实验九 VGA 接口控制器实现

实验报告

181860085 汤昊

数字电路与数字系统二班

邮箱：1174639585@qq.com

2019.11.1

一．实验目的

学习 vga 接口的工作原理，学习 vga 接口控制器的设计方法。

二．实验原理

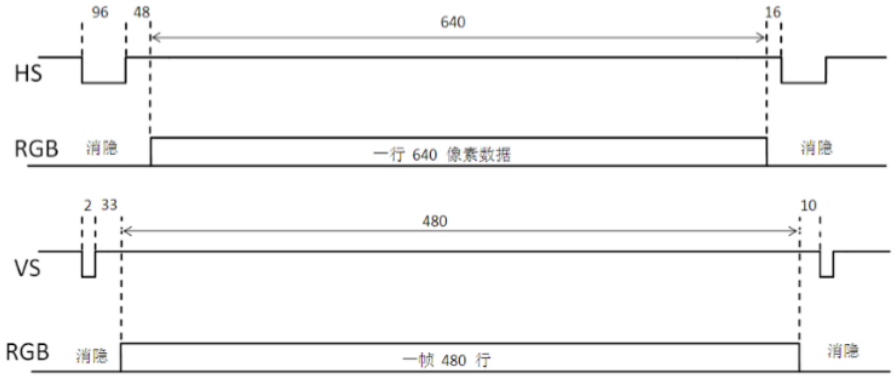
（1）vga 图像显示

图像的显示是以像素（点）为单位，显示器的分辨率是指屏幕每行有多少 个像素及每帧有多少行，标准的 VGA 分辨率是 640×480

VGA 显示器一般的刷新频率是 60HZ。 每一帧图像的显示都是从屏幕的左上角开始一行一行进行的，行同步信号 是一个负脉冲，行同步信号有效后，由 RGB 端送出当前行显示的各像素点的 RGB 电压值，当一帧显示结束后，由帧同步信号送出一个负脉冲，重新开始从屏的左上端开始显示下一帧图像。

RGB 端并不是所有时间都在传送像素信息，CRT 的电子束从上一行的行尾到下一行的行头需要时间，从屏幕的右下角回到左上角开始下一帧也需 要时间，这时 RGB 送的电压值为 0（黑色），这些时间称为电子束的行消隐时 间和场消隐时间，行消隐时间以像素为单位，帧消隐时间以行为单位。

（2）vga 扫描时序



有效显示一行信号需要 96+48+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，然后每行显示640 个像素点，最后行消 隐前沿需要 16 个像素点的时间。所以一行中显示像素的时间为 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+33+480+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，然后每场显示480行，场消隐前沿需要 10 个行显示时间，一帧显示时间 为 525行显示时间，一帧消隐时间为 45行显示时间。

因此，在 640×480 的 VGA 上的一幅图像需要 525×800 =420k 个像素点的 时间。而每秒扫描 60 帧共需要约 25M 个像素点的时间。

（3）vga 接口信号

开发板和 ADV7123 芯片之间的接口引脚包括 3 组 8bit 的颜色信号 VGA_R[7:0], VGA_G[7:0], VGA_B[7:0]，行同步信号 VGA_HS，帧同步信号 VGA_VS，VGA 时钟信号 VGA_CLK，VGA 同步（低有效）VGA_SYNC_N，和 VGA 消隐信号（低有效）VGA_BLANK_N

三．实验环境及器材

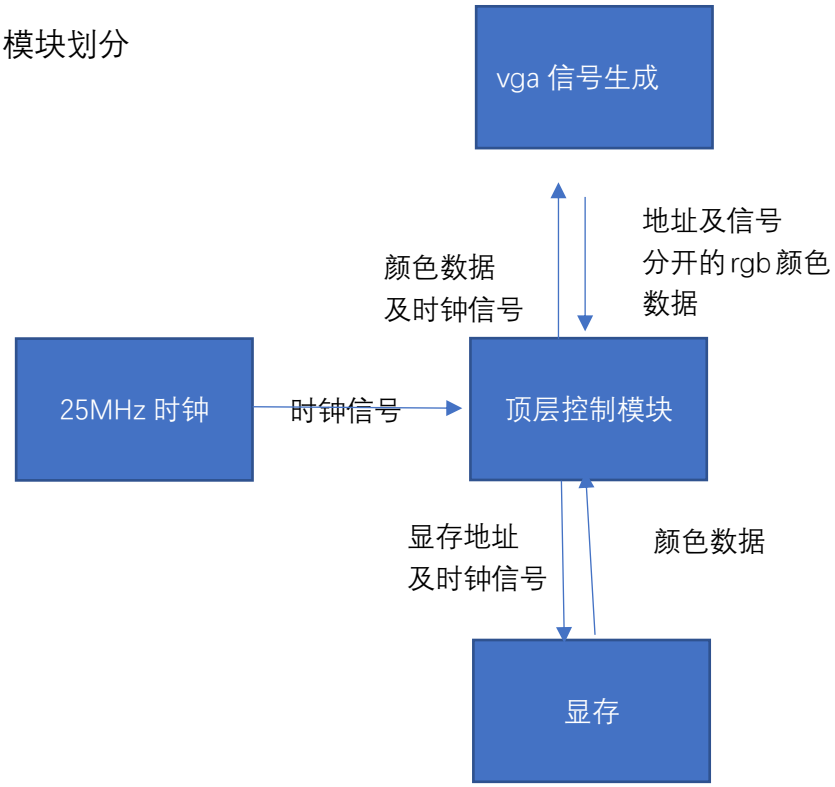
开发软件：quartus prime 17.1

开发器材：DE-standard 开发板

vga 接口的显示器

四．接口控制器

（1）模块划分



（2）25MHz 时钟——clkgen 模块

一个可以根据调用时参数生成不同频率时钟的生成器。

原理：使用了 parameter 量 clk_freq，而生成器产生一次输出的计数次数与此参数有关，countlimit=50000000/2/clk_freq。

在本实验中需要 25MHz 的时钟，故传参 25000000。

（3）vga 信号产生——vga_ctrl 模块

根据扫描时序产生以下 vga 信号：
x_cnt，y_cnt：根据时钟信号计数
h_sync，v_sync：x_cnt>96 产生行同步信号，y_cnt>2 产生列同步信号
h_addr，v_addr：行和列地址，在 x_cnt>144 和 y_cnt>35 时才是有效像素坐标
h_valid，v_valid：x_cnt 超过 784 产生行消隐信号，y_cnt 超过 515 产生列消隐信号
valid：行列都消隐时消隐
vga_r/g/b：RGB 颜色数据

（4）显存——memory 模块

实验采用低比特颜色显示方案，将 v_addr 和 h_addr 合成显存地址，在时钟沿获得输出（这样 quartus 可以使用 BLOCKRAM(M10K)综合）。

addr[8:0]=v_addr[8:0];
addr[18:9]=h_addr;

可以这样赋值的原因是显存的空间和照片 mif 的大小都是 640*512，这样 addr=h_addr*512+v_addr，故低九位直接对应 v_addr 低 九位，高十位直接对应 h_addr。

