**BÁO CÁO VỀ PHẦN CHẾ ĐỘ DEEP-SLEEP VÀ NGHIÊN CỨU NGUỒN CẤP CHO ESP32-S3-SUPER MINI**

**I. PHẦN DEEP-SLEEP**

1. Lý thuyết: nghiên cứu trên ESP32-S3-SUPER MINI

**Tính năng tiết kiệm điện (Deep Sleep)**

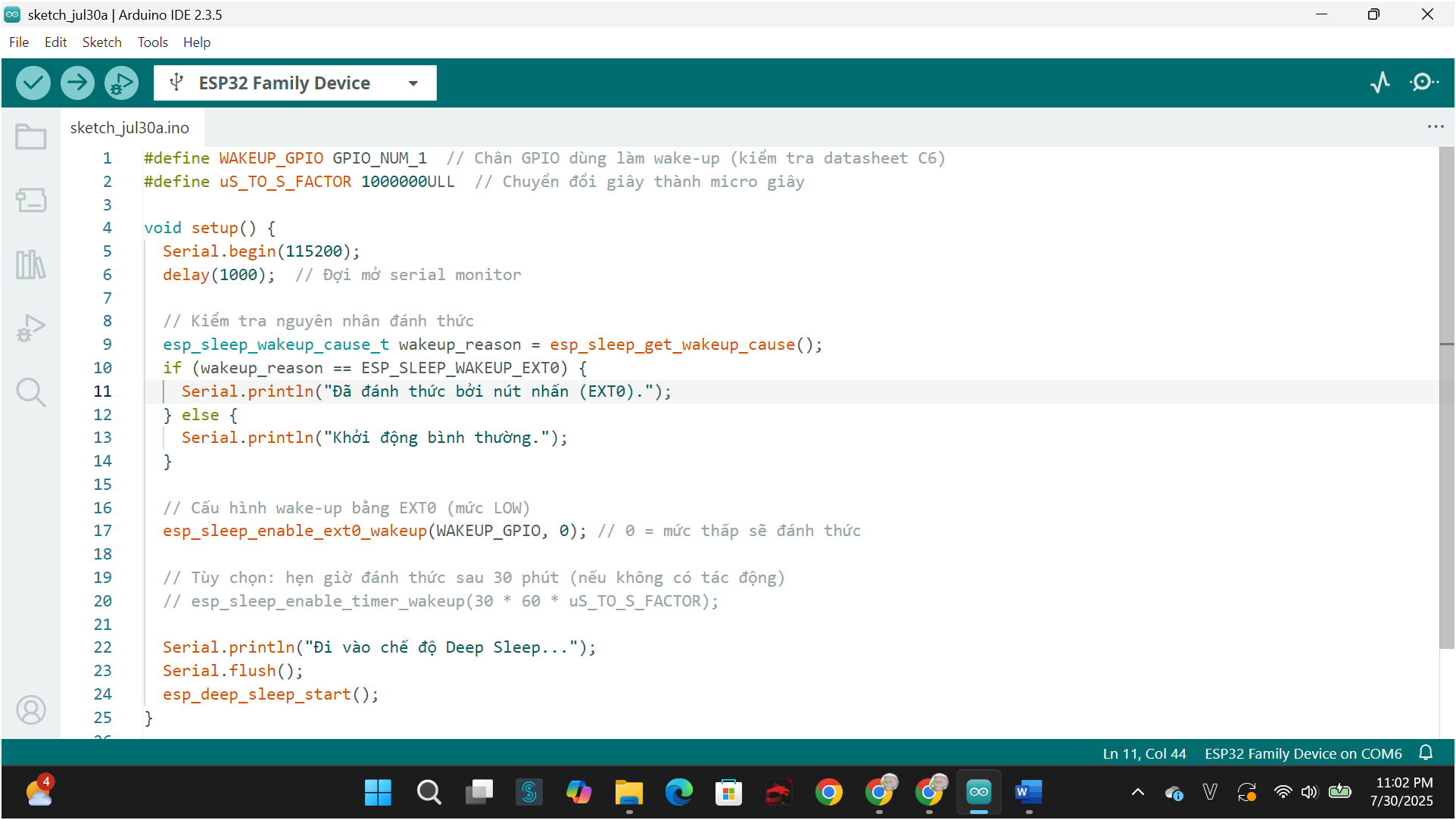
* ESP32-S3 hỗ trợ chế độ Deep Sleep với tiêu hao ~8µA
* Sử dụng GPIO0 (chân RTC GPIO) kết nối nút nhấn để đánh thức vi điều khiển (EXT0 wake-up)
* Kết hợp với RTC Timer wake-up theo chu kỳ 30-60 phút

