Logo, company name

Description automatically generatedNational University of Sciences and Technology

School of Electrical Engineering and Computer Science

Department of Electrical Engineering

|  |  |
| --- | --- |
| **EE-421: Digital System Design** | |
| **Faculty Member** | **Semester** |
| Dr. Rehan Ahmed | 6th |
| **Class/Section** | **Date** |
| BEE11 | 3rd April 2022 |

**Assignment 1**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Name** | **Registration No.** |  |
|  | Ali Hamza Malik | 291480 |  |

## Task 2:

## Code:

module top(CLOCK\_50,Reset,Enable,x,y,colour,plot,Done);

input CLOCK\_50,Reset,Enable;

output reg plot;

output reg[2:0]colour;

output reg [7:0] x;

output reg [6:0] y;

output Done;

always@(posedge CLOCK\_50,negedge Reset)begin

if(~Reset)begin plot<=1; x<=0; y<=0; end

else if((x<159)) begin x<=x+1; colour<=Enable?(y%8):3'b000; end

else if(x==159 & y<119) begin x<=0; y<=y+1; end

else if((y==119)&(x==159))begin plot<=0; end

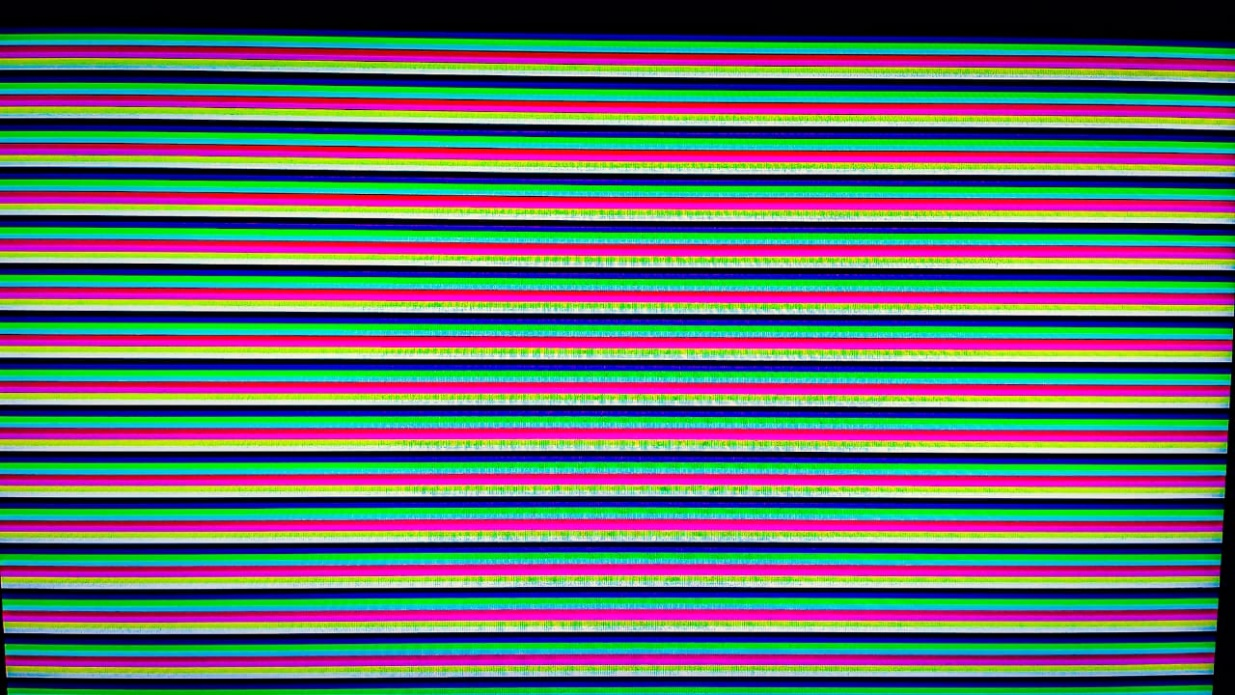
else begin x<=0;y<=0;end

end

assign Done=(~plot);

endmodule

## Output:



## Task 3:

## Code:

module top (CLOCK\_50,KEY\_N,SW,x,y,vga\_x,vga\_y,vga\_plot,vga\_colour);

input CLOCK\_50;

input [9:0]SW;

input [3:0]KEY\_N;

output reg [7:0] vga\_x;

output reg [6:0] vga\_y;

output reg [2:0]vga\_colour;

output reg vga\_plot;

output [7:0] x;

output [6:0] y;

wire [7:0] xc;

wire [6:0] yc;

assign xc=79;

assign yc=59;

wire [7:0] x1,x2;

wire [6:0] y1,y2;

wire Reset,key,writeR,plotclear,plotcircle;

wire Done,Clear;

assign Reset=KEY\_N[3];

assign key=KEY\_N[0];

wire [2:0]colourcircle;

wire [2:0]colourclear;

