VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Fall Semester 2021-2022

ECE5017 - Digital Design with FPGA ELA

M.Tech VLSI Design

School of Electronics Engineering

Vellore Institute of Technology

Name: Shreyas S Bagi

Register Number: 21MVD0086

Slot: L5+L6

Date:31/12/2021

Lab Task 03

Behavioural Modelling – LIFT CONTROLLER

Aim: Construct a sequence detector that accepts a binary number entered one bit at a time, most significant bit first, and indicates the output with a LED light if the sequence are of {011,110}. Implement this design on Altera DE2-115 FPGA Kit using Quartus II.

Objective:

- To design the Lift Controller using Behavioural Modelling.
- The Lift is having UP and DOWN direction button for Lift. If he wants to go up he has to make switch UP and Down Pull UP Switch down.
- There are four switches which uses as input for floor reading.

Real Time Lift Switches and Control:

Figure 1.1 The Lift switch having UP and DOWN control

Date:31/12/2021

Block Diagram:

Real Time Implementation :

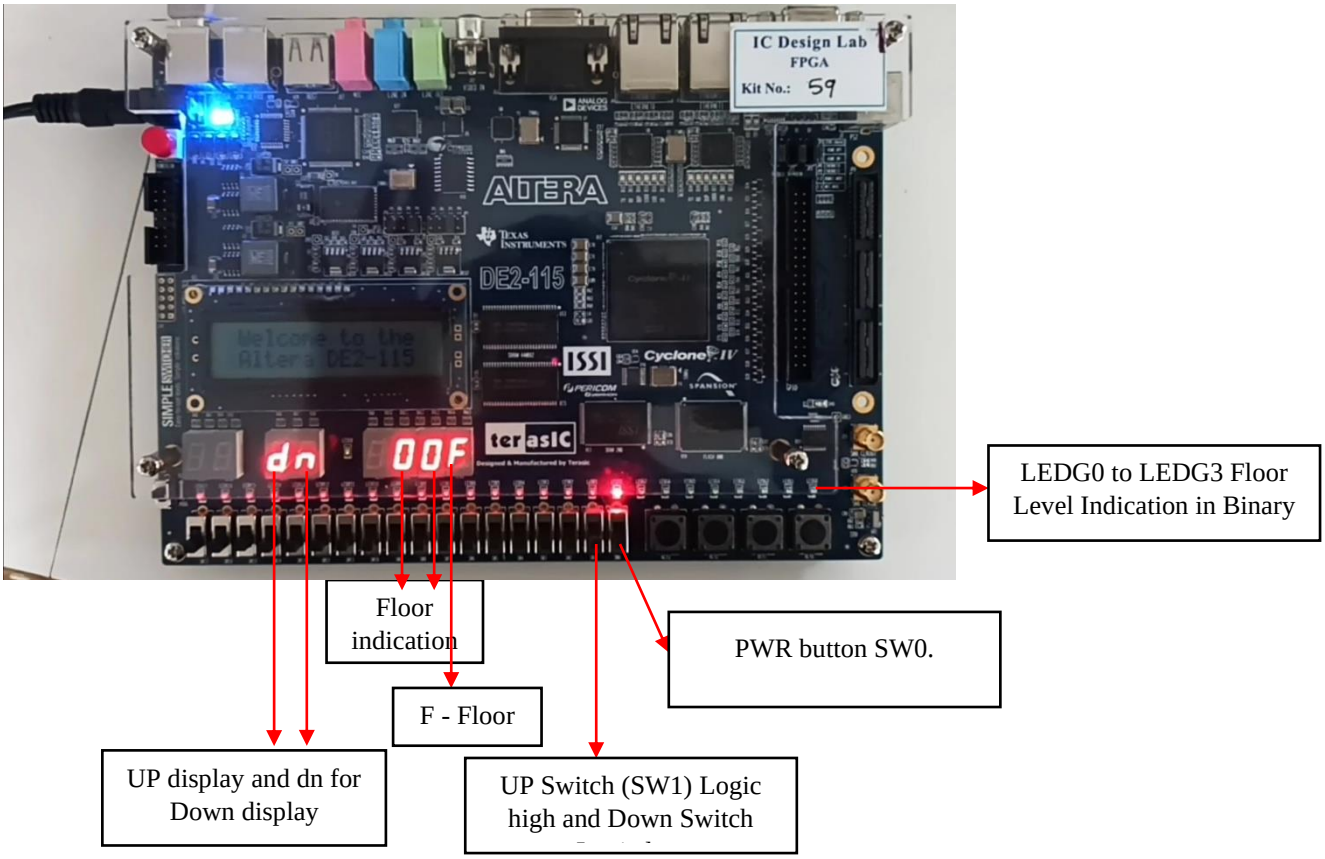


Figure 1.2 The detailed explanation of LIFT Controller implemented in DE2-115 kit.

Date:31/12/2021

Software and Hardware Details:

For design Functional Simulation: Modelsim-INTEL FPGA STARTER EDITION.

For Implementation of the Function using ALTERA Quartus Prime and DE2-115 FPGA Kit (Cyclone IV E: EP4CE115F29C7).

Verilog Code :

```
module Lift_Controller (clk, cnt, IP_floor, UP, PWR, HEXD0, HEXD1, HEXD2, HEXD5, HEXD4,clk_div);

input clk, UP, PWR;

input [3:0] IP_floor;

output clk_div;

output reg [6:0] cnt;

output [6:0] HEXD0, HEXD1, HEXD2, HEXD5, HEXD4;

wire [3:0] OUT,OUT1,OUT2;

reg [6:0] count;

wire D=4'b0000;

reg D1,D2,D3;

reg SEL,SEL1,SEL2;

wire [3:0] IN0,IN1,IN01,IN11,IN02,IN12;

assign IN0 = 4'b0000;

assign IN1 = 4'b0001;

seven_segmentdisp_F G1 (HEXD0,D);

seven_segmentdisp_U G4 (HEXD5,D1);

seven_segmentdisp_P G5 (HEXD4,D2);

