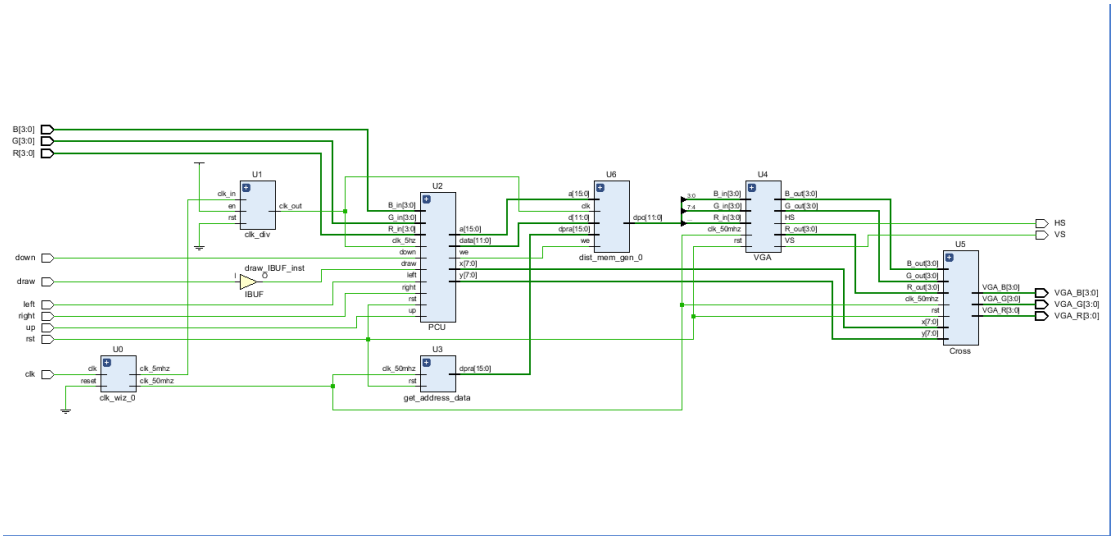
Lab4 存储器与显示控制器

1. 设计逻辑

使用模块化设计，输出 800*600 分辨率的显示器中部 256*256 大小白色画布，实现画笔。分为时钟分频模块、控制模块、扫描十字模块、VGA 扫描模块以及视频存储器模块。



时钟分频模块 U0 U1，使得输出的时钟频率分别为 50MHz 与 5Hz，作为 VGA 显示扫描时钟以及控制时钟

控制模块以画笔位置为参数，5Hz 的频率使得按一下移动 5 像素，当画笔到边界时，不能继续移动下去。

获取地址模块是用来获取在哪个地址上操作，得到 16 位地址

VGA 扫描模块使用 800*600 50Hz 的输出

2. 核心代码

(1) 顶层模块

```
module VGA_top(
    input [3:0]R,
    input [3:0]G,
    input [3:0]B,
    input draw,
    input up,
    input left,
    input right,
    input down,
    input clk,
    input rst,
    output [3:0]VGA_R,
    output [3:0]VGA_G,
    output [3:0]VGA_B,
    output VS,
    output HS
    );
    wire [15:0]a;
    wire [15:0]dpra;
    wire [11:0]d;
    wire [11:0]dpo;
    wire [3:0]R_in;
    wire [3:0]G_in;
    wire [3:0]B_in;
    wire [3:0]R_out;
    wire [3:0]G_out;
    wire [3:0]B_out;
    wire we;
    wire [7:0]x;
    wire [7:0]y;
    wire clk_50mhz;
    wire clk_5mhz;
    wire clk_5hz;
    wire locked;

    assign {R_in,G_in,B_in}=dpo;

    clk_wiz_0 U0(.clk_5mhz(clk_5mhz),.clk_50mhz(clk_50mhz),.reset(0),.locked(locked),.clk(clk));
    clk_div U1(.clk_in(clk_5mhz),.en(1),.rst(0),.clk_out(clk_5hz));
    PCU U2(.clk_5hz(clk_5hz),.rst(rst),.draw(draw),.up(up),.left(left),.right(right),.down(down),.R_in(R),.G_in(G),.B_in(B),.a(a),.data(d),.x(x),.y(y),.we(we));
    get_address_data U3(.clk_50mhz(clk_50mhz),.rst(rst),.dpra(dpra));
    VGA U4(.clk_50mhz(clk_50mhz),.R_in(R_in),.G_in(G_in),.B_in(B_in),.rst(rst),.R_out(R_out),.G_out(G_out),.B_out(B_out),.HS(HS),.VS(VS));
    Cross U5(.clk_50mhz(clk_50mhz),.rst(rst),.R_out(R_out),.G_out(G_out),.B_out(B_out),.x(x),.y(y),.VGA_R(VGA_R),.VGA_G(VGA_G),.VGA_B(VGA_B));

    dist_mem_gen_0 U6(
                    .a(a),
                    .d(d),
                    .dpra(dpra),
                    .clk(clk_5hz),
                    .we(we),
                    .dpo(dpo)
    );
endmodule
```

