In LAB: Implement Two-Digit-Counter in Verilog, start from 0 and increment 1 every second. When it reaches 99 or external reset signal is HIGH, the counter should be reset. And upload it to Logic Design Board. Use 50MHz oscillator as clock input, tactile switch as reset button, and two 7-segment displays as output.

Codes:

1. Frequency Divider:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 19:05:52 05/15/2018

// Design Name:

// Module Name: FreqDiv

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module FreqDiv(

input clkin,

input clr,

output reg clkout

);

reg [31:0] cnt;

always @(posedge clkin) begin

if(clr) begin

cnt <= 32'd0;

clkout <= 1'b0;

end

else if(cnt == 32'd25000000) begin

cnt <= 32'd0;

clkout <= ~clkout;

end

else begin

cnt <= cnt+1;

end

end

endmodule

2. BCD to 7-Segment Decoder:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 19:04:48 05/15/2018

// Design Name:

// Module Name: BCDto7

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module BCDto7(

input [3:0] bcd,

output reg [6:0] seg

);

always @(bcd) begin

case (bcd)

4'd0: seg <= 7'b0111111;

4'd1: seg <= 7'b0000110;

4'd2: seg <= 7'b1011011;

4'd3: seg <= 7'b1001111;

4'd4: seg <= 7'b1100110;

4'd5: seg <= 7'b1101101;

4'd6: seg <= 7'b1111101;

4'd7: seg <= 7'b0000111;

4'd8: seg <= 7'b1111111;

4'd9: seg <= 7'b1101111;

endcase

end

endmodule

3. 100 Counter:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 19:17:29 05/15/2018

// Design Name:

// Module Name: counter100

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module counter100(

input clk,

input clr,

input stop,

output reg [3:0] high\_num,

output reg [3:0] low\_num

);

always @(posedge clk) begin

if (clr) begin

high\_num <= 4'd0;

low\_num <= 4'd0;

end

else if (stop) begin

end

else begin

if (low\_num == 4'd9) begin

case (high\_num)

4'd0: high\_num <= 4'd1;

4'd1: high\_num <= 4'd2;

4'd2: high\_num <= 4'd3;

4'd3: high\_num <= 4'd4;

4'd4: high\_num <= 4'd5;

4'd5: high\_num <= 4'd6;

4'd6: high\_num <= 4'd7;

4'd7: high\_num <= 4'd8;

4'd8: high\_num <= 4'd9;

4'd9: high\_num <= 4'd0;

endcase

end

case(low\_num)

4'd0: low\_num <= 4'd1;

4'd1: low\_num <= 4'd2;

4'd2: low\_num <= 4'd3;

4'd3: low\_num <= 4'd4;

4'd4: low\_num <= 4'd5;

4'd5: low\_num <= 4'd6;

4'd6: low\_num <= 4'd7;

4'd7: low\_num <= 4'd8;

4'd8: low\_num <= 4'd9;

4'd9: low\_num <= 4'd0;

endcase

end

end

endmodule

4. Two Digit Counter:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 19:17:29 05/15/2018

// Design Name:

// Module Name: counter100

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module counter100(

input clk,

input clr,

input stop,

output reg [3:0] high\_num,

output reg [3:0] low\_num

);

always @(posedge clk) begin

if (clr) begin

high\_num <= 4'd0;

low\_num <= 4'd0;

end

else if (stop) begin

end

else begin

if (low\_num == 4'd9) begin

case (high\_num)

4'd0: high\_num <= 4'd1;

4'd1: high\_num <= 4'd2;

4'd2: high\_num <= 4'd3;

4'd3: high\_num <= 4'd4;

4'd4: high\_num <= 4'd5;

4'd5: high\_num <= 4'd6;

4'd6: high\_num <= 4'd7;

4'd7: high\_num <= 4'd8;

4'd8: high\_num <= 4'd9;

4'd9: high\_num <= 4'd0;

endcase

end

case(low\_num)