seven_segmentdisp G2 (HEXD1,cnt);

Clock_Division G3 (clk,PWR,clk_div);

Mux_2to1 GM1 (SEL,OUT,IN0,IN1);

seven_segmentdisp_1 G12 (HEXD2,OUT);


always @(posedge clk_div)

begin

if(PWR)
```

Date:31/12/2021

```
cnt <= count;
else
cnt <= count;
end
always @(posedge clk_div)
begin
if(count < 10)
SEL = 0;
else
SEL = 1;
end
always @(posedge clk_div)
begin
if(UP)
        begin
        D1 =4'b0000;
        D2 =4'b0000;
        // Doing UP Counting sequence
        if(count < IP_floor)
        begin
                count <= count+1;
        end
        else if (count == IP_floor)
        begin
                count <= count;
        end
        else if (count == 10)
        begin
          count <= 10;
        end
```

Date:31/12/2021

```
        else if (count > 10)
        begin
          count <= count;
        end
        else
                count <= count;
        end
else
        begin
        D1=4'b0001;
        D2=4'b0001;
        // Doing DOWN Count sequence
        if(count > IP_floor)
                count <= count-1;
        else if(count == IP_floor)
                count <= count;
        else if (count < 0)
                count <= 7'b000000;
        else
                count <= 0;
        end
end
endmodule

module seven_segmentdisp_F (HEX,D);
input [3:0] D;
output reg [6:0] HEX;
always @(D)
begin
casex (D)
```

Date:31/12/2021

```
4'b0000: HEX   <= 7'b0001110;
default : HEX  <= 7'b0000000;
endcase
end
endmodule

module seven_segmentdisp_U (HEX,D);
input [3:0] D;
output reg [6:0] HEX;
always @(D)
begin
casex (D)
4'b0000: HEX   <= 7'b1000001;
4'b0001: HEX   <= 7'b0100001;
4'b0010: HEX   <= 7'b1111111;
default : HEX  <= 7'b0000000;
endcase
end
endmodule
module seven_segmentdisp_P (HEX,D);
input [3:0] D;
output reg [6:0] HEX;
always @(D)
begin
casex (D)
4'b0000: HEX   <= 7'b0001100;
4'b0001: HEX   <= 7'b0101011;
4'b0010: HEX   <= 7'b1111111;
default : HEX  <= 7'b0000000;
endcase
```

