|  |  |
| --- | --- |
| logo[1] | 杭州电子科技大学 |

实验报告

Experimental report

|  |  |
| --- | --- |
| **实验课程名称** | **数字系统与处理器** |
| **实验序号** | 7 |
| **实验内容** | VGA显示控制电路设计 |
| **班级** | 18184111 |
| **姓名** | 杨臻 |
| **学号** | 18052032 |
| **指导教师** | 黄继业 |

二○一九 年 十一月 八日

## 实验目的与要求

实验名称VGA显示控制电路设计

## 实验原理

## 通用VGA显示卡系统主要由控制电路、显示缓存区和视频 BIOS程序三个部分组成。控制电路如图1所示。控制电路主要完成时序发生、显示缓冲区数据操作、主时钟选择和 D／A转换等功能；显示缓冲区提供显示数据缓存空间；视频BIOS作为控制程序固化在显示卡的 ROM中。

## 1.1 VGA时序分析

## 通过对VGA显示卡基本工作原理的分析可知，要实现VGA显示就要解决数据来源、数据存储、时序实现等问题，其中关键还是如何实现VGA时序。VGA的标准参考显示时序如图2所示。行时序和帧时序都需要产生同步脉冲(Sync a)、显示后沿(Back porch b)、显示时序段(Display interval c)和显示前沿(Front porch d)四个部分。几种常用模式的时序参数如表1所示。

## 1.2 VGA时序实现

## 首先，根据刷新频率确定主时钟频率，然后由主时钟频率和图像分辨率计算出行总周期数，再把表1中给出的a、b、c、d各时序段的时间按照主计数脉冲源频率折算成时钟周期数。在CPLD中利用计数器和RS触发器，以计算出的各时序段时钟周期数为基准，产生不同宽度和周期的脉冲信号，再利用它们的逻辑组合构成图2中的a、b、c、d各时序段以及D／A转换器的空白信号BLANK和同步信号SYNC。

## 1.3 读SRAM地址的产生方法

## 主时钟作为像素点计数脉冲信号，同时提供显存SRAM的读信号和D／A转换时钟，它所驱动的计数器的输出端作为读SRAM的低位地址。行同步信号作为行数计数脉冲信号，它所驱动的计数器的输出端作为读SRAM的高位地址。由于采用两片SRAM，所以最高位地址作为SRAM的片选使用。由于信号经过CPLD内部逻辑器件时存在一定的时间延迟，在CPLD产生地址和读信号读取数据时，读信号、地址信号和数据信号不能满足SRAM读数据的时序要求。可以利用硬件电路对读信号进行一定的时序调整，使各信号之间能够满足读SRAM和为DAC输入数据的时序要求。

## 1.4 数据宽度和格式

## 如果VGA显示真彩色BMP图像，则需要R、G、B三个分量各8位，即24位表示一个像素值，很多情况下还采用32位表示一个像素值。为了节省显存的存储空间，可采用高彩色图像，即每个像素值由16位表示，R、G、B三个分量分别使用5位、6位、5位，比真彩色图像数据量减少一半，同时又能满足显示效果。

## 实验内容

实验步骤：

1. 配置Vivado，建立工程文件
2. 编写程序代码
3. 编写仿真代码
4. 执行仿真
5. 编写约束文件
6. 编译
7. 烧录程序
8. 测试

设计框图：

第一次实验：

第一次实验所用的开发板频率为100MHz，故在原有程序基础上增加了四分频代码，代码如下：

module vga\_disp

(

input clk,

input reset,

output VGA\_HSYNC,

output VGA\_VSYNC,

output reg[11:0] VGA\_D

);

reg[9:0] hcnt;

reg[9:0] vcnt;

reg hs;

reg vs;

reg clk25M;

reg clk50M;

wire[2:0] rgb;

wire[9:0] x;

wire[9:0] y;

wire dis\_en;

assign x = hcnt;

assign y = vcnt;

assign VGA\_VSYNC = vs;

assign VGA\_HSYNC = hs;

assign dis\_en = {x<10'd640 && y<10'd480};

assign rgb = x[8:6];

always @(posedge clk or posedge reset) begin //获得50MHz频率

if(reset)

clk50M <= 1'b0;

else

clk50M <= ~clk50M;

end

always @(posedge clk50M or posedge reset) begin

if(reset)

clk25M <= 1'b0;

else

clk25M <= ~clk25M;

end

always@(posedge clk25M or posedge reset) begin

if(reset)

hcnt <= 1'b0;

else begin

if(hcnt < 800)

hcnt <= hcnt+1'b1;

else

hcnt <= 1'b0;

end

end

always @(posedge clk25M or posedge reset)begin

if(reset)

vcnt <= 1'b0;

else begin

if(hcnt == 10'd640 + 10'd8)begin

if(vcnt < 10'd525)

vcnt <= vcnt + 1'b1;

else

vcnt <= 1'b0;

end

end

end

always @(posedge clk25M or posedge reset)begin

if(reset)

hs <= 1'b1;

else begin

if((hcnt >= 640+8+8) & (hcnt < 640+8+8+96))

hs <=1'b0;

else

hs <= 1'b1;

end

end

always @(vcnt or reset) begin

if(reset)

vs <= 1'b1;

else begin

if((vcnt >= 480+8+2) && (vcnt < 480+8+2+2))

vs <= 1'b0;

else

vs <= 1'b1;

end

end

always @(posedge clk25M or posedge reset)begin

if(reset)

VGA\_D <= 1'b0;

else begin

if(hcnt < 10'd640 & vcnt < 10'd480 && dis\_en) begin

VGA\_D[11:8] <= rgb[0]?4'hf:0;

VGA\_D[ 7:4] <= rgb[1]?4'hf:0;

VGA\_D[ 3:0] <= rgb[2]?4'hf:0;

end

else begin

VGA\_D <= 1'b0;

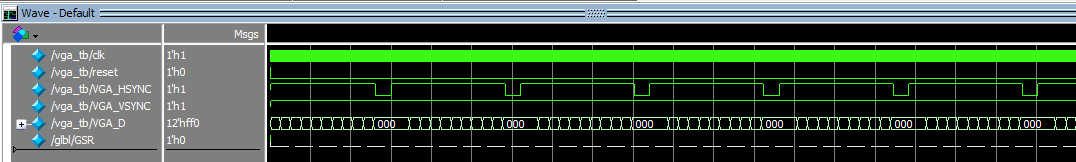
end

end

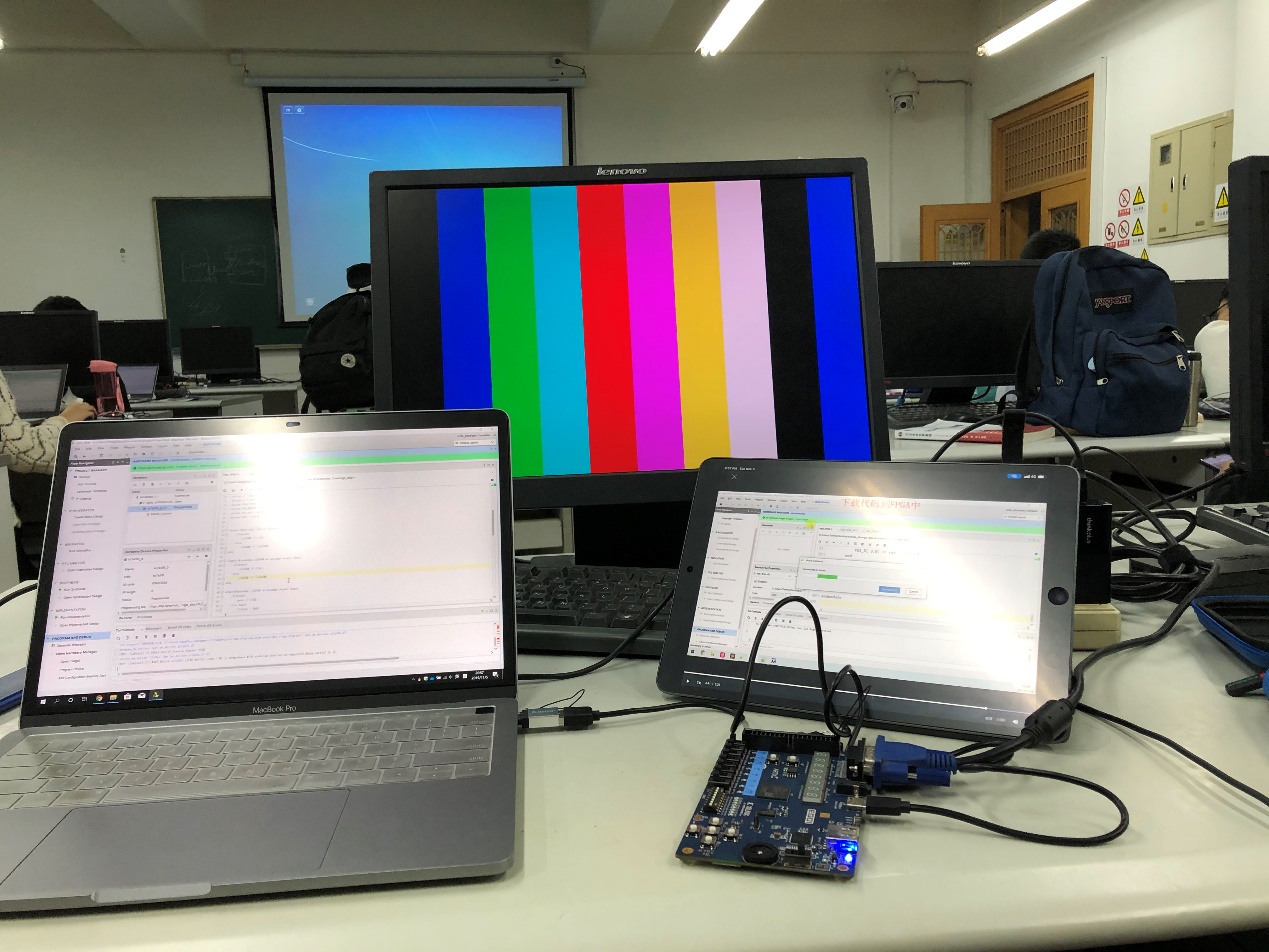
end

endmodule

仿真波形：



硬件演示：



第二次实验：

按照要求完成了由学号对应的彩色条纹

代码如下：

module vga\_disp(

input clk,

input reset,

output VGA\_HSYNC,

output VGA\_VSYNC,

output reg[11:0] VGA\_D

);

reg [9:0] hcnt;

reg [9:0] vcnt;

reg hs;

reg vs;

reg clk25M;

reg [2:0] rgb;

wire [9:0] x;

wire [9:0] y;

wire dis\_en;

assign x=hcnt;

assign y=vcnt;

assign VGA\_VSYNC=vs;

assign VGA\_HSYNC=hs;

assign dis\_en=(x<10'd640 && y<10'd480);

always @(posedge clk or negedge reset) begin

if(!reset) begin

clk25M<=1'b0;

end

else

clk25M<=~clk25M;

end

always @(posedge clk25M or negedge reset) begin

if(!reset)

hcnt<=1'b0;

else begin

if(hcnt<800)

hcnt<=hcnt+1'b1;

else

hcnt <=1'b0;

end

end

always @ (posedge clk25M or negedge reset)begin

if(!reset)

rgb<=0;

else

case(hcnt[8:6])

0:rgb<=1;

1:rgb<=8;

2:rgb<=0;

3:rgb<=5;

4:rgb<=2;

5:rgb<=0;

6:rgb<=3;

7:rgb<=2;

default: rgb<=0;

endcase

end

always @(posedge clk25M or negedge reset) begin

if(!reset)

vcnt<=1'b0;

else begin

if(hcnt==10'd640 +10'd8)begin

if(vcnt<10'd525)

vcnt<=vcnt+1'b1;

else

vcnt<=1'b0;

end

end

end

always @(posedge clk25M or negedge reset) begin

if(!reset)

hs<=1'b1;

else begin

if((hcnt>=640+8+8) & (hcnt<640+8+8+96))

hs<=1'b0;

else

hs<=1'b1;

end

end

always @(vcnt or reset) begin

if(!reset)

vs<=1'b1;

else begin

if((vcnt>=480+8+2) && (vcnt<480+8+2+2))

vs<=1'b0;

else

vs<=1'b1;

end

end

always @(posedge clk25M or negedge reset) begin

if(!reset)

VGA\_D<=1'b0;

else begin

if(hcnt<10'd640 & vcnt <10'd480 && dis\_en)

VGA\_D <={rgb[0],rgb[0],rgb[0],rgb[0],rgb[1],rgb[1],rgb[1],rgb[1],rgb[2],rgb[2],rgb[2],rgb[2]};

else begin

VGA\_D<=1'b0;

