**实验四 Verilog编程复杂实验**

本次报告的提交地址为：https://send2me.cn/dBr3uW-3/SsOs9SWaLbQXiA

提交截止时间为：2025-5-25晚上11点

1. **设计一个序列信号“1011”检测电路。**

clk为时钟输入端，时钟周期为10us。din是信号输入端。dout是输出端。rst复位信号，当rst为1时，开始进行序列信号的检测。否则不检测。随着clk时钟的进入，din引脚输入序列信号(时钟clk的上升沿处信号有效)。当输入到电路中的序列信号中出现“1011”序列时（允许位重复使用），输出1，否则输出0。请用verilog代码设计该检测电路，并进行仿真。设电路的初始状态为S0,接收到一个“1”后的状态为S1, 接收到 “10”后的状态为S2, 接收到 “101”后的状态为S3, 接收到 “1011”后的状态为S4。画出从S0到S4的状态转换图，并根据状态转换图，用verilog代码设计出该检测电路。对设计出来的检测电路进行仿真。仿真时，clk的时钟周期设定为10us，每个clk的上升沿输入din中的一位，假定输入的din信号序列是“1101101101000101101110111101000101010110101”。该电路的输入和输出引脚如下图1所示：

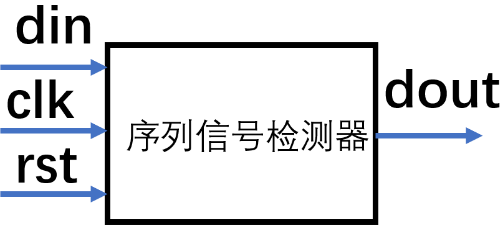


图1 序列信号检测器引脚图

根据上面的有限状态变化的规律，在vivado中用verilog语言设计与之对应的逻辑电路，并进行仿真测试，看看仿真波形是否与预期结果一致。

如果你的学号的后4位是X3X2X1X0(例如：你学号的后4位是3729,对应于X3=3,X2=7,X1=2,X0=9),则输入信号的激励值依次为：

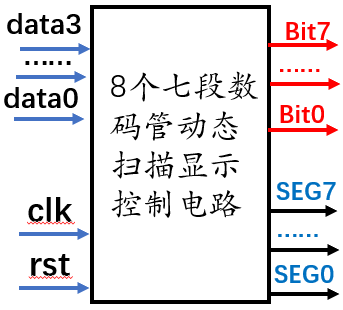
rst=0,延时X3个单位；然后使rst一直为1（rst=1）。

always #X2 clk=~clk;

每个clk时钟周期，输入din中的一位。din输入的值如下：

din：1101101101000101101110111101000101010110101

结束仿真，将仿真波形截图和vivado中的2个schematic截图存入实验报告文档中。

**二、设计一个对8个七段数码管进行动态扫描显示的控制电路，该电路的输入和输出引脚如下图2所示：**

在图中，clk是时钟输入端，rst是复位端。data3~data0是第一个要显示的数值。

输出信号Bit7~Bit0对应于8个七段数码管的位码控制端。输出信号SEG7~SEG0对应于8个七段数码管的段码控制端。

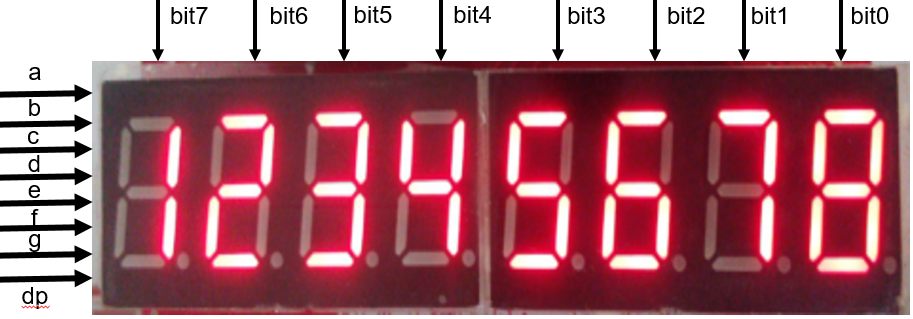
 图2 动态扫描控制电路引脚图

图3 8个七段数码管显示效果图

在上图的两个图中，seg7~seg0的输出正好对应于8个七段数码管的段码输入a~g和dp（代表小数字）。

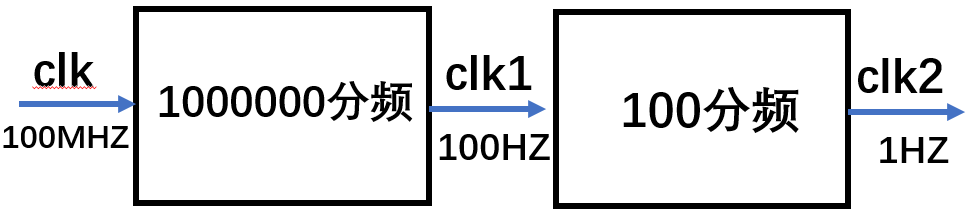
系统需要从clk输入信号通过分频的方式生成周期为10ms和周期为1s的两种不同的方波信号。该模块如下图4所示：

