

TRƯỜNG ĐẠI HỌC KHOA HỌC  
KHOA CÔNG NGHỆ THÔNG TIN

ĐƯƠNG VĂN TUẤN

21T1020801

XÂY DỰNG ĐỒNG HỒ THỜI GIAN THỰC VỚI  
ESP 32 VÀ CẢM BIẾN MQ

PHÁT TRIỂN ỨNG DỤNG IoT – TIN4024  
GIẢNG VIÊN HƯỚNG DẪN: VÕ VIỆT DŨNG

HUẾ, THÁNG 4 NĂM 2025

## **DANH MỤC CÁC TỪ VIẾT TẮT**

- \* ESP32: Embedded Serial Processor 32-bit – Ví điều khiển có WiFi/Bluetooth
- \* NTP: Network Time Protocol – Giao thức đồng bộ thời gian qua mạng
- \* OLED: Organic Light Emitting Diode – Màn hình hiển thị
- \* RTC: Real Time Clock – Đồng hồ thời gian thực

## MỤC LỤC

<b>MỞ ĐẦU.....</b>	1
<b>NỘI DUNG .....</b>	2
I. Giới thiệu khái quát về ESP32, giao thức NTP và màn hình Oled .....	2
1. Khái niệm.....	2
2. Nguyên lý hoạt động của NTP.....	3
3. Phần cứng sử dụng.....	3
II. Mã nguồn và sơ đồ chương trình xây dựng đồng hồ thời gian thực với ESP32 và NTP .....	5
1. Phần mềm và nền tảng lập trình: .....	5
2. Cấu trúc chương trình: .....	5
3. Lưu đồ thuật toán .....	5
4. Mã nguồn của dự án.....	6
5. Sơ đồ Wokwi và kết nối .....	8
6. Gửi thông tin bằng Telegram.....	8
<b>Phần Kết Luận .....</b>	10

## MỞ ĐẦU

Trong thời đại hiện nay, công nghệ ngày càng phát triển mạnh mẽ và một trong những yếu tố quan trọng để các thiết bị điện tử hoạt động chính xác và hiệu quả là việc đồng bộ thời gian. Thời gian chính xác không chỉ giúp các thiết bị hoạt động đúng đắn mà còn hỗ trợ trong nhiều ứng dụng quan trọng, chẳng hạn như trong các hệ thống cảm biến, mạng máy tính hay các hệ thống điều khiển tự động. Một trong những phương pháp phổ biến để đồng bộ thời gian giữa các thiết bị là sử dụng **NTP (Network Time Protocol)**, một giao thức cho phép đồng bộ hóa thời gian giữa các thiết bị thông qua mạng Internet. NTP giúp các thiết bị có thể lấy thời gian từ các máy chủ trung gian, từ đó đảm bảo thời gian của các thiết bị luôn chính xác và không bị lệch. Nhờ vào việc sử dụng NTP, các hệ thống có thể duy trì đồng hồ hệ thống chính xác mà không cần phụ thuộc vào nguồn thời gian cục bộ. Trong bài tiểu luận này, em sẽ giới thiệu về cách sử dụng **ESP32**, một vi điều khiển mạnh mẽ với khả năng kết nối Wi-Fi, để đồng bộ thời gian từ máy chủ NTP và hiển thị thời gian này trên màn hình **OLED**. **ESP32** là một vi điều khiển phổ biến trong các ứng dụng IoT (Internet of Things) nhờ vào tính linh hoạt và khả năng kết nối Wi-Fi ổn định, giúp cho việc đồng bộ thời gian qua NTP trở nên đơn giản và dễ dàng. Ngoài ra, em cũng sẽ sử dụng môi trường mô phỏng **Wokwi** để thử nghiệm và kiểm tra các đoạn mã của mình. Wokwi là một nền tảng trực tuyến cho phép mô phỏng các dự án sử dụng vi điều khiển mà không cần phải có phần cứng thực tế. Điều này giúp tiết kiệm thời gian và chi phí trong quá trình phát triển dự án. Bài tiểu luận của em vẫn còn nhiều sót, em mong nhận được sự góp ý của thầy cô. Em xin cảm ơn ạ.

