Báo Cáo Giữa Kỳ Lập Trình Nhúng

Đề Tài

1. Input

**a) Cảm biến DHT11 (Nhiệt độ & độ ẩm)**

Cấu tạo & nguyên lý: DHT11 là cảm biến số, bên trong có NTC để đo nhiệt độ và điện cực polymer để đo độ ẩm. Xuất dữ liệu qua 1 chân duy nhất (Data).

-Kết nối:

DHT11 → ESP32/Arduino: VCC (3.3V), GND, DATA (GPIO4/D4).

-Khai báo thư viện & biến:

*#include <DHT.h>*

*#define DHTPIN 4*

*#define DHTTYPE DHT11*

*DHT dht(DHTPIN, DHTTYPE);*

-Sử dụng trong code:

dht.begin(); khởi động cảm biến.

dht.readTemperature() và dht.readHumidity() để lấy dữ liệu.

**b) Nút bấm BTN1 (GPIO18)**

Chức năng:

Nhấn ngắn → đọc nhiệt độ/độ ẩm + LED nháy 2 lần.

Giữ ≥1.2s → dừng toàn bộ chương trình (STOP).

**-**Kết nối: Nút bấm nối với GPIO18 và dùng điện trở kéo lên (INPUT\_PULLUP).

**-**Khai báo:

*#define BTN\_SHOW 18*

*pinMode(BTN\_SHOW, INPUT\_PULLUP);*

**c) Nút bấm BTN2 (GPIO19)**

-Chức năng: Bật/tắt relay bằng tay, ưu tiên 10s.

-Kết nối: Nút nối GPIO19, dùng INPUT\_PULLUP.

-Khai báo:

*#define BTN\_TOGGLE 19*

*pinMode(BTN\_TOGGLE, INPUT\_PULLUP);*

**2. Output**

a) Relay + thiết bị tải (GPIO5)

**-**Cấu tạo: Relay module 1 kênh, đóng/ngắt nguồn AC hoặc DC.

**-**Chức năng:

**+**Tự động bật khi nhiệt độ > 30°C.

**+**Bật/tắt thủ công bằng BTN2.

**-**Kết nối: IN → GPIO5, VCC → 5V, GND → GND.

**-**Khai báo:

*#define RELAY\_PIN 5*

*pinMode(RELAY\_PIN, OUTPUT);*

b) LED báo hiệu (GPIO13)

Chức năng: Nháy 2 lần khi BTN1 được nhấn ngắn.

Khai báo:

*#define LED\_BTN1 13*

*pinMode(LED\_BTN1, OUTPUT);*

**3. Process (Điều khiển trung tâm)**

-Phần cứng: ESP32 hoặc Arduino. ESP32 được khuyên dùng để dễ mở rộng IoT.

-Cơ chế hoạt động:

Auto: Nếu không nhấn nút, ESP32 sẽ đọc DHT11 định kỳ (0.8s).

Nhiệt độ > 30°C → bật relay (quạt/thiết bị tải).

Nhiệt độ ≤ 30°C → tắt relay.

Manual (BTN2): Ưu tiên điều khiển relay bằng tay trong 10s.

Stop (BTN1 giữ lâu): Ngắt toàn bộ hệ thống.

**4. Yêu Cầu Trình Bày**

1. Cảm biến thì trả về giá trị gì? Biên độ giá trị và tính tương đương với thông tin thực ngoài môi trường. Lấy ví dụ trong chương trình đang code

-Cảm biến trả về giá trị gì?

+Loại cảm biến: DHT11 (cảm biến nhiệt độ và độ ẩm).

+Dữ liệu trả về gồm 2 giá trị số:

+Nhiệt độ: đơn vị °C, đọc bằng hàm:

*float t = dht.readTemperature();*

+Độ ẩm tương đối: đơn vị %RH, đọc bằng hàm

*float h = dht.readHumidity();*

-Biên độ đo của DHT11

+Nhiệt độ: đo được từ 0 °C đến 50 °C, với sai số khoảng ±2 °C.

+Độ ẩm: đo được từ 20 %RH đến 90 %RH, với sai số khoảng ±5%.

+Chu kỳ đo: tối thiểu 1 giây/lần (trong code bạn đặt READ\_PERIOD\_MS = 800, tức 0.8 giây, hơi nhanh hơn khuyến nghị nhưng vẫn chạy được).

-Tính tương đương với thông tin ngoài môi trường

+Khi cảm biến trả về số đo, giá trị đó xấp xỉ điều kiện thực tế:

25 °C, 50 %RH → nhiệt độ phòng thoải mái, khô ráo.

32 °C, 70 %RH → môi trường nóng ẩm, dễ khó chịu, nên bật quạt/điều hòa.

<15 °C → lạnh, cần sưởi ấm.