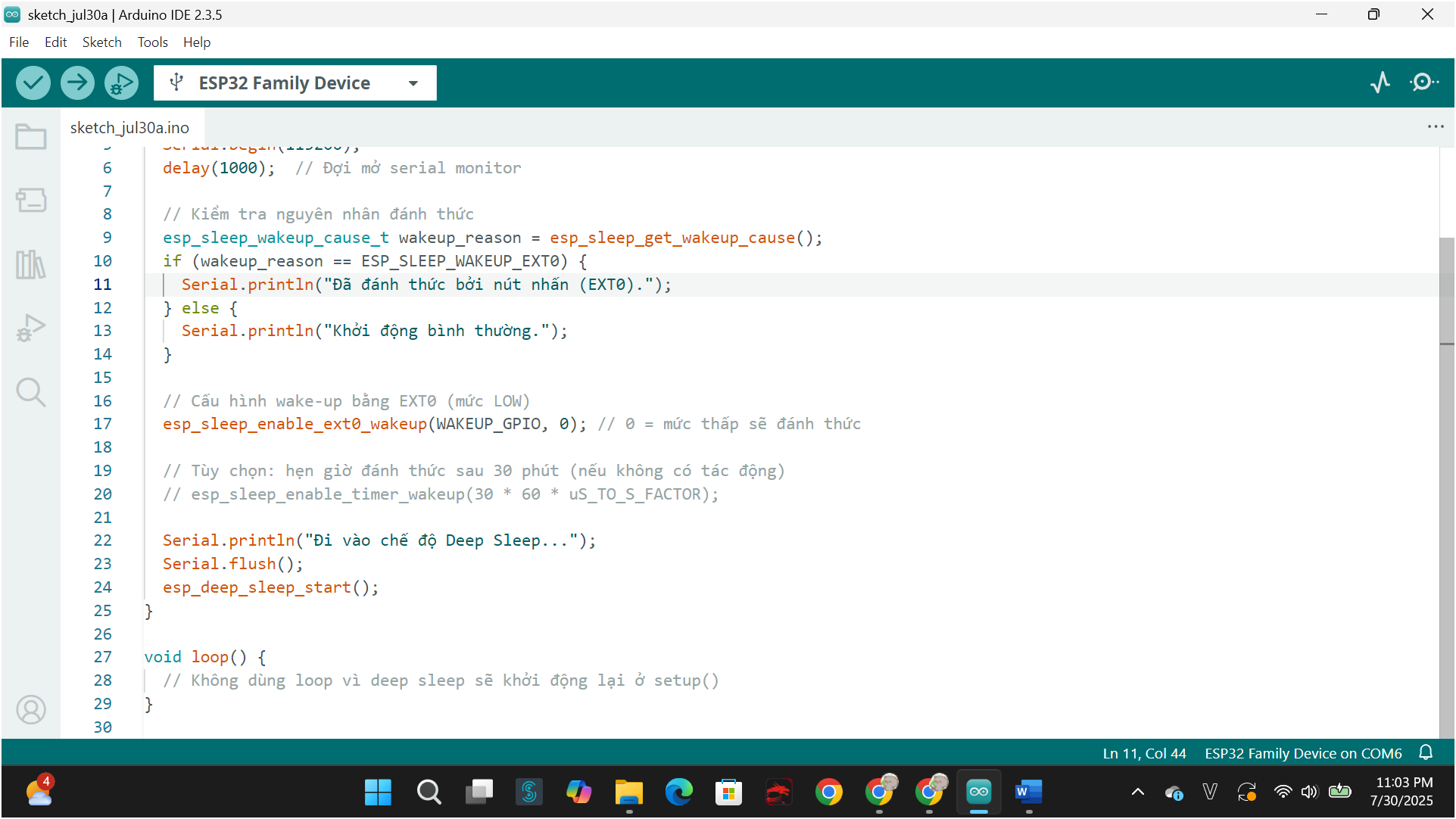
**2. Thực nghiệm: trên ESP32-C6-SUPER MINI**

