安路 SparkRoad 开发板例程说明

FPGA ： Field Programmable Gate Array　　现场可编程门阵列

TD ： Tang Dynasty　　安路 FPGA 开发环境

1 拨码开关例程

1.1 硬件说明

本例使用拨码开关进行 LED 的控制，共 16 路拨码开关，控制 16 个 LED 灯的亮灭。

拨码开关的电路原理图如图 1 所示：

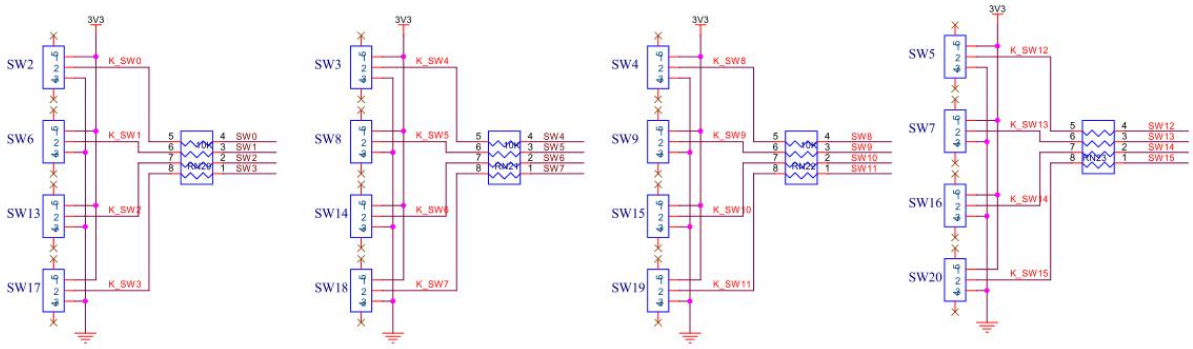


图 1

LED 灯的电路原图如图 2 所示:

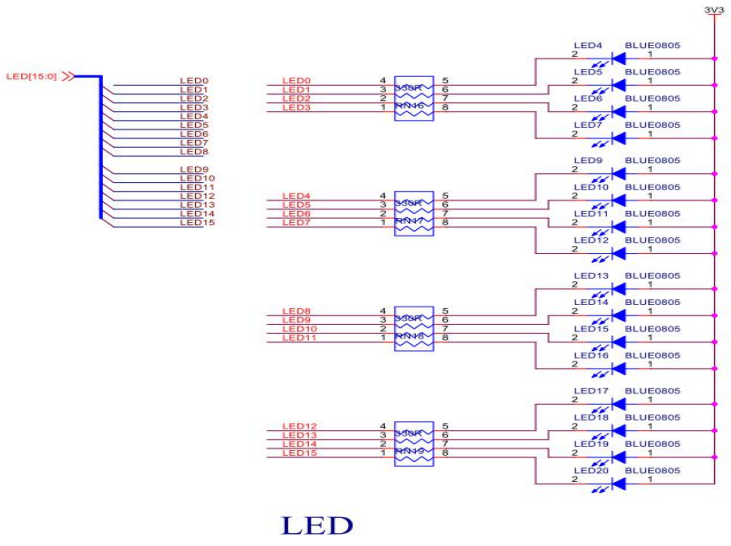


图 2

LED 灯采用共阳极的连接方式，在工程中只需拉低引脚的电平即可点亮 LED，相反可以熄灭 LED 灯。

1.2 功能与现象

拨动拨码开关即可点亮或熄灭对应的 LED 灯。

1.3 所用资源

本例程在 TD 软件资源使用情况如图 3 所示。

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	32		
#input	16		
#output	16		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	0	19600	0.00%
#reg	0	19600	0.00%
#le	0		
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	32	188	17.02%
#ireg	0		
#oreg	0		
#treg	0		
#pll	0	4	0.00%

图 3

2 按键消抖例程

本例程使用按键来控制 LED 灯的亮灭，主要实现按键消抖的功能。

2.1 硬件说明

按键消抖原理分析：

按键消抖可分为硬件消抖和软件消抖。硬件消抖的原理是在信号输入系统之前消除抖动干扰，在按键较少的情况下比较适宜。如果按键较多，则使用软件消抖。软件消抖的实质在于降低键盘输入端口的采样频率，将高频抖动略去。需要注意的是，软件消抖需要占据一定的系统资源。 尽管硬件消抖和软件消抖能实现按键消抖功能，串行处理的方式都存在一定的局限性，显得不那么完美。而硬件资源丰富的 FPGA 系统采用并行处理的模式，利用硬件来减轻软件工作量，通过硬件加速软件消抖处理，即可做到软件消抖并行化，因而在按键消抖处理方面具备非常明显的优势。一般按键所用开关为机械弹性开关，由于机械触点的弹性作用，每个按键开关在闭合时不会马上稳定地接通，在断开时也不会一下子断开。因而在闭合及断开的瞬间均伴随有一连串的抖动，如图 4 所示。抖动时间的长短由按键的机械特性决定，一般为 5 ms～10 ms。

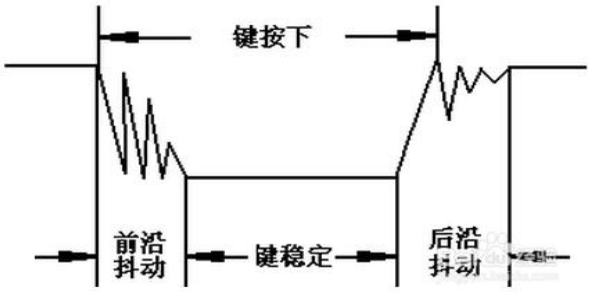


图 4

当系统检测出按键闭合后，执行一个延时程序，产生 5ms～10ms 的延时；前沿抖动消失后，再一次检测键的状态；如果仍保持闭合状态电平，则确认为真正有键按下。当检测到按键释放后，也要给 5ms～10ms 的延时，待后沿抖动消失后才能转入该键的处理程序。本案例我们设置经过 20 ms 后的高电平才是真正的按键功能。

