

(Diagram 1 M, truth table 1 M)

e) Resolution of DAC.

2 M

Resolution of a DAC is the smallest change in analog output voltage for a change of 1 LSB in the digital input. It is also expressed as a fraction (or %) of full scale: $1/(2^n - 1)$.

For $n = 8$, $V_{FS} = 10 \text{ V}$: Resolution = $V_{FS}/(2^8 - 1) = 10/255 = \mathbf{39.2 \text{ mV}}$ per step; % resolution = $100/255 = \mathbf{0.392 \%}$.

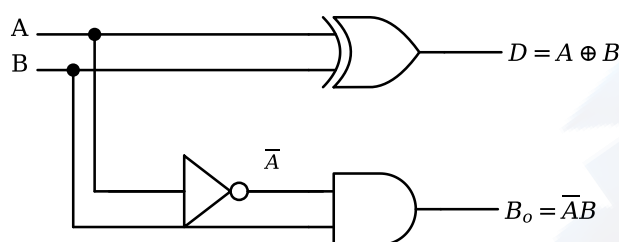
(Some books use 2^n in the denominator: $10/256 = 39.06 \text{ mV}$ — nearly the same value.)

(Definition 1 M, calculation 1 M)

f) Half subtractor.

2 M

Difference $D = \bar{A}B + A\bar{B} = A \oplus B$; Borrow $B_o = \bar{A}B$



Half subtractor logic diagram

(Expressions 1 M, diagram 1 M)

g) Combinational vs sequential circuits.

2 M

Combinational circuit	Sequential circuit
Output depends only on the present inputs.	Output depends on present inputs and past outputs (stored state).
No memory element.	Has memory elements (flip-flops).
No clock is needed.	Usually clock driven.
Examples: adder, MUX, decoder.	Examples: flip-flop, counter, shift register.

(Any two points, 1 M each)

Q.2

a) 2's complement subtraction.

4 M

i) $58 - 23$

$(58)_{10} - (23)_{10} = (58)_{10} + (2\text{'s complement of } (23)_{10})$. Use 8-bit numbers.

$(58)_{10} = 00111010$ $(23)_{10} = 00010111$

1's complement of $00010111 = 11101000$

2's complement = $11101000 + 1 = \mathbf{11101001}$

00111010	(58)
+ 11101001	(2's complement of 23)

100100011

A final carry is generated → discard it. The result is positive: 00100011 = +35

ii) 23 – 58

$(23)_{10} - (58)_{10} = (23)_{10} + (2\text{'s complement of } (58)_{10})$. Use 8-bit numbers.

$(23)_{10} = 00010111$ $(58)_{10} = 00111010$

1's complement of 00111010 = 11000101

2's complement = 11000101 + 1 = **11000110**

```

00010111   (23)
+ 11000110 (2's complement of 58)
-----
11011101

```

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

Take 2's complement of 11011101: 1's complement = 00100010, + 1 = 00100011 = 35

∴ $(23)_{10} - (58)_{10} = -35$ (check: $23 - 58 = -35$)

(2 M each)

b) De Morgan's theorems; simplify $Y = \overline{\overline{A}B + \overline{C}}$.

4 M

Theorem 1: $\overline{A + B} = \overline{A} \cdot \overline{B}$ **Theorem 2:** $\overline{A \cdot B} = \overline{A} + \overline{B}$

Step	Rule
$Y = \overline{\overline{A}B + \overline{C}}$	Given
$= \overline{\overline{A}B} \cdot \overline{\overline{C}}$	De Morgan theorem 1
$= (\overline{\overline{A}} + \overline{\overline{B}}) \cdot C$	De Morgan theorem 2 and double complement
$= (A + \overline{B}) \cdot C = AC + \overline{B}C$	Double complement, distributive law

$Y = (A + \overline{B})C = AC + \overline{B}C$

A	B	C	$\overline{\overline{A}B + \overline{C}}$	$(A + \overline{B})C$
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	1	1
1	1	0	0	0
1	1	1	1	1

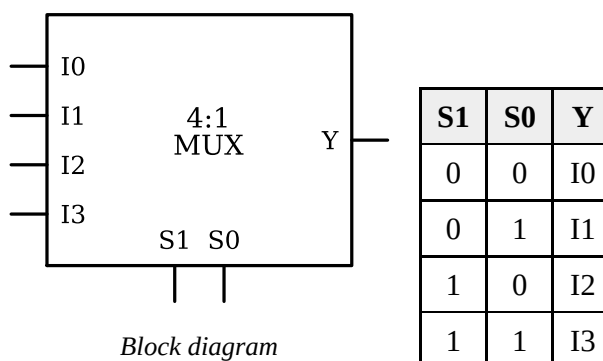
Both columns are identical — simplification verified.

