input clk, rst,Key\_2,Key\_0;

output reg [Length-1:0] R = 8'b1000\_1001;

output reg [7:0] R\_b = 8'd0 ;

output reg [3:0] d\_0 = 4'd0 ; //The output is the single digit that needs to be displayed

output reg [3:0] d\_1 = 4'd0 ; //The output is the tens digit that needs to be displayed

reg start = 1'd0 ; //Record whether the Key\_2 is pressed

reg [31:0] Y = 32'd0 ; //Record the number of seconds the Key\_0 is pressed

reg [3:0] Z = 4'd0; //Record the number of times the Key\_0 is pressed

reg [3:0] t = 1'd0; //Whether to start timing

// Count the number of keystrokes

always@(posedge clk )

begin

if(rst) //If rst is active it will reset

begin

start <= 1'd0;

Z <= 4'd0;

end

else if(Key\_2 || Key\_0) //If Key \_2 or Key\_0 is pressed count will active

begin

if(Key\_2)

start <= 1'd1;

else if(Key\_0)

begin

if(Z== 4'd4) //It returns to 0 when the Key\_0 is pressed the fourth time

begin

Z <= 4'd0;

start <= 1'd0;

end

else

begin

Z <= Z + 1'd1; //Press at least 4 times

end

end

end

end

// Random number generation module

always @ (posedge clk)

if (rst) //If rst is active it will reset

R <= initial\_state;

R\_b <= 1'd0;

else if(start) //Generate a random number

begin

R[0] <= R[7];

R[1] <= Tap\_Coefficient[1] ? R[0] ^ R[7] : R[0];

R[2] <= Tap\_Coefficient[2] ? R[1] ^ R[7] : R[1];

R[3] <= Tap\_Coefficient[3] ? R[2] ^ R[7] : R[2];

R[4] <= Tap\_Coefficient[4] ? R[3] ^ R[7] : R[3];

R[5] <= Tap\_Coefficient[5] ? R[4] ^ R[7] : R[4];

R[6] <= Tap\_Coefficient[6] ? R[5] ^ R[7] : R[5];

R[7] <= Tap\_Coefficient[7] ? R[6] ^ R[7] : R[6];

end

else if (R[7:0] > 8'd0 && R[7:0] <= 8'd73 && t == 1'd0)

begin //Filter random numbers[1-73]

R\_b <= R[7:0];

// Condition of display

always @ (posedge clk)

begin

if (rst) //If rst is active it will reset

begin

d\_0 <= 8'd0;

d\_1 <= 8'd0;

end

else if (R[7:0] > 8'd0 && R[7:0] <= 8'd73 && t == 1'd0)

begin //Filter random numbers[1-73]

R\_b <= R[7:0];

d\_0 <= R[3:0];

d\_1 <= R[7:4];

end

else if (t == 1'd1) //Keep this number unchanged for 4 seconds when you press the Key\_0

begin

d\_0 <= d\_0;

d\_1 <= d\_1;

end

end

// Display time of 4 seconds

always @ (posedge clk) //clk was choose 50Mhz

begin

if (rst) //If rst is active it will reset

begin

Y <= 4'd0;

t <= 1'd0;

end

else if (Y == 32'd200000) //Counting to 200M times equal 4s

begin

Y <= 4'd0;

t <= 1'd0;

end

else if (Key\_0)

t <= 1'd1; //start timing

else if (t == 1'd1)

begin

Y <= Y + 1'd1;

end

end

endmodule

module Bcd(

input clk ,

input rst ,

input [7:0] R\_b ,

output reg [3:0] d\_0 = 4'd0 , //The output is the single digit that needs to be displayed

output reg [3:0] d\_1 = 4'd0 //The output is the tens digit that needs to be displayed

) ;

reg [7:0] bcd; //Binary to bcd

reg [3:0] count; //Need to shift 8 times

reg [15:0] shift\_reg;

reg [3:0] R = 8'b00011111;

//Calculate the number of shift

always @ (posedge clk)

begin

if (rst) //If rst is active it will reset

count <= 4'd0;

else if (count == 4'd8)

count <= 4'd0;

else

count <= count + 1'd1;

end

//Binary to Bcd

always @ (posedge clk)

begin

if (rst) //If rst is active it will reset

shift\_reg <= 1'd0;

else if (count == 1'd0)

begin

shift\_reg = {8'b0000\_0000,R};

end

else if (count <= 4'd8)

begin

if(shift\_reg[15:12]>=5)

shift\_reg[15:12]= shift\_reg[15:12]+ 3;

if(shift\_reg[11:8]>=5)

shift\_reg[11:8]= shift\_reg[11:8]+ 3;

if(shift\_reg[7:4]>=5)

shift\_reg[7:4]= shift\_reg[7:4]+ 3;

if(shift\_reg[3:0]>=5)

shift\_reg[3:0]= shift\_reg[3:0]+ 3;

shift\_reg = shift\_reg << 1;

end

end

// Output assignment

always @ (posedge clk)

begin

if (rst) //If rst is active it will reset

bcd <= 1'd0;

else if (count == 4'd8)

begin

bcd <= shift\_reg[15:8];

d\_0 <= bcd[3:0];

d\_1 <= bcd[7:4];

end

end

endmodule

module Bcd(

input clk ,

input rst ,

input [7:0] R\_n ,

output reg [3:0] d\_0 = 4'd0 , //The output is the single digit that needs to be displayed

output reg [3:0] d\_1 = 4'd0 //The output is the tens digit that needs to be displayed

) ;

integer i;

always @ (posedge clk)

begin

if (rst) //If rst is active it will reset

begin

d\_0 <= 4'd0;

d\_1 <= 4'd0;

end

else

for (i=7;i>=0;i=i-1)

begin //add 3 to columns >=5

if(d\_1 >=5)

d\_1 = d\_1 + 3;

if(d\_0 >=5)

d\_0 = d\_0 + 3;

d\_1 = d\_1 << 1;

d\_1[0] = d\_0[3];

d\_0 = d\_0 << 1;

d\_0[0] = R\_n[i];

end

end

endmodule