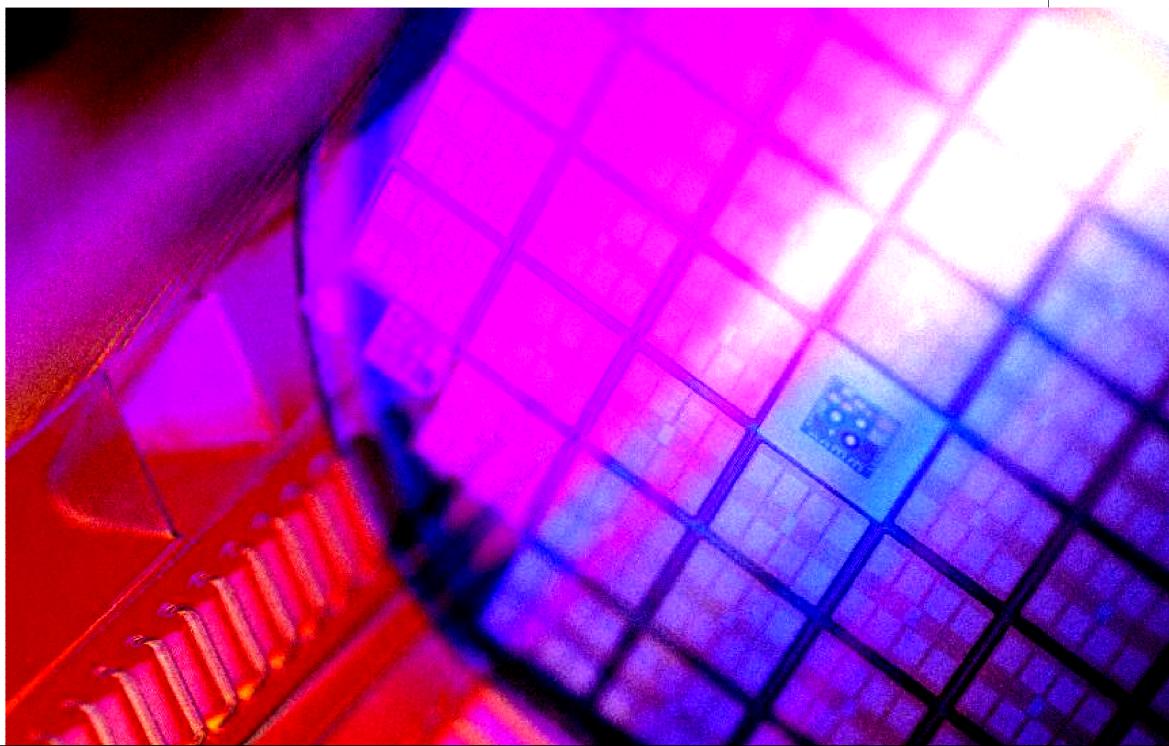


FEQ1301T05-V0.00



ZRtech

FPGA 开发套件 HDL 实验教程

—VGA 实验之一



www.zr-tech.com

实验 5 VGA 实验之一

1. 概述

相较 LED 和数码管，VGA 接口逻辑较为复杂和系统，因此我们分期分批推出教程，内容涵盖了：VGA 接口说明、VGA 显示驱动详解、九宫格色块、渐变色、BMP 图像显示等，想较软件实现图像处理的能力而言，FPGA 处理图形仅次于 GPU，在灵活性方面甚至还有胜出；作为简易的图像接口设计应该是绰绰有余了。

2. 准备工作

- 板上供给时钟频率为 48Mhz，图像数据 16 位宽，其中用于红色显示 5 位，绿色显示 6 位，蓝色 5 位；RGB 数据权位及排列方式如下：

最高位																最低位
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	E3	B2	B1	B0	
红高位		→	红低位绿高位		→	绿低位	兰高		绿高位	兰低		兰低				

图 3

3. VGA 的传说

如同江湖中久远的传说一样，自 PC 诞生起就有了 VGA，VGA 全称 Video Graphic Array，也叫显示绘图阵列，是逐行扫描的显示制式，其支持的分辨率为 640X480，对更高分辨率 800X600，称为 SVGA (Super VGA) 模式；1024X768 称为 XGA (Extended Graphics Array，扩展图形阵列)；

- VGA 时序分行时序和帧（场）时序，两者都包含同步脉冲(Sync a)、显示后沿(Back porch b)、显示时段(Display interval c)和显示前沿(Front porch d)四个部分，图样如下：

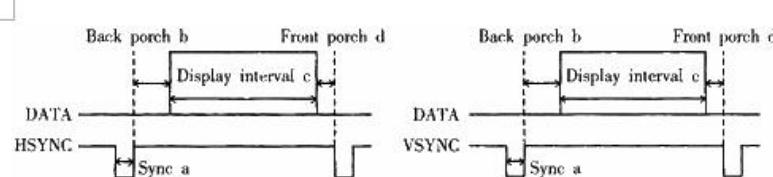


图 2 VGA 标准参考时序图

图 4

- 各种分辨率的行场同步时序如表 xxxx 所示，

VGA 参考时序数据表

图像模式	行时序(μs)				帧时序(lines)				时钟(MHz)
	a	b	c	d	a	b	c	d	
1024×768XGA(75Hz)	1.2	2.2	13.0	0.2	3	28	768	1	78.8
1024×768XGA(60Hz)	2.1	2.5	15.8	0.4	6	29	768	3	65
800×600SVGA(60Hz)	3.2	2.2	20.0	1.0	4	23	600	1	40
640×480VGA(75Hz)	2.0	3.8	20.3	0.5	3	16	480	1	31.5

图 5

还是老样子，更多的知识诸君请百度之。

4. 实战 VGA 之一九宫格色彩

4.1. 例程烧录

以九宫格形式显示 7 种色块，开始我们的第一个图形显示测试，本次采用倒叙的手法，先烧录例程中的二进制文件，再解析代码。

该实验通常用来测试图像显示功能是否能实现；该步骤的关键是看是否能产生规定的行场同步时序，即上表所作要求；

- 打开例程目录，开启例子工程；将其编译后下载到目标板，操作如下：

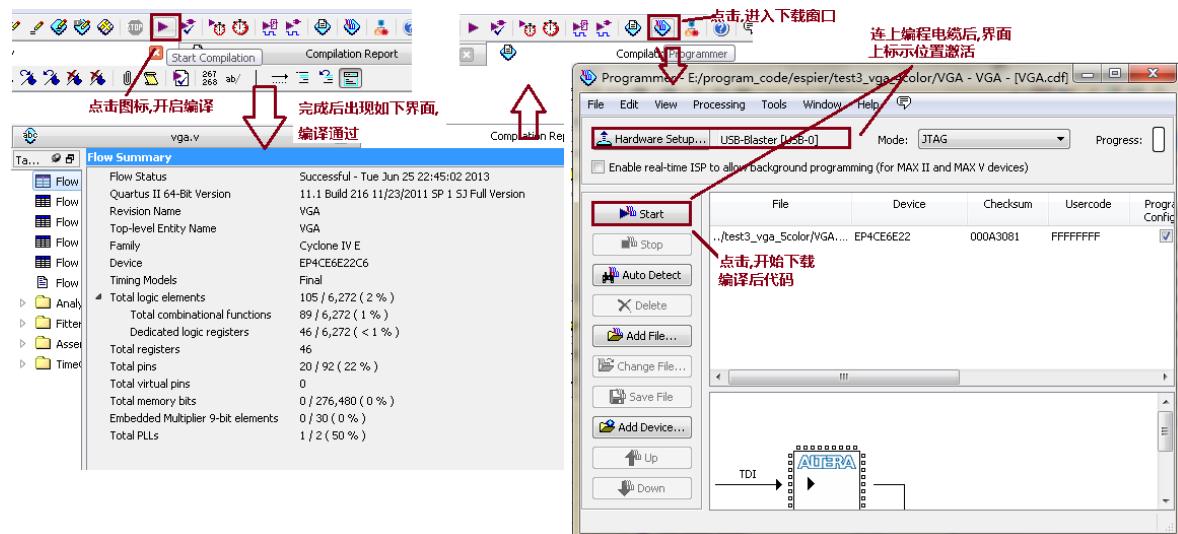


图 6

- 或将编译后生成的.sof 文件转换为.jic 文件，下载到 fpga 外挂的 flash 中；使之可以在再次开电时，自加载代码而不需要电缆下载代码运行；转换方法如下：

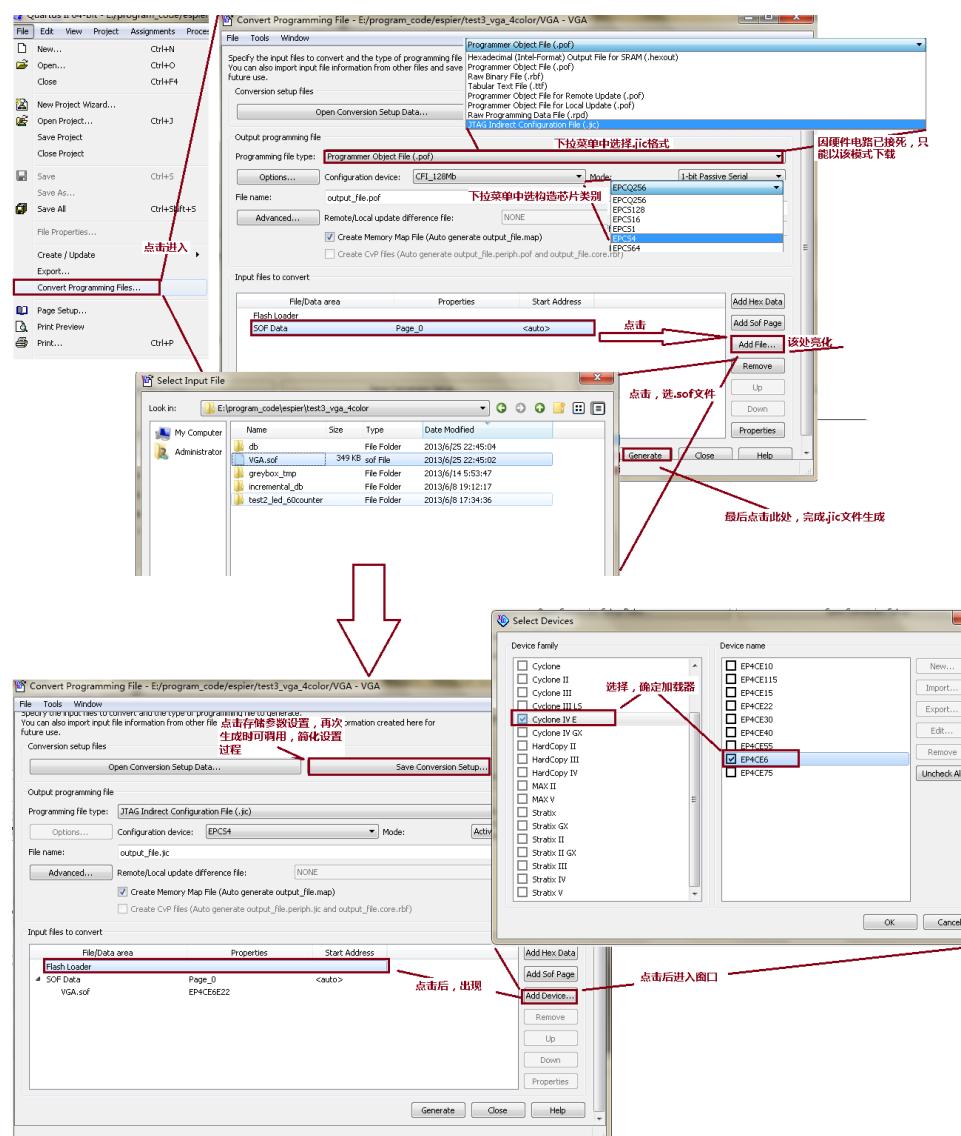


图 7

➤ 烧录.jic 文件方法如下（烧录后，重新开电，fpga 即自动加载 flash 中代码）：

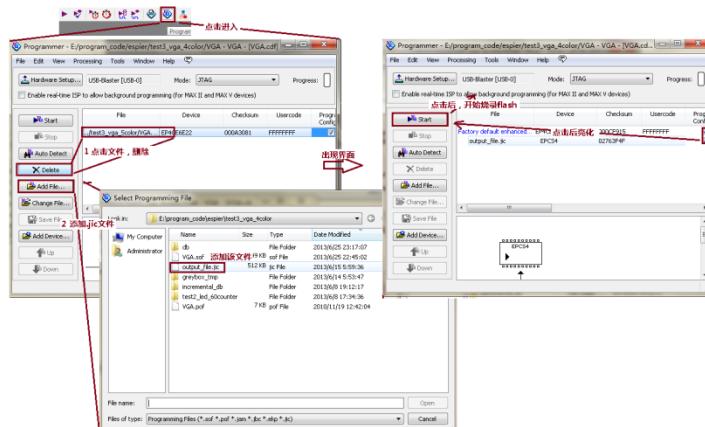


图 8

➤ 显示效果如图，箭头有点复杂啊，请老手跳过，新手留步；

4.2. 代码解析

敲字艰难啊，例程就直接图片了啊，下面分段解析 vga.v 文件代码含义；

◆ 定义输入输出端口

```
module VGA
  | input  clk,           输入时钟为48mhz
  | output hsync,         行同步信号
  | output reg vsync,    场同步信号
  | output reg [15:0] pixel, 16位像素
  | output clk40m        像素点频时钟
  // output [7:0] red
);
通过内部锁相环实现，  
具体需用宏模块(ip核)
```

图 9

◆ 定义几个固定参数

8种不同颜色所对应的像素值	行场同步参数
<pre>parameter RED = 'h001; parameter GREEN = 'h07e; parameter BLUE = 'hf100; parameter YELLOW = 'h07f; parameter CYAN = 'hffed; parameter INDIGO = 'hffff; parameter WHITE = 'hffff; parameter BLACK = 'h0000;</pre>	<pre>//Horizontal timing constants parameter H_PIXELS = 'd804;//20m=403,d*640, parameter H_FRONTPORCH = 'd40;//20m-19, parameter H_SYNCTIME = 'd128;//20m-64, parameter H_BACKPORCH = 'd88;//20m-42, parameter H_SYNCSTART = 'd900;//20m-421,H_PIXELS+H_FRONTPORCH, parameter H_SYNCEND = 'd1025;//20m-485,H_SYNCSTART+H_SYNCTIME, parameter H_PERIOD = 'd1056;//20m-528,H_SYNCEND+H_BACKPORCH,</pre>
场同步占有的时钟数	行同步占有的时钟数
<pre>V_LINES = 'd604;//48, V_FRONTPORCH = 'd1//d*10, V_SYNCTIME = 'd4//2,</pre>	<pre>H_PIXELS = 'd804;//20m=403,d*640, H_FRONTPORCH = 'd40;//20m-19, H_SYNCTIME = 'd128;//20m-64,</pre>
场同步占有的行数	场同步占有的行数
<pre>V_BACKPORCH = 'd23;//33, V_SYNCSTART = 'd603;//V_LINES+V_FRONTPORCH, V_SYNCEND = 'd607;//V_SYNCSTART+V_SYNCTIME,</pre>	<pre>V_BACKPORCH = 'd23;//33, V_SYNCSTART = 'd603;//V_LINES+V_FRONTPORCH, V_SYNCEND = 'd628;//V_SYNCEND+V_BACKPORCH;</pre>
场周期占有的行数	

图 10

◆ 调用宏模块（下面会对具体生成作说明）

```
clk_pll pll_inst
(
  .inclk0(clk),
  .c0(clk40m)
);
调用宏模块clk_pll，将其命名为pll_inst  
将clk信号传递给inclk0端口  
将c0端口信号传递给clk40m
```

图 11

◆ 同步信号产生

形成行同步	形成场同步
<pre>always @ (posedge clk40m) 以点频为计算时钟 if(hcnt<H_PERIOD) hcnt <= hcnt+1; else hcnt <= 0; //internal horizontal synchronization pulse generation (negative polarity) always @ (posedge clk40m) if(hcnt>=H_SYNCSTART && hcnt<H_SYNCEND) hsyncint <= 0; else hsyncint <= 1;</pre>	<pre>//vertical counter of lines always @ (posedge hsyncint) 以行为计数时钟 if(vcnt<V_PERIOD) vcnt <= vcnt+1; else vcnt <= 0; //vertical synchronization pulse generation (negative polarity) always @ (posedge hsyncint) if(vcnt>=V_SYNCSTART && vcnt<V_SYNCEND) vsync <= 0; else vsync <= 1;</pre>

图 12

◆ 确定哪个区域显示何种数据

```

    确定哪个区域显示那种数据
    if((vcnt>0) && (vcnt<180)) begin      0~180行
        if((hcnt>260) && (hcnt<368))  260~368列 范围内显红色
            PIX = RED;
        else if((hcnt>368) && (hcnt<380))
            PIX = BLACK;
        else if((hcnt>380) && (hcnt<488))
            PIX = GREEN;                   //2
        else if((hcnt>488) && (hcnt<500))
            PIX = BLACK;
        else if((hcnt>500) && (hcnt<608))
            PIX = BLUE;                  //3
        else
            PIX = BLACK;
    end

```

图 13

➤ 时钟宏模块的生成

因 800*600@60hz 画面生成所需的点频为 40MHz，而输入却是 48MHz，这须要借助 fpga 内部锁相环，由宏模块管理器完成，具体如下：

✧ 进入 MegaWizard Plug-In Manager 宏模块生成窗，如下：

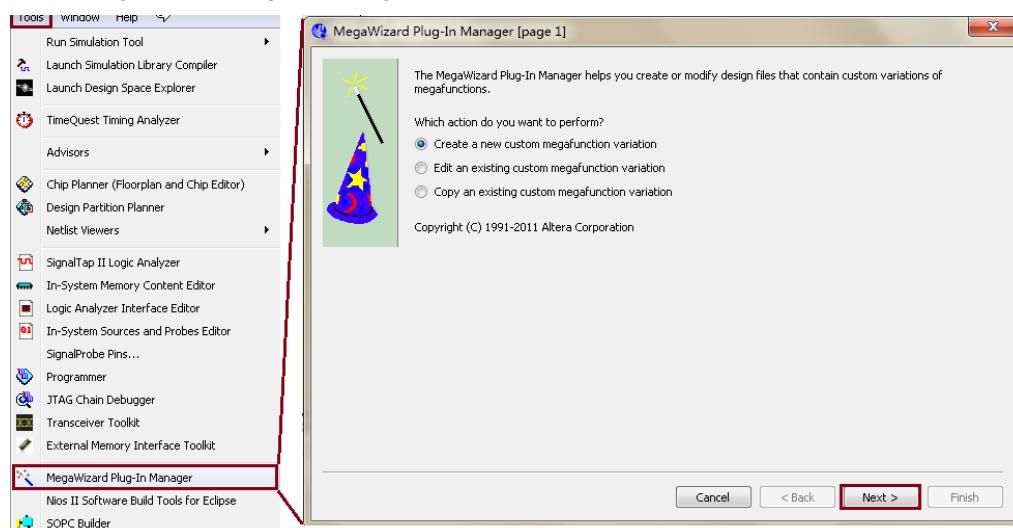


图 14

✧ 选择宏模块类别，并定义模块名称，

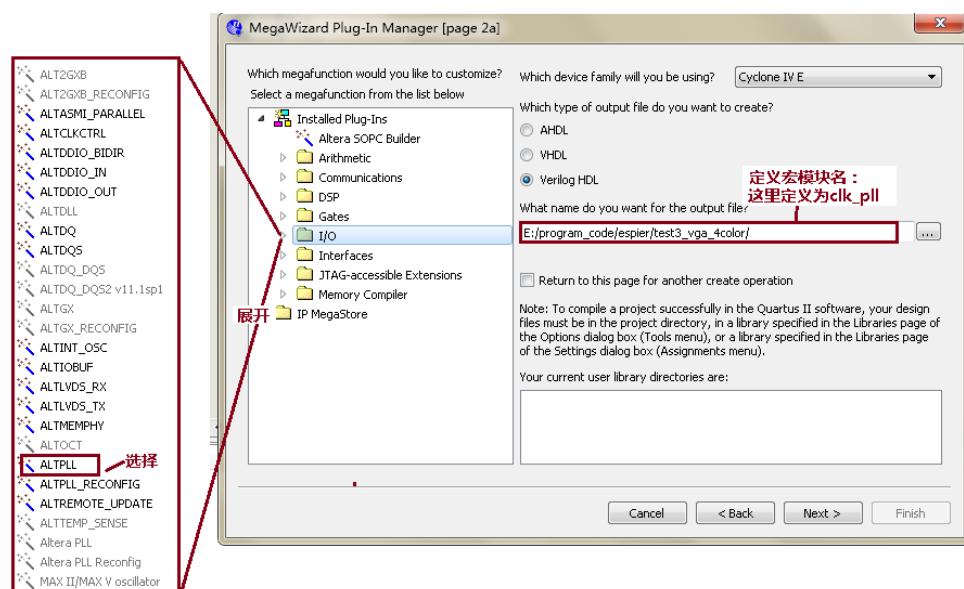


图 15

✧ 下一步，进入界面后设置输入时钟

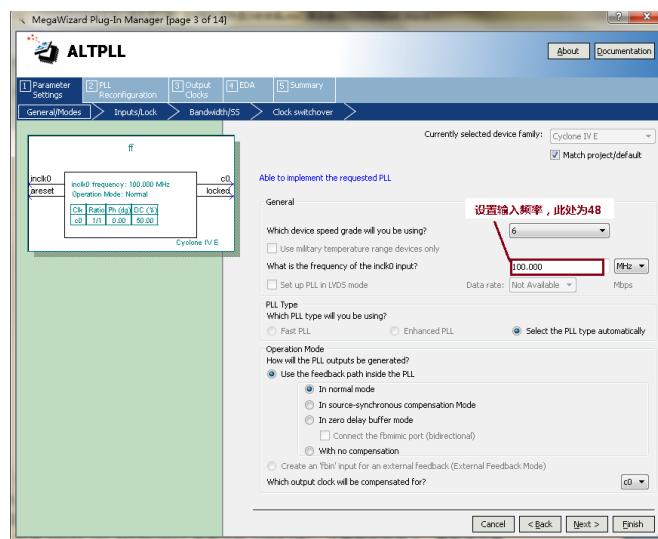


图 16

✧ 下一步，进入界面后清除如图的设置

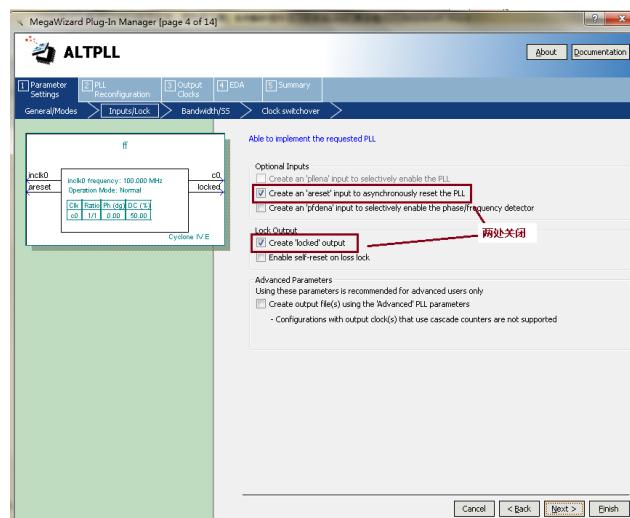


图 17

✧ 下一步，

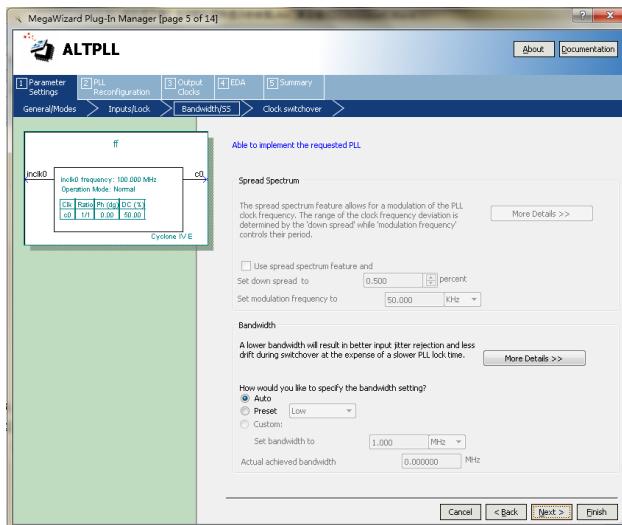


图 18

✧ 下一步，

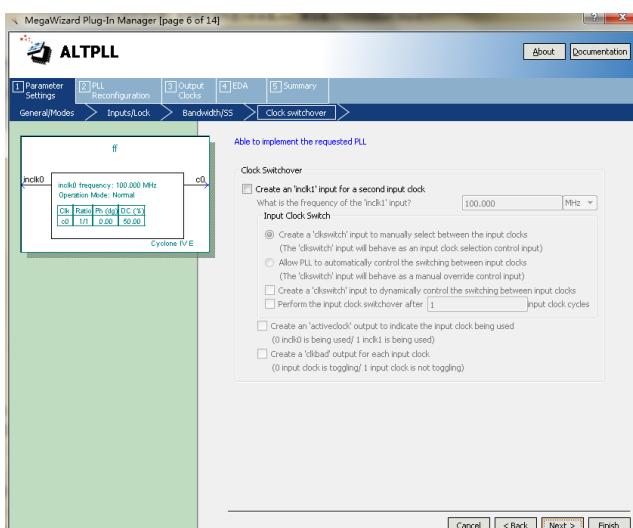


图 19

✧ 下一步，

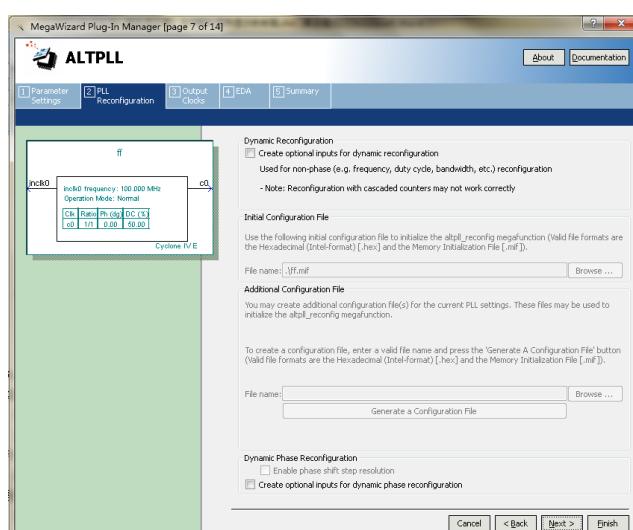


图 20

✧ 下一步，进入界面后设置输出时钟 40mhz

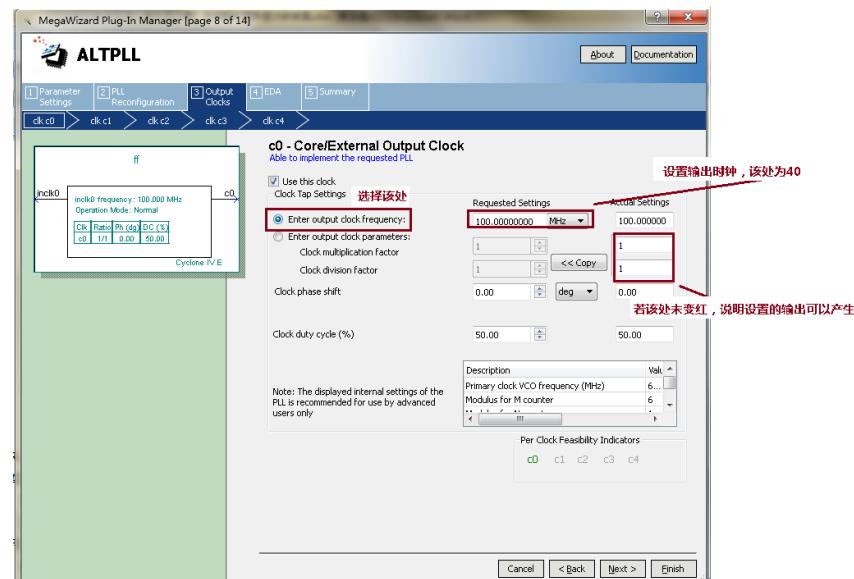


图 21

✧ 连续下一步，跳过 c1~c4 的时钟输出设置，如下

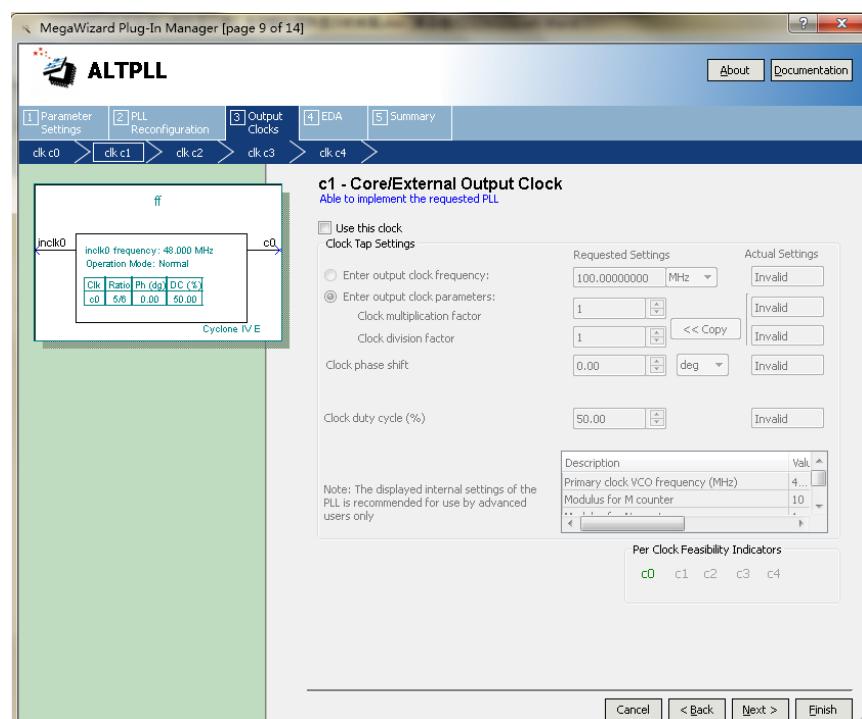


图 22

✧ 下一步，进入界面

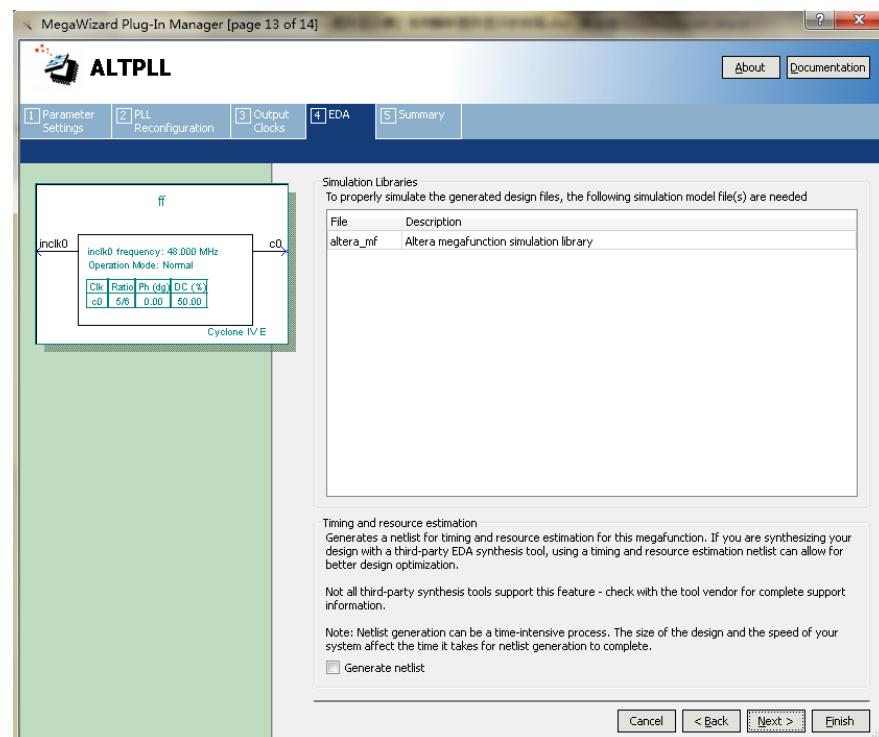


图 23

✧ 是不是下一步烦了，点击 finish 吧，完成宏模块创建；

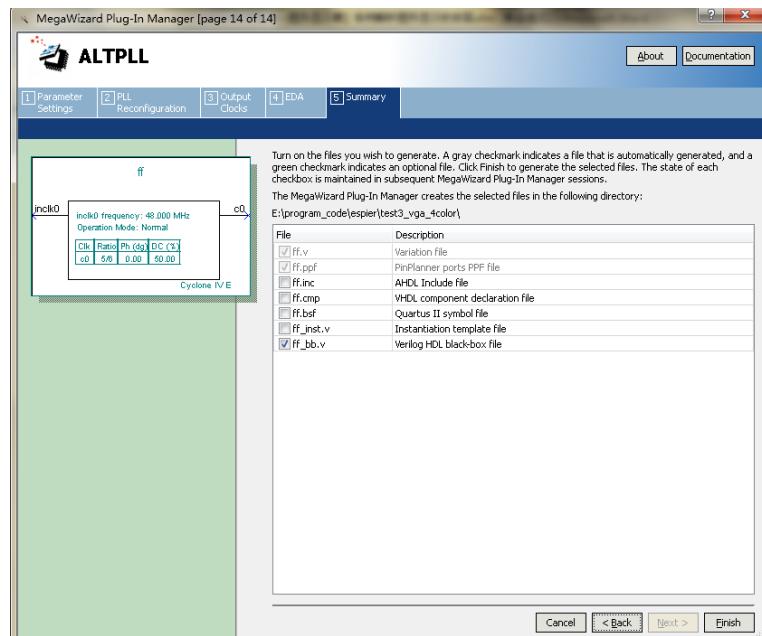


图 24

➤ 图像显示效果如下图，宽屏显示器就这效果，谁让咱习惯按 4:3 做呢，下面的实验中会有改变分辨率的实验，请期待。



图 25

5. 实战 VGA 之二渐变色

说实话，有了前面这么详细的讲解，我想，VGA 对各位已经不成问题了吧，下面做一个显示器测试常用的实验，对 R、G、B 进行颜色渐变测试（分单色渐变和混色渐变），在这里该实验的目的，是验证由电阻阵列实现视频 Da 变换的有效性；而通常这个实验室用了测试显示器的灰度指标。

打开相应例程，下载到目标板上，看到显示器产生效果如下；代码自己理解，不难的。

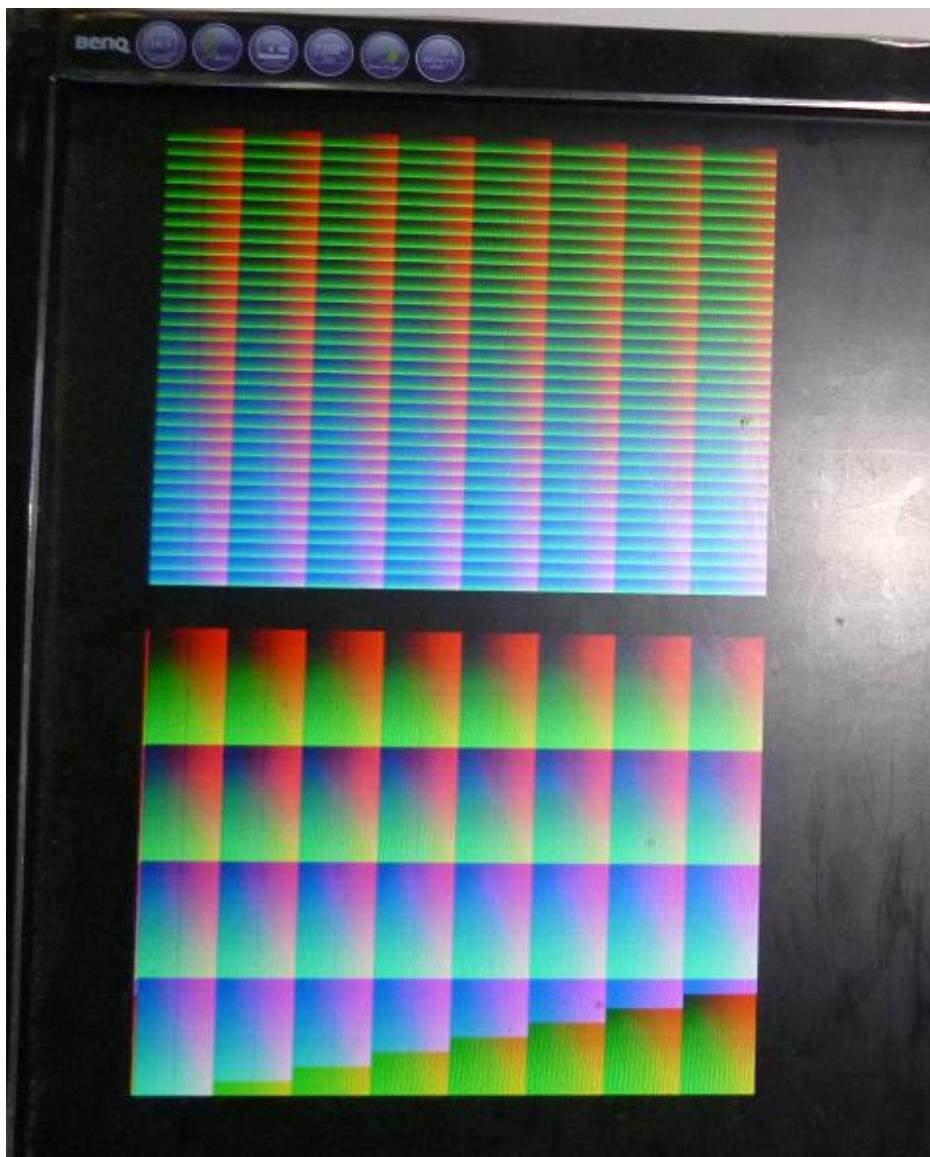


图 26

文档内部编号 : FEQ1301T05

编号说明 :

首字母 : F-FPGA 系列

首二字母 : L-理论类 E-实验类 T-专题类

首三字母 : C-普及类 Q-逻辑类 S-软核类

数字前两位 : 代表年度

数字后两位 : 同类文档顺序编号

尾字母/数字 : C 目录 , T 正文 , 数字表示章节号

修订记录

版本号	日期	描述	修改人
0.00	2013.9.25	FEQ1301T05 文档建立	kdy