中山大学移动信息工程学院本科生实验报告

（2014 学年秋季学期）

课程名称：**数字电路实验**　　任课教师：**王军**　　助教:**李贤琴/李正**

年级&班级	13 级 09 班	专业（方向）	软件工程（移动信息工程）
学号	13354315	姓名	汤建鹏
电话	137526215446	Email	291391649@qq.com
开始日期	2015/1/3	完成日期	2014/1/11

一、 实验题目

A Basic Video Controller（VGA）

在 FPGA 中，用 Verilog 语言实现 VGA 的设计。

具体实现：在显示器实现一个很基本的屏幕保护动态视频，即一个彩色方框在屏幕内运动，当遇到边界时发生反弹，继续运动。

二、实验目的

1、学习使用 verilog 语言实现 VGA

2、理解 VGA 的原理

3、了解视频在显示屏上显示的具体过程

三、实验内容

1. 实验设计

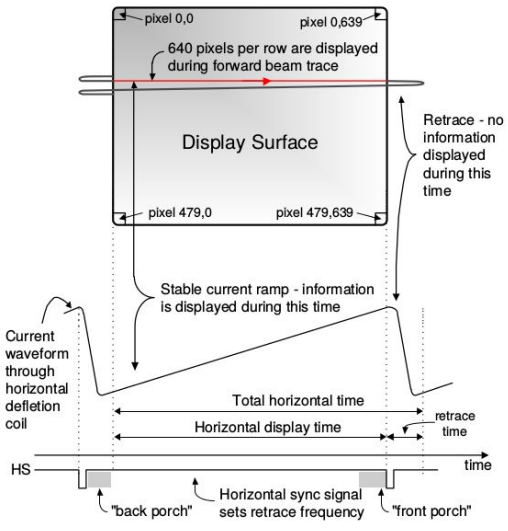


图 1-1-1：VGA 原理图

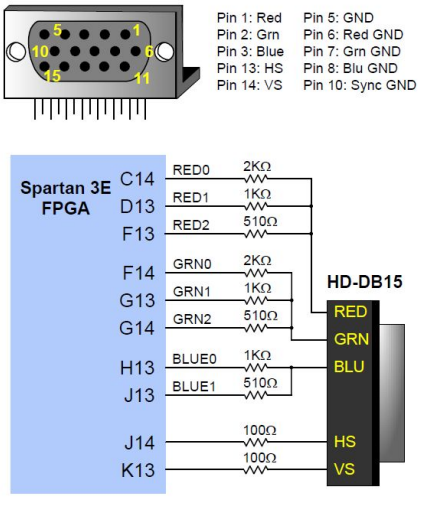
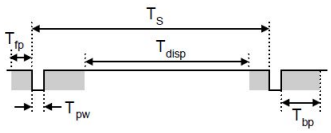


图 1-1-2：VGA 引脚图



Symbol	Parameter	Vertical Sync			Horiz. Sync	
		Time	Clocks	Lines	Time	Clks
T_S	Sync pulse	16.7ms	416,800	521	32 us	800
T_{disp}	Display time	15.36ms	384,000	480	25.6 us	640
T_{pw}	Pulse width	64 us	1,600	2	3.84 us	96
T_{fp}	Front porch	320 us	8,000	10	640 ns	16
T_{bp}	Back porch	928 us	23,200	29	1.92 us	48

图 1-1-3：行同步和场同步脉冲的时间分布

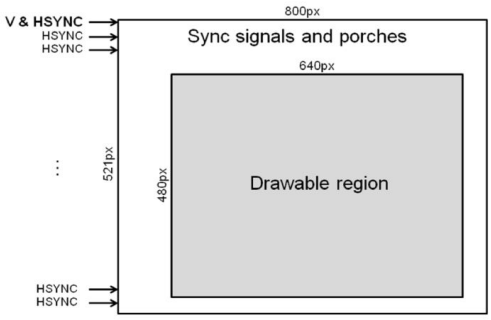


图 1-1-4：实际有效像素点的分布

这里使用的是刷新频率为 60Hz，像素点为 800（96+48+640+16）×521（2+29+480+10）的 VGA（有效显示区为 640×480），可以算出点时钟频率为 800*521*60=25008000，约为 25MHz，所以将内部时钟 4 分频后即为点时钟。再计算 drawable region 在图中的边界分别为 hbf、hfp、vbp、vfp。可知 hbf=144，hfp=784，vbp=31，vfp=511，这就之后我们的方框能够移动的四个边界。

确定好一些参数后就该考虑如何显示一个彩色方块了，首先要明确的是只有在有效显示区域内可以显示，其他方位，rgb 的值必须为 0，防止在回溯过程中在屏幕留下轨迹。所以必须设立一个 vidon 信号，在有效区域内它为 1，否则为 0。

接下来要考虑的就是如何让方块移动，在初始时我们必定要给这个方块确定边界，而让方块移动也就是让边界移动，所以可以以场向量 vsync 为触发信号，即在每一帧图像变化时使边界变化就可以是方块运动了。当然，为了使方块能反弹，就必须处理方块到达有效显示范围时的行为，让其方向运动就行了。

四、实验结果

1.综合得出的 RTL 电路图

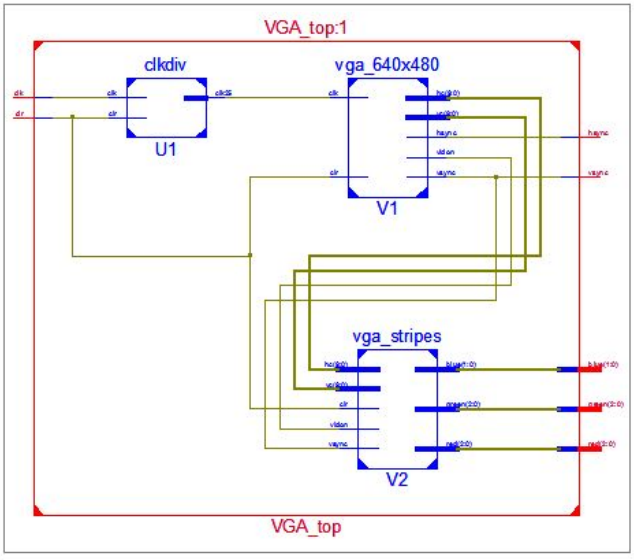


图 2：Millisecond Counter 的 RTL 图

2. VGA 显示截图

运行说明：首先按下板子上的复位键初始化，之后屏幕上就会出现一个移动的小方格了。（完整过程见附带的视频）

3. 综合报告

```
Timing Summary:
---------------
Speed Grade: -3

   Minimum period: 3.720ns (Maximum Frequency: 268.835MHz)
   Minimum input arrival time before clock: 4.520ns
   Maximum output required time after clock: 8.407ns
   Maximum combinational path delay: No path found
```

图 4：timing 报告截图

Device 报告没有什么亮点，就不截图了，这里看下时序报告，可以知道电路要求的最小周期为 3.720ns，建立时间为 4.520ns，保持时间为 8.407ns，最后的到的时钟频率是远小于 25MHz 的，所以 FPGA 必然是可以跑得动这种清晰度的视频的，并且还有很大的提升空间，跑跑 1080p 的应该也是没问题的！

五、实验感想

这次写 VGA 的实验，我明白了视频的显示原理，其实就是一个电子枪发射电子在屏幕上形成显示，但是电子枪只能让一个像素点显示，所以要像遍历一个矩阵一样遍历屏幕，扫完一屏接着一屏，利用余晖和人眼可识别的范围有限，在 1s 内显示 60 帧图片或以上，这样基本上看起看就是一个连贯的视频了。写 VGA 的关键是确定行脉冲和场脉冲以及屏幕（即有效显示区）在整个区域中的位置，还有就是计算点时钟，即扫描每一个像素点所花的时间。由于电子枪扫完一行或者一帧后要回溯，所以要预留回溯的时间防止回溯时留下痕迹，所以并不是整个矩阵都是显示区域，同时在行场时钟边缘一样呀有类似 setup time 和 hold time 的部分，所以实际的矩阵有的是不显示的，我们必须找到显示边界，确定显示区域。而时钟

可以通过 1s 扫描的像素点个数算出，1s 扫过的个数就是频率。这是基本的 VGA 显示，而如果要自己显示一个动画效果，关键还是要处理边界，这个边界指的是你要显示的物体的边界，而且这个边界应该在每一帧都更新，更新的逻辑就看你自己怎么设计你要显示的动画了。

附录：

实验设计代码	
VGA_top	**clkdiv(分频子模块)**

```
module VGA_top(
        input clk,
    input clr,
    output hsync,
    output vsync,
    output [2:0] red,
    output [2:0] green,
    output [1:0] blue
);
wire clk25,vidon;
wire [9:0] hc,vc;

clkdiv U1(
        .clk(clk),
        .clr(clr),
        .clk25(clk25)
);

vga_640x480 V1(
        .clk(clk25),
        .clr(clr),
        .hsync(hsync),
        .vsync(vsync),
        .hc(hc),
        .vc(vc),
        .vidon(vidon)
);

vga_stripes V2(
        .vidon(vidon),
            .vsync(vsync),
            .hsync(hsync),
            .clk25(clk25),
            .clr(clr),
        .hc(hc),
        .vc(vc),
        .red(red),
        .green(green),
        .blue(blue)
);
endmodule
```

```
module clkdiv(
    input clk,
    input clr,
    output clk25
    );

    reg   [24:0] q;
    //25 位计数器
    always@(posedge clk or posedge clr) begin
        if(clr)
            q <= 0;
        else
            q <= q + 1'b1;
    end

    assign clk25 = q[1];

endmodule
```