(2) 时钟分频模块

```
module clk_div(
    input clk_in,
    input en,
    input rst,
    output reg clk_out
    );
    reg [22:0]cnt;

    initial
    begin
        clk_out<=0;
        cnt<=23'b0;
    end

    always @(posedge clk_in)
    begin
        if(en)
        begin
            if(rst)
            begin
                clk_out<=0;
                cnt<=23'b0;
            end
            else if(cnt>=23'd49999)
            begin
                clk_out<=~clk_out;
                cnt<=23'b0;
            end
            else
            begin
                cnt<=cnt+23'd1;
            end
        end
    end
endmodule
```

(3) 控制模块

```
44
45      assign a=x+256*y;
46      assign u=up && ~left && ~right && ~down;          //打表表示实际控制方向，共8个方向
47      assign l=~up && left && ~right && ~down;
48      assign r=~up && ~left && right && ~down;
49      assign d=~up && ~left && ~right && down;
50      assign ul=up && left && ~right && ~down;
51      assign ur=up && ~left && right && ~down;
52      assign dl=~up && left && ~right && down;
53      assign dr=~up && ~left && right && down;
54      assign we=draw;

always @(posedge clk_5hz or posedge rst)
begin
    if(rst)
    begin
        x<=RST_X;
        y<=RST_Y;
    end
    else
    begin
        if(u)
        begin
            x<=x;
            y<=(y>0)? (y-1):y;    //碰到边界则无法继续下去
        end
        else if(l)
        begin
            x<=(x>0)? (x-1):x;
            y<=y;
        end
        else if(r)
        begin
            x<=(x<255)? (x+1):x;
            y<=y;
        end
        else if(d)
        begin
            x<=x;
            y<=(y<255)? (y+1):y;
        end
        else if(ul)
```

```
    always @(posedge clk_5hz)
    begin
        if(draw)
        begin
            data<={R_in,G_in,B_in};
        end
        else
        begin
            data<=12'b111111111111;
        end
    end
endmodule
```

(4) 获取地址模块

```
                        parameter LINE1=272;
                        parameter LINE2=528;
                        parameter LINE3=800;
                        parameter LINE4=856;
                        parameter LINE5=976;
                        parameter LINE6=1040;

                        parameter ROW1=172;
                        parameter ROW2=428;
                        parameter ROW3=600;
                        parameter ROW4=637;
                        parameter ROW5=643;
                        parameter ROW6=666;


always @(posedge clk_50mhz or posedge rst)
begin
    if(rst)
    begin
        line<=0;
        row<=0;
    end
    else
    begin
        if(line<LINE6-1)
        begin
            line<=line+1;
            row<=row;
        end
        else if(line==LINE6-1 && row!=ROW6-1)
        begin
            line<=0;
            row<=row+1;
        end
        else if(line==LINE6-1 && row==ROW6-1)
        begin
            line<=0;
            row<=0;
        end
    end
end
```

```
72      always @(posedge clk_50mhz or posedge rst)
73      begin
74          if(rst)
75          begin
76              dpra<=0;
77          end
78          else
79          begin
80              if(row<ROW1-1)
81              begin
82                  dpra<=0;
83              end
84              else if(row>=ROW1-1 && row<ROW2-1)
85              begin
86                  if(line<LINE1-1)
87                  begin
88                      dpra<=0;
89                  end
90                  else if(line>=LINE1-1 && line<LINE2-1)
91                  begin
92                      dpra<=(line-LINE1+1)+(row-ROW1+1)*256;
93                  end
94                  else
95                  begin
96                      dpra<=0;
97                  end
98              end
99              else
100             begin
```