按键的电路原理图如图 5 所示：

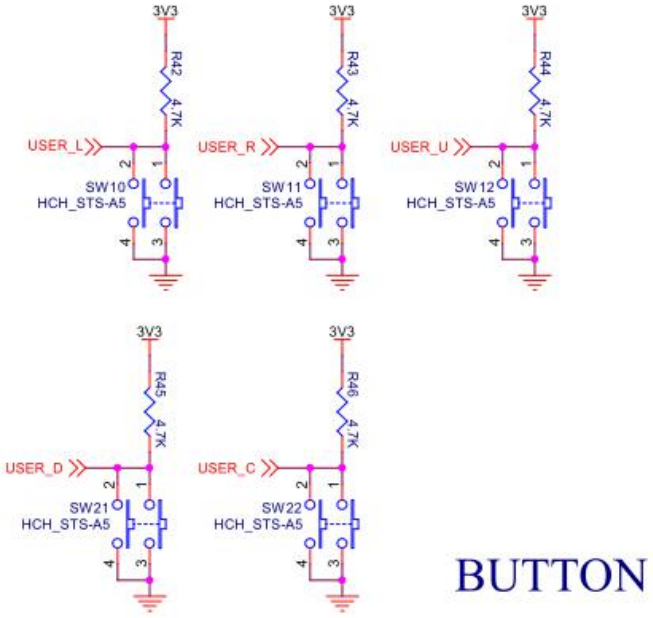


图 5

LED 灯的硬件电路原理图如图 6 所示：

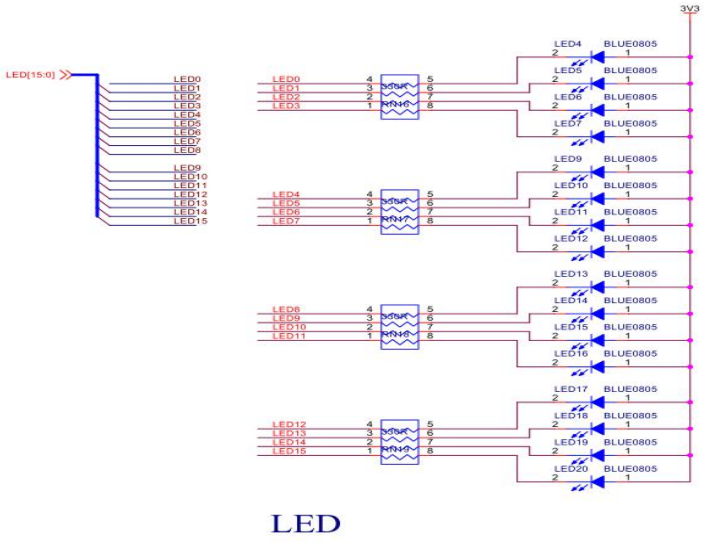


图 6

2.2 功能与现象

当按下一次按键后，对应的 LED 灯将被点亮，再按一次对应的 LED 灯将被熄灭。

2.3 所用资源

本例程在 TD 平台所用资源情况如图 7 所示。

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	22		
#input	6		
#output	16		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	69	19600	0.35%
#reg	45	19600	0.23%
#le	69		
#lut only	24	69	34.78%
#reg only	0	69	0.00%
#lut&reg	45	69	65.22%
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	22	188	11.70%
#ireg	5		
#oreg	0		
#treg	0		
#pll	0	4	0.00%

图 7

3 动态数码管例程

3.1 硬件说明

本例程演示的是七段数码管（不算小数点）的动态显示，原理也很简单，它无非是由 7 个发光二极管组成。这 7 个发光二极管有一个公共端，必须接 GND（共阴极数码管）或者接 VCC（共阳极数码管）。对 7 个二极管的另一端进行控制，相应的就能控制他们的亮暗。不同的亮暗组合就产生了数

字 0-9 的显示效果。若希望数码管显示某个数字，只要给数码管的 7 个段选接口送相应的译码信号即可。数码管具体的电路原理图如图 8 所示：

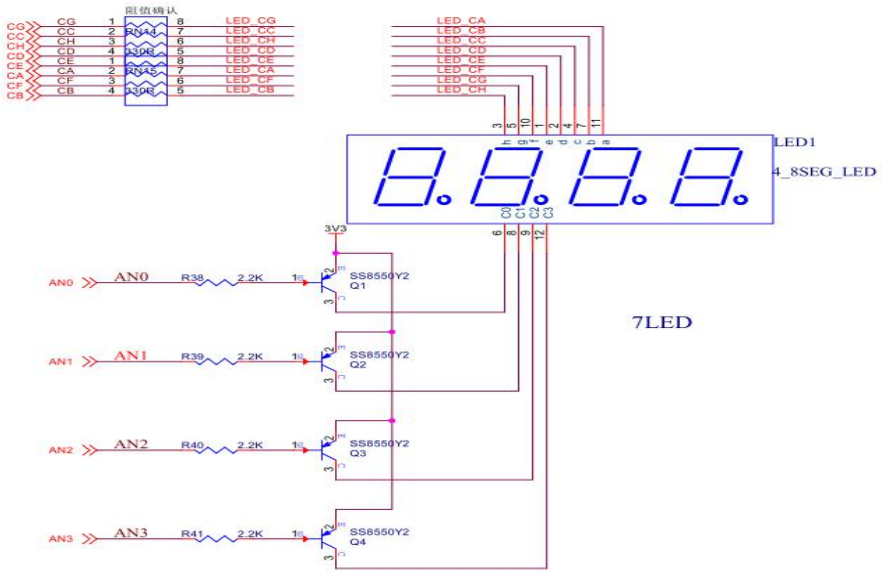


图 8

数码管的动态显示是对每个数码管采用分时复用的方式轮流点亮每个数码管，在同一时间只会点亮一个数码管。分时复用的扫描显示利用了人眼的视觉暂留特性，如果公共端的控制信号刷新速度足够快，人眼就不会区分出 LED 的闪烁，认为 4 个数码管是同时点亮的。

3.2 功能与现象

数码管动态显示的特点：

1、将所有的数码管的段选线并联在一起

2、由位选线控制是哪一位数码管被选中

3、选亮数码管采用动态扫描方式显示

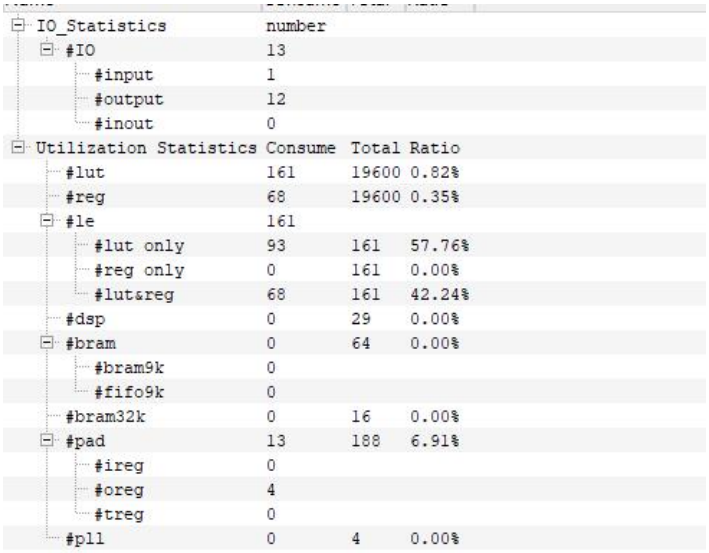
4、即轮流向各位数码管送出字形码和相应的位选线

5、利用发光二极管的余晖效应和人眼的视觉暂留作用，使人感觉好像所有数码管同时在显示

程序下载后，四位数码管从 0 到 9999 之间循环显示，每 0.1 秒进行累加一。

3.3 所用资源

本例程在 TD 平台所用资源情况如图 9 所示。



IO_Statistics	number		
#IO	13		
#input	1		
#output	12		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	161	19600	0.82%
#reg	68	19600	0.35%
#le	161		
#lut only	93	161	57.76%
#reg only	0	161	0.00%
#lut&reg	68	161	42.24%
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	13	188	6.91%
#ireg	0		
#oreg	4		
#treg	0		
#pll	0	4	0.00%

图 9

4 循环流水灯例程

4.1 硬件说明

流水广告灯主要应用于 LED 灯光控制。通过程序控制 LED 的亮和灭， 多个 LED 灯组成一个阵列，依次逐个点亮的时候像流水一样，所以叫流水灯。由于其形成美观大方的视觉效果，因此广泛应用于店铺招牌、广告、大型建筑夜间装饰、景观装饰等。

在本案例中，使用常用的 verilog 语言完成该程序，设计并控制 16 个灯的花式或循环点亮。具体功能要求如下：

上电后，实现左移和右移交替的流水灯。

右移流水灯：16 个灯最左边第一个灯灭，其他灯亮；隔 0.5s 后，第二个灯灭，其他灯亮；隔 0.5s 后，第三个灯灭，其他灯亮；如此类推，直到第 16 个灯灭 0.5s 后进行左移流水灯操作。

左移流水灯：16 个灯最右边第一个灯灭，其他灯亮；隔 0.5s 后第二个灯灭，其他灯亮；再隔 0.5s 后，第三个灯灭，其他灯亮；如此类推，直到第 16 个灯灭 0.5s 后进行右移流水灯操作。具体硬件介绍如图 10 所示。

