SƠ ĐỒ NỐI DÂY ESP32-S3 - TRẠM KIỂM SOÁT LŨ

Phiên bản: v4.1 - OLED 0.96" I2C + ESP-IDF v5.5.1

Âm thanh: DFPlayer Mini + Buzzer 5V nhỏ
Nguồn: USB Type-C
Màn hình: OLED 0.96" I2C (128x64)
Trạng thái: ✅ Đã tối ưu

---

1. KẾT NỐI MODULE LORA RA-02 (SPI)

Chân LoRa	ESP32-S3	Chức năng
VCC	3V3	Nguồn 3.3V
GND	GND	Mass
NSS	GPIO 10	FSPICS0
MOSI	GPIO 11	FSPID
MISO	GPIO 13	FSPIQ
SCK	GPIO 12	FSPICLK
RST	GPIO 14	Reset + pull-up 4.7kΩ
DIO0	GPIO 2	Interrupt

Sơ đồ RST:

```
3V3 ---|4.7kΩ|--- RST --- GPIO 14
```

---

2. OLED 0.96" I2C (128x64)

Chân OLED	ESP32-S3	Ghi chú
VCC	3V3	Nguồn 3.3V
GND	GND	Mass
SDA	GPIO 8	I2C Data
SCL	GPIO 9	I2C Clock

⚠️ **Lưu ý:**

- Địa chỉ I2C thường: **0x3C** (hoặc 0x3D)
- Độ phân giải: 128x64 pixel
- Driver: SSD1306

---

3. LED CẢNH BÁO

LED	GPIO	Màu	Cảnh báo
LED 1	GPIO 15	Xanh	Thấp
LED 2	GPIO 16	Vàng	TB
LED 3	GPIO 4	Đỏ	Cao

Mỗi LED qua điện trở **220Ω** xuống GND.

---

4. DFPLAYER MINI

```
DFPlayer        ESP32-S3
──────────────────────────────────────
VCC ──────────────────── 5V
RX ───────────────────── GPIO 17 (U1TXD)
TX ───────────────────── GPIO 18 (U1RXD)
SPK+ ──────────────────── Loa (+)
SPK- ──────────────────── Loa (-)
GND ──────────────────── GND

[Thẻ SD FAT32: mp3/001.mp3, 002.mp3, 003.mp3]
```

---

5. BUZZER 5V

```
GPIO 5 ---|--- [Buzzer 5V] ---|--- GND
```

⚠️ Active Buzzer 5V, dòng < 30mA

---

6. BẢNG TỔNG HỢP GPIO

GPIO	Chức năng
2	LoRa DIO0
4	LED Đỏ + 220Ω
5	Buzzer
8	**OLED SDA**
9	**OLED SCL**
10	LoRa NSS
11	LoRa MOSI
12	LoRa SCK
13	LoRa MISO
14	LoRa RST
15	LED Xanh + 220Ω
16	LED Vàng + 220Ω
17	DFPlayer TX
18	DFPlayer RX

---

7. CODE ESP-IDF v5.5.1

CMakeLists.txt (project root)

cmake

```
cmake_minimum_required(VERSION 3.16)
include($ENV{IDF_PATH}/tools/cmake/project.cmake)
project(tram_kiem_soat)
```

main/CMakeLists.txt

cmake

```
idf_component_register(
    SRCS "main.c" "oled_ssd1306.c" "dfplayer.c"
    INCLUDE_DIRS "."
)
```

main/oled_ssd1306.h

c

```c
#ifndef OLED_SSD1306_H
#define OLED_SSD1306_H

#include "driver/i2c_master.h"
#include "esp_err.h"

#define OLED_ADDR          0x3C
#define I2C_MASTER_SDA_IO   8
#define I2C_MASTER_SCL_IO   9
#define I2C_MASTER_FREQ_HZ  400000

#define OLED_WIDTH         128
#define OLED_HEIGHT        64

esp_err_t oled_init(void);
void oled_clear(void);
void oled_set_cursor(uint8_t x, uint8_t y);
void oled_print(const char *str);
void oled_display(void);
void oled_deinit(void);

#endif
```

main/oled_ssd1306.c

c

```c
#include "oled_ssd1306.h"
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "esp_log.h"
#include <string.h>

static const char *TAG = "OLED";
static i2c_master_bus_handle_t bus_handle = NULL;
static i2c_master_dev_handle_t dev_handle = NULL;

static uint8_t oled_buffer[OLED_WIDTH * OLED_HEIGHT / 8];
static uint8_t cursor_x = 0, cursor_y = 0;

// Font 6x8 cơ bản
static const uint8_t font_6x8[][6] = {
    {0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, // Space
    {0x00, 0x00, 0x5F, 0x00, 0x00, 0x00}, // !
    // ... (thêm các ký tự khác)
};

static esp_err_t oled_send_cmd(uint8_t cmd) {
    uint8_t data[2] = {0x00, cmd}; // 0x00 = command
    return i2c_master_transmit(dev_handle, data, 2, 1000);
}

static esp_err_t oled_send_data(uint8_t *data, size_t len) {
    uint8_t buffer[len + 1];
    buffer[0] = 0x40; // 0x40 = data
    memcpy(&buffer[1], data, len);
    return i2c_master_transmit(dev_handle, buffer, len + 1, 1000);
}

esp_err_t oled_init(void) {
    // Cấu hình I2C Master Bus
    i2c_master_bus_config_t bus_config = {
        .i2c_port = I2C_NUM_0,
        .sda_io_num = I2C_MASTER_SDA_IO,
        .scl_io_num = I2C_MASTER_SCL_IO,
        .clk_source = I2C_CLK_SRC_DEFAULT,
        .glitch_ignore_cnt = 7,
        .flags.enable_internal_pullup = true,
```

```
};
ESP_ERROR_CHECK(i2c_new_master_bus(&bus_config, &bus_handle));

// Thêm thiết bị OLED
i2c_device_config_t dev_config = {
    .dev_addr_length = I2C_ADDR_BIT_LEN_7,
    .device_address = OLED_ADDR,
    .scl_speed_hz = I2C_MASTER_FREQ_HZ,
};
ESP_ERROR_CHECK(i2c_master_bus_add_device(bus_handle, &dev_config, &dev_handle));

// Khởi tạo OLED SSD1306
vTaskDelay(pdMS_TO_TICKS(100));
oled_send_cmd(0xAE); // Display OFF
oled_send_cmd(0xD5); // Set clock divide
oled_send_cmd(0x80);
oled_send_cmd(0xA8); // Set multiplex
oled_send_cmd(0x3F);
oled_send_cmd(0xD3); // Set display offset
oled_send_cmd(0x00);
oled_send_cmd(0x40); // Set start line
oled_send_cmd(0x8D); // Charge pump
oled_send_cmd(0x14);
oled_send_cmd(0x20); // Memory mode
oled_send_cmd(0x00); // Horizontal
oled_send_cmd(0xA1); // Segment remap
oled_send_cmd(0xC8); // COM scan direction
oled_send_cmd(0xDA); // COM pins
oled_send_cmd(0x12);
oled_send_cmd(0x81); // Contrast
oled_send_cmd(0xCF);
oled_send_cmd(0xD9); // Pre-charge
oled_send_cmd(0xF1);
oled_send_cmd(0xDB); // VCOMH
oled_send_cmd(0x40);
oled_send_cmd(0xA4); // Display all on resume
oled_send_cmd(0xA6); // Normal display
oled_send_cmd(0xAF); // Display ON

oled_clear();
ESP_LOGI(TAG, "OLED initialized");
```

```
    return ESP_OK;
}

void oled_clear(void) {
    memset(oled_buffer, 0, sizeof(oled_buffer));
    cursor_x = 0;
    cursor_y = 0;
}

void oled_set_cursor(uint8_t x, uint8_t y) {
    cursor_x = x;
    cursor_y = y;
}

void oled_print(const char *str) {
    while (*str) {
        // Vẽ ký tự vào buffer (simplified)
        // TODO: Implement full font rendering
        str++;
    }
}

void oled_display(void) {
    oled_send_cmd(0x21); // Column address
    oled_send_cmd(0x00);
    oled_send_cmd(0x7F);
    oled_send_cmd(0x22); // Page address
    oled_send_cmd(0x00);
    oled_send_cmd(0x07);
    oled_send_data(oled_buffer, sizeof(oled_buffer));
}

void oled_deinit(void) {
    if (dev_handle) {
        i2c_master_bus_rm_device(dev_handle);
    }
    if (bus_handle) {
        i2c_del_master_bus(bus_handle);
    }
}
```

main/dfplayer.h

c

```c
#ifndef DFPLAYER_H
#define DFPLAYER_H

#include <stdint.h>
#include "esp_err.h"

#define DFPLAYER_TXD  17
#define DFPLAYER_RXD  18
#define UART_NUM      UART_NUM_1

esp_err_t dfplayer_init(void);
void dfplayer_set_volume(uint8_t volume);
void dfplayer_play(uint8_t track);
void dfplayer_deinit(void);

#endif
```

main/main.c

c

```c
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/gpio.h"
#include "esp_log.h"
#include "oled_ssd1306.h"
#include "dfplayer.h"

static const char *TAG = "MAIN";

// Định nghĩa GPIO
#define LED_LOW     15
#define LED_MED     16
#define LED_HIGH    4
#define BUZZER_PIN  5

// Hàm khởi tạo GPIO
static void gpio_init_all(void) {
    gpio_config_t io_conf = {
        .pin_bit_mask = (1ULL << LED_LOW) | (1ULL << LED_MED) |
                        (1ULL << LED_HIGH) | (1ULL << BUZZER_PIN),
        .mode = GPIO_MODE_OUTPUT,
        .pull_up_en = GPIO_PULLUP_DISABLE,
        .pull_down_en = GPIO_PULLDOWN_DISABLE,
        .intr_type = GPIO_INTR_DISABLE,
    };
    gpio_config(&io_conf);

    gpio_set_level(LED_LOW, 0);
    gpio_set_level(LED_MED, 0);
    gpio_set_level(LED_HIGH, 0);
    gpio_set_level(BUZZER_PIN, 0);
}

static void beep(int duration_ms) {
    gpio_set_level(BUZZER_PIN, 1);
    vTaskDelay(pdMS_TO_TICKS(duration_ms));
    gpio_set_level(BUZZER_PIN, 0);
}

static void handle_water_level(float level) {
```

```
    gpio_set_level(LED_LOW, 0);
    gpio_set_level(LED_MED, 0);
    gpio_set_level(LED_HIGH, 0);

    oled_clear();
    oled_set_cursor(0, 0);
    oled_print("Muc nuoc:");
    oled_set_cursor(0, 2);
    char buffer[32];
    snprintf(buffer, sizeof(buffer), "%.1f cm", level);
    oled_print(buffer);
    oled_display();

    if (level < 50) {
        ESP_LOGI(TAG, "✅ An toan");
        gpio_set_level(LED_LOW, 1);
    } else if (level >= 50 && level < 100) {
        ESP_LOGW(TAG, "⚠ Canh bao muc 1");
        gpio_set_level(LED_MED, 1);
        dfplayer_play(1);
    } else if (level >= 100 && level < 150) {
        ESP_LOGW(TAG, "⚠⚠ Canh bao muc 2");
        gpio_set_level(LED_MED, 1);
        gpio_set_level(LED_HIGH, 1);
        dfplayer_play(2);
        beep(100);
        vTaskDelay(pdMS_TO_TICKS(100));
        beep(100);
    } else {
        ESP_LOGE(TAG, "🚨 Khan cap!");
        gpio_set_level(LED_HIGH, 1);
        dfplayer_play(3);
        for (int i = 0; i < 3; i++) {
            beep(100);
            vTaskDelay(pdMS_TO_TICKS(50));
        }
    }
}

void app_main(void) {
    ESP_LOGI(TAG, "Khoi dong he thong...");
```

```
gpio_init_all();

// Test LED + Buzzer
gpio_set_level(LED_LOW, 1);
beep(100);
vTaskDelay(pdMS_TO_TICKS(200));
gpio_set_level(LED_LOW, 0);

gpio_set_level(LED_MED, 1);
beep(100);
vTaskDelay(pdMS_TO_TICKS(200));
gpio_set_level(LED_MED, 0);

gpio_set_level(LED_HIGH, 1);
beep(100);
vTaskDelay(pdMS_TO_TICKS(200));
gpio_set_level(LED_HIGH, 0);

// Khởi tạo OLED
ESP_ERROR_CHECK(oled_init());
oled_clear();
oled_set_cursor(0, 0);
oled_print("Tram Kiem Soat");
oled_set_cursor(0, 2);
oled_print("Khoi dong...");
oled_display();

// Khởi tạo DFPlayer
ESP_ERROR_CHECK(dfplayer_init());
dfplayer_set_volume(25);

// TODO: Khởi tạo LoRa ở đây

oled_clear();
oled_set_cursor(0, 2);
oled_print("San sang!");
oled_display();
beep(100);
vTaskDelay(pdMS_TO_TICKS(100));
beep(100);
```

```
    ESP_LOGI(TAG, "He thong san sang!");

    // Main loop - Test
    while (1) {
        float test_levels[] = {30, 70, 120, 180};
        for (int i = 0; i < 4; i++) {
            beep(50);
            handle_water_level(test_levels[i]);
            vTaskDelay(pdMS_TO_TICKS(5000));
        }
    }
}
```