- Sơ đồ kết nối chân:

|  |  |  |
| --- | --- | --- |
| Chân kết nối ESP32-C6-SUPER MINI | Nút nhấn | MAX30102 |
| GPIO1 | 1 chân | --- |
| GND | 1 chân | --- |
| GPIO5 | --- | SDA |
| GPIO6 | --- | SCL |
| 3.3V | --- | VIN |
| GND | --- | GND |

***\*Nguồn cấp điện: sử dụng nguồn từ laptop***





- Kết quả:

+ Trạng thái chưa ổn định, đạt 50%. Chưa tiết kiệm được năng lượng đáng kể. Sử dụng tối đa được 1 ngày (dưới 20h)

- Cần cải tiến-

+ Tiết kiệm năng lượng hơn

+ Kéo dài thời gian sử dụng từ 1 đến 2 ngày

+ Có thể thay thế chế độ đánh thức bằng nút nhấn bằng: chu kì thời gian, tín hiệu từ cảm biến (như nhịp tim,…..), kết hợp nhiều nguồn như nút nhấn, hẹn giờ, chuyển động, …..

**II. NGUỒN CẤP CHO ESP32-S3-SUPER MINI**

**1. Lý thuyết**: dựa trên ESP32-S3-SUPER MINI

- **Không cấp trực tiếp pin LiPo (3.0–4.2V)** vào chân 3.3V vì dễ gây hỏng chip.

- **Dùng bộ ổn áp (LDO) 3.3V** với dòng tĩnh thấp và điện áp dropout thấp, ví dụ:

* HT7833, SPX3819, MCP1700
* Tránh AMS1117 (dòng tĩnh cao ~11 mA, hao pin)

**- Thêm tụ lọc 10–100 µF** ở đầu ra và **470 µF gần ESP32** để chống sụt áp khi Wi-Fi hoạt động.

|  |  |  |
| --- | --- | --- |
| **Cảm biến** | Khi hoạt động | Khi không đo (shutdown) |
| MAX30102 | ~600 µA | ~0.7 µA |
| MLX90614 | ~1.3 mA | Không có chế độ shutdown |
| MPU6050 | ~0.5–3.6 mA | ~5 µA (idle) |

**2. Thực nghiệm: trên ESP32-C6-SUPER MINI**

- Sau khi cấp nguồn trực tiếp từ pin Lipo vào MCU, nhận thấy có hiện tượng nóng lên và cháy khét

- Sử dụng tụ lọc và điện trở thì thấy hiện tượng nóng ở MCU đã được thuyên giảm khoảng 20%. Khiến cho MCU hoạt động không ổn định

**KẾ HOẠCH TRIỂN KHAI**

**Tại sao phải sử dụng Deep-Sleep trên ESP32-S3-SUPER MINI?**

*Trả lời:* Ví dụ: Pin lipo 2000mAh cho thiết bị

- Nếu không dùng Deep-Sleep

+ ESP tiêu thụ trung bình khoảng 100mA

+ Thời sử dụng = 2000mAh / 100mAh = ~20 giờ

=> Chưa được 1 ngày   
- Nếu dùng Deep-Sleep: Giả sử 30s hoạt động 1 lần, trong đó:

+ Hoạt động 1s tốn 100mA

+ Ngủ 29s tốn 20µA

Vậy ta tính được mức tiêu thu trung bình như sau:

(1s \* 100mA + 29s \* 0.02mA) / 30s = 3.35mA

+ Thời gian dùng: 2000mAh / 3.35mA = 358 giờ = ~ 15 ngày

**THỰC NGHIỆM TRÊN ESP-32-C6-SUPER MINI**

**Các linh kiện kèm theo: Cảm biến nhịp tim và cảm biến gia tốc**

**Bước 1 – Basic:**

* **Deep-sleep → wake bằng timer/nút → đo → xử lý AI → hiển thị → ngủ lại.**

Luồng hoạt động: Bật nguồn / Wake → Khởi động cảm biến → Đọc dữ liệu → Xử lý → In kết quả → Cắt GND cảm biến → Sleep → (Wake lại sau 30s hoặc do nút nhấn)

- Kết quả mong đợi (hiển thị trên Serial Monitor):

+ Thông báo trạng thái ban đầu: ESP32-C6 đang khởi động...

