Ho Chi Minh University of Technology

Faculty of Computer Science and Engineering
Course: Logic Design with HDL (CO1025)

Logic Design Lab:

Weekly report - 2

Lecturer: Pham Quoc Cuong
Student: Phan Minh Toan - 1852798
Tran Nguyen Khoi - 1952797
Hua Vu Minh Hieu - 2052990
Nguyen Quang Khoi - 2153485

Ho Chi Minh, April 2022

Contents

List of Figures

Listings

1 Exercise 1

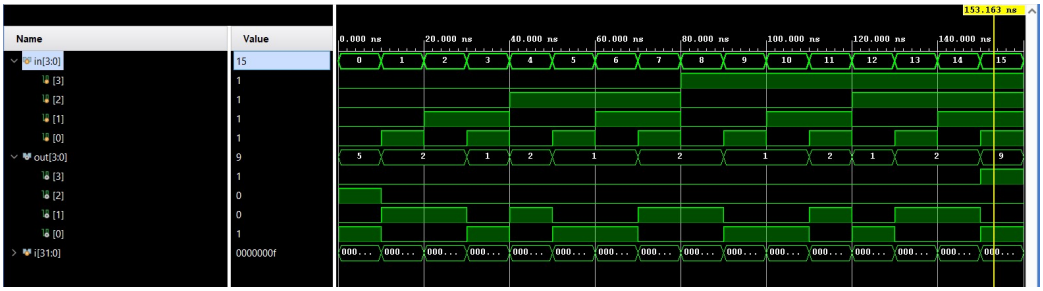

Figure 1: Exercise 1 waveform

```
module ex1(c_in, c_out);
    input [3:0] c_in;
    output [3:0] c_out;

    assign c_out[0] = ^c_in[3:0];
    assign c_out[1] = (^c_in[3:2])^(^c_in[1:0]);
    assign c_out[2] = ~|c_in[3:0];
    assign c_out[3] = &c_in[3:0];

endmodule
```

Listing 1: Exercise 1 Implemetation

```
`timescale 1 ns/10 ps
module ex1_tb();
    reg [3:0] in;
    wire [3:0] out;
    integer i;
    ex1 UUT(.c_in(in), .c_out(out));

    initial begin
        for(i = 0;i < 16;i = i+1) //run loop for 0 to 15.
        begin
            in = i;
            #10; //wait for 10 ns
        end
        $stop;
    end
endmodule
```

Listing 2: Exercise 1 Testbench Implemetation

2 Exercise 2

Value from MSB to LSB will be assign to a,b,c,d,e,f,g of 7 segments led.

Figure 2: Exercise 2 Waveform

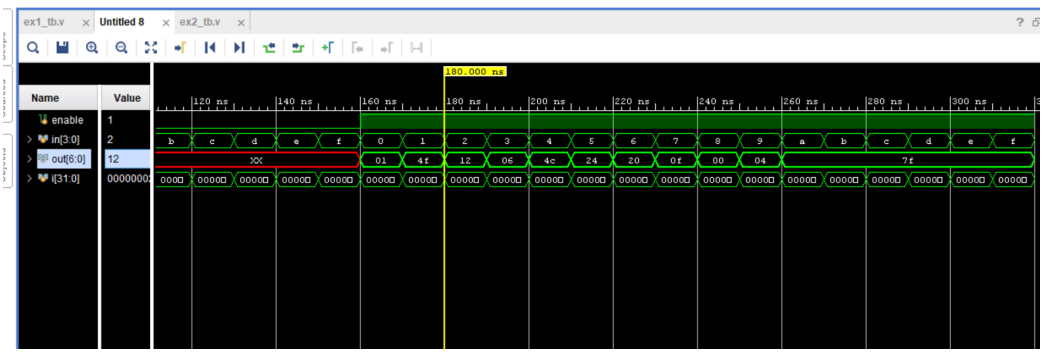

Figure 3: Exercise 2-1 Waveform

```
module ex2(en, bin_in, led_out);
    input en;
    input [3:0] bin_in;
    output reg [6:0] led_out;


    always @(bin_in or en)
    if(en) begin
        case (bin_in)
            4'b0000: led_out = 7'b0000001;//0
            4'b0001: led_out = 7'b1001111;//1
            4'b0010: led_out = 7'b0010010;//2
            4'b0011: led_out = 7'b0000110;//3
            4'b0100: led_out = 7'b1001100;//4
            4'b0101: led_out = 7'b0100100;//5
            4'b0110: led_out = 7'b0100000;//6
            4'b0111: led_out = 7'b0001111;//7
```

```
            4'b1000: led_out = 7'b0000000;//8
            4'b1001: led_out = 7'b0000100;//9
            default: led_out = 7'b1111111;
        endcase
    end

endmodule
```