Date:31/12/2021

```
end
endmodule

module seven_segmentdisp_1 (HEX,D);
input [3:0] D;
output reg [6:0] HEX;
always @(D)
begin
case (D)
4'b0000:   HEX <= 7'b1000000;
4'b0001:   HEX <= 7'b1111001;
default : HEX  <= 7'b1111111;
endcase
end
endmodule

module Clock_Division(CLK, reset, LED);
input CLK, reset;
output reg LED = 1'b0;
reg [25:0] CLK_DIV;
always @(posedge CLK)
begin
if(CLK_DIV == 26'b1011_1110_1011_1100_0010_0000_00)
begin
if (reset)
LED <= 0;
else
LED <= ~LED;
CLK_DIV <= 26'b0;
end
```

Date:31/12/2021

```
else
CLK_DIV = CLK_DIV + 1;
end
endmodule

module Mux_3to1 (SEL,OUT,IN0,IN1,IN2);
input SEL;
input [3:0] IN0,IN1,IN2;
output reg [3:0] OUT;
always @(SEL)
begin
if(SEL)
OUT = IN1;
else if(SEL==2)
OUT = IN2;
else
OUT = IN0;
end
endmodule
module Mux_2to1 (SEL,OUT,IN0,IN1);
input SEL;
input [3:0] IN0,IN1;
output reg [3:0] OUT;
always @(SEL)
begin
if(SEL)
OUT = IN1;
else
OUT = IN0;
end
```

Date:31/12/2021

```
endmodule


module seven_segmentdisp (HEX,D);
input [3:0] D;
output reg [6:0] HEX;
wire extra_one;
always @(D)
begin
casex (D)
4'b0000: HEX <= 7'b1000000;
4'b0001: HEX <= 7'b1111001;
4'b0010: HEX <= 7'b0100100;
4'b0011: HEX <= 7'b0110000;
4'b0100: HEX <= 7'b0011001;
4'b0101: HEX <= 7'b0010010;
4'b0110: HEX <= 7'b0000010;
4'b0111: HEX <= 7'b1111000;
4'b1000: HEX <= 7'b0000000;
4'b1001: HEX <= 7'b0011000;
4'b1010: HEX <= 7'b1000000; // Made as A to 0 for making it proper
default : HEX  <= 7'b1000000;
endcase
end
endmodule
```

Testbench Code :

```
module lift_control_tb();
reg clk, UP, PWR;
reg [3:0] IP_floor;
wire [3:0] cnt;
```

Date:31/12/2021

```
lift_control GA1 (clk, cnt, IP_floor, UP, PWR);
initial
begin
$monitor($time, "CLK=%b, cnt=%b, IP_floor=%b, UP=%b, PWR=%b", clk, cnt,
IP_floor, UP, PWR);
clk=1'b0;
end
always #5 clk = (~clk);
initial
begin
PWR=1;
#2 PWR=0;
#2 UP=1;
#2 IP_floor=2;
#100 UP=1;
#2 IP_floor=6;
#100 UP=1;
#2 IP_floor=10;
#100 UP=0;
#2 IP_floor=0;
#100 $stop;
//
end
endmodule
```

Date:31/12/2021

Simulation Waveform :