图4 多个时钟信号之间的联系

上图4中clk1是周期为10ms的方波，用clk1作为动态控制依次显示8个七段数码管的控制信号。也就是每隔10ms就给一个七段数码管送一个要显示的数值。依次给8个七段显示器送数，使8个七段数码管依次点亮，由于人的视觉暂停效应，看起来好像8个七段数码管是同时点亮的。 clk2是一个周期为1s的方波，每隔1s就使8个七段数码管上显示的数值加1（超过10的对10取模），第1个七段数码管加到9后就开始减1，减到data3~data0对应的值，完成了一次循环显示。依此不停的循环往复显示，直到按下了rst复位（rst=0停止显示，所有数码管全灭）。根据上面的有限状态变化的规律，在vivado中用verilog语言设计与之对应的逻辑电路，并进行仿真测试，看看仿真波形是否与预期结果一致。

如果你的学号的后4位是X3X2X1X0(例如：你学号的后4位是3729,对应于X3=3,X2=7,X1=2,X0=9),则输入信号的激励值依次为：

rst=0,延时X3个单位；然后使rst为1（rst=1）。延时10000000000个时间单位后结束仿真。仿真输入的数值和输出如表1所示：

表1 仿真时的激励值

|  |  |  |  |
| --- | --- | --- | --- |
| 输入data3~data0 | 输出Bit7~Bit0 | 输出SEG7~SEG0 | 选中的七段数码管显示的字型 |
| 0001 | 10000000 | 01100000 | 1 |
| 0010 | 01000000 | 11011010 | 2 |
| 0011 | 00100000 | 11110010 | 3 |
| 0100 | 00010000 | 01100110 | 4 |
| 0101 | 00001000 | 10110110 | 5 |
| 0110 | 00000100 | 10111110 | 6 |
| 0111 | 00000010 | 11100000 | 7 |
| 1000 | 00000001 | 11111110 | 8 |

结束仿真，将仿真波形截图和vivado中的2个schematic截图存入实验报告文档中。

**三、设计一个控制电路，在VGA显示器上显示彩色条纹，显示效果如下图5所示。根据显示器显示效果的变化规律，在vivado中用verilog语言设计与之对应的逻辑电路，并进行仿真测试，看看仿真波形是否与预期结果一致。**



图5 VGA显示器显示效果图

1. **VGA接口的结构**

通过VGA接口，可以很方便的连接VGA显示器。VGA接口的结构如下图6所示。

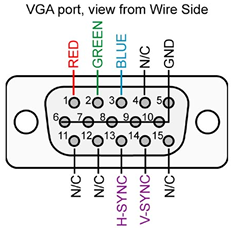






图6 VGA接口结构

VGA接口原理如图7所示，接口信号可以分为三大类

VGA\_HSYNC(1bit)：行同步信号，FPGA输出，默认为高电平，每—行显示之前，都要将其拉低一定的周期，用于同步。

VGA\_VSYNC(1bit)：帧同步信号，FPGA输出，默认为高电平，每—帧图像显示之前，都要将其拉低一定的周期，用于同步。

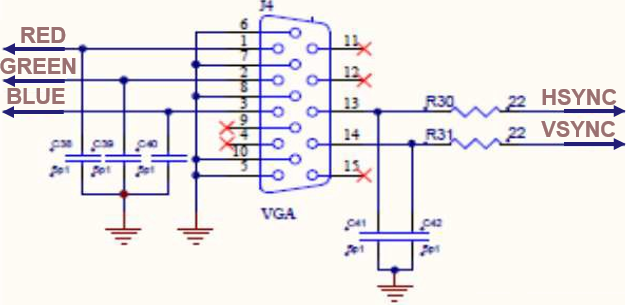
显示器依据从RED、GREEN和BLUE这3条线接收到的模拟电压在VGA显示器的某点显示一定亮度的颜色。

