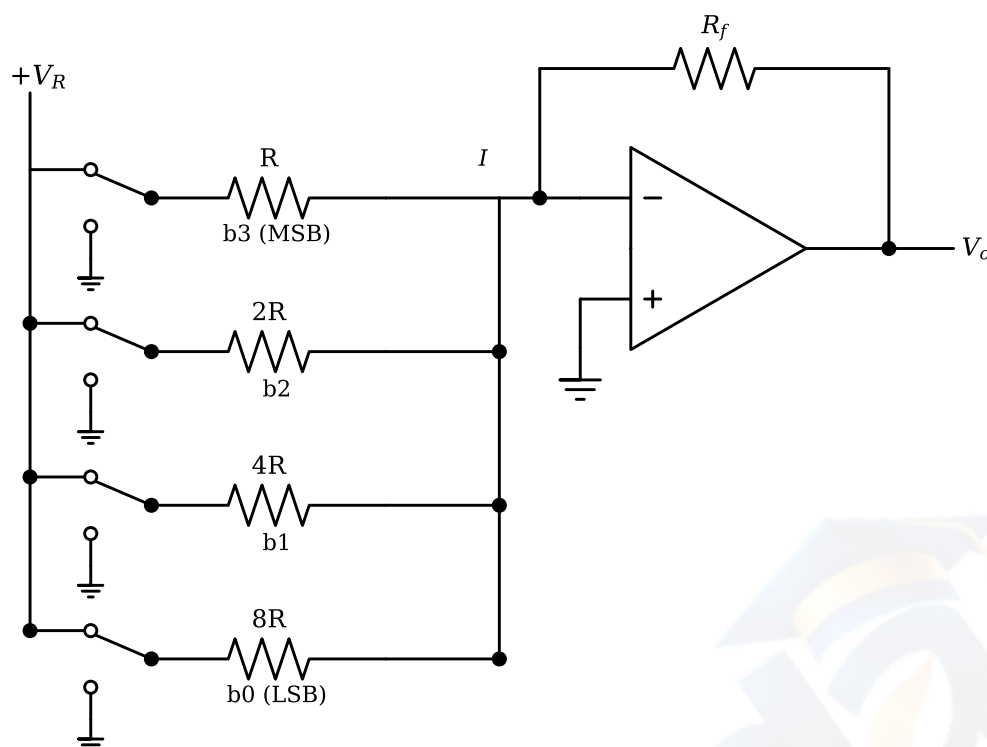
S3 S2	MUX selected (level 1)	Inputs passed by S1 S0
0 0	MUX-1	I0 – I3
0 1	MUX-2	I4 – I7
1 0	MUX-3	I8 – I11
1 1	MUX-4	I12 – I15

Example: S3 S2 S1 S0 = 1001 → level-1 MUX-3 passes I9 (S1 S0 = 01) and the final MUX selects its input 2 (S3 S2 = 10) → Y = I9.

(Diagram 3 M, explanation 1 M)

d) Weighted resistor DAC.

4 M



Switch up = bit 1 (to $+V_R$), switch down = bit 0 (to ground)

4-bit binary weighted resistor DAC

The resistors are binary weighted: MSB $b_3 \rightarrow R$, $b_2 \rightarrow 2R$, $b_1 \rightarrow 4R$, LSB $b_0 \rightarrow 8R$. Each switch connects its resistor to $+V_R$ when the bit is 1 and to ground when the bit is 0. The op-amp works as a summing amplifier; its inverting input is at virtual ground.

Derivation: current through each resistor = $V_R \cdot b_i / R_i$

$$I = V_R (b_3/R + b_2/2R + b_1/4R + b_0/8R), \quad V_o = -I \cdot R_f$$

$$V_o = -V_R R_f (b_3/R + b_2/2R + b_1/4R + b_0/8R) = -(V_R R_f / 8R) (8b_3 + 4b_2 + 2b_1 + b_0)$$

So the output is proportional to the binary value of the input. Drawback: wide range of resistor values is needed for more bits.

(Diagram 2 M, derivation 2 M)

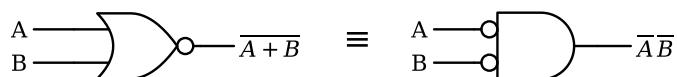
Q.4

a) De Morgan's theorems.

4 M

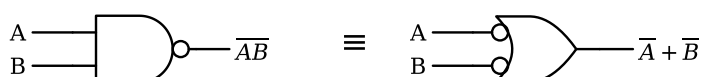
Theorem 1: The complement of a sum is equal to the product of the complements: $\overline{A + B} = \overline{A} \cdot \overline{B}$ (NOR = bubbled AND).

A	B	$\overline{A + B}$	$\overline{A} \cdot \overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0



Theorem 2: The complement of a product is equal to the sum of the complements: $\overline{A \cdot B} = \overline{A} + \overline{B}$ (NAND = bubbled OR).

A	B	$\overline{A} \overline{B}$	$\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

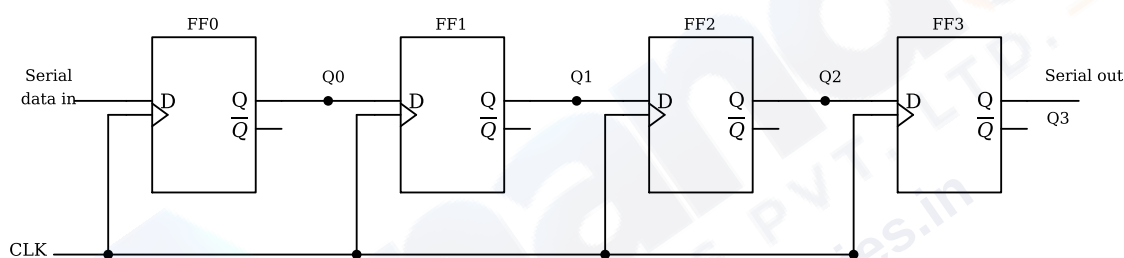


In both tables the last two columns are identical for all input combinations, hence both theorems are proved.

(Statements 1 M, proof 2 M, diagrams 1 M)

b) 4-bit SISO shift register.

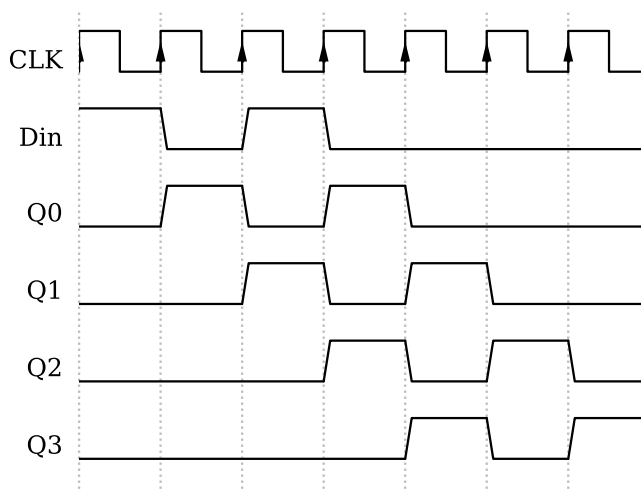
4 M



4-bit SISO shift register using D flip-flops

The Q output of each flip-flop is connected to the D input of the next. Data enters FF0 serially, one bit per clock pulse, and every clock pulse shifts the stored bits one position to the right. After 4 clock pulses the 4-bit word is stored; it comes out serially at Q3 with the next pulses (total 7 clocks for full serial out).

Clock	D _{in}	Q0	Q1	Q2	Q3
Initial	—	0	0	0	0
Clock 1	1	1	0	0	0
Clock 2	0	0	1	0	0
Clock 3	1	1	0	1	0
Clock 4	0	0	1	0	1
Clock 5	0	0	0	1	0
Clock 6	0	0	0	0	1
Clock 7	0	0	0	0	0



Waveforms (positive-edge triggered, data 1010, first bit = 1)

(Diagram 2 M, table/waveform 2 M)

c) TTL vs CMOS.

4 M

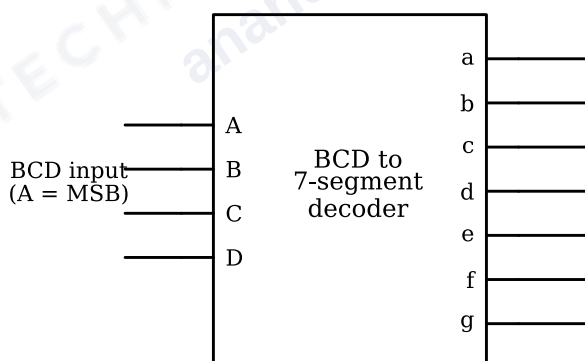
Parameter	TTL (74 series)	CMOS (4000 series)
i) Propagation delay	Small, about 10 ns per gate	Larger, about 50–100 ns per gate
ii) Power dissipation	More, about 10 mW per gate	Very low, about 0.01 mW (static)
iii) Noise margin	Low, about 0.4 V	High, about 1.5 V (at 5 V supply)
iv) Fan-out	10	50 (more)
Supply voltage	+5 V $\pm$ 5%	+3 V to +15 V
Basic gate	NAND	NAND / NOR (inverter)

(1 M each point)

d) BCD to 7-segment decoder.

4 M

The decoder converts a 4-bit BCD input (A B C D, A = MSB) into seven outputs a–g that drive the segments of the display. For a **common-cathode** display a segment glows when its output is 1.



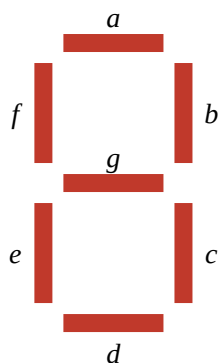


Fig.: BCD to 7-segment decoder driving a common-cathode display

Digit	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
10–15	X	X	X	X	X	X	X	X	X	X	X

Inputs 1010–1111 are invalid BCD codes and are treated as don't care (X).

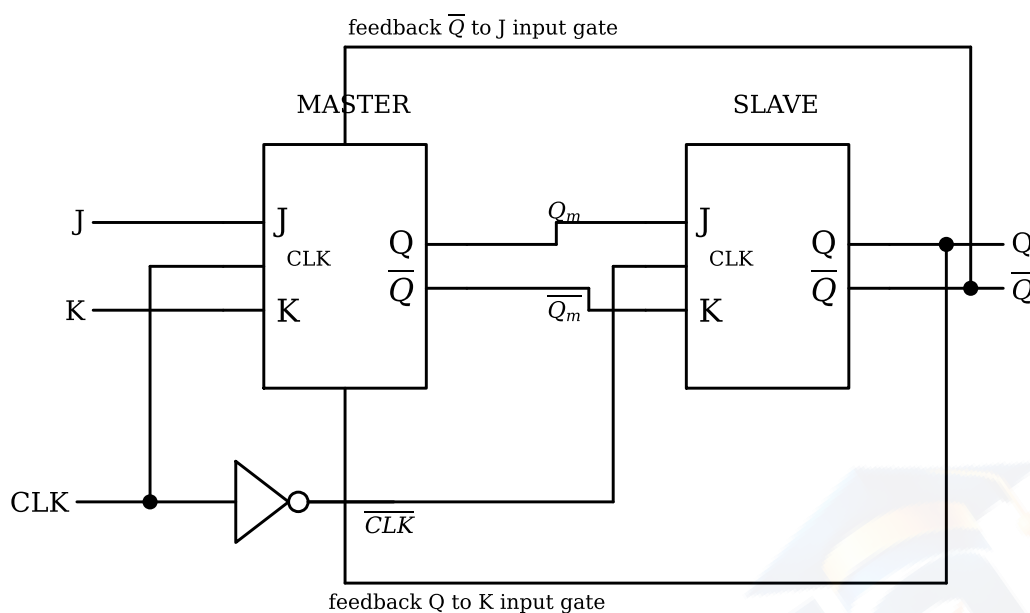
(Block diagram 2 M, truth table 2 M)

e) Race-around condition; master-slave J-K.

