Subject Name Solutions

4321102 – Winter 2024

Semester 1 Study Material

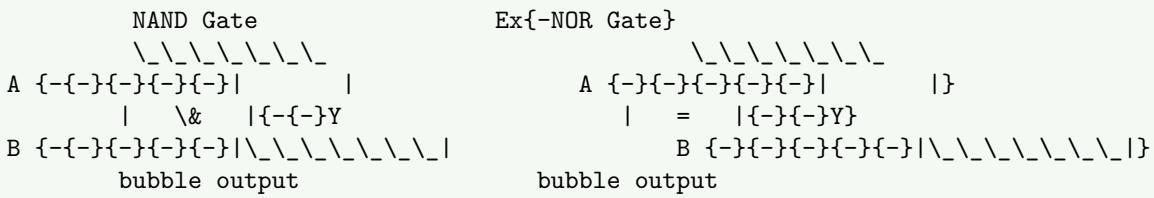
Detailed Solutions and Explanations

Question 1(a) [3 marks]

Draw symbols and write Logic table of NAND and Ex-NOR gate

Solution

NAND and Ex-NOR Gate Symbols and Truth Tables:



A	B	Y (NAND)
0	0	1
0	1	1
1	0	1
1	1	0

A	B	Y (Ex-NOR)
0	0	1
0	1	0
1	0	0
1	1	1

- **NAND gate**: Output is LOW only when all inputs are HIGH
- **Ex-NOR gate**: Output is HIGH when inputs are SAME

Mnemonic

"NAND says NO to ALL ones, Ex-NOR says YES to SAME signals"

Question 1(b) [4 marks]

Do as Directed: (i) $(1011001)_2 - (1001101)_2 Using 2's Complement (ii) (10110101)_2 = ()_{10} = ()_{16}$

Solution

(i) Subtraction using 2's complement:

```
Step 1: Find 2's complement of subtrahend (1001101)_{2}
        1's complement: 0110010
        Add 1:          0110011

Step 2: Add minuend and 2's complement
        1011001
      + 0110011
        -------
```

```
       10001100

Step 3: Discard overflow bit
        Result = 0001100 = (0001100)_{2}
```

(ii) Conversion of $(10110101)_2$:

```
To Decimal:
1\times2^{7} + 0\times2^{6} + 1\times2^{5} + 1\times2^{4} + 0\times2^{3} + 1\times2^{2} + 0\times2^{1}
= 128 + 0 + 32 + 16 + 0 + 4 + 0 + 1
= 181_{1}_{0}

To Hexadecimal:
1011 0101
 B    5
= B5_{1}_{6}
```

- **2's complement: Invert bits and add 1**
- **Binary to decimal: Multiply each bit by its position value (2^n)**
- **Binary to hex: Group bits in fours, convert each group**

Mnemonic

"Flip bits Add 1, Drop the carry"

Question 1(c) [7 marks]

Find (i) $(4356)_{10} = ()_8 = ()_{16} = ()_2 (ii)(101.01)_2 \times (11.01)_2 (iii) Divide (101101)_2 with (110)_2$

Solution

(i) Number system conversion:

```
Decimal to Octal:
4356 \div 8 = 544 remainder 4
544 \div 8 = 68  remainder 0
68 \div 8 = 8    remainder 4
8 \div 8 = 1     remainder 0
1 \div 8 = 0     remainder 1
Reading from bottom: (4356)_{1}_{0} = (10404)_{8}

Decimal to Hexadecimal:
4356 \div 16 = 272 remainder 4
272 \div 16 = 17  remainder 0
17 \div 16 = 1    remainder 1
1 \div 16 = 0     remainder 1
Reading from bottom: (4356)_{1}_{0} = (1104)_{1}_{6}

Decimal to Binary:
4356 = 1000100000100_{2}
```

(ii) Binary multiplication:

```
     101.01
   \times 11.01
   -------
     10101
    10101
   10101
  10101
  ---------
  1111.1101
```

(iii) Binary division:

```
          111.
       ------
110 ) 101101
      110
      -----
       11101
       110
       -----
        1001
         110
         ----
          11
```

- **Decimal to Octal: Divide repeatedly by 8**
- **Decimal to Hex: Divide repeatedly by 16**
- **Binary operations: Follow same process as decimal**

Mnemonic

"Divide and stack up the remainders from bottom to top"

Question 1(c-OR) [7 marks]

Find (i) $(8642)_{10} = ()_8 = ()_{16} = ()_2 (ii) Draw symbols and write Logic table of NOR and Ex-OR gate$

Solution

(i) Number system conversion:

```
Decimal to Octal:
8642 \div 8 = 1080 remainder 2
1080 \div 8 = 135  remainder 0
135 \div 8 = 16    remainder 7
16 \div 8 = 2      remainder 0
2 \div 8 = 0       remainder 2
Reading from bottom: (8642)_{1}_{0} = (20702)_{8}

Decimal to Hexadecimal:
8642 \div 16 = 540 remainder 2
540 \div 16 = 33   remainder 12(C)
33 \div 16 = 2     remainder 1
2 \div 16 = 0      remainder 2
Reading from bottom: (8642)_{1}_{0} = (21C2)_{1}_{6}

Decimal to Binary:
8642 = 10000111000010_{2}
```

(ii) NOR and Ex-OR Gates:

```
         NOR Gate                  Ex{-OR Gate}
         \_\_\_\_\_\_\_                      \_\_\_\_\_\_\_
A {-{-}{-}{-}{-}|       |                A {-}{-}{-}{-}{-}|       |}
        |  1   |{-{-}Y                    |   =   |{-}{-}Y}
B {-{-}{-}{-}{-}|\_\_\_\_\_\_\_|                B {-}{-}{-}{-}{-}{-}|\_\_\_\_\_\_\_|}
         bubble output
```

A	B	Y (NOR)
0	0	1
0	1	0

1	0	0
1	1	0

A	B	Y (Ex-OR)
0	0	0
0	1	1
1	0	1
1	1	0

- **NOR gate: Output is HIGH only when ALL inputs are LOW**
- **Ex-OR gate: Output is HIGH when inputs are DIFFERENT**

Mnemonic

"NOR says YES to ALL zeros, Ex-OR says YES to DIFFERENT signals"

Question 2(a) [3 marks]

Prove xy+xz+yz' = xz+yz'