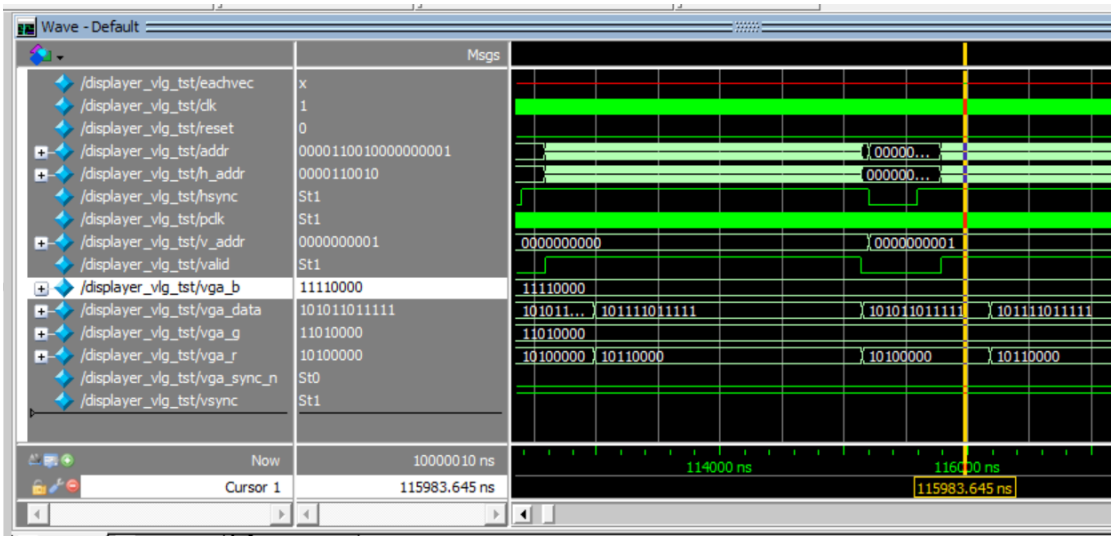
（5）顶层控制模块——displayer 模块

- 实例化时钟，将 25MHz 时钟信号送给 vga_ctrl 和 memory，在不需要显存如显示条纹时，需要在时钟上升沿对颜色数据处理。
- 实例化 vga_ctrl，将各种信号引出，根据行列信号合成显存地址
- 实例化 memory，得到显存地址的数据送给 vga_ctrl 分成 RGB 数据输出

五．显示不同颜色条纹

（1）设计思路：在时钟上升沿对行地址进行判断，若小于某值则为一种颜色，不小于为另一种颜色。设置的值为 100

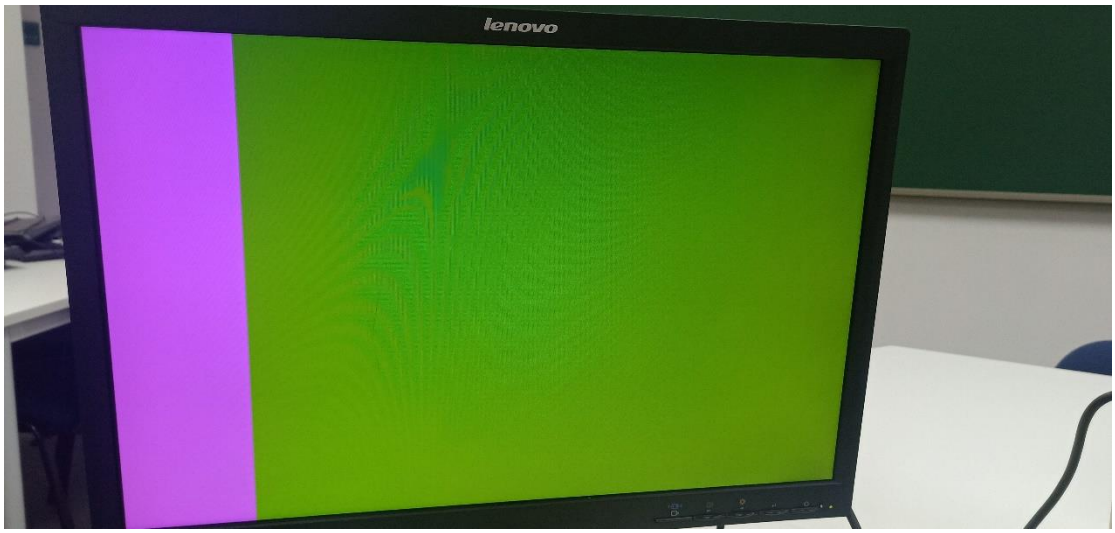
（2）仿真模拟结果



设置 h_addr<100 时，数据为 12'hADF，h_addr>=100 时数据为 12'hBDF，仿真结果中可以看到在光标所放的位置附近有一次 101011011111 变成 101111011111

（3）实际效果

由于 12'hADF，12'hBDF 对应颜色较淡，最终换成了对比较强的紫色和绿色，数据分别为 12'hB3F 和 12'h480，效果如下图所示。



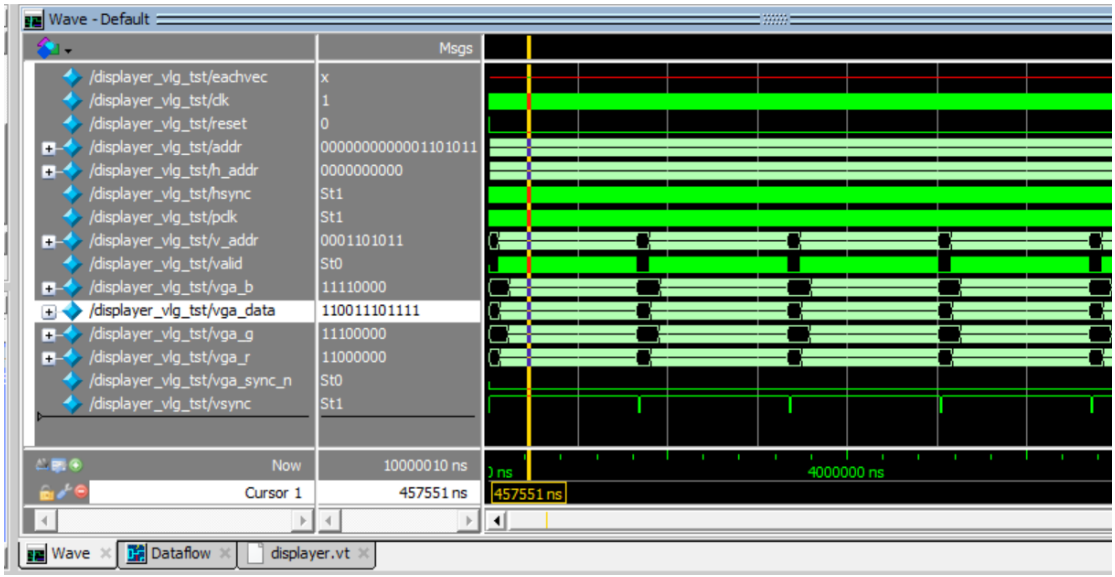
六．显示静态图片

1.显示给出图片

（1）设计思路

原有基础上在顶层模块中引入显存，用给出的 mif 文件初始化。显存 rom 使用 IP 核创建。

（2）仿真模拟



可以看到随着 v_addr，h_addr 变化，addr，vga_data，vga_r/g/b 都在不停变化

（3）显示结果

图片格式为 640*480

2.显示自定义图片

（1）设计思路

原有基础上使用 matlab 脚本生成自定义图片的 mif 文件，初始化显存。

初始化方式为(* ram_init_file = "my_picture.mif" *) reg [11:0] mem[327679:0];

但使用此语句则无法使用仿真模拟测试。

（2）显示结果

使用的图片格式为 640*400

七．动态显示图片

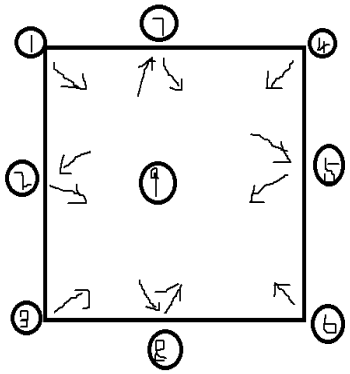
1.确定图片显示区域

代码中使用 logo_x 和 logo_y 确定图片的起始位置（左上角），再根据 logo 的高度和宽度确定当前的行列地址是否在图片区域内，在则 rom_addr 加一，取出对应颜色显示，不在则 rom_addr 保持原来的位置，颜色显示为黑色

2.实现图片的移动和反弹

flag_add_sub 代表移动方式，logo_x 和 logo_y 会根据这个标志进行相应的加减坐标，如当标志为 00 时，在时钟沿两个坐标都要加一，标志位 11 时反之。

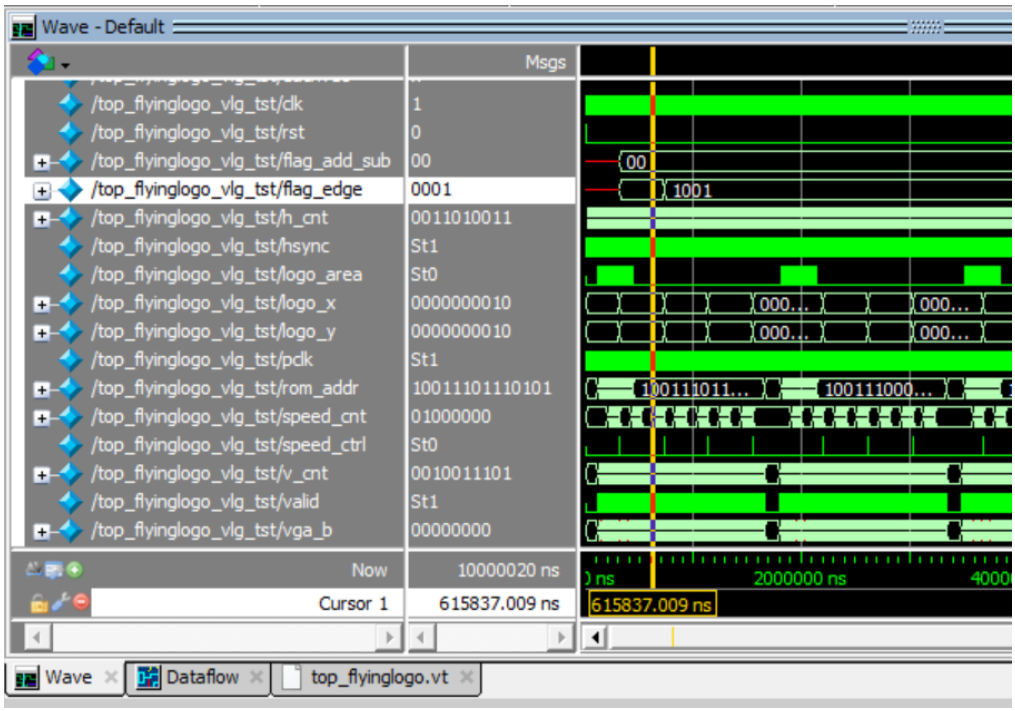
00	右下
01	右上
10	左下
11	左上



根据 logo_x 和 logo_y 目前所在的位置分成了如图所示 9 种情况，flag_edge 记录了是那种情况

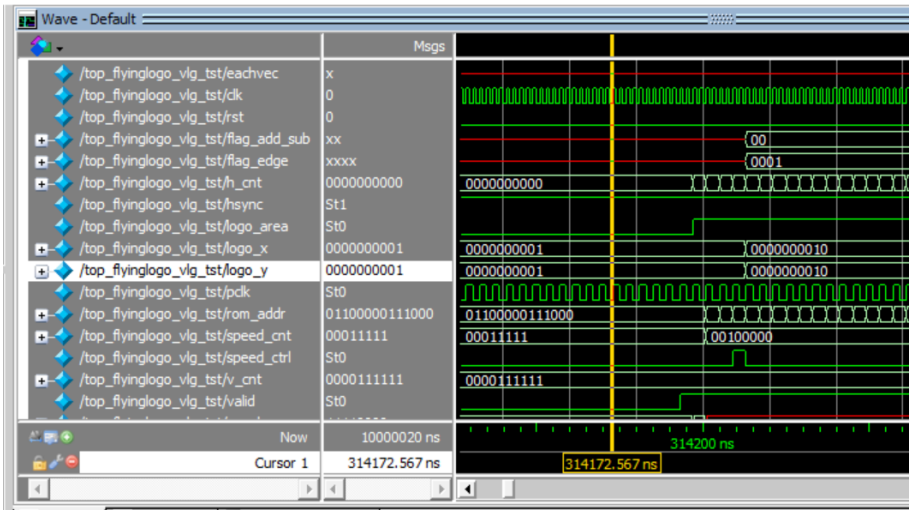
（1） 图片在左上角，应该右下移动
（2） 图片在左边缘，应该向相反方向移动，实现反弹效果
（3） 图片在左下角，应该向右上移动
（4） 图片在右上角，应该向左下移动
（5） 图片在右边缘，应该向相反方向移动，实现反弹效果
（6） 图片在右下角，应该向左上角移动
（7） 图片在上边缘，应该向相反方向移动，实现反弹效果
（8） 图片在下边缘，应该向相反方向移动，实现反弹效果
（9） 图片未触及边缘，移动方向不变

可以看到下图仿真模拟中 logo_x,y 从 1 到 2 时（右下移动），flag_edge 的状态从 0001 变为 1001，从第一个状态到第九个状态



3.速度方向的控制信号

speed_cnt 初始为 0，在 v_cnt[5]==0h_cnt==1 时加一，即当行数在 32-64 时（不包括 64）且列数为 1 时计数，而 speed_ctrl 有效的条件是 speed_cnt[5]先 0，再连续两个为 1 才会输出 speed_ctrl 为 1，即在 speed_cnt[5]变为 32 时有效一次，故行数为 63 时 speed_ctrl 有效（如下图仿真模拟所示），判断当前的状态，得出移动的方向



八．实验反思和收获

- 使用初始化方式为(* ram_init_file = “*.mif" *) 时仿真无法执行，故体现不出颜色数据的变化
- 在动态显示模块中显存地址的寻址方式为直接得到显存地址，即在上升沿及 logo_area 有效时加一，即行优先，但这样与生成 mif 文件数据的格式冲突（代码中颜色数据的排列是列优先），实际测试中显示的图像是原来的转置，在修改生成 mif 文件的顺序后正常。