ELC 2137 Lab 11: FSM: Guessing Game

Alexander Noll

April 23, 2020

Summary

In this lab a small game of chance was created using a finite state machine(FSM) and a Basys3 board. In order to make the game run better, debouncing was coded for each button input as a way of stabilization. Two different mode of the game were formed by allowing to player to chose between two different play speeds. The object of the game was to press the button corresponding to the currently lit segment on the display. since the normal clock moves degrees higher than a human eye can see, it becomes a game of chance. The clock is slowed to a visible yet fast rate for the easier mode.

Q&A

1. wait1= 220ns; one=245ns; wait0=620ns; zero=645ns; Twait=25ns.
2. Because the next course of action depends what state the system is in.
3. I utilized a Moore machine as the output (y) was defined within each state.

Results

Table 1: *register* Test Results: Fast Mode = (3/10) Success Rate; Slow Mode = 1/2 Success Rate

Test:	1	2	3	4	5	6	7	8	9	10
Fast Mode	lose	win	lose	lose	lose	win	lose	lose	lose	win
Segment	bottom	bottom	bottom	top	top	left	left	bottom	right	top
Slow Mode	win	lose	win	lose	lose	win	lose	lose	win	win
Segment	top	left	bottom	left	top	right	right	top	top	bottom

0.1 Test Photos

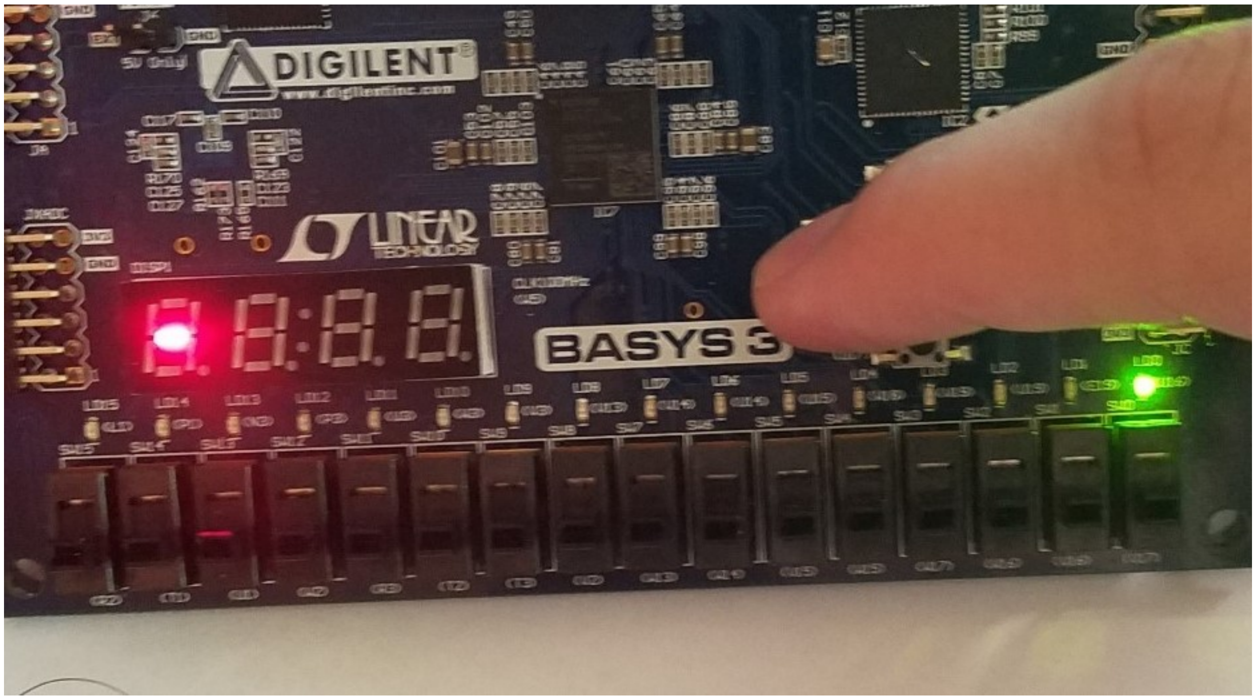

Figure 1: Fast Mode Test 1

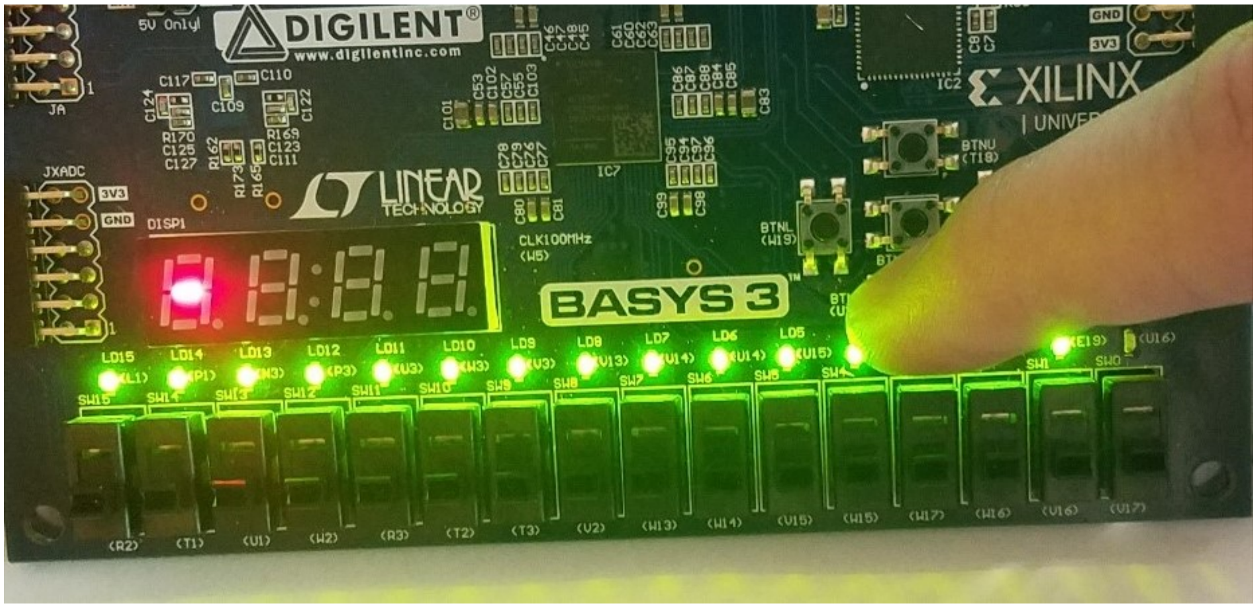
Figure 2: Fast Mode Test 2

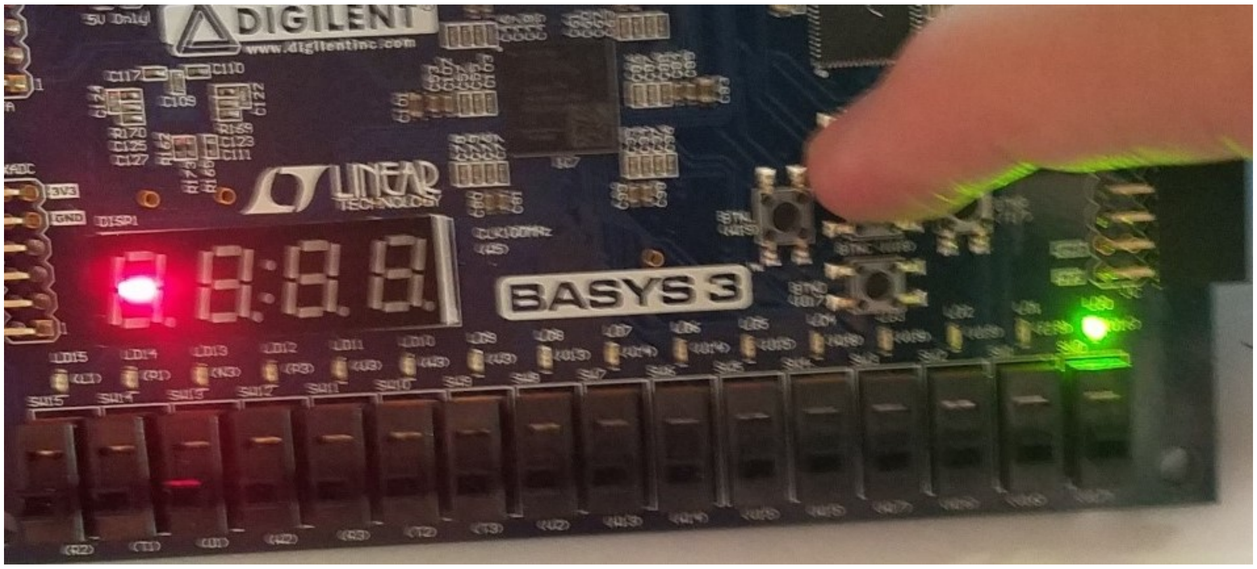
Figure 3: Fast Mode Test 3

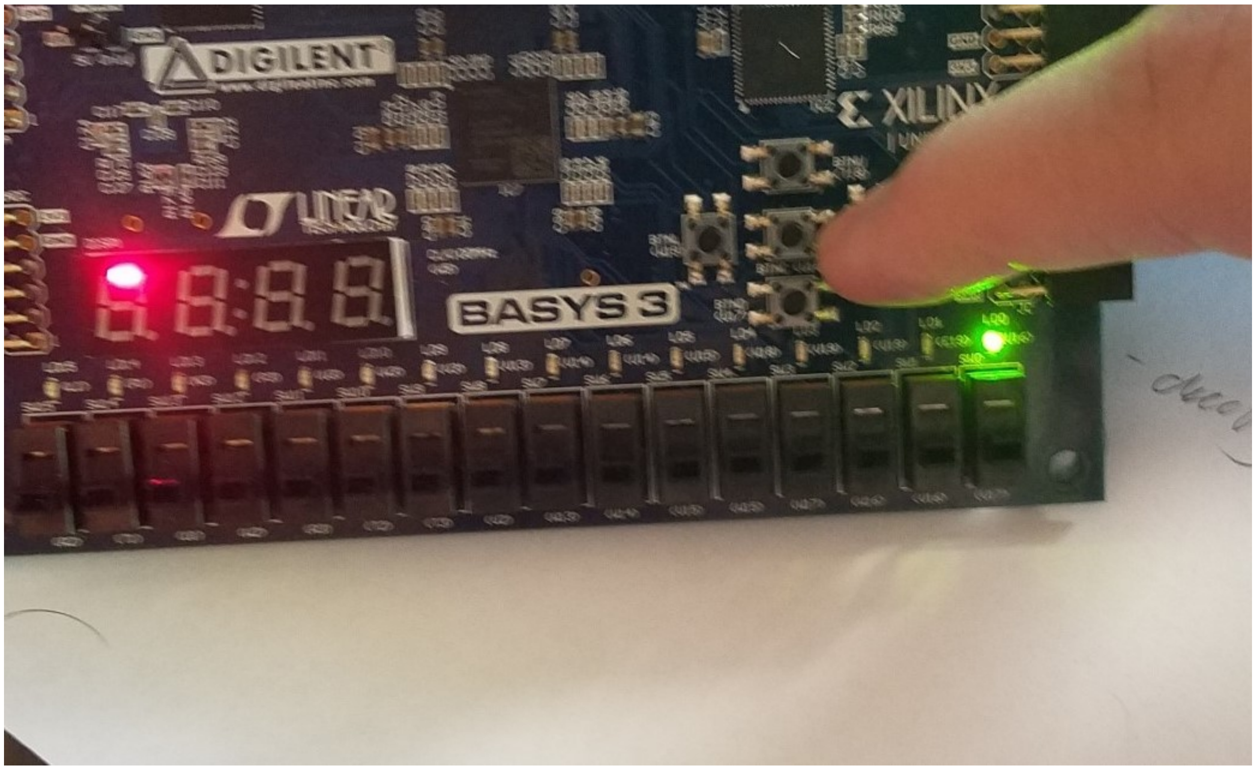

Figure 4: Fast Mode Test 4

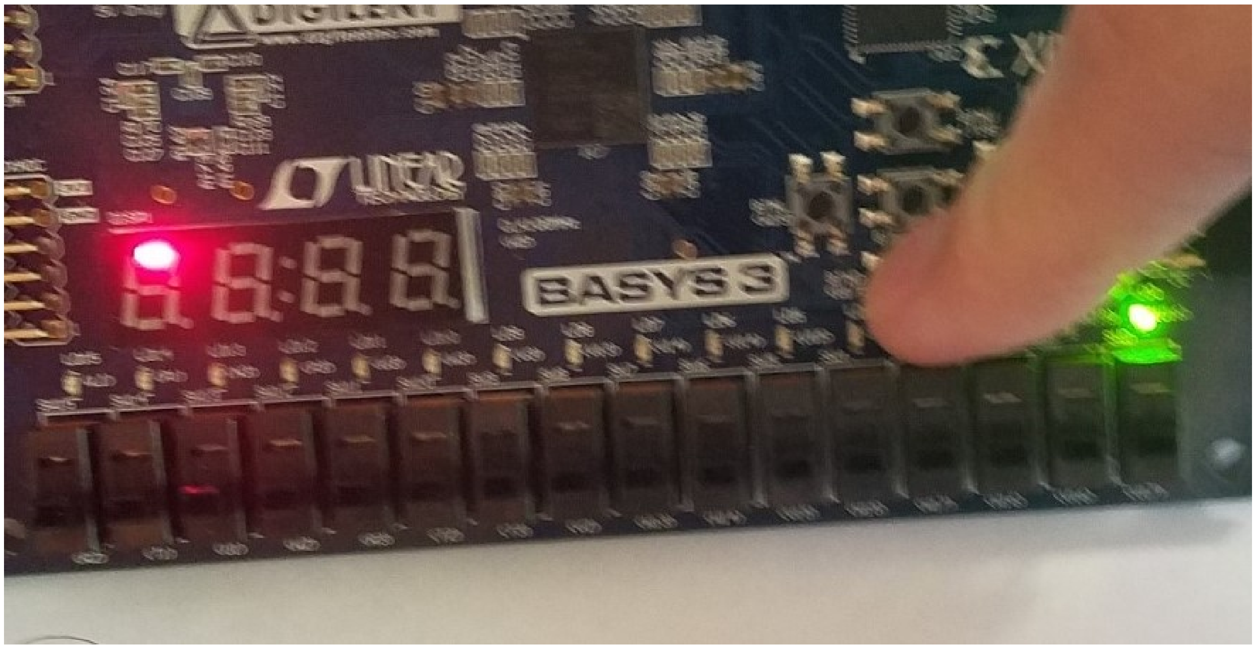

Figure 5: Fast Mode Test 5

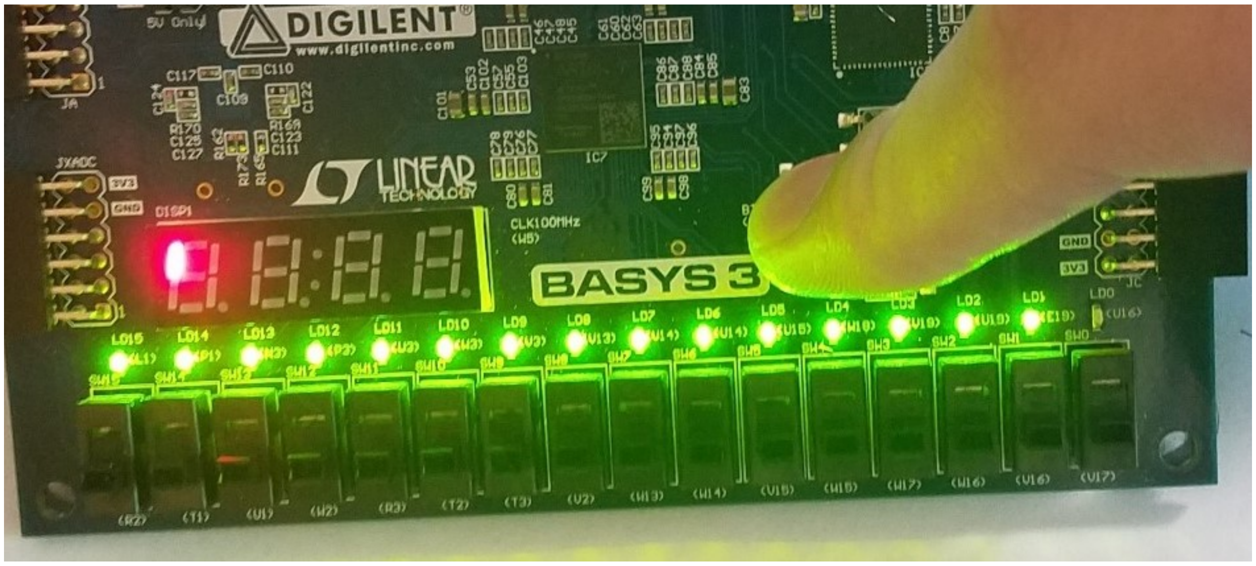

Figure 6: Fast Mode Test 6

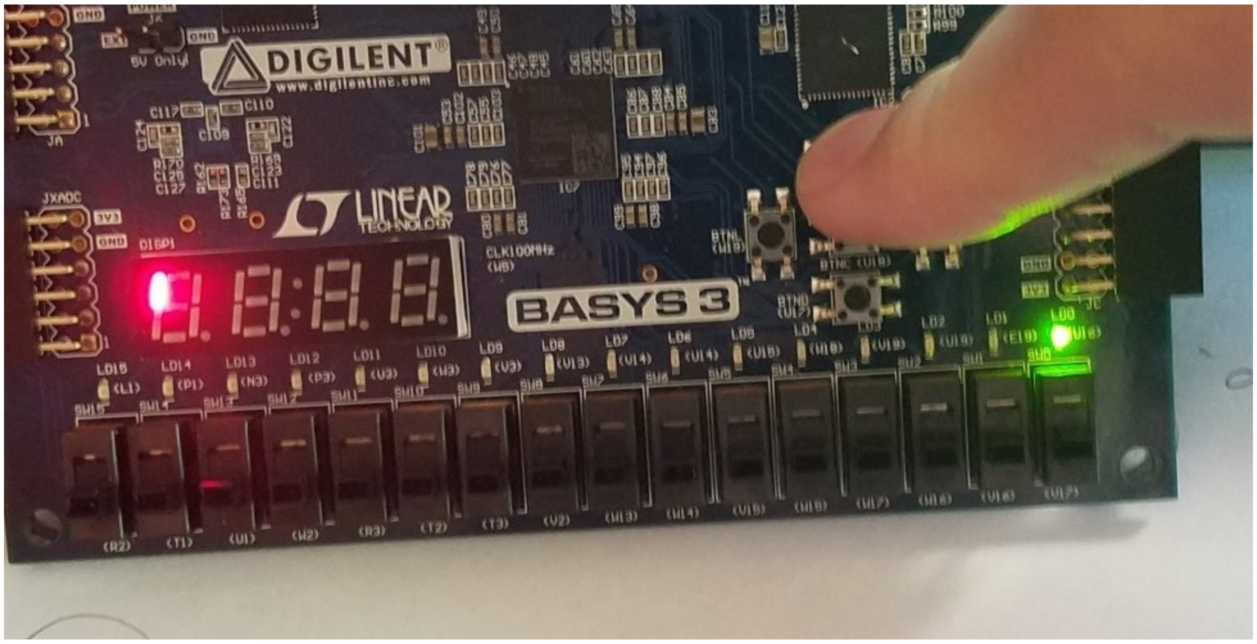

Figure 7: Fast Mode Test 7

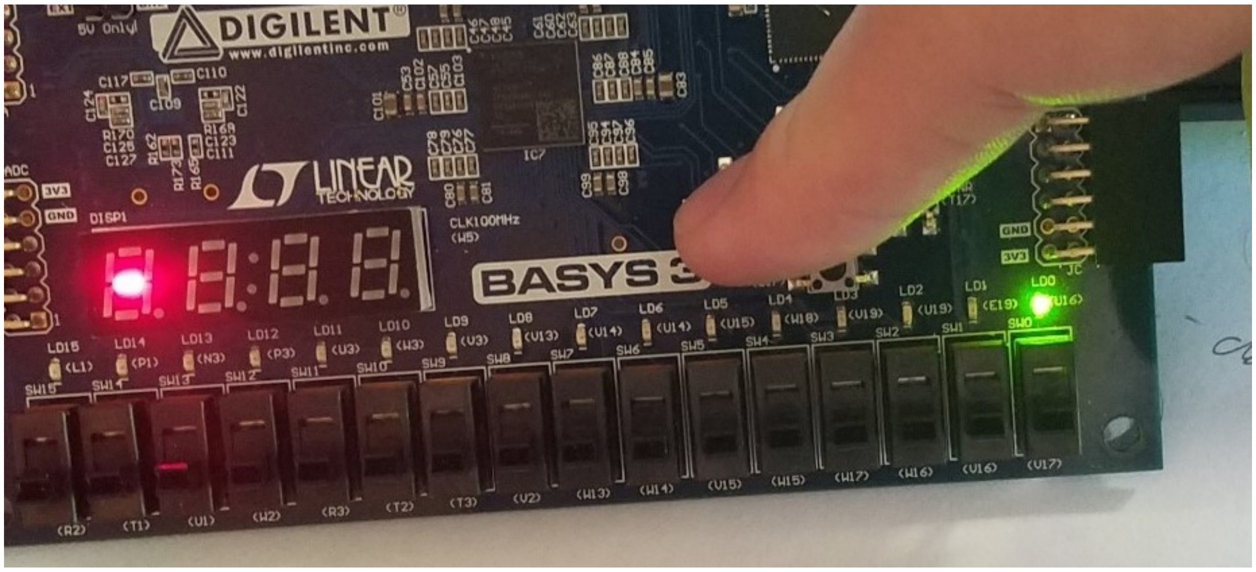

Figure 8: Fast Mode Test 8

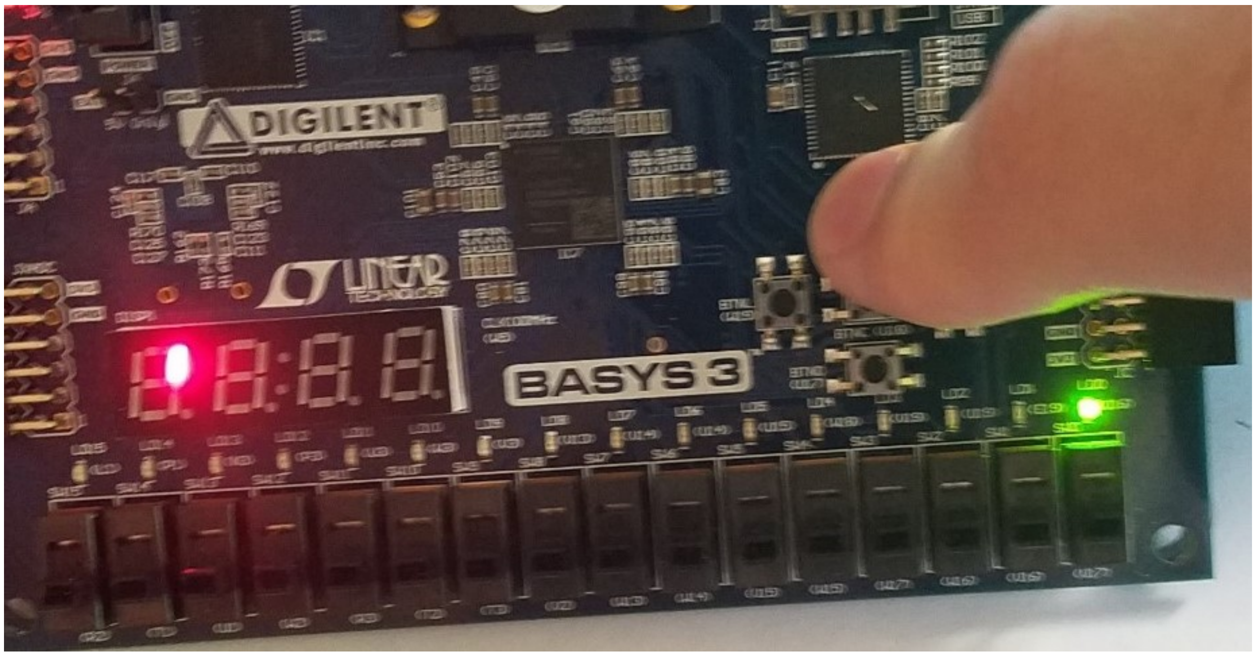

Figure 9: Fast Mode Test 9

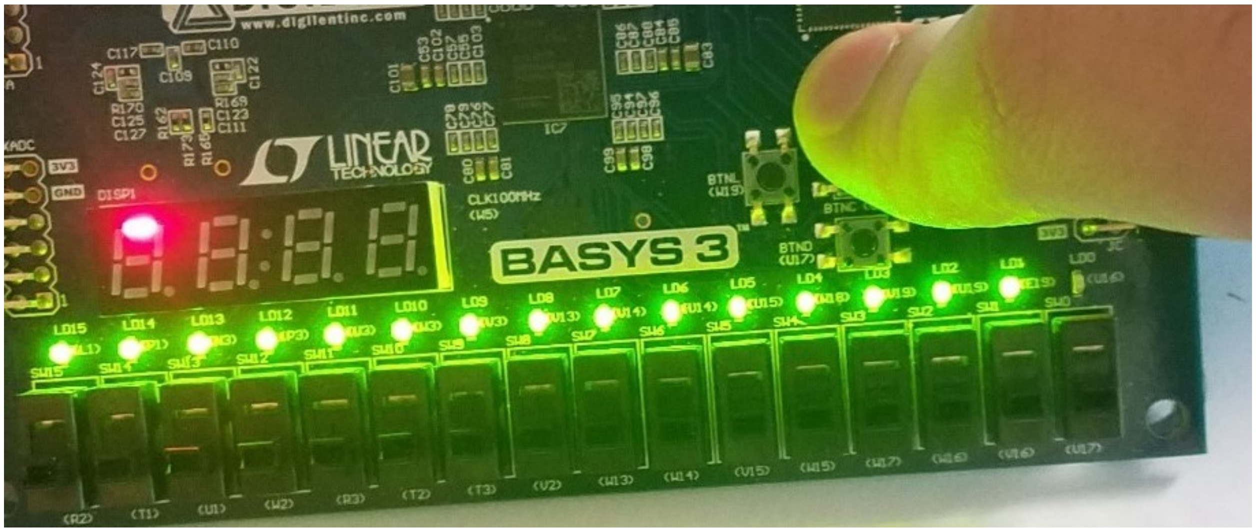
Figure 10: Fast Mode Test 10

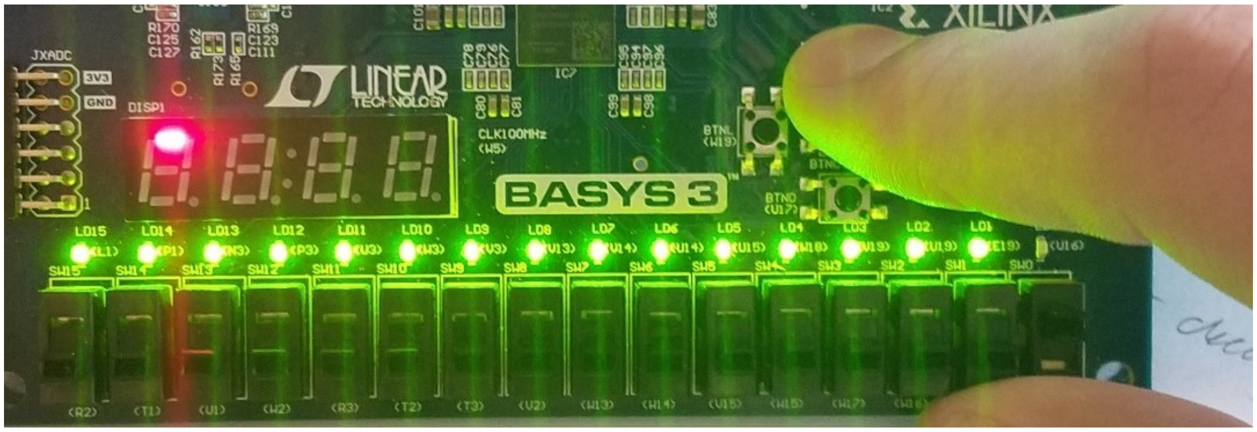
Figure 11: Slow Mode Test 1

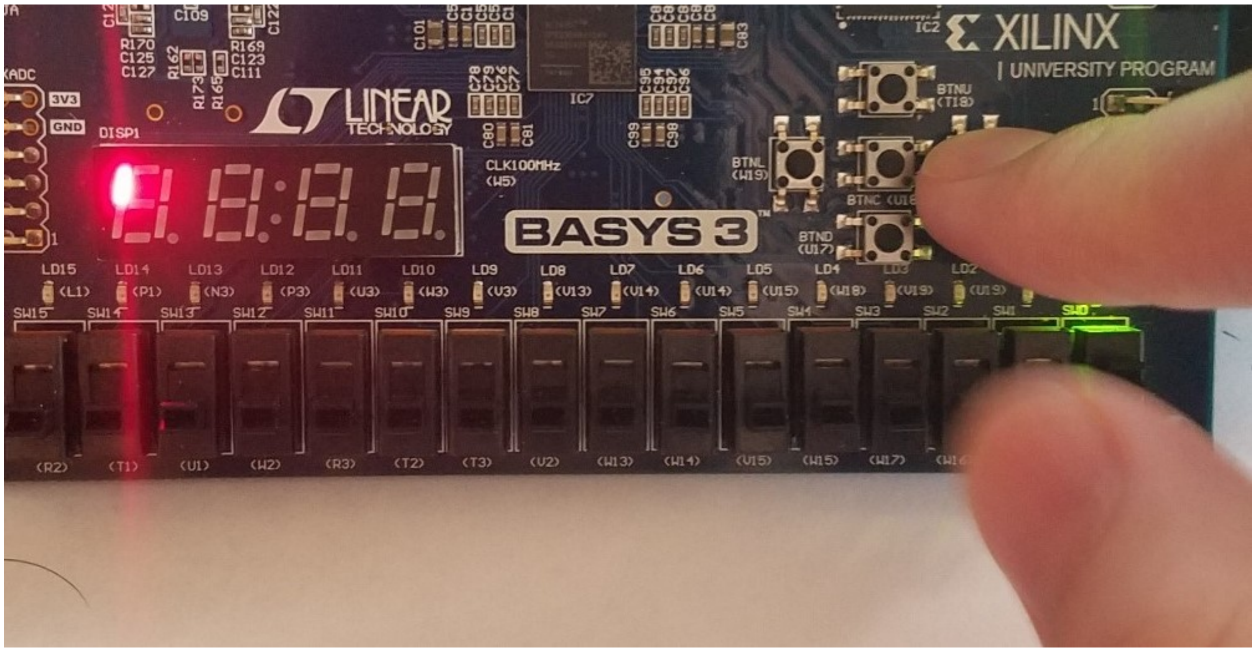

Figure 12: Slow Mode Test 2