---

8. HƯỚNG DẪN BUILD

bash

```bash
# Cài đặt ESP-IDF v5.5.1
cd ~/esp
git clone -b v5.5.1 --recursive https://github.com/espressif/esp-idf.git esp-idf-v5.5.1
cd esp-idf-v5.5.1
./install.sh esp32s3

# Activate
. ./export.sh

# Build project
cd your_project_folder
idf.py set-target esp32s3
idf.py build
idf.py -p /dev/ttyUSB0 flash monitor
```

---

9. CHECKLIST

- [ ] Tất cả kết nối GND
- [ ] LoRa: GPIO 10-14, 2
- [ ] **OLED: GPIO 8 (SDA), 9 (SCL)**

- ☐ DFPlayer: GPIO 17 (TX), 18 (RX)
- ☐ LED: GPIO 15, 16, 4 + điện trở 220Ω
- ☐ Buzzer: GPIO 5
- ☐ Thẻ SD FAT32 có mp3/001.mp3, 002.mp3, 003.mp3
- ☐ Pull-up 4.7kΩ cho LoRa RST
- ☐ **Kiểm tra địa chỉ I2C OLED (0x3C hoặc 0x3D)**

✅ HOÀN TẤT

Sơ đồ đã tối ưu với:

- ✅ LoRa SPI: GPIO 10-13 (FSPI hardware)
- ✅ **OLED 0.96" I2C: GPIO 8 (SDA), 9 (SCL)**
- ✅ DFPlayer UART: GPIO 17, 18 (UART1)
- ✅ LED 3 màu: GPIO 15, 16, 4
- ✅ Buzzer: GPIO 5
- ✅ Code ESP-IDF v5.5.1 cho OLED SSD1306

⚠️ **Lưu ý quan trọng:**

- OLED cần nguồn **3.3V** (không phải 5V như LCD)
- Địa chỉ I2C mặc định: **0x3C** (dùng I2C scanner để kiểm tra)
- Code OLED cần thư viện font đầy đủ (simplified trong ví dụ)

Tài liệu phiên bản: v4.1
Ngày cập nhật: 2025
Nền tảng: ESP32-S3 + ESP-IDF v5.5.1