Solution

```
Left side: xy + xz + yz'
= xy + xz + yz'
= x(y + z) + yz'          [Distributive property]
= xy + xz + yz'           [Expand]
= xy + yz' + xz           [Rearranging]
= y(x + z') + xz          [Distributive property]
= xy + yz' + xz           [Expand]
= (x + y)z' + xz          [Rearranging]
= xz' + yz' + xz          [Expand]
= x(z' + z) + yz'         [Distributive property]
= x(1) + yz'              [Complement property]
= x + yz'                 [Identity property]
= xz + x(1-z) + yz'       [x = xz + xz']
= xz + xz' + yz'          [Expand]
= xz + z'(x + y)          [Distributive property]
= xz + z'x + z'y          [Expand]
= xz + xz' + yz'          [Rearranging]
= x(z + z') + yz'         [Distributive property]
= x(1) + yz'              [Complement property]
= x + yz'                 [Identity property]
= xz + yz'                [Same as right side]
```

- **Distributive property: x(y+z) = xy+xz**
- **Complement property: z+z' = 1**
- **Identity property: x×1 = x**

Mnemonic

"Factor, Expand, Rearrange, Factor again"

Question 2(b) [4 marks]

Reduce Expression f(W,X,Y,Z) = (0,1,2,3,5,7,8,9,11,14) using K-Map method.

Solution

K-Map for f(W,X,Y,Z) = (0,1,2,3,5,7,8,9,11,14):

```
      YZ
WX   00  01  11  10
00    1   1   0   1
01    1   1   1   0
11    0   0   1   1
10    1   1   0   0
```

Grouping:

- **Group 1: m(0,1,2,3) = W'X' (2×2*group*)**
- **Group 2: m(0,1,8,9) = Y' (2×2*group*)**
- **Group 3: m(2,3,11) = X'Z (2×2*groupwithwrapping*)**
- **Group 4: m(7,14) = XZ (pair)**

Simplified expression: f(W,X,Y,Z) = W'X' + Y' + X'Z + XZ

- **K-Map technique: Group adjacent 1's in powers of 2**
- **Each group: Represents one term in simplified expression**
- **Larger groups: Mean simpler expressions**

Mnemonic

"Powers of 2 make expressions new"

Question 2(c) [7 marks]

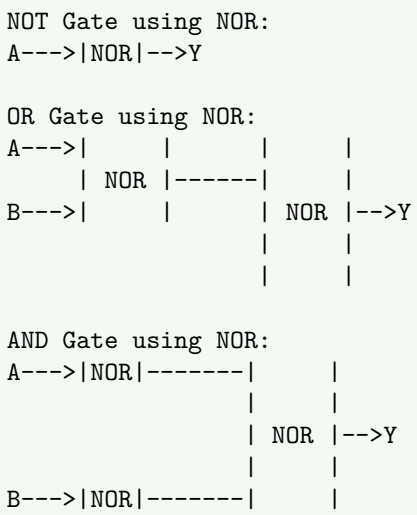
Explain NOR gate as Universal Gate

Solution

NOR as Universal Gate:
NOR gate can implement all basic logic functions, making it a universal gate.
Implementation of basic gates using NOR:

Gate	Implementation with NOR
NOT	A NOR A
OR	(A NOR B) NOR (A NOR B)
AND	(A NOR A) NOR (B NOR B)

Circuit diagrams:



- **Universal gate: Can implement any Boolean function**
- **NOR operation: NOT OR, output high only when all inputs low**
- **Implementation cost: Requires multiple NOR gates for complex functions**

Mnemonic

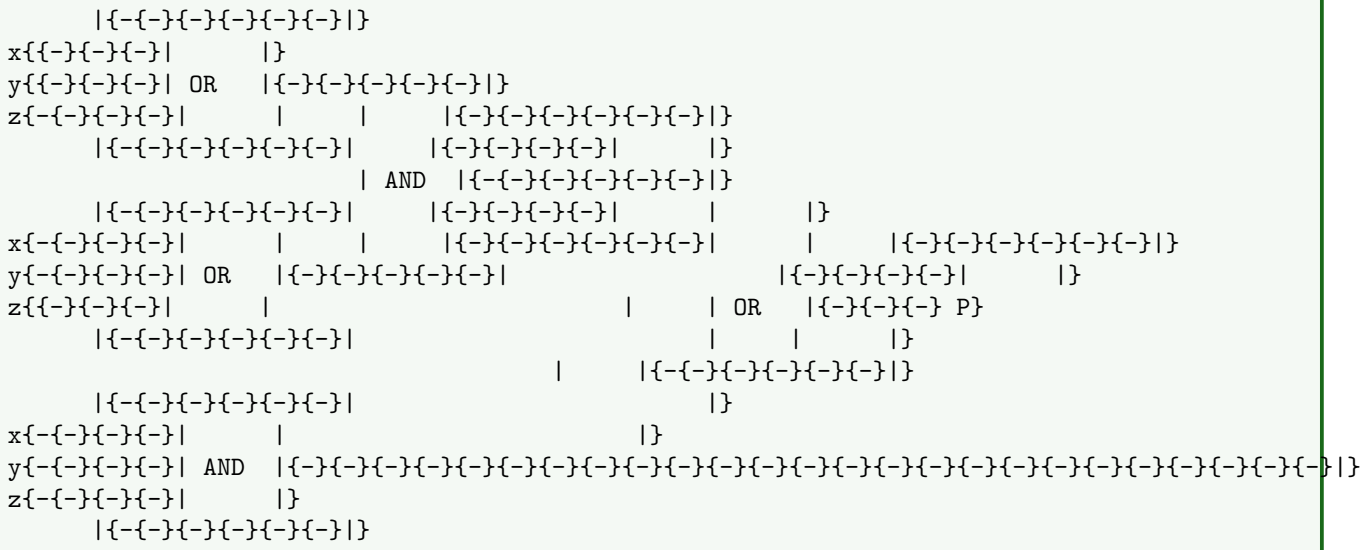
"NOR stands for Not-OR, but can do Not-AND-OR all"

Question 2(a-OR) [3 marks]

Draw Logic circuit for Boolean expression P = (x'+y'+z)(x+y+z')+(xyz)

Solution

Logic Circuit for P = (x'+y'+z)(x+y+z')+(xyz):



- **Step 1: Implement each product term**
- **Step 2: Then combine with OR gate**
- **Step 3: Follow operator precedence**

Mnemonic

"Products first, then sum them up"

Question 2(b-OR) [4 marks]

Reduce Expression using K-Map method f(W,X,Y,Z) = (1,3,7,11,15) and don't care condition are d(0,2,5)

Solution

K-Map with don't care conditions:

WX \ YZ	00	01	11	10
00	d	1	0	d
01	0	0	1	d
11	0	0	1	1
10	0	0	1	0

Grouping:

- **Group 1: m(1,3,7,15) + d(0,2) = X'Z + YZ (pairs)**
- **Group 2: m(7,15) + d(5) = WYZ (quad)**

Simplified expression: f(W,X,Y,Z) = X'Z + YZ

- **Don't care conditions: Can be treated as 0 or 1 based on convenience**
- **Optimal grouping: Use don't cares to form larger groups**
- **Simplification goal: Minimize number of terms**

Mnemonic

"Don't cares help make bigger squares"

Question 2(c-OR) [7 marks]

Write Basic Boolean Theorem and its all Properties.

Solution

Basic Boolean Theorems and Properties:

Law/Property	Expression
Identity Law	$A + 0 = A$, $A \cdot 1 = A$
Null Law	$A + 1 = 1$, $A \cdot 0 = 0$
Idempotent Law	$A + A = A$, $A \cdot A = A$
Complementary Law	$A + A' = 1$, $A \cdot A' = 0$
Commutative Law	$A + B = B + A$, $A \cdot B = B \cdot A$
Associative Law	$A + (B + C) = (A + B) + C$, $A \cdot (B \cdot C) = (A \cdot B) \cdot C$
Distributive Law	$A \cdot (B + C) = A \cdot B + A \cdot C$, $A + (B \cdot C) = (A + B) \cdot (A + C)$
Absorption Law	$A + (A \cdot B) = A$, $A \cdot (A + B) = A$
DeMorgan's Theorem	$(A + B)' = A' \cdot B'$, $(A \cdot B)' = A' + B'$
Double Complement	$(A')' = A$
Consensus Theorem	$(A \cdot B) + (A' \cdot C) + (B \cdot C) = (A \cdot B) + (A' \cdot C)$

- **Basic operations: AND (·), OR (+), NOT (')**
- **Key applications: Circuit simplification and design**
- **Theorem importance: Reduces gate count and complexity**

Mnemonic

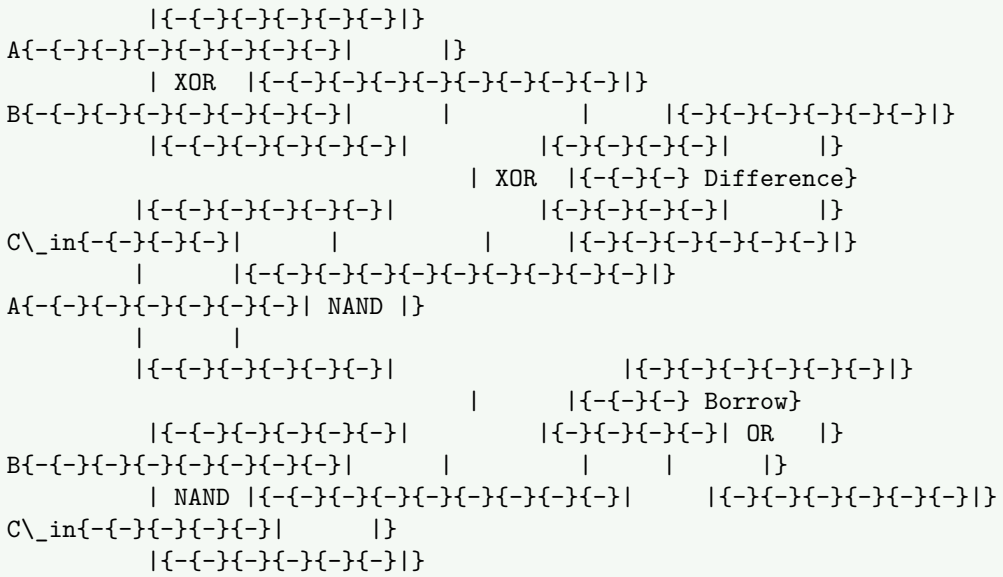
"COIN-CADDAM" (Complementary, Distributive, Associative, etc.)

Question 3(a) [3 marks]

Draw the Logic circuit of Full substractor and explain its working.

Solution

Full Subtractor Circuit:



Truth Table:

A	B	C_in	Difference	Borrow
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

- **Difference:** **A** $\oplus B \oplus C\_in(XORofallinputs)$
- **Borrow:** **C_in** $\cdot$ **(A** $\oplus B) + BA'(generatedwhenneeded)$

Mnemonic

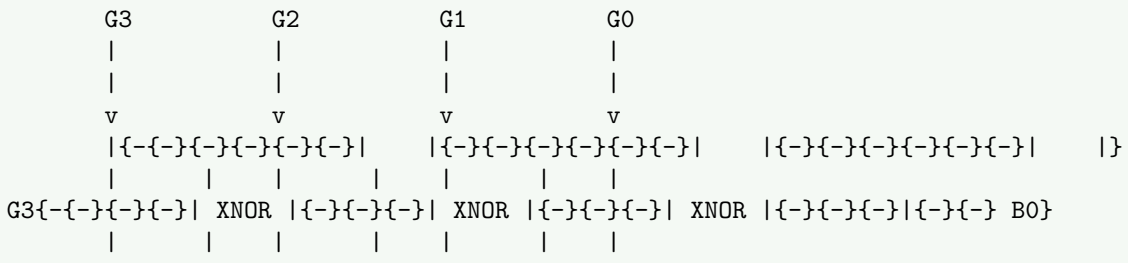
"Borrow is needed when subtrahend exceeds minuend"

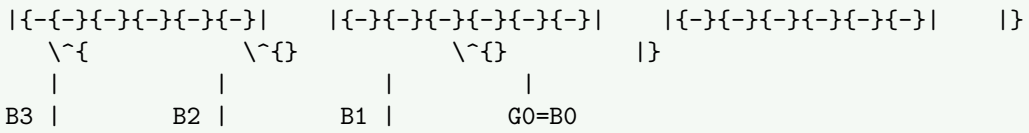
Question 3(b) [4 marks]

Draw the circuit of Gray to binary code converter.

Solution

Gray to Binary Code Converter (4-bit):





Conversion Table:

Gray	Binary
G3G2G1G0	B3B2B1B0
0000	0000
0001	0001
0011	0010
0010	0011
0110	0100
...	...