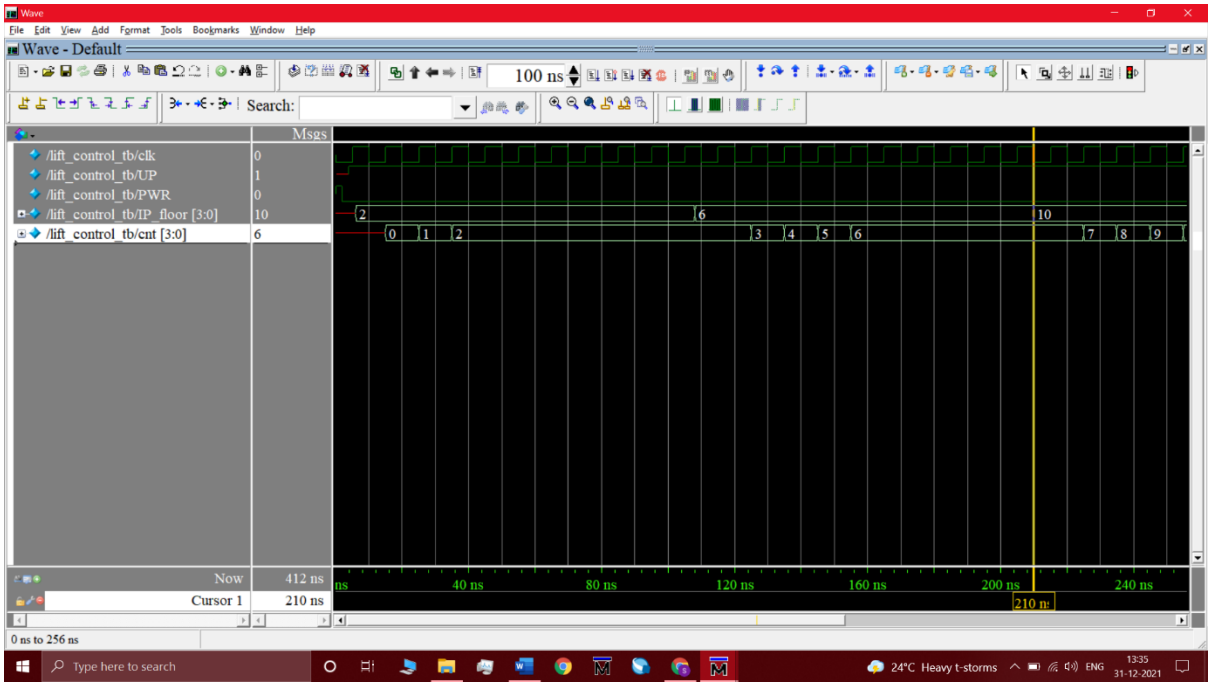


Figure 1.1 Simulation result for LIFT Detector at every positive edge of the clock the input is read.

In the Figure 1.1 we observe that at time 6ns user wants to up, so he has pressed UP switch in the Ground floor and entered the Lift. Now, he pressed floor 2 read using IP_floor variable so the count increase from zero to two and stays there unless next set of input comes. Now at 110ns user pressed 6 so it will go to 6tjh Floor.

