**ECT312**

**DIGITAL SYSTEM DESIGN**

**ASSIGNMENT 1**

*Submitted by : M NihalRahman*

*S6 ECE*

*Roll no 47*

**QUESTION:**

Write the RTL Code and simulate the given design.

1. Design of vending machine

2. Design of ALU

**CODE:**

1. **Vending Machine**

module VendingMachine(

input clk,

input rst,

input [1:0] coin, // 00: No coin, 01: 5 unit, 10: 10 unit

output reg dispense

);

reg [3:0] total;

always @(posedge clk or posedge rst) begin

if (rst) begin

total <= 0;

dispense <= 0;

end else begin

case (coin)

2'b01: total <= total + 5;

2'b10: total <= total + 10;

default: total <= total;

endcase

if (total >= 15) begin

dispense <= 1;

total <= 0;

end else begin

dispense <= 0;

end

end

end

endmodule

**TESTBENCH:**

module VendingMachine\_tb();

reg clk, rst;

reg [1:0] coin;

wire dispense;

VendingMachine vm(clk, rst, coin, dispense);

always #5 clk = ~clk; // Clock generation

initial begin

clk = 0; rst = 1; coin = 2'b00;

#10 rst = 0;

#10 coin = 2'b01; // Insert 5 units

#10 coin = 2'b10; // Insert 10 units

#10 coin = 2'b01; // Insert 5 units

#20 $stop;

end

Endmodule

1. **ALU:**

module ALU(

input [3:0] A, B,

input [3:0] ALU\_Sel,

output reg [3:0] ALU\_Out,

output reg CarryOut

);

always @(\*) begin

CarryOut = 0;

case (ALU\_Sel)

4'b0000: {CarryOut, ALU\_Out} = A + B;

4'b0001: {CarryOut, ALU\_Out} = A - B;

4'b0010: ALU\_Out = A & B;

4'b0011: ALU\_Out = A | B;

4'b0100: ALU\_Out = A ^ B;

4'b0101: ALU\_Out = ~A;

4'b0110: ALU\_Out = A << 1;

4'b0111: ALU\_Out = A >> 1;

4'b1000: ALU\_Out = A \* B;

4'b1001: ALU\_Out = (B != 0) ? A / B : 4'b0000;

default: ALU\_Out = 4'b0000;

endcase

end

endmodule

**TESTBENCH :**

module ALU\_sim;

reg [3:0] A, B;

reg [3:0] ALU\_Sel;

wire [3:0] ALU\_Out;

wire CarryOut;

ALU uut (

.A(A),

.B(B),

.ALU\_Sel(ALU\_Sel),

.ALU\_Out(ALU\_Out),

.CarryOut(CarryOut)

);

initial begin

$monitor("Time=%0t | A=%b, B=%b, ALU\_Sel=%b, ALU\_Out=%b, CarryOut=%b",

$time, A, B, ALU\_Sel, ALU\_Out, CarryOut);

A = 4'b0101; B = 4'b0011;

ALU\_Sel = 4'b0000; #10; // Test Addition

ALU\_Sel = 4'b0001; #10; // Test Subtraction

ALU\_Sel = 4'b0010; #10; // Test AND

ALU\_Sel = 4'b0011; #10; // Test OR

ALU\_Sel = 4'b0100; #10; // Test XOR

ALU\_Sel = 4'b0101; #10; // Test Left Shift

ALU\_Sel = 4'b0110; #10; // Test Right Shift

ALU\_Sel = 4'b0111; #10; // Test Multiplication

ALU\_Sel = 4'b1000; #10; // Test Comparison

ALU\_Sel = 4'b1001; #10; // Test Division

$finish; // End simulation

end

endmodule**Output :**

1. **Vending machine:**



1. **ALU :**