## NỘI DUNG

### I. Giới thiệu khái quát về ESP32, giao thức NTP và màn hình Oled

#### 1. Khái niệm

- ESP32 là một bộ vi điều khiển thuộc danh mục vi điều khiển trên chip công suất thấp và tiết kiệm chi phí. Hầu hết tất cả các biến thể ESP32 đều tích hợp Bluetooth và Wi-Fi chế độ kép, làm cho nó có tính linh hoạt cao, mạnh mẽ và đáng tin cậy cho nhiều ứng dụng.

Nó là sự kế thừa của vi điều khiển NodeMCU ESP8266 phổ biến và cung cấp hiệu suất và tính năng tốt hơn. Bộ vi điều khiển ESP32 được sản xuất bởi Espressif Systems và được sử dụng rộng rãi trong nhiều ứng dụng khác nhau như IoT, robot và tự động hóa.

ESP32 cũng được thiết kế để tiêu thụ điện năng thấp, lý tưởng cho các ứng dụng chạy bằng pin. Nó có hệ thống quản lý năng lượng cho phép nó hoạt động ở chế độ ngủ và chỉ thức dậy khi cần thiết, điều này có thể kéo dài tuổi thọ pin rất nhiều.

- Một số ưu điểm:

- + Kết nối Wi-Fi tích hợp, giúp truy cập internet dễ dàng để đồng bộ NTP.
- + Hỗ trợ nhiều giao thức giao tiếp như I2C, SPI, UART.
- + Có thể kết nối với màn hình OLED, cảm biến, relay...
- + Năng lượng tiêu thụ thấp, phù hợp với các thiết bị hoạt động liên tục.

Ưu điểm của module ESP-WROOM-32 là PCB có các cạnh đúc. Nhờ đó các nhà sản xuất bên thứ ba có thể lấy module ESP-WROOM-32 và thiết kế một bo break-out cho module này.

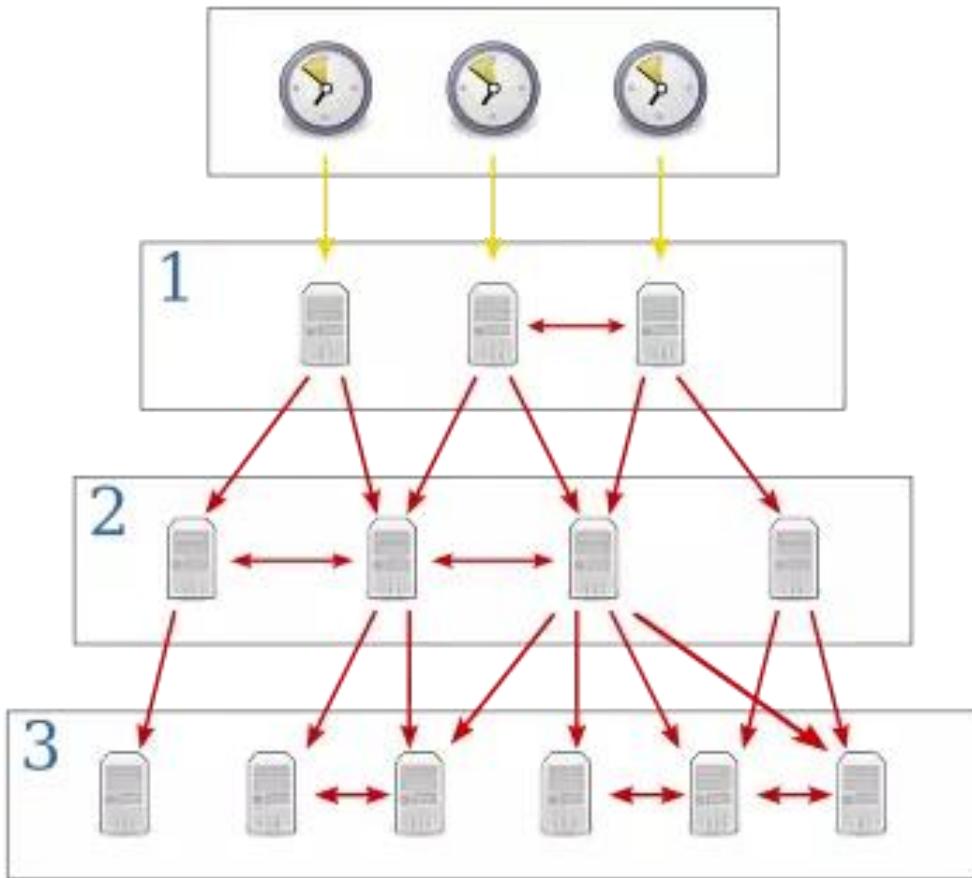
Một trong những bo như vậy là ESP32 DevKit Board. Nó chứa ESP-WROOM-32 làm module chính và một số phần cứng bổ sung để dễ dàng lập trình ESP32 và tạo kết nối với các chân GPIO.

- Giao thức NTP (Network Time Protocol) là một giao thức để đồng bộ đồng hồ của các hệ thống máy tính thông qua mạng dữ liệu chuyển mạch gói với độ trễ biến đổi.

+ Giao thức này được thiết kế để tránh ảnh hưởng của độ trễ biến đổi bằng cách sử dụng bộ đệm jitter. NTP cũng là tên gọi của phần mềm được triển khai trong dự án Dịch vụ NTP Công cộng (NTP Public Services Project).

+ Trên mạng Internet, NTP đồng bộ đồng hồ của các hệ thống máy tính theo UTC; trong môi trường LAN độc lập, NTP cũng thường được sử dụng để đồng bộ với UTC, nhưng về nguyên tắc nó có thể được sử dụng để đồng bộ với một mốc thời gian khác, ví dụ như múi giờ tại chỗ.

## 2. Nguyên lý hoạt động của NTP



- NTP client gửi một gói tin, trong đó chứa một thẻ thời gian tới cho NTP server.
- NTP server nhận được gói tin, gửi trả lại NTP client một gói tin khác, có thẻ thời gian là thời điểm nó gửi gói tin đó đi.
- NTP client nhận được gói tin đó, tính toán độ trễ, dựa và thẻ thời gian mà nó nhận được cùng với độ trễ đường truyền, NTP client sẽ set lại thời gian của nó.

Cách hoạt động tóm tắt:

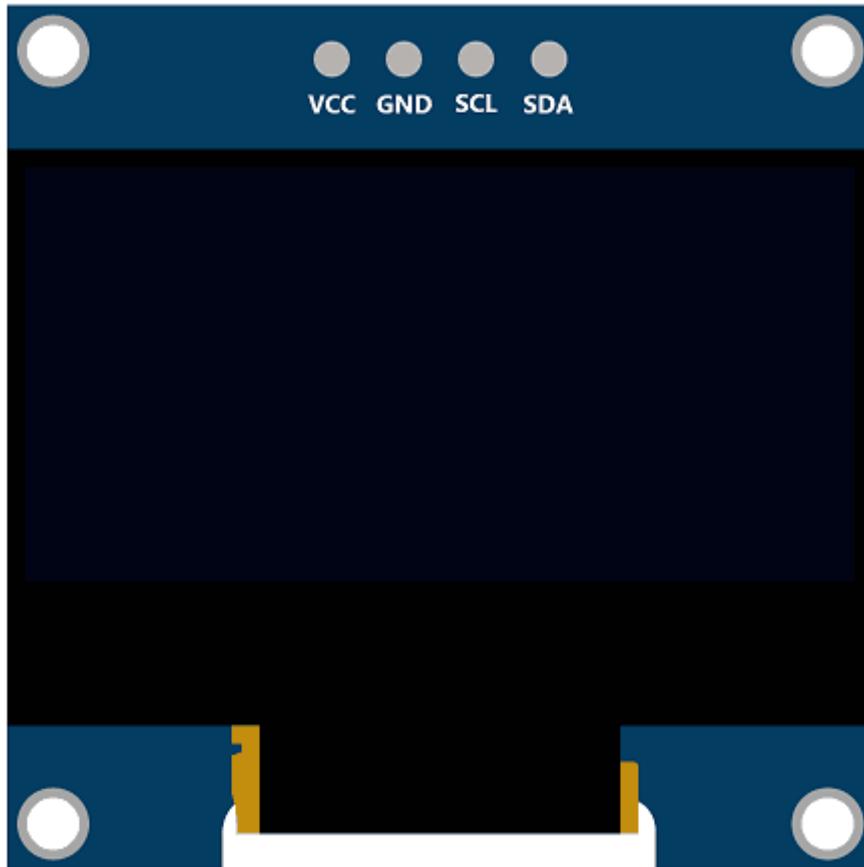
- + ESP32 kết nối Wi-Fi và gửi yêu cầu đến máy chủ NTP (ví dụ: pool.ntp.org).
- + Nhận thời gian chuẩn UTC.
- + Chuyển đổi sang múi giờ địa phương nếu cần (ví dụ: GMT+7 cho Việt Nam).
- + Cập nhật đồng hồ nội bộ của ESP32.