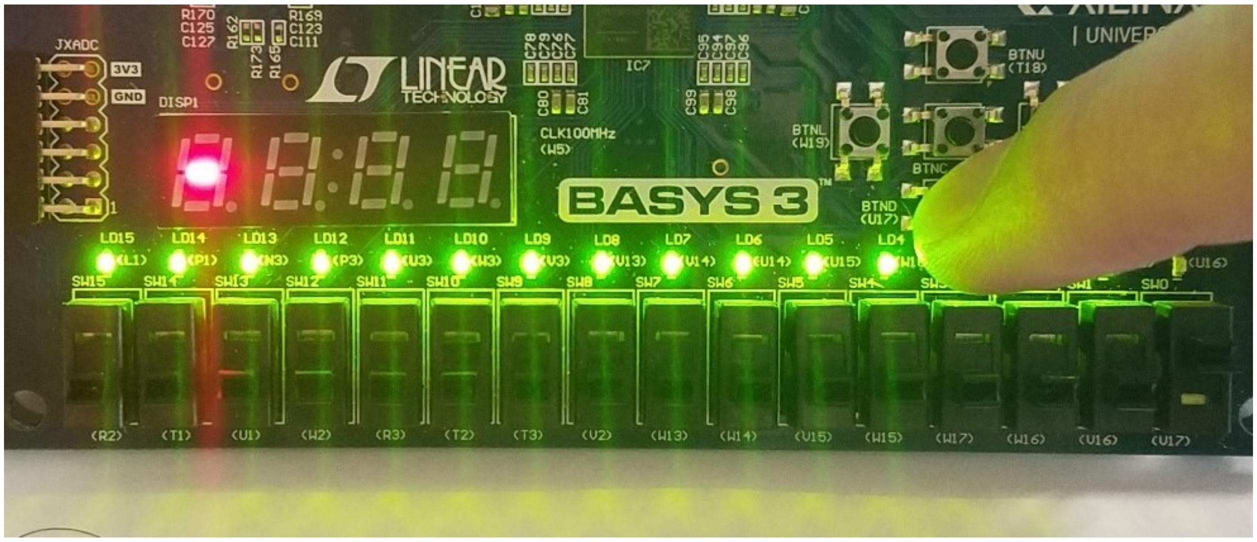

Figure 13: Slow Mode Test 3

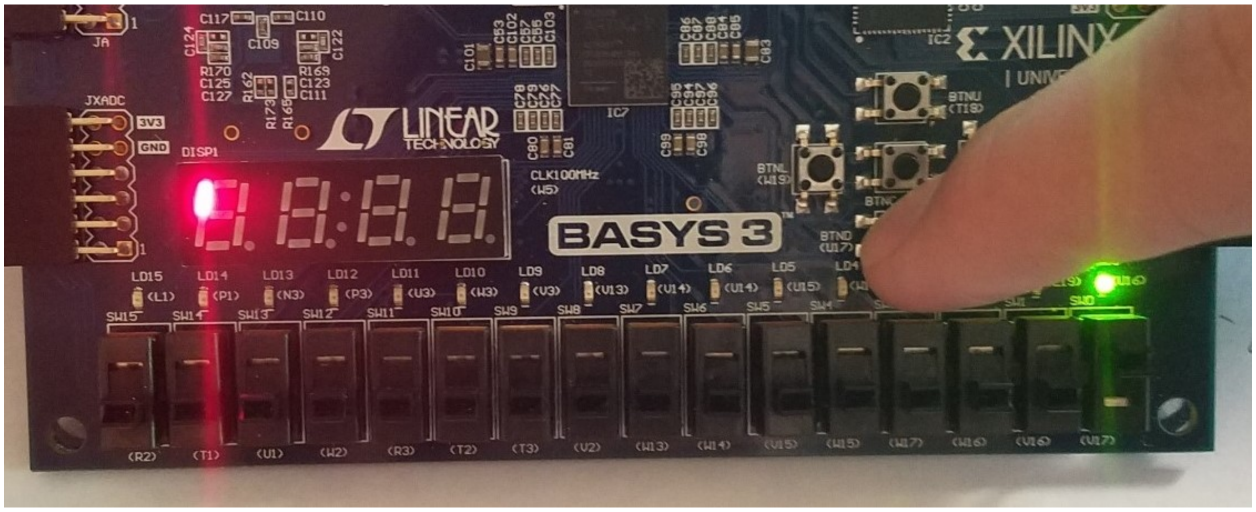
Figure 14: Slow Mode Test 4

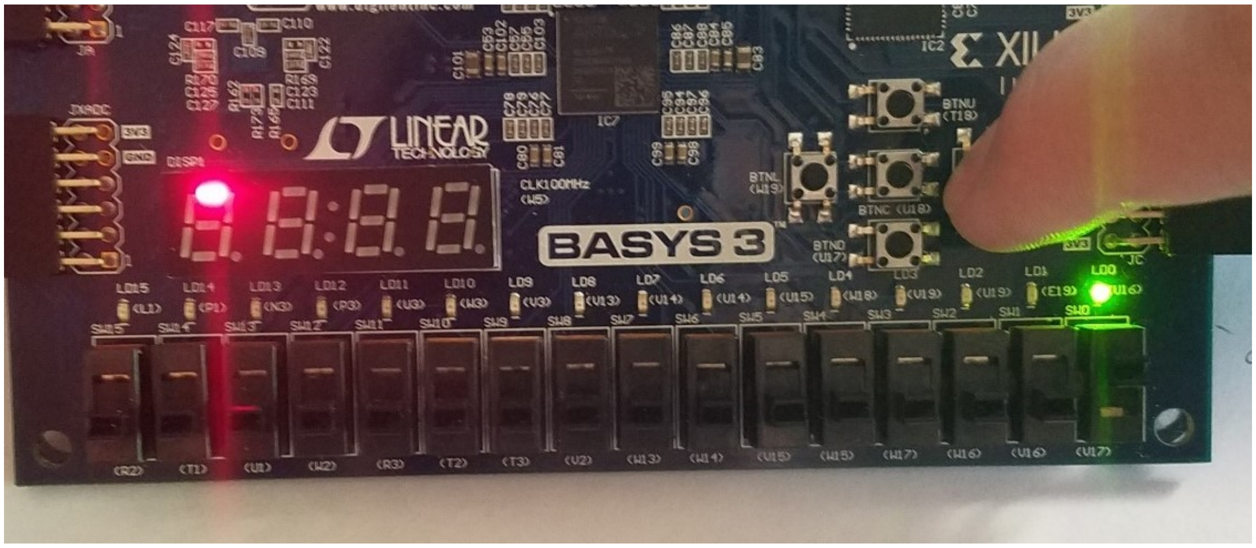
Figure 15: Slow Mode Test 5

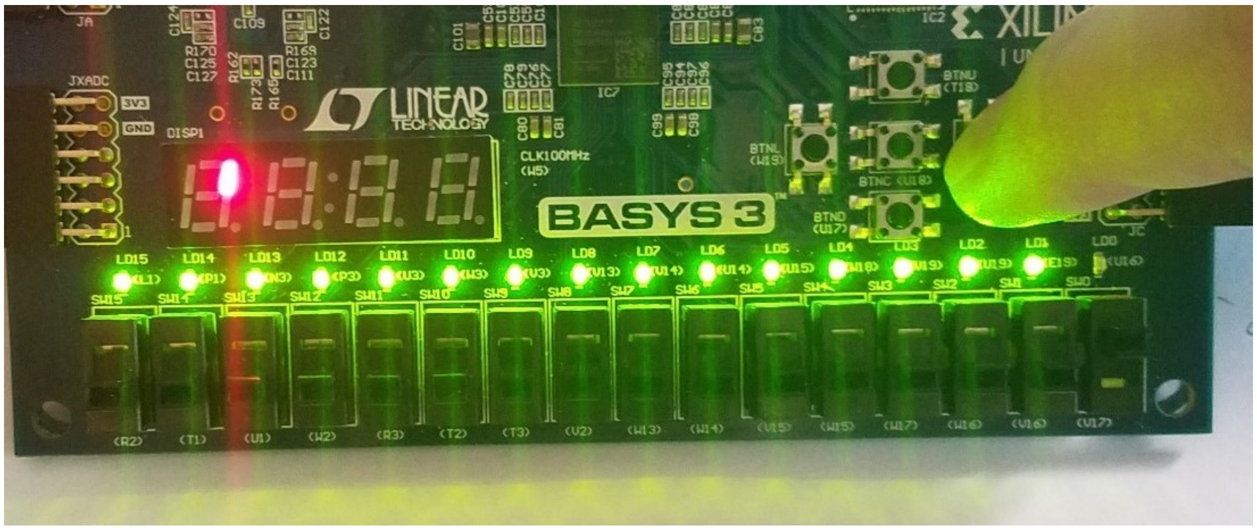

Figure 16: Slow Mode Test 6

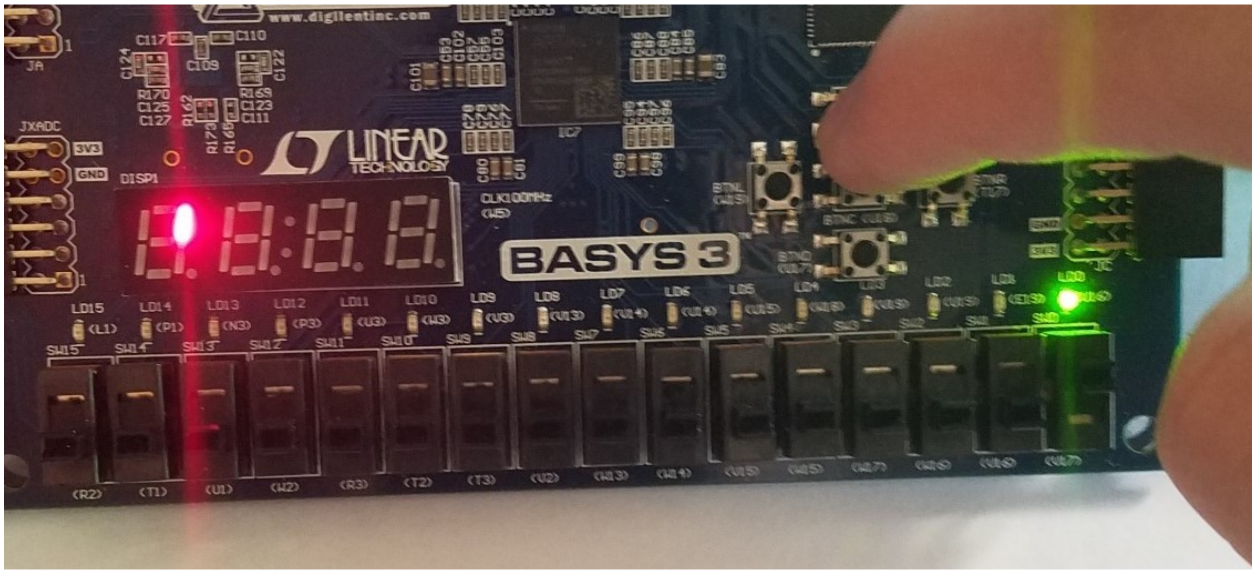

Figure 17: Slow Mode Test 7

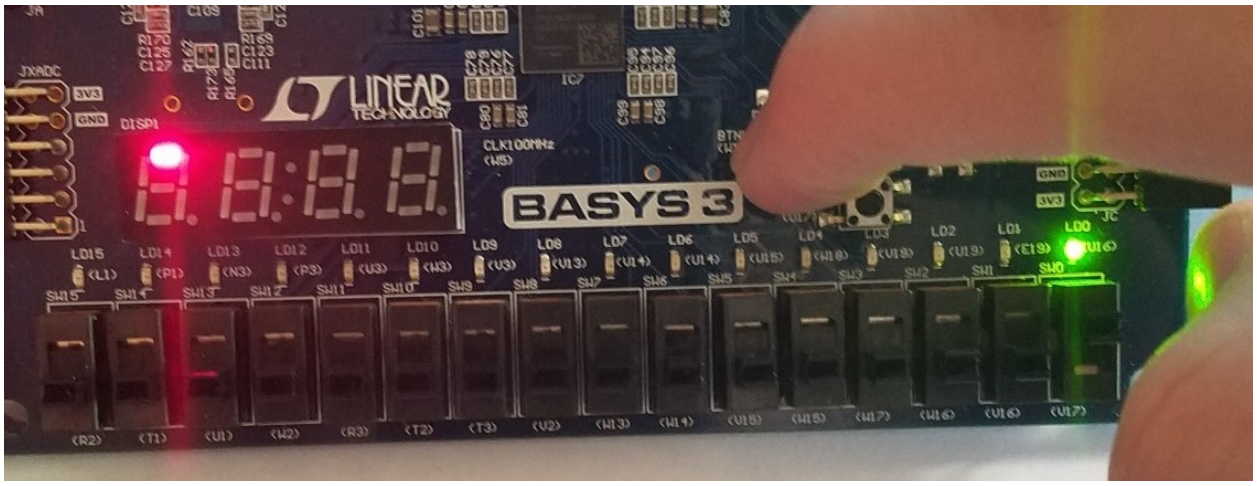

Figure 18: Slow Mode Test 8

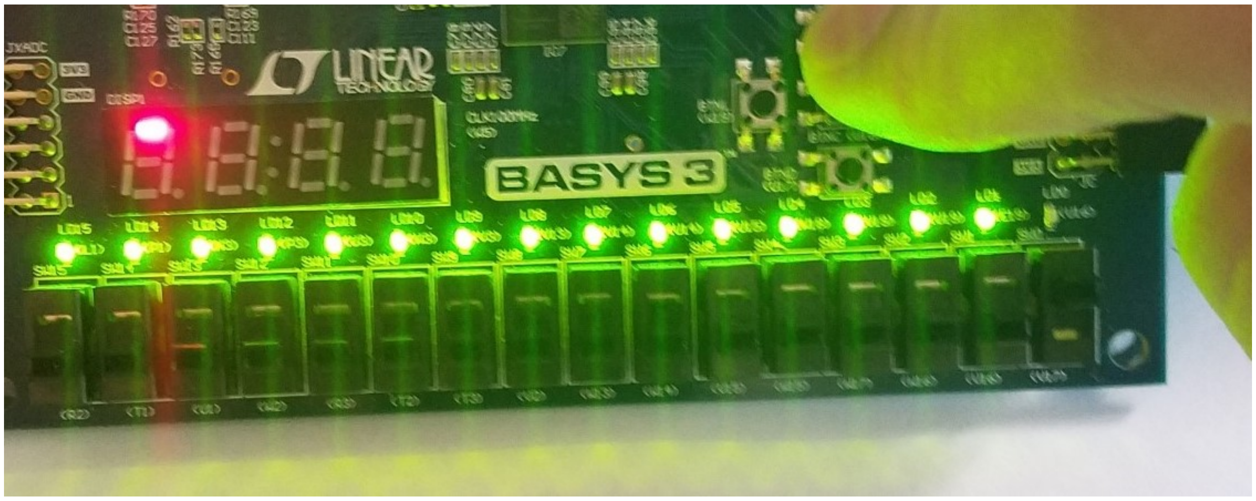

Figure 19: Slow Mode Test 9

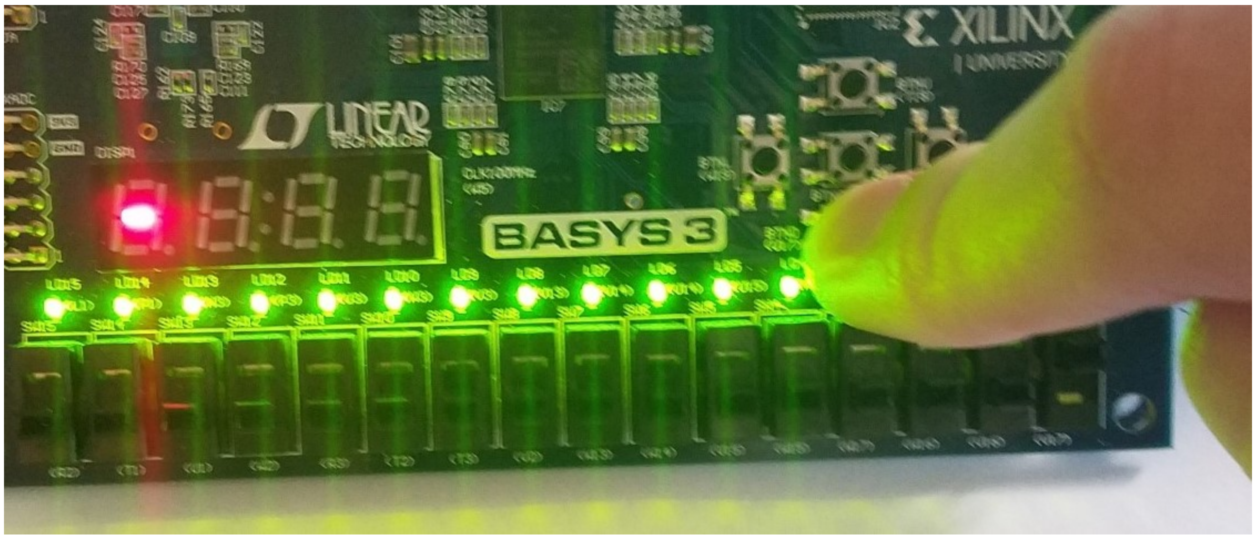

Figure 20: Slow Mode Test 10

0.2 Waveforms

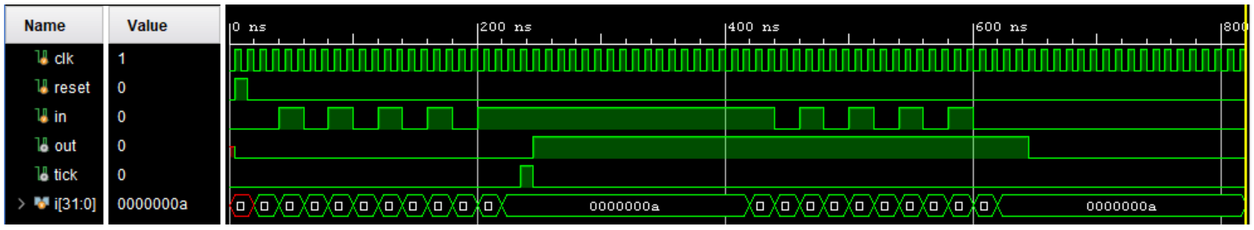

Figure 21: Debounce Test Waveform

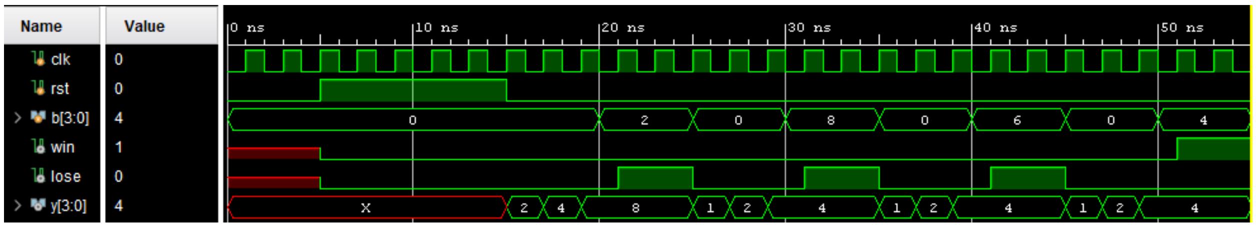

Figure 22: Finite Test Machine Test Waveform

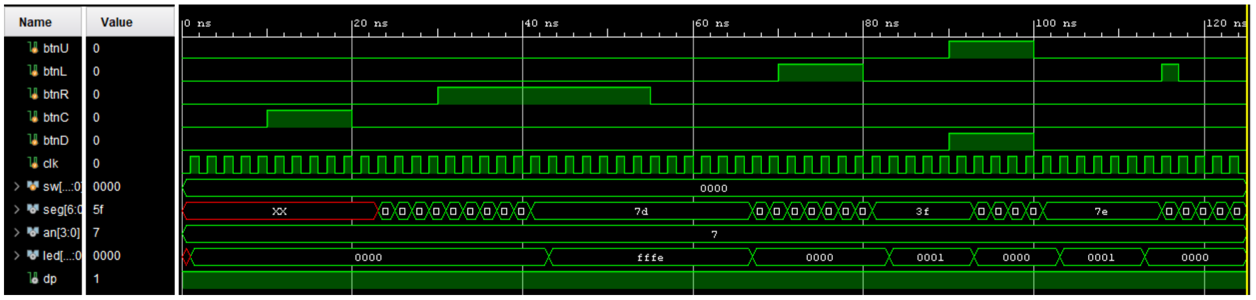

Figure 23: Game Test Waveform

Code

Listing 1: Verilog Guessing Game Source

```
module guessing_game(
    input btnU, btnL, btnR, btnC, btnD, clk,
    input [15:0] sw,
    output reg [6:0] seg,
    output reg [3:0] an,
    output reg [15:0] led,
    output reg dp);