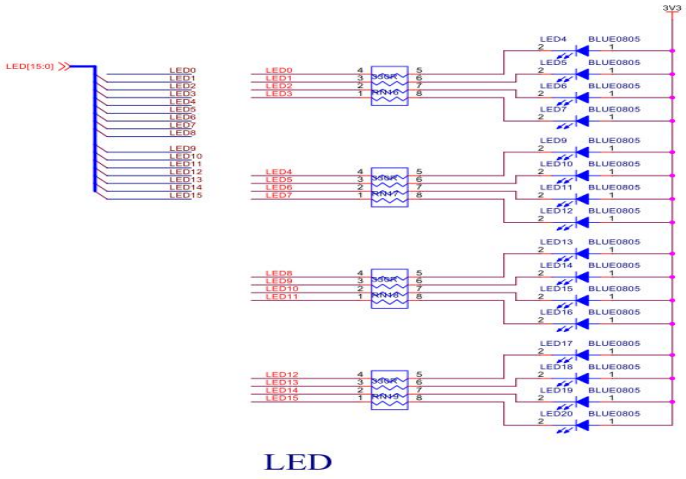


图 10

4.2 功能与现象

左右闪烁，每个灯点亮时长为 0.5s。

4.3 所用资源

本例程在 TD 平台所用资源情况如图 11 所示

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	17		
#input	1		
#output	16		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	77	19600	0.39%
#reg	42	19600	0.21%
#le	94		
#lut only	52	94	55.32%
#reg only	17	94	18.09%
#lut&reg	25	94	26.60%
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	17	188	9.04%
#ireg	0		
#oreg	0		
#treg	0		
#pll	0	4	0.00%

图 11

5 三色灯例程

5.1 硬件说明

RGB 三色 LED 灯其实就是三个灯，具体电路如图 12 所示：

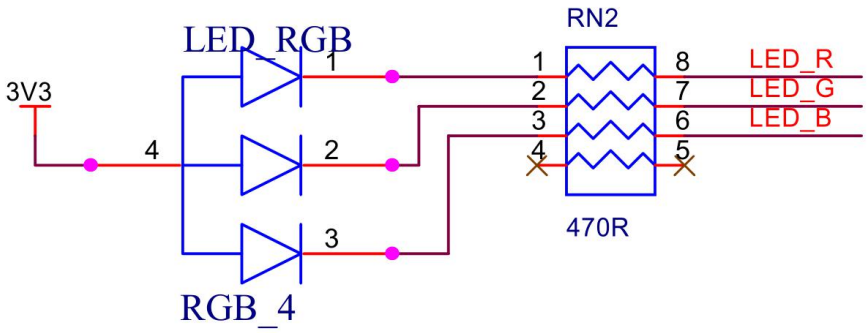


图 12

通过程序控制 LED 的亮和灭， 多个 LED 灯组成一个阵列，依次逐个点亮的时候像流水一样，叫流水灯。也可以通过控制不同颜色的灯进行组合显示出其他颜色。

5.2 功能与现象

依次点亮 RGB 三个灯，类似流水灯效果。

5.3 所用资源

本例程在 TD 平台所用资源情况如图 13 所示。

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	4		
#input	1		
#output	3		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	56	19600	0.29%
#reg	33	19600	0.17%
#le	58		
#lut only	25	58	43.10%
#reg only	2	58	3.45%
#lut&reg	31	58	53.45%
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	4	188	2.13%
#ireg	0		
#oreg	0		
#treg	0		
#pll	0	4	0.00%

图 13

6 UART loopback 例程

6.1 实现功能

FPGA 实时监测 uart_rx 信号是否有数据，若接收到 PC 端发送的数据，则把接收到的数据通过 uart_tx 返回给 PC 端。

6.2 串口通信格式

串口帧格式如图 14 所示，它由 1 个起始位（必须为 0）、8 个数据位、1 个奇偶校验位和 1 或 2 个停止位（必须为 1）组成。除了奇偶校验位，其他三个部分都是必须的。当信号线空闲时，必须为高电平。要发起数据传输时，1 个低电平的脉冲表示起始位，然后连续传输 8 个数据位和若干个高电平的停止位，这样便完成一次传输。

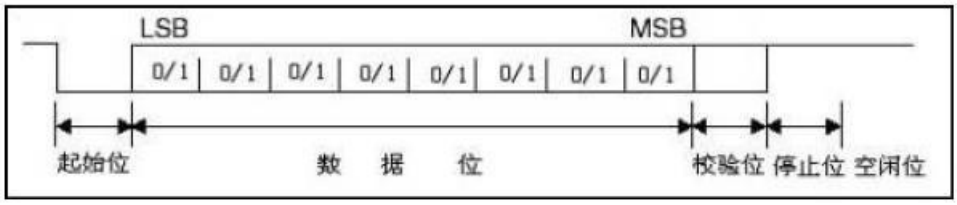


图 14

6.3 实现结构框图

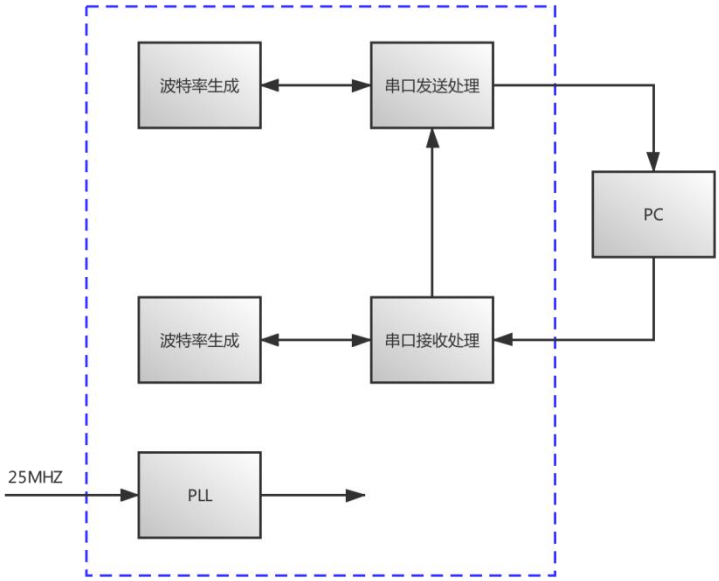


图 15

6.4 各模块功能

模块层次如图 16 所示。

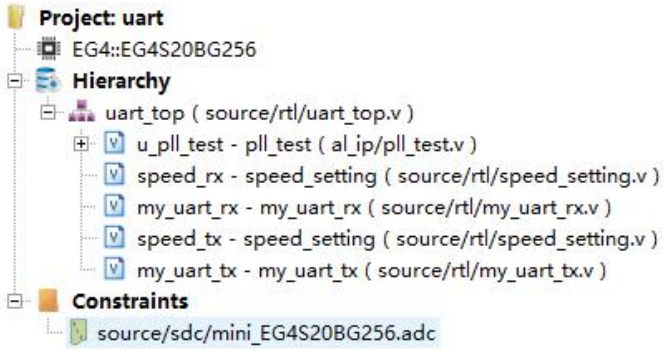


图 16

Uart_top 模块主要是对各模块进行例化，不实现任何逻辑功能。

my_uart_rx 模块主要是完成数据的接收。

speed_setting 模块主要是对 my_uart_rx 模块和 my_uart_tx 模块进行波特率控制。speed_rx 模块和 speed_tx 模块里可以修改收发数据的波特率，如 9600bps，19200bps，38400bps，57600bps 或 115200bps 等。

my_uart_tx 模块在 my_uart_rx 模块接收一个完整的数据帧后启动运行，将接收到的数据作为发送数据返回给 PC 端。

6.5 硬件说明

板卡上已经集成了 USB2UART 模块，根据板卡原理图，在引脚约束时按照下面约束即可。

```
set_pin_assignment {uart_tx} { LOCATION = E16; }        ##使用板卡上USB2UART模块时，收发引脚映射到F16 E16
set_pin_assignment {uart_rx} { LOCATION = F16; }
```