always@(\*)begin

if(Done)begin vga\_x=x2; vga\_y=y2; vga\_plot=plotcircle; vga\_colour=colourcircle; end

else begin vga\_x=x1; vga\_y=y1; vga\_plot=plotclear; vga\_colour=colourclear; end

end

task2 T2(CLOCK\_50,Reset,SW[9],x1,y1,colourclear,plotclear,Done);

task3 T3(CLOCK\_50,Reset,Done,key,xc,yc,SW[8:0],x,y,x2,y2,plotcircle,colourcircle);

endmodule

module task3(CLOCK\_50,Reset,Done,key,xc,yc,SW,x,y,xp,yp,plot,colourcircle);

input key,Reset,Done;

input CLOCK\_50;

input [7:0] xc;

input [6:0] yc;

input [8:0] SW;

output reg [7:0] xp;

output reg [6:0] yp;

output reg [7:0] x;

output reg [6:0] y;

output reg [2:0]colourcircle;

output plot;

wire ch;

assign ch=(x>y);

reg [5:0]r;

reg [3:0]state,nstate;

parameter R=4'd0,P1=4'd1,P2=4'd2,P3=4'd3,P4=4'd4,P5=4'd5,P6=4'd6,P7=4'd7,P8=4'd8;//Done=4'd9;

always@(\*)begin

case(state)

R:nstate=key?R:P1;

P1:nstate=P2;

P2:nstate=P3;

P3:nstate=P4;

P4:nstate=P5;

P5:nstate=P6;

P6:nstate=P7;

P7:nstate=(y==0)?R:P8;

P8:nstate=ch?R:P1;

default:nstate=state;

endcase

end

always@(posedge CLOCK\_50,negedge Reset)begin

if(~Reset)state<=R;

else if(Done) state<=nstate;

else state<=state;

end

reg [8:0]d;

always@(posedge CLOCK\_50)begin

case(state)

R:begin

if(SW[8:3]>59) r=59; else r=SW[8:3];

colourcircle=SW[2:0];

d=(9'd3)-(9'd2\*r);

x=(8'd0);

y=r;

xp=xc + x; yp=yc + y;

end

P1:begin xp=xc + x; yp=yc + y; end

P2:begin xp=xc - x; yp=yc + y; end

P3:begin xp=xc + x; yp=yc - y; end

P4:begin xp=xc - x; yp=yc - y; end

P5:begin xp=xc + y; yp=yc + x; end

P6:begin xp=xc - y; yp=yc + x; end

P7:begin xp=xc + y; yp=yc - x; end

P8:begin

xp=xc - y; yp=yc - x;

x=x+8'd1;

if ((d>256))begin d=d+((9'd4)\*x)+(9'd6); end

else if ((d<=256))begin d=d+((9'd4)\*(x-y))+(9'd10); y=y-7'd1; end

else d=d;

end

endcase

end

assign plot=~(state==R);

endmodule

module task2(CLOCK\_50,Reset,Enable,x,y,colour,plot,Done);

input CLOCK\_50,Reset,Enable;

output reg plot;

output reg[2:0]colour;

output reg [7:0] x;

output reg [6:0] y;

output Done;

always@(posedge CLOCK\_50,negedge Reset)begin

if(~Reset)begin plot<=1; x<=0; y<=0; end

else if((x<159)) begin x<=x+1; colour<=Enable?(y%8):3'b000; end

else if(x==159 & y<119) begin y<=y+1; x<=0; end

else if((y==119)&(x==159))begin plot<=0; end

else begin x<=0;y<=0;end

end

assign Done=(~plot);

endmodule

## Output:

## 

## Challenge Task:

## Code:

module top (CLOCK\_50,KEY\_N,x,y,vga\_x,vga\_y,vga\_plot,vga\_colour);

input CLOCK\_50;

input [3:0]KEY\_N;

output reg [7:0] vga\_x;

output reg [6:0] vga\_y;

output reg [2:0]vga\_colour;

output reg vga\_plot;

output [7:0] x;

output [6:0] y;

wire [7:0] xc;

wire [6:0] yc;

wire [7:0] x1,x2;

wire [6:0] y1,y2;

wire Reset,plotclear,plotcircle;

wire Done;

assign Reset=KEY\_N[3];

wire [2:0]colour,colourcircle,colourclear;

always@(\*)begin

if(Done)begin vga\_x=x2; vga\_y=y2; vga\_plot=plotcircle; vga\_colour=colourcircle; end

else begin vga\_x=x1; vga\_y=y1; vga\_plot=plotclear; vga\_colour=colourclear; end

end

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

reg check;

parameter dn=26;

reg [dn-1:0]count;

always@(posedge CLOCK\_50,negedge Reset)begin

if(!Reset) count<=0;

else if(count==50000010) count<=0;

else if(~plotcircle & Done)count<=count+1;

else count<=count;

end

//50000001

wire [4:0]tmp;

LFSR #(8,2,6,7,159,97)L0 (8'd79,CLOCK\_50,Reset,(count==50000009),xc);

