厦门大学

XIAMEN UNIVERSITY

嵌入式系统开发

实验二 I2C 与 OLED 屏显示

姓 名	
学 号	
学 院	信息学院
专 业	计算机科学与技术专业
年 级	2022
日 期	2024/11/22

目录

1 实验目的

（1） 掌握 I2C 总线通讯的基本原理和使用 I2C 读写 EEPROM 的方法。

（2） 掌握使用 OLED 显示屏显示数据的方法

2 实验方案设计

2.1 基础实验

（1）实验要求

定义 Tab="日期+学号后三位+姓名英文首字母"，使用 I2c 将 Tab 写入 EEPROM。将 EEPROM 中内容读出并存放在 ReadE2P 中。将 Tab 的内容与 ReadE2P 的内容同时显示在 LCD 屏上，若二者内容相等则显示 "="，LED1 灯亮，若不相等显示"≠"，LED2 灯亮。

（2）实验步骤

步骤一:创建工程文件,在 Soft Drive 文件中新建 EEPROM 驱动文件、OLED 驱动文件以及包含的 include 头文件。

步骤二:在 I2c.c 文件中配置 PB10、PB11 管脚为 I2C1 总线模式，使能 I2c1 总线，在 at24cxx.c 文件中根据 EEPROM 通讯协议编写 EEPROM 读写程序。

步骤三:在 oled.c 文件中配置 PB3、PB5 管脚为 SPI0 总线模式，配置 PB4、PB13、PB14 为普通 GPIO 口输出，根据 OLED 通讯协议编写 OLED 驱动程序，完成在 OLED 对应位置显示字符串、汉字、图片等相关函数。利用如图 3.7 所示的 OLED 取模工具，可以得到需要图片或文字的 16 进制格式。将得到的字模数组存到 oledfont.h 文件中,可以根据实验需求通过 OLED 的相关函数调用即可在 OLED 屏上对应位置显示字符，

本实验中需要制作得到“=”和“≠”两个字符的 16 进制格式数组。

步骤四:硬件连接上需要把 H16 的三个跳线冒连接上。

2.2 提高实验

（1）实验要求

自己制作 bmp 格式的图片，在 OLED 中显示图片，在 EEPROM 中存数字，比如 00、01、02、03, 每三秒从中取显示对应数字的图片，显示三秒

（2）实验步骤

在基础实验的基础上，编写相对应的程序逻辑，下载验证实验完成实验内容。

2.3 拓展实验

（1）实验要求

分五次将二字词语写入 EEPROM。随机读出一个二字词语并将其显示在 OLED 屏的左方，再随机读出 1 个二字词语显示在 OLED 屏右方。若二者相等，则 OLED 屏下方显示 Bingo，词语消失，同时 EEPROM 中删去该词。若二者不等，则 OLED 屏下方显示 False，同时右边的词语消失，程序再次从 EEPROM 中读出一个词语进行配对。所有词语配对成功后清空 OLED 和 EEPROM,而后屏上显示“Congradulations!”

3 实验过程与结果

3.1 基础实验

3.1.1 I2C 与 OLED 屏显示文字

首先在 main 函数中中调用 systeminit()函数，执行初始化

```
void systemInit(void)
{
    systick_config();    // 系统时钟配置
    gd_XII_systeminit(); // 实验箱外设初始化

    OLED_Gpio_Init();  // OLED 管脚复用初始化配置
    OLED_Init();       // OLED 初始化
    i2c_gpio_config(); // i2c 管脚配置
    i2c_config();      // i2c 配置
    i2c_eeprom_init(); // eeprom 初始化
    OLED_Clear();
}
```

①定义全局变量 tab，定义了一个字符串数组 tab，用于存储要写入 EEPROM 的数据。

定义了 BUFFER_SIZE，它是 tab 数组的大小，用于确定读写操作的数据量。

定义了两个数组 i2c_buffer_write 和 i2c_buffer_read，分别用于存储写入和读取的数据。

②函数 i2c_24c02_test 是测试 I2C EEPROM 读写的核心函数：使用一个 for 循环将 tab 数组中的数据复制到 i2c_buffer_write 数组中。调用 OLED_ShowString 函数在 OLED 显示屏上显示 tab 数组的内容。调用 eeprom_buffer_write 函数将数据写入 EEPROM 的指定页面。调用 eeprom_buffer_read 函数从 EEPROM 的指定页面读取数据到 i2c_buffer_read 数组。再次调用 OLED_ShowString 函数显示读取的数据。使用

两个 for 循环比较写入和读取的数据是否一致，如果不一致则返回 I2C_FAIL。如果数据一致，则返回 I2C_OK。

③test1

清空 OLED 显示屏。进入一个无限循环，不断执行 I2C EEPROM 的读写测试。如果测试成功（返回 I2C_OK），则点亮 LED1，显示中文“对”，并通过串口打印“对”；如果测试失败，则点亮 LED2，显示中文“错”，并通过串口打印“错”。