图 17

如果外接 USB2UART 模块，可以根据板卡原理图，将收发端口约束到合适的引脚上。图 18 中是外接 USB2UART 模块时引脚约束的一个实例，也是 DEMO 中测试时的引脚约束。

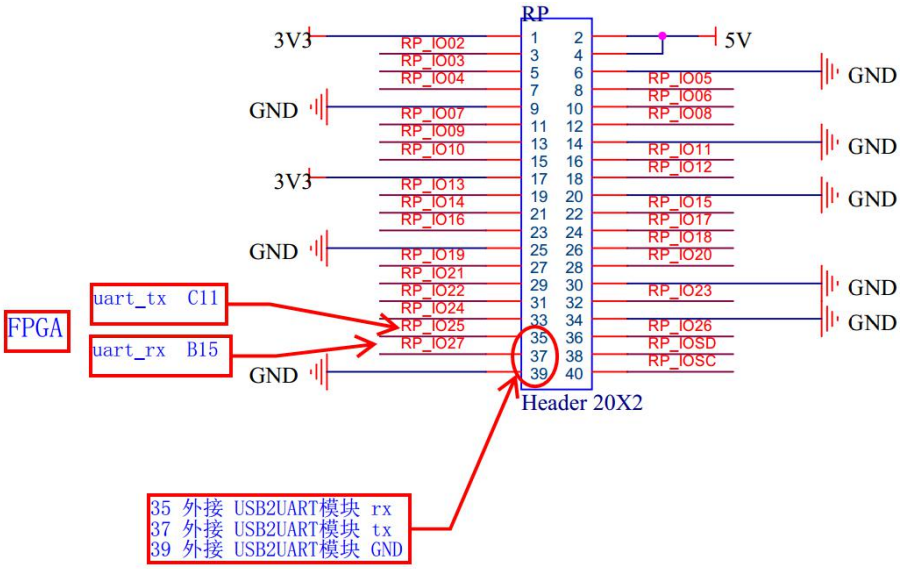


图 18

6.6 测试效果如图 19 所示。

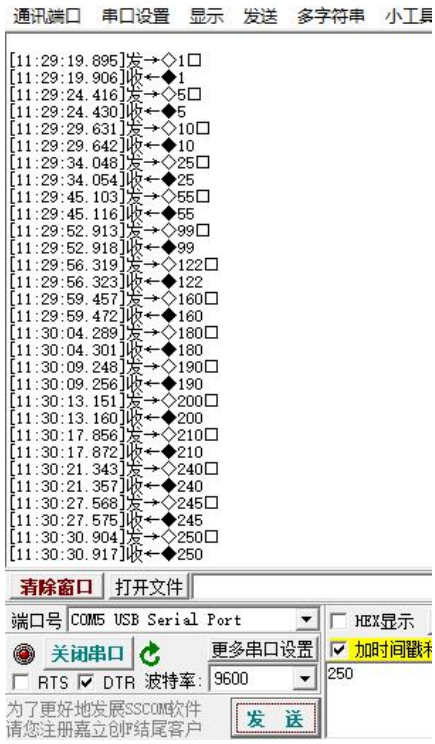


图 19

6.7 所用资源

本例程在 TD 平台所用资源情况如图 20 所示。

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	4		
#input	3		
#output	1		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	110	19600	0.56%
#reg	73	19600	0.37%
#le	128		
#lut only	55	128	42.97%
#reg only	18	128	14.06%
#lut&reg	55	128	42.97%
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	4	188	2.13%
#ireg	0		
#oreg	1		
#treg	0		
#pll	1	4	25.00%

图 20

7 UART 读取 RAM 例程

7.1 实现功能

上电初始在 RAM 中写入与其地址对应的递增数据,写入完成点亮 LED[0]进行指示；此时 RAM 中存储 256 个 8bit 数据，等待 UART 接收到一个字节数据;以此数据作为地址，读出该地址对应的数据;然后通过 UART 发送出去。

7.2 实现结构框图

实现结构框图如图 21 所示。

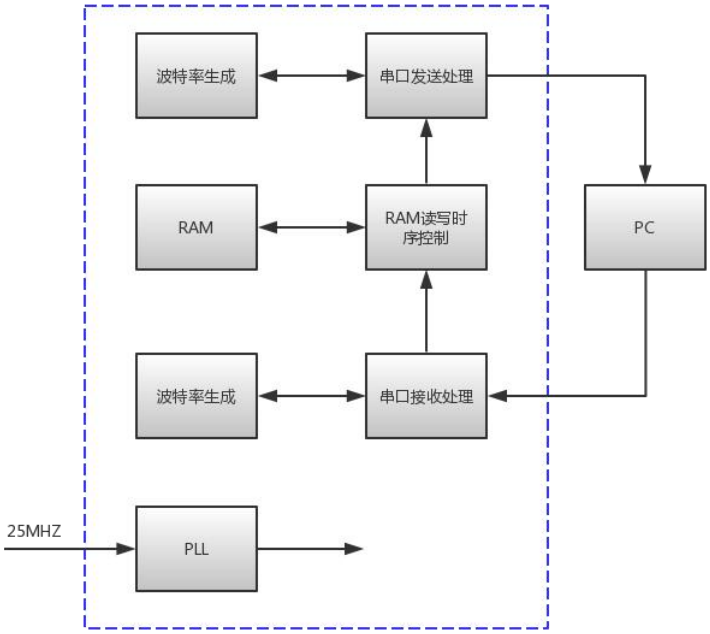


图 21

7.3 各模块功能

模块层次如图 22 所示。

图 22

Uart_top 模块主要是对各模块进行例化，不实现任何逻辑功能。

my_uart_rx 模块主要是完成数据的接收。

speed_setting 模块主要是对 my_uart_rx 模块和 my_uart_tx 模块进行波特率控制。speed_rx 模块和 speed_tx 模块里可以修改收发数据的波特率，如 9600bps，19200bps，38400bps，57600bps 或 115200bps 等。

ram_rw_control 模块产生 RAM 核的基本读写时序，上电初始在 RAM 核中写入与其地址对应的递增数据,写入完成点亮 LED[0]进行指示，然后等待接收读地址数据。在接收到 my_uart_rx 模块的数据后，读取该地址对应的数据，然后发送给 my_uart_tx 模块。

my_uart_tx 模块通过 UART 发送数据给 PC 端。

7.4 测试效果如图 23 所示。

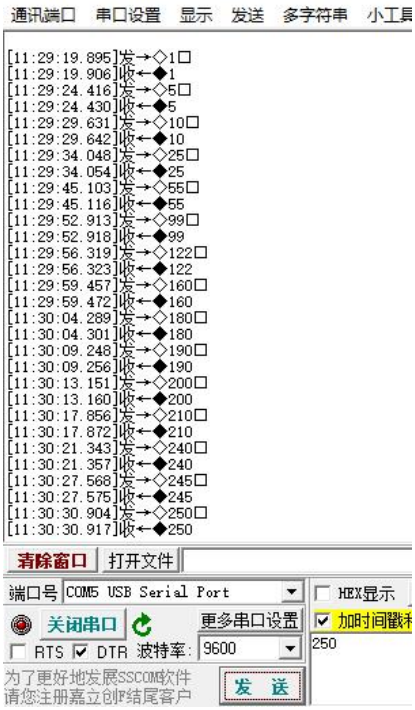


图 23

7.5 所用资源

本例程在 TD 平台所用资源情况如图 24 所示。

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	5		
#input	3		
#output	2		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	142	19600	0.72%
#reg	102	19600	0.52%
#le	175		
#lut only	73	175	41.71%
#reg only	33	175	18.86%
#lut&reg	69	175	39.43%
#dsp	0	29	0.00%
#bram	1	64	1.56%
#bram9k	1		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	5	188	2.66%
#ireg	0		
#oreg	1		
#treg	0		
#pll	1	4	25.00%