4'd0: low\_num <= 4'd1;

4'd1: low\_num <= 4'd2;

4'd2: low\_num <= 4'd3;

4'd3: low\_num <= 4'd4;

4'd4: low\_num <= 4'd5;

4'd5: low\_num <= 4'd6;

4'd6: low\_num <= 4'd7;

4'd7: low\_num <= 4'd8;

4'd8: low\_num <= 4'd9;

4'd9: low\_num <= 4'd0;

endcase

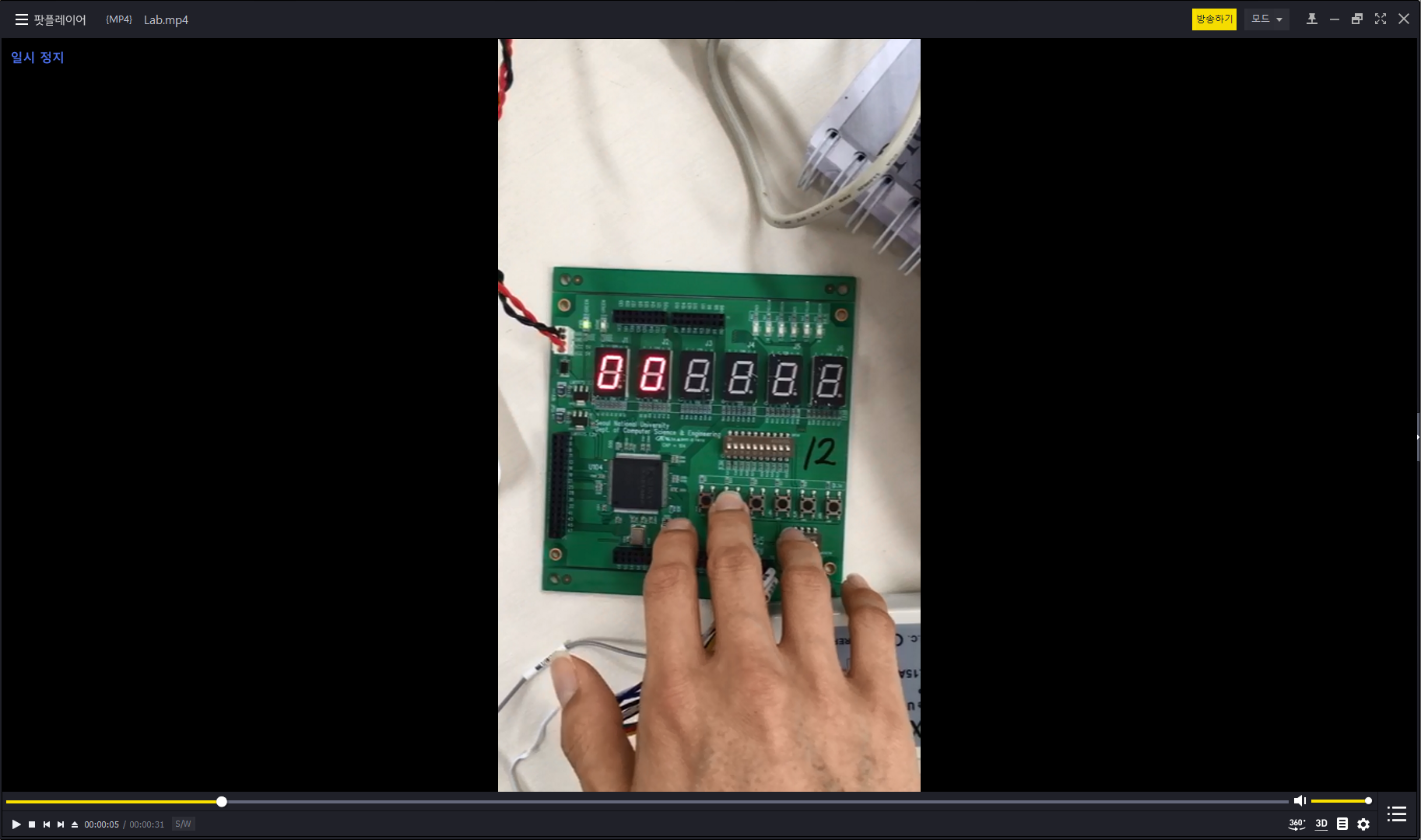
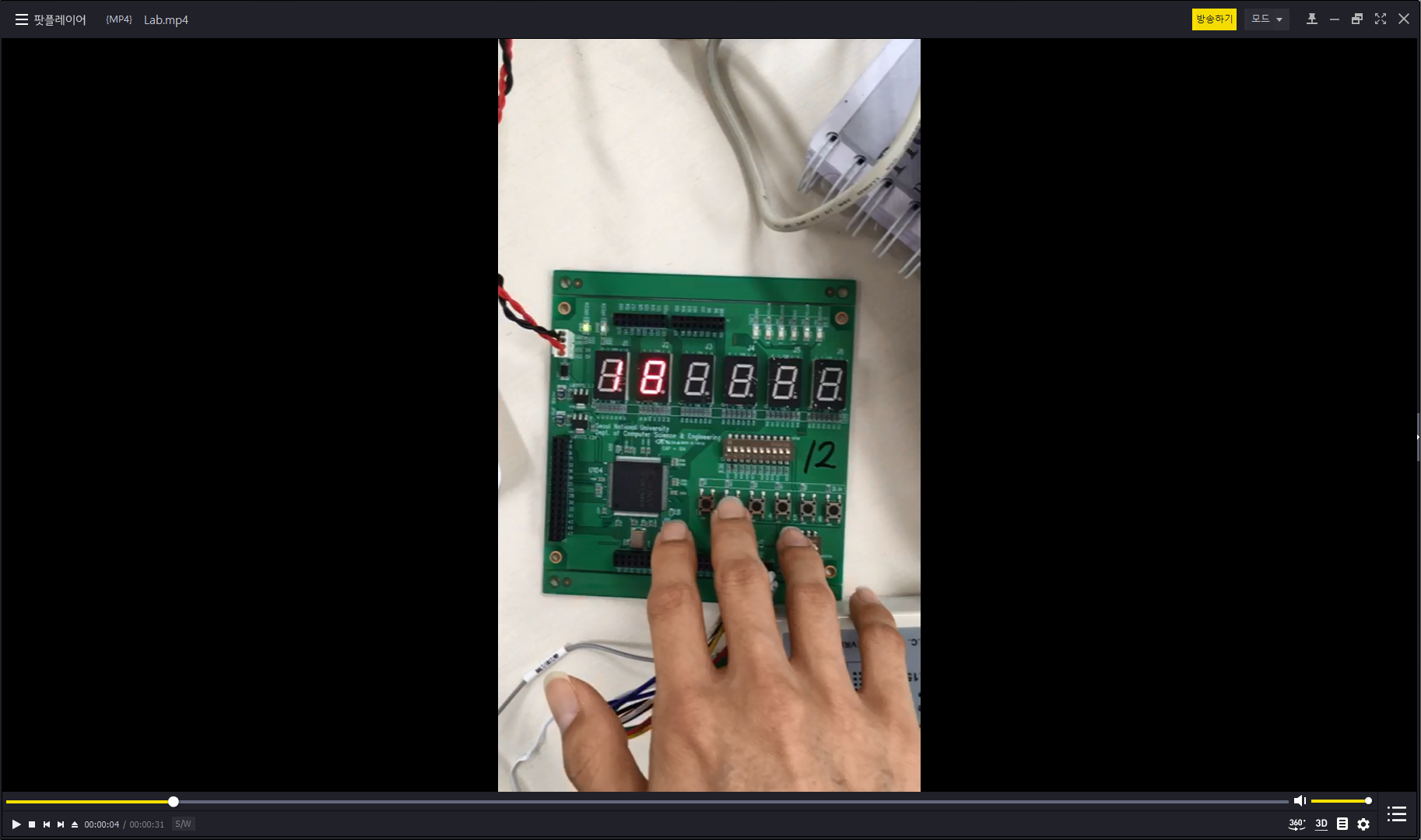
end