(5) VGA 扫描代码

```
always @(posedge clk_50mhz)
begin
    if(row<ROW1-1)
    begin
        R_out<=0;
        G_out<=0;
        B_out<=0;
    end
    else if(row>=ROW1-1 && row<ROW2-1)
    begin
        if(line<LINE1-1)
        begin
            R_out<=0;
            G_out<=0;
            B_out<=0;
        end
        else if(line>=LINE1-1 && line<LINE2-1)
        begin
            R_out<=R_in;
            G_out<=G_in;
            B_out<=B_in;
        end
```

```
    always @(posedge clk_50mhz or posedge rst)
    begin
        if(rst)
            HS<=0;
        else
        begin
            if(line<LINE4-1)
            begin
                HS<=1;
            end
            else if(line>=LINE4-1 && line<LINE5-1)
            begin
                HS<=0;
            end
            else
                HS<=1;
        end
    end
always @(posedge clk_50mhz or posedge rst)
begin
    if(rst)
    begin
        line<=0;
        row<=0;
    end
    else
    begin
        if(line<LINE6-1)
        begin
            line<=line+1;
            row<=row;
        end
        else if(line==LINE6-1 && row!=ROW6-1)
        begin
            line<=0;
            row<=row+1;                         //循环，不出边界
        end
        else if(line==LINE6-1 && row==ROW6-1)
        begin
            line<=0;
            row<=0;
        end
        else if(line==LINE6-1 && row==ROW6-1)
        begin
            line<=0;
            row<=0;
        end
    end
endmodule
```