图 24

8 VGA 驱动例程

8.1 硬件说明

板卡上集成了专用视频转换 DAC 芯片 GM7123，其原理图如图 25 所示。

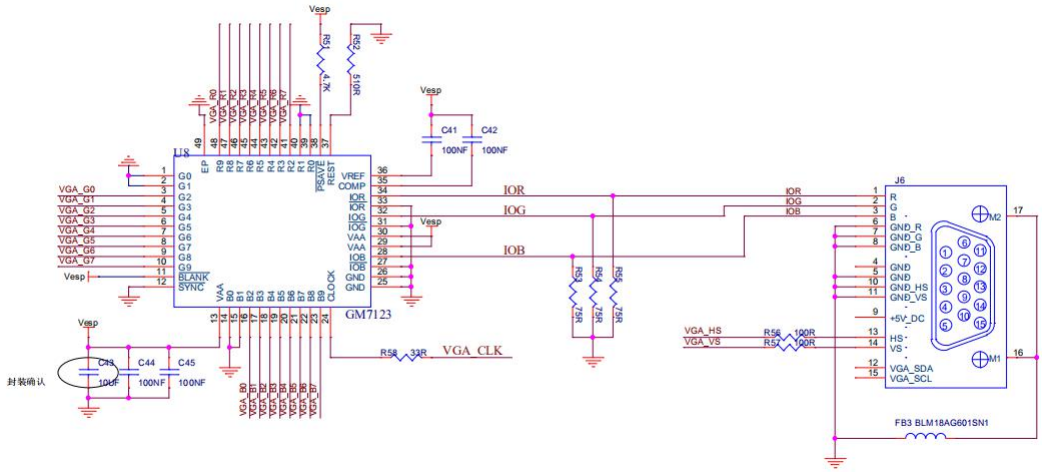


图 25

8.2 各模块功能

模块层次如图 26 所示。

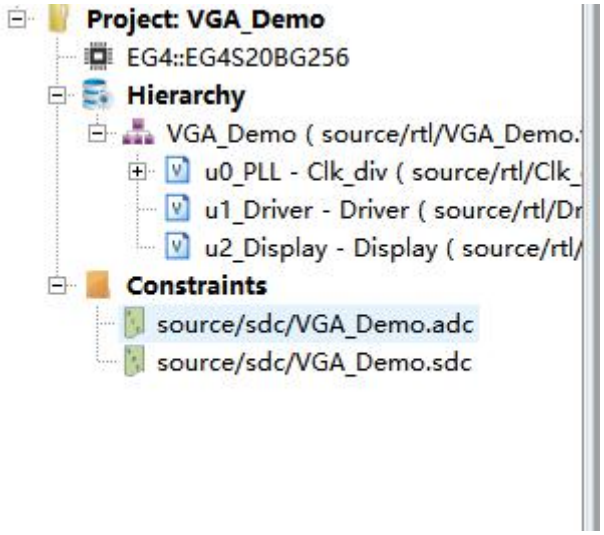


图 26

该 demo 主要分为 3 个模块，用于实现 VGA 的驱动显示功能。其中顶层模块例化了 vga 时钟模块，VGA 驱动模块以及 VGA 显示测试模块。

vga 时钟模块是通过调用 PLL　IP 核产生 VGA 的驱动时钟。需要注意的是选择不同的分辨率要生成不同的驱动时钟。

vga 驱动模块是用于驱动 VGA 显示。用户可以通过改变 VGA 分辨率的时序参数来自主选择实现不同的分辨率。具体时序参数已在注释中给出。

vga 显示模块是用于测试 VGA 显示。该模块定义了 4 种测试模式，可以选择测试不同的显示效果。

8.3 vga 测试效果

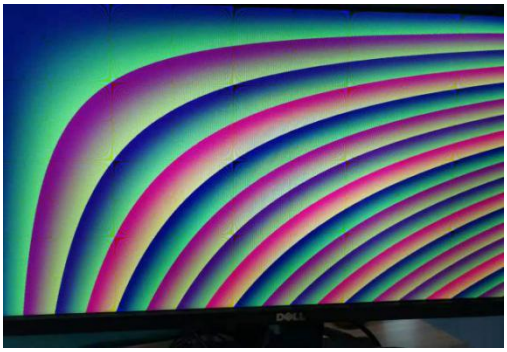
4 种测试模式下的显示效果：

1) 水平色彩条测试效果

2) 垂直色彩条测试效果

2) GRAY GRAPH 测试效果

3) GRAFTAL GRAPH 测试效果



8.4 资源占用

本例程在 TD 平台所用资源如图 27 所示。

Utilization Statistics	Consume	Total	Ratio
#lut	241	19600	1.23%
#reg	24	19600	0.12%
#le	241		
#lut only	217	241	90.04%
#reg only	0	241	0.00%
#lut&reg	24	241	9.96%
#dsp	1	29	3.45%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	30	188	15.96%
#ireg	0		
#oreg	0		
#treg	0		
#pll	1	4	25.00%

图 27

9 OV2640 摄像头 VGA 显示例程

本例程使用 DVP 接口的 OV2640 摄像头作为输入模块，然后通过安路 FPGA 芯片驱动和配置摄像头的工作模式，最后从 VGA 接口进行输出显示。

9.1 硬件说明

整体设计架构如图 28 所示：

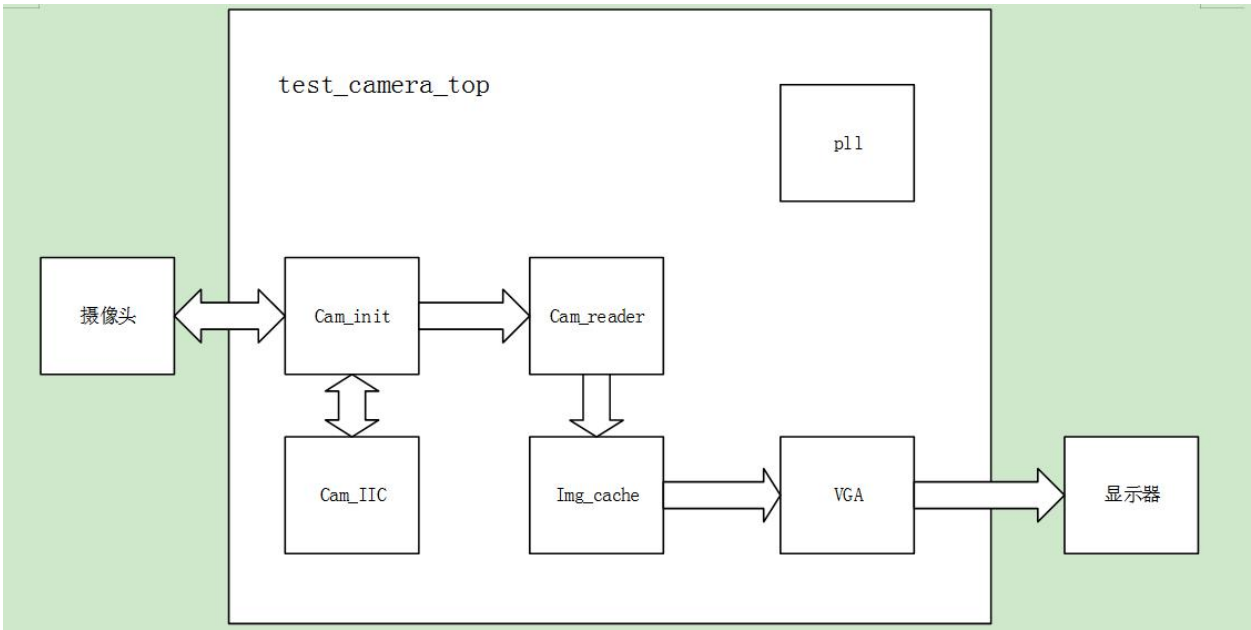


图 28

摄像头初始化通过 IIC 进行寄存器的配置，配置好寄存器后，摄像头的数据产生到 Cam_reader 模块，然后转换成 RGB565 格式缓存到 Img_cache 模块，然后输出到 VGA 模块进行显示输出。

