



**数字集成电路设计实验报告**

|  |  |
| --- | --- |
| 实验题目： | 实验二 AHB VGA外设设计与实现 |
|  |  |
| 学生学号： | 35320212200396 |
| 学生姓名： | 黄腾熙 |
| 院 系： | 电子科学与技术学院 |
| 专 业： | 集成电路设计与集成系统 |

# 实验二 AHB VGA外设设计与实现

# 一、实验原理

本实验旨在设计并实现一个基于AHB总线的VGA外设，实现SoC（系统级芯片）对VGA显示器的图像显示功能，包括文字显示、像素绘制，并通过软件实现基本操作和调试。完成任务的同时，深入理解SoC架构设计、VGA显示原理、AHB接口及相关硬件调试方法。

# 二、实验步骤

1、软件实现清屏（文本和图像分开实现）、图像区画直线、矩形和显示位图

2、硬件增加文本当前位置（光标）寄存器XCursor和YCursor，可以随意定位到文本区任意位置开始打印字符

3、硬件增加文本当前颜色寄存器TextColor，可以改变打印字符的颜色

4、硬件增加显示配置寄存器TextRowChars， 可以调整文本区每行的字符数。相应的图像显示区大小也跟着调整。如果这个数值为0那整个屏幕都是图像显示。如果此数值大于等于屏幕一行能显示的最大字符数，那么整个屏幕都是文本显示

5、硬件增加显示配置寄存器ImageZoom, 可以调整图像区的分辨率缩放倍数。缩小倍数为2^n， 即0表示原始VGA分辨率； 1表示缩小2倍；2表示缩小4倍。对于其它值写入无效，不修改当前分辨率。例如基本实验是按照缩小4倍的比例，分辨率为100x120

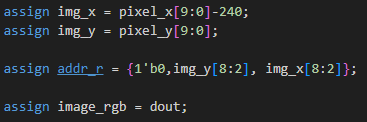
6、硬件把颜色方案改成RGB444，实现4096色

# 三、实验结果分析

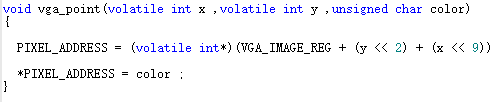
**（一）软件实现清屏（文本和图像分开实现）、图像区画直线、矩形和显示位图**

1、首先软件实现图像区画点函数

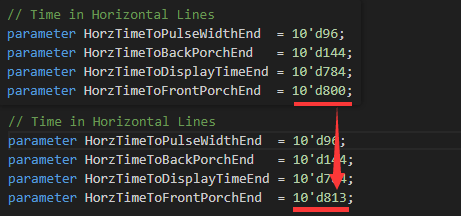
阅读代码，发现以下代码控制了图像区的RAM读取，其中addr\_r即为屏幕内任意像素的寄存器，向其中写入的颜色信息会被更新至RAM中



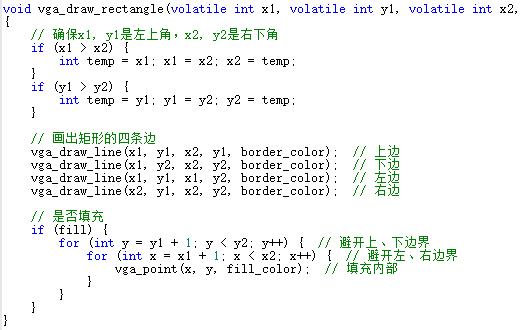
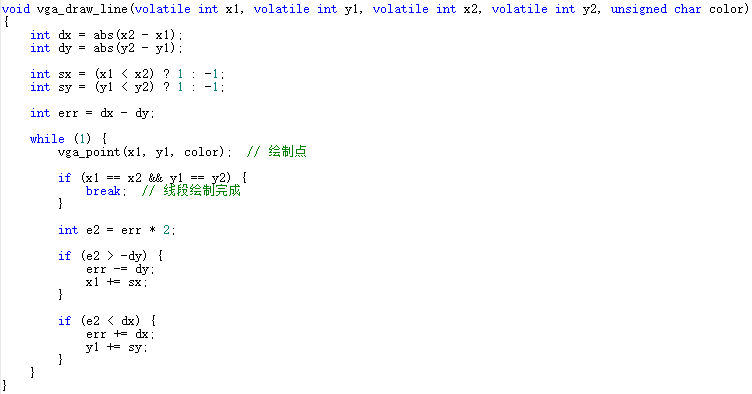
因此，可以根据Verilog硬件代码，编写如下函数，即可实现绘制任意颜色、任意位置的像素点



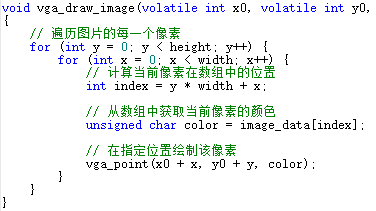
在测试画点函数时，发现在边界位置有部分点显示不完整，因此微调了vga\_sync文件中的参数，使得边界像素点能与显示屏边界相贴合



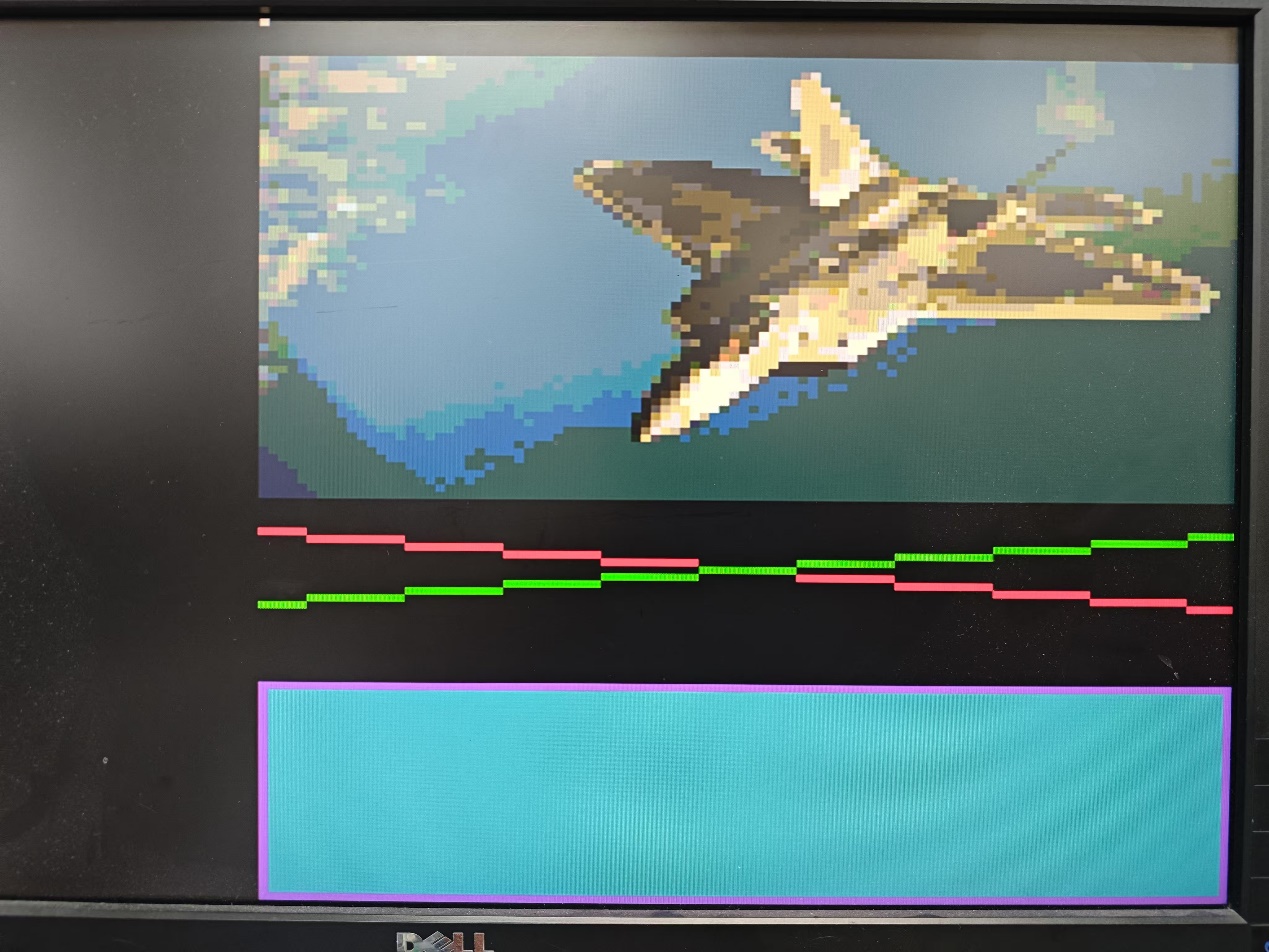
实现画点函数后，画线、画矩形、显示位图均可较简单的实现



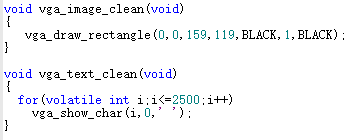
显示位图时，需要对位图进行取模，取模方法如下，将图片读取至取模软件中，即可在显示屏上正常显示图片

最后图像区的显示测试效果如下，能正常显示位图、画线、画矩形

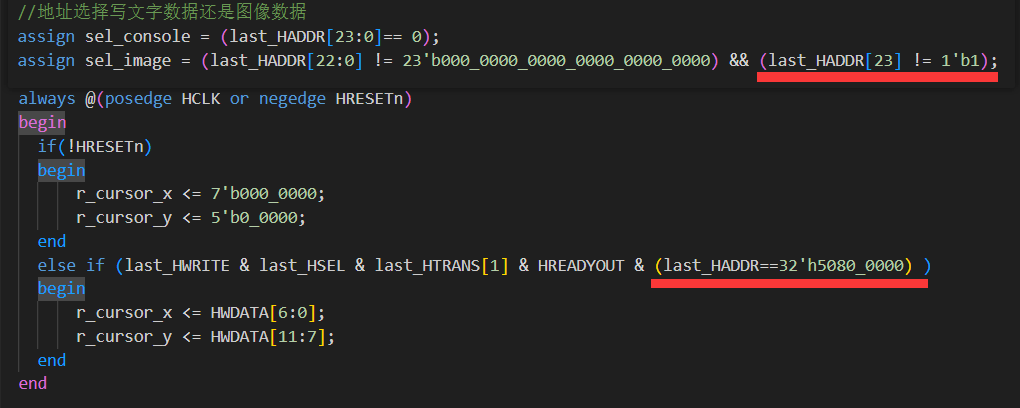


图像部分的清屏也较为简单，向所有像素点中写入黑色即可，而文本部分有自动移动光标的功能，只需向文本区输入足够多的空格，即可实现清屏

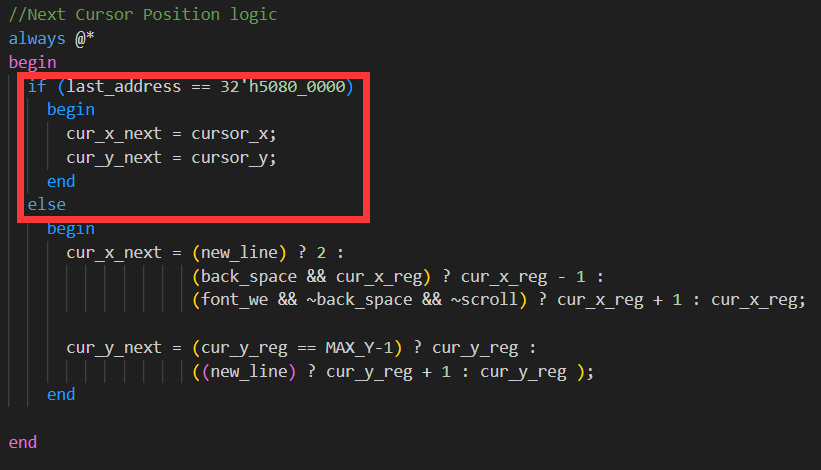


**（二）硬件增加文本当前位置（光标）寄存器XCursor和YCursor，可以随意定位到文本区任意位置开始打印字符**

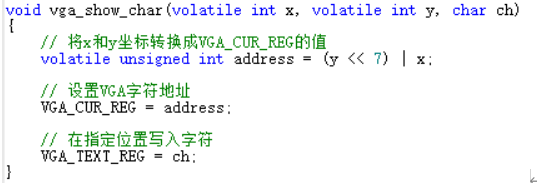
1、将图像区地址空间末端的部分地址分配给光标，用于光标坐标的写入。光标坐标的数据利用一个周期的HWDATA传递，其中0-6位为x坐标，7-11位为y坐标

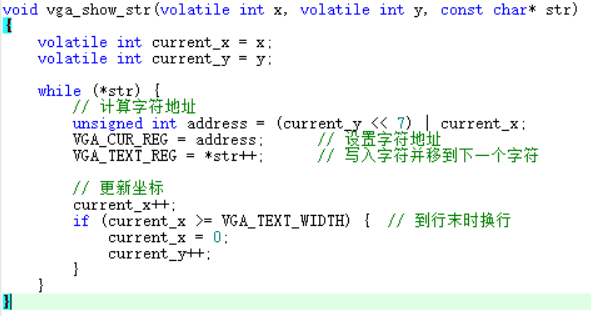


2、随后在vga\_console模块里，将对应地址的数据写入cur\_next寄存器，即可控制字符写入的坐标，同时考虑字符串的显示，保留cur\_next原本的逻辑

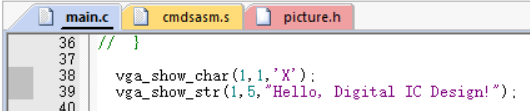


3、编写C语言函数并封装，此时地址数据的转换与之前的设定相对应（0-6位为x坐标，7-11位为y坐标），同时根据字符显示函数即可编写字符串显示函数





4、将软件编译并烧录后进行测试



经过测试，能在自定义坐标位置下，显示所输入的字符

电脑萤幕

描述已自动生成

**（三）硬件增加文本当前颜色寄存器TextColor，可以改变打印字符的颜色**

1、与字符位置类似，在VGASYS模块中为字体颜色分配地址0x5080\_0004

文本

描述已自动生成

文本

描述已自动生成

2、在vga\_console中添加相关逻辑，使得颜色值能被正确读取

文本

描述已自动生成

3、对硬件进行综合并编写对应的软件测试功能

图形用户界面, 文本, 应用程序, 聊天或短信

描述已自动生成

4、显示效果如下，能改变所打印字符的颜色

屏幕上有字

描述已自动生成

**（四）硬件增加显示配置寄存器TextRowChars， 可以调整文本区每行的字符数。相应的图像显示区大小也跟着调整。如果这个数值为0那整个屏幕都是图像显示。如果此数值大于等于屏幕一行能显示的最大字符数，那么整个屏幕都是文本显示**

1、阅读代码后，发现代码中有包含文本区宽度的参数，为这个参数分配一个寄存器和地址，分配的地址为0x5080\_0008

图形用户界面, 文本

描述已自动生成文本

描述已自动生成

文本

描述已自动生成

2、同时，在原本的图像区中，显示的分辨率最大为128\*128，如果文本区宽度小于240，那么多出的图像区将显示错误的信息，因此需要为图像区显示地址的x地位宽从原来的7位增加至8位，这样才能实现全屏显示图像的要求。

在此将所有与图像相关的地址位宽全增加一位

文本

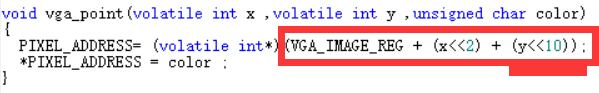
描述已自动生成

双端口RAM的地址位宽也需要增加，修改后显示的最大分配率变为256\*128

文本

描述已自动生成

3、修改Verilog硬件后，软件中对于地址的换算也需要做对应的调整



4、修改后综合硬件进行测试

运行如下程序，测试全图像区显示



显示效果如下，可见无文本区，全屏正常显示图像

电脑屏幕

描述已自动生成

运行如下程序，测试文本区宽度调节

文本

描述已自动生成

能正常修改文本区宽度

电脑屏幕

描述已自动生成

运行如下程序，测试全文本区显示

文本

描述已自动生成

可见，可以在全屏任意位置显示文字

屏幕上有个电视

描述已自动生成

经过以上测试，可见文本区与图像区大小可以自由调节，并均能正常执行程序，显示相应的字符或图像

**（四）硬件增加显示配置寄存器ImageZoom, 可以调整图像区的分辨率缩放倍数。缩小倍数为2^n， 即0表示原始VGA分辨率； 1表示缩小2倍；2表示缩小4倍。对于其它值写入无效，不修改当前分辨率。例如基本实验是按照缩小4倍的比例，分辨率为100x120**

1、要实现图像区的高分辨率显示，首先需要为图像数据分配足够大小的RAM，要实现640\*480的全屏图像显示，需要分配512\*1024\*8，即512kb大小的RAM

文本

描述已自动生成文本

描述已自动生成

2、分配对应大小的RAM后进行综合，发现无法进行调试，但查看时序报告也未发现异常

因此改用DCM IP核来输出时钟，再进行综合可以发现实际的时序问题

文本

描述已自动生成图形用户界面, 表格

描述已自动生成

修改时钟后进行综合，发现布线综合的时序不符合要求，分析发现导致时序问题的原因是RAM占据面积过大，导致布线过长，路径延时大于时钟周期（50MHz）

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

因此需要降低核心频率，在测试后选择以25MHz作为ARM软核的核心频率。同时，为保持VGA输出的时序，需要用50MHz时钟驱动VGA的输出模块

文本

描述已自动生成文本

描述已自动生成

作出如上修改后，能正常进行综合

4、现在需要对vga\_image模块做出对应修改，以实现高分辨率的显示

阅读代码，发现将img\_x和img\_y的位数做对应限制即可实现不同分辨率

在VGASYS中先为image\_zoom寄存器分配地址

文本

描述已自动生成

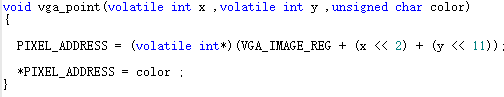
之后在vga\_image中添加根据image\_zoom的写入值修改分辨率的逻辑