4 M

Race-around condition: In a level-triggered J-K flip-flop with $J = K = 1$, the output toggles. If the clock pulse width t_p is larger than the propagation delay Δt of the flip-flop, the output toggles again and again while $CLK = 1$. At the end of the pulse the output is uncertain. This is called race-around.

Ways to avoid: i) keep $t_p < \Delta t < T$ (difficult), ii) use edge triggering, iii) use a **master-slave J-K flip-flop**.



Master-slave J-K flip-flop

Working: When $CLK = 1$ the master follows J and K while the slave is disabled (its clock is $\overline{CLK} = 0$), so the output Q does not change. When CLK goes to 0 the master is disabled (its outputs are frozen) and the slave copies the master. Thus the output changes only once per clock pulse, at the negative edge, and race-around cannot occur.

(Race-around 2 M, master-slave explanation 2 M)

Q.5

a) SOP / POS; standard POS of $\Sigma m(0, 2, 3, 6, 7)$.

6 M

SOP (Sum of Products): the expression is a logical sum (OR) of product (AND) terms, e.g. $Y = AB + \overline{A}C$. If every term contains all variables it is **standard / canonical SOP** (sum of minterms).

POS (Product of Sums): the expression is a logical product (AND) of sum (OR) terms, e.g. $Y = (A + B)(\overline{A} + C)$. If every term contains all variables it is **standard / canonical POS** (product of maxterms).

Standard POS: $F = \Sigma m(0, 2, 3, 6, 7) \rightarrow F$ is 0 for the remaining minterms 1, 4, 5. Hence $F = \Pi M(1, 4, 5)$. For a maxterm, a variable that is 0 is written true and a variable that is 1 is complemented:

Maxterm	A B C	Sum term
M1	0 0 1	$(A + B + \overline{C})$
M4	1 0 0	$(\overline{A} + B + C)$
M5	1 0 1	$(\overline{A} + B + \overline{C})$

$$F = (A + B + \overline{C}) \cdot (\overline{A} + B + C) \cdot (\overline{A} + B + \overline{C}) \text{ (standard POS)}$$

Minimization using K-map (grouping 0s):

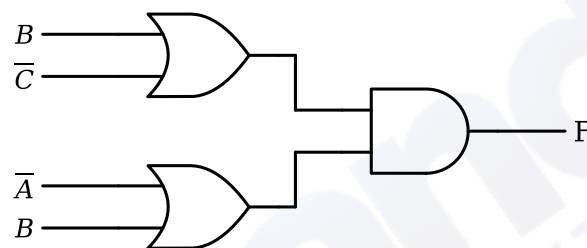
BC A		00	01	11	10
		0	1	0	1
	1	0	0	1	1

K-map for F (grouping 0s)

For POS, 0s are grouped; in each group a variable that is 0 is written as it is and a variable that is 1 is complemented (sum term).

Group	Cells (maxterms)	Sum term
Group 1	1, 5	$(B + \bar{C})$
Group 2	4, 5	$(\bar{A} + B)$

Minimal POS: $F = (B + \bar{C})(\bar{A} + B)$



Logic diagram (OR-AND)