DVP 接口的硬件原理图如图 29、图 30 所示：

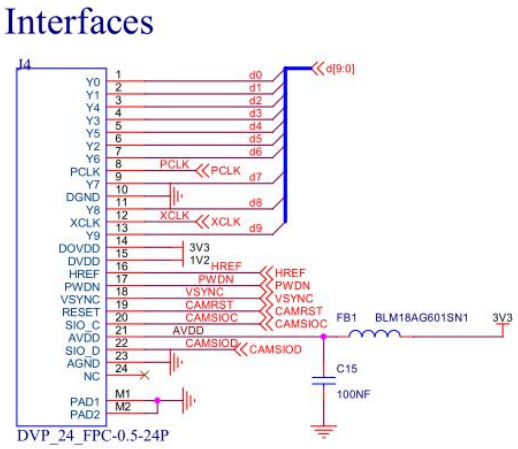


图 29

可以看到 DVP 接口的数据位有 10 个，但是在实际的程序设计中只用到了八位数据，取十位数据的高八位进行输出，这个设置与选择的输出 RGB565 模式有关，具体参考 OV2640 数据手册。

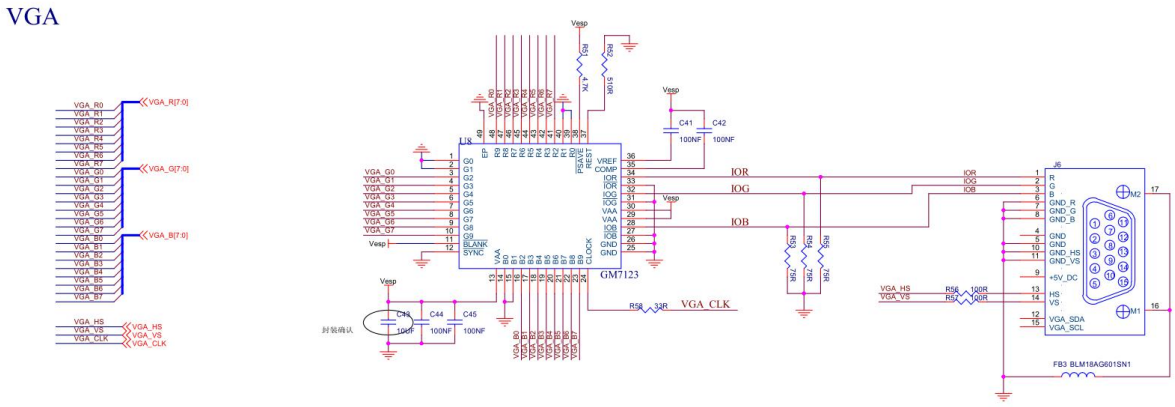


图 30

9.2 功能与现象

本例程需要 SparkRoad 板卡连接 VGA 接口到显示器，将 DVP 接口 OV2640 摄像头连接在 SparkRoad 板卡上。板卡上电下载程序进去就可以看到 VGA 上的实时视频显示。效果如图 31 所示。

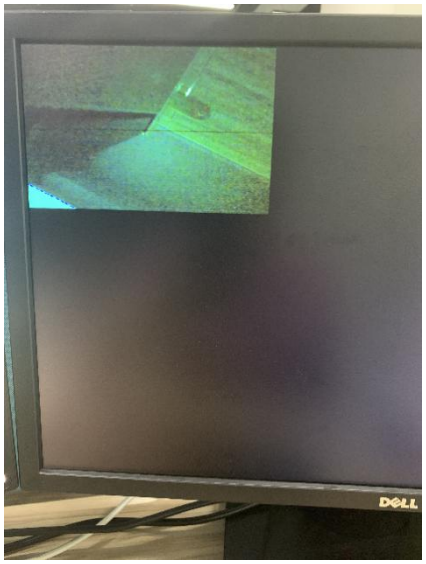


图 31

9.3 所用资源

本例程在 TD 平台所用资源如图 32 所示。

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	45		
#input	13		
#output	31		
#inout	1		
Utilization Statistics	Consume	Total	Ratio
#lut	1624	19600	8.29%
#reg	222	19600	1.13%
#le	1692		
#lut only	1470	1692	86.88%
#reg only	68	1692	4.02%
#lut&reg	154	1692	9.10%
#dsp	1	29	3.45%
#bram	61	64	95.31%
#bram9k	61		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	45	188	23.94%
#ireg	0		
#oreg	0		
#treg	0		
#pll	1	4	25.00%

图 32

10 SDRAM 读写例程

10.1 硬件说明

本例使用安路芯片内部的 SDRAM IP 进行读写功能测试。代码层次关系如图 33 所示：

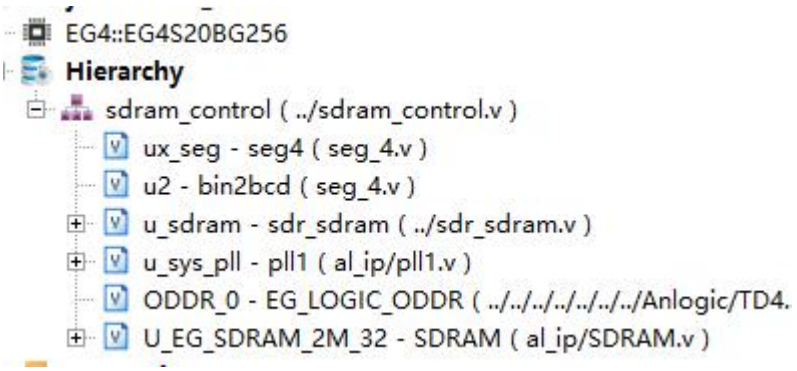


图 33

10.2 功能与现象

使用数码管动态显示数据输出低八位的 BCD 码值，每 0.5 秒刷新一次。

10.3 所用资源

本例程所用到的 TD 软件的资源如图 34 所示。

```
simple
IO Statistics
#IO                      16
  #input                  2
  #output                14
  #inout                  0

Utilization Statistics
#lut                   577   out of  19600    2.94%
#reg                   295   out of  19600    1.51%
#le                    621
  #lut only            326   out of    621   52.50%
  #reg only             44   out of    621    7.09%
  #lut&reg             251   out of    621   40.42%
#dsp                     0   out of     29    0.00%
#bram                    0   out of     64    0.00%
  #bram9k                0
  #fifo9k                0
#bram32k                 0   out of     16    0.00%
#pad                    71   out of    243   29.22%
  #ireg                 24
  #oreg                 23
  #treg                  0
#pll                     1   out of      4   25.00%
#clknet                  3   out of     16   18.75%
  #gclk                  1
```

图 34

11 Sparkroad 上位机控制系统

11.1 FPGA 部分

1. 硬件说明

板卡上为用户提供了两个 flash，其中一片为 FPGA 配置 Flash，一片为用户 flash。其原理图如图 35 所示。该 demo 使用的是用户 flash。