LFSR #(7,2,3,6,119, 9) L1 (7'd59,CLOCK\_50,Reset,(count==50000009),yc);

LFSR #(5,1,2,4, 7, 0) L2 (5'd7 ,CLOCK\_50,Reset,(count==50000009),tmp);

assign colour=tmp[2:0];

wire [5:0] r;

Comparator C0(CLOCK\_50,xc,yc,r);

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

task2 T2(CLOCK\_50,Reset,1'b0,x1,y1,colourclear,plotclear,Done);

task3 T3(CLOCK\_50,Reset,Done,~(count==50000010),xc,yc,{r,colour},x,y,x2,y2,plotcircle,colourcircle);

endmodule

module Comparator(CLOCK\_50,xc,yc,Q3);

input CLOCK\_50;

input [7:0] xc;

input [6:0] yc;

output reg [5:0] Q3;

reg [7:0] a;

reg [6:0] b;

reg [7:0] Q1;

reg [6:0] Q2;

always@(\*)begin

a=(159-xc);

b=(119-yc);

Q1=((159-xc)<xc)?(159-xc):xc;

Q2=((119-yc)<yc)?(119-yc):yc;

Q3= (Q1<Q2)?Q1[5:0] :Q2[5:0];

end

endmodule

module LFSR #(parameter n=8,tap1=2,tap2=6,tap3=7,check=159,sub=97)(input [n-1:0] Din,input Clk,input L,input Enable,output reg [n-1:0] Qout);

reg [n-1:0]Q;

integer k;

//posedge Clk

always@(posedge Clk,negedge L)begin

if(!L) begin Q<=Din; end

else if(Enable) begin

for(k=0;k<n-1;k=k+1)begin

Q[k+1]<=Q[k];

end

Q[0]<=Q[tap1]^Q[tap2]^Q[tap3];

end

end

always@(\*)begin

if(Q>check) Qout=Q-sub;

else Qout=Q;

end

endmodule

module task3(CLOCK\_50,Reset,Done,key,xc,yc,SW,x,y,xp,yp,plot,colourcircle);

input key,Reset,Done;

input CLOCK\_50;

output reg [7:0] xp;

output reg [6:0] yp;

output reg [7:0] x;

output reg [6:0] y;

output reg [2:0]colourcircle;

output plot;

input [7:0] xc;

input [6:0] yc;

input [8:0] SW;

wire ch;

assign ch=(x>y);

reg [5:0]r;

reg [3:0]state,nstate;

parameter R=4'd0,P1=4'd1,P2=4'd2,P3=4'd3,P4=4'd4,P5=4'd5,P6=4'd6,P7=4'd7,P8=4'd8;//Done=4'd9;

always@(\*)begin

case(state)

R:nstate=key?R:P1;

P1:nstate=P2;

P2:nstate=P3;

P3:nstate=P4;

P4:nstate=P5;

P5:nstate=P6;

P6:nstate=P7;

P7:nstate=(y==0)?R:P8;

P8:nstate=ch?R:P1;

default:nstate=state;

endcase

end

always@(posedge CLOCK\_50,negedge Reset)begin

if(~Reset)state<=R;

else if(Done) state<=nstate;

else state<=state;

end

reg [8:0]d;

always@(posedge CLOCK\_50)begin

case(state)

R:begin

if(SW[8:3]>59) r=59; else r=SW[8:3];

colourcircle=SW[2:0];

d=(9'd3)-(9'd2\*r);

x=(8'd0);

y=r;

xp=xc + x; yp=yc + y;

end

P1:begin xp=xc + x; yp=yc + y; end

P2:begin xp=xc - x; yp=yc + y; end

P3:begin xp=xc + x; yp=yc - y; end

P4:begin xp=xc - x; yp=yc - y; end

P5:begin xp=xc + y; yp=yc + x; end

P6:begin xp=xc - y; yp=yc + x; end

P7:begin xp=xc + y; yp=yc - x; end

P8:begin

xp=xc - y; yp=yc - x;

x=x+8'd1;

if ((d>256))begin d=d+((9'd4)\*x)+(9'd6); end

else if ((d<=256))begin d=d+((9'd4)\*(x-y))+(9'd10); y=y-7'd1; end

else d=d;

end

endcase

end

assign plot=~(state==R);

endmodule

module task2(CLOCK\_50,Reset,Enable,x,y,colour,plot,Done);

input CLOCK\_50,Reset,Enable;

output reg plot;

output reg[2:0]colour;

output reg [7:0] x;

output reg [6:0] y;

output Done;

always@(posedge CLOCK\_50,negedge Reset)begin

if(~Reset)begin plot<=1; x<=0; y<=0; end

else if((x<159)) begin x<=x+1; colour<=Enable?(y%8):3'b000; end

else if(x==159 & y<119) begin y<=y+1; x<=0; end

else if((y==119)&(x==159))begin plot<=0; end

else begin x<=0;y<=0;end

end

assign Done=(~plot);

endmodule