

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

1	Exercise 1	3
2	Exercise 2	4
3	Exercise 3	5



List of Figures

1	Exercise 1 waveform	3
2	Exercise 2 Waveform	4
3	Exercise 2-1 Waveform	4

Listings

1	Exercise 1 Implemetation	3
2	Exercise 1 Testbench Implemetation	3
3	Exercise 2 Implemetation	4
4	Exercise 2 Testbench Implemetation	5
5	Exercise 3 Implemetation	5
6	Exercise 3 Testbench Implemetation	7



1 Exercise 1

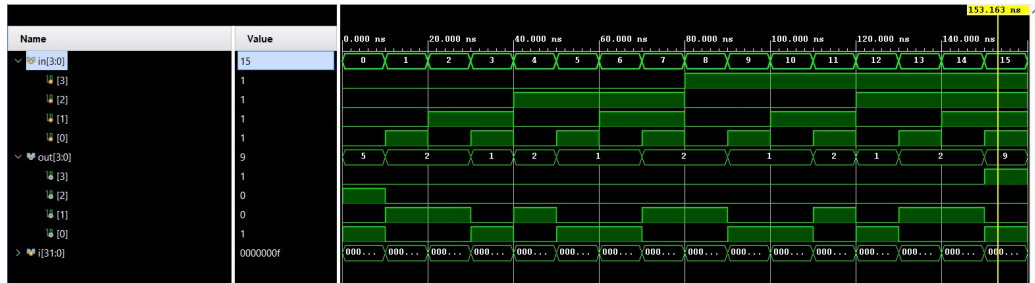


Figure 1: Exercise 1 waveform

```
1 module ex1(c_in , c_out);
2     input  [3:0] c_in;
3     output [3:0] c_out;
4
5     assign c_out[0] = ^c_in[3:0];
6     assign c_out[1] = (^c_in[3:2])^(^c_in[1:0]);
7     assign c_out[2] = ~|c_in[3:0];
8     assign c_out[3] = &c_in[3:0];
9
10    endmodule
```

Listing 1: Exercise 1 Implementation

```
1 `timescale 1 ns/10 ps
2 module ex1_tb();
3     reg [3:0] in;
4     wire [3:0] out;
5     integer i;
6     ex1 UUT(.c_in(in), .c_out(out));
7
8     initial begin
9         for(i = 0; i < 16; i = i+1) //run loop for 0 to 15.
10            begin
11                in = i;
12                #10; //wait for 10 ns
13            end
14            $stop;
15    end
16 endmodule
```

Listing 2: Exercise 1 Testbench Implementation



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

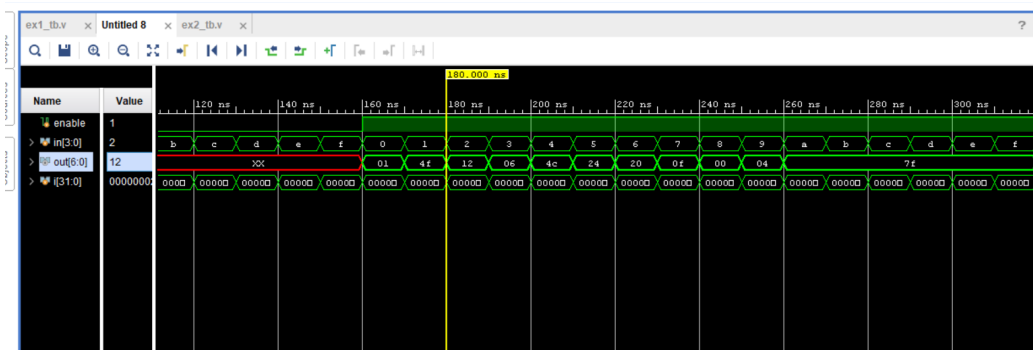


Figure 3: Exercise 2-1 Waveform

```
1 module ex2(en, bin_in , led_out);
2     input en;
3     input [3:0] bin_in;
4     output reg [6:0] led_out;
5
6
7     always @(bin_in or en)
8     if(en) begin
9         case (bin_in)
10            4'b0000: led_out = 7'b0000001; //0
11            4'b0001: led_out = 7'b1001111; //1
12            4'b0010: led_out = 7'b0010010; //2
13            4'b0011: led_out = 7'b0000110; //3
14            4'b0100: led_out = 7'b1001100; //4
15            4'b0101: led_out = 7'b0100100; //5
16            4'b0110: led_out = 7'b0100000; //6
17            4'b0111: led_out = 7'b0001111; //7
```



```
18         4'b1000: led_out = 7'b00000000; //8
19         4'b1001: led_out = 7'b0000100; //9
20         default: led_out = 7'b1111111;
21     endcase
22 end
23
24 endmodule
```