+ Đọc dữ liệu từ cảm biến nhịp tim và gia tốc và in ra kết quả

+ Sau đó đi vào chế độ Deep-Sleep trong 10s

=> quá trình này lặp đi lặp lại liên tục   
BẢNG KẾT NỐI PHẦN CỨNG:

|  |  |  |
| --- | --- | --- |
| Thiết bị | ESP32-C6 | Ghi chú |
| MPU6050 (I2C) | | |
| SDA | GPIO4 | Dùng chung với MAX30102 |
| SCL | GPIO5 | Dùng chung với MAX30102 |
| VCC | 3.3V | Không cấp 5V vì ESP32 logic 3.3V |
| GND | Collector của transistor NPN | GND chỉ được cấp khi GPIO6 HIGH |
| MAX30102 (I2C) | | |
| SDA | GPIO4 | Dùng chung với MPU6050 |
| SCL | GPIO5 | |  | | --- | |  |   Dùng chung với MPU6050 |
| VCC | 3.3V | Không cấp 5V vì ESP32 logic 3.3V |
| GND | Collector của transistor NPN | Dùng chung GND qua transistor |
| Transistor NPN C1815 GR | | Dùng để bật/tắt GND cho cả MPU6050 + MAX30102 |
| Collector | GND của 2 cảm biến | Khi GPIO6 HIGH thì dòng chảy từ GND cảm biến về GND ESP |
| Emitter | GND ESP32 | Nối trực tiếp |
| Base qua điện trở 1k | GPIO6 | Điều khiển BẬT/TẮT cảm biến |
| Nút nhấn wakeup | GPIO0 | Một đầu nối GND, đầu còn lại nối GPIO0 – khi nhấn sẽ tạo mức LOW |
| Nguồn cho ESP32-C6 | | Qua cổng USB của laptop |

CODE:

#include <Wire.h>

#include <Adafruit\_Sensor.h>

#include <Adafruit\_MPU6050.h>

#include "MAX30105.h"

#define SDA\_PIN 4               // SDA cho cả MPU6050 và MAX30102

#define SCL\_PIN 5               // SCL cho cả MPU6050 và MAX30102

#define SENSOR\_POWER\_PIN 6      // Dùng GPIO6 để cấp GND cho cảm biến (qua transistor)

#define WAKE\_BUTTON\_PIN 0       // Nút nhấn wakeup dùng GPIO0

#define TIME\_TO\_SLEEP 30        // Thời gian deep sleep (đơn vị: giây)

MAX30105 particleSensor;

Adafruit\_MPU6050 mpu;

void setup() {

  Serial.begin(115200);

  delay(500);

  // Khởi tạo GPIO

  pinMode(SENSOR\_POWER\_PIN, OUTPUT);

  digitalWrite(SENSOR\_POWER\_PIN, HIGH); // Bật transistor cấp GND cho cảm biến

  delay(100);

  // Khởi tạo I2C

  Wire.begin(SDA\_PIN, SCL\_PIN);

  delay(100);

  // Khởi tạo MPU6050

  if (!mpu.begin()) {

    Serial.println("MPU6050 không kết nối được!");

  } else {

    Serial.println("MPU6050 kết nối thành công.");

  }

  // Khởi tạo MAX30102

  if (!particleSensor.begin(Wire, I2C\_SPEED\_STANDARD)) {

    Serial.println("MAX30102 không kết nối được!");

  } else {

    Serial.println("MAX30102 kết nối thành công.");

  }

  // Đọc dữ liệu mẫu

  readAndProcessData();

  // Tắt nguồn cảm biến để tiết kiệm điện

  digitalWrite(SENSOR\_POWER\_PIN, LOW);

  delay(100);

  // Cấu hình wakeup từ GPIO0 (LOW level)

  esp\_deep\_sleep\_enable\_gpio\_wakeup(1ULL << WAKE\_BUTTON\_PIN, ESP\_GPIO\_WAKEUP\_GPIO\_LOW);

  // Cấu hình sleep sau TIME\_TO\_SLEEP giây

  esp\_sleep\_enable\_timer\_wakeup(TIME\_TO\_SLEEP \* 1000000ULL);

  Serial.println("Đang chuyển sang chế độ deep sleep...");

  delay(500);

  esp\_deep\_sleep\_start();

}

void readAndProcessData() {

  Serial.println("Đang đọc dữ liệu...");

  // Đọc MPU6050

  sensors\_event\_t a, g, temp;

  mpu.getEvent(&a, &g, &temp);

  Serial.print("Gia tốc (m/s^2): X=");

  Serial.print(a.acceleration.x);

  Serial.print(" Y=");

  Serial.print(a.acceleration.y);

  Serial.print(" Z=");

  Serial.println(a.acceleration.z);

  // Đọc MAX30102 (hồng ngoại)

  long irValue = particleSensor.getIR();

  Serial.print("Giá trị IR: ");

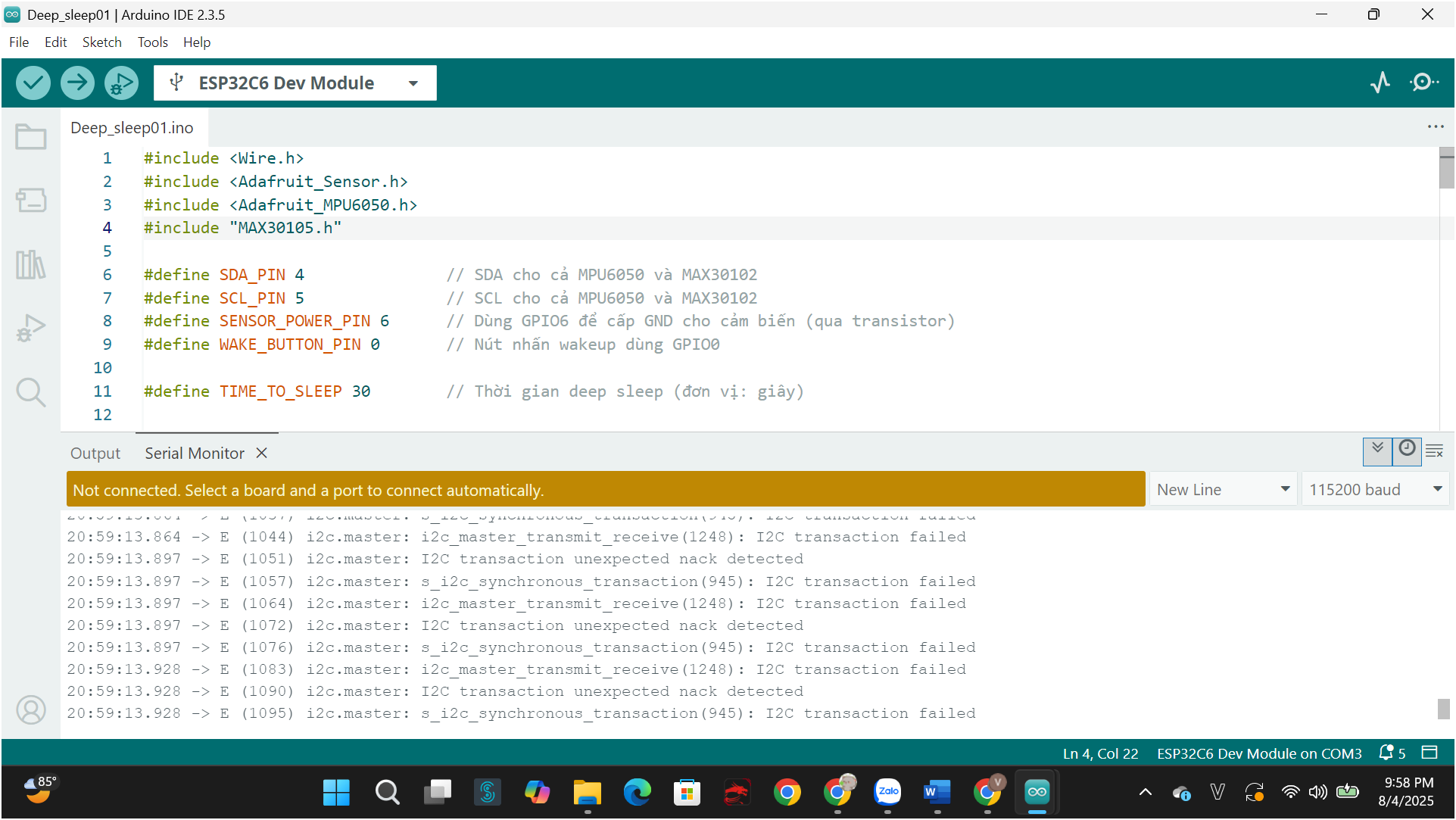
  Serial.println(irValue);