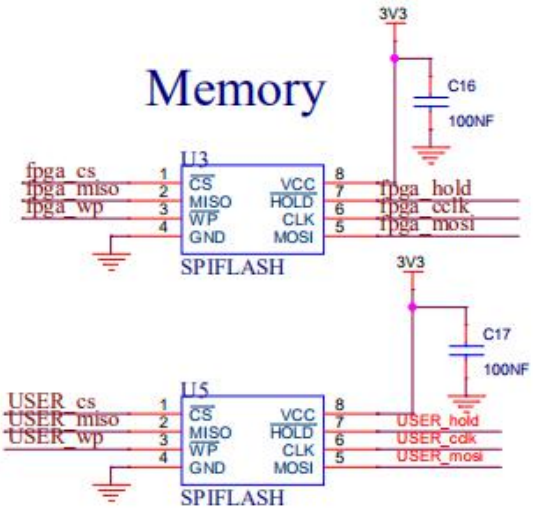


图 35

2. 各模块功能

模块层次如图 36 所示。

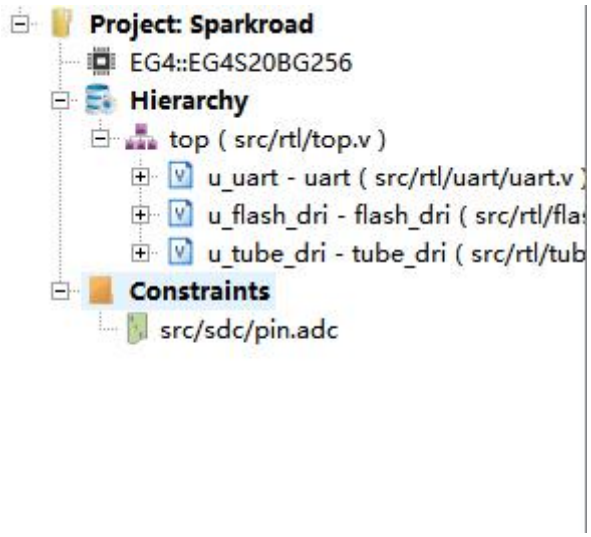


图 36

该 demo 主要分为 3 个模块。其中 Uart 串口传输模块用于实现 FPGA 和上位机之间的交互。其中波特率默认是 115200，用户可自己定义修改。flash 控制器用于控制 Sparkroad 板卡上的用户 flash，主要分为 7 个模块。用户可以通过输入不同的 mod_sel 命令来实现 flash 功能。其中 mod_sel 命令类型定义如表 1 所示。数码管显示模块用于驱动 Sparkroad 板卡上 4 个数码管的显示，其中 4 个数码管分别表示个、十、百、千位。

mod_sel	实现功能
0	等待 flash 空闲
1	flash 写使能
2	擦除整个 flash
3	写入数据(1byte)到 flash
4	从 flash 读取数据(1byte)
5	读取整个 flash.
6	读取芯片 ID
7	擦除 flash 一个扇区

表 1

3. 资源占用

本例程在 TD 平台所用资源如图 37 所示。

Utilization Statistics	Consume	Total	Ratio
#lut	854	19600	4.36%
#reg	1039	19600	5.30%
#le	1483		
#lut only	444	1483	29.94%
#reg only	629	1483	42.41%
#lut&reg	410	1483	27.65%
#dsp	0	29	0.00%
#bram	0	64	0.00%
#bram9k	0		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	36	188	19.15%
#ireg	0		
#oreg	7		
#treg	0		
#pll	0	4	0.00%

图 37

4. 硬件连接图如图 38 所示。

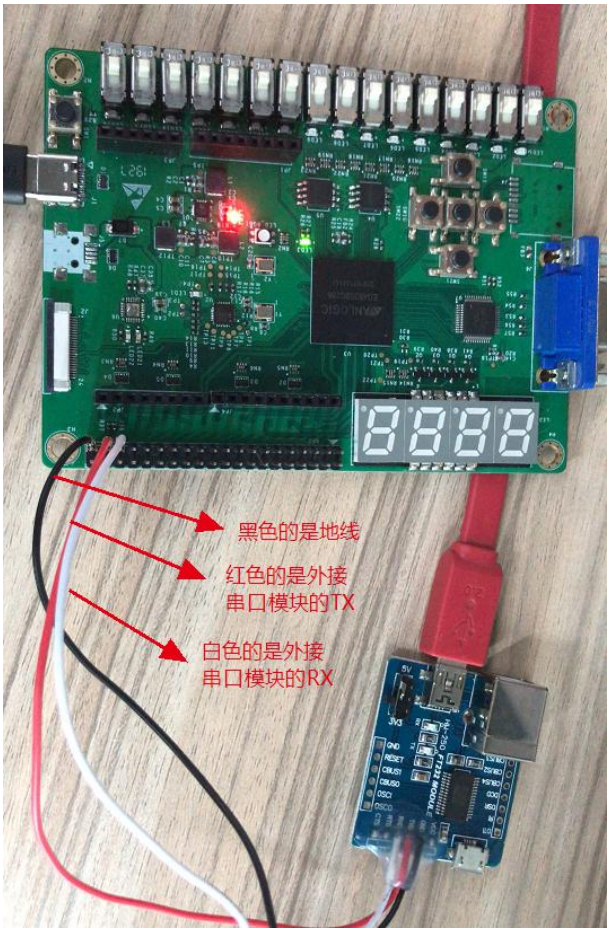


图 38

11.2 Sparkroad 上位机简略说明

11.3 上位机串口通信协议简介

Sparkroad 开发板配套了一个简单的上位机，上位机使用 matlab 软件编写，附件包含 matlab 源码和编译后的 exe 软件，用户可以在此基础上进一步开发。

上位机使用串口与 FPGA 通信，uart 自定义数据帧格式：

帧头	地址	数据

帧头：占 6 个字节，分为读数据帧头，写数据帧头。

写数据帧头：“ABCDEF”。

读数据帧头：“FEDCBA”。

地址：占 4 个字节。

数据：占 4 个字节。

写帧头：依次传输"A","B","C","D","E","F"。

读帧头：依次传输"F","E","D","C","B","A"。

上位机使用 matlab2016a 编译，用户如果电脑上没有安装 matlab2016a 版本，需要在下面网站下载一个运行插件，才可以运行程序。

https://ww2.mathworks.cn/products/compiler/matlab-runtime.html

图 39

如果用户安装了其它版本 matlab，可以直接在 matlab 里面运行 Sparkroad.m 源代码。

11.4 上位机软件界面简介

上位机打开界面如图 40 所示：

图 40

上位机软件分为如下 4 个部分

.串口设置区，包含串口号选择、波特率设置、串口打开与关闭。

.FLASH 读写、FLASH 擦除等操作。

. 板载 LED 灯控制。

. 板载数码管显示，显示数字范围 0-9999.