(Statements 1 M, simplification 2 M, verification 1 M)

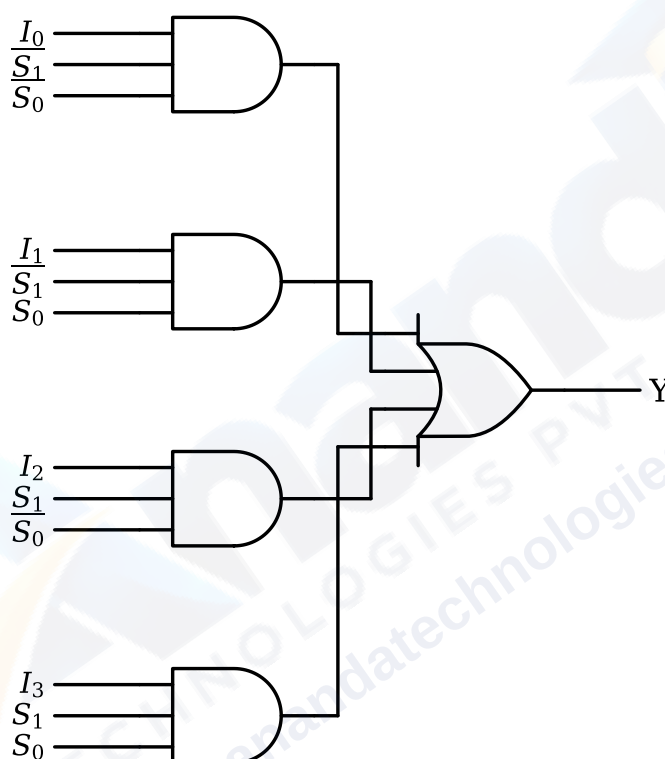
c) 4 : 1 multiplexer.

4 M

A **multiplexer** selects one of many data inputs and connects it to a single output; the selection is done by select lines. A 4:1 MUX has 4 inputs (I₀–I₃), 2 select lines (S₁ S₀) and one output Y.



$$Y = \bar{S}_1\bar{S}_0I_0 + \bar{S}_1S_0I_1 + S_1\bar{S}_0I_2 + S_1S_0I_3$$

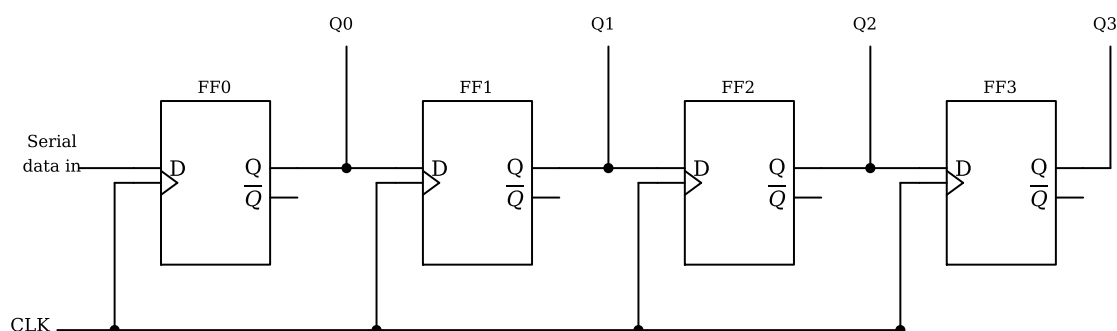


Logic diagram of 4:1 MUX (S₁, S₀ and their complements from NOT gates)

(Block diagram + truth table 2 M, expression 1 M, logic diagram 1 M)

d) 4-bit SIPO shift register.

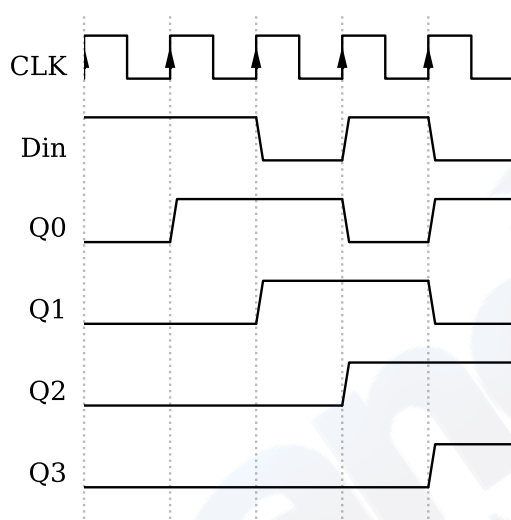
4 M



4-bit SIPO shift register (parallel outputs Q₀–Q₃)

Data is entered serially at D of FF0, one bit per clock pulse, and shifts one position right at each positive clock edge. After 4 clock pulses all 4 bits are available simultaneously at Q0–Q3 (parallel output).

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



Timing diagram for serial data 1101 (first bit = 1); after the 4th clock Q3 Q2 Q1 Q0 = 1101

(Diagram 2 M, table/timing 2 M)

Q.3

a) K-map with don't cares.

4 M

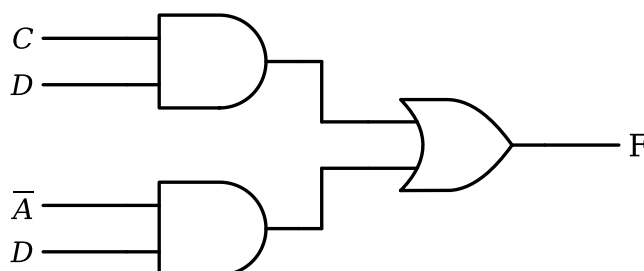
Don't-care cells (X) are used as 1 only when they help to make a bigger group.