### 3. Phần cứng sử dụng

- Bo mạch: ESP32 DevKit: ESP32 DevKit V1 là một bo mạch phát triển mạnh mẽ và linh hoạt, lý tưởng cho các dự án IoT và điện tử. Với kích thước nhỏ gọn và khả năng tích hợp nhiều tính năng, nó là sự lựa chọn phổ biến cho cả người mới bắt đầu và các chuyên gia.
- + Tính năng nổi bật:
  1. Kết nối linh hoạt: ESP32 hỗ trợ cả Wi-Fi và Bluetooth, cho phép xây dựng các ứng dụng IoT đa dạng.
  2. Hiệu suất cao: Với CPU Dual-core 240MHz, ESP32 mang lại hiệu suất xử lý tốt cho các tác vụ phức tạp.
  3. Tiêu thụ điện năng thấp: Có chế độ ngủ sâu giúp tiết kiệm điện năng trong các ứng dụng không cần kết nối liên tục.
  4. Dễ dàng lập trình: Hỗ trợ nhiều môi trường lập trình như Arduino IDE, PlatformIO,

và ESP-IDF, giúp lập trình viên dễ dàng tiếp cận và phát triển.

- Màn hình OLED SSD1306 : Màn hình OLED được điều khiển bởi IC trình điều khiển SSD1306. SSD1306 là trình điều khiển CMOS OLED với bộ điều khiển cho hệ thống hiển thị đồ họa ma trận điểm OLED. Do sử dụng trình điều khiển SSD1306, số lượng linh kiện bên ngoài cần thiết và mức tiêu thụ điện năng đã giảm.
- + Chân màn hình OLED (giao diện I2C)



SDA (Dữ liệu nối tiếp): SDA được sử dụng để truyền dữ liệu giữa master và slave. Dữ liệu và xác nhận được gửi qua SDA.

SCL (Đồng hồ nối tiếp): Đó là một tín hiệu đồng hồ. Chân này truyền đồng hồ đến phụ, SCL. Dữ liệu sẽ được gửi đến các thiết bị khác vào sự kiện tick đồng hồ. Chỉ thiết bị chính mới có quyền kiểm soát dòng SCL này

VCC: Đây là chân cung cấp điện. Cần có nguồn cung cấp + 3.3V. Nguồn cung cấp hơn 3.3 V có thể làm hỏng màn hình.

GND: Đây là Ghim nối đất. Kết nối mặt đất của nguồn cung cấp với chân này.

- Nguồn cấp: USB 5V hoặc pin sạc
- Loa mini / relay / đèn LED nếu cần gửi thông báo định kỳ.
- Dây nối jumper (nếu làm thực tế).