end

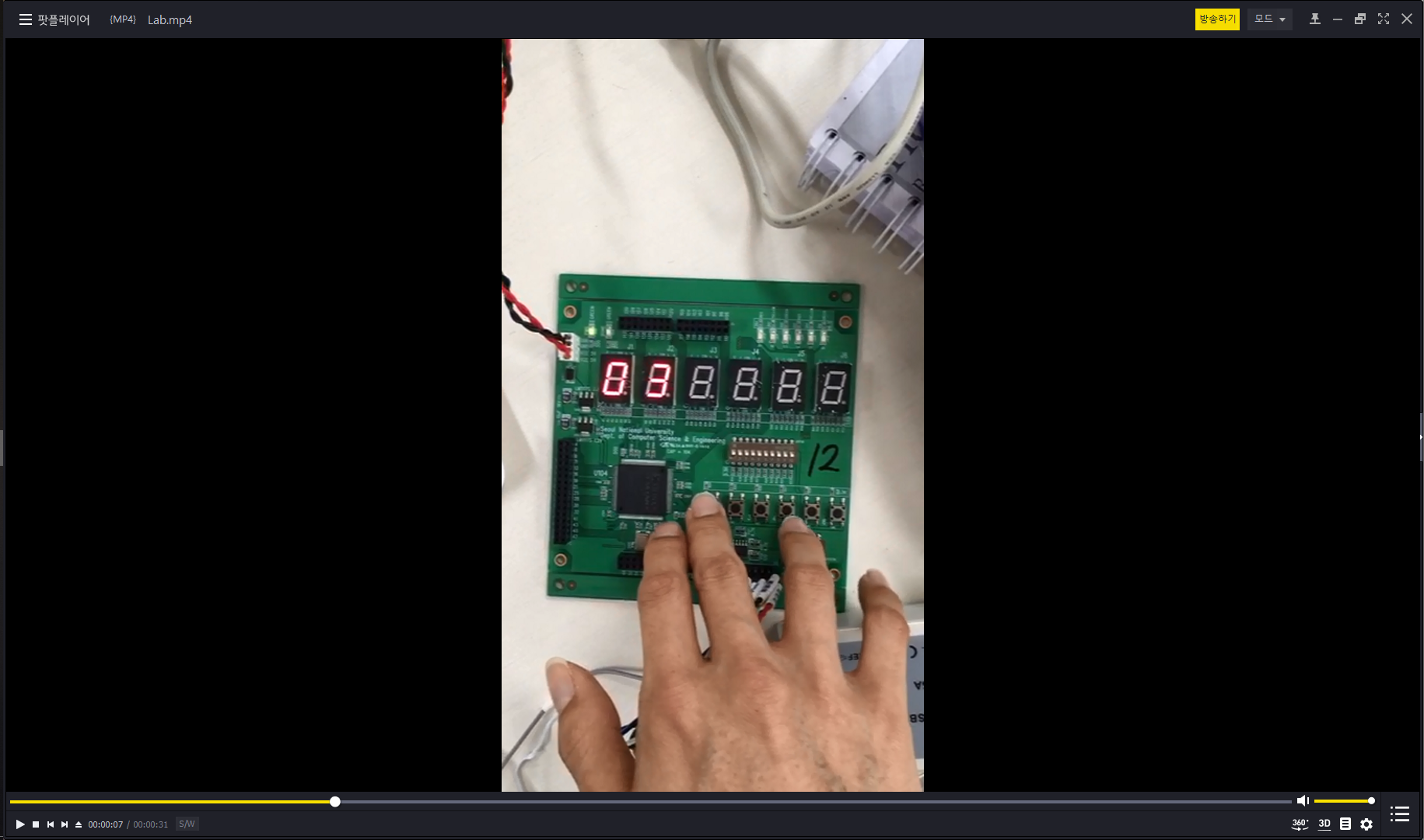
endmodule

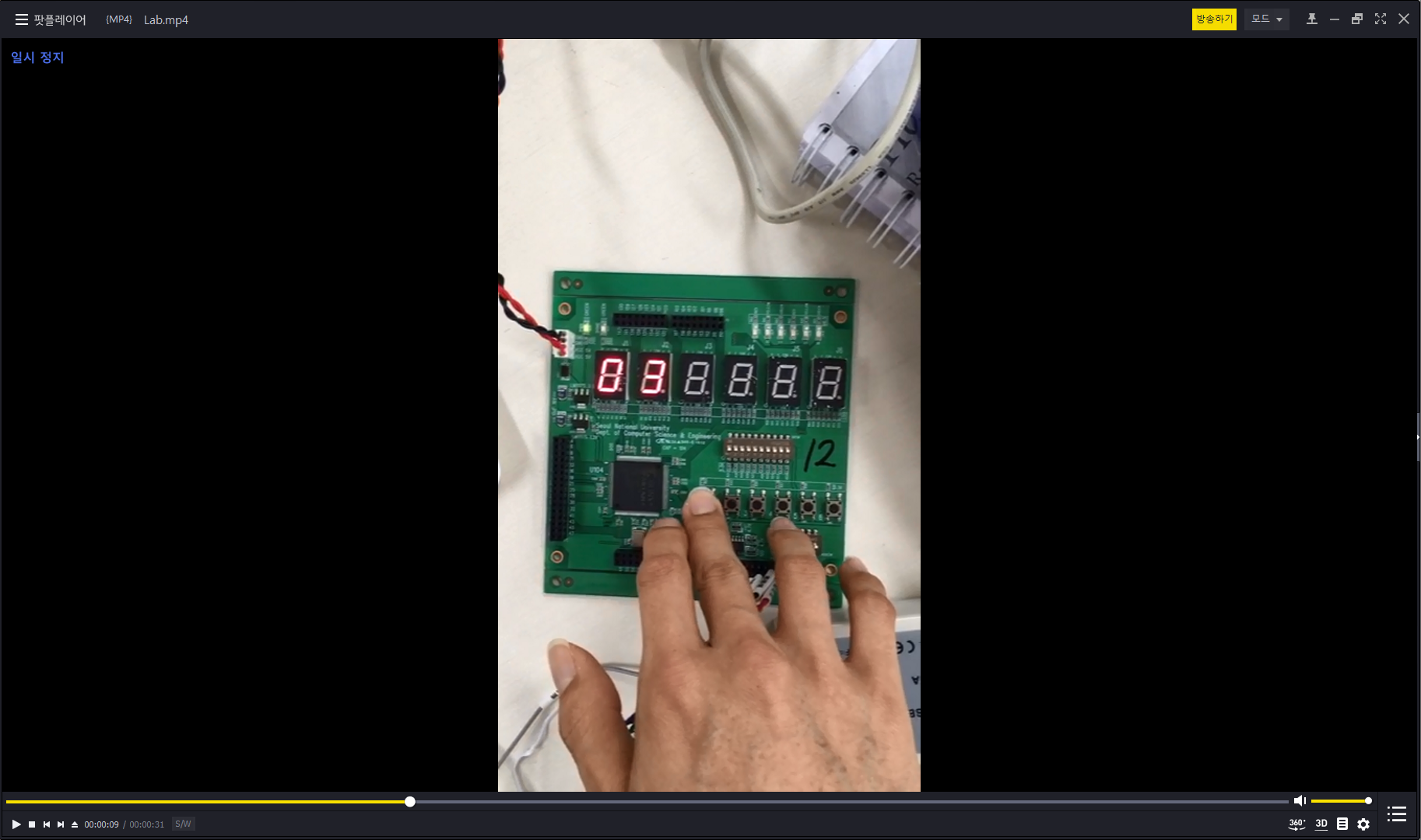
실행 결과:

Reset의 작동: 누르면 0으로 초기화



Hold의 작동: 누르면 숫자가 증가하지 않음





Homework: add external ‘toggle’ signal, which makes TDC count backwards when set HIGH.

