ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI THỰC HÀNH LAB02

LỚP CE232.N21.1

Giảng viên hướng dẫn: TRẦN HOÀNG LỘC

Sinh viên thực hiện nhóm 7:

Trương Hữu Khang	20520211
Nguyễn Linh Anh Khoa	20520219
Hà Vĩnh Kiện	20520597
Phan Duy Thông	20520789

TP. Hồ Chí Minh, 2023

PHẦN ĐỀ:

1. Tạo một project từ ví dụ esp-idf/peripherals/i2c/i2c_self_test:

- Chỉnh sửa lại project để giao tiếp với ESP32
- Giải thích các bước cài đặt

2. Sử dụng font, thư viện và các hàm cho sẵn theo tài nguyên đính kèm, viết chương trình hiển thị MSSV của các thành viên trong nhóm lên OLED SSD1306:

- Viết chương trình, upload source code lên github và đính kèm link vào báo cáo.
- Giải thích cách thức hoạt động của source code (không giải thích bằng cách ghi chú code), đặc biệt cần làm rõ cách hoạt động của các hàm hiển thị, các command trên SSD1306
- Demo chương trình bằng cách quay video và upload lên Youtube (chế độ Unlisted), đính kèm link vào báo cáo.

3. Vẽ logo UIT lên OLED SSD1306. Giải thích cách làm và kết quả?

- Viết chương trình, upload source code lên github và đính kèm link vào báo cáo.
- Giải thích cách thức hoạt động của source code (không giải thích bằng cách ghi chú code)
- Demo chương trình bằng cách quay video và upload lên Youtube (chế độ Unlisted), đính kèm link vào báo cáo.

PHẦN BÁO CÁO

```
// Control byte
#define OLED_CONTROL_BYTE_CMD_SINGLE 0x80
#define OLED_CONTROL_BYTE_CMD_STREAM 0x00
#define OLED_CONTROL_BYTE_DATA_STREAM 0x40

// Fundamental commands (pg.28)
#define OLED_CMD_SET_CONTRAST 0x81 // follow with 0x7F
#define OLED_CMD_DISPLAY_RAM 0xA4
#define OLED_CMD_DISPLAY_ALLON 0xA5
#define OLED_CMD_DISPLAY_NORMAL 0xA6
#define OLED_CMD_DISPLAY_INVERTED 0xA7
#define OLED_CMD_DISPLAY_OFF 0xAE
#define OLED_CMD_DISPLAY_ON 0xAF

// Addressing Command Table (pg.30)
#define OLED_CMD_SET_LOWER_COLUMN_START 0x00 // can be used only in PAGE mode
#define OLED_CMD_SET_HIGHER_COLUMN_START 0x10 // can be used only in PAGE mode
#define OLED_CMD_SET_MEMORY_ADDR_MODE 0x20   // follow with 0x00 = HORZ mode = Behave like
a KS108 graphic LCD
#define OLED_CMD_SET_COLUMN_RANGE 0x21      // can be used only in HORZ/VERT mode - follow
with 0x00 and 0x7F = COL127
#define OLED_CMD_SET_PAGE_RANGE 0x22       // can be used only in HORZ/VERT mode - follow
with 0x00 and 0x07 = PAGE7
#define OLED_CMD_SET_PAGE_START_ADDRESS 0xB0 // can be used only in PAGE mode - follow
with 0x00 and 0x07 = PAGE7

// Hardware Config (pg.31)
#define OLED_CMD_SET_DISPLAY_START_LINE 0x40
#define OLED_CMD_SET_SEGMENT_REMAP 0xA1
#define OLED_CMD_SET_MUX_RATIO 0xA8 // follow with 0x3F = 64 MUX
#define OLED_CMD_SET_COM_SCAN_MODE 0xC8
#define OLED_CMD_SET_DISPLAY_OFFSET 0xD3 // follow with 0x00
#define OLED_CMD_SET_COM_PIN_MAP 0xDA   // follow with 0x12
#define OLED_CMD_NOP 0xE3           // NOP

// Timing and Driving Scheme (pg.32)
#define OLED_CMD_SET_DISPLAY_CLK_DIV 0xD5 // follow with 0x80
#define OLED_CMD_SET_PRECHARGE 0xD9     // follow with 0xF1
#define OLED_CMD_SET_VCOMH_DESELCT 0xDB  // follow with 0x30

// Charge Pump (pg.62)
#define OLED_CMD_SET_CHARGE_PUMP 0x8D // follow with 0x14
```