Listing 3: Exercise 2 Implemetation

```
module tb_segment7;
    reg enable;
    reg [3:0]in;

    wire [6:0]out;
    integer i;

    // Instantiate the Unit Under Test (UUT)
    ex2 uut (.en(enable), .bin_in(in), .led_out(out));

//Apply inputs
    initial begin
        enable = 1'b0;
        for(i = 0;i < 16;i = i+1) //run loop for 0 to 15.
        begin
            in = i;
            #10; //wait for 10 ns
        end

        enable = 1'b1;
        for(i = 0;i < 16;i = i+1) //run loop for 0 to 15.
        begin
            in = i;
            #10; //wait for 10 ns
        end
        $stop;

    end

endmodule
```

Listing 4: Exercise 2 Testbench Implemetation

3 Exercise 3

RBG Led Implementation:

```
module rbg_led (sw, btn, led_r, led_b, led_g, led_w);
    input sw;
    input [3:0] btn;
    output reg led_r;
    output reg led_b;
    output reg led_g;
    output reg led_w;

    always @(btn or sw or led_r or led_b or led_g)
    case(btn)
        4'b0001: begin
            led_r = 1'b0;
            led_b = 1'b0;
            led_g = 1'b0;
            led_w = 1'b0;
        end

        //CASE LED CYAN = BLUE + GREEN
        4'b0001: begin
            if(sw == 1'b1) begin
                led_r = 1'b0;
                led_b = 1'b1;
                led_g = 1'b1;
                led_w = 1'b0;
            end
            else if(sw == 1'b0) begin
                led_r = 1'b0;
                led_b = 1'b1;
                led_g = 1'b0;
                led_w = 1'b0;
            end
        end

        //CASE LED YELLOW = RED + GREEN
        4'b0010: begin
            if(sw == 1'b1) begin
                led_r = 1'b1;
                led_b = 1'b0;
                led_g = 1'b1;
                led_w = 1'b0;
            end
            else if(sw == 1'b0) begin
                led_r = 1'b0;
                led_b = 1'b0;
                led_g = 1'b1;
                led_w = 1'b0;
            end
        end

```

```
        //CASE PURPLE = RED + BLUE
        4'b0100: begin
            if(sw == 1'b1) begin
                led_r = 1'b1;
                led_b = 1'b1;
                led_g = 1'b0;
                led_w = 1'b0;
            end
            else if(sw == 1'b0) begin
                led_r = 1'b1;
                led_b = 1'b0;
                led_g = 1'b0;
                led_w = 1'b0;
            end
        end

        //CASE WHITE
        4'b1000: begin
            led_r = 1'b0;
            led_b = 1'b0;
            led_g = 1'b0;
            led_w = 1'b1;
        end

        //
        default: begin
            if(sw == 1'b1) begin
                led_r = 1'b0;
                led_b = 1'b0;
                led_g = 1'b0;
                led_w = 1'b0;
            end
            else if(sw == 1'b0) begin
                led_r = 1'b1;
                led_b = 1'b1;
                led_g = 1'b1;
                led_w = 1'b1;
            end
        end
    endcase
endmodule
```

Listing 5: Exercise 3 Implemetation

```
`timescale 1ns/1ps

module rbg_led_tb();
    integer i;
    reg sw;
```

```
    reg [3:0] btn;

    wire led_r, led_b, led_g, led_w;
    rbg_led UUT (sw, btn, led_r, led_b, led_g, led_w);


    initial begin
        sw =1'b0;
        for(i = 0;i < 16;i = i+1)
            btn = i;
            #10

        sw =1'b1;
        for(i = 0;i < 16;i = i+1)
            btn = i;
            #10
        $stop;
    end
endmodule
```

Listing 6: Exercise 3 Testbench Implemetation