基于DE2-115开发板的课程设计

2018年4月16日

目录

1 概述

本项目是基于DE2-115开发板进行开发的。通过按键改变LED流水灯方向，屏幕中打印流水灯当前流向，数码管输出当前被点亮LED的序号。

1.1 DE2-115简介

DE2系列FPGA开发板长久以来因其丰富多样的周边应用界面而广受好评，一直处于教育开发平台的领先地位。DE2-115开发平台配备Cyclone IV E FPGA，除可满足用户移动视频、语音、资料存取与高清影像处理的应用与验证需求外，更在cost、power、logic resource、memory与DSP效能方面提供最佳解决方案，以符合各类型研发应用需求。DE2-115配备Cyclone IV E系列芯片中最大容量的Cyclone EP4CE115 FPGA，提供114,480个逻辑单元（LE）以及高速3.9 Mbits RAM与266 multipliers。除此之外，相较前一代Cyclone FPGA,此开发平台在low cost、low power与functionality层面提供了前所未有的绝佳组合。 DE2-115除了继承DE2系列丰富多样的周边应用界面外，还新增了支持高速Gigabit Ethernet（GbE）的介面。并提供一个High-Speed Mezzanine Card(HSMC)介面，可连接HSMC daughter cards来扩充周边应用，也可透过HSMC cable连续多片DE2-115开发平台，来实现大型ASIC prototype的开发验证。

2 硬件设计

2.1 CPU设计

利用Quartus II的Qsys功能进行项目硬核的CPU设计，添加项目中用到的相应模块：时钟、sdram、epcs、sysid、jtag接口及LED、LCD、数码管和按键的PIO接口，并添加相应中断。连接时钟、复位及相关连线，修改sdram、epcs、LED、LCD、按键及数码管的外部连接名称。以上修改如图1中红框所示。分配总线地址后生成CPU设计，在BDF图中添加设计好的CPU模块和锁相环，如图2。BDF中完成连线后生成符号引脚，编译后分配引脚再次编译即生成可利用USB-BLASTER烧录进DE2-115的FPGA芯片中。

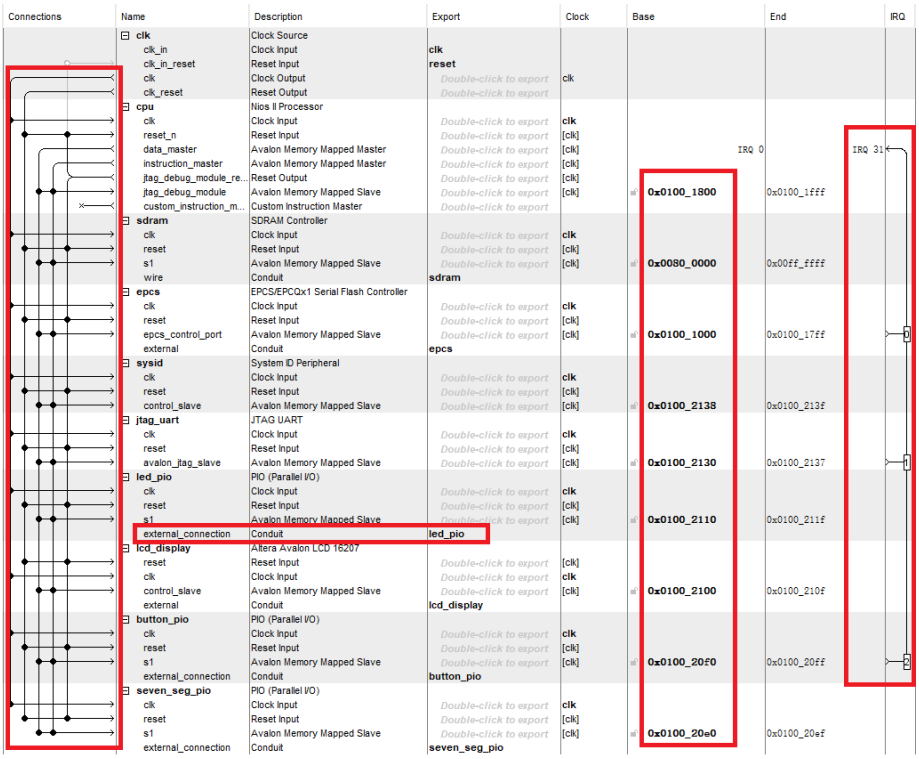

图 1: CPU设计内部逻辑

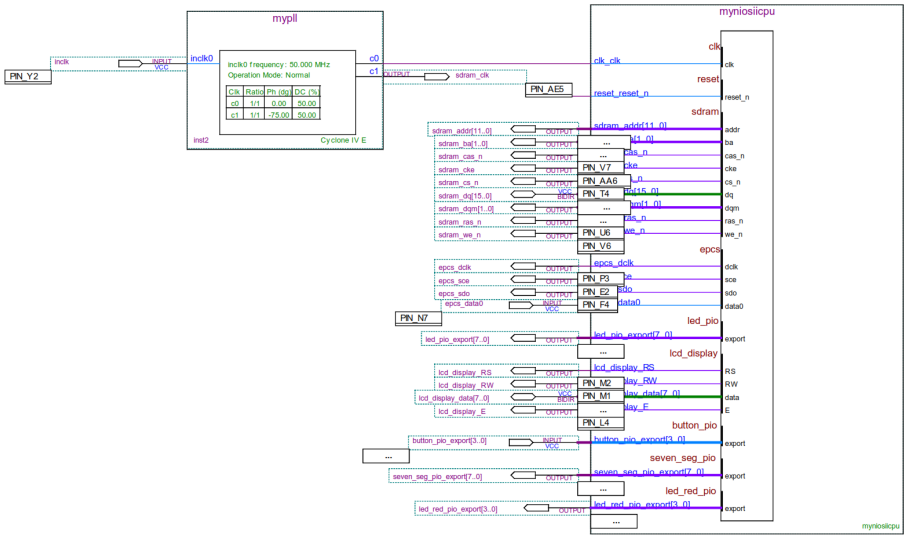

图 2: CPU设计bdf图

2.2 项目相关外设

本项目主要使用了如下外设：

绿色LED 利用8个绿色LED实现流水灯。

按键 利用一个按键实现中断。

LCD16207屏幕 在屏幕中打印流水灯当前流向是向左还是向右。

七段数码管 输出当前被点亮LED的序号，即数码管的数字随流水灯的变化而变化。

3 软件设计

3.1 功能简介

本项目实现了利用按键实现改变8个LED组成的流水灯流向的功能，如图当LED向左流动时，在LCD16027屏幕上打印“left”，如图3，当LED向右流动时，在LCD16207上打印“right”，如图4，同时，使用一个数码管实现显示当前点亮LED的序号。

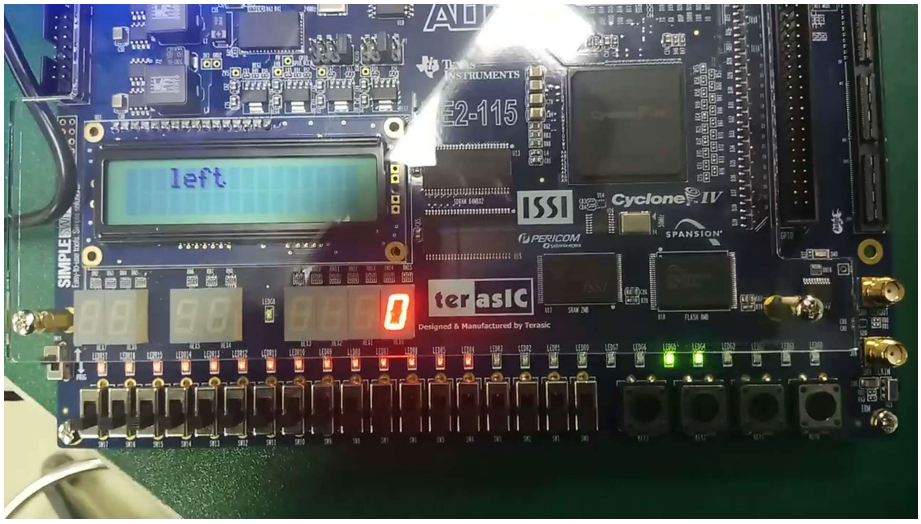

图 3: 向左流动

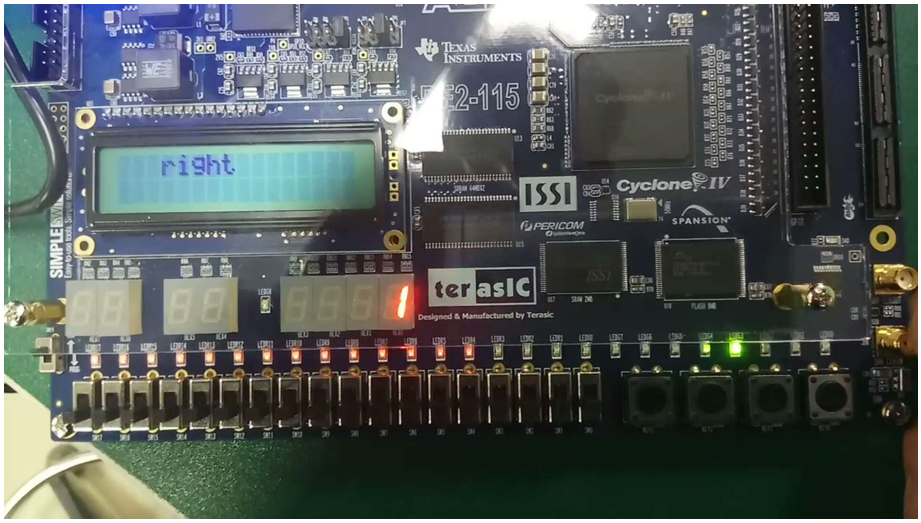

图 4: 向右流动

3.2 流水灯

流水灯是本项目的最基础的功能。如下为使流水灯左移的实现代码：

```
int i;
for (i = 0; i <= 7; i++)
{
        IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_BASE, 1<<i);
        usleep(100000);
}
```

基本原理为定义一个无符号的整型变量i，通过for循环使i增加，调用API函数

```
        IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_BASE, 1<<i);
```

使1左移i位来实现点亮某个LED的功能，然后调用usleep();函数进行延时，使LED的亮灭变化速率能够被人的眼睛识别。使流水灯右移的代码只需修改i的方向，代码如下：

```
int i;
for (i = 7; i >= 0; i--)
{
        IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_BASE, 1<<i);
```

```
        usleep(100000);
}
```

3.3 中断

通过外部按键来产生中断，当按键按下时，就会产生一个上升沿，这时就会进入中断函数。在中断函数中，我们对 key_flag 进行取反。而在主函数中，我们不断地进行查询，当key_flag 为 1 时，流水灯向左移动；当 key_flag 为 0 时，流水灯向右移动。

首先是一个中断处理程序ISR_handle_buttom，当按键中断发生时这个程序就会执行。其中，key_flag 是一个全局变量，每进一次中断，就将 key_flag 的值取反。另外用IOWR_ALTERA_AVALON_PIO_EDGE_CAP函数来清除边沿捕获寄存器，IOWR_ALTERA_AVALON_PIO_EDGE_CAP函数在头文件altera_avalon_pio_regs.h 中定义。

主函数中的按键中断初始化程序 init_button_pio，使用 API 函数IOWR_ALTERA_AVALON_PIO_IRQ_MASK 来使能按键的中断，再用IOWR_ALTERA_AVALON_PIO_EDGE_CAP 函数来清除边沿捕获寄存器。alt_ic_isr_register函数来完成中断的注册，里面的参数PIO_KEY_IRQ_INTERRUPT_CONTROLLER_ID 和PIO_KEY_IRQ 来自 system.h。 ISR_handle_button 是指 ISR 函数，按键中断产生时进入此函数。

```
alt_u8 key_flag = 0;
void ISR_handle_button(void* context)
{
        key_flag = ~key_flag;
        IOWR_ALTERA_AVALON_PIO_EDGE_CAP(PIO_KEY_BASE, 0x0);
           //CLEAR INTERRUPT
}
void init_button_pio(void)
{
        IOWR_ALTERA_AVALON_PIO_IRQ_MASK(PIO_KEY_BASE, 0x1);
           //ENABLE KEY_INTERRUPT
        IOWR_ALTERA_AVALON_PIO_EDGE_CAP(PIO_KEY_BASE, 0x0);
           //DISABLE KEY_EDGE_CAPTURE
        alt_ic_isr_register(PIO_KEY_IRQ_INTERRUPT_CONTROLLER_ID,
            PIO_KEY_IRQ, ISR_handle_button,NULL, 0x0);
}
```

```
int main()
{
        ...
    init_button_pio();
        ...
        while(1)
        {

                if(key_flag)
                {
                        ...
                }
                else
                {
                        ...
                }
        }
        return 0;
}
```

3.4 LCD16207屏幕

LCD16207屏幕主要实现在程序开始运行时打印“SC17023010 JqZHANG”的信息，然后显示当前流水灯流向的功能。定义一个名为lcd的指针，对LCD进行初始化，然后调用LCD_PRINTF()实现打印输出。

```
static void lcd_init( FILE *lcd )
{
        /* If the LCD Display exists,   write a simple message
           on the first line. */
        LCD_PRINTF(lcd, "%c%s SC17023010 JqZHANG", ESC, ESC_TOP_
           LEFT);
}
int main()
{
```

```
        ...
        FILE * lcd;
        /* Initialize the LCD, if there is one.*/
        lcd = LCD_OPEN();
        if(lcd != NULL) {lcd_init( lcd );}
        ...
        while(1)
        {
                if(key_flag)
                {
                        LCD_PRINTF(lcd, "%c%s %c%s  left \n",
                           ESC, ESC_TOP_LEFT, ESC, ESC_CLEAR,
                           ESC, ESC_COL1_INDENT5);
                ...
                }
                else
                {
                        LCD_PRINTF(lcd, "%c%s %c%s  right \n",
                           ESC, ESC_TOP_LEFT, ESC, ESC_CLEAR,
                           ESC, ESC_COL1_INDENT5);
                ...
                }
        }
        return 0;
}
```

3.5 七段数码管

为了实现七段数码管能够随着LED位置的变化而改变数字，即显示当前被点亮的LED的序号，本人定义了一个整型函数led7(int j);其实现的功能是输入整型变量j，利用switchcase语句返回十六进行的数字，即实现译码功能。然后在主循环中在for语句中调用sevenseg_set_hex(led7(i))函数实现输出。

```
/* Seven Segment Display PIO Functions
* sevenseg_set_hex() --  implements a hex digit map.
```

```
*/
#ifdef SEVEN_SEG_PIO_BASE
        static void sevenseg_set_hex(int hex)
        {
                static alt_u8 segments[16] = {
                0x81, 0xCF, 0x92, 0x86, 0xCC, 0xA4, 0xA0, 0x8F,
                   0x80, 0x84,     /* 0-9 */
                0x88, 0xE0, 0xF2, 0xC2, 0xB0, 0xB8 }; /* a-f */

                unsigned int data = segments[hex & 15] | (
                   segments[(hex >> 4) & 15] << 8);
                IOWR_ALTERA_AVALON_PIO_DATA(SEVEN_SEG_PIO_BASE,
                   data);
        }
#endif
int led7(int j)
{
        switch (j)
        {
                case 0:
                        return 0x81;
                case 1:
                        return 0xCF;
                case 2:
                        return 0x92;
                case 3:
                        return 0x86;
                case 4:
                        return 0xCC;
                case 5:
                        return 0xA4;
                case 6:
                        return 0xA0;
                case 7:
```

```
                        return 0x8F;
                default:
                        return 0x80;
        }
}
int main()
{
        ...
        while(1)
        {
                if(key_flag)
                {
                        ...
                        for (i = 0; i < 8; i++)
                        {
                        sevenseg_set_hex(led7(i));
                        ...
                        }
                }
                else
                {
                        ...
                        for (i = 7; i >= 0; i--)
                        {
                        sevenseg_set_hex(led7(i));
                        ...
                        }
                }
        }
        return 0;
}
```

4 总结

经过一个学期的“可编程逻辑器件原理与应用”课程的学习，我初步了解了可编程逻辑器件的原理，学习了Verilog HDL，并利用自己所学使用Quartus II完成了五个实验。最终的课程大作业使用NIOS II在DE2-115开发板上完成流水灯项目的开发，虽然课程设计的项目相对简单，但是已经对于入门已经完全足够，所谓“师傅领进门，修行在个人”。可编程逻辑器件在现代电子的设计中已经愈发重要，相信在以后的学习和科研中，“可编程逻辑器件原理与应用”这门课一定会给我带来很大帮助。正如宋老师在最后一节课上所分享的小故事，你所上过的课可能在你意想不到的地方给你带来意想不到的收获：在某一次的博士论文审稿时他发现这篇论文很有自己讲课时的影子，经过查询后确认这位博士生正是自己当年教过的学生。

最后感谢宋克柱老师一个学期的教导，和助教在实验中的指导。

A main.c

```
#include "count_binary.h"
#include "altera_avalon_timer_regs.h"
#include "alt_types.h"


/* Seven Segment Display PIO Functions
 * sevenseg_set_hex() --  implements a hex digit map.
 */
#ifdef SEVEN_SEG_PIO_BASE
static void sevenseg_set_hex(int hex)
{
    static alt_u8 segments[16] = {
        0x81, 0xCF, 0x92, 0x86, 0xCC, 0xA4, 0xA0, 0x8F, 0x80, 0
           x84, /* 0-9 */
        0x88, 0xE0, 0xF2, 0xC2, 0xB0, 0xB8 };
                                /* a-f */

    unsigned int data = segments[hex & 15] | (segments[(hex >>
       4) & 15] << 8);

    IOWR_ALTERA_AVALON_PIO_DATA(SEVEN_SEG_PIO_BASE, data);
}
#endif


alt_u8 key_flag = 0;


void ISR_handle_button(void* context)
{
        key_flag = ~key_flag;
        IOWR_ALTERA_AVALON_PIO_EDGE_CAP(PIO_KEY_BASE, 0x0);//
```

```
            CLEAR INTERRUPT
}

void init_button_pio(void)
{
        IOWR_ALTERA_AVALON_PIO_IRQ_MASK(PIO_KEY_BASE, 0x1);//
           ENABLE KEY_INTERRUPT
        IOWR_ALTERA_AVALON_PIO_EDGE_CAP(PIO_KEY_BASE, 0x0);//
           DISABLE KEY_EDGE_CAPTURE
        alt_ic_isr_register(PIO_KEY_IRQ_INTERRUPT_CONTROLLER_ID,
            PIO_KEY_IRQ, ISR_handle_button,NULL, 0x0);

}


static void lcd_init( FILE *lcd )
{
    /* If the LCD Display exists, write a simple message on the
       first line. */
    LCD_PRINTF(lcd, "%c%s SC17023010 JqZHANG", ESC,
                ESC_TOP_LEFT);
}

static void initial_message()
{
    printf("\n\n**************************\n");
    printf("* Hello from Nios II!    *\n");
    printf("* SC17023010 JqZHANG     *\n");
    printf("**************************\n");
}

int led7(int j)
{
        switch (j)
```

```
        {
                case 0:
                        return 0x81;

                case 1:
                        return 0xCF;

                case 2:
                        return 0x92;

                case 3:
                        return 0x86;

                case 4:
                        return 0xCC;

                case 5:
                        return 0xA4;

                case 6:
                        return 0xA0;

                case 7:
                        return 0x8F;

                default:
                        return 0x80;

        }
}



int main()
```

```
{
        int i;
    FILE * lcd;
        initial_message();
    /* Initialize the LCD, if there is one.
     */
    lcd = LCD_OPEN();
    if(lcd != NULL) {lcd_init( lcd );}

    init_button_pio();

        while(1)
        {

                if(key_flag)
                {

                        LCD_PRINTF(lcd, "%c%s %c%s  left \n",
                           ESC, ESC_TOP_LEFT, ESC, ESC_CLEAR,
                           ESC, ESC_COL1_INDENT5);
                        for (i = 0; i < 8; i++)
                        {
                                sevenseg_set_hex(led7(i));
                                IOWR_ALTERA_AVALON_PIO_DATA(PIO_
                                   LED_BASE, 1<<i);
                                usleep(300000);
                        }

                }

                else
                {

                        LCD_PRINTF(lcd, "%c%s %c%s  right \n",
```

```
                        ESC, ESC_TOP_LEFT, ESC, ESC_CLEAR,
                        ESC, ESC_COL1_INDENT5);
                    for (i = 7; i >= 0; i--)
                    {
                        sevenseg_set_hex(led7(i));
                        IOWR_ALTERA_AVALON_PIO_DATA(PIO_
                            LED_BASE, 1<<i);
                        usleep(300000);
                    }
                }



        }
        return 0;
}
```

B LEDWaterLamp.h

```
#include "alt_types.h"
#include "altera_avalon_pio_regs.h"
#include "sys/alt_irq.h"
#include "system.h"
#include <stdio.h>
#include <unistd.h>

#ifndef LCD_DISPLAY_NAME
/* Some hardware is not present because of system or because of
   simulation */
#   define LCD_CLOSE(x) /* Do Nothing */
#   define LCD_OPEN() NULL
#   define LCD_PRINTF(lcd, args...) /* Do Nothing */

#else
```

```
/* With hardware devices present, use these definitions */
#   define LCD_CLOSE(x) fclose((x))
#   define LCD_OPEN() fopen("/dev/lcd_display", "w")
#   define LCD_PRINTF fprintf

#endif

/* Cursor movement on the LCD */
/* Clear */
#define ESC 27
/* Position cursor at row 1, column 1 of LCD. */
#define ESC_CLEAR "K"
/* Position cursor at row1, column 5 of LCD. */
#define ESC_COL1_INDENT5 "[1;5H"
/* Position cursor at row2, column 5 of LCD. */
#define ESC_COL2_INDENT5 "[2;5H"
/* Integer ASCII value of the ESC character. */
#define ESC_TOP_LEFT "[1;0H"
```