    reg [3:0] b; reg [3:0] y; reg [22:0] Cclk; reg [4:0] tick;
    reg win, lose, Gclk, reset, Mode;

    assign dp=1;
    assign reset=btnC;
    assign Mode=sw[0];
    assign an=4'b0111;

    debounce #(.N(1)) debU (.clk(clk), .reset(reset), .in(btnU), .out(b
       [0]),
        .tick(tick[0]));
    debounce #(.N(1)) debL (.clk(clk), .reset(reset), .in(btnL), .out(b
       [1]),
        .tick(tick[1]));
    debounce #(.N(1)) debR (.clk(clk), .reset(reset), .in(btnR), .out(b
       [2]),
        .tick(tick[2]));
    debounce #(.N(1)) debD (.clk(clk), .reset(reset), .in(btnD), .out(b
       [3]),
        .tick(tick[3]));

    counter  #(.N(23)) EasyMode (.clk(clk), .en(dp), .rst(reset), .count(
       Cclk), .tick(tick[4]));

    parameter in0=0;
    parameter in1=1;
    parameter in2=4'b0010;
    parameter in3=4'b0100;
    parameter in4=4'b1000;

    always @*
    begin
        case(Mode)
            default: Gclk=clk;
            in1: Gclk=tick[4];
        endcase
    end

guess_FSM GuessGame(.clk(Gclk),.reset(reset),.b(b),.win(win),.lose(lose),.
   y(y));