Listing 3: Exercise 2 Implementation

```
1 module tb_segment7;
2     reg enable;
3     reg [3:0] in;
4
5     wire [6:0] out;
6     integer i;
7
8     // Instantiate the Unit Under Test (UUT)
9     ex2 uut (.en(enable), .bin_in(in), .led_out(out));
10
11 //Apply inputs
12 initial begin
13     enable = 1'b0;
14     for(i = 0; i < 16; i = i+1) //run loop for 0 to 15.
15     begin
16         in = i;
17         #10; //wait for 10 ns
18     end
19
20     enable = 1'b1;
21     for(i = 0; i < 16; i = i+1) //run loop for 0 to 15.
22     begin
23         in = i;
24         #10; //wait for 10 ns
25     end
26     $stop;
27
28 end
29
30 endmodule
```

Listing 4: Exercise 2 Testbench Implementation

3 Exercise 3

RBG Led Implementation:



```
1 module rbg_led (sw, btn, led_r, led_b, led_g, led_w);
2     input sw;
3     input [3:0] btn;
4     output reg led_r;
5     output reg led_b;
6     output reg led_g;
7     output reg led_w;
8
9     always @(btn or sw or led_r or led_b or led_g)
10    case(btn)
11        4'b0001: begin
12            led_r = 1'b0;
13            led_b = 1'b0;
14            led_g = 1'b0;
15            led_w = 1'b0;
16        end
17
18        //CASE LED CYAN = BLUE + GREEN
19        4'b0001: begin
20            if(sw == 1'b1) begin
21                led_r = 1'b0;
22                led_b = 1'b1;
23                led_g = 1'b1;
24                led_w = 1'b0;
25            end
26            else if(sw == 1'b0) begin
27                led_r = 1'b0;
28                led_b = 1'b1;
29                led_g = 1'b0;
30                led_w = 1'b0;
31            end
32        end
33
34        //CASE LED YELLOW = RED + GREEN
35        4'b0010: begin
36            if(sw == 1'b1) begin
37                led_r = 1'b1;
38                led_b = 1'b0;
39                led_g = 1'b1;
40                led_w = 1'b0;
41            end
42            else if(sw == 1'b0) begin
43                led_r = 1'b0;
44                led_b = 1'b0;
45                led_g = 1'b1;
46                led_w = 1'b0;
47            end
48        end
49    end
```



```
50 //CASE PURPLE = RED + BLUE
51 4'b0100: begin
52     if(sw == 1'b1) begin
53         led_r = 1'b1;
54         led_b = 1'b1;
55         led_g = 1'b0;
56         led_w = 1'b0;
57     end
58     else if(sw == 1'b0) begin
59         led_r = 1'b1;
60         led_b = 1'b0;
61         led_g = 1'b0;
62         led_w = 1'b0;
63     end
64 end
65
66 //CASE WHITE
67 4'b1000: begin
68     led_r = 1'b0;
69     led_b = 1'b0;
70     led_g = 1'b0;
71     led_w = 1'b1;
72 end
73
74 //
75 default: begin
76     if(sw == 1'b1) begin
77         led_r = 1'b0;
78         led_b = 1'b0;
79         led_g = 1'b0;
80         led_w = 1'b0;
81     end
82     else if(sw == 1'b0) begin
83         led_r = 1'b1;
84         led_b = 1'b1;
85         led_g = 1'b1;
86         led_w = 1'b1;
87     end
88 end
89 endcase
90 endmodule
```

Listing 5: Exercise 3 Implementation

```
1 `timescale 1ns/1ps
2
3 module rgb_led_tb();
4     integer i;
5     reg sw;
```




```
6   reg [3:0] btn;  
7  
8   wire led_r , led_b , led_g , led_w;  
9   rbg_led UUT (sw, btn, led_r , led_b , led_g , led_w);  
10  
11  
12   initial begin  
13       sw =1'b0;  
14       for (i = 0; i < 16; i = i+1)  
15           btn = i;  
16           #10  
17  
18       sw =1'b1;  
19       for (i = 0; i < 16; i = i+1)  
20           btn = i;  
21           #10  
22       $stop;  
23   end  
24 endmodule
```

Listing 6: Exercise 3 Testbench Implementation