Giải thích các command của OLED:

- OLED_CONTROL_BYTE_CMD_SINGLE: một byte tiếp theo được gửi đến OLED sẽ là byte command.
- OLED_CONTROL_BYTE_CMD_STREAM: một chuỗi byte lệnh sẽ được gửi đến OLED.

- OLED_CONTROL_BYTE_DATA_STREAM: một chuỗi byte dữ liệu sẽ được gửi đến OLED.
- OLED_CMD_SET_CONTRAST: lệnh dùng để setting độ tương phản cho OLED đi kèm với lệnh này sẽ gồm một byte có giá trị từ 0-255 để cài đặt tương phản cho màn hình.
- OLED_CMD_DISPLAY_RAM: màn hình sẽ hiển thị dữ liệu được lưu trong RAM. Sử dụng lệnh này để có thể ghi nội dung cần hiển thị vào RAM của SSD1306.
- OLED_CMD_DISPLAY_ALLON: bật sáng tất cả các pixel.
- OLED_CMD_DISPLAY_NORMAL: chế độ hiển thị bình thường, khi bit được ghi là giá trị 1 thì pixel tại vị trí đó sẽ sáng lên còn 0 thì ngược lại.
- OLED_CMD_DISPLAY_INVERTED: nội dung hiển thị trên màn hình sẽ bị đảo ngược.
- OLED_CMD_DISPLAY_OFF: tắt màn hình
- OLED_CMD_DISPLAY_ON: bật màn hình.
- OLED_CMD_SET_LOWER_COLUMN_START: lệnh dùng để cài đặt vị trí cột (bit thấp) – SSD1306 sử dụng 8 bit để access vào một vị trí của cột.
- OLED_CMD_SET_HIGHER_COLUMN_START: lệnh dùng để cài đặt vị trí cột (bit cao)
- OLED_CMD_SET_MEMMORY_ADDR_MODE: cài đặt chế độ địa chỉ cho nội dung trong RAM. 0x00: pixel trên màn hình sẽ được map theo chiều ngang từ trái sang phải, trên xuống dưới. 0x01: pixel trên màn hình sẽ được map theo chiều dọc từ trên xuống dưới, từ trái sang phải. 0x02: ở chế độ này bộ nhớ của SSD1306 sẽ được chia ra thành các page, mỗi page sẽ được map vào pixel được hiển thị, từ trái sang phải, từ trên xuống dưới.
- OLED_CMD_SET_COLUMN_RANGE: dùng để set vị trí bắt đầu cột và kết thúc cột của bộ nhớ hiển thị dữ liệu. Lệnh này đặt một con trỏ vào vị trí bắt đầu cột trong bộ nhớ. Con trỏ này sẽ tự động tăng lên khi đọc dữ liệu.
- OLED_CMD_SET_PAGE_RANGE: tương tự