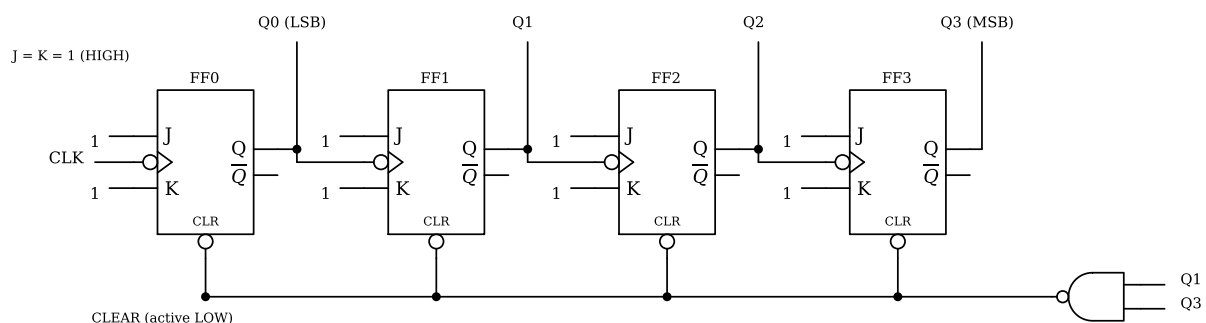
Check (SOP from the same K-map): $F = B + \bar{A}\bar{C}$; both forms give identical truth tables (verified for all 8 inputs).

(Definitions 2 M, standard POS 2 M, K-map + diagram 2 M)

b) MOD-10 asynchronous counter.

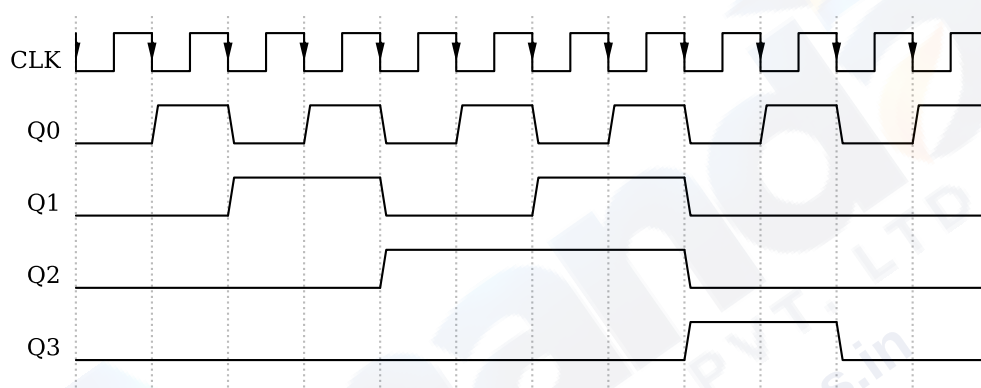
6 M

MOD-10 needs 4 flip-flops ($2^4 = 16 \geq 10$). The counter must count 0000 $\rightarrow$ 1001 and reset to 0000 at the 10th clock pulse. When the count reaches 1010 (decimal 10), $Q_3 = 1$ and $Q_1 = 1$ for the first time, so a NAND gate with inputs Q_3 and Q_1 gives a 0 that clears all flip-flops immediately. All J and K inputs are kept at 1 (toggle mode); each flip-flop is negative-edge triggered and its clock comes from the Q output of the previous flip-flop.



MOD-10 (decade) ripple counter using J-K flip-flops

Clock	Q3	Q2	Q1	Q0	Decimal
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	2
3	0	0	1	1	3
4	0	1	0	0	4
5	0	1	0	1	5
6	0	1	1	0	6
7	0	1	1	1	7
8	1	0	0	0	8
9	1	0	0	1	9
10	1 → 0	0	1 → 0	0	10 → 0 (reset)



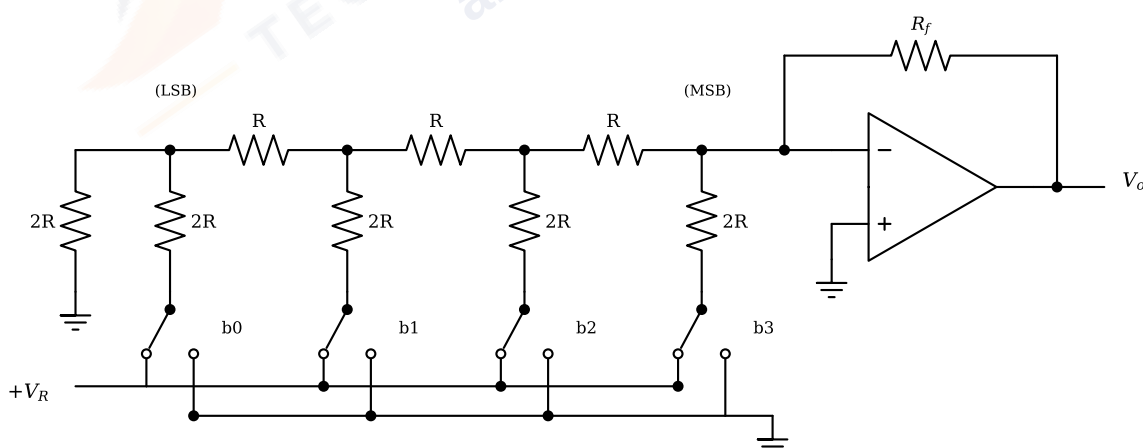
Timing diagram of MOD-10 ripple counter ($Q_0 = \text{LSB}$); after the 10th pulse the counter returns to 0000