}

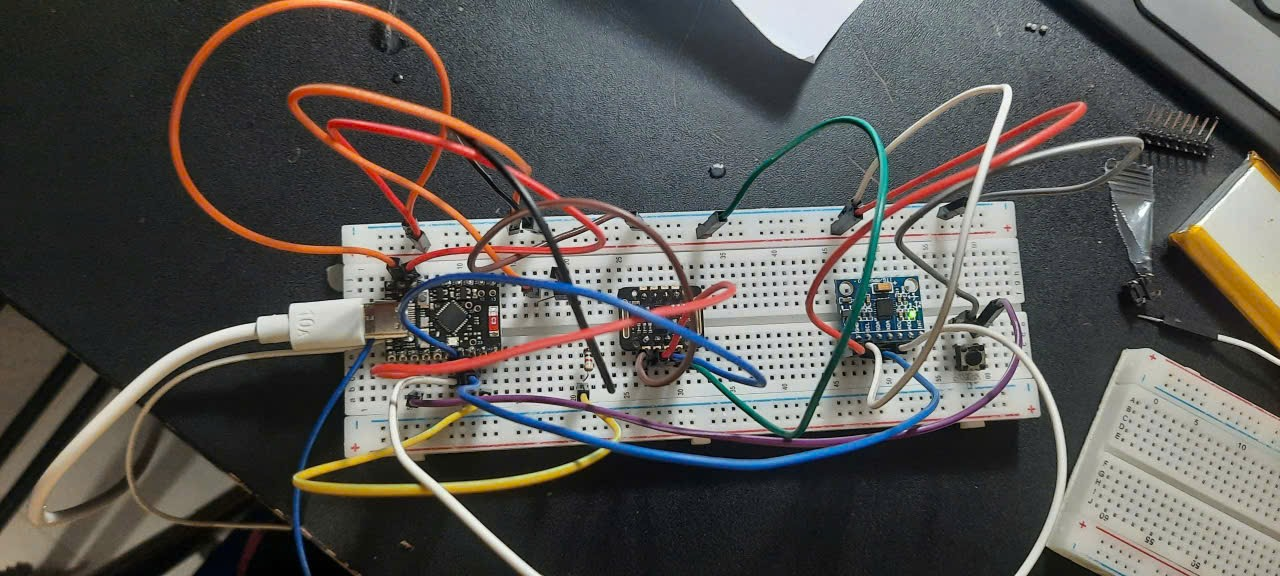
void loop() {

  // Không dùng loop trong deep sleep project

}

KẾT QUẢ: Hiển thị lỗi   


- ESP32-C6 đang cố giao tiếp với một thiết bị qua giao thức **I2C (ví dụ: MPU6050 hoặc MAX30102)** nhưng thiết bị **không trả lời**, dẫn đến lỗi NACK (Not Acknowledge).



-Từ ảnh thực tế:

+ Đèn của cảm biến nhịp tim sáng yếu hoặc đôi khi không sáng. Suy ra có thể cảm biến không hoạt động bình thường   
+ Đèn của cảm biến gia tốc sáng liên tục, rất bình thường. Suy ra cảm biến nhận đủ nguồn năng lượng nhưng chân tín hiệu hoạt động chưa đúng.   
+ Khi cấp nguồn bằng pin Lipo thì nhận thấy các linh kiện không nhận được năng lượng mặc dù pin Lipo đã được sạc đầy (điện áp khoảng 4.16V)

Bước 2 – Tối ưu điện:

* Tắt màn hình khi không dùng.
* Cắt nguồn cho cảm biến I2C khi không đo (qua MOSFET).

Bước 3 – Wake Stub:

* Kiểm tra nút nhấn hoặc cờ trong RTC RAM để boot hoặc không boot.

Bước 4 – ULP (nâng cao, có thể có hoặc không):

* Gán ULP đo ADC pin hoặc tín hiệu rung → wake CPU khi cần.