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

K-map for F (grouping 1s)

Group	Cells (minterms)	Term
Group 1	3, 7, 11, 15	CD
Group 2	1, 3, 5, 7	$\bar{A}D$

Minimal SOP: $F = CD + \bar{A}D$

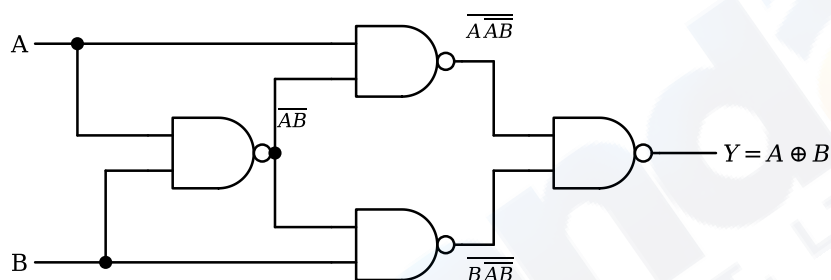


Logic diagram (AND-OR)

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

b) EX-OR using only NAND gates.

4 M



EX-OR using four NAND gates

Gate 1: $\bar{A}\bar{B}$ Gate 2: $A\bar{A}\bar{B}$ Gate 3: $\bar{B}\bar{A}\bar{B}$

Gate 4: $Y = \overline{\overline{A\bar{A}\bar{B}} \cdot \overline{\bar{B}\bar{A}\bar{B}}} = A\bar{A}\bar{B} + \bar{B}\bar{A}\bar{B}$ (De Morgan)

$= A(\bar{A} + \bar{B}) + B(\bar{A} + \bar{B}) = A\bar{B} + \bar{A}B = A \oplus B$

(Diagram 2 M, gate outputs 2 M)

c) Weighted resistor vs R-2R ladder DAC.

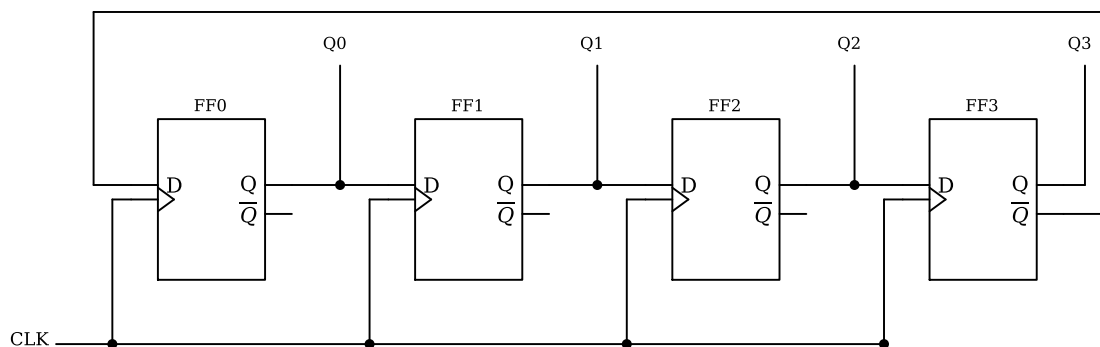
4 M

Weighted resistor DAC	R-2R ladder DAC
Needs n different resistor values ($R, 2R, 4R, \dots 2^{n-1}R$).	Needs only two values, R and $2R$, for any number of bits.
Very wide range of resistors for more bits; accurate high values are hard to make.	Easy to fabricate accurately (suitable for ICs).
Difficult to expand — every extra bit needs a new resistor value.	Easily expanded by adding one more R-2R section.
Fewer resistors ($n + 1$ including R_f).	More resistors (about $2n$).
Less accurate for large n.	More accurate.

(Any four points, 1 M each)

d) 4-bit twisted ring (Johnson) counter.

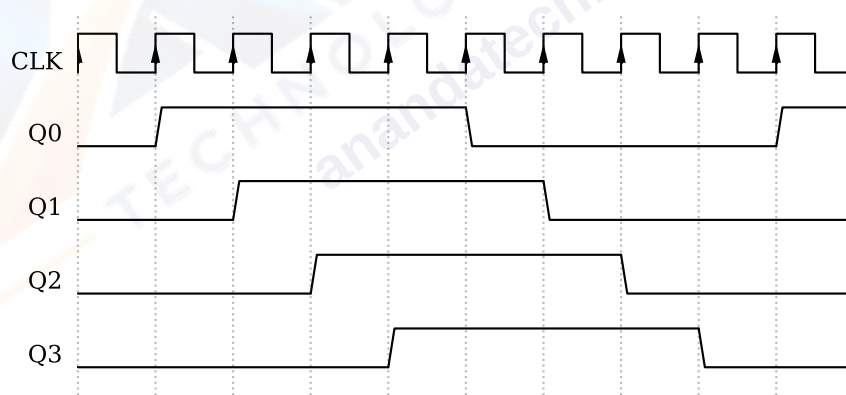
4 M



4-bit Johnson counter: complement output of FF3 fed back to D0

In a twisted ring (Johnson / switch-tail) counter the **complement output** $\overline{Q_3}$ of the last flip-flop is fed back to the input of the first. Starting from 0000, 1s fill in from the left and then 0s fill in. An n-bit Johnson counter has **2n states**; here 8 states (MOD-8).