Codes:

1. New 100 Counter

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 20:27:29 05/15/2018

// Design Name:

// Module Name: counter100\_toggle

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module counter100\_toggle(

input clk,

input clr,

input stop,

input toggle,

output reg [3:0] high\_num,

output reg [3:0] low\_num

);

always @(posedge clk) begin

if (clr) begin

high\_num <= 4'd0;

low\_num <= 4'd0;

end

else if (stop) begin

end

else if (toggle == 1'b0) begin

if (low\_num == 4'd9) begin

case (high\_num)

4'd0: high\_num <= 4'd1;

4'd1: high\_num <= 4'd2;

4'd2: high\_num <= 4'd3;

4'd3: high\_num <= 4'd4;

4'd4: high\_num <= 4'd5;

4'd5: high\_num <= 4'd6;

4'd6: high\_num <= 4'd7;

4'd7: high\_num <= 4'd8;

4'd8: high\_num <= 4'd9;

4'd9: high\_num <= 4'd0;

endcase

end

case(low\_num)

4'd0: low\_num <= 4'd1;

4'd1: low\_num <= 4'd2;

4'd2: low\_num <= 4'd3;

4'd3: low\_num <= 4'd4;

4'd4: low\_num <= 4'd5;

4'd5: low\_num <= 4'd6;

4'd6: low\_num <= 4'd7;

4'd7: low\_num <= 4'd8;

4'd8: low\_num <= 4'd9;

4'd9: low\_num <= 4'd0;

endcase

end

else begin

if (low\_num == 4'd0) begin

case (high\_num)

4'd0: high\_num <= 4'd9;

4'd1: high\_num <= 4'd0;

4'd2: high\_num <= 4'd1;

4'd3: high\_num <= 4'd2;

4'd4: high\_num <= 4'd3;

4'd5: high\_num <= 4'd4;

4'd6: high\_num <= 4'd5;

4'd7: high\_num <= 4'd6;

4'd8: high\_num <= 4'd7;

4'd9: high\_num <= 4'd8;

endcase

end

case(low\_num)

4'd0: low\_num <= 4'd9;

4'd1: low\_num <= 4'd0;

4'd2: low\_num <= 4'd1;

4'd3: low\_num <= 4'd2;

4'd4: low\_num <= 4'd3;

4'd5: low\_num <= 4'd4;

4'd6: low\_num <= 4'd5;

4'd7: low\_num <= 4'd6;

4'd8: low\_num <= 4'd7;

4'd9: low\_num <= 4'd8;

endcase

end

end

endmodule

2. New Two-Digit-Counter:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 20:26:36 05/15/2018

// Design Name:

// Module Name: TDC\_toggle

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

module TDC\_toggle(

input tclk,

input stop,

input clr,

input reset,

input toggle,

output [6:0] low\_seg,

output [6:0] high\_seg

);

wire [3:0] high\_num;

wire [3:0] low\_num;

wire clk;

FreqDiv T1(.clkin(tclk), .clr(clr), .clkout(clk));

BCDto7 T2(.bcd(high\_num), .seg(high\_seg));

