Benjamin Hunt

815841735

PartA

module bcd7segdecoder(input [3:0] bcdin, output reg [6:0] segs);

always @(bcdin) begin

    case (bcdin)

            0 : segs = 7'b0000001;

            1 : segs = 7'b1001111;

            2 : segs = 7'b0010010;

            3 : segs = 7'b0000110;

            4 : segs = 7'b1001100;

            5 : segs = 7'b0100100;

            6 : segs = 7'b0100000;

            7 : segs = 7'b0001111;

            8 : segs = 7'b0000000;

            9 : segs = 7'b0000100;

10 : segs = 7'b0001000;

11 : segs = 7'b1100000;

12 : segs = 7'b0110001;

13 : segs = 7'b1000010;

14 : segs = 7'b0110000;

15 : segs = 7'b0111000;

            default : segs = 7'b1111111;

    endcase

end

endmodule

PartB

module ioTest (//input  M\_CLOCK,

input  [3:0] IO\_PB,       // IO Board Pushbutton Switches

   input  [7:0] IO\_DSW,      // IO Board Dip Switchs

//output [3:0] F\_LED,       // FPGA LEDs

   output reg [7:0] IO\_LED,  // IO Board LEDs

output [3:0] IO\_SSEGD,    // IO Board Seven Segment Digits

output [7:0] IO\_SSEG,     //7=dp, 6=g, 5=f,4=e, 3=d,2=c,1=b, 0=a

output IO\_SSEG\_COL,       // Seven segment column

output DEC\_POINT);        // Decimal point in the seven segment

//====================================

//Variable declaration

assign IO\_SSEG\_COL = 1; // deactivate the column displays

assign DEC\_POINT = 1;   // deactivate the the decimal point of the seven segment

reg [3:0] \_and, \_or, \_nand, \_nor; //reg because used in always @\*

assign IO\_SSEGD[3:0] = 4'b1111;

assign IO\_SSEG[7:0] = 8'b11111111;

//===================================

always @\* begin

\_and = IO\_DSW[7:4] & IO\_DSW[3:0];

\_or = IO\_DSW[7:4] | IO\_DSW[3:0];

\_nand = ~(IO\_DSW[7:4] & IO\_DSW[3:0]);

\_nor = ~(IO\_DSW[7:4] | IO\_DSW[3:0]);   end

//==================================

always  @\* begin

if (IO\_PB[0] == 0)

IO\_LED <= \_and;

else if (IO\_PB[1] == 0)

IO\_LED <= \_or;

else if (IO\_PB[2] == 0)

IO\_LED <= \_nand;

else if (IO\_PB[3] == 0)

IO\_LED <= \_nor;

else

IO\_LED <= 4'b0000;

end

endmodule

PartC

module ioTest (//input  M\_CLOCK,

input  [3:0] IO\_PB,       // IO Board Pushbutton Switches

   input  [7:0] IO\_DSW,      // IO Board Dip Switchs

//output [3:0] F\_LED,       // FPGA LEDs

   output reg [7:0] IO\_LED,  // IO Board LEDs

output reg [3:0] IO\_SSEGD,    // IO Board Seven Segment Digits

output reg [7:0] IO\_SSEG,     //7=dp, 6=g, 5=f,4=e, 3=d,2=c,1=b, 0=a

output IO\_SSEG\_COL,       // Seven segment column

output DEC\_POINT);        // Decimal point in the seven segment

//====================================

//Variable declaration

assign IO\_SSEG\_COL = 1; // deactivate the column displays

assign DEC\_POINT = 1;   // deactivate the the decimal point of the seven segment

reg [3:0] tot; //reg because used in always @\*

//===================================

// Dip switches control

always @\* begin

tot = IO\_DSW[7:4] + IO\_DSW[3:0];

end

//==================================

// Push button controls

// priority encoded push buttons

always  @\* begin

if (~IO\_PB[0]) IO\_SSEGD <= 4'b0111;

else IO\_SSEGD <= 4'b1111;

end

//=======================================

// Controlling seven segments

always @(\*) begin

case(tot)

            0 : IO\_SSEG <= {DEC\_POINT,7'b1000000};

            1 : IO\_SSEG <= {DEC\_POINT,7'b1111001};

            2 : IO\_SSEG <= {DEC\_POINT,7'b0100100};

            3 : IO\_SSEG <= {DEC\_POINT,7'b0110000};

            4 : IO\_SSEG <= {DEC\_POINT,7'b0011001};

            5 : IO\_SSEG <= {DEC\_POINT,7'b0010010};

            6 : IO\_SSEG <= {DEC\_POINT,7'b0000010};

            7 : IO\_SSEG <= {DEC\_POINT,7'b1111000};

            8 : IO\_SSEG <= {DEC\_POINT,7'b0000000};

            9 : IO\_SSEG <= {DEC\_POINT,7'b0011000};

10 : IO\_SSEG <= {DEC\_POINT,7'b0001000};

11 : IO\_SSEG <= {DEC\_POINT,7'b0000011};

12 : IO\_SSEG <= {DEC\_POINT,7'b1000110};

13 : IO\_SSEG <= {DEC\_POINT,7'b0100001};

14 : IO\_SSEG <= {DEC\_POINT,7'b0000110};

15 : IO\_SSEG <= {DEC\_POINT,7'b0001110};

            default : IO\_SSEG <= {DEC\_POINT,7'b1111111};

    endcase

end

endmodule

PartD

module ioTest (input  M\_CLOCK,  // FPGA clock

  input  [3:0] IO\_PB,  // IO Board Pushbutton Switches

  input  [7:0] IO\_DSW,  // IO Board Dip Switchs

      output [3:0] F\_LED,  // FPGA LEDs

    output [7:0] IO\_LED,  // IO Board LEDs

  output [3:0] IO\_SSEGD, // IO Board Seven Segment Digits

  output [7:0] IO\_SSEG,  // 7=dp, 6=g, 5=f,4=e, 3=d,2=c,1=b, 0=a

  output IO\_SSEG\_COL);  // Seven segment column

reg [31:0] counter; // count the clock cycles

parameter integer offset = 28; // divides the clock by 2^(offset+1)

//====================================

assign IO\_SSEG\_COL = 1; // deactivate the colon displays

assign IO\_SSEGD = 4'b1111; // deactivate the seven segment display

assign IO\_SSEG = 8'b11111111; // deactivate the seven segment display

// assign IO\_LED = 8'b00000000; // deactivate the IO board LEDs

always @(posedge M\_CLOCK) begin

counter = counter + 1;

end

// MSBs of Counter activate the FPGA LEDs

assign F\_LED = 4'b0000;  // on the FPGA board itself

assign IO\_LED = counter[offset+3:offset-4]; // and the IO board LEDs

endmodule