- OLED_CMD_SET_COLUMN_RANGE như dữ liệu được tổ chức ở dạng page.
- OLED_CMD_SET_PAGE_START_ADDRESS: đặt vị trí bắt đầu cho page để ghi dữ liệu hiển thị vào.
- OLED_CMD_SET_DISPLAY_START_LINE: lệnh này set giá trị cho thanh ghi Display Start Line để xác định vị trí của RAM, với giá trị từ 0 đến 63. Với giá trị 0,cột 0 của RAM sẽ được map với COM0, với giá trị là 1 thì cột 1 của RAM sẽ được map vs COM0.
- OLED_CMD_SEGMENT_REMAP: lệnh này đảo ngược mapping mặc định của OLED theo chiều ngang (từ phải sang trái) để chữ được hiển thị đúng hướng.
- OLED_CMD_SET_MUX_RATIO: lệnh này thay đổi 63 kênh đa thức mặc định sang một tỉ lệ khác, từ 16 đến 63. Output pad từ COM0-COM63 sẽ được chọn dựa vào tín hiệu COM.
- OLED_CMD_SET_COM_SCAN_MODE: lệnh này đảo ngược mapping mặc định theo chiều dọc của OLED (từ dưới lên)
- OLED_CMD_SET_DISPLAY_OFFSET: lệnh này cài đặt map của display start line vào COM0 đến COM63.
- OLED_CMD_SET_COM_PIN_MAP: lệnh này cài đặt tín hiệu từ COM để phù hợp với kích thước của màn hình.
- OLED_CMD_NOP: không làm gì cả.
- OLED_CMD_SET_DISPLAY_CLK_DIV: dùng để điều chỉnh bộ dao động và bộ chia xung.
- OLED_CMD_SET_PRECHARGE: lệnh cài đặt độ dài của giai đoạn pre-charge.
- OLED_CMD_SET_VCOMH_DESELECT: lệnh này điều chỉnh output của bộ ổn áp V_{COMH}.
- OLED_CMD_SET_CHARGE_PUMP: lệnh này để sử dụng điện áp từ charge pump (charge pump là một linh kiện dùng để chuyển đổi điện áp thấp hơn sang điện áp cao hơn). OLED cần điện áp từ 12V-15V để hoạt động, hiện tại các MCU đều hoạt động ở điện áp 3.3V-5V. Việc này đảm bảo việc OLED hoạt động bình thường.

```
#define _I2C_NUMBER(num) I2C_NUM_##num
#define I2C_NUMBER(num) _I2C_NUMBER(num)

#define DATA_LENGTH 512              /*!< Data buffer length of test buffer */
#define RW_TEST_LENGTH 128           /*!< Data length for r/w test, [0,DATA_LENGTH] */
#define DELAY_TIME_BETWEEN_ITEMS_MS 1000 /*!< delay time between different test items */

#define I2C_MASTER_SCL_IO 4       /*!< gpio number for I2C master clock */
#define I2C_MASTER_SDA_IO 5       /*!< gpio number for I2C master data  */
#define I2C_MASTER_NUM I2C_NUMBER(0) /*!< I2C port number for master dev */
#define I2C_MASTER_FREQ_HZ 100000   /*!< I2C master clock frequency */
#define I2C_MASTER_TX_BUF_DISABLE 0  /*!< I2C master doesn't need buffer */
#define I2C_MASTER_RX_BUF_DISABLE 0  /*!< I2C master doesn't need buffer */

#define WRITE_BIT I2C_MASTER_WRITE /*!< I2C master write */
#define READ_BIT I2C_MASTER_READ   /*!< I2C master read */
#define ACK_CHECK_EN 0x1        /*!< I2C master will check ack from slave*/
#define ACK_CHECK_DIS 0x0       /*!< I2C master will not check ack from slave */
#define ACK_VAL 0x0            /*!< I2C ack value */
#define NACK_VAL 0x1           /*!< I2C nack value */
```

Define các giá trị cần sử dụng

```
static esp_err_t i2c_master_init(void)
{
   int i2c_master_port = I2C_MASTER_NUM;
   i2c_config_t conf = {
      .mode = I2C_MODE_MASTER,
      .sda_io_num = I2C_MASTER_SDA_IO,
      .sda_pullup_en = GPIO_PULLUP_ENABLE,
      .scl_io_num = I2C_MASTER_SCL_IO,
      .scl_pullup_en = GPIO_PULLUP_ENABLE,
      .master.clk_speed = I2C_MASTER_FREQ_HZ,
      // .clk_flags = 0,       /*!< Optional, you can use I2C_SCLK_SRC_FLAG_* flags to
choose i2c source clock here. */
   };
   esp_err_t err = i2c_param_config(i2c_master_port, &conf);
   if (err != ESP_OK)
   {
      return err;
   }
   return i2c_driver_install(i2c_master_port, conf.mode, I2C_MASTER_RX_BUF_DISABLE,
I2C_MASTER_TX_BUF_DISABLE, 0);
}
```