图7 VGA接口原理图

1. **VGA显示器的像素点扫描显示过程**

VGA驱动的原理大概如下：给引脚HSYNC一定的行同步信号，同时再给VSYNC一定的场同步信号（也称帧同步信号），那么VGA信号就从第一行第一个像素（屏幕最左上角）开始扫描，先是行方向上的像素点从左往右，一行结束后，从下一行的左端开始扫描，直到最后一行最后一个像素（屏幕最右下角）。如果扫描到屏幕上某个像素的时候，RED、GREEN、BLUE引脚上恰好是某个模拟电压，那么屏幕上的这个像素就显示与这3个模拟电压对应的颜色。

根据行扫描、场扫描时序要求，VGA显示时序设计的关键在于规定行（场）同步消隐，行（场）显示时间。对于800\*525分辨率显示器，VGA图像显示扫描如图8所示。

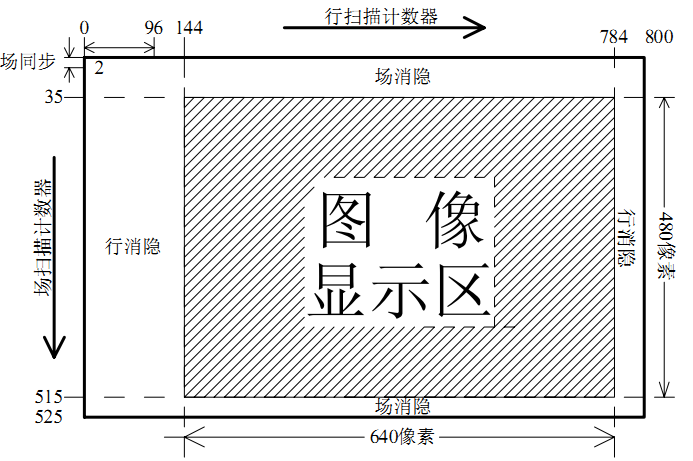


图8 VGA图像显示扫描示意图

行同步时序如图9所示。



图8 行同步时序示意图

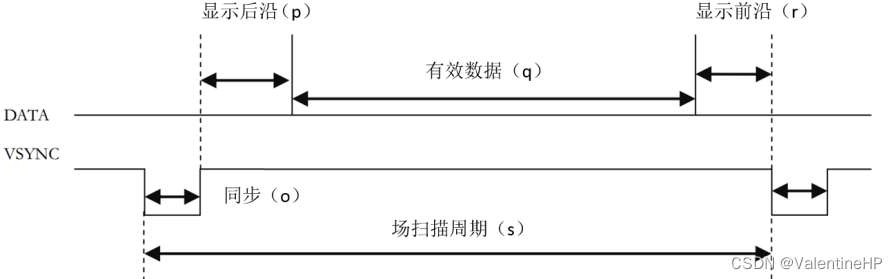
帧同步时序如图9所示。

图9 帧同步时序示意图

图8和图9中，都标注出了五个关键点，图中的参数（a）~（s）如表2和表3所示，根据这个表格中的数据，就可以编写VGA控制程序了。

表2 VGA显示器常见刷新率时序表

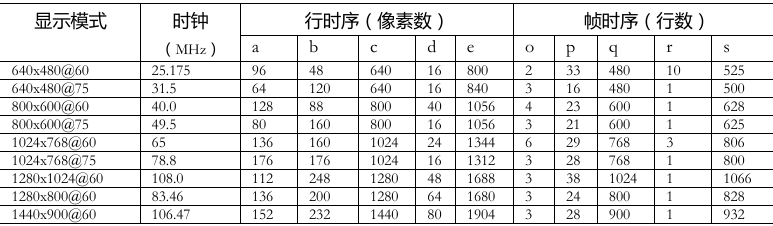
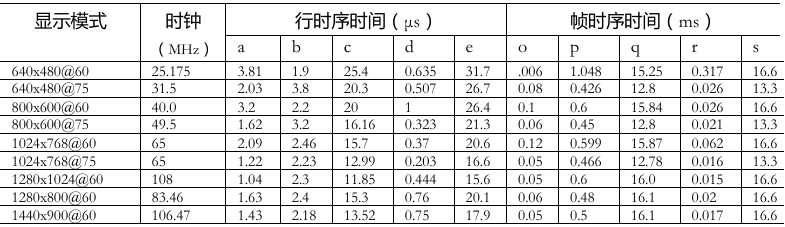