(The state 1010 exists only for a very short time — a few nanoseconds — before the counter is cleared.)

(Design/logic 1 M, diagram 2 M, truth table 1 M, timing diagram 2 M)

c) 4-bit R-2R ladder DAC; V_o for 1010.

6 M



4-bit R-2R ladder DAC (inverting summing amplifier)

Working: Only two resistor values, R and $2R$, are used. Each $2R$ leg is switched to $+V_R$ when its bit is 1 and to ground when the bit is 0. The MSB end of the ladder is connected to the inverting input of the op-amp, which is at virtual ground. Looking into any node of the ladder the resistance is $2R$, so the current divides into two equal parts at every node; therefore the currents of the bits b_3, b_2, b_1, b_0 are $V_R/2R, V_R/4R, V_R/8R$ and $V_R/16R$ — binary weighted.

$$I = (V_R/R)(b_3/2 + b_2/4 + b_1/8 + b_0/16), \quad V_o = -I R_f$$

$$V_o = -(R_f/R) V_R (b_3/2 + b_2/4 + b_1/8 + b_0/16)$$

Numerical: $D = 1010$ ($b_3 = 1, b_2 = 0, b_1 = 1, b_0 = 0$), $V_R = 5 \text{ V}$, $R_f = R$

$$V_o = -5 \times (1/2 + 0 + 1/8 + 0) = -5 \times 0.625 = -3.125 \text{ V}$$

(Equivalently $V_o = -V_R \times D/16 = -5 \times 10/16$.)

Advantages over weighted resistor DAC: only two resistor values (easy to make accurate), easily expanded to more bits.

(Diagram 2 M, working + expression 2 M, numerical 2 M)

Q.6

a) 3-bit synchronous up counter using J-K flip-flops.

6 M

Step 1: 3 flip-flops ($Q_2 Q_1 Q_0$) are required; sequence $000 \rightarrow 001 \rightarrow \dots \rightarrow 111 \rightarrow 000$. All flip-flops get the same clock.

Step 2: Excitation table (J-K excitation: $0 \rightarrow 0$: $J=0, K=X$; $0 \rightarrow 1$: $J=1, K=X$; $1 \rightarrow 0$: $J=X, K=1$; $1 \rightarrow 1$: $J=X, K=0$)

Q_2	Q_1	Q_0	Q_2^+	Q_1^+	Q_0^+	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	1	1	1	X	0	X	0	1	X
1	1	1	0	0	0	X	1	X	1	X	1