Clock	Q0	Q1	Q2	Q3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1
8	0	0	0	0



Waveforms of 4-bit Johnson counter

(Diagram 2 M, truth table 1 M, waveforms 1 M)

Q.4

a) Conversions.

4 M

i) $(2F9.A)_{16}$

Hex → Binary: replace each hex digit by its 4-bit binary equivalent.

Hex digit	2	F	9	.	A
Binary	0010	1111	1001	.	1010

$$(2F9.A)_{16} = (1011111001.101)_2$$

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

3-bit group	001	011	111	001	.	101
Octal	1	3	7	1	.	5

$$\therefore (2F9.A)_{16} = (1011111001.101)_2 = (1371.5)_8$$

ii) $(0.6875)_{10} \rightarrow$ binary

Fraction part (0.6875): multiply repeatedly by 2, take the integer parts from top to bottom.

Fraction $\times 2$	Product	Integer part
0.6875×2	1.375	1
0.375×2	0.75	0
0.75×2	1.5	1
0.5×2	1	1

$$\therefore (0.6875)_{10} = (0.1011)_2$$

iii) $(1001\ 0110)_{BCD}$

BCD group	1001	0110
Decimal digit	9	6

$$(1001\ 0110)_{BCD} = (96)_{10}$$

Integer part (96): divide repeatedly by 16, write remainders from bottom to top.

Division	Quotient	Remainder
$96 \div 16$	6	0
$6 \div 16$	0	6

$$\therefore (96)_{10} = (60)_{16}$$

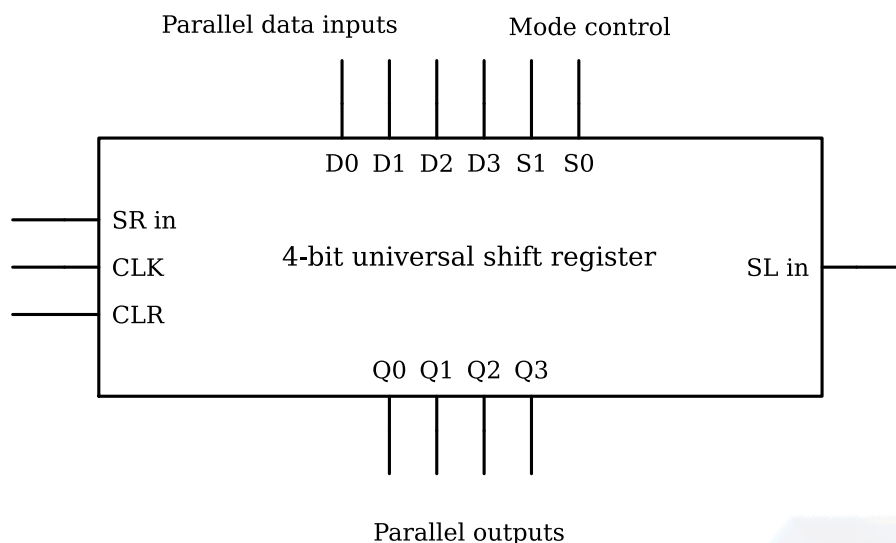
(i) 2 M, ii) 1 M, iii) 1 M)

b) Universal shift register.

4 M

A **universal shift register** can shift data right, shift data left, load data in parallel and hold data. So it can work in all modes: SISO, SIPO, PISO, PIPO and bidirectional. The mode is selected by two

mode-control inputs S1 S0, which drive a 4:1 MUX at the D input of every flip-flop.



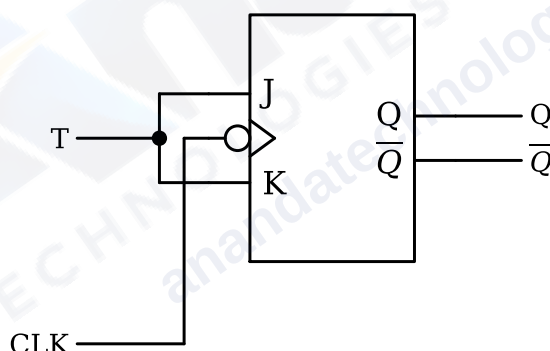
Block diagram (SR in = serial input for shift right, SL in = serial input for shift left)

S1	S0	Mode	Operation at clock edge
0	0	Hold	No change (each FF gets its own output)
0	1	Shift right	Serial-in (right) → Q0 → Q1 → Q2 → Q3
1	0	Shift left	Serial-in (left) → Q3 → Q2 → Q1 → Q0
1	1	Parallel load	D0–D3 loaded into Q0–Q3

(Block diagram 2 M, explanation + mode table 2 M)

c) T flip-flop.