>40 °C → rất nóng, có nguy cơ ảnh hưởng sức khỏe.

+Sai số nhỏ của DHT11 không làm thay đổi nhiều trong các ứng dụng cơ bản như bật quạt, giám sát phòng, hoặc demo IoT.

- Ví dụ ngay trong chương trình

*// ===== CHẾ ĐỘ AUTO =====*

*static unsigned long lastRead = 0;*

*unsigned long now = millis();*

*// Chỉ auto nếu không override tay*

*if (!(manualOverride && now < overrideUntil)) {*

*manualOverride = false;*

*if (now - lastRead >= READ\_PERIOD\_MS) {*

*lastRead = now;*

*float t = dht.readTemperature();*

*if (!isnan(t)) {*

*if (t > TEMP\_THRESHOLD) digitalWrite(RELAY\_PIN, HIGH); // Nhiệt độ cao -> bật relay*

*else digitalWrite(RELAY\_PIN, LOW); // Nhiệt độ thấp -> tắt relay*

*} else {*

*digitalWrite(RELAY\_PIN, LOW); // Lỗi đọc cảm biến -> tắt relay cho an toàn*

*}*

*}*

*}*

-Nghĩa là:

+Nếu nhiệt độ đo được > 30 °C (ví dụ t = 32.5 °C, h = 70% RH) → relay bật → có thể điều khiển quạt hoặc motor để làm mát.

+Nếu ≤ 30 °C (ví dụ t = 27 °C, h = 50% RH) → relay tắt → hệ thống nghỉ.

2. Lập trình nút bấm để mạch Arduino/ESP đọc giá trị cảm biến và hiển thị ra màn hình LCD/màn hình máy tính

2.1.Nút bấm được sử dụng

-Nút bấm chính là BTN1, được gán vào chân GPIO18 trên ESP32

*#define BTN\_SHOW 18*

*pinMode(BTN\_SHOW, INPUT\_PULLUP);*

- Ở chế độ INPUT\_PULLUP, nút chưa nhấn thì đọc là HIGH, khi nhấn xuống thì đọc là LOW.

-Đây là cách chống trôi tín hiệu mà không cần thêm điện trở ngoài.

2.2 Cơ chế xử lý nút bấm trong code

-Trong hàm handleButton1():

-Khi nhấn ngắn BTN1:

+LED GPIO13 (LED báo hiệu) nháy 2 lần để xác nhận thao tác.

+ESP đọc dữ liệu từ cảm biến DHT11:

*float t = dht.readTemperature();*

*float h = dht.readHumidity();*

+ Giá trị được hiển thị ra màn hình Serial Monitor trên máy tính:

*Serial.print("Nhiet do: "); Serial.print(t); Serial.print(" \*C, ");*

*Serial.print("Do am: "); Serial.print(h); Serial.println(" %");*

- Khi giữ BTN1 ≥ 1.2 giây: chương trình chuyển sang trạng thái STOP (tắt relay, tắt LED, dừng loop).

-code

*if (held < HOLD\_STOP\_MS) { // Nhấn ngắn*

*blinkTwice(); // LED GPIO13 nháy 2 lần*

*float t = dht.readTemperature();*

*float h = dht.readHumidity();*

*if (isnan(t) || isnan(h)) {*

*Serial.println("DHT11 loi (NaN) -> kiem tra day & 3V3");*

*} else {*

*Serial.print("Nhiet do: "); Serial.print(t); Serial.print(" \*C, ");*

*Serial.print("Do am: "); Serial.print(h); Serial.println(" %");*

*}*

*}*

2.3. Hiển thị trên màn hình máy tính

Sau khi nạp code, mở Serial Monitor (hoặc Serial Plotter) trên Arduino IDE/PlatformIO.

Chọn đúng tốc độ truyền 115200 baud.

Khi nhấn BTN1 → trên Serial sẽ in ra kết quả dạng:

-Nhiet do: 28.5 \*C, Do am: 62 %

3.Ứng dụng của cảm biến là gì?

-Trong Code:

+DHT11 cung cấp nhiệt độ và độ ẩm → được dùng để điều khiển relay.

+Khi nhiệt độ > 30 °C → relay bật (có thể nối quạt/motor để làm mát).

+Khi nhiệt độ ≤ 30 °C → relay tắt.

+Đây là ứng dụng tự động bật/tắt thiết bị làm mát theo môi trường.

- Ứng dụng thực tế trong đời sống

+Điều khiển quạt / điều hòa tự động

+Khi phòng nóng (nhiệt độ cao) → bật quạt, điều hòa.

+Khi mát hơn → tự động tắt để tiết kiệm điện.

+Trạm thời tiết mini

+Ghi nhận nhiệt độ và độ ẩm theo thời gian.

+Có thