2020-11-29

秦嘉余

191220088

1348288404@qq.com

数字电路实验十一 字符输入界面

目录

[1.实验目的 1](#_Toc57580279)

[2.实验原理 1](#_Toc57580280)

[1.字符显示 1](#_Toc57580281)

[2.扫描显示 1](#_Toc57580282)

[3.显存读写 2](#_Toc57580283)

[3.实验环境/器材 2](#_Toc57580284)

[4. 显示器交互界面 2](#_Toc57580285)

[1. 设计思路 2](#_Toc57580286)

[2. 实验代码 2](#_Toc57580287)

[3.RTL视图 4](#_Toc57580288)

[4.引脚分配 4](#_Toc57580289)

[5.实验结果 5](#_Toc57580290)

[5.实验实验中遇到的问题及解决办法 5](#_Toc57580291)

1.实验目的

本实验的主要目的是本实验将利用前面实现过的键盘和显示器功能来搭建一个简单的字符输入界面，通过该系统的实现深入理解多个模块之间的交互和接口的设计。

2.实验原理

## 1.字符显示

字符显示界面只在屏幕上显示ASCII 字符，其所需的资源比较少。首先，ASCII 字符用7bit 表示，共128 个字符。大部分情况下，我们会用8bit 来表示单个字符，所以一般系统会预留256 个字符。我们可以在系统中预先存储这256 个字符的字模点阵，每个字符高为16 个点，宽为9 个点。因此单个字符可以用16 个9bit数来表示，每个9bit 数代表字符的一行，对应的点为“1” 时显示白色，为“0”时显示黑色。因此，我们只需要256 × 16 × 9 ≈37kbit 的空间即可存储整个点阵。

## 2.扫描显示

我们之前已经实现了VGA 控制模块，该模块可以输出当前扫描到的行和列的位置信息，我们只需要稍加改动，即可让其输出当前扫描的位置对应30×70 字符阵列的坐标(0 ≤ x ≤ 69, 0 ≤ y ≤ 29)。利用该坐标，我们可以查询字符显存，获取对应字符的ASCII 编码。利用ASCII 编码，我们可以查询对应的点阵ROM，再根据扫描线的行和列信息，可以知道当前扫描到的是字符内的哪个点。这时，可以根据该点对应的bit 是1 还是0，选择输出白色还是黑色。

## 3.显存读写

对于键盘输入，我们可以复用之前实现的键盘控制器。在键盘有输入的时候对字符显存进行改写，将按键对应的ASCII 码写入显存的合适位置，这样输入就可以直接反馈到屏幕上了。

3.实验环境/器材

本次实验的环境为Quartus17.1版本

本次实验的器材为DE10 Standard开发板

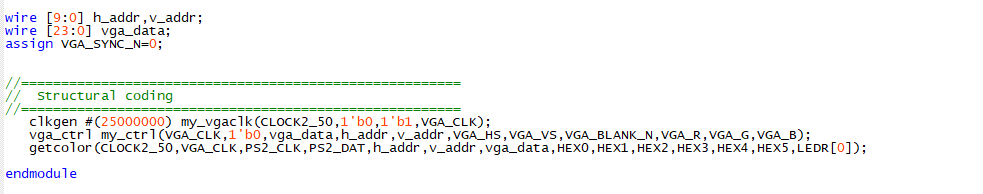
4. 显示器交互界面

## 1. 设计思路

本次实验主要是对之前模块的复用，在对于交互方面的代码进行重写。由于显示字符有大量的数据需要存储，所以此次实验需要多个rom或者ram来存储各种数据，包括界面上各个位置字符的ascii码数据，字符的显示数据，以及freq频率的数据等等。之后，复用原来的键盘代码改进，根据键码向对银行rom中写入数据，然后显示屏不断读取rom中的数据，根据ascii字符以及行列数显示黑白的像素点即可。

## 2. 实验代码

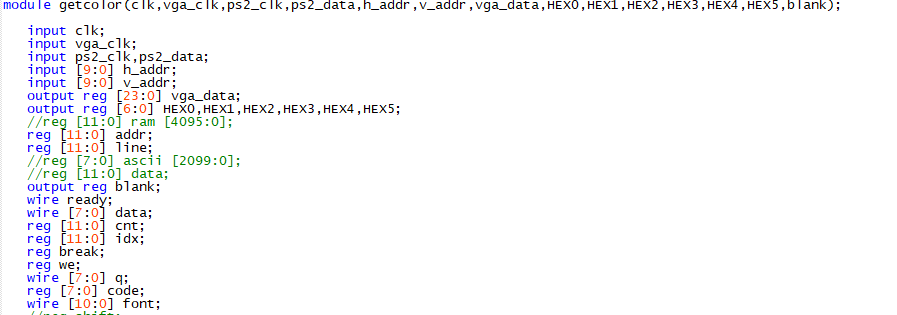
顶层模块如下，与之前的实验9代码基本相同，主要的修改集中于getcolor模块中。



