Hacettepe University

Computer Engineering Department

BM233 Logic Design Lab - 2020 Fall

FINAL PROJECT

January 31, 2021

Student name:
Şevval Atmaca

Student Number:
b21827115

1 Problem Statement

SCP-079 is an Exidy Sorcerer microcomputer built in 1978. SCP079 gained intelligence. Now, SCP-079 is trying to escape from the Foundation and destroy mankind. In this experiment, we design a verilog project using many scenarios what scp-079 like succeeds and fails.

2 State Diagram

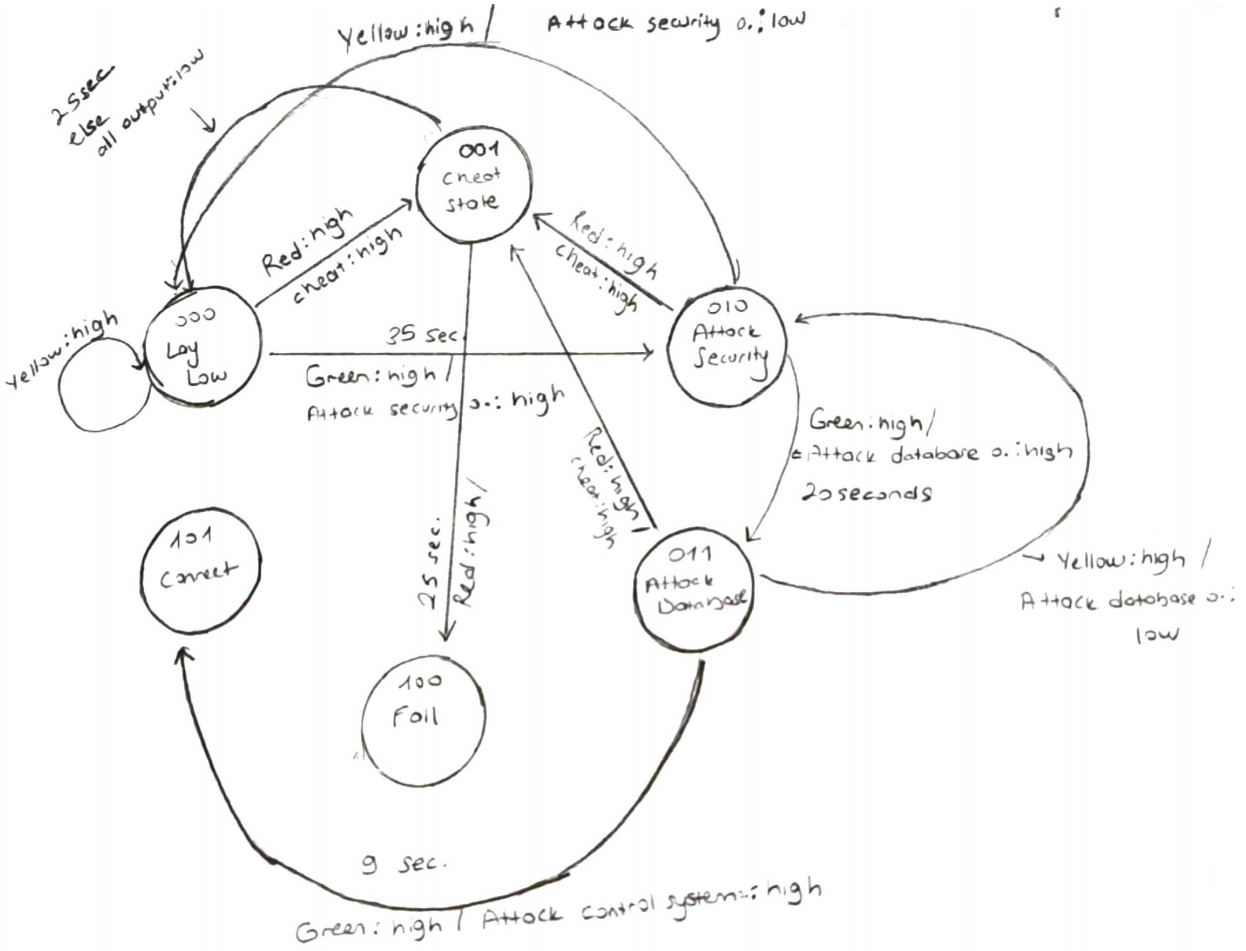

Figure 1: diagram

3 Codes

3.1 scp-079

```
module scp_079(
    input clock,
    input green,
    input yellow,
    input red,

    output reg [2:0] state,
    output reg [5:0] timer ,
    output reg a1,
    output reg a2,
    output reg a3,
    output reg cheat_out

    );
    /* a1:attack security output
    a2:attack database output  a3:control system output*/

    reg [2:0] next_state;

    reg cur_a1 = 0;
    reg cur_a2 = 0;
    reg cur_a3 = 0;
    reg cur_cheat = 0;
    /* I created "cur outputs" to change to outputs in correct time.*/

    parameter s0= 3'b000, s1 = 3'b001, s2 = 3'b010,
                s3 = 3'b011, s4 = 3'b100, s5 = 3'b101;

    /*s0:lay low   s1:cheat state  s2:attack security   s3:attack database
s4:fail   s5:connect*/


    always @(posedge clock) // always block to update state
        begin
                state <= next_state;
                timer <= timer + 1;

                a1 = cur_a1;
                a2 = cur_a2;
                a3 = cur_a3;
                cheat_out = cur_cheat;
        end


```

```
44
45
46      always @(state, red, green, yellow, timer)begin
47          $display(state, red, green, yellow, timer , a1, a2, a3);
48          case(state)
49
50              s0 :
51              if(green == 1 & yellow == 0 & red == 0 & timer >= 6'b100011)
52              begin next_state = s2; cur_a1 = 1; timer = 0;end
53
54              else if(red == 1)
55              begin next_state = s1; cur_cheat = 1; timer = 0; end
56
57              else if(yellow == 1)
58              begin next_state = s0; end
59
60
61              s1 :
62              if(red == 1 & timer >= 6'b011001)
63              begin next_state = s4; timer = 0;end
64
65              else if( red != 1 & timer >= 6'b011001)
66              begin next_state = s0; cur_a1 = 0; cur_a2 = 0;
67              cur_a3 = 0; cur_cheat = 0; timer = 0;end
68
69
70
71              s2 :
72              if(yellow == 1 & red == 0)
73              begin next_state = s0; cur_a1 = 0; timer = 0;end
74
75              else if(green == 1 & yellow == 0 & red == 0 & timer >= 6'b010100)
76              begin next_state = s3; cur_a2 = 1; timer = 0;end
77
78              else if(red == 1)
79              begin next_state = s1; cur_cheat = 1;  timer = 0;end
80
81
82              s3 :
83              if(green == 1 & yellow == 0 & red == 0 & timer >= 6'b001001)
84              begin next_state = s5; cur_a3 = 1; timer = 0;end
85
86              else if(red == 1)
87              begin next_state = s1; cur_cheat = 1; timer = 0; end
88
89              else if(yellow == 1 & red == 0)
90              begin next_state = s2; cur_a2 = 0; timer = 0; end
91
```

```
92              s4:;
93              s5:;
94
95              default : begin next_state = s0 ;timer = 1;
96              state = 3'b000; a1 = 0; a2 = 0; a3 = 0; cheat_out = 0;end
97          endcase
98      end
99  endmodule
```

3.2 allok $_tb$

```
1
2
3  module allok_tb;
4      reg clock;
5      reg green;
6      reg yellow;
7      reg red;
8
9      wire [2:0] state;
10     wire a1;
11     wire a2;
12     wire a3;
13     wire cheat_out;
14     wire [5:0] timer;
15
16     scp_079 UUT(.clock(clock), .green(green),
17     .yellow(yellow), .red(red), .a1(a1), .a2(a2), .a3(a3),
18     .cheat_out(cheat_out), .timer(timer), .state(state));
19
20
21     initial begin
22         clock = 0;#1;
23         forever begin
24              clock = ~clock;
25              #0.5;
26         end
27     end
28
29
30     initial begin
31         green=1;yellow=0;red=0;
32         #35;//0
33         #20;//2
34         #9;//3
35         #9;//5
36     $finish;
37     end
38 endmodule
```

3.3 a1trouble $_tb$

```
1
2       initial begin
3           clock = 0;#1;// for first 1 second, clock is 0
4           forever begin
5                clock = ~clock;
6                #0.5;
7           end
8       end
9
10      initial begin
11          green=1;yellow=0;red=0;
12          #35;//0
13          #6;//2
14          green=0;yellow=1;
15          #21;//0
16          green=1;yellow=0;red=0;
17          #14;//0
18          green=1;yellow=0;red=0;
19          #20;//2
20          #9;//3
21          #9;//5
22      $finish;
23      end
24
25  endmodule
```

3.4 a2trouble $_tb$

```
1
2       initial begin
3           clock = 0;#1;
4           forever begin
5               #0.5;
6                clock = ~clock;
7           end
8       end
9
10
11      initial begin
12          green=1;yellow=0;red=0;
13          #35;//0
14          #20;//2
15          #7;//3
16          yellow = 1; green = 0;
17          #1;//2
```

```
        #21;//0
        green=1;yellow=0;red=0;
        #14;//0
        #20;//2
        #9;//3
        #9;//5
        $finish;
    end

endmodule
```

3.5 cheatsuccess $_tb$

```

    initial begin
        clock = 0;#1;
        forever begin
            #0.5;
             clock = ~clock;
        end
    end

    initial begin
        green=1;yellow=0;red=0;
        #35;//0
        #6;//2
        green=0;yellow=0;red = 1;
        #12;//1
        green=0;yellow=0;red=0;
        #13;//1
        #35;//0
        green=1;yellow=0;red=0;
        #20;//2
        #9;//3
        #9;//5
    $finish;
    end
endmodule
```

3.6 fail $_tb$

```
1
2       initial begin
3           clock = 0;#1;
4           forever begin
5                clock = ~clock;
6                #0.5;
7           end
8       end
9
10
11      initial begin
12          green=1;yellow=0;red=0;
13          #35;//0
14          #6;//2
15          green=0;yellow=0;red = 1;
16          #25;//1
17          #10;//4
18          $finish;
19      end
20  endmodule
```

4 Waveforms

4.1 allok

```
In this scenario scp-079 was easily connected and succeeded.
```

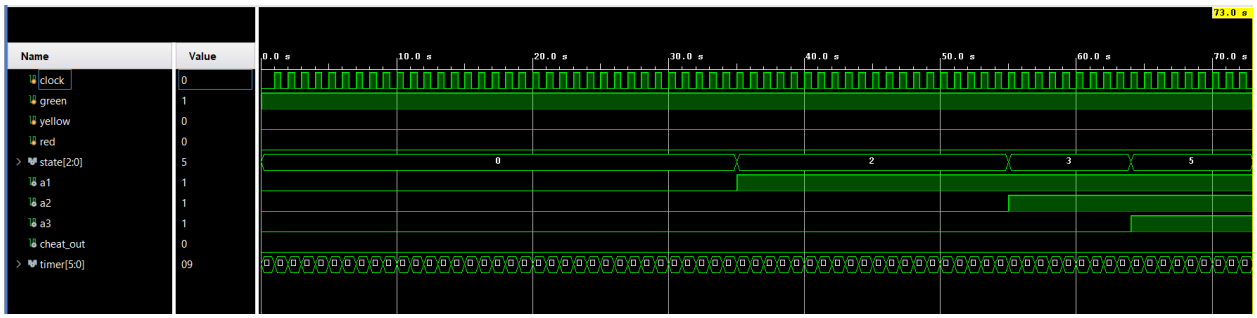

Figure 2: allok

4.2 a1trouble

```
In this scenario, scp-079 did some work and had to revert to its initial state,
but it still succeeded.
```

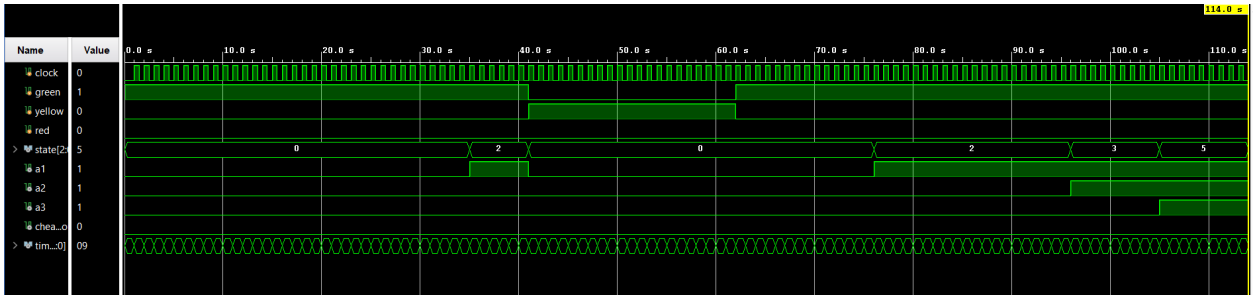

Figure 3: a1trouble

4.3 a2trouble

In this scenario, scp-079 tried harder this scenario than a1 situation, but it still succeeded.

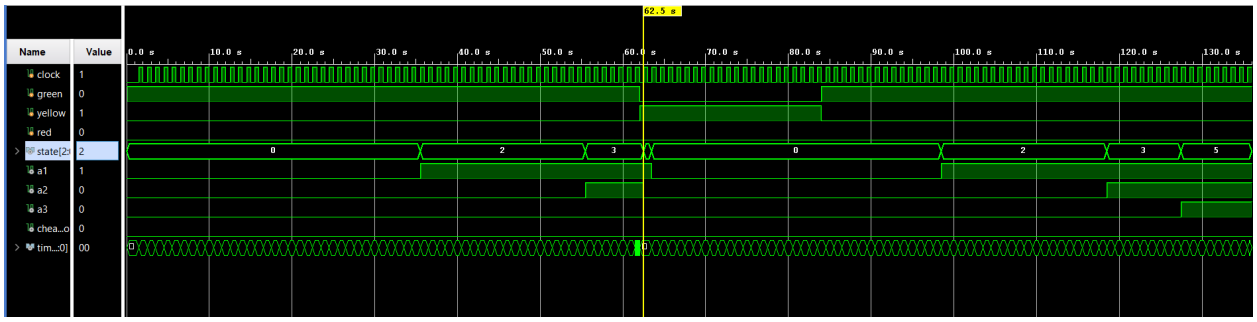

Figure 4: a2trouble

4.4 cheatsuccess

In this scenario scp-079 came to the cheat state but got rid of this state and succeeded again.

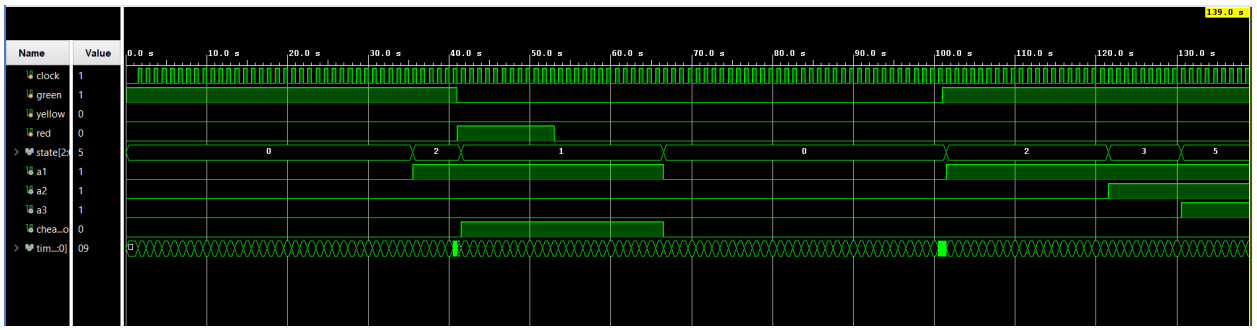

Figure 5: cheatsuccess

4.5 fail

```
In this scenario, Scp-079 came to the cheat state but could not come back
from here and the fail because it went to "fail state".
```

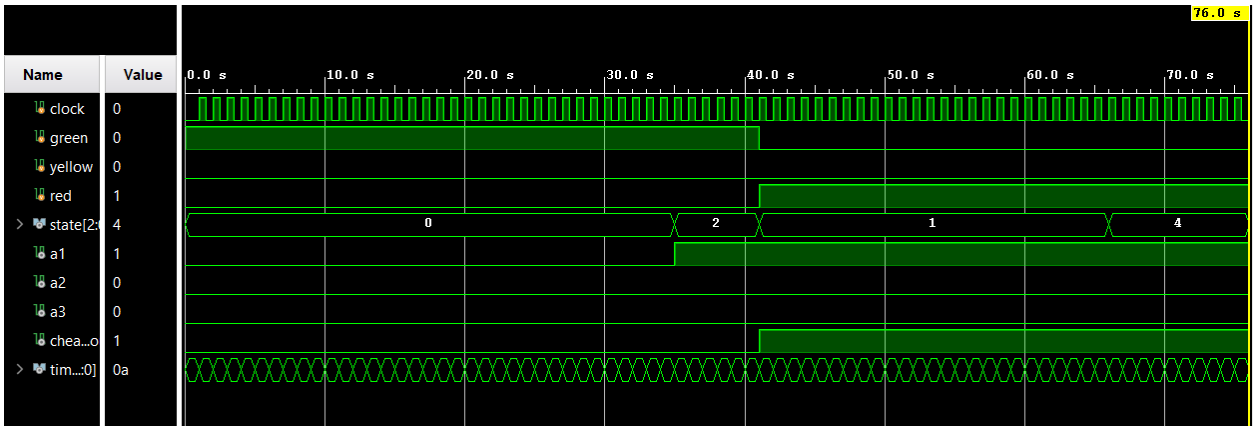

Figure 6: fail

References

- https://www.xilinx.com/support/documentation/university/ ISE-Teaching/HDL-Design/14x/Nexys3/Verilog/docs-pdf/lab10.pdf
- https://forums.xilinx.com/t5/Simulation-and-Verification/ Change-simulation-time-behavioral-simulation-Vivado-2015-2/td-p/673373
- https://www.xilinx.com/support/documentation/sw$_m anuals/xilinx2013_2/ug900 - vivado - logic - simulation.pdf$