(6) 十字扫描代码

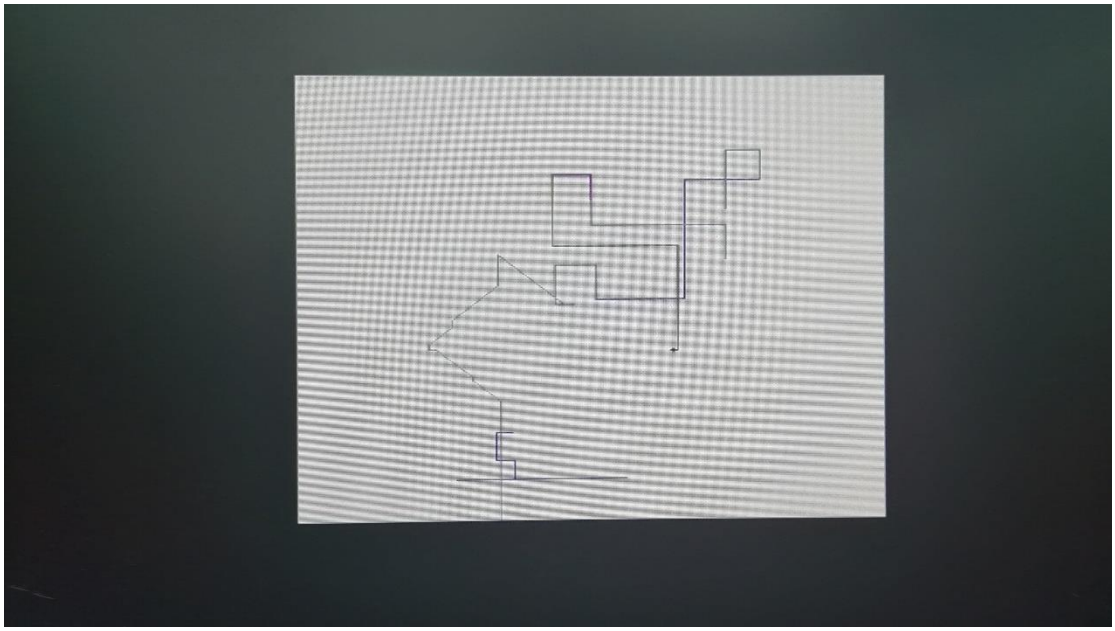
```
wire is_cross;
wire is_mark_x,is_mark_y,is_mark;
wire is_mark_u_x,is_mark_u_y,is_mark_u;
wire is_mark_l_x,is_mark_l_y,is_mark_l;
wire is_mark_r_x,is_mark_r_y,is_mark_r;
wire is_mark_d_x,is_mark_d_y,is_mark_d;

parameter LINE1=272;
parameter LINE2=528;
parameter LINE3=800;
parameter LINE4=856;
parameter LINE5=976;
parameter LINE6=1040;

parameter ROW1=172;
parameter ROW2=428;
parameter ROW3=600;
parameter ROW4=637;
parameter ROW5=643;
parameter ROW6=666;

reg [15:0]line;
reg [15:0]row;
```

```
assign is_mark_x=((line>=LINE1-1) && (line<LINE2-1) && (line-LINE1+1==x))? 1:0;    //画十字,先找到中心点
assign is_mark_y=((row>=ROW1-1) && (row<ROW2-1) && (row-ROW1+1==y))? 1:0;
assign is_mark=is_mark_x && is_mark_y;

assign is_mark_u_x=((line>=LINE1-1) && (line<LINE2-1) && (line-LINE1+1==x))? 1:0;    //中心点上部
assign is_mark_u_y=((row>=ROW1) && (row<ROW2-1) && (row-ROW1+1==y-1))? 1:0;
assign is_mark_u=is_mark_u_x && is_mark_u_y;

assign is_mark_l_x=((line>=LINE1-1) && (line<LINE2-1) && (line-LINE1+1==x-1))? 1:0;    //中心点左部
assign is_mark_l_y=((row>=ROW1) && (row<ROW2-1) && (row-ROW1+1==y))? 1:0;
assign is_mark_l=is_mark_l_x && is_mark_l_y;

assign is_mark_r_x=((line>=LINE1-1) && (line<LINE2-1) && (line-LINE1+1==x+1))? 1:0;    //中心点右部
assign is_mark_r_y=((row>=ROW1) && (row<ROW2-1) && (row-ROW1+1==y))? 1:0;
assign is_mark_r=is_mark_r_x && is_mark_r_y;

assign is_mark_d_x=((line>=LINE1-1) && (line<LINE2-1) && (line-LINE1+1==x))? 1:0;    //中心点下部
assign is_mark_d_y=((row>=ROW1) && (row<ROW2-1) && (row-ROW1+1==y+1))? 1:0;
assign is_mark_d=is_mark_d_x && is_mark_d_y;

assign is_cross=is_mark || is_mark_u || is_mark_l || is_mark_r || is_mark_d;

assign VGA_R=is_cross? 4'b0000:R_out;
assign VGA_G=is_cross? 4'b0000:G_out;
assign VGA_B=is_cross? 4'b0000:B_out;
```

```
84
85      always @(posedge clk_50mhz or posedge rst)
86      begin
87          if(rst)
88          begin
89              line<=0;
90              row<=0;
91          end
92          else
93          begin
94              if(line<LINE6-1)
95              begin
96                  line<=line+1;
97                  row<=row;
98              end
99              else if(line==LINE6-1 && row!=ROW6-1)
100             begin
101                 line<=0;
102                 row<=row+1;
103             end
104             else if(line==LINE6-1 && row==ROW6-1)
105             begin
106                 line<=0;
107                 row<=0;
108             end
109         end
110     end
111 endmodule
112
```

3. 仿真结果与下载结果：

下载结果：



4. 结果分析

下载结果正确， 斜向移动有轻微抖动

5. 实验总结

本次试验学习了 VGA 的扫描与存储器 IP 核的使用，成功在显示屏上扫描出了 256*256 的画布以及十字画笔。可以对 VGA 扫描的使用有了更深入的了解。但是生成比特流时间过长，导致 debug 的时间也非常长。