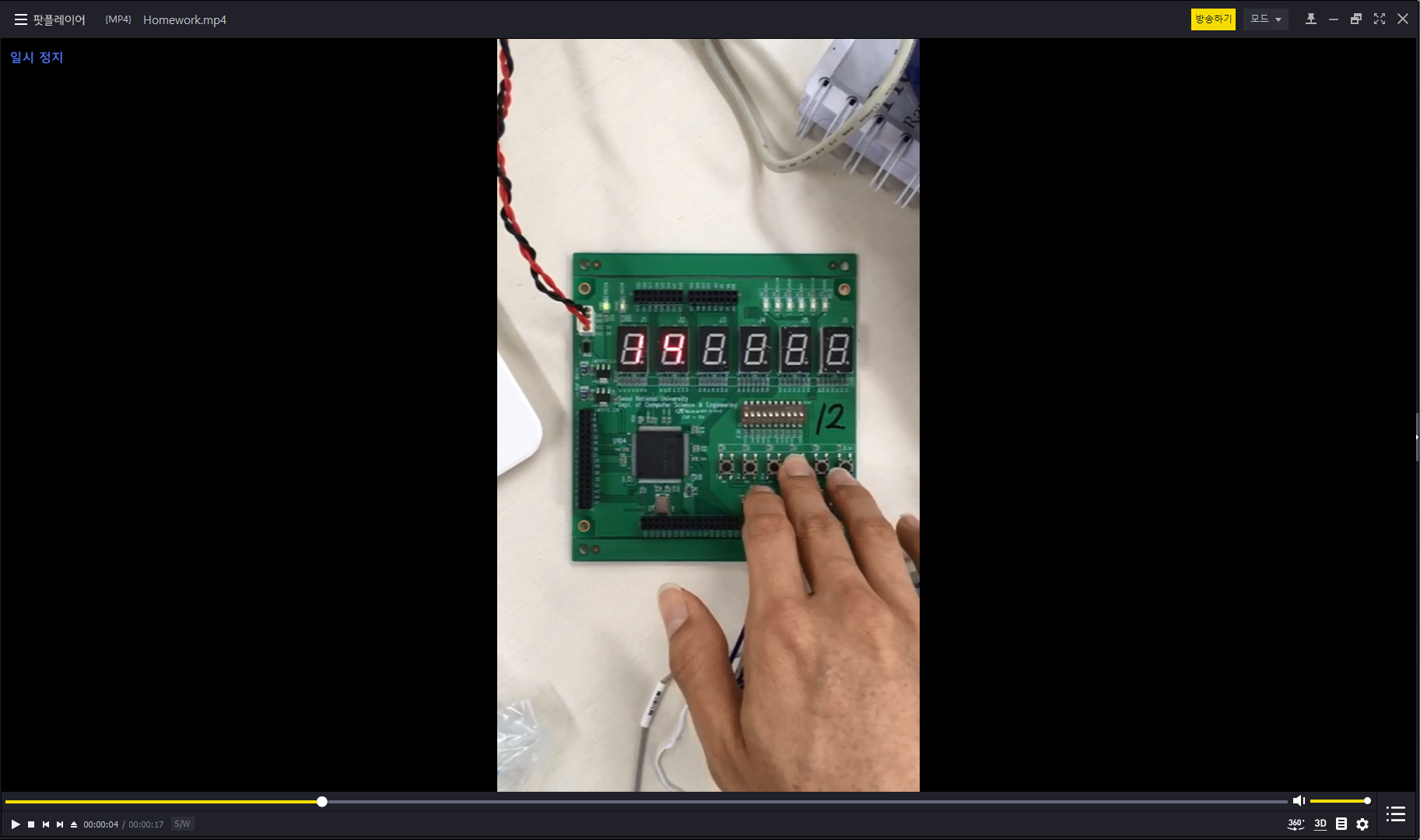
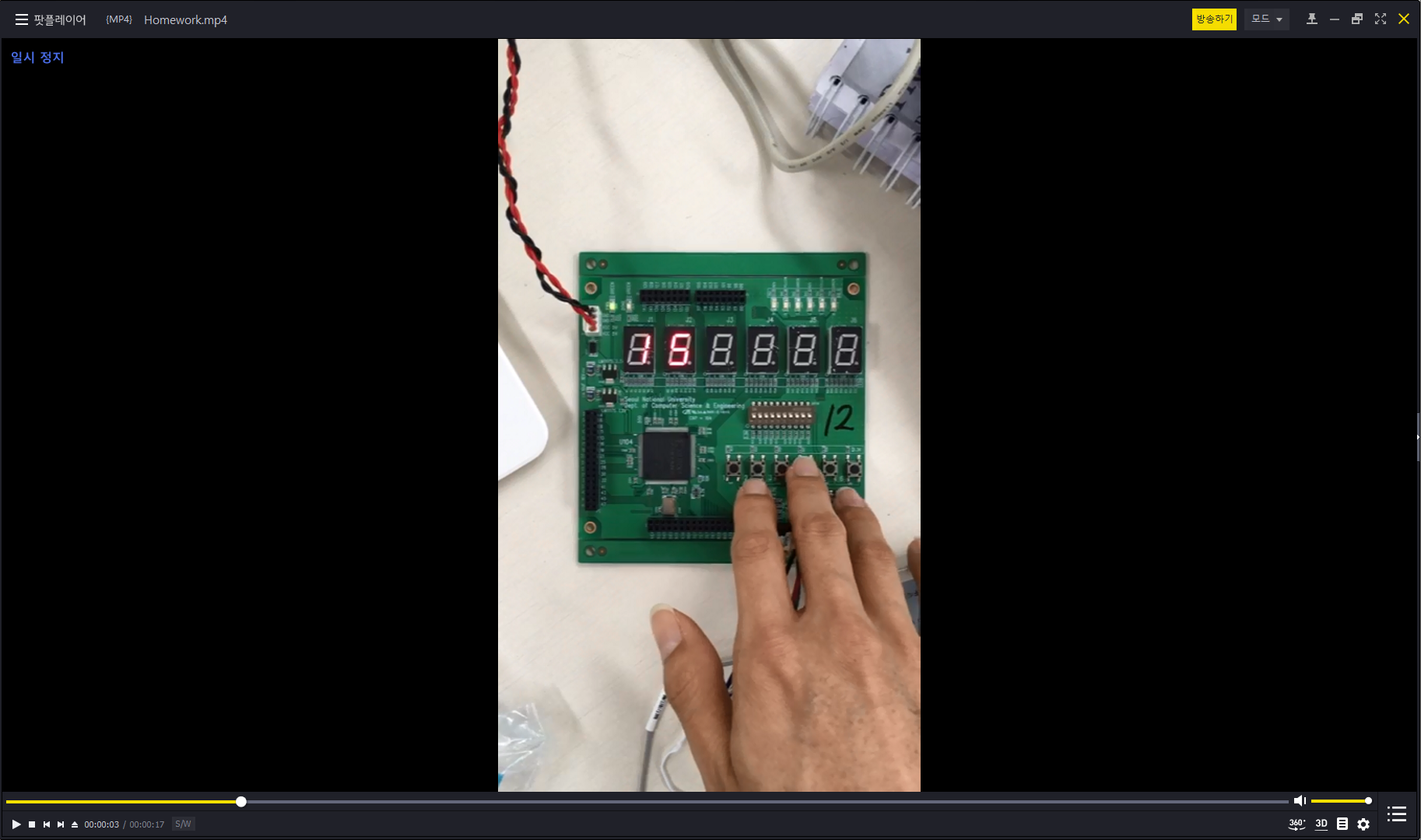
BCDto7 T3(.bcd(low\_num), .seg(low\_seg));

counter100\_toggle T4(.clk(clk), .clr(reset), .stop(stop), .toggle(toggle), .high\_num(high\_num), .low\_num(low\_num));

endmodule

실행 결과:

Toggle의 작동: 누르고 있는 동안 숫자가 줄어든다.



떼면 다시 숫자가 증가:

