module ssegkey ( output wire [3:0] rowwrite,

input [3:0] colread,

input clk,

output wire [3:0] data,

output wire [3:0] grounds,

output wire [6:0] display

);

reg [15:0] data\_all;

wire [3:0] keyout;

reg [25:0] clk1;

reg [1:0] ready\_buffer;

reg ack;

wire ready;

assign data=keyout;

sevensegment ss1 (.datain(data\_all), .grounds(grounds), .display(display), .clk(clk));

keypad\_ex kp1(.rowwrite(rowwrite),.colread(colread),.clk(clk),.keyout(keyout),.ready(ready),.ack(ack));

always @(posedge clk)

ready\_buffer<= {ready\_buffer[0],ready}; //

always @(posedge clk)

if (ready\_buffer==2'b01)

begin

data\_all<={data\_all[11:0],keyout};

ack<=1;

end

else

ack<=0;

initial

begin

data\_all=0;

ack=0;

end

endmodule

module keypad\_ex (

output reg [3:0] rowwrite,

input [3:0] colread,

input clk,

input ack,

output reg [3:0] keyout,

output reg ready

);

wire keypressed;

reg [25:0] clk1;

reg [3:0] keyread;

reg [3:0] rowpressed;

reg [3:0] pressedcol [0:3];

reg [11:0] rowpressed\_buffer0;

reg [11:0] rowpressed\_buffer1;

reg [11:0] rowpressed\_buffer2;

reg [11:0] rowpressed\_buffer3;

reg [3:0] rowpressed\_debounced;

reg [1:0] keypressed\_buffer;

always @(posedge clk)

clk1<=clk1+1;

always @(posedge clk1[15])

rowwrite<={rowwrite[2:0], rowwrite[3]};

always @(posedge clk1[15])

if (rowwrite== 4'b1110)

begin

rowpressed[0]<= ~(&colread); //colread=1111--> none of them pressed, colread=1110 --> 1, colread=1101-->2, 1011-->3, 0111->A

pressedcol[0]<=colread;

end

else if (rowwrite==4'b1101)

begin

rowpressed[1]<= ~(&colread); //colread=1111--> none of them pressed, colread=1110 --> 4, colread=1101-->5, 1011-->6, 0111->B

pressedcol[1]<=colread;

end

else if (rowwrite==4'b1011)

begin

rowpressed[2]<= ~(&colread); //colread=1111--> none of them pressed, colread=1110 --> 7, colread=1101-->8, 1011-->9, 0111->C

pressedcol[2]<=colread;

end

else if (rowwrite==4'b0111)

begin

rowpressed[3]<= ~(&colread); //colread=1111--> none of them pressed, colread=1110 --> \*(E), colread=1101-->0, 1011-->#(F), 0111->D

pressedcol[3]<=colread;

end

wire transition0\_10;

wire transition0\_01;

assign transition0\_10=~|rowpressed\_buffer0;

assign transition0\_01=&rowpressed\_buffer0;

wire transition1\_10;

wire transition1\_01;

assign transition1\_10=~|rowpressed\_buffer1;

assign transition1\_01=&rowpressed\_buffer1;

wire transition2\_10;

wire transition2\_01;

assign transition2\_10=~|rowpressed\_buffer2;

assign transition2\_01=&rowpressed\_buffer2;

wire transition3\_10;

wire transition3\_01;

assign transition3\_10=~|rowpressed\_buffer3; //kpd=1-->0

assign transition3\_01=&rowpressed\_buffer3; //kpd=0-->1

always @(posedge clk1[15])

begin

rowpressed\_buffer0<= {rowpressed\_buffer0[10:0],rowpressed[0]};

if (rowpressed\_debounced[0]==0 && transition0\_01)

rowpressed\_debounced[0]<=1;

if (rowpressed\_debounced[0]==1 && transition0\_10)

rowpressed\_debounced[0]<=0;

rowpressed\_buffer1<= {rowpressed\_buffer1[10:0],rowpressed[1]};

if (rowpressed\_debounced[1]==0 && transition1\_01)

rowpressed\_debounced[1]<=1;

if (rowpressed\_debounced[1]==1 && transition1\_10)