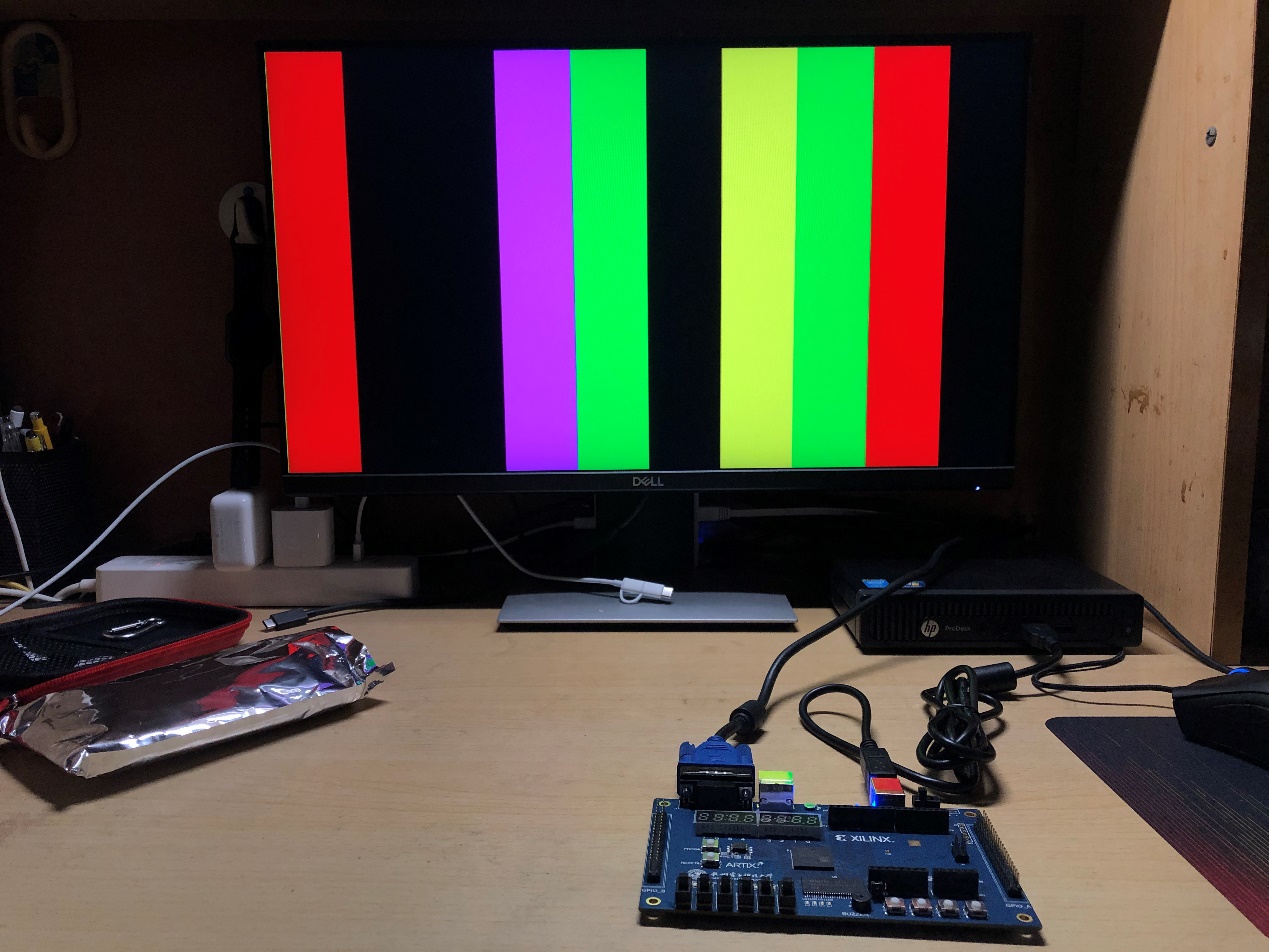
end

end

end

endmodule

硬件演示：



## 总结与体会

这一章节的实验我首先用教程中的代码进行试验，结果屏幕没有显示，研究了VGA的输出原理之后，发现是实际输出频率与显示器支持的频率不符，而教程中用的开发板晶振频率为50MHz，我所使用的开发板为100MHZ，故加上了一个四分频代码。修改实验要求后，我根据VGA的输出原理，将源程序改成了符合要求的版本。