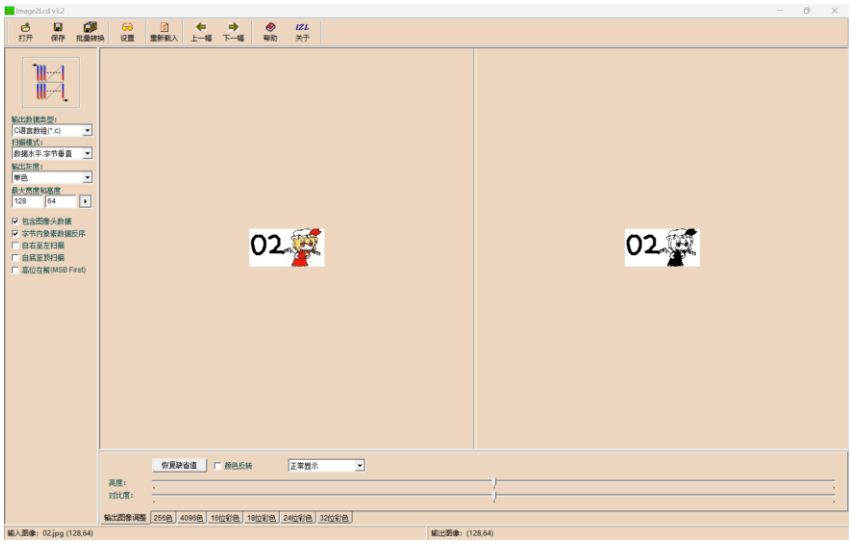
```
uint8_t tab[] = "2024/11/15+???+XXW";
const uint8_t BUFFER_SIZE = sizeof(tab) / sizeof(tab[0]);
uint8_t i2c_buffer_write[BUFFER_SIZE];
uint8_t i2c_buffer_read[BUFFER_SIZE];

uint8_t i2c_24c02_test(void)
{
    uint16_t i;
    for (i = 0; tab[i] != '\0'; i++)
    {
        i2c_buffer_write[i] = tab[i];
    }
    OLED_ShowString(0, 0, tab);
    eeprom_buffer_write(i2c_buffer_write, EEP_FIRST_PAGE, BUFFER_SIZE);
    eeprom_buffer_read(i2c_buffer_read, EEP_FIRST_PAGE, BUFFER_SIZE);
    OLED_ShowString(0, 4, i2c_buffer_read);
    for (i = 0; i < BUFFER_SIZE; i++)
    {
        //    ReadE2P[i] = i2c_buffer_read[i];//会报错
    }
    for (i = 0; i < BUFFER_SIZE; i++)
    {
        if (i2c_buffer_read[i] != i2c_buffer_write[i])
        {
            return I2C_FAIL;
        }
    }
    return I2C_OK;
}

/* 基础实验 */
void test1(void)
```

```
{
    OLED_Clear();
    while (1)
    {
        if (I2C_OK == i2c_24c02_test())
        {
            gd_eval_led_on(LED1);
            OLED_ShowCHinese(0, 2, 1); // ???
            Uart0Printf("对\r\n");
            // Res[0] = Res1[0];
        }
        else
        {
            gd_eval_led_on(LED2);
            OLED_ShowCHinese(0, 2, 0);
            Uart0Printf("错\r\n");
            // Res[0] = Res0[0];
        }
        // json_student_imformation03 (machinenum, tab1, ReadE2P, Res);
        while (1)
        {
        }
    }
}
```

3.2 提高实验

3.2.1 I2C 与 OLED 屏显示图片

先制作一个 128*64 的图片，然后用图片取模软件取模。



定义一个索引和图片数组。这里图片数组用了二维数组，方便后边的切换。

```
int idx = 0;
unsigned char gImage[][1030]; //具体值略
```

接下来只需要调用展示图片的函数，延时三秒之后索引加一。

```
void test2(void)
{
    while (1)
    {
        OLED_DrawBMP(0, 0, 127, 7, gImage[idx]);
        OLED_Display_On();
        delay_1ms(3000);
        idx = (idx + 1) % 4;
    }
}
```

3.3 拓展实验

```
#define WORD_COUNT 5
#define WORD_LENGTH 5
#define EEPROM_START_ADDRESS 0x00

uint8_t words[WORD_COUNT][WORD_LENGTH] = {"我要", "成为", "编程", "高手", "好耶
"};
uint8_t vis[WORD_COUNT];
```

```
uint8_t
congratulations[]="Congratulations!",bingo[]="Bingo",false_str[]="False";
uint8_t word_count=WORD_COUNT;

void writeWordsToEEPROM(uint8_t);
void displayWords(uint8_t, uint8_t);
uint8_t getRandomWordIndex(void);
void clearEEPROM(uint8_t,uint16_t);

// 写入 EEPROM
void writeWordsToEEPROM(uint8_t start_address) {
    for (uint8_t i = 0; i < WORD_COUNT; i++) {
        eeprom_buffer_write(words[i], start_address + i * WORD_LENGTH,
WORD_LENGTH);
    }
}
// 获取随机数
uint8_t getRandomWordIndex(void) {
    return rand() % WORD_COUNT;
}


void clearEEPROM(uint8_t start_address, uint16_t size) {
    uint8_t empty_buffer[WORD_LENGTH] = {0};
    for (uint16_t i = 0; i < size; i += WORD_LENGTH) {
        eeprom_buffer_write(empty_buffer, start_address + i, WORD_LENGTH);
    }
}

void test3(void) {
    //srand(time(NULL));//会出错
    writeWordsToEEPROM(EEPROM_START_ADDRESS);
        uint8_t left_index,right_index;
        uint8_t is_bingo=1;
        word_count=WORD_COUNT;
        delay_1ms(1000);
    while(word_count){
            if(is_bingo){
                left_index=getRandomWordIndex();
                while(vis[left_index]){
                    left_index=getRandomWordIndex();
                }
            }
            right_index=getRandomWordIndex();
```

```
            while(vis[right_index]){
                right_index=getRandomWordIndex();
            }

            OLED_ShowCHinese(0, 0,left_index*2+3);
            OLED_ShowCHinese(16, 0,left_index*2+4);
            OLED_ShowCHinese(64, 0,right_index*2+3);
            OLED_ShowCHinese(80, 0,right_index*2+4);


            if(left_index==right_index){
                OLED_ShowString(0,4,bingo);
                vis[left_index]=1;
                word_count--;
                is_bingo=1;
            }
            else{
                OLED_ShowString(0,4,false_str);
                is_bingo=0;
            }
            delay_1ms(1000);
            OLED_Clear();
        }
    OLED_ShowString(0,4,congratulations);
}
```

4 实验分析

在本次实验中，我成功地实现了 I2C 总线通讯与 OLED 显示屏的结合应用，包括基础的文本显示、图片显示以及更复杂的数据存储与匹配显示。

4.1 I2C 总线通讯原理与实现

I2C 总线是一种同步的、多主机、多从机、基于两条线的串行通讯总线。在实验中，我通过配置 PB10 和 PB11 管脚为 I2C1 总线模式，并使能 I2C1 总线，实现了与 EEPROM 的通讯。通过编写 EEPROM 读写程序，我能够将数据写入 EEPROM，并从 EEPROM 中读取数据。这一过程验证了 I2C 通讯协议的实现和 EEPROM 的读写功能。

4.2 OLED 显示屏数据展示

OLED 显示屏作为一种常用的显示设备，其通过 SPI 总线与微控制器通讯。在实验中，我配置了 PB3、PB5 管脚为 SPI0 总线模式，并根据 OLED 通讯协议编写了驱动程序，实现了在 OLED 屏上显示字符串、汉字和图片的功能。通过使用 OLED 取模工具，我将所需的字符和图片转换为 16 进制格式数组，并存储在 oledfont.h 文件中，以便在 OLED 屏上显示。

4.3 数据一致性验证

在基础实验中，我通过比较写入 EEPROM 的数据与从 EEPROM 读取的数据，验证了数据的一致性。当数据一致时，OLED 屏显示“=”，并点亮 LED1；当数据不一致时，显示“≠”，并点亮 LED2。这一过程不仅验证了 EEPROM 的存储功能，也展示了 OLED 屏在显示逻辑判断结果方面的应用。

4.4 图片显示与动态更新

在提高实验中，我成功地在 OLED 屏上显示了自制的 bmp 格式图片，并实现了图片的动态更新。通过定义图片数组和索引，我能够在每三秒切换显示不同的图片，这一实验步骤展示了 OLED 屏在动态显示方面的应用。

4.5 随机数据匹配与显示

在拓展实验中，我实现了将二字词语写入 EEPROM，并随机读取词语进行匹配显示的功能。当匹配成功时，显示“Bingo”并清除对应的词语；当匹配失败时，显示“False”并继续匹配。这一过程不仅展示了 EEPROM 的数据存储功能，也体现了 OLED 屏在显示逻辑结果和动态内容方面的应用。

5 实验总结

通过本次实验，我不仅掌握了 I2C 总线通讯的基本原理和使用方法，还学会了如何使用 OLED 显示屏进行数据的显示。实验中，我成功实现了数据的存储、读取、显示和逻辑判断，以及图片的动态显示和随机数据匹配。这些实验步骤不仅加深了我对嵌入式系统开发的理解，也提高了我解决实际问题的能力。