**Decoder2to4:**

module decoder2to4 (

input [1:0] in,

input en,

output reg [3:0] out

);

always @(\*)begin

if (en) begin

case(in)

2'b00:out=4'b0001;

2'b01:out=4'b0010;

2'b10:out=4'b0100;

2'b11:out=4'b1000;

default:out=4'bxxxx;

endcase

end else begin

out=4'b0000;

end

end

Endmodule

**Test-bench:**

`timescale 1ns / 1ps

module decoder2to4\_tb;

// Testbench signals

reg [1:0] in;

reg en;

wire [3:0] out;

// Instantiate the decoder

decoder2to4 uut (

.in(in),

.en(en),

.out(out)

);

initial begin

// Display header

$display("Time\t en in | out");

$monitor("%0t\t %b %b | %b", $time, en, in, out);

// Test all input combinations with enable = 1

en = 1;

in = 2'b00; #10;

in = 2'b01; #10;

in = 2'b10; #10;

in = 2'b11; #10;

// Test with enable = 0 (should always output 0000)

en = 0;

in = 2'b00; #10;

in = 2'b01; #10;

in = 2'b10; #10;

in = 2'b11; #10;

$finish;

end

endmodule

**Encoder4to2:**

module encoder4to2 (

input [3:0] in,

output reg [1:0] out

);

always @(\*) begin //instead of a or b or c ,lisiting all the signals that the block depends on.

case (in)

4'b0001: out = 2'b00;

4'b0010: out = 2'b01;

4'b0100: out = 2'b10;

4'b1000: out = 2'b11;

default: out = 2'bxx; // Undefined if more than one input is high

endcase

end

Endmodule

**Test-bench:**

`timescale 1ns / 1ps

module encoder4to2\_tb;

// Testbench signals

reg [3:0] in;

wire [1:0] out; //Testbenches do not have input/output ports ,because it will be driven from DUT,(design under test)

// Instantiate the encoder module

encoder4to2 uut ( //it creates instance encoder module and connects its ports to the testbench variables

.in(in),

.out(out)

);

initial begin

// Display header

$display("Time\tInput\tOutput");

$monitor("%0t\t%b\t%b", $time, in, out);

// Apply valid test inputs one at a time

in = 4'b0001; #10; // Expected output: 00

in = 4'b0010; #10; // Expected output: 01

in = 4'b0100; #10; // Expected output: 10

in = 4'b1000; #10; // Expected output: 11

// Optional: Apply invalid input (more than one bit high)

in = 4'b1100; #10; // Undefined output (xx)

in = 4'b0000; #10; // Undefined output (xx)

$finish;

end

Endmodule

**I solved hacker rank programs:**