表3 VGA显示器常见刷新率时序表（时间）



每个部分持续的周期数，都在表2中给出来了，根据选择的显示模式，确定行和帧的周期数。这里我选择的是第—种640\*480@60的模式。



图10 行同步和帧同步时序示意图

由图10可看出，有效地显示一行信号需要96十48十640+16=800个像素点的时间，其中行同步负脉冲宽度为96个像素点时间，行消隐后沿需要48个像素点时间，每行显示640个像素点，行消隐前沿需要16个像素点的时间，一行显示时间为640个像素点时间，一行消隐时间为160个像素点时间。有效显示一帧图像需要2十33十480十10=525行时间，其中场同步负脉冲宽度为2个行显示时间，场消隐后沿需要33个行显示时间，每场显示480行，场消隐前沿需要10个行显示时间，一帧显示时间为480行显示时间，一帧消隐时间为45行显示时间。

**（三）EGO1开发板上与颜色有关的数据线以及对应的颜色编码**

在EGO1开发板上，每种颜色对应了4条信号线，4条信号线经过电阻网络对应的数模转换后，得到对应的模拟信号，如图11所示。在EGO1开发板中，每种颜色的位宽为4位，3种颜色共12位，对应了不同颜色的输出。单种颜色信号用4位二进制数来表示，这意味着该颜色可以有24 = 16种不同的亮度级别。对应RGB颜色模型的一小部分颜色编码如表4所示。

表4 RGB模型颜色编码

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 颜色 | 黑 | 蓝 | 绿 | 青 | 红 | 品 | 黄 | 白 |
| 蓝色（B） | 0000 | 1111 | 0000 | 1111 | 0000 | 1111 | 0000 | 1111 |
| 绿色（G） | 0000 | 0000 | 1111 | 1111 | 0000 | 0000 | 1111 | 1111 |
| 红色（R） | 0000 | 0000 | 0000 | 0000 | 1111 | 1111 | 1111 | 1111 |

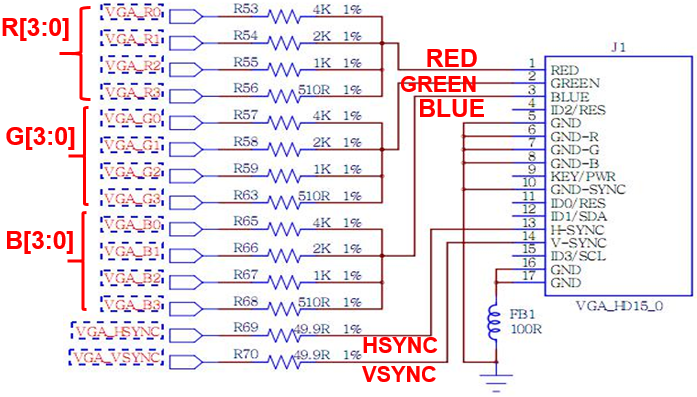
上表中，RED[3:0] = 4'b0000; 对应最低的红色亮度（可能接近黑色）。RED[3:0] = 4'b1111; 对应最高的红色亮度（最亮的红色）。3种颜色RED[3:0]、GREEN[3:0]和BLUE[3:0]混合在一起就形成了最终显示在显示器上的颜色和亮度。

图11 与颜色有关数据线示意图

**（四）用FPGA芯片设计VGA控制电路**

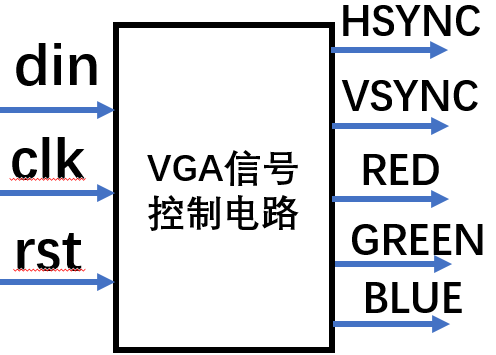
在图像显示区显示彩色条纹，要设计的是一个将显示数据进行动态传送的控制电路，控制VGA显示器进行动态显示。该电路的输入和输出引脚如图12所示：

图12 VGA控制电路引脚示意图

该控制电路可以看成是由3个模块构成的，分别是：时钟信号生成模块、同步信号生成模块和像素点颜色数据生成模块。其中的

时钟信号生成模块是根据显示器的显示频率和显示器的显示像素点，生成对应的信号时钟。例如：显示器刷新频率是60HZ,屏幕的像素点为：800\*525，则意味着：每秒要刷新60帧，每帧要显示800\*525个像素点。计算出每秒要送的像素点为：60\*800\*525=25.175\*106像素点/秒=25.18MHZ。由于EGO1开发板上的P17号引脚提供的是100MHZ的时钟，需要进行4分频才能得到接近25MHZ的时钟信号，这个分频后的时钟信号可以作为VGA显示器的刷新控制时钟。类似的，如果显示器刷新频率是60HZ,屏幕的像素点为：1344\*806，则意味着：每秒要刷新60帧，每帧要显示1344\*806个像素点。计算出每秒要送的像素点为：60\*1344\*806=65\*106像素点/秒=65MHZ。