4 M



T flip-flop from J-K flip-flop ($J = K = T$)

Joining J and K of a J-K flip-flop gives a **T (toggle) flip-flop**. When $T = 0$ ($J = K = 0$) the output does not change; when $T = 1$ ($J = K = 1$) the output toggles at every active clock edge.

CLK	T	Q_{n+1}	Q_n	Q_{n+1}	T
↓	0	Q_n (no change)	0	0	0
↓	1	$\overline{Q_n}$ (toggle)	0	1	1
			1	0	1
			1	1	0

Application: frequency divider by 2 ($T = 1$: output frequency = $f_{CLK}/2$); used in ripple counters.

(Diagram 1 M, truth table 1 M, excitation table 1 M, application 1 M)

d) EPROM vs Flash memory.

4 M

EPROM	Flash memory
Erased by ultraviolet light through a quartz window.	Erased electrically.
Whole chip is erased at once.	Erased block by block (sector-wise).
Chip must be removed from the circuit; erasing takes 15–20 minutes.	Erased and reprogrammed in the circuit within milliseconds.
Lower density, older technology.	Very high density, low cost per bit.
Used in older microcontroller program storage / development.	Used in pen drives, memory cards, SSDs, BIOS chips, mobile phones.

(Any four points, 1 M each)

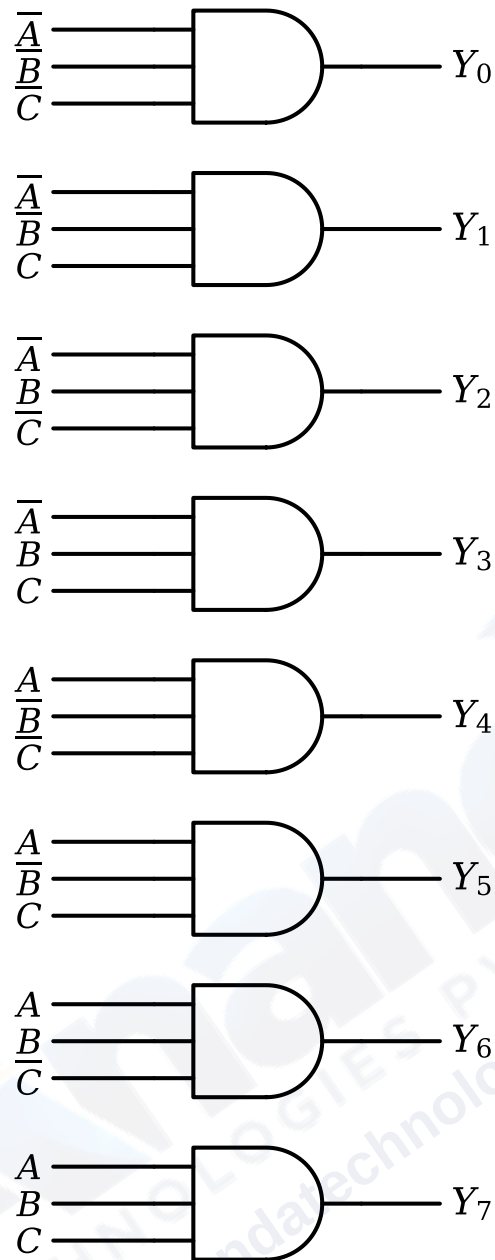
e) 3 : 8 line decoder.

4 M

A 3:8 decoder has 3 inputs (A B C, A = MSB) and 8 outputs Y0–Y7; for each input combination exactly one output (the minterm) is 1.

A	B	C	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

$$Y_0 = \overline{A}\overline{B}\overline{C}, Y_1 = \overline{A}\overline{B}C, Y_2 = \overline{A}B\overline{C}, Y_3 = \overline{A}BC, Y_4 = A\overline{B}\overline{C}, Y_5 = A\overline{B}C, Y_6 = AB\overline{C}, Y_7 = ABC$$



Logic diagram of 3:8 decoder (eight 3-input AND gates; complements from NOT gates)

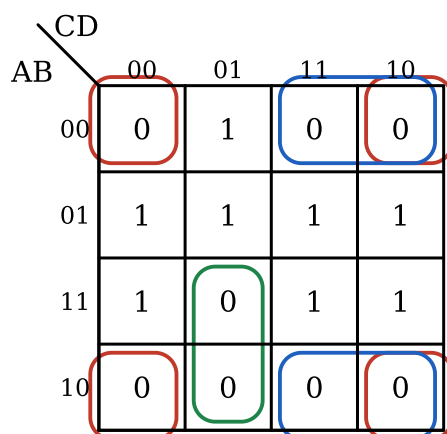
(Truth table 2 M, logic diagram 2 M)

Q.5

a) Minimal POS and NOR realization.

6 M

Put 0 in the cells of the given maxterms, 1 elsewhere, and group the 0s.