Date:31/12/2021

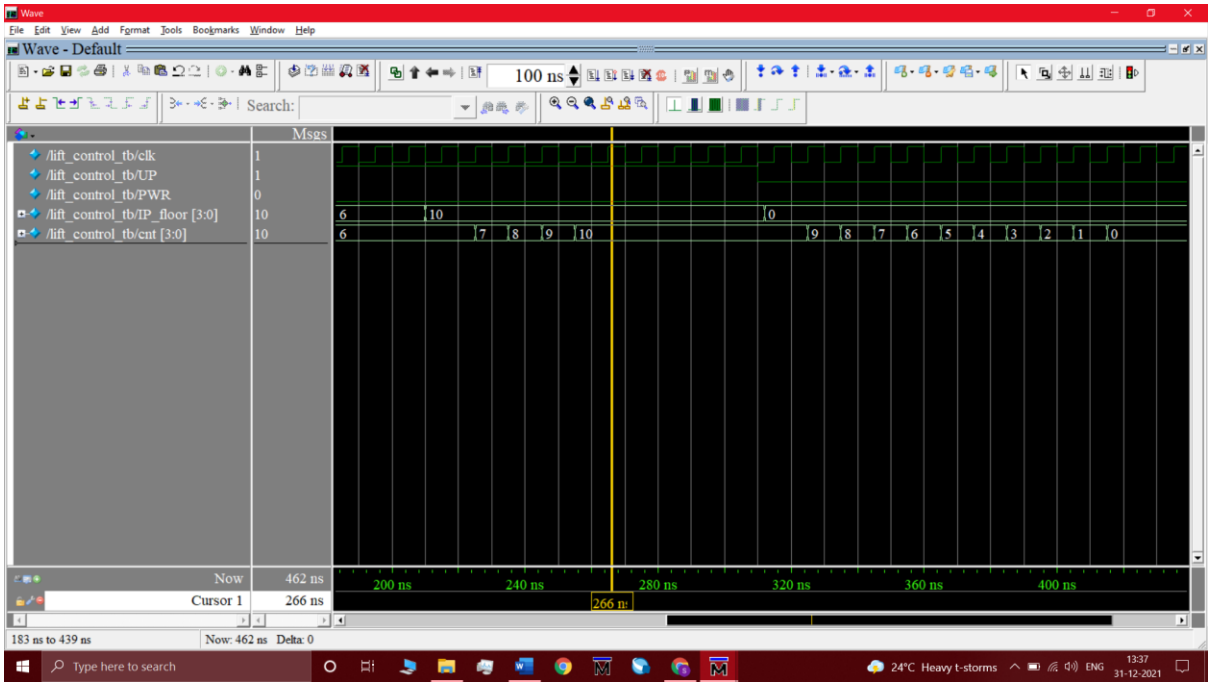
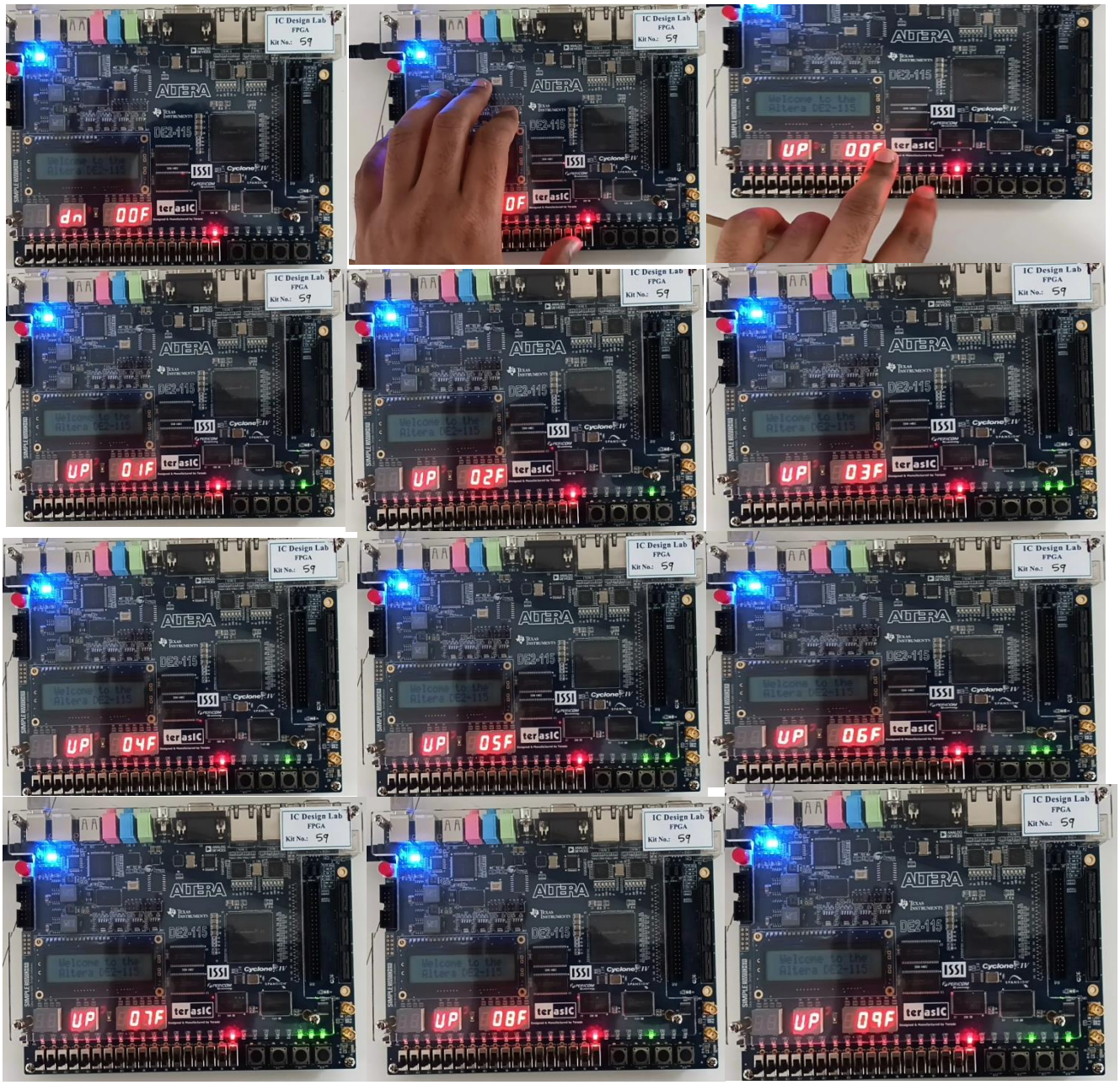