Hàm khởi tạo I2C:

- Mode: đặt mode cho I2C là Master
- Pin SDA theo define là pin số 5

- Pin SCL theo define là pin số 4
- Setting tần số cho I2C
- Sau khi đã có những config cho I2C tiến hành nạp config cho I2C bằng lệnh i2c_param_config vào I2C0

```
void ssd1306_init()
{
   esp_err_t espRc;

   i2c_cmd_handle_t cmd = i2c_cmd_link_create();

   i2c_master_start(cmd);
   i2c_master_write_byte(cmd, (OLED_I2C_ADDRESS << 1) | I2C_MASTER_WRITE, true);
   i2c_master_write_byte(cmd, OLED_CONTROL_BYTE_CMD_STREAM, true);

   i2c_master_write_byte(cmd, OLED_CMD_SET_CHARGE_PUMP, true);
   i2c_master_write_byte(cmd, 0x14, true);

   i2c_master_write_byte(cmd, OLED_CMD_SET_SEGMENT_REMAP, true); // reverse left-right
mapping
   i2c_master_write_byte(cmd, OLED_CMD_SET_COM_SCAN_MODE, true); // reverse up-bottom
mapping

   i2c_master_write_byte(cmd, OLED_CMD_DISPLAY_NORMAL, true);
   i2c_master_write_byte(cmd, OLED_CMD_DISPLAY_OFF, true);
   i2c_master_write_byte(cmd, OLED_CMD_DISPLAY_ON, true);
   i2c_master_stop(cmd);

   espRc = i2c_master_cmd_begin(I2C_NUM_0, cmd, 10 / portTICK_PERIOD_MS);
   if (espRc == ESP_OK)
   {
      ESP_LOGI(TAG, "OLED configured successfully");
   }
   else
   {
      ESP_LOGE(TAG, "OLED configuration failed. code: 0x%.2X", espRc);
   }
   i2c_cmd_link_delete(cmd);
}
```

Hàm khởi tạo cho I2C

Đầu tiên ta tiến hành gửi lệnh ghi vào địa chỉ của OLED, sau đó sử dụng command byte cmd stream để tiến hành gửi một chuỗi các lệnh. Lệnh Charge Pump để đảm bảo cho OLED hoạt động bình thường, Segment remap và set com scan mode để map lại vị trí cho các pixel theo từ trái sang phải, từ trên xuống dưới. Display normal đặt màn hình vào chế độ