rowpressed\_debounced[1]<=0;

rowpressed\_buffer2<= {rowpressed\_buffer2[10:0],rowpressed[2]};

if (rowpressed\_debounced[2]==0 && transition2\_01)

rowpressed\_debounced[2]<=1;

if (rowpressed\_debounced[2]==1 && transition2\_10)

rowpressed\_debounced[2]<=0;

rowpressed\_buffer3<= {rowpressed\_buffer3[10:0],rowpressed[3]};

if (rowpressed\_debounced[3]==0 && transition3\_01)

rowpressed\_debounced[3]<=1;

if (rowpressed\_debounced[3]==1 && transition3\_10)

rowpressed\_debounced[3]<=0;

end

always @\*

begin

if (rowpressed\_debounced[0]==1)

begin

if (pressedcol[0]==4'b1110)

keyread=4'h1;

else if (pressedcol[0]==4'b1101)

keyread=4'h2;

else if (pressedcol[0]==4'b1011)

keyread=4'h3;

else if (pressedcol[0]==4'b0111)

keyread=4'hA;

else keyread=4'b0000;

end

else if (rowpressed\_debounced[1]==1)

begin

if (pressedcol[1]==4'b1110)

keyread=4'h4;

else if (pressedcol[1]==4'b1101)

keyread=4'h5;

else if (pressedcol[1]==4'b1011)

keyread=4'h6;

else if (pressedcol[1]==4'b0111)

keyread=4'hB;

else keyread=4'b0000;

end

else if (rowpressed\_debounced[2]==1)

begin

if (pressedcol[2]==4'b1110)

keyread=4'h7;

else if (pressedcol[2]==4'b1101)

keyread=4'h8;

else if (pressedcol[2]==4'b1011)

keyread=4'h9;

else if (pressedcol[2]==4'b0111)

keyread=4'hC;

else keyread=4'b0000;

end

else if (rowpressed\_debounced[3]==1)

begin

if (pressedcol[3]==4'b1110)

keyread=4'hE;

else if (pressedcol[3]==4'b1101)

keyread=4'h0;

else if (pressedcol[3]==4'b1011)

keyread=4'hF;

else if (pressedcol[3]==4'b0111)

keyread=4'hD;

else keyread=4'b0000;

end

else keyread=4'b0000;

end //always

assign keypressed= rowpressed\_debounced[0]||rowpressed\_debounced[1]||rowpressed\_debounced[2]||rowpressed\_debounced[3];

always @(posedge clk)

keypressed\_buffer<={keypressed\_buffer[0],keypressed};

always @(posedge clk)

if ((keypressed\_buffer==2'b01)&&(ready==0))

begin

keyout<=keyread;

ready<=1;

end

else if (ack==1)

ready<=0;

initial

begin

rowwrite=4'b1110;

ready=0;

end

endmodule

module sevensegment(datain, grounds, display, clk);

input wire [15:0] datain;

output reg [3:0] grounds;

output reg [6:0] display;

input clk;

reg [3:0] data [3:0];

reg [1:0] count;

reg [25:0] clk1;

always @(posedge clk1[15])

begin

grounds <= {grounds[2:0],grounds[3]};

count <= count + 1;

end

always @(posedge clk)

clk1 <= clk1 + 1;

always @(\*)

case(data[count])

0:display=7'b1111110; //starts with a, ends with g

1:display=7'b0110000;

2:display=7'b1101101;

3:display=7'b1111001;

4:display=7'b0110011;

5:display=7'b1011011;

6:display=7'b1011111;

7:display=7'b1110000;

8:display=7'b1111111;

9:display=7'b1111011;

'ha:display=7'b1110111;

'hb:display=7'b0011111;

'hc:display=7'b1001110;

'hd:display=7'b0111101;

'he:display=7'b1001111;

'hf:display=7'b1000111;

default display=7'b1111111;

endcase

always @\*

begin

data[0] = datain[15:12];

data[1] = datain[11:8];

data[2] = datain[7:4];

data[3] = datain[3:0];

end

initial begin

count = 2'b0;

grounds = 4'b1110;

clk1 = 0;

end

endmodule