## II. Mã nguồn và sơ đồ chương trình xây dựng đồng hồ thời gian thực với ESP32 và NTP

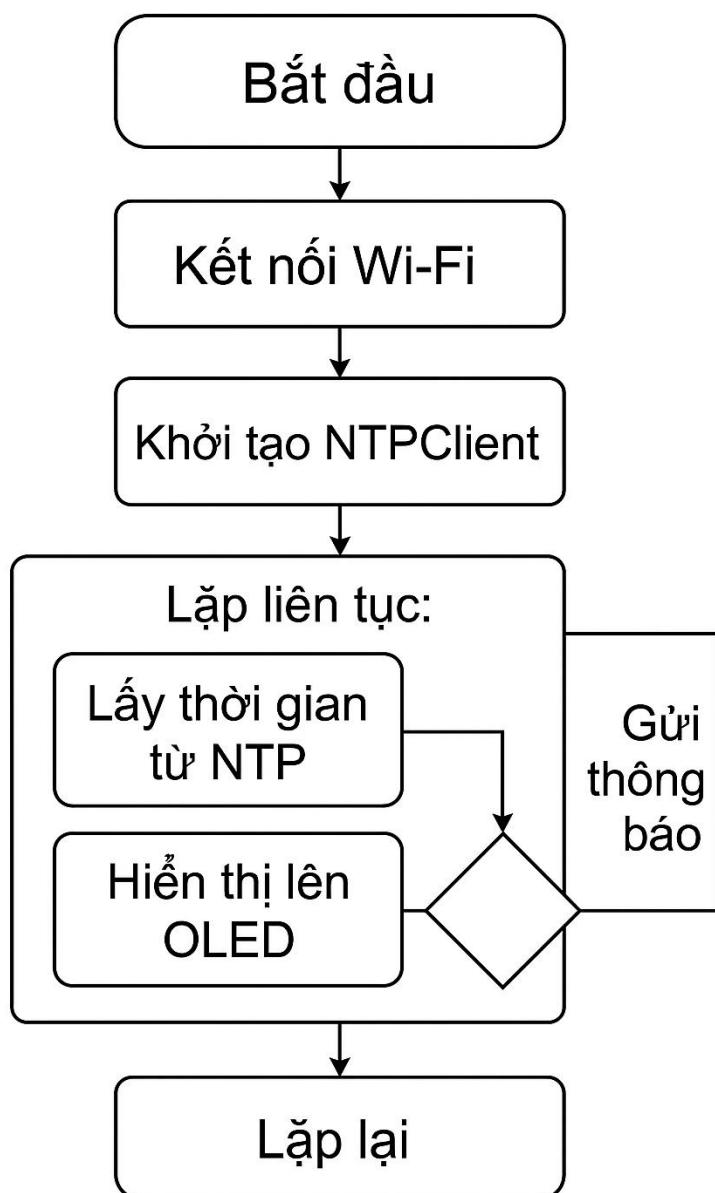
### 1. Phần mềm và nền tảng lập trình:

- Nền tảng lập trình: PlatformIO IDE và Wokwi.
- Thư viện NTP: WiFi.h, WiFiUdp.h, NTPClient.h
- Thư viện OLED: Adafruit\_GFX.h, Adafruit\_SSD1306.h

### 2. Cấu trúc chương trình:

- Kết nối Wi-Fi: nhập tên và mật khẩu mạng.
- Khởi tạo NTP Client: lấy thời gian từ máy chủ.
- Hiển thị thời gian lên OLED.
- Lập lịch gửi thông báo.

### 3. Lưu đồ thuật toán



#### 4. Mã nguồn của dự án

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <WiFiUdp.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire,
OLED_RESET);

// Wi-Fi credentials
const char* ssid    = "Wokwi-GUEST";
const char* password = "";

// Telegram Bot Token và Chat ID
String botToken = "7329594929:AAE6IhGA1gJ6b3gidu3bMJt8nELAP3RgyCY";
String chatId = "6766097161"; // Thay bằng chat ID thật

// NTP cấu hình
const char* ntpServer = "pool.ntp.org";
const long gmtOffset_sec = 7 * 3600;
const int daylightOffset_sec = 0;

bool daGuiThongBao = false;

void guiThongBaoTelegram(const String &message) {
    HTTPClient http;
    String url = "https://api.telegram.org/bot" + botToken + "/sendMessage?chat_id=" +
    chatId + "&text=" + message;
    http.begin(url);
    int httpCode = http.GET();
    if (httpCode > 0) {
        Serial.printf("Telegram response: %s\n", http.getString().c_str());
    } else {
        Serial.printf("Telegram failed, error: %s\n", http.errorToString(httpCode).c_str());
    }
    http.end();
}

void setup() {
    Serial.begin(115200);

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
```

```

delay(500);
Serial.print(".");
}
Serial.println("\nWiFi connected");

configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);

struct tm timeinfo;
while (!getLocalTime(&timeinfo)) {
    Serial.println("Waiting for NTP time...");
    delay(1000);
}

if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;);
}
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(WHITE);
}

void loop() {
    struct tm timeinfo;
    if (!getLocalTime(&timeinfo)) {
        Serial.println("Failed to obtain time");
        return;
    }

    char timeStr[9];
    char dateStr[11];
    strftime(timeStr, sizeof(timeStr), "%H:%M:%S", &timeinfo);
    strftime(dateStr, sizeof(dateStr), "%d/%m/%Y", &timeinfo);

    // OLED hiển thị
    display.clearDisplay();
    display.setCursor(0, 0);
    display.print("Online ");
    display.print(timeStr);
    display.setCursor(0, 16);
    display.print(dateStr);
    display.print(" UTC");
    display.display();

    // Gửi thông báo lúc 07:00:00
    if (strcmp(timeStr, "16:00:00") == 0 && !daGuiThongBao) {
        guiThongBaoTelegram("ESP32: Đã đến 4 giờ chiều!");
        daGuiThongBao = true;
    }
}

```

```

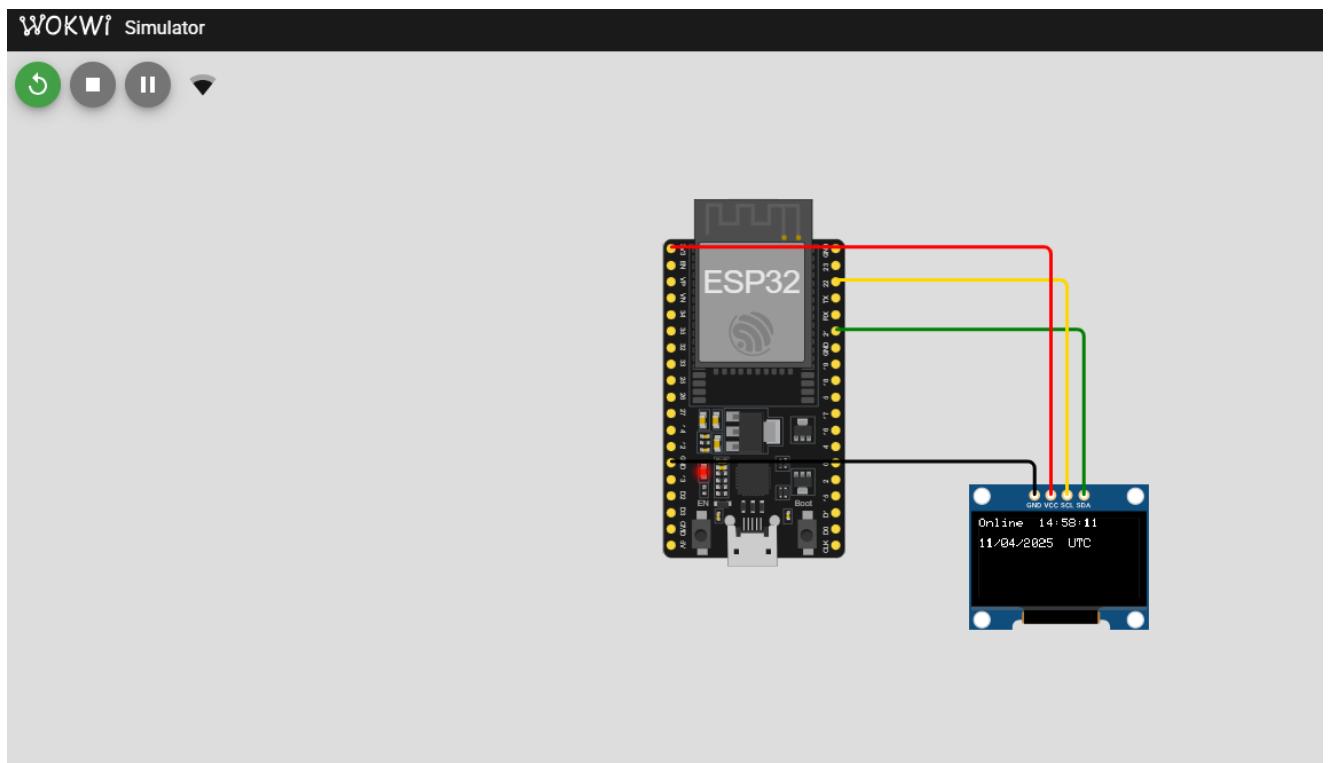
    }

// Reset cờ sau 1 phút
if (strcmp(timeStr, "07:01:00") == 0) {
    daGuiThongBao = false;
}

delay(1000);
}

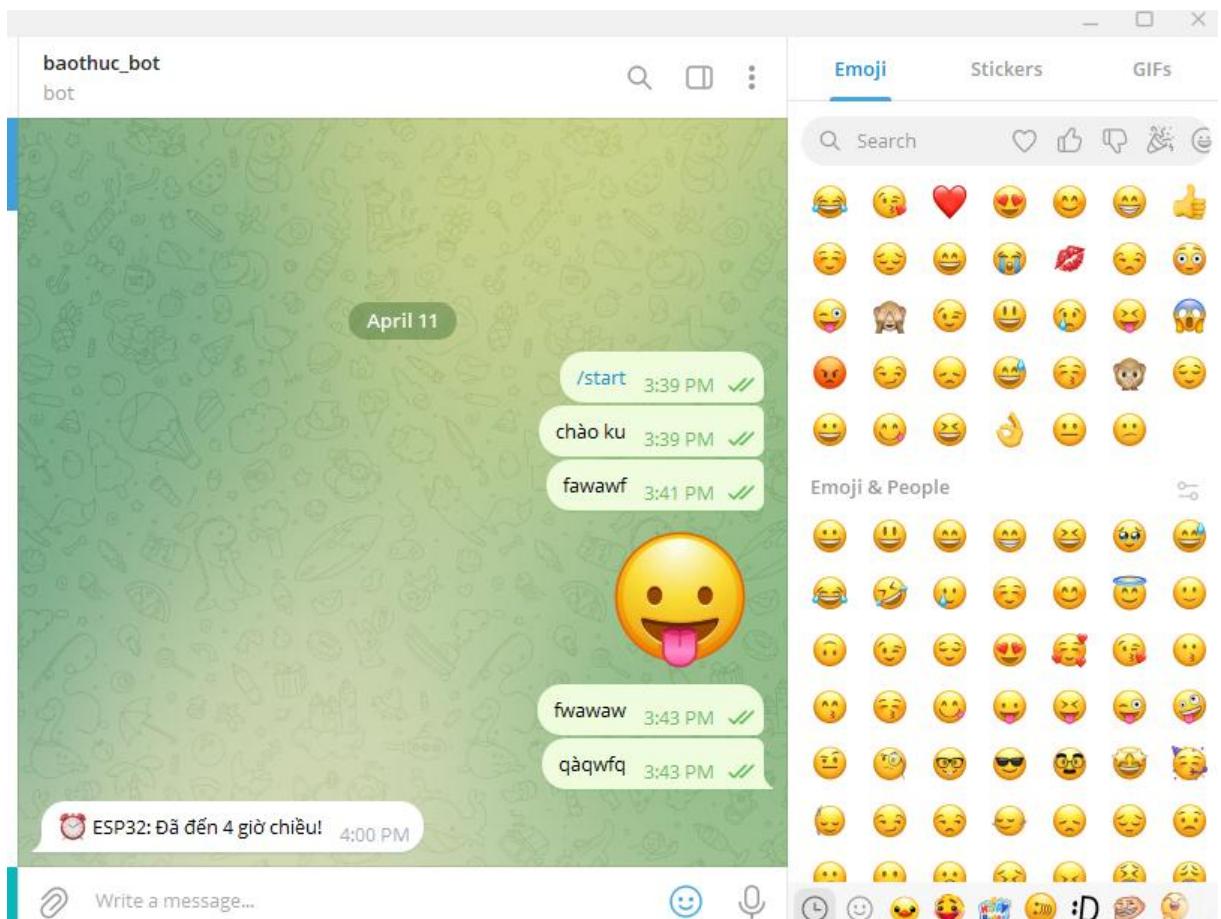
```

## 5. Sơ đồ Wokwi và kết nối



## 6. Gửi thông tin bằng Telegram

- Hình ảnh sau khi gửi thành công thông báo đến giờ báo thức rồi.



## Phần Kết Luận

Dự án xây dựng hệ thống đồng hồ thời gian thực sử dụng ESP32 kết hợp với NTP đã chứng minh được tính khả thi và hiệu quả trong việc đồng bộ thời gian chính xác mà không cần thêm phần cứng như mô-đun RTC. Việc sử dụng giao thức NTP giúp hệ thống luôn đảm bảo giờ giấc chính xác theo múi giờ được cài đặt, phù hợp với các ứng dụng cần định thời hoặc hẹn giờ chính xác.

Qua việc tích hợp **màn hình OLED**, thời gian thực được hiển thị rõ ràng, phục vụ mục đích theo dõi trực quan. Đồng thời, hệ thống cũng đã được mở rộng thêm khả năng **gửi thông báo định kỳ qua Telegram**, giúp nâng cao tính tương tác và khả năng giám sát từ xa cho người dùng mà không cần phụ thuộc vào nền tảng như Blynk.

Tuy nhiên, hệ thống có một số điểm cần lưu ý:

- + Phụ thuộc hoàn toàn vào kết nối Internet: Nếu Wi-Fi không ổn định hoặc mất kết nối, hệ thống sẽ không thể đồng bộ giờ từ NTP hoặc gửi thông báo Telegram.
- + Cần tối ưu việc gửi thông báo tránh gửi trùng lặp hoặc quá nhiều lần trong một khoảng thời gian ngắn, dễ gây phiền hà cho người dùng.
- + Khi triển khai thực tế cần xử lý các tình huống ngoại lệ như: mất Wi-Fi, lỗi khi kết nối đến NTP server, lỗi gửi HTTP...
- Trong tương lai, hệ thống có thể được mở rộng theo các hướng sau:
  - + Tự động hóa: Bật/tắt thiết bị điện theo lịch (ví dụ: tưới cây tự động, bật đèn ban đêm...).
  - + Ứng dụng thực tế: Ứng dụng trong hệ thống cảnh báo, quản lý giờ học/làm việc, báo thức cá nhân...
  - + Tối ưu năng lượng: Kết hợp với chế độ ngủ của ESP32 để tiết kiệm điện năng trong các hệ thống chạy bằng pin.
  - + Bổ sung cơ chế dự phòng: Tích hợp thêm mô-đun RTC như DS3231 để đảm bảo hệ thống vẫn hoạt động chính xác khi mất mạng dài hạn.

## Tài Liệu Tham Khảo

- [1] [Mô-đun cảm biến Màn hình OLED Ssd1306 | Mô-đun cảm biến](#) .Truy cập ngày 4/10/2025.
- [2] [Tổng quan về giao thức Network Time Protocol \(NTP\)](#). Truy cập ngày 4/10/2025.
- [3] [Giới thiệu về bo mạch phát triển ESP32 | Mecsu.vn](#). Truy cập ngày 4/8/2025.