VLSI (4361102) - Winter 2024 Solution

Milav Dabgar

November 21, 2024

Question 1 [a marks]

3 Write advantages of High K FINFET.

Solution

Table 1. Advantages of High K FINFET

Advantage	Description
Reduced leakage current	Better gate control reduces power consumption
Improved performance	Higher drive current and faster switching
Better scalability	Enables continued Moore's law scaling

- **High K dielectric**: Reduces gate leakage significantly
- **3D structure**: Better electrostatic control over channel
- **Lower power**: Reduced static and dynamic power consumption

Mnemonic

High Performance, Low Power, Better Control

Question 1 [b marks]

4 Define terms: (1) pinch off point (2) Threshold Voltage.

Solution

Table 2. Key MOSFET Parameters

Term	Definition	Significance
Pinch-off Point	Point where channel becomes completely depleted	Marks transition to saturation region
Threshold Voltage	Minimum V_{GS} needed to form conducting channel	Determines ON/OFF switching point

- **Pinch-off point**: $V_{DS} = V_{GS} - V_T$, channel narrows to zero width
- **Threshold voltage**: Typically 0.7V for enhancement MOSFET
- **Critical parameters**: Both determine MOSFET operating regions

Mnemonic

Threshold Turns ON, Pinch-off Points to Saturation

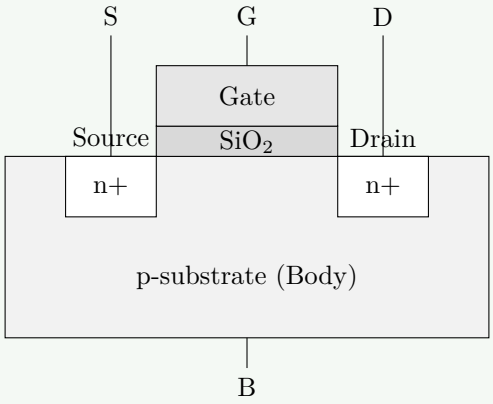
Question 1 [c marks]

7 Draw and explain structure of MOSFET transistor.

Solution

Table 3. Structure Components

Component	Material	Function
Gate	Polysilicon/Metal	Controls channel formation
Gate oxide	SiO2	Insulates gate from substrate
Source/Drain	n+ doped silicon	Current entry/exit points
Substrate	p-type silicon	Provides body connection



- **Channel formation**: Occurs at oxide-semiconductor interface
- **Enhancement mode**: Channel forms when $V_{GS} > V_T$
- **Four-terminal device**: Gate, Source, Drain, Body connections

Mnemonic

Gate Controls, Oxide Isolates, Source-Drain Conducts

Question 1 [c marks]

7 Compare Full Voltage Scaling and Constant Voltage Scaling.

Solution

Table 4. Full Voltage Scaling vs Constant Voltage Scaling

Parameter	Full Voltage Scaling	Constant Voltage Scaling
Supply voltage	Scaled down by α	Remains constant
Gate oxide thickness	Scaled down by α	Scaled down by α
Channel length	Scaled down by α	Scaled down by α
Power density	Remains constant	Increases by α^2
Performance	Moderate improvement	Better performance
Reliability	Better	Degraded due to high fields

- **Full scaling**: All dimensions and voltages scaled proportionally
- **Constant voltage**: Only physical dimensions scaled, voltage unchanged

- **Trade-off**: Performance vs power vs reliability

Mnemonic

Full scales All, Constant keeps Voltage

Question 2 [a marks]

3 Draw Resistive Load Inverter. Write the input voltage range for different operating region of operation.

Solution

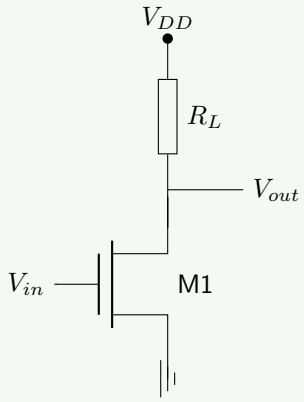


Table 5. Operating Regions

Region	Input Voltage Range	Output State
Cut-off	$V_{in} < V_T$	$V_{out} = V_{DD}$
Triode	$V_T < V_{in} < V_{DD} - V_T$	Transition
Saturation	$V_{in} > V_{DD} - V_T$	$V_{out} \approx 0V$