11.5 上位机软件-连接串口

1. 打开上位机软件如图 41 所示：

图 41

2. 配置串口参数，并打开串口如图 42 所示：

图 42

11.6 上位机软件-控制 LED 灯

SparkRoad-FPGA 程序下载到硬件板后，效果如下，16 个 LED 全部亮起。展示效果如图 43 所示：

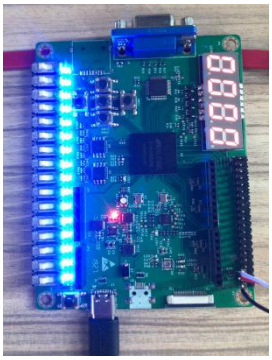


图 43

按照【上位机软件-连接串口】章节连接后，所有 LED 灯熄灭，效果如图 44 所示。

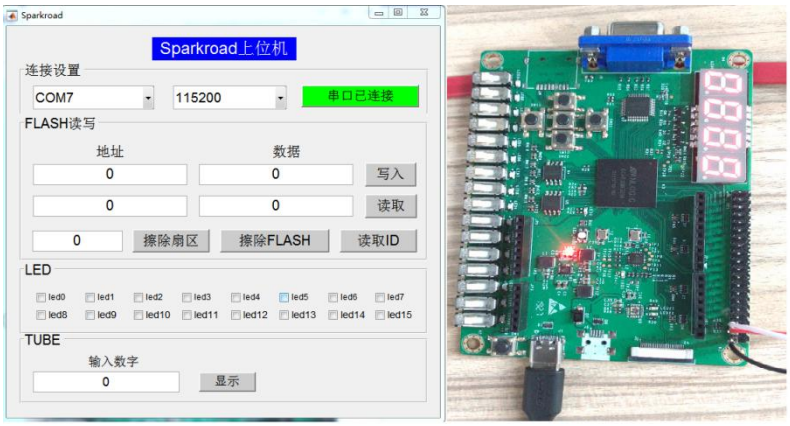


图 44

在软件界面勾选 LED，则对应的板载 LED 灯亮起。效果如图 45 所示。

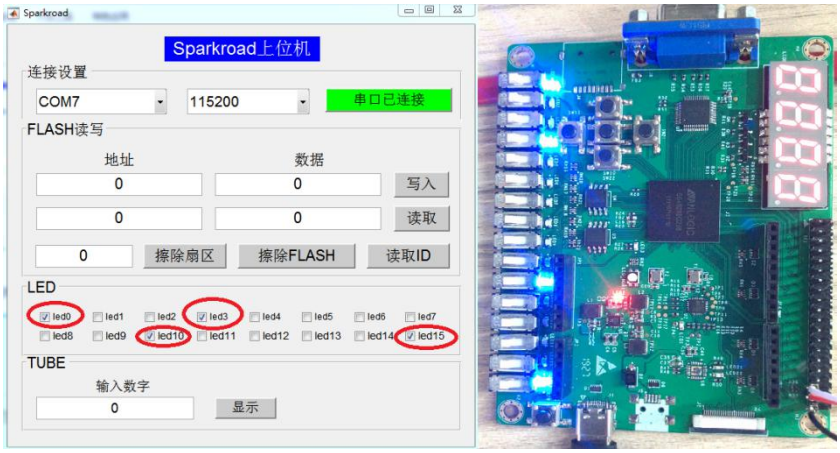


图 45

11.7 上位机软件-控制数码管显示

在软件界面 TUBE 栏目输入一个数字，若数字小于 9999，则显示到数码管上，若数字大于 9999，则显示 9999。展示效果如图 46 所示。

图 46

11.8 上位机软件-FLASH 控制

读取 FLASH ID 号，如图 47 所示：

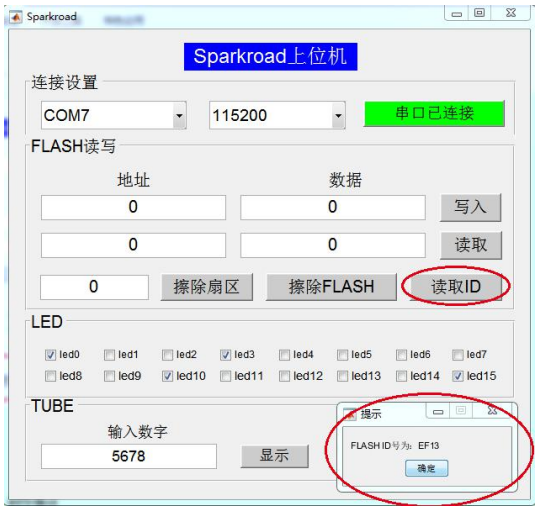


图 47

擦除 FLASH 扇区 / 擦除整个 FLASH 如图 48 所示：

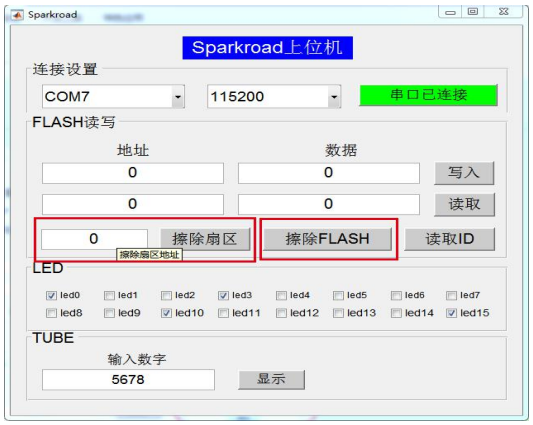


图 48

写入 FLASH 数据(地址 7 写入数据 123)

因为 FLASH 器件的特性，在 FLASH 某个地址写入数据之前，必须确保这个地址原有数据为 255，否则写入的数据不正确。关于 FLASH 的器件特性，请百度自行查阅。

图 49

读取 FLASH 数据(读取地址 7 的数据)

图 50

12 打砖块游戏例程

12.1 游戏说明

下载游戏工程到开发版，接上 vga 显示器，屏幕内部会显示一个打砖块游戏。

游戏说明如图 51 所示。

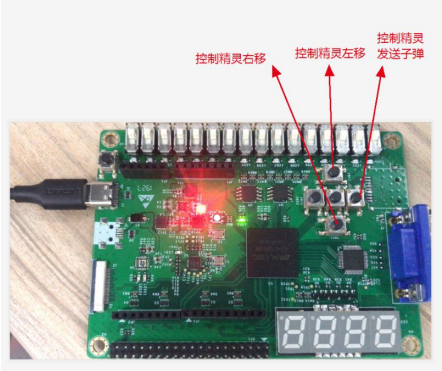


图 51

游戏效果图如图 52 所示。

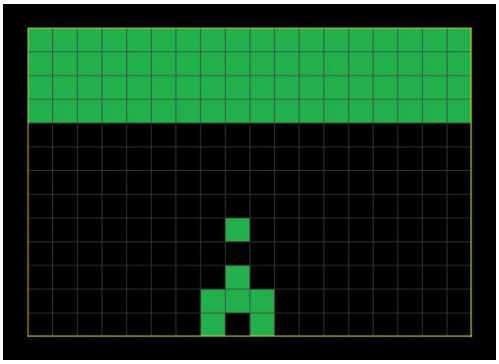


图 52

12.2 所用资源

本例程所用到的 TD 软件的资源如图 53 所示。

game_phy_area

Name	Consume	Total	Ratio
IO_Statistics	number		
#IO	37		
#input	8		
#output	29		
#inout	0		
Utilization Statistics	Consume	Total	Ratio
#lut	1104	19600	5.63%
#reg	822	19600	4.19%
#le	1381		
#lut	559	1381	40.48%
#reg	277	1381	20.06%
#lut&reg	545	1381	39.46%
#dsp	1	29	3.45%
#bram	7	64	10.94%
#bram9k	7		
#fifo9k	0		
#bram32k	0	16	0.00%
#pad	37	188	19.68%
#ireg	0		
#oreg	3		
#treg	0		
#pll	1	4	25.00%

图 53

12.3 相关配套代码说明见网盘下载链接：

https://pan.baidu.com/s/1zkjoTJqcJtdL430a8TW7MA#list/path=%2F

http://blog.sina.com.cn/xqandwn

版本信息

日期	版本	说明
09/08/2019	1.0	初版建立

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任 何知识产权许可。除安路科技在其产品的销售条款和条件中声明的责任之外，安路科技概不承担任何法律或非法律责任。安路科技对安路科技产品的销售和/或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。安路科技对文档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，安路科技保留修改文档中任何内容的权利，恕不另行通知。安路科技不承诺对这些文档进行适时的更新。