Step 3: K-maps

		$Q_1 Q_0$			
		00	01	11	10
Q_2	0	0	0	1	0
	1	X	X	X	X

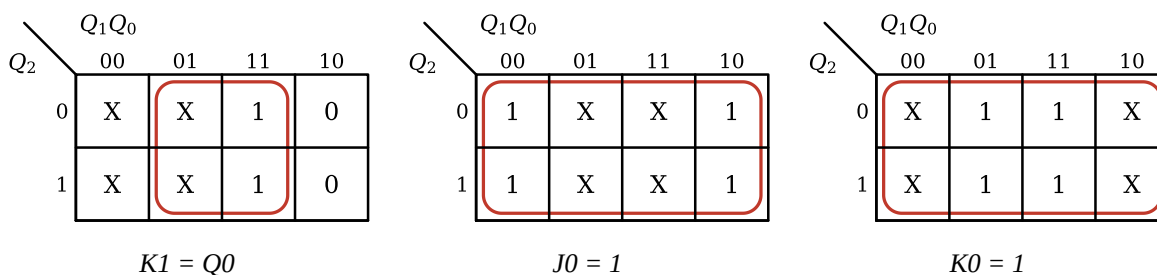
$$J_2 = Q_1 Q_0$$

		$Q_1 Q_0$			
		00	01	11	10
Q_2	0	X	X	X	X
	1	0	0	1	0

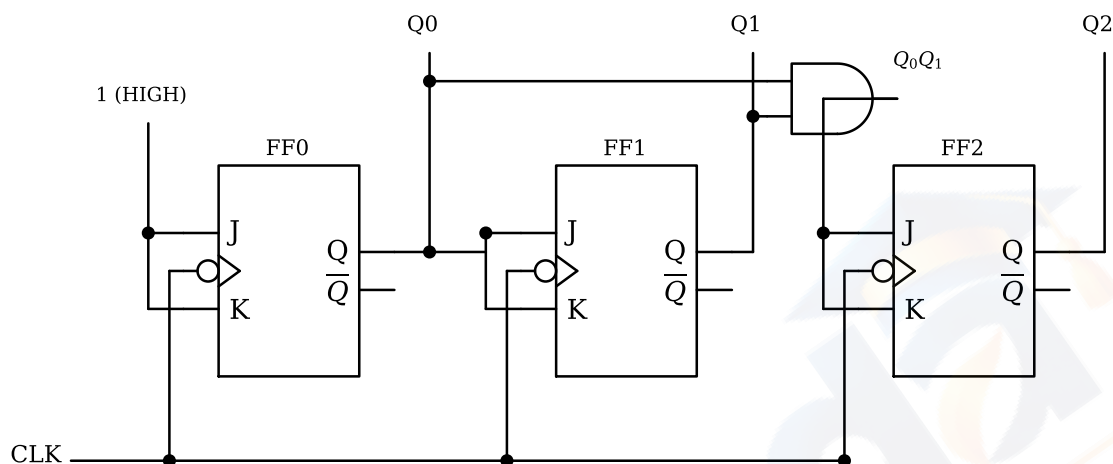
$$K_2 = Q_1 Q_0$$

		$Q_1 Q_0$			
		00	01	11	10
Q_2	0	0	1	X	X
	1	0	1	X	X

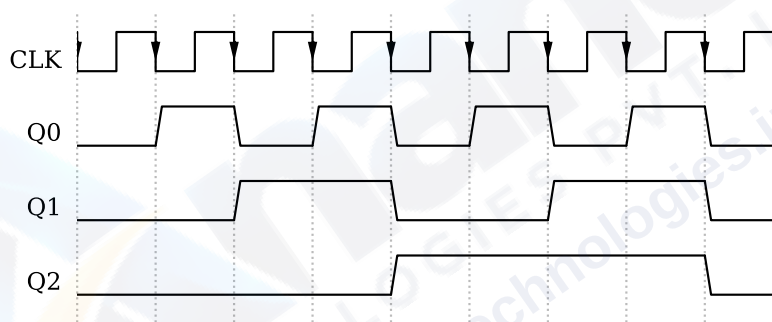
$$J_1 = Q_0$$



$$J0 = K0 = 1, \quad J1 = K1 = Q0, \quad J2 = K2 = Q1Q0$$



3-bit synchronous up counter (common clock)



Output waveforms: $Q0 = f/2$, $Q1 = f/4$, $Q2 = f/8$

(Excitation table 2 M, K-maps 1 M, diagram 2 M, waveforms 1 M)

b) Volatile vs non-volatile; EPROM vs EEPROM.