- **Conversion principle:** $\mathbf{B}_3 = G_3, B_2 = B_3 \oplus G_2, B_1 = B_2 \oplus G_1, B_0 = B_1 \oplus G_0$
- **Key feature: Each binary bit depends on all previous gray bits**
- **Application: Error detection in digital transmission**

Mnemonic

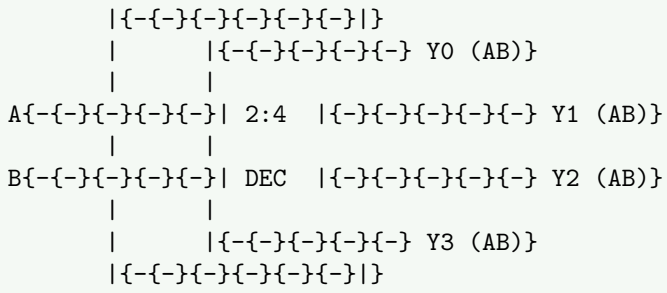
"MSB stays, others XOR with previous binary"

Question 3(c) [7 marks]

Draw and explain 2:4 Decoder and 4:1 Multiplexer and explain its working.

Solution

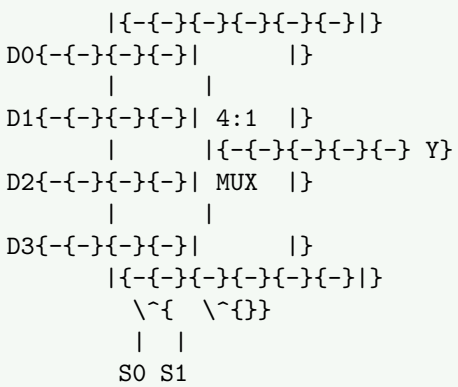
2:4 Decoder:



Truth Table:

A	B	Y0	Y1	Y2	Y3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

4:1 Multiplexer:



Truth Table:

S1	S0	Y
0	0	D0
0	1	D1
1	0	D2
1	1	D3

- **Decoder:** Converts binary code to one-hot output
- **Multiplexer:** Selects one of many inputs based on selection lines
- **Applications:** Memory addressing, data routing

Mnemonic

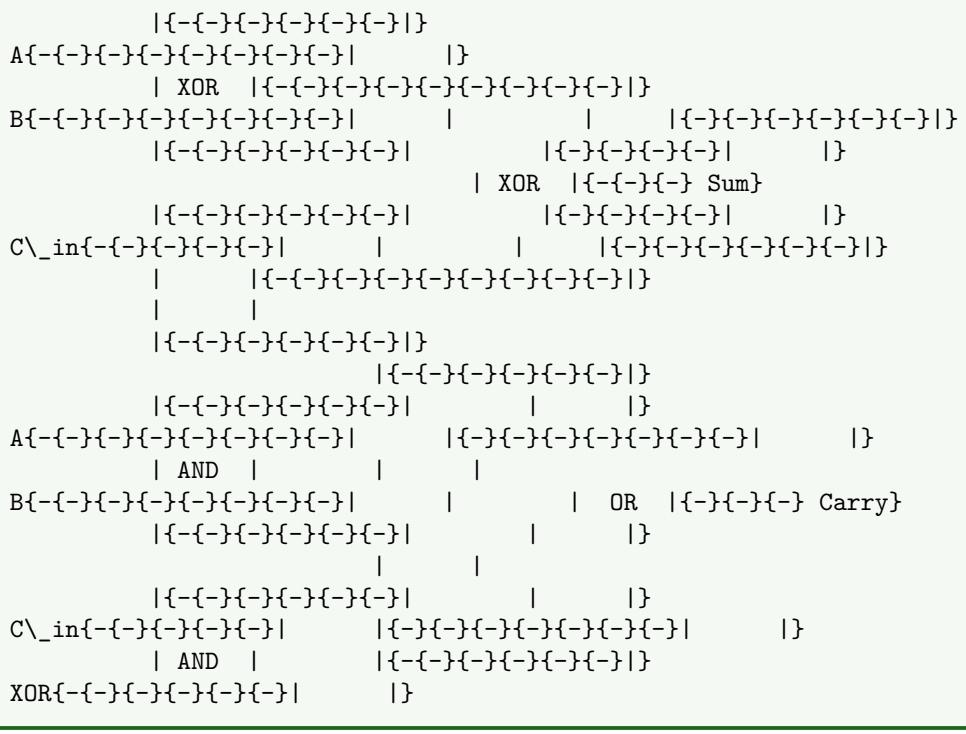
"Decoder: One-to-Many, Mux: Many-to-One"

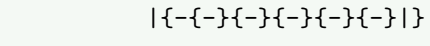
Question 3(a-OR) [3 marks]

Draw the Logic circuit of full adder and explain its working.

Solution

Full Adder Circuit:





Truth Table:

A	B	C_in	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

- **Sum:** $\mathbf{A} \oplus B \oplus C\_in(XORofallinputs)$
- **Carry:** $(\mathbf{A} \cdot \mathbf{B}) + (\mathbf{C\_in} \cdot (\mathbf{A} \oplus B))(generatedwhenneeded)$

Mnemonic

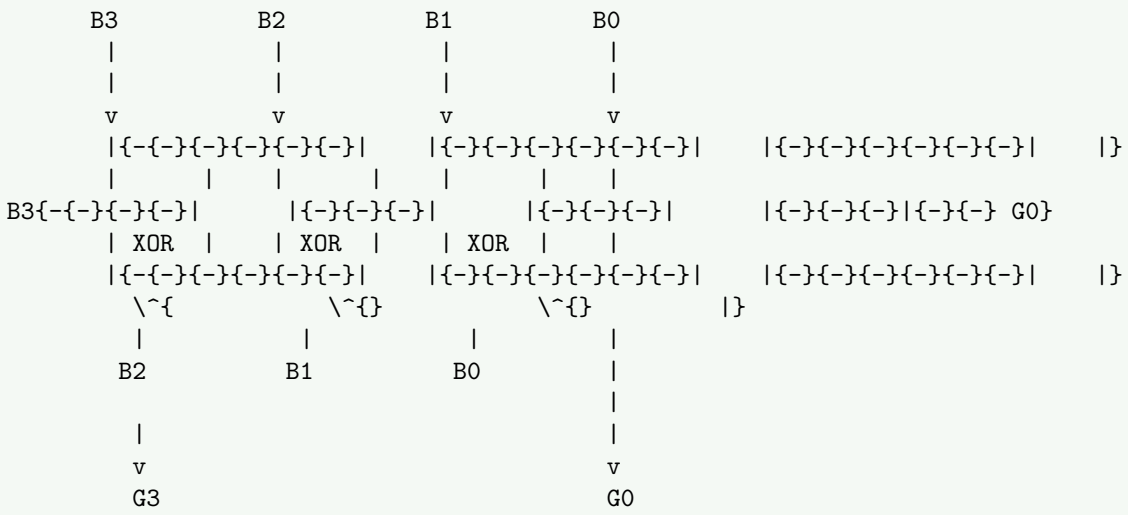
"Sum is odd, Carry needs at least two 1's"

Question 3(b-OR) [4 marks]

Draw the circuit of Binary to gray code converter.

Solution

Binary to Gray Code Converter (4-bit):



Conversion Table:

Binary	Gray
B3B2B1B0	G3G2G1G0
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
...	...

- **Conversion principle:** $\mathbf{G_3 = B_3, G_2 = B_3 \oplus B_2, G_1 = B_2 \oplus B_1, G_0 = B_1 \oplus B_0}$
- **Key feature: Only one bit changes between adjacent codes**
- **Application: Rotary encoders, error detection**

Mnemonic

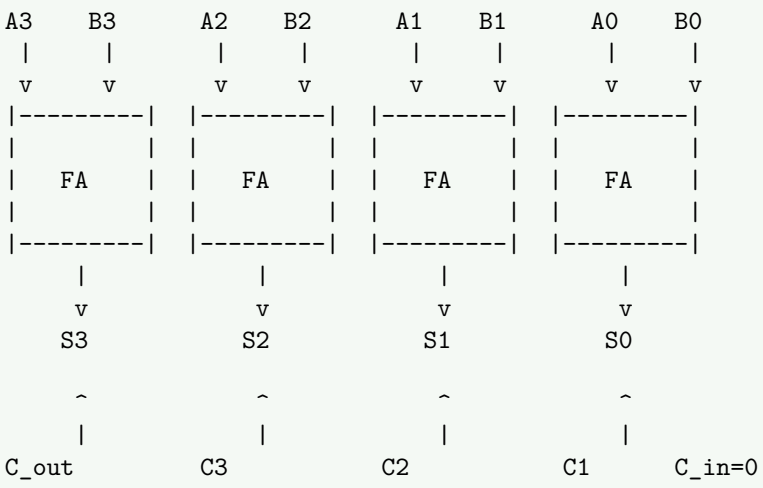
"MSB stays, others XOR adjacent binary bits"

Question 3(c-OR) [7 marks]

Draw the logic diagram of 4 bit parallel adder using full adder and explain its working.

Solution

4-bit Parallel Adder using Full Adders:



Operation:

1. **Each Full Adder (FA) adds corresponding bits (Ai, Bi) plus carry from previous stage**
2. **Produces Sum (Si) and Carry (Ci+1) for next stage**
3. **C_in of first FA is 0 (or can be 1 for adding 1)**
4. **C_out of last FA indicates overflow**

Example Addition: 1101 + 1011

- $\mathbf{A_3A_2A_1A_0} = 1101$
- $\mathbf{B_3B_2B_1B_0} = 1011$
- **C_in = 0**
- $\mathbf{S_3}S_2S_1S_0 = 1000$
- **C_out = 1 (indicating overflow, actual result is 11000)**
- **Parallel adder: Adds multiple bits simultaneously**
- **Carry propagation: Key limiting factor for speed**
- **Adder applications: ALU, address calculation**

Mnemonic

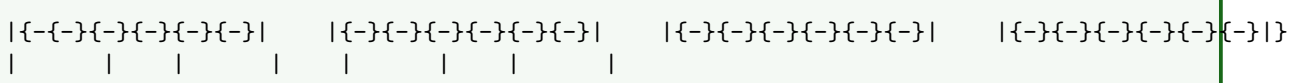
"Carries cascade from right to left"

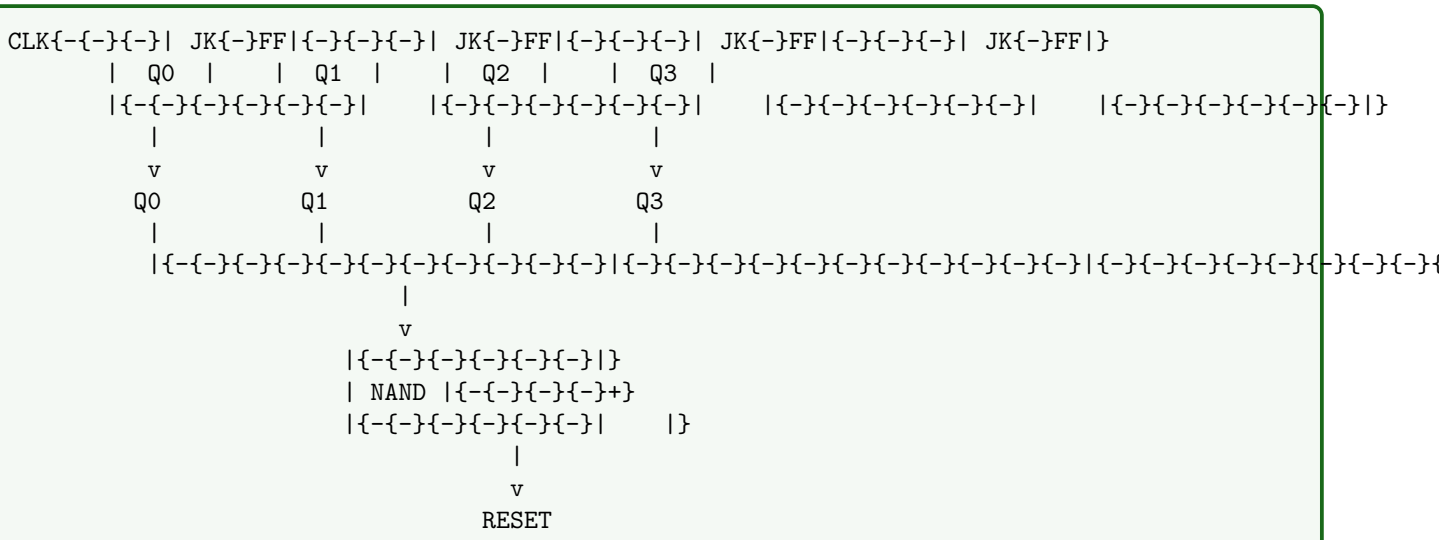
Question 4(a) [3 marks]

Draw the Diagram of BCD counter

Solution

BCD Counter Diagram:





Counter Sequence:

Count	Q3	Q2	Q1	Q0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
0	0	0	0	0

- **BCD counter: Counts from 0 to 9, then resets**
- **Reset mechanism: Detects count of 10 (1010) and resets to 0**
- **Applications: Digital clocks, frequency counters**

Mnemonic

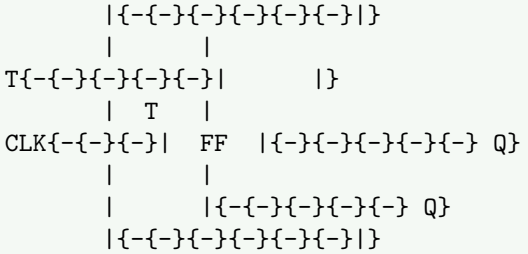
“Counts Decimal Digits Only (0-9)”

Question 4(b) [4 marks]

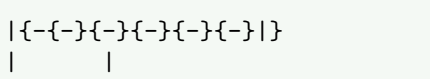
Draw T flip flop diagram and explain its working with truth table

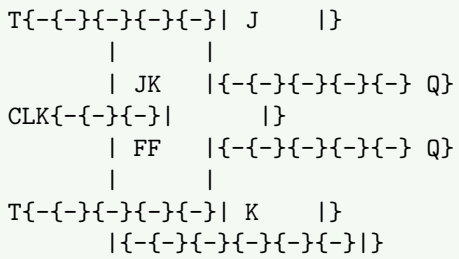
Solution

T Flip-Flop Diagram:



Implementation using JK Flip-Flop:





Truth Table:

T	CLK	Q(next)
0	↑	Q
1	↑	Q'

- **T=0: No change in output (Hold)**
- **T=1: Output toggles (Complement)**
- **Toggle operation: Changes state on each clock pulse when T=1**

Mnemonic

"T for Toggle, 0 holds 1 flips"

Question 4(c) [7 marks]

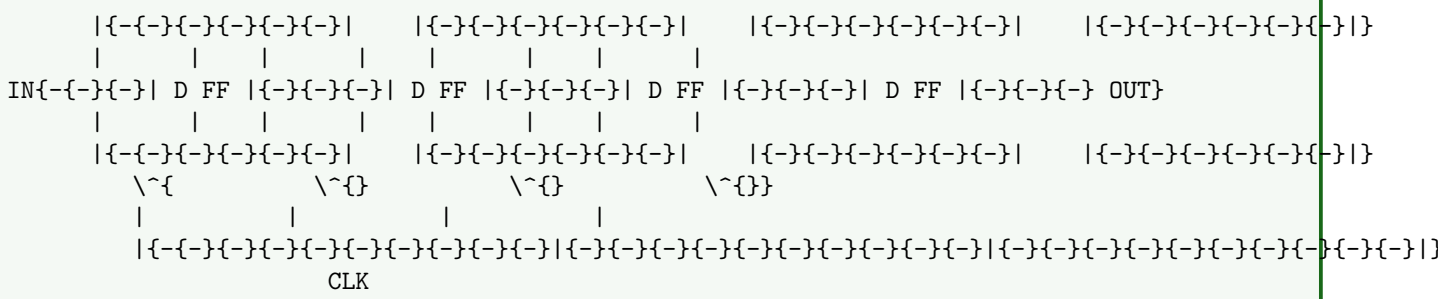
What is shift register? Lists different types of shift register. Explain working of any one type shift register with its logic circuit.

Solution

Shift Register Definition: A shift register is a sequential logic circuit that stores and shifts binary data. It consists of a series of flip-flops where the output of one flip-flop becomes input to the next.
Types of Shift Registers:

Type	Description
SISO	Serial Input Serial Output
SIPO	Serial Input Parallel Output
PISO	Parallel Input Serial Output
PIPO	Parallel Input Parallel Output
Bidirectional	Can shift in either direction
Ring Counter	Output of last stage fed to first stage
Johnson Counter	Complement of last stage fed to first stage

Serial-In Serial-Out (SISO) Shift Register:



Operation:

1. **Data enters serially bit by bit through the input**
2. **With each clock pulse, data shifts one position to the right**
3. **After 4 clock pulses, the first input bit appears at the output**
4. **Example: For input "1101", need 4 clock pulses for complete transmission**

- **Primary use: Data conversion between serial and parallel formats**
- **Applications: Communication systems, data transfer between devices**
- **Advantages: Simple design, minimal interconnections**

Mnemonic

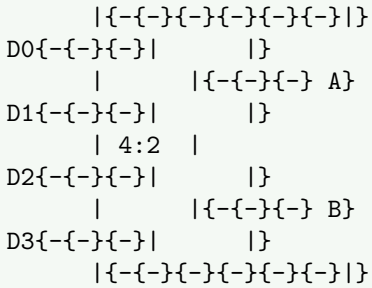
"Shift registers pass bits like a bucket brigade"

Question 4(a-OR) [3 marks]

Draw and Explain 4:2 Encoder.

Solution

4:2 Encoder Diagram:



Truth Table:

D3	D2	D1	D0	B	A
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Logical Expressions:

- **$A = D1 + D3$**
- **$B = D2 + D3$**
- **Encoder function: Converts one-hot input to binary code**
- **Priority encoders: Handle multiple active inputs by priority**
- **Applications: Keyboard scanning, interrupt handling**

Mnemonic

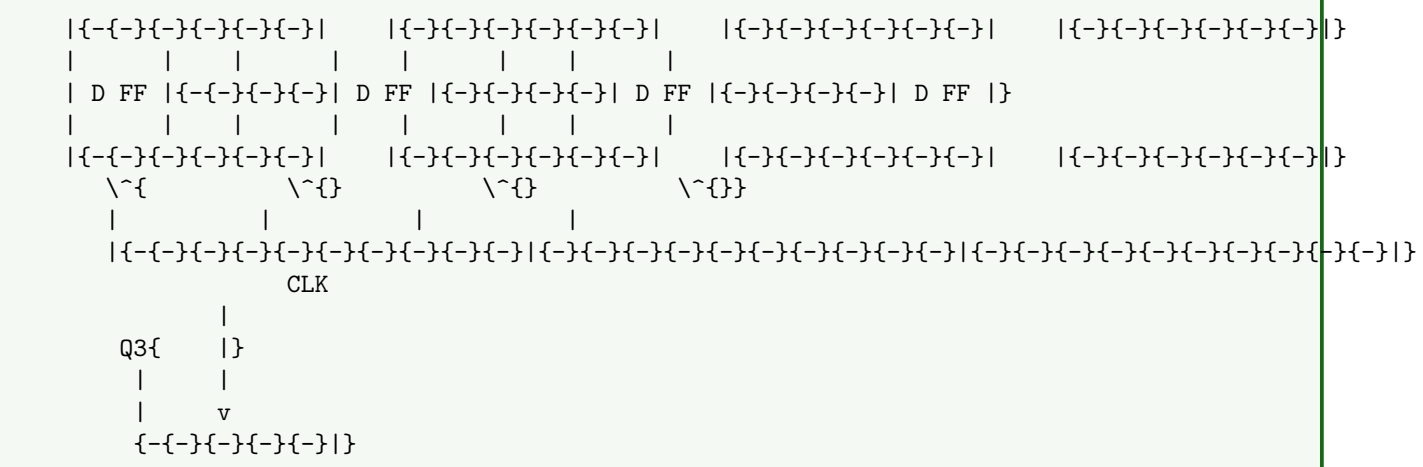
"One active line IN, binary code OUT"

Question 4(b-OR) [4 marks]

Draw and explain Johnson counter.