其中在xy像素坐标中加入若干位0是为了使画点函数能在不同分辨率下通用

屏幕上有字

描述已自动生成

根据以上硬件逻辑，只需将y坐标比x坐标多移动9位即可



5、软件修改

在C代码中添加对应的宏定义并编写相应程序后



将各个参数设置集中至VGA\_INIT中

文本

描述已自动生成

测试程序如下

文本

中度可信度描述已自动生成

6、综合Verilog并烧录C程序，可以在显示屏上看到如下图像，可以看到图像以640\*480的分辨率正常显示，全屏均可作为显示区域

电视上的卡通

低可信度描述已自动生成

调节分辨率通过改变INIT函数的参数即可实现

图片包含 文本

描述已自动生成

2倍分辨率显示效果如下

电脑萤幕画面

中度可信度描述已自动生成

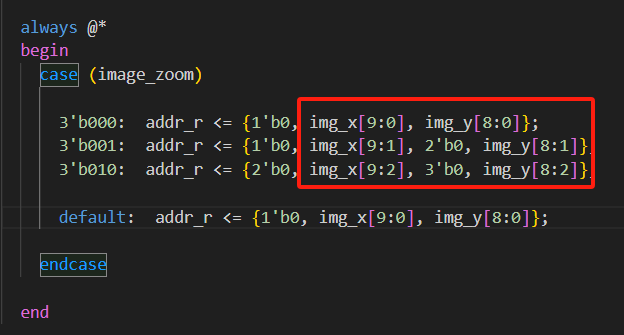
四倍分辨率显示效果如下



图片包含 电子, 室内, 显示器, 电脑

描述已自动生成

7、（改进）修改图像区的寻址方式，将x和y的坐标交换，图像区ram的大小可以降低至320kb（640\*512）



文本

描述已自动生成

软件部分仅需交换x和y的坐标即可



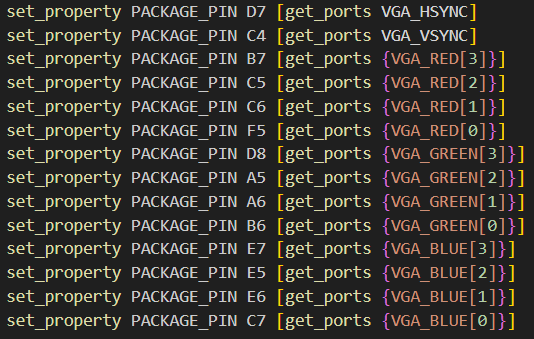
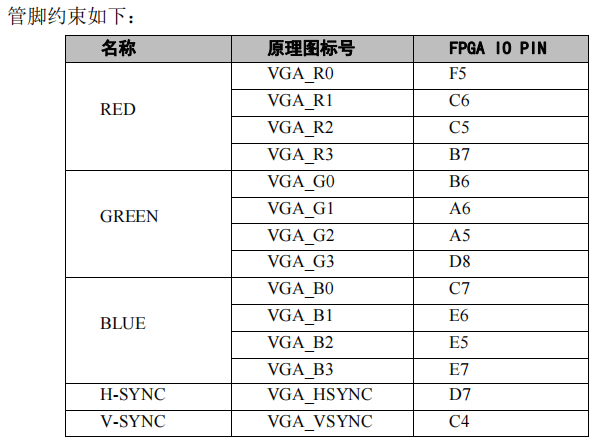
经测试能正常显示

显示效果与之前无异，但所用的硬件资源明显较少

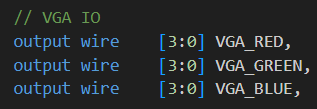
**（四）硬件把颜色方案改成RGB444，实现4096色**

由于4096色实现时参数调整过多，因此实验文件单独存放与Lab2\_VGA\_4096文件夹

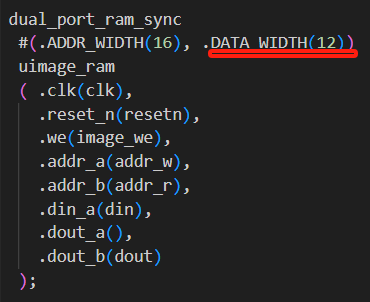
1、要实现4096色首先需要根据数据手册添加管脚约束



2、添加管脚约束后将所有原本8位RGB的端口全修改为12位RGB



3、同时，需要将图像区的数据宽度由8位改至12位，这样才能正常显示



4、经过测试，能正常显示4096色，以下为4096色全色显示效果