    always @(posedge Gclk)
    begin
```

```
        case(y)
        in1: seg=7'b1111110;
        in2: seg=7'b1011111;
        in3: seg=7'b1111101;
        in4: seg=7'b0111111;
        endcase
    end

    always @(posedge Gclk)
    begin
        if (win==0&&lose==1)
            led=16'b0000000000000001;
        else if (lose==0&&win==1)
            led=16'b1111111111111110;
        else
            led=16'b0000000000000000;
    end
endmodule
```

Listing 2: Verilog Guessing Game Test Bench

```
module Game_Test();

reg btnU,btnL,btnR,btnC,btnD,clk;
reg [15:0] sw;
wire [6:0] seg;
wire [3:0] an;
wire [15:0] led;
wire dp;


guessing_game GameTest(.clk(clk),.btnC(btnC),.btnL(btnL),.btnR(btnR),.btnD
   (btnD),.sw(sw),.seg(seg),.an(an),.led(led),.dp(dp));

always
begin
    #1 clk = ~clk;
end



    initial begin
        clk=0; btnC=0; btnU=0; btnL=0; btnR=0; btnD=0; sw=16'
           b0000000000000000; #10;
        btnC=1; #15;
        btnC=0; #15;
        btnR=1; #25;
        btnR=0; #15;
        btnL=1; #15;
        btnL=0; #15;
        btnD=1; btnU=1; #15;
        btnD=0; btnU=0; #15;
        btnL=1; #2;
        btnL=0; #8;
        $finish;
        end

endmodule
```

Listing 3: Verilog Finite State Machine Source

```
module guess_FSM(
    input [3:0] b,
    input clk, reset,
    output reg [3:0] y,
    output reg win, reg lose);

reg [2:0] present_state, next_state;
reg [3:0] y0,y1,y2,y3;

assign y0=4'b0001;
assign y1=4'b0010;
assign y2=4'b0100;
assign y3=4'b1000;

parameter S0=3'b000;
parameter S1=3'b001;
parameter S2=3'b010;
parameter S3=3'b011;
parameter Swin=3'b100;
parameter Slose=3'b101;


always @(posedge clk)

begin
    case(present_state)
        S0:
        begin
            if (b[3]||b[2]||b[1])
                next_state=Slose;
            else if (~b[3]&&~b[2]&&~b[1]&&b[0])
                next_state=Swin;
            else
                next_state=S1;
        end
        S1:
        begin
            if (b[3]||b[2]||b[0])
                next_state=Slose;
            else if (~b[3]&&~b[2]&&b[1]&&~b[0])
                next_state=Swin;
            else
                next_state=S2;
        end
        S2:
        begin
            if (b[3]||b[0]||b[1])
```

```
                next_state=Slose;
            else if (~b[3]&&b[2]&&~b[1]&&~b[0])
                next_state=Swin;
            else
                next_state=S3;
        end
        S3:
        begin
            if (b[0]||b[2]||b[1])
                next_state=Slose;
            else if (b[3]&&~b[2]&&~b[1]&&~b[0])
                next_state=Swin;
            else
                next_state=S0;
        end
        Swin:
        begin
            if (b[3]||b[2]||b[1]||b[0])
                next_state=Swin;
            else
                next_state=S0;
        end
        Slose:
        begin
            if (b[3]||b[2]||b[1]||b[0])
                next_state=Slose;
            else
                next_state=S0;
        end
    endcase

end




always @(posedge clk, posedge reset)
begin
    if (reset == 1'b1)
        begin
        present_state=S0;
        win=0;
        lose=0;
        end
    else
    begin
        present_state = next_state;
        begin
    if (present_state==S0)
        begin
        y<=y0;
        win=1'b0;
        lose=1'b0;
        end
    else if (present_state==S1)
        y<=y1;
    else if (present_state==S2)
        y<=y2;
    else if (present_state==S3)
        y<=y3;
    else if (present_state==Swin)
        begin
        win=1;
        next_state=present_state;
        end
    else if (present_state==Slose)
        begin
        lose=1;
        next_state=present_state;
        end
    end
    end
end


endmodule
```

Listing 4: Verilog Finite State Machine Test Bench

```
module guess_Test();

reg clk, rst;
reg [3:0] b;
wire win, lose;
wire [3:0] y;

guess_FSM GuessTest(.clk(clk),.reset(rst),.b(b),.win(win),.lose(lose),.y(y
   ));

always
begin
    #1 clk = ~clk;
end



    initial begin
        clk=0; rst=0; b=4'b0000; #5;
        rst=1; #10;
        rst=0; #5;
        b=4'b0010; #5;
        b=4'b0000; #5;
        b=4'b1000; #5;
        b=4'b0000; #5;
        b=4'b0110; #5;
        b=4'b0000; #5;
        b=4'b0100; #5;
        $finish;
        end
endmodule
```