Mnemonic

Cut-off High, Triode Transition, Saturation Low

Question 2 [b marks]

4 Draw and Explain VDS-ID and VGS-ID characteristics of N channel MOSFET.

Solution

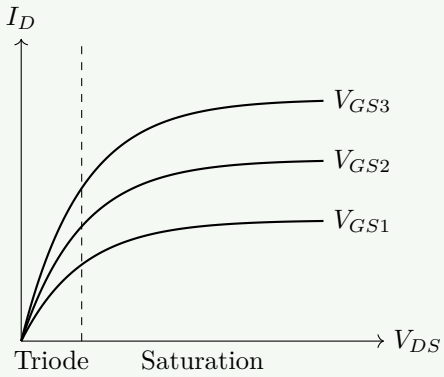


Table 6. NMOS Characteristics

Characteristic	Region	Behavior
$V_{DS} - I_D$	Triode	Linear increase with V_{DS}
$V_{DS} - I_D$	Saturation	Constant I_D (square law)
$V_{GS} - I_D$	Sub-threshold	Exponential increase
$V_{GS} - I_D$	Above V_T	Square law relationship

- **Triode region**: I_D increases linearly with V_{DS}
- **Saturation**: I_D independent of V_{DS}, depends on V_{GS}
- **Square law**: $I_D \propto (V_{GS} - V_T)^2$ in saturation

Mnemonic

Linear in Triode, Square in Saturation

Question 2 [c marks]

7 Draw & Explain working of Depletion Load NMOS Inverter circuit.

Solution

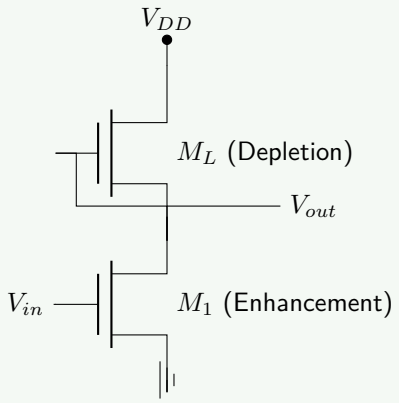


Table 7. Depletion Load Inverter Operation

Input	M1 State	ML State	Output
Low (0V)	Cut-off	Active load	High (V_{DD})
High (V_{DD})	Saturated	Linear	Low

- **Depletion load**: Always conducting, acts as current source
- **Better performance**: Higher output voltage swing than resistive load
- **Gate connection**: ML gate tied to source for depletion operation
- **Improved noise margin**: Better V_{OH} compared to enhancement load

Mnemonic

Depletion Always ON, Enhancement Controls Flow

Question 2 [a marks]

3 Describe advantages of CMOS Inverter.

Solution

Table 8. Advantages of CMOS Inverter

Advantage	Benefit
Zero static power	No current in steady state
Full voltage swing	Output swings from 0V to V_{DD}
High noise margins	Better noise immunity

- **Complementary operation**: One transistor always OFF
- **High input impedance**: Gate isolation provides high impedance
- **Fast switching**: Low parasitic capacitances

Mnemonic

Zero Power, Full Swing, High Immunity

Question 2 [b marks]

4 Draw and Explain Noise Margin in detail.

Solution

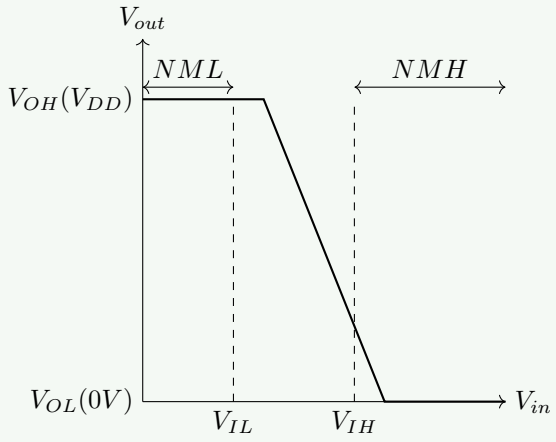


Table 9. Noise Margin Parameters

Parameter	Formula	Typical Value
N_{MH}	$V_{OH} - V_{IH}$	40% of V_{DD}
N_{ML}	$V_{IL} - V_{OL}$	40% of V_{DD}

- **High noise margin**: Immunity to positive noise
- **Low noise margin**: Immunity to negative noise
- **Better CMOS**: Higher noise margins than other logic families

Mnemonic

High goes Higher, Low goes Lower

Question 2 [c marks]

7 Draw and Explain VTC of N MOS Inverter.

Solution

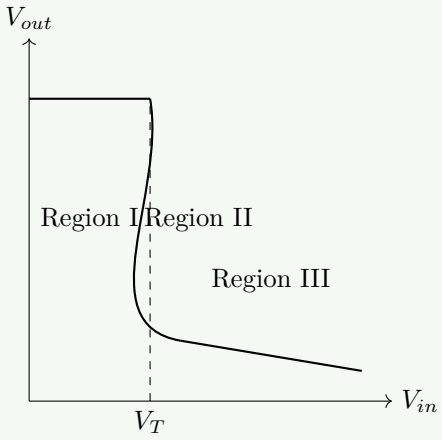


Table 10. NMOS Inverter Operating Regions

Region	V_{in} Range	M1 State	V_{out}
I	0 to V_T	Cut-off	V_{DD}
II	V_T to $V_T + V_{TL}$	Saturation	Decreasing
III	$V_T + V_{TL}$ to V_{DD}	Triode	Low

- **Region I**: M1 OFF, no current flow, $V_{out} = V_{DD}$
- **Region II**: M1 in saturation, sharp transition
- **Region III**: M1 in triode, gradual decrease
- **Load line**: Determines operating point intersection

Mnemonic

Cut-off High, Saturation Sharp, Triode Low

Question 3 [a marks]

3 Draw and explain generalized multiple input NOR gate structure with Depletion NMOS load.

Solution

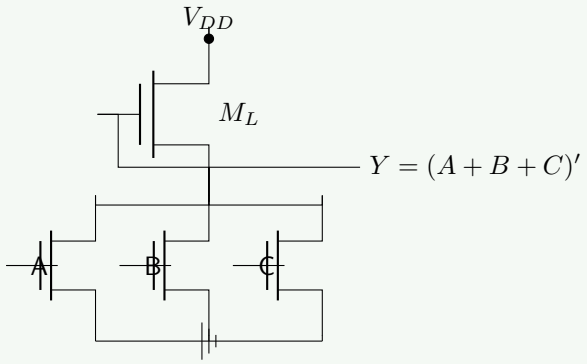


Table 11. Truth Table for NOR Gate

Inputs	Any Input High?	Output Y
All Low	No	High (1)
Any High	Yes	Low (0)

- **Parallel NMOS**: Any input HIGH pulls output LOW
- **NOR operation**: $Y = (A + B + C)'$
- **Depletion load**: Provides pull-up current

Mnemonic

Parallel Pulls Down, Depletion Pulls Up

Question 3 [b marks]

4 Differentiate AOI and OAI logic circuits.

Solution

Table 12. AOI vs OAI Logic

Parameter	AOI (AND-OR-Invert)	OAI (OR-AND-Invert)
Logic function	$Y = (AB + CD)'$	$Y = ((A + B)(C + D))'$
NMOS structure	Series-parallel	Parallel-series
PMOS structure	Parallel-series	Series-parallel
Complexity	Moderate	Moderate

- **AOI**: AND terms ORed then inverted
- **OAI**: OR terms ANDed then inverted
- **CMOS implementation**: Dual network structure
- **Applications**: Complex logic functions in single stage

Mnemonic

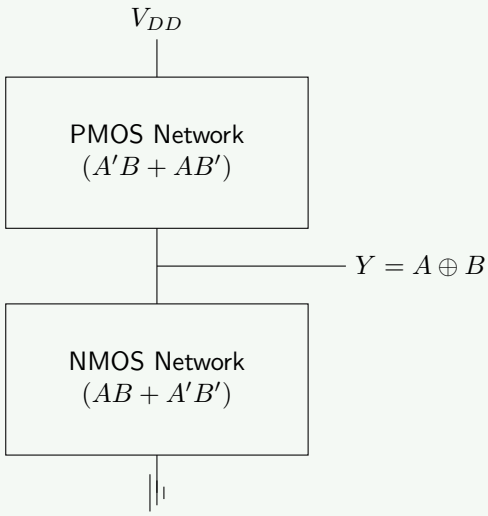
AOI: AND-OR-Invert, OAI: OR-AND-Invert

Question 3 [c marks]

7 Implement two input EX-OR gate using CMOS, and logic function Z = (AB +CD)' using NMOS Load.

Solution

EX-OR CMOS Implementation:



Z = (AB + CD)' NMOS Implementation:

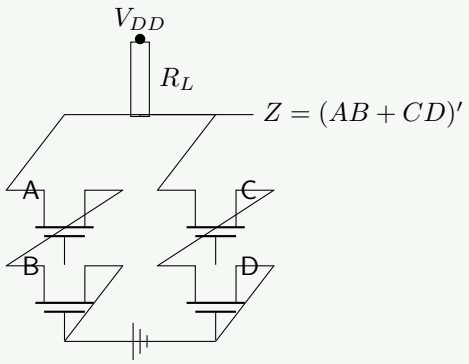


Table 13. Logic Implementation Comparison

Circuit	Function	Implementation
EX-OR	$A \oplus B$	Complementary CMOS
AOI	$(AB + CD)'$	Series-parallel NMOS

- **EX-OR**: Requires transmission gates for efficient implementation
- **AOI function**: Natural NMOS implementation
- **Power consideration**: CMOS has zero static power

Mnemonic

EX-OR needs Transmission, AOI uses Series-Parallel

Question 3 [a marks]

3 Draw and explain generalized multiple input NAND gate structure with Depletion NMOS load.

Solution

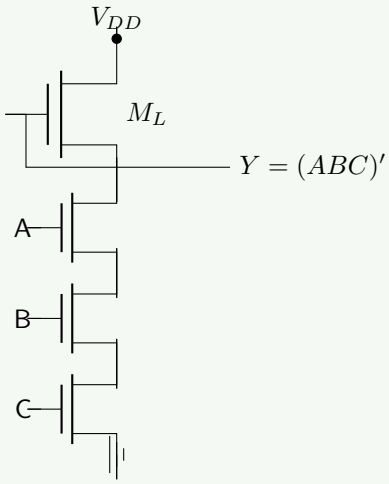


Table 14. NAND Gate Operation

Condition	Path to Ground	Output Y
All inputs HIGH	Complete path	Low (0)
Any input LOW	Broken path	High (1)

- **Series NMOS**: All inputs must be HIGH to pull output LOW
- **NAND operation**: $Y = (ABC)'$
- **Depletion load**: Always provides pull-up current

Mnemonic

Series Needs All, NAND Says Not-AND

Question 3 [b marks]