6 M

i) Volatile and non-volatile memory

Volatile memory	Non-volatile memory
Stored data is lost when power is switched off.	Stored data is retained even when power is switched off.
Used for temporary storage of programs and data while working.	Used for permanent storage (boot program, firmware, files).
Generally faster.	Generally slower (except ROM read).
Examples: SRAM, DRAM.	Examples: ROM, PROM, EPROM, EEPROM, Flash, SSD.

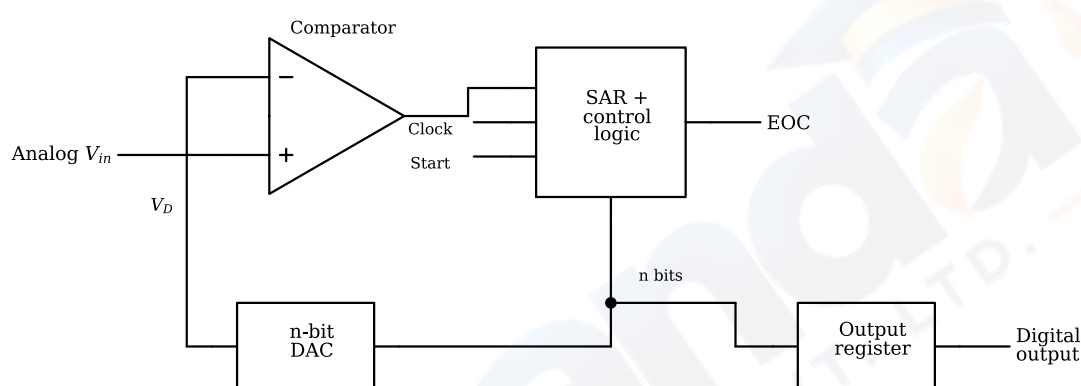
ii) EPROM and EEPROM

EPROM	EEPROM
Erased by exposing the chip to ultraviolet (UV) light through a quartz window.	Erased electrically by applying a suitable voltage.
Whole chip is erased at a time.	Individual bytes (locations) can be erased and rewritten.
Chip must be removed from the circuit for erasing; erasing takes about 15–20 minutes.	Erased and reprogrammed in the circuit; takes milliseconds.
Cheaper, used in older firmware / development work.	Costlier per bit; used to store settings, BIOS data, etc.

(Any three points each, 1 M each)

c) Successive approximation ADC.

6 M



Block diagram of successive approximation ADC

Working:

1. On start of conversion the SAR is cleared and its MSB is set to 1. The DAC converts this code into voltage V_D .
2. The comparator compares V_{in} with V_D . If $V_D > V_{in}$ the bit is reset to 0, otherwise it is kept 1.
3. The next lower bit is then set to 1 and the comparison is repeated. In this way each bit is decided, one per clock pulse, from MSB to LSB.
4. After n clock pulses the SAR contains the digital equivalent of V_{in} ; it is transferred to the output register and EOC is given. Conversion time is fixed: n clock periods.

Example: 4-bit SAR ADC, DAC step = 1 V (full scale 15 V), $V_{in} = 10.6$ V

Bit tested	SAR code	DAC output V_D	Decision
b3	1000	8 V	$8 \leq 10.6 \rightarrow$ keep 1
b2	1100	12 V	$12 > 10.6 \rightarrow$ reset to 0
b1	1010	10 V	$10 \leq 10.6 \rightarrow$ keep 1
b0	1011	11 V	$11 > 10.6 \rightarrow$ reset to 0

Digital output = 1010 (= 10 V, nearest lower level).

Applications of ADC: digital voltmeters / multimeters; data acquisition systems (temperature, pressure sensors to computer); digital audio recording; mobile phones and digital communication.

(Block diagram 2 M, working 2 M, example 1 M, applications 1 M)

Note: This is a **model question paper prepared by Ananda Technologies** only for practice and exam preparation, based on the pattern of previous MSBTE question papers and the K-scheme syllabus. It is **not an official MSBTE paper**, and it does **not** mean that these questions will come in the exam. Answers follow standard textbook points; marks shown are indicative. Free for students — anandatechnologies.in