normal, khi bit được ghi là 1 thì pixel tại vị trí đó sẽ sáng, sau đó kiểm tra việc tắt mở màn hình và kết thúc quá trình khởi tạo.

```
void ssd1306_display_text(const void *arg_text)
{
   i2c_cmd_handle_t cmd;
   char *text = (char *)arg_text;
   uint8_t text_len = strlen(text);

   uint8_t cur_page = 0;

   cmd = i2c_cmd_link_create();
   i2c_master_start(cmd);
   i2c_master_write_byte(cmd, (OLED_I2C_ADDRESS << 1) | I2C_MASTER_WRITE, true);

   i2c_master_write_byte(cmd, OLED_CONTROL_BYTE_CMD_STREAM, true);
   i2c_master_write_byte(cmd, OLED_CMD_SET_LOWER_COLUMN_START | 0, true);
   i2c_master_write_byte(cmd, OLED_CMD_SET_HIGHER_COLUMN_START | 0, true);
   i2c_master_write_byte(cmd, OLED_CMD_SET_PAGE_START_ADDRESS | cur_page, true);
   i2c_master_stop(cmd);
   i2c_master_cmd_begin(I2C_NUM_0, cmd, 10 / portTICK_PERIOD_MS);
   i2c_cmd_link_delete(cmd);

   for (uint8_t i = 0; i < text_len; i++)
   {
      if (text[i] == '\n')
      {
         cmd = i2c_cmd_link_create();
         i2c_master_start(cmd);
         i2c_master_write_byte(cmd, (OLED_I2C_ADDRESS << 1) | I2C_MASTER_WRITE, true);

         i2c_master_write_byte(cmd, OLED_CONTROL_BYTE_CMD_STREAM, true);
         i2c_master_write_byte(cmd, OLED_CMD_SET_LOWER_COLUMN_START | 0, true);
         i2c_master_write_byte(cmd, OLED_CMD_SET_HIGHER_COLUMN_START | 0, true);
         i2c_master_write_byte(cmd, OLED_CMD_SET_PAGE_START_ADDRESS | ++cur_page, true);

         i2c_master_stop(cmd);
         i2c_master_cmd_begin(I2C_NUM_0, cmd, 10 / portTICK_PERIOD_MS);
         i2c_cmd_link_delete(cmd);
      }
      else
      {
         cmd = i2c_cmd_link_create();
         i2c_master_start(cmd);
         i2c_master_write_byte(cmd, (OLED_I2C_ADDRESS << 1) | I2C_MASTER_WRITE, true);

         i2c_master_write_byte(cmd, OLED_CONTROL_BYTE_DATA_STREAM, true);
         i2c_master_write(cmd, font8x8_basic_tr[(uint8_t)text[i]], 8, true);

         i2c_master_stop(cmd);
         i2c_master_cmd_begin(I2C_NUM_0, cmd, 10 / portTICK_PERIOD_MS);
         i2c_cmd_link_delete(cmd);
      }
   }
}
```

Hàm dùng để in chữ ra màn hình

Đầu tiên master sẽ tiến hành gửi lệnh write đến oled. Sau đó gửi một chuỗi các command để reset vị trí con trỏ về vị trí (0,0). Sau đó tiến hành gửi giá trị của từng kí tự trong chuỗi string, nếu kí tự là \n thì tiến hành tăng page lên (mỗi page có thể coi là một line mới) và tiến hành reset vị trí con trỏ trở về đầu line.

```
void ssd1306_display_clear()
{
   i2c_cmd_handle_t cmd;

   uint8_t clear[128];
   memset(clear, 0, sizeof(clear));
   for (uint8_t i = 0; i < 8; i++)
   {
      cmd = i2c_cmd_link_create();
      i2c_master_start(cmd);
      i2c_master_write_byte(cmd, (OLED_I2C_ADDRESS << 1) | I2C_MASTER_WRITE, true);
      i2c_master_write_byte(cmd, OLED_CONTROL_BYTE_CMD_SINGLE, true);
      i2c_master_write_byte(cmd, OLED_CMD_SET_PAGE_START_ADDRESS | i, true);

      i2c_master_write_byte(cmd, OLED_CONTROL_BYTE_DATA_STREAM, true);
      i2c_master_write(cmd, clear, 128, true);
      i2c_master_stop(cmd);
      esp_err_t ret = i2c_master_cmd_begin(I2C_NUM_0, cmd, 10 / portTICK_PERIOD_MS);
      if (ret != ESP_OK)
      {
         ESP_LOGE(TAG, "Error: i2c_master_cmd_begin() failed, ret=%d", ret);
      }
      i2c_cmd_link_delete(cmd);
   }
}
```

Hàm dùng để xóa màn hình

Đầu tiên tạo một mảng có 128 giá trị (chiều ngang của OLED là 128 pixel), sau đó fill mảng với giá trị là 0. Sau đó tiến hành lặp để xóa 8 page trên màn hình (mỗi page là một line). Ở mỗi vòng lặp, đầu tiên ta tiến hành gửi lệnh write đến OLED, sau đó tiến hành chuyển con trỏ về vị trí của line cần được xóa, sau đó tiền hành gửi 128 byte dữ liệu với giá trị 0x00 đến OLED tiến hành lần lượt như vậy cho đến khi hết 8 page.

Source code: https://github.com/otaros/CE232_Lab/tree/main/Lab2

Video minh họa: https://youtu.be/AGcRYDBFVa0