同步信号生成模块是用来生成同步信号的模块，包含生成行同步信号HSYNC和帧同步信号VSYNC的控制电路。行同步信号HSYNC的特点是每一行显示开始时，需要给出维持若干时间（对应若干个像素点）的低电平，作为行显示同步标志。实现行同步控制时，需要设计一个行计数器，记录一行中已经显示的像素点数量。帧同步信号VSYNC的特点是每一帧显示开始时，需要给出维持若干时间（对应若干个像素点）的低电平，作为帧显示同步标志。实现帧同步控制时，需要设计一个帧计数器，记录该帧中已经显示的行的数量。如果一行有800个像素点，则行同步信号HSYNC每显示800个行像素后就要给出维持一定时间的低电平信号（与显示96个像素点时间相对应），表明新一行显示开始。如果一帧有800\*525个像素点，则帧同步信号VSYNC每显示800\*525个像素点（可以看作显示525行）后就要给出维持一定时间的低电平信号（与显示2行像素点时间相对应），表明新一帧显示开始。该模块还要生成有效的像素区域信号HSACTIVE和VSACTIVE。当像素点位于有效区域时，即HSACTIVE和VSACTIVE都为1时，才将对应的像素点的颜色数据经过电阻网络转换后送给R、G、B这3条信号线，否则将像素点颜色数据0000经过电阻网络转换后送给R、G、B这3条信号线，使这3条信号线上的模拟电压为0（显示黑色）。

像素点颜色数据生成模块就是将每个像素点对应的R[3:0]、G[3:0]、B[3:0]这3个颜色数据值送给电阻网络构成的数模转换模块,转换成RED、GREEN、BLUE这3个方向的模拟数据送给VGA显示器自带的控制器。VGA 接口属于模拟接口，所以需要把数字信号转换为模拟信号。实际硬件当中，在 FPGA 或者其他数字电路和 VGA 接口之间，会有一个数模转换器（由电阻网络构成的DAC）。这个DAC会把与某种颜色信号对应的多位数值，转换为对应的模拟电压，只需要一根线来传输该模拟色信号，以此来控制该颜色的亮度。由于EGO1开发板上的红、绿、蓝色信号线是4位数据线的结构，4位红色信号线R[3:0]经过电阻网络对应的数模转换后，得到对应的模拟信号，送到VGA接口引脚RED线上。4位绿色信号线G[3:0]经过电阻网络对应的数模转换后，得到对应的模拟信号，送到VGA接口引脚GREEN线上。4位蓝色信号线B[3:0]经过电阻网络对应的数模转换后，得到对应的模拟信号，送到VGA接口引脚BLUE线上。

要在图像显示区显示彩色条纹，由于512<525<800<1024=210，考虑设置用来保存坐标位置的两个寄存器X[9:0]和Y[9:0],X用来保存，Y用来保存垂直方向的坐标。图像显示区对应的X和Y坐标起点为：10’d144和10’d35。X和Y坐标终点为：10’d784和10’d515。这个显示区域内要显示彩色条纹，彩色条纹的颜色主要与X坐标有关，具体的关系如下表5所示。

表5 彩色条纹对应的坐标与颜色编码

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 水平坐标 | 水平坐标 | 红色值 | 绿色值 | 蓝色值 | 最后显示的颜色 | 备注 |
| X[9:0] | X[9:0] | R[3:0] | G[3:0] | B[3:0] |
| 10’d144 | 10’d223 | 1111 | 0000 | 0000 | 红 | 每种颜色对应了80个像素点。 |
| 10’d224 | 10’d303 | 0000 | 1111 | 0000 | 绿 |
| 10’d304 | 10’d383 | 0000 | 0000 | 1111 | 蓝 |
| 10’d384 | 10’d463 | 1011 | 1111 | 1111 | 蓝白 |
| 10’d464 | 10’d543 | 0000 | 0000 | 0000 | 黑 |
| 10’d544 | 10’d623 | 1111 | 1111 | 0111 | 黄 |
| 10’d624 | 10’d703 | 1111 | 0111 | 1111 | 紫 |
| 10’d704 | 10’d783 | 0000 | 1111 | 1111 | 蓝绿 |
| 其他 | | 0000 | 0000 | 0000 | 黑 |  |

设置时钟频率为100MHZ,仿真10秒后，结束仿真。将仿真波形截图和vivado中的2个schematic截图存入实验报告文档中。