Solution

Johnson Counter (4-bit):



Counter Sequence:

Count	Q3	Q2	Q1	Q0
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
0	0	0	0	0

- **Johnson counter:** Also called twisted ring counter
- **Sequence length:** 2n states where n is number of flip-flops
- **Key feature:** Only one bit changes between adjacent states

Mnemonic

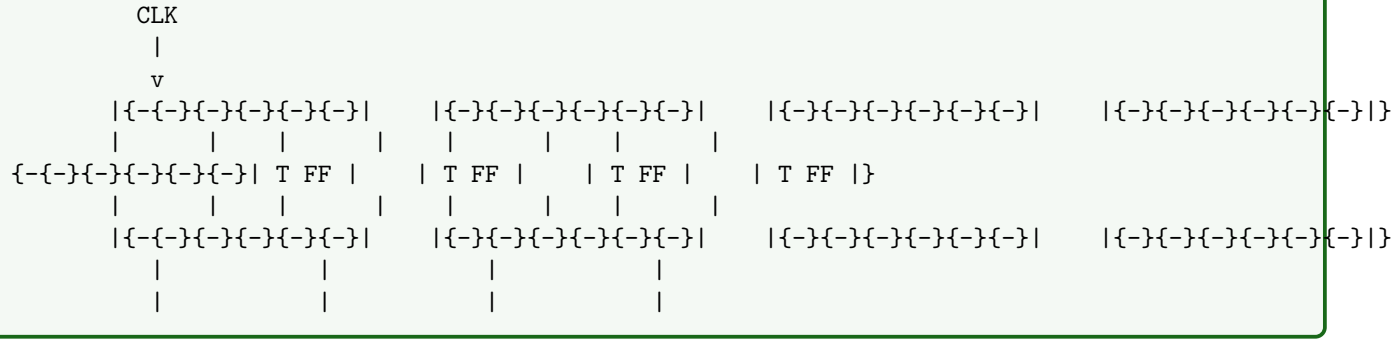
"Fill with 1's then clear with 0's"

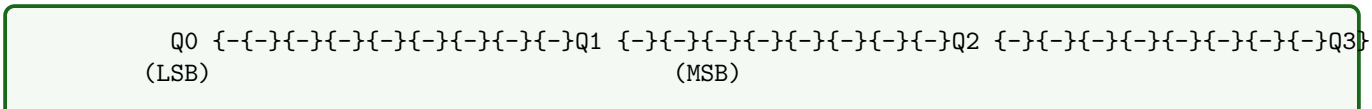
Question 4(c-OR) [7 marks]

Draw and explain 4 bit Ripple counter.

Solution

4-bit Ripple Counter:





Truth Table (Counting Sequence):

Count	Q3	Q2	Q1	Q0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
...	..	..	..	..
14	1	1	1	0
15	1	1	1	1
0	0	0	0	0

Working Principle:

1. All T inputs are connected to logic 1 (toggle mode)
2. First flip-flop toggles on every clock pulse
3. Each subsequent flip-flop toggles when the previous one changes from 1 to 0
4. Propagation delay increases with each stage

- Asynchronous counter: Clock drives only first flip-flop
- Ripple effect: Changes propagate through stages
- Disadvantage: Slower due to cumulative propagation delays

Mnemonic

"Change ripples through like falling dominoes"

Question 5(a) [3 marks]

Explain DRAM in short.

Solution

Dynamic Random Access Memory (DRAM):
DRAM is a type of semiconductor memory that stores each bit in a separate capacitor.
Key Features:

Feature	Description
Storage element	Single capacitor + transistor per bit
Density	Very high (more bits per chip)
Speed	Moderate (slower than SRAM)
Refresh	Required periodically (typically every few ms)
Power consumption	Lower than SRAM
Cost	Less expensive than SRAM

- Dynamic nature: Charge leaks over time, requiring refresh
- Applications: Main memory in computers
- Advantage: High density, low cost per bit

Mnemonic

"DRAM needs refreshing like a tired mind"

Question 5(b) [4 marks]

Define the following (1) Fan in (2) Propagation Delay

Solution

Fan-in:
Fan-in refers to the maximum number of inputs that a logic gate can accept.
Characteristics of Fan-in:

- **Measures input load capability**
- **Affects circuit complexity and design**
- **Higher fan-in reduces gate count but increases complexity**
- **Different logic families have different fan-in limits**

Example: A standard TTL NAND gate typically has a fan-in of 8 inputs.
Propagation Delay:
Propagation delay is the time taken for a signal to travel from input to output of a logic gate.
Characteristics of Propagation Delay:

- **Measured in nanoseconds (ns)**
- **Critical for high-speed circuit performance**
- **Varies with temperature, loading, and supply voltage**
- **Different for rising and falling transitions**

Example: A typical TTL gate has a propagation delay of 10-20 ns.

- **Impact on circuits: Limits maximum operating frequency**
- **Calculation: Time between 50% points of input and output signals**

Mnemonic

"Fan-in counts inputs, Prop-delay counts time"

Question 5(c) [7 marks]

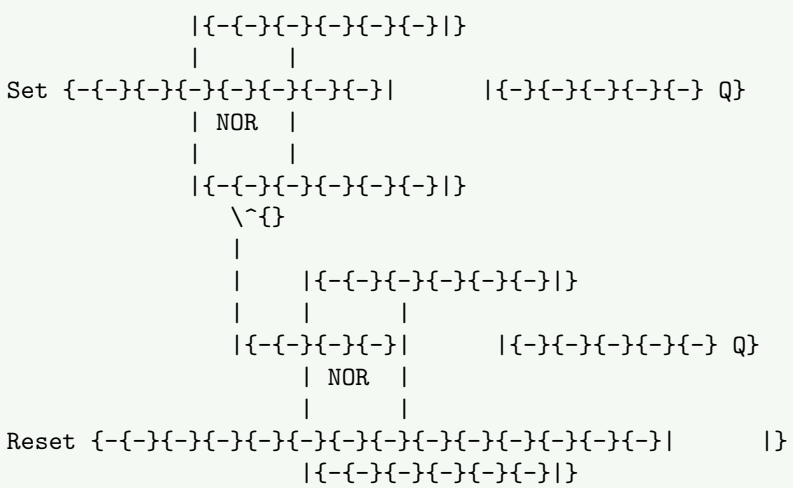
Do as Directed (i) Compare Logic families TTL and CMOS (ii) Draw Circuit Diagram of SR flip flop.

Solution

(i) Comparison of TTL and CMOS Logic Families:

Parameter	TTL	CMOS
Technology	Bipolar transistors	MOSFETs
Supply voltage	5V (fixed)	3-15V (flexible)
Power consumption	Higher	Very low (static)
Speed	Medium to high	Low to very high
Noise margin	Moderate	High
Fan-out	10-20	>50
Propagation delay	5-10 ns	10-100 ns (standard)
Input impedance	4-40 kΩ	Very high (10^{12})
Output impedance	100-300 Ω	Variable
Susceptibility to static	Low	High

(ii) SR Flip-Flop Circuit Diagram:



Truth Table:

S	R	Q	Q'	Remarks
0	0	Q	Q'	Memory (no change)
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Invalid (avoid)

- **SR flip-flop:** Basic memory element in digital circuits
- **Operation:** Set (S=1, R=0) makes Q=1; Reset (S=0, R=1) makes Q=0
- **Memory state:** When S=0, R=0, output remains unchanged

Mnemonic

"SR: Set-Reset, memory when both low"

Question 5(a-OR) [3 marks]

Write short note on E Waste of Digital Chips.

Solution

E-Waste of Digital Chips:
E-waste from digital chips refers to discarded electronic devices containing semiconductor components that require special handling and disposal.
Key Concerns:

Aspect	Details
Hazardous materials	Lead, mercury, cadmium, brominated flame retardants
Environmental impact	Soil and water contamination if improperly disposed
Resource recovery	Contains valuable metals (gold, silver, copper)
Volume	Growing rapidly with technological advancement
Regulations	Governed by WEEE, RoHS directives in many countries

Management Approaches:

- Recycling through authorized e-waste handlers
- Recovery of precious metals
- Safe disposal of toxic components
- Extended producer responsibility programs
- Challenges: Informal recycling causing health hazards
- Solutions: Design for disassembly, green manufacturing

Mnemonic

"Digital waste needs digital-age solutions"

Question 5(b-OR) [4 marks]

Define the following (1) Fan out (2) Noise margin

Solution

Fan-out:
Fan-out is the maximum number of gate inputs that can be driven by a single logic gate output while maintaining proper logic levels.
Characteristics of Fan-out:

- Measures output drive capability
- Affects design flexibility and cost
- Higher fan-out allows simpler wiring
- Limited by current sourcing/sinking capacity

Example: A standard TTL gate has a fan-out of 10, meaning it can drive 10 similar gates.
Noise Margin:
Noise margin is the amount of noise voltage that can be added to an input signal without causing an undesired change in the circuit output.
Characteristics of Noise Margin:

- Expressed in volts
- Measures circuit immunity to electrical noise
- Higher noise margin means more reliable operation
- Different for high and low logic levels

Example: TTL has noise margins of approximately 0.4V for logic low and 0.7V for logic high.

- Calculation: Difference between guaranteed output and required input levels
- Importance: Critical in electrically noisy environments

Mnemonic

"Fan-out counts outputs, Noise margin fights interference"

Question 5(c-OR) [7 marks]

Do as Directed (i) Write short note on ROM (ii) Explain JK master slave flipflop.

Solution

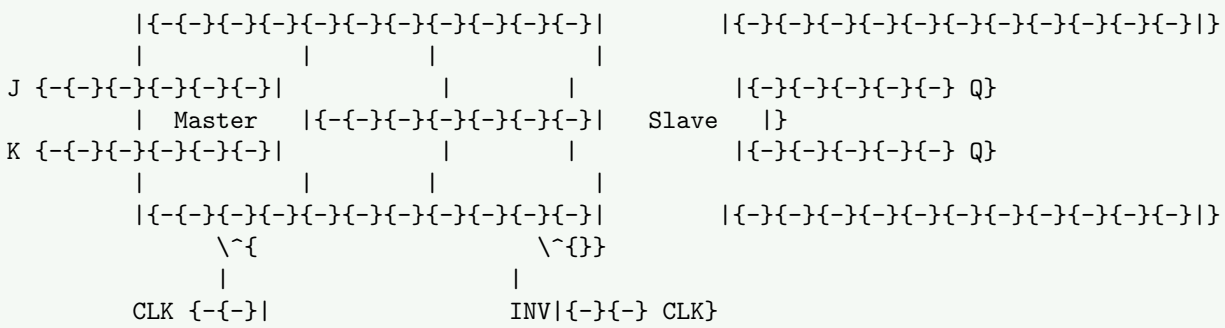
(i) Short Note on ROM:
ROM (Read-Only Memory) is a non-volatile memory used to store permanent or semi-permanent data.
Types of ROM:

Type	Characteristics	Programming
Mask ROM	Factory programmed	During manufacturing
PROM	One-time programmable	Electrical fusing by user
EPROM	Erasable with UV light	Electrical programming
EEPROM	Electrically erasable	Electrical programming/erasing
Flash ROM	Fast electrical erase	Block-wise erase/write

Applications:

- **Firmware and BIOS storage**
- **Look-up tables for fixed functions**
- **Microcode in processors**
- **Boot code in computers**
- **Data retention: Maintains data without power**
- **Access time: Typically 45-150 ns**
- **Density: High storage capacity**

(ii) JK Master-Slave Flip-Flop:



Truth Table:

J	K	Q(next)	Function
0	0	Q	No change
0	1	0	Reset
1	0	1	Set
1	1	Q’	Toggle

Operation:

1. **Master stage: When CLK=1, master latch samples J and K inputs**
2. **Slave stage: When CLK=0, slave latch samples master output**
3. **Two-phase operation: Prevents race condition (changes occur only on clock edge)**
4. **Advantage: More versatile than SR flip-flop (no invalid state)**

- **Toggle mode: When J=K=1, output toggles on each clock cycle**
- **Applications: Counters, shift registers, sequential circuits**

Mnemonic

“J-K: Set-Reset-Toggle, Master leads Slave follows”