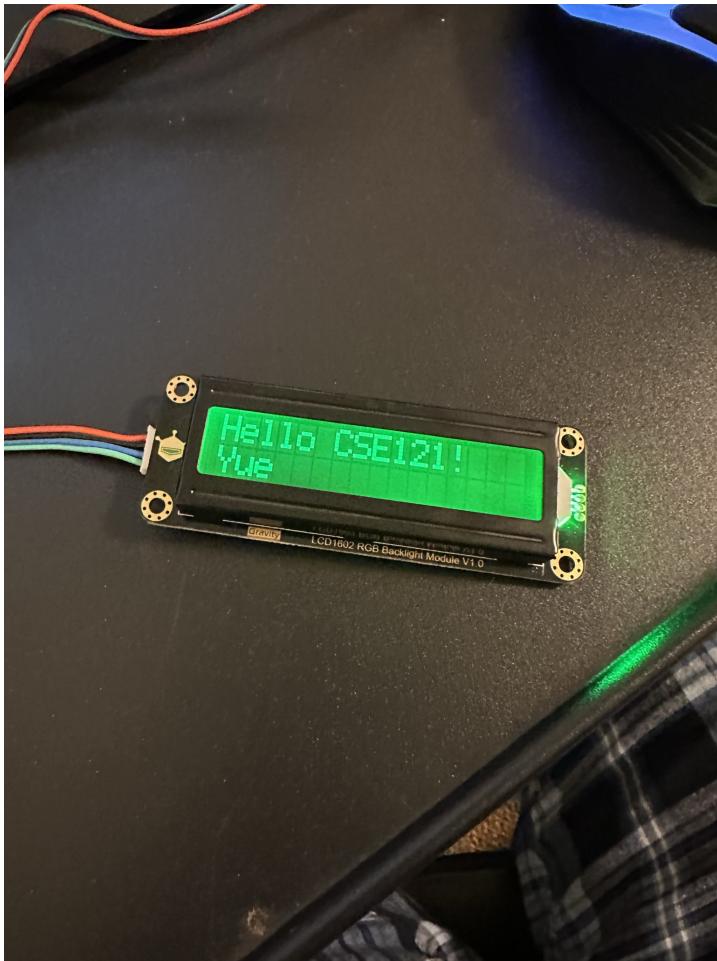


Lab 3

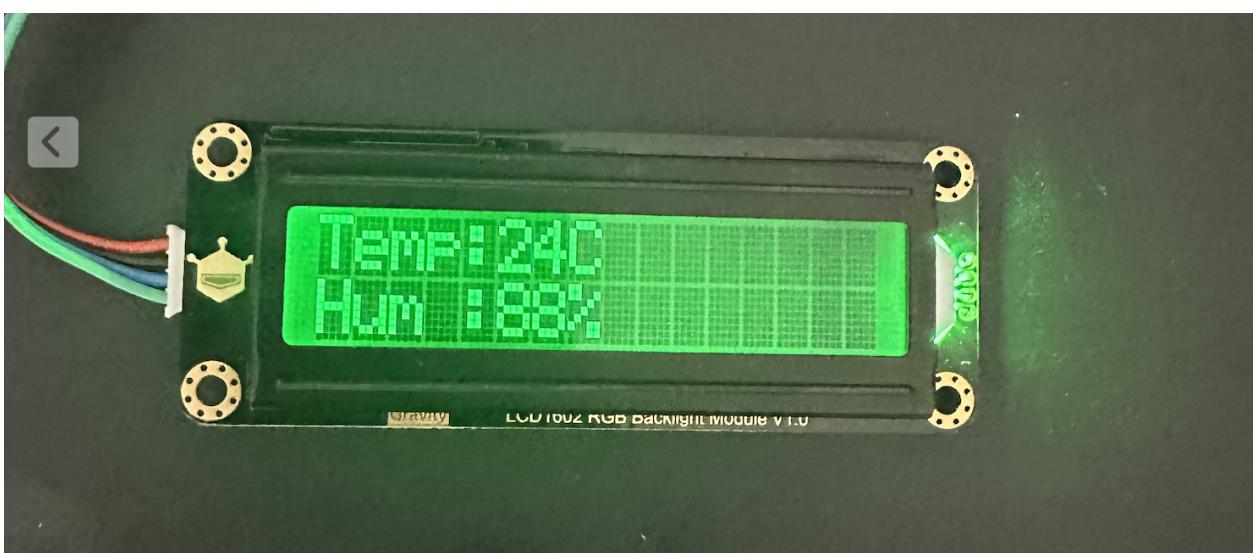
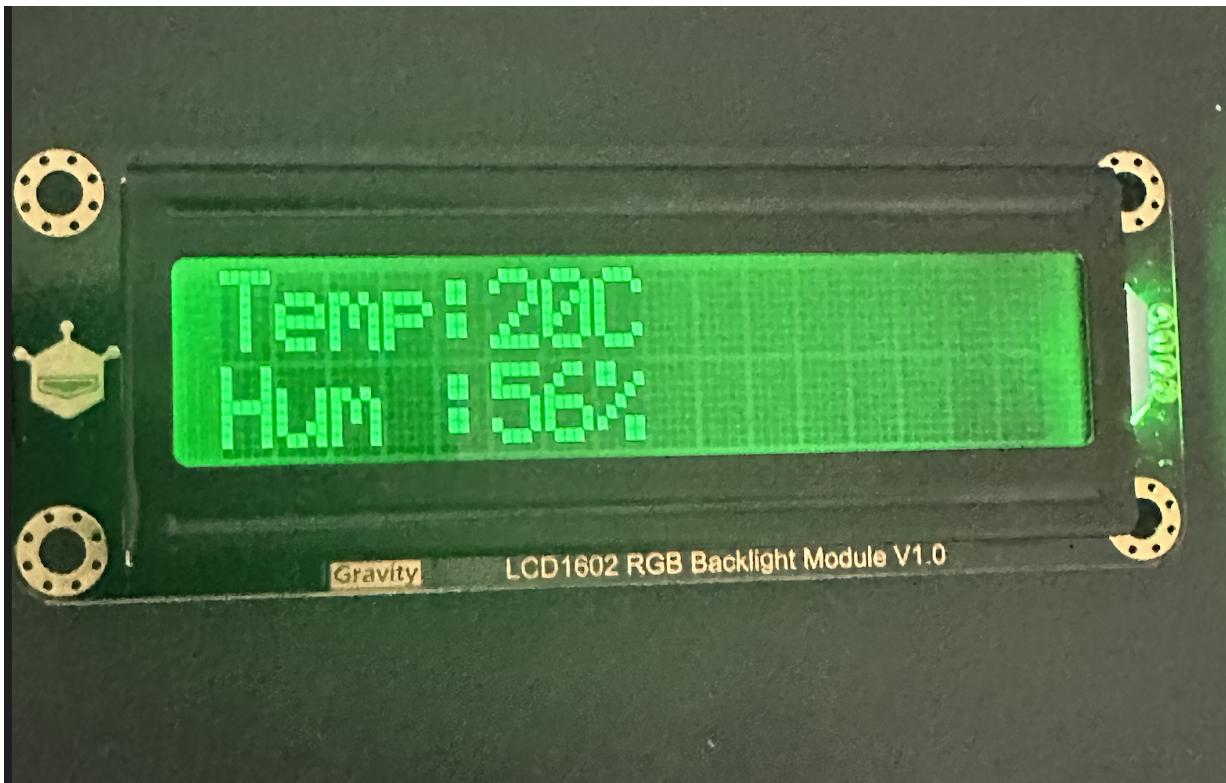
Part2:

Had to import the DFRobot_LCD library and heavily modify its contents in order to print two lines onto the LCD board. The file I had to code with was in C++ so I had to remove a bunch of classes that didn't work, including the print and wire classes, the latter in which I had to replace with i2c commands.



Part 3:

Basically the same thing, but used the code from lab 2 part 3 to print onto the board instead. It's pretty straightforward.



Connection to the esp32c3:



Resources:

ChatGPT

chat.openai.com

<https://docs.espressif.com/projects/esp-matter/en/latest/esp32/faq.html>

APACHE license:

<https://drive.google.com/drive/u/1/folders/15uqJ8zO42qNE-cszcmcdxg0Ajq6x94aC>

DFRobot_LCD library:

https://github.com/DFRobotD/DFRobot_LCD

Other:

https://wiki.dfrobot.com/Gravity_I2C_16x2_Arduino_LCD_with_RGB_Backlight_Display_SKU_DFR0464

<https://www.bareconductive.com/blogs/resources/how-to-solder-headers-onto-the-touch-board>

ChatGPT results:

```
cpp Copy code

#include <Wire.h>
#include "DFRobot_LCD.h" // Assuming you have a custom LCD library

// Define your I2C configuration
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,
};

void DFRobot_LCD::init()
{
    // Initialize I2C with your configuration
    i2c_param_config(I2C_MASTER_NUM, &conf);
    i2c_driver_install(I2C_MASTER_NUM, I2C_MODE_MASTER, 0, 0, 0);

    // Initialize your LCD module (replace with actual LCD initialization code)
    // For example:
    lcd_init(); // This function should be provided by your LCD library
    lcd_begin(_cols, _rows);

    // Set your desired display options
    _showfunction = LCD_4BITMODE | LCD_1LINE | LCD_5x8DOTS;
}
```

cpp

 Copy code

```
#include <Wire.h>
#include "DFRobot_LCD.h" // Assuming you have a custom LCD library

void initializeLCDWithI2C() {
    // Define your I2C configuration
    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,
    };

    // Initialize I2C with your configuration
    i2c_param_config(I2C_MASTER_NUM, &conf);
    i2c_driver_install(I2C_MASTER_NUM, I2C_MODE_MASTER, 0, 0, 0);

    // Initialize your LCD module (replace with actual LCD initialization code)
    // For example:
    lcd_init(); // This function should be provided by your LCD library
    lcd_begin(_cols, _rows);

    // Set your desired display options
    _showfunction = LCD_4BITMODE | LCD_1LINE | LCD_5x8DOTS;
}

void setup() {
    Wire.begin();
    initializeLCDWithI2C();
}

void loop() {
    // Your main loop code here
}
```

cpp

 Copy code

```
void DFRobot_LCD::setReg(uint8_t addr, uint8_t data)
{
    // Start the I2C transmission to the RGB device with address _RGBAddr
    i2c_cmd_handle_t cmd = i2c_cmd_link_create();
    i2c_master_start(cmd);
    i2c_master_write_byte(cmd, (_RGBAddr << 1) | I2C_MASTER_WRITE, true);

    // Write the register address and data to the RGB device
    i2c_master_write_byte(cmd, addr, true);
    i2c_master_write_byte(cmd, data, true);

    // End the I2C transmission
    i2c_master_stop(cmd);
    i2c_master_cmd_begin(I2C_MASTER_NUM, cmd, 1000 / portTICK_RATE_MS);
    i2c_cmd_link_delete(cmd);
}
```

cpp

 Copy code

```
void DFRobot_LCD::send(uint8_t *data, uint8_t len)
{
    // Start the I2C transmission to the LCD device with address _lcdAddr
    i2c_cmd_handle_t cmd = i2c_cmd_link_create();
    i2c_master_start(cmd);
    i2c_master_write_byte(cmd, (_lcdAddr << 1) | I2C_MASTER_WRITE, true);

    // Write the data to the LCD
    for (int i = 0; i < len; i++) {
        i2c_master_write_byte(cmd, data[i], true);
        delay(5);
    }

    // End the I2C transmission
    i2c_master_stop(cmd);
    i2c_master_cmd_begin(I2C_MASTER_NUM, cmd, 1000 / portTICK_RATE_MS);
    i2c_cmd_link_delete(cmd);
}
```

cpp

 Copy code

```
#include <Arduino.h> // Include Arduino.h if using Arduino framework
#include <stdio.h>
#include <string.h>
#include <inttypes.h>

#include "DFRobot_LCD.h"
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"

const uint8_t color_define[4][3] =
{
    {255, 255, 255},           // white
    {255, 0, 0},               // red
    {0, 255, 0},               // green
    {0, 0, 255},               // blue
};

DFRobot_LCD::DFRobot_LCD(uint8_t lcd_cols,uint8_t lcd_rows,uint8_t lcd_Addr,
{
    _lcdAddr = lcd_Addr;
    _RGBAddr = RGB_Addr;
    _cols = lcd_cols;
    _rows = lcd_rows;
}

void DFRobot_LCD::init()
{
    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,
    };
    i2c_param_config(I2C_MASTER_NUM, &conf);
    i2c_driver_install(I2C_MASTER_NUM, I2C_MODE_MASTER, 0, 0, 0);
    _showfunction = LCD_4BITMODE | LCD_1LINE | LCD_5x8DOTS;
    begin(_cols, _rows);
}

// ...

void DFRobot_LCD::delayMicroseconds(uint32_t us)
{
    vTaskDelay(pdMS_TO_TICKS(us / 1000)); // Convert microseconds to millise
}

void DFRobot_LCD::delay(uint32_t ms)
{
    vTaskDelay(pdMS_TO_TICKS(ms)); // Use FreeRTOS task delay
}
```

c

 Copy code

```
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "DFRobot_LCD.h" // Include your LCD library

extern "C" void app_main() {
    DFRobot_LCD lcd(16, 2, 0x27, RGB_ADDRESS); // Adjust LCD settings according to your board
    lcd.init();
    lcd.setRGB(0, 255, 0);

    while (true) {
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.printstr("Hello CSE121!");

        lcd.setCursor(0, 1);
        lcd.printstr("Siero");

        vTaskDelay(pdMS_TO_TICKS(1000)); // Delay for 1 second (adjust as needed)
    }
}
```

c

 Copy code

```
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/i2c.h" // Include the I2C driver library
#include "DFRobot_LCD.h" // Include your LCD library
#include <string.h>
#include <cstdint>
#include <stdio.h>

extern "C" void app_main() {
    DFRobot_LCD lcd(16, 2, 0x27, RGB_ADDRESS); // Adjust LCD settings according to your setup
    lcd.init();
    lcd.setRGB(0, 255, 0);

    while (true) {
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.printstr("Hello CSE121!");

        lcd.setCursor(0, 1);
        lcd.printstr("Siero");

        vTaskDelay(pdMS_TO_TICKS(1000)); // Delay for 1 second (adjust as needed)
    }
}
```