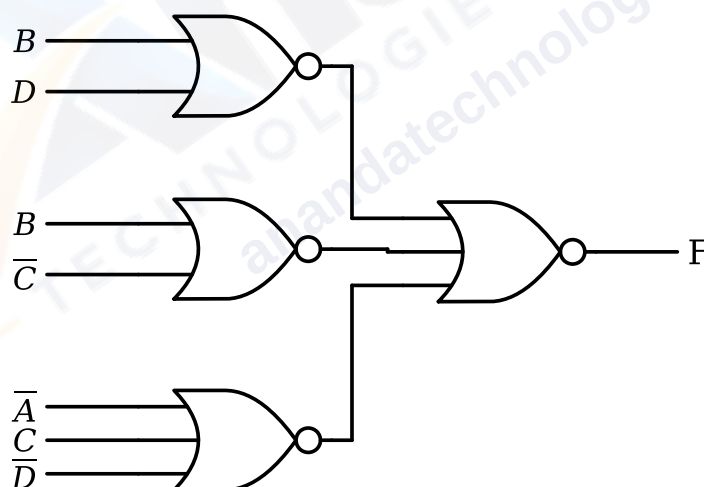


K-map for F (grouping 0s)

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	0, 2, 8, 10	$(B + D)$
Group 2	2, 3, 10, 11	$(B + \bar{C})$
Group 3	9, 13	$(\bar{A} + C + \bar{D})$

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



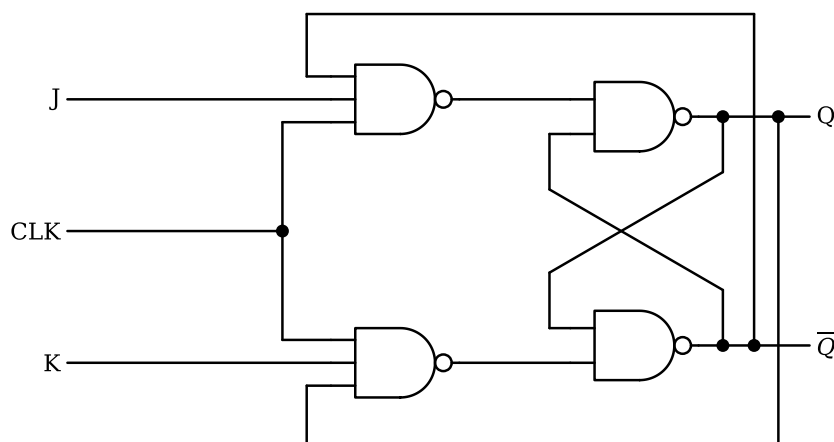
Logic diagram (NOR-NOR)

NOR realization: a POS expression maps directly to two-level NOR-NOR logic: each sum term is produced by a NOR gate (giving its complement) and the final NOR gate gives the product of the sum terms (by De Morgan: $\overline{\overline{X} + \overline{Y}} = X \cdot Y$). Single complemented inputs come from NOR gates used as inverters.

(K-map 2 M, minimal POS 2 M, NOR circuit 2 M)

b) J-K flip-flop; race-around condition.

6 M



Clocked J-K flip-flop using NAND gates

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

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

The feedback from Q and $\bar{Q}$ to the input gates removes the invalid state of the S-R flip-flop: for J = K = 1 the output toggles.

Race-around condition: for J = K = 1, while CLK stays HIGH the output toggles; after a delay Δt (propagation delay) the new output is fed back and it toggles again. If the clock pulse width $t_p > \Delta t$, the output keeps toggling ($0 \rightarrow 1 \rightarrow 0 \rightarrow \dots$) during one pulse, and its final value is unpredictable.



Race-around: output toggles many times during one long clock pulse ($J = K = 1$)

Methods to avoid: i) keep $t_p < \Delta t < T$ (not practical); ii) use **edge-triggered J-K flip-flop** (output changes only at the clock edge); iii) use **master-slave J-K flip-flop** — the master works when CLK = 1 and the slave copies it when CLK = 0, so the output changes only once per clock.

(Diagram 2 M, tables 2 M, race-around and remedies 2 M)

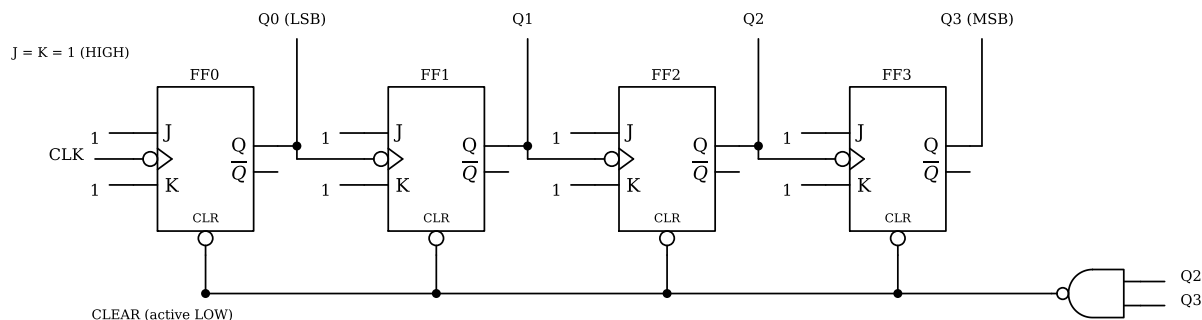
c) MOD-12 asynchronous counter.

6 M

MOD-12 needs 4 flip-flops ($2^3 = 8 < 12 \leq 2^4 = 16$). The counter counts 0000 to 1011 (0 to 11).

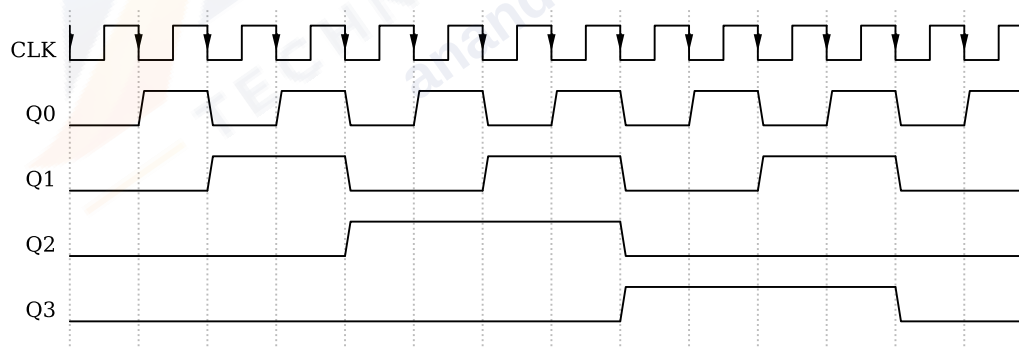
When it reaches 1100 (decimal 12), $Q_3 = Q_2 = 1$ for the first time, so a 2-input NAND gate fed by

Q3 and Q2 clears all flip-flops to 0000. J = K = 1 for all flip-flops; each flip-flop (negative-edge triggered) is clocked by the Q output of the previous one.



MOD-12 ripple counter using J-K flip-flops (clear when Q3 Q2 = 11)

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	1	0	10
11	1	0	1	1	11
12	1 → 0	1 → 0	0	0	12 → 0 (reset)



Timing diagram of MOD-12 ripple counter

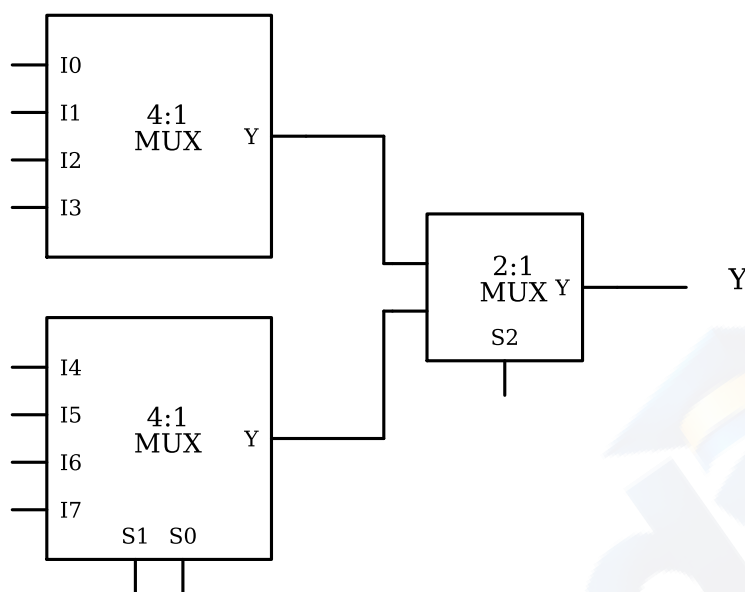
(Design 1 M, diagram 2 M, truth table 1 M, timing diagram 2 M)

Q.6

a) 8 : 1 MUX using 4 : 1 and 2 : 1 MUX; MUX implementation.

6 M

i) 8 inputs need 3 select lines $S_2 S_1 S_0$. Two 4:1 MUXes (common $S_1 S_0$) select one input from I_0 – I_3 and one from I_4 – I_7 ; a 2:1 MUX controlled by S_2 selects between them.

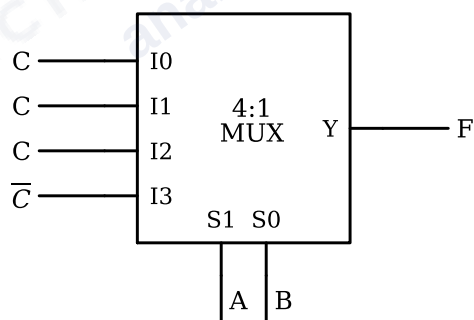


(S_1, S_0 common to all)

8:1 MUX using two 4:1 MUX and one 2:1 MUX

ii) Use A, B as select lines and express each data input in terms of C:

A B (select)	Minterms	F for C = 0	F for C = 1	Data input
0 0 $\rightarrow I_0$	m0, m1	0	1	C
0 1 $\rightarrow I_1$	m2, m3	0	1	C
1 0 $\rightarrow I_2$	m4, m5	0	1	C
1 1 $\rightarrow I_3$	m6, m7	1	0	$\overline{C}$



$F = \Sigma m(1, 3, 5, 6)$ using 4:1 MUX ($S_1 = A, S_0 = B$)

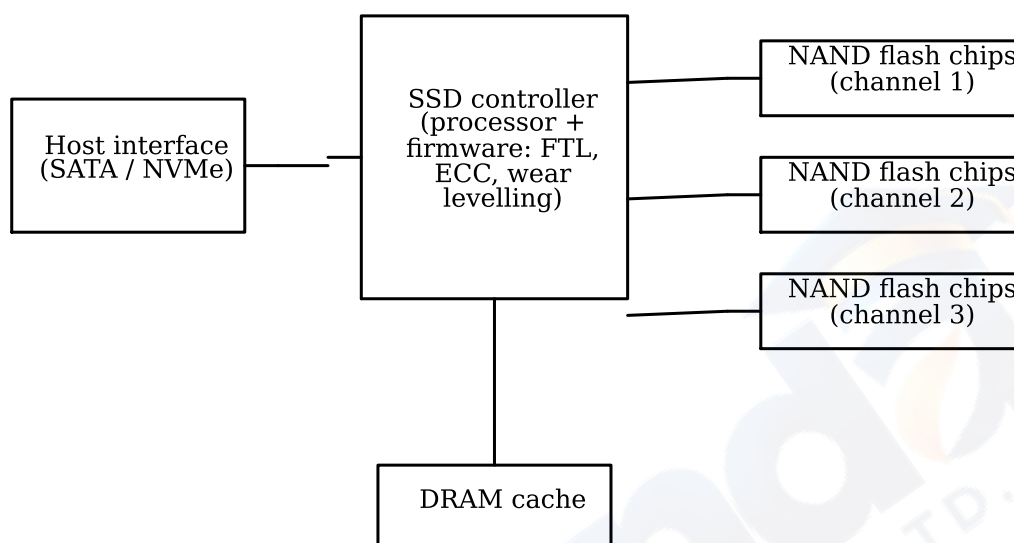
(Checked for all 8 input combinations.)

(8:1 design 3 M, implementation table 2 M, diagram 1 M)

b) Solid State Drive (SSD).

6 M

An **SSD** is a non-volatile storage device that stores data electronically in **NAND flash memory** chips; it has no moving parts (unlike a hard disk). Each flash cell is a floating-gate (or charge-trap) MOS transistor: during writing, electrons are pushed into the floating gate by a high voltage (programming), which changes the threshold voltage of the transistor; the presence or absence of this charge represents the stored bit (SLC: 1 bit, MLC/TLC/QLC: 2/3/4 bits per cell using several charge levels). Data is read by sensing the threshold voltage. Cells are written in **pages** and erased in larger **blocks**.



Block diagram of a Solid State Drive

Main parts:

1. **NAND flash memory chips** — store the data permanently; arranged in several channels so that many chips work in parallel (high speed).
2. **SSD controller** — an embedded processor that manages reading/writing; its firmware does the *Flash Translation Layer* (logical-to-physical address mapping), *wear levelling* (spreads writes evenly), *error correction (ECC)*, garbage collection and bad-block management.
3. **DRAM cache (buffer)** — volatile memory that holds the mapping table and buffers data to increase speed (some low-cost SSDs are DRAM-less).
4. **Host interface** — SATA, PCIe/NVMe or USB connector that connects the SSD to the computer.

Features: very fast access and boot time; no moving parts, so shock-resistant and silent; low power consumption and heat; light and compact; limited number of write/erase cycles; higher cost per GB than HDD.

(Data storage 2 M, block diagram 1 M, parts 2 M, features 1 M)

c) Full subtractor using K-map.

6 M

A full subtractor subtracts B and the borrow-in B_{in} (written C) from A: $A - B - B_{in}$, giving

Difference D and Borrow-out B_{out}

A	B	B_{in}	D	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

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

K-map for D

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

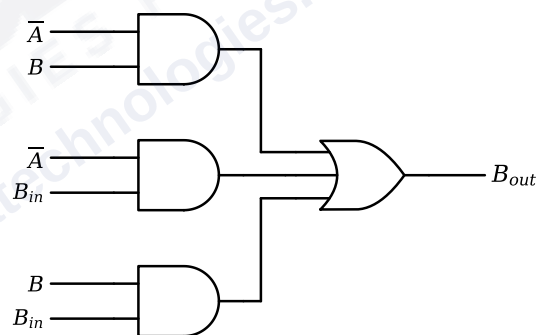
K-map for B_{out}

$$D = \overline{A}BC + \overline{A}\overline{B}C + A\overline{B}C + ABC = A \oplus B \oplus B_{in}$$

$$B_{out} = BC + \overline{A}C + \overline{A}B = \overline{A}B + \overline{A}B_{in} + B \cdot B_{in}$$



Difference



Borrow

(Truth table 2 M, K-maps 2 M, logic diagram 2 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