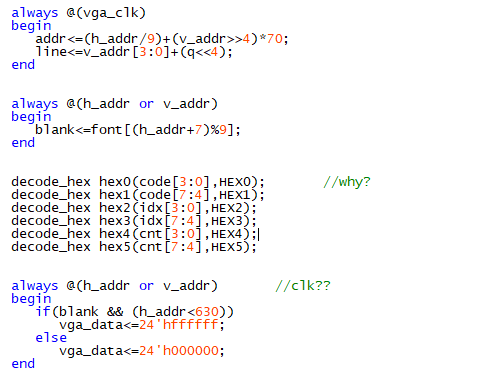
对于getcolor模块，声明的rom，ram如下：a为键盘键码与ascii码的映射ram，与ps2\_keyboard配合使用，display模块为显示rom，记录显示屏上每个位置的字符的ascii码信息，font\_data模块为每个字符显示的颜色rom，是显示屏上每个像素的显示需要的信息



getcolor的接口如下：



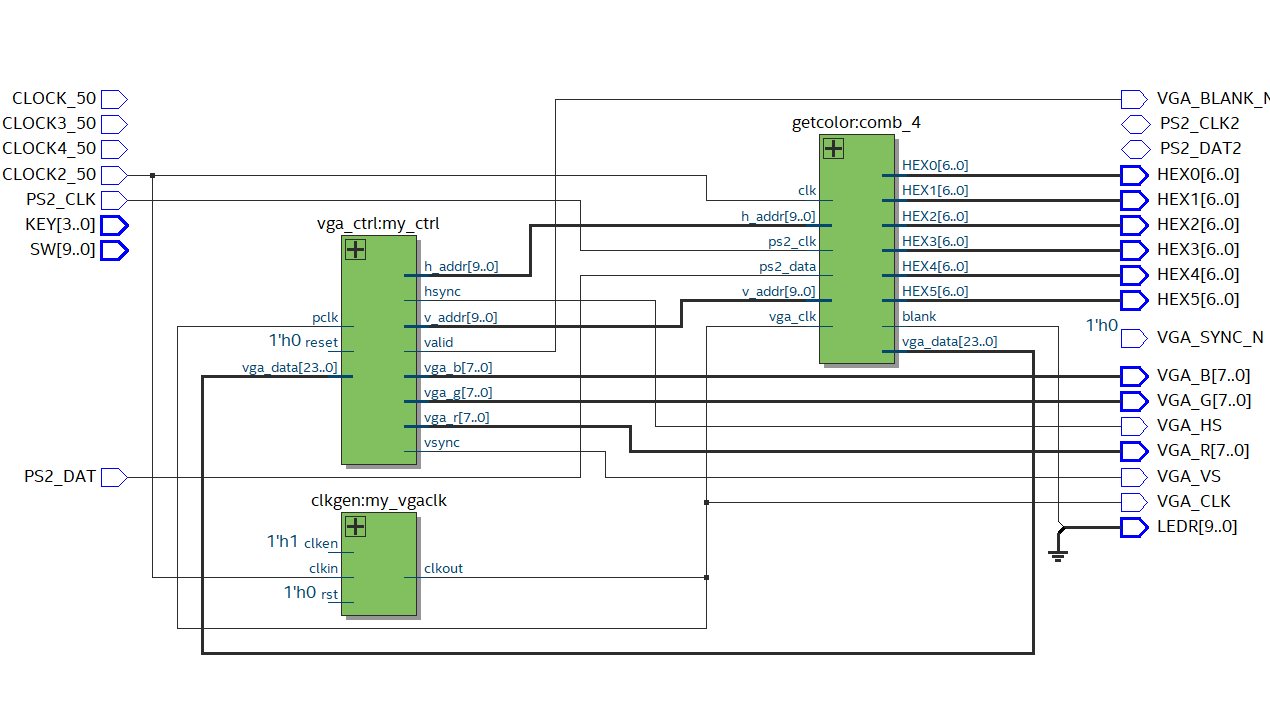
程序利用如下代码计算对应点的像素信息，传给显示屏显示，其中blank为当前像素前的明暗信息，addr为当前的ascii字符索引，line为当前显示的ascii字符的显示数据的当前行



we为display这个rom的读写信号，idx与cnt为分别为当前输入位置的索引与下一个需要输入的位置的索引



## 3.RTL视图



## 4.引脚分配

该实验为system软件自动分配的引脚，所以不再展示、

## 5.实验结果

本实验已在开发板上验收

5.实验实验中遇到的问题及解决办法

问题一：

实验中，显示的时序逻辑十分重要，如果读写的顺序不合适，会导致输入输出不同步，从而显示出现故障，在此次实验中，选择vga 的时钟比较合适，时钟速度较为合理

问题二：

由于在本次实验中，各种位置信息的计算，以及rom的读写都有一定的时间消耗，所以在显示字符时会导致字符的显示出现偏移现象，这种现象由于本身时序逻辑的复杂性较难消除，为了解决这个问题，在显示字符时，加上一个偏移量读取像素点的明暗信息，抵消这一偏移，可以完美的显示字符

问题三：

本次实验中很多rom的内容量较大，不使用IP核来生成rom，直接使用寄存器来读取，会发生无法写入读取的问题，换用合适的rom即可解决这一问题