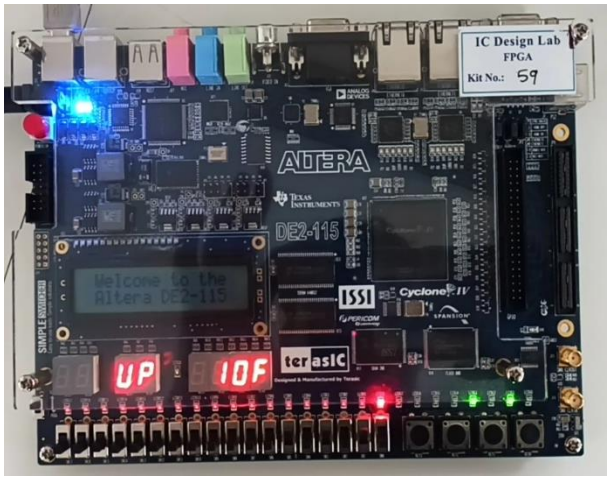
Figure 1.2 Simulation result for LIFT Detector at every positive edge of the clock the input is read.

In the Figure 1.1 we observe that at time 210ns user wants to go up, entered the Lift. Now, he pressed floor 10 read using IP_floor variable so the count increase from previous value to 10 and stays there unless next set of input comes. Now at 310ns user pressed 0 so it will go to Ground Floor.

Date:31/12/2021

Output For Upward Direction :



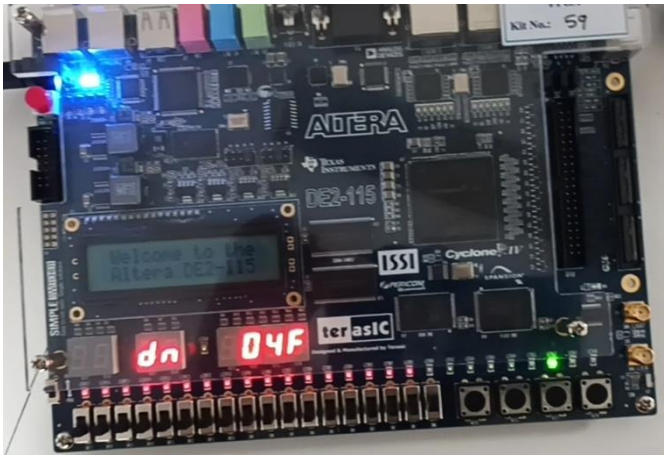


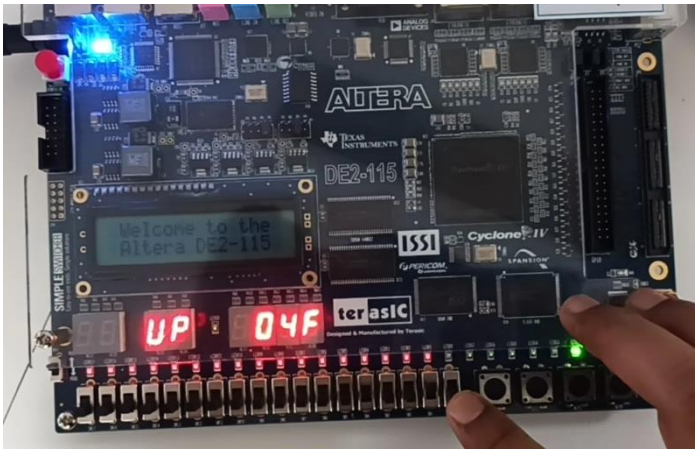
Date:31/12/2021

Output For Downward Direction :

Date:31/12/2021

PWR Button working :





Output Video Link:

https://drive.google.com/folderview?id=1JTYMjKYIjR7UmH7c2KLAmmJ-3vBs0YIf

Date:31/12/2021