子模块 vga_640x480（计算行场同步信号）

```
module vga_640x480(
    input clk,
```

子模块 vga_stripes（显示）

```
module vga_stripes(
    input vidon,
```

```
    input clr,
    output reg hsync,
    output reg vsync,
    output reg [9:0] hc,
    output reg [9:0] vc,
    output reg vidon
);

    parameter hpixels = 10'b11001_00000; //行像素点
=800
    parameter vlines = 10'b10000_01001;   //行数=521
    parameter hbp = 10'b00100_10000;        //144
    parameter hfp = 10'b11000_10000;        //784
    parameter vbp =10'b00000_11111;         //29
    parameter vfp = 10'b01111_11111;        //10
    reg vsensable;   //enable for the vertical counter

    //行同步信号计数器·
    always@(posedge clk)
      begin
        if(clr)
                hc <= 0;
        else
        begin
            if(hc == hpixels - 1)
                begin
                    hc <= 0;
                    vsensable <= 1;
                    //enable teh vertical counter
to increase
            end
            else
            begin
                    hc <= hc + 1'b1;
                    vsensable <= 0; //leave the
vsenable off
                end
        end
    end


    //产生 hsync 脉冲
    //当 hc 乾 127 的时候，行同步信号为低电幀
      always@(*)
```

```
    input vsync,
    input hsync,
    input clk25,
    input clr,
   input [9:0] hc,
   input [9:0] vc,
   output reg [2:0] red,
   output reg [2:0] green,
   output reg [1:0] blue
   );

    parameter hbp = 10'b00100_10000;
//144
   parameter hfp = 10'b11000_10000;
//784
   parameter vbp =10'b00000_11111;
//31
   parameter vfp = 10'b01111_11111;
//511
    reg [9:0] left, right, top, botton;
    reg hmove, lmove;
    always @(posedge vsync or posedge
clr)
    begin
       if(clr)
       begin
           hmove <= 1;
           lmove <= 1;
           left <= hbp+10'b0000_00101;
           right <=
hbp+10'b00010_00100;
           top <= vbp+10'b0000_00101;
           botton <=
vbp+10'b00001_10100;
       end
       else
       begin
           if(left<=hbp+3'b100)
               hmove <= 1;
           else if(right>=hfp-3'b100)
               hmove <= 0;
           else
               hmove <= hmove;
           if(top<=vbp+3'b100)
```

```
        begin
            if(hc < 96)
                hsync <= 0;
            else
                hsync <= 1;
    end

    //场同步信号计数器
    always@(posedge clk)
        begin
            if(clr)
                vc <= 0;
            else
                begin
                if(vsensable == 1)
                        begin
                    if(vc == vlines - 1)
                        vc <= 0;
                    else
                        vc <= vc + 1'b1;
                end
            end
    end

    //产生 vsync 脉冲
    //当 hc 书?聘?时候，场同步脉冲为低电帧
    always@(*)
        begin
            if( vc < 2)
                vsync <= 0;
            else
                vsync <= 1;
    end

    always@(*)
        begin
            if((hc < hfp)&&(hc >= hbp)&&(vc < vfp)&&(vc >= vbp))
                vidon <= 1;
            else
                vidon <= 0;
    end

endmodule
```

```
                lmove <= 1;
            else
if(botton>=vfp-3'b100)
                lmove <= 0;
            else
                lmove <= lmove;

            left <= ((hmove) ? (left+1'b1) : (left-1'b1));
            right <= hmove ? right+1'b1 : right-1'b1;
            top <= lmove ? top+1'b1 : top-1'b1;
            botton <= lmove ? botton+1'b1 : botton-1'b1;
        end
    end

    always@(posedge clk25)
    begin
        if(vidon == 1)
          begin

if((hc>=hbp&&hc<hbp+4)||(hc<hfp&&hc>=hfp-4)||(vc>=vbp&&vc<vbp+4)||(vc<vfp&&vc>=vfp-4))
                begin
                    red   <= 3'b111;
                    green <= 3'b111;
                    blue  <= 2'b11;
                end
             else if((hc >= left)&&(hc <= right)&&(vc >= top)&&(vc <= botton))
                begin
                    red   <= 3'b111;
                    green <= 3'b000;
                    blue  <= 2'b00;
                end
                else
                begin
                    red   <= 0;
                    green <= 0;
                    blue  <= 0;
                end
```

```
        end
        else
          begin
            red   <= 0;
            green <= 0;
            blue  <= 0;
          end
    end

endmodule
```

实验管脚分配

```
NET "clr"    LOC = "B8";
NET "clk"    LOC = "V10";
NET "red[0]" LOC = "U7";
NET "red[1]" LOC = "V7";
NET "red[2]" LOC = "N7";
NET "green[0]" LOC = "P8";
NET "green[1]" LOC = "T6";
NET "green[2]" LOC = "V6";
NET "blue[0]"    LOC = "R7";
NET "blue[1]"    LOC = "T7";
NET "hsync" LOC = "N6";
NET "vsync" LOC = "P7";
```

提交文件清单：

文件名: