至简设计系列_LCD 显示叠加图片

--作者：肖肖肖

1.1 总体设计

1.1.1 概述

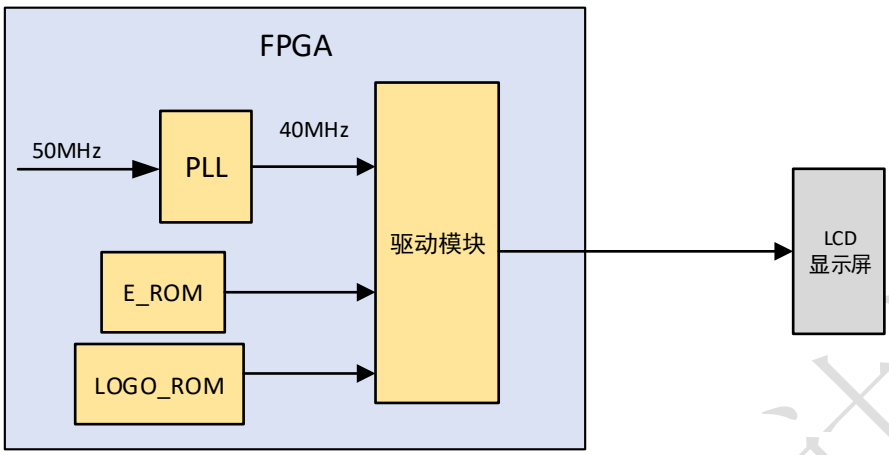
液晶显示器是一-种通过液晶和色彩过滤器过滤光源，在平面面板上产生图像的数字显示器。LCD 的构造是在两片平行的玻璃基板当中放置液晶盒，下基板玻璃上设置薄膜晶体管,.上基板玻璃上设置彩色滤光片，通过薄膜晶体管上的信号与电压改变来控制液晶分子的转动方向，从而达到控制每个像素点偏振光出射与否而达到显示目的。与传统的阴极射线管相比，LCD 具有占用空间小，低功耗，低辐射，无闪烁，降低视觉疲劳等优点。现在 LCD 已渐替代 CRT 成为主流，价格也已经下降了很多，并已充分的普及。

1.1.2 设计目标

在 7 寸 LCD 显示屏上实现图片显示。
其中，在显示屏左上角显示明德扬的 LOGO 图标，在显示屏的中间居中显示字母“E”。

1.1.3 系统结构框图

系统结构框图如下所示：



图一

1.1.4 模块功能

- **PLL 模块实现功能**

1. 将输入的 50MHz 时钟分频输出 40MHz 时钟。

- **ROM 模块实现功能**

1. fpga_rom 存储明德扬 LOGO 的图像数据；
2. e_rom 存储字母“E”的图像数据。

- **LCD 驱动模块实现功能**

1、 产生驱动 LCD 屏显示的时序
2、 读取 ROM 里存储的数据并输出显示

1.1.5 顶层信号

信号名	I/O	位宽	定义
clk	I	1	系统工作时钟 50M
rst_n	I	1	系统复位信号，低电平有效
hys	O	1	LCD 行时序信号

vys	O	1	LCD 场时序信号
lcd_de	O	1	LCD 数据输入使能信号
lcd_rgb	O	24	LCD RGB 信号，RGB 格式为使用 24 位来表示一个像素，RGB 分量都用 8 位表示，取值范围为 0-255。
lcd_dclk	O	1	LCD 数据采样时钟

1.1.6 参考代码

```
1.  module top_mdyLcdPic(
2.      clk       ,
3.      rst_n     ,
4.      hys       ,
5.      vys       ,
6.      lcd_de    ,
7.      lcd_rgb   ,
8.      lcd_dclk
9.      );
10.
11.     parameter   PICTURE_W = 24  ;
12.
13.     input                 clk        ;
14.     input                 rst_n      ;
15.     output                hys        ;
16.     output                vys        ;
17.     output                lcd_de     ;
18.     output  [PICTURE_W-1:0] lcd_rgb    ;
19.     output                lcd_dclk   ;
20.
21.
22.     wire                   clk_0     ;
23.
24.     wire                   hys       ;
25.     wire                   vys       ;
26.     wire                   lcd_de    ;
27.     wire   [PICTURE_W-1:0]   lcd_rgb    ;
```

```
28.     wire                    lcd_dclk   ;
29.
30.
31. //40MHz
32. pll_40m u_pll_40m(
33.        .areset     (~rst_n ),
34.         .inclk0     (clk    ),
35.        .c0         (clk_0  )
36.     );
37.
38.
39. lcd_driver  u2(
40.    .clk          (clk_0       ),//40MHz
41.    .rst_n        (rst_n       ),
42.
43.    .hys          (hys         ),
44.    .vys          (vys         ),
45.    .lcd_de       (lcd_de      ),
46.    .lcd_rgb      (lcd_rgb     ),
47.    .lcd_dclk     (lcd_dclk    )
48.     );
49.
50. endmodule
51.
```

1.2 PLL 模块设计

1.2.1 接口信号

下面为使用矩阵键盘时的接口信号：

信号名	I/O	位宽	定义
areset	I	1	PLL 复位信号，高电平有效
inclk0	I	1	PLL 输入时钟 50MHz
c0	O	1	PLL 输出时钟 40MHz

1.2.2 设计思路

本模块主要用于产生 LCD 驱动时序所需要的时钟，关于 PLL 的使用详细介绍请看下方链接：
http://www.fpgabbs.cn/forum.php?mod=viewthread&tid=322&fromuid=100105

1.3 ROM 模块设计

1.3.1 接口信号

信号名	I/O	位宽	定义
address	I	16	ROM 数据存放地址
clock	I	1	ROM 工作时钟 40MHz
q	O	8	ROM 输出数据

1.3.2 设计思路

本模块主要用于存储需要显示的图像数据，关于 ROM 的使用详细介绍请查看 IP 核右上角数据手册“Documentation”。

1.4 LCD 驱动模块设计

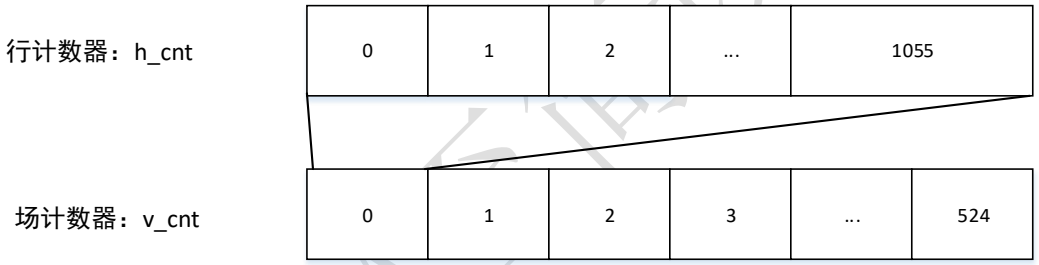
1.4.1 接口信号

信号名	I/O	位宽	定义
clk	I	1	模块工作时钟 40MHz
rst_n	I	1	系统复位信号，低电平有效
hys	O	1	LCD 行时序信号

vys	O	1	LCD 场时序信号
lcd_de	O	1	LCD 数据输入使能信号
lcd_rgb	O	24	LCD RGB 信号，RGB 格式为使用 24 位来表示一个像素，RGB 分量都用 8 位表示，取值范围为 0-255。
lcd_dclk	O	1	LCD 数据采样时钟

1.4.2 设计思路

产生驱动 LCD 显示的行场时序信号，其计数器架构如下图所示：



行计数器 h_cnt：该计数器用来计算行同步信号的帧长。加一条件为 1，表示一直在计数。结束条件为数 1056 个，也就是一行有 1056 个像素。

场计数器 v_cnt：该计数器用来计算场同步信号的帧长。加一条件为 end_h_cnt，即行计数器的计数器的结束条件，表示每计数完一行像素就加一。结束条件为数 525 个，也就是一共有 525 行像素。

其中，在从存储图像“E”的 ROM 里读取数据的时候，有一个操作就是读取的地址从第 8 位开始，也就是说 18 位数据地址，低三位不读，读取的是 `e_rom_addr[16:3]`。有这样一个操作的话就能实现对存储的图像进行 8 倍的放大显示。

1.4.3 参考代码

```
1.  module lcd_driver(
2.      clk          ,//40MHz
3.      rst_n        ,
4.
5.      hys          ,
6.      vys          ,
7.      lcd_de       ,
```

```
8.      lcd_rgb      ,
9.      lcd_dclk
10. );
11.
12.    input                   clk            ;
13.    input                   rst_n          ;
14.
15.    output                  hys            ;
16.    output                  vys            ;
17.    output                  lcd_de         ;
18.    output [23:0]           lcd_rgb        ;
19.    output                  lcd_dclk       ;
20.
21.    reg                     hys            ;
22.    reg                     vys            ;
23.    reg                     lcd_de         ;
24.    reg    [23:0]           lcd_rgb        ;
25.    wire                    lcd_dclk       ;
26.
27.    //1056
28.    parameter         THPW      = 20        ;
29.    parameter         THB       = 46        ;
30.    parameter         THD       = 800       ;
31.    parameter         THFP      = 210       ;
32.
33.    //525
34.    parameter         TVPW      = 10        ;
35.    parameter         TVB       = 23        ;
36.    parameter         TVD       = 480       ;
37.    parameter         TVFP      = 22        ;
38.
39.    parameter       HDE_CENTRE  = THD/2      ;//400
40.    parameter       VDE_CENTRE  = TVD/2      ;//240
41.
42.    parameter       LOGO_X0     = (0   + (THB-1))        ;
43.    parameter       LOGO_X1     = (120 + (THB-1))        ;
44.    parameter       LOGO_Y0     = (0   + (TVB-1))        ;
45.    parameter       LOGO_Y1     = (55  + (TVB-1))        ;
46.
47.    parameter         E_X0      = ((HDE_CENTRE-200) + (THB-1))       ;
48.    parameter         E_X1      = ((HDE_CENTRE+200) + (THB-1))       ;
49.    parameter         E_Y0      = ((VDE_CENTRE-150) + (TVB-1))       ;
50.    parameter         E_Y1      = ((VDE_CENTRE+150) + (TVB-1))       ;
```

```
51.
52.    reg   [ 10:0]           h_cnt          ;
53.    wire                   add_h_cnt      ;
54.    wire                   end_h_cnt      ;
55.    reg   [ 9:0]           v_cnt          ;
56.    wire                   add_v_cnt      ;
57.    wire                   end_v_cnt      ;
58.
59.
60.    wire                   active_area    ;
61.    reg                    logo_rom_area       ;
62.    reg      [15:0]         logo_rom_addr       ;
63.    wire     [7:0]          logo_rom_data       ;
64.    reg                    e_rom_area       ;
65.    reg      [17:0]         e_rom_addr       ;
66.    wire     [7:0]          e_rom_data       ;
67.    reg      [2:0]          e_rom_addr_low    ;
68.    reg                    e_sel            ;
69.
70.
71.
72. always @(posedge clk or negedge rst_n) begin
73.     if (rst_n==0) begin
74.         h_cnt <= 0;
75.     end
76.     else if(add_h_cnt) begin
77.         if(end_h_cnt)
78.             h_cnt <= 0;
79.         else
80.             h_cnt <= h_cnt+1 ;
81.    end
82. end
83. assign add_h_cnt = 1;
84. assign end_h_cnt = add_h_cnt  && h_cnt == (THB + THD + THFP)-1 ;
85.
86.
87. always @(posedge clk or negedge rst_n) begin
88.     if (rst_n==0) begin
89.         v_cnt <= 0;
90.     end
91.     else if(add_v_cnt) begin
92.         if(end_v_cnt)
93.             v_cnt <= 0;
```

```
94.         else
95.             v_cnt <= v_cnt+1 ;
96.    end
97. end
98. assign add_v_cnt = end_h_cnt;
99. assign end_v_cnt = add_v_cnt  && v_cnt == (TVB + TVD + TVFP)-1 ;
100.
101.  /*****************************************************/
102.      //dclk
103.      assign lcd_dclk = clk;
104.
105.      //hsync
106.      always  @(posedge clk or negedge rst_n)begin
107.          if(rst_n==1'b0)begin
108.              hys <= 0;
109.          end
110.          else if(add_h_cnt && h_cnt==THPW-1)begin
111.              hys <= 1;
112.          end
113.          else if(end_h_cnt)begin
114.              hys <= 0;
115.          end
116.      end
117.
118.
119.      //vsync
120.      always  @(posedge clk or negedge rst_n)begin
121.          if(rst_n==1'b0)begin
122.              vys <= 0;
123.          end
124.          else if(add_v_cnt && v_cnt==TVPW-1)begin
125.              vys <= 1;
126.          end
127.          else if(end_v_cnt)begin
128.              vys <= 0;
129.          end
130.      end
131.
132.
133.      //lcd_de
134.      always  @(posedge clk or negedge rst_n)begin
135.          if(rst_n==1'b0)begin
136.              lcd_de <= 0;
```

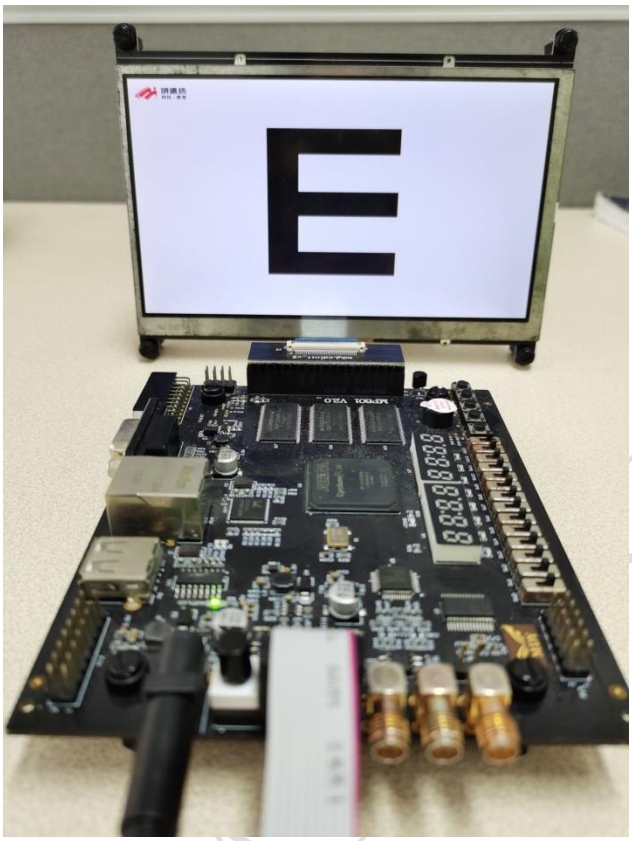
```
137.          end
138.          else if(active_area)begin
139.              lcd_de <= 1;
140.          end
141.          else begin
142.              lcd_de <= 0;
143.          end
144.      end
145.
146.  /*****************************************************************/
147.
148.  assign active_area = h_cnt>=(THB-1) && h_cnt<(THB+THD-1) && v_cnt>=(TVB-1) &&
    v_cnt<(TVB+TVD-1);
149.
150.
151.  always  @(*)begin
152.      logo_rom_area = h_cnt >=LOGO_X0 && h_cnt < LOGO_X1 && v_cnt >= LOGO_Y0 &&
    v_cnt < LOGO_Y1;
153.  end
154.
155.  always  @(*)begin
156.      e_rom_area = h_cnt >=E_X0 && h_cnt < E_X1 && v_cnt >= E_Y0 && v_cnt < E_Y1;
157.  end
158.
159.
160.  always  @(posedge clk or negedge rst_n)begin
161.      if(rst_n==1'b0)begin
162.          lcd_rgb <= 0;
163.      end
164.      else if(active_area)begin
165.          if(logo_rom_area)
166.              lcd_rgb <=
    {logo_rom_data[7:5],5'b11111,logo_rom_data[4:2],5'b11111,logo_rom_data[1:0],
    6'b111111};
167.          else if(e_rom_area)
168.              lcd_rgb <= {24{e_sel}};
169.          else
170.              lcd_rgb <= {24{1'b1}};
171.      end
172.      else begin
173.          lcd_rgb <=0;
174.      end
175.  end
```

```
176.
177.
178.  always  @(*)begin
179.      logo_rom_addr = (h_cnt-LOGO_X0) + 120*(v_cnt-LOGO_Y0);
180.  end
181.
182.  always  @(*)begin
183.      e_rom_addr = (h_cnt-E_X0) + 400*(v_cnt-E_Y0);
184.  end
185.
186.
187.
188.  always  @(posedge clk or negedge rst_n)begin
189.      if(rst_n==1'b0)begin
190.          e_rom_addr_low <= 0;
191.      end
192.      else begin
193.          e_rom_addr_low <= e_rom_addr[2:0];
194.      end
195.  end
196.
197.  always  @(*)begin
198.      e_sel = ~e_rom_data[7-e_rom_addr_low];
199.  end
200.
201.
202.  fpga_rom u_fpga_rom(
203.                .address (logo_rom_addr),
204.                .clock   (clk     ),
205.                .q       (logo_rom_data));
206.
207.  e_rom u_e_rom(
208.      .address   (e_rom_addr[16:3]  ),
209.      .clock     (clk  ),
210.      .q         (e_rom_data  ));
211.
212.
213.      endmodule
```

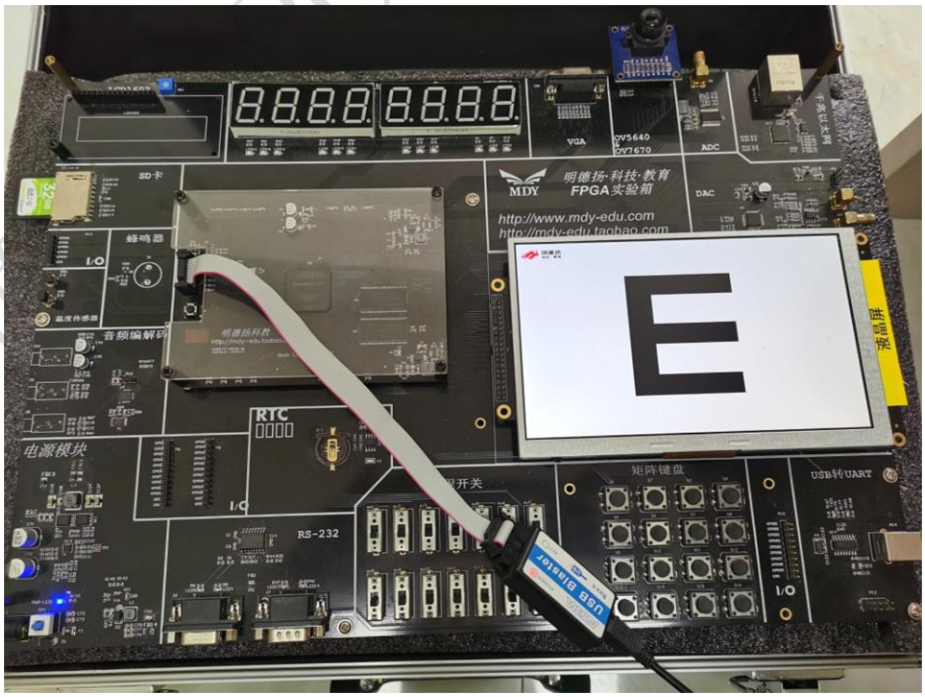
1.5 效果和总结

以下为工程上板后的现象效果图：

- **mp801 开发板**



- **ms980 试验箱**



也可以看一下我们往期的文章：

《至简设计系列_LCD 入门案例_边框显示》
《至简设计系列_BCD 译码实现》
《至简设计系列_简易计算器》
《至简设计系列_基于 FPGA 的超声波测距系统设计》
《至简设计系列_串口回环工程》
《至简设计系列_矩阵按键检测》
《至简设计系列_闹钟》
《至简设计系列_7 段数码管显示》
《阻塞赋值与非阻塞赋值》
《参数例化时自动计算位宽的解决办法》

1.6 公司简介

明德扬是一家专注于 FPGA 领域的专业性公司，公司主要业务包括开发板、教育培训、项目承接、人才服务等多个方向。

点拨开发板——学习 FPGA 的入门之选。

MP801 开发板——千兆网、ADDA、大容量 SDRAM 等，学习和项目需求一步到位。

网络培训班——不管时间和空间，明德扬随时在你身边，助你快速学习 FPGA。

周末培训班——明天的你会感激现在的努力进取，升职加薪明德扬来助你。

就业培训班——七大企业级项目实训，获得丰富的项目经验，高薪就业。

专题课程——高手修炼课：提升设计能力；实用调试技巧课：提升定位和解决问题能力；FIFO 架构设计课：助你快速成为架构设计师；时序约束、数字信号处理、PCIE、综合项目实践课等你来选。

项目承接——承接企业 FPGA 研发项目。

人才服务——提供人才推荐、人才代培、人才派遣等服务。