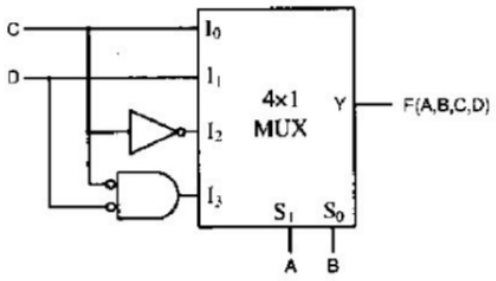
Question

Q.39 The Boolean function realized by the logic circuit shown is



(A) $F = \Sigma m(0, 1, 3, 5, 9, 10, 14)$
(B) $F = \Sigma m(2, 3, 5, 7, 8, 12, 13)$
(C) $F = \Sigma m(1, 2, 4, 5, 11, 14, 15)$
(D) $F = \Sigma m(2, 3, 5, 7, 8, 9, 12)$

Implementation of 4x1 Multiplexer using Verilog and Vaman Board

Hardware Implementation Report

Q.1 FPGA Experiment

Objective

To design, simulate, and physically implement a 4x1 multiplexer using Verilog HDL on a Vaman FPGA development board. The output of the multiplexer is observed using LEDs.

Hardware Required

- Vaman FPGA Board
- LEDs
- Resistors (220 Ω)
- Jumper Wires
- Breadboard
- sevensegment

Theory

A multiplexer (MUX) is a combinational circuit that selects one of several input signals and forwards the selected input to a single output line. A 4x1 multiplexer has 4 input lines, 2 selection lines, and 1 output line.

The truth table for a 4x1 MUX is as follows:

Select Lines (S1 S0)	Selected Input	Output (Y)
00	I0	I0
01	I1	I1
10	I2	I2
11	I3	I3

Verilog Code

```
\begin{lstlisting}[style=verilog, caption={4x1 Multiplexer Verilog Code}]
module counter_7seg(
    input clk,        // Clock input from FPGA (e.g. 12 MHz)
    input rst,        // Reset button (active high)
    output reg [6:0] seg // 7-segment outputs (a,b,c,d,e,f,g)
);

    reg [23:0] count_div;  // Clock divider
    reg [3:0] counter;     // 4-bit counter (0{9)
    wire slow_clk;

    // Clock divider to slow down the blinking rate
    always @(posedge clk or posedge rst) begin
        if (rst)
            count_div <= 0;
        else
            count_div <= count_div + 1;
    end

    // Divide clock frequency
    assign slow_clk = count_div[23];  // Adjust this bit for speed

    // 4-bit counter logic
    always @(posedge slow_clk or posedge rst) begin
        if (rst)
            counter <= 0;
        else if (counter == 9)
            counter <= 0;
        else
            counter <= counter + 1;
    end

    // 7-segment decoder
```

```
    always @(*) begin
        case (counter)
            4'd0: seg = 7'b0000001; // 0
            4'd1: seg = 7'b1001111; // 1
            4'd2: seg = 7'b0010010; // 2
            4'd3: seg = 7'b0000110; // 3
            4'd4: seg = 7'b1001100; // 4
            4'd5: seg = 7'b0100100; // 5
            4'd6: seg = 7'b0100000; // 6
            4'd7: seg = 7'b0001111; // 7
            4'd8: seg = 7'b0000000; // 8
            4'd9: seg = 7'b0000100; // 9
            default: seg = 7'b1111111; // off
        endcase
    end

endmodule
\end{lstlisting}
```

PCF File (Pin Constraint File)

```
set_io a 3
set_io b 64
set_io c 62
set_io d 63
set_io e 61
set_io f 59
set_io g 57


set_io Z 56
set_io Y 23
set_io X 33
set_io W 27
```

Listing 1: .pcf File for Vaman Board Pin Mapping

Procedure

1. Write the Verilog code for a 4x1 multiplexer in counter.
2. Create a new project in the Vaman IDE or through command line using Yosys and nextpnr.
3. Add the Verilog file and PCF file to the project.
4. Synthesize and implement the design.
5. Generate the bitstream or binary (.bin) file.
6. Upload the bitstream to the Vaman board.

7. Connect LEDs and switches to observe the multiplexer output.

Output and Observation

When the select lines are changed, the output LED glows corresponding to the selected input line as per the truth table.

Conclusion

The 4x1 multiplexer was successfully implemented on the Vaman FPGA board. The output matched the expected behavior as per the truth table. This experiment demonstrates how combinational logic can be implemented on FPGA hardware using Verilog HDL.

Result

A 4x1 MUX was designed, simulated, and verified on the Vaman FPGA board using LEDs as output indicators.