4 Implement logic function Y = ((P+R)(S+T))' using CMOS logic.

Solution

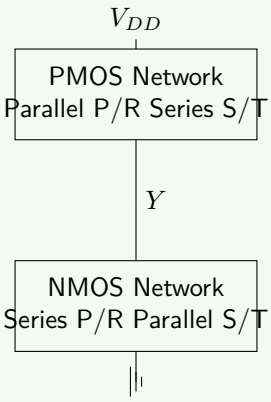


Table 15. Truth Table Implementation

PMOS Network	NMOS Network	Operation
$(P+R)'+(S+T)'$	$(P+R)(S+T)$	Complementary
$P'R'+S'T'$	$PS+PT+RS+RT$	De Morgan's law

- **PMOS**: Parallel within groups, series between groups
- **NMOS**: Series within groups, parallel between groups
- **Dual network**: Ensures complementary operation

Mnemonic

PMOS does Opposite of NMOS

Question 3 [c marks]

7 Describe the working of SR latch circuit.

Solution

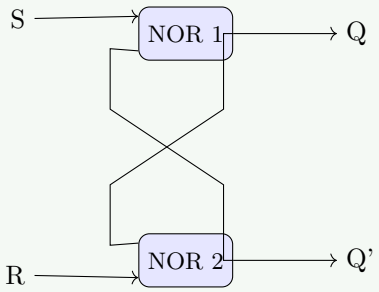


Table 16. SR Latch Truth Table

S	R	Q(n+1)	Q'(n+1)	State
0	**0**	Q(n)	Q'(n)	Hold
0	**1**	0	1	Reset
1	**0**	1	0	Set
1	**1**	0	0	Invalid

- **Set operation**: S=1, R=0 makes Q=1
- **Reset operation**: S=0, R=1 makes Q=0
- **Hold state**: S=0, R=0 maintains previous state
- **Invalid state**: S=1, R=1 should be avoided
- **Cross-coupled**: Output of one gate feeds input of other

Mnemonic

Set Sets, Reset Resets, Both Bad

Question 4 [a marks]

3 Compare Etching methods in chip fabrication.

Solution

Table 17. Etching Methods Comparison

Method	Type	Advantages	Disadvantages
Wet Etching	Chemical	High selectivity, simple	Isotropic, undercut
Dry Etching	Physical/Chemical	Anisotropic, precise	Complex equipment
Plasma Etching	Ion bombardment	Directional control	Damage to surface

- **Wet etching**: Uses liquid chemicals, attacks all directions
- **Dry etching**: Uses gases/plasma, better directional control
- **Selectivity**: Ability to etch one material over another

Mnemonic

Wet is Wide, Dry is Directional

Question 4 [b marks]

4 Write short note on Lithography.

Solution

Table 18. Lithography Process Steps

Step	Process	Purpose
Resist coating	Spin-on photoresist	Light-sensitive layer
Exposure	UV light through mask	Pattern transfer
Development	Remove exposed resist	Reveal pattern
Etching	Remove unprotected material	Create features

- **Pattern transfer**: From mask to silicon wafer
- **Resolution**: Determines minimum feature size
- **Alignment**: Critical for multiple layer processing
- **UV wavelength**: Shorter wavelength gives better resolution

Mnemonic

Coat, Expose, Develop, Etch

Question 4 [c marks]

7 Explain Regularity, Modularity and Locality.

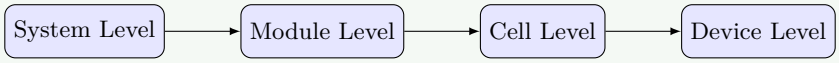
Solution

Table 19. Design Principles

Principle	Definition	Benefits	Example
Regularity	Repeated identical structures	Easier design, testing	Memory arrays
Modularity	Hierarchical design blocks	Reusability, maintainability	Standard cells
Locality	Related functions grouped	Reduced interconnect	Functional blocks

Implementation Details:

- **Regularity**: Same cell repeated multiple times reduces design complexity
- **Modularity**: Top-down design with well-defined interfaces
- **Locality**: Minimizes wire delays and routing congestion
- **Design benefits**: Faster design cycle, better testability
- **Manufacturing**: Improved yield through regular patterns



Mnemonic

Regular Modules with Local Connections

Question 4 [a marks]

3 Define Design Hierarchy.

Solution

Table 20. Design Hierarchy Levels

Level	Description	Components
System	Complete chip functionality	Processors, memories
Module	Major functional blocks	ALU, cache, I/O
Cell	Basic logic elements	Gates, flip-flops

- **Top-down approach**: System broken into smaller modules
- **Abstraction levels**: Each level hides lower level details
- **Interface definition**: Clear boundaries between levels

Mnemonic

System to Module to Cell

Question 4 [b marks]

4 Draw and Explain VLSI design flow chart.

Solution

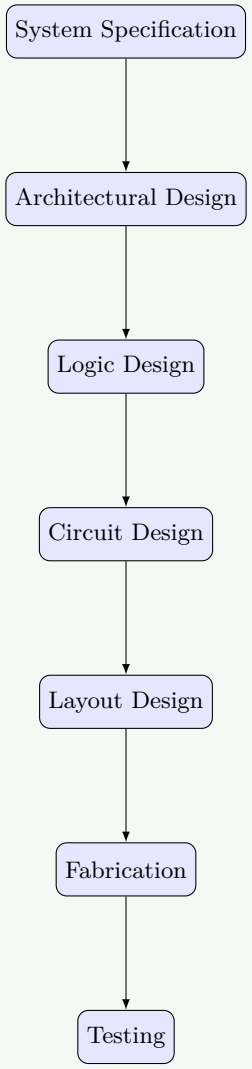


Table 21. Design Flow Stages

Stage	Input	Output	Tools
Architecture	Specifications	Block diagram	High-level modeling
Logic	Architecture	Gate netlist	HDL synthesis
Circuit	Netlist	Transistor sizing	SPICE simulation
Layout	Circuit	Mask data	Place & route

Mnemonic

Specify, Architect, Logic, Circuit, Layout, Fabricate, Test

Question 4 [c marks]

7 Write short note on 'VLSI Fabrication Process'

Solution

Table 22. Major Fabrication Steps

Process	Purpose	Result
Oxidation	Grow SiO2 layer	Gate oxide formation
Lithography	Pattern transfer	Define device areas
Etching	Remove unwanted material	Create device structures
Ion Implantation	Add dopants	Create P/N regions
Deposition	Add material layers	Metal interconnects
Diffusion	Spread dopants	Junction formation

Process Flow:

- **Wafer preparation**: Clean silicon substrate
- **Device formation**: Create transistors through multiple steps
- **Interconnect**: Add metal layers for connections
- **Passivation**: Protect completed circuit
- **Testing**: Verify functionality before packaging

Clean Room Requirements:

- **Class 1-10**: Ultra-clean environment needed
- **Temperature control**: Precise process control
- **Chemical purity**: High-grade materials required

Mnemonic

Oxidize, Pattern, Etch, Implant, Deposit, Diffuse

Question 5 [a marks]

3 Compare different styles of Verilog programming in VLSI.

Solution

Table 23. Verilog Modeling Styles

Style	Description	Application
Behavioral	Algorithm description	High-level modeling
Dataflow	Boolean expressions	Combinational logic
Structural	Gate-level description	Hardware representation

- **Behavioral**: Uses always blocks, if-else, case statements
- **Dataflow**: Uses assign statements with Boolean operators
- **Structural**: Instantiates gates and modules explicitly

Mnemonic

Behavior Describes, Dataflow Assigns, Structure Connects

Question 5 [b marks]

4 Write Verilog program of NAND gate using behavioral method.

Solution

```
module nand_gate_behavioral(
    input wire a, b,
    output reg y
);

always @(a or b) begin
    if (a == 1'b1 && b == 1'b1)
        y = 1'b0;
    else
        y = 1'b1;
end

endmodule
```

Code Explanation:

- **Always block**: Executes when inputs change
- **Sensitivity list**: Contains all input signals
- **Conditional statement**: Implements NAND logic
- **Reg declaration**: Required for procedural assignment

Mnemonic

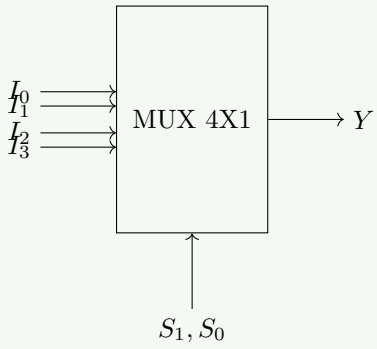
Always watch, IF both high THEN low ELSE high

Question 5 [c marks]

7 Draw 4X1 multiplexer circuit. Develop Verilog program of the circuit using case statement.

Solution

4X1 Multiplexer Circuit:



Verilog Code:

```
module mux_4x1_case(
    input wire [1:0] sel,
    input wire i0, i1, i2, i3,
    output reg y
);

always @(*) begin
    case (sel)
        2'b00: y = i0;
        2'b01: y = i1;
        2'b10: y = i2;
```

```
        2'b11: y = i3;
        default: y = 1'bx;
    endcase
end

endmodule
```

Table 24. MUX Truth Table

S1	S0	Output Y
0	**0**	I0
0	**1**	I1
1	**0**	I2
1	**1**	I3

Mnemonic

Case Selects, Default Protects

Question 5 [a marks]

3 Define Testbench with example.

Solution

Testbench Definition: Testbench is a Verilog module that provides stimulus to design under test (DUT) and monitors its response.
Example Testbench:

```
module test_and_gate;
    reg a, b;
    wire y;

    and_gate dut(.a(a), .b(b), .y(y));

    initial begin
        a = 0; b = 0; #10;
        a = 0; b = 1; #10;
        a = 1; b = 0; #10;
        a = 1; b = 1; #10;
    end
endmodule
```

- **DUT instantiation**: Creates instance of design under test
- **Stimulus generation**: Provides input test vectors
- **No ports**: Testbench is top-level module

Mnemonic

Test Provides Stimulus, Monitors Response

Question 5 [b marks]

4 Write Verilog program of Half Adder using Dataflow method.

Solution

```
module half_adder_dataflow(
    input wire a, b,
    output wire sum, carry
);

assign sum = a ^ b;    // XOR for sum
assign carry = a & b;  // AND for carry

endmodule
```

Logic Implementation:

- **Sum**: XOR operation between inputs
- **Carry**: AND operation between inputs
- **Assign statement**: Continuous assignment for dataflow
- **Boolean operators**: ^ (XOR), & (AND)

Table 25. Half Adder Truth Table

A	B	Sum	Carry
0	**0**	0	0
0	**1**	1	0
1	**0**	1	0
1	**1**	0	1

Mnemonic

XOR Sums, AND Carries

Question 5 [c marks]

7 Write function of Encoder. Develop code of 8X3 Encoder using if....else statement.

Solution

Encoder Function: Encoder converts 2^n input lines to n output lines. 8X3 encoder converts 8 inputs to 3-bit binary output.

Priority Table:

Input	Binary Output
I7	111
I6	110
I5	101
I4	100
I3	011
I2	010
I1	001
I0	000

Verilog Code:

```
module encoder_8x3(
    input wire [7:0] i,
    output reg [2:0] y
);

```

```
always @(*) begin
    if (i[7])
        y = 3'b111;
    else if (i[6])
        y = 3'b110;
    else if (i[5])
        y = 3'b101;
    else if (i[4])
        y = 3'b100;
    else if (i[3])
        y = 3'b011;
    else if (i[2])
        y = 3'b010;
    else if (i[1])
        y = 3'b001;
    else if (i[0])
        y = 3'b000;
    else
        y = 3'bxxx;
end

endmodule
```

- **Priority encoding**: Higher index inputs have priority
- **If-else chain**: Implements priority logic
- **Binary encoding**: Converts active input to binary representation

Mnemonic

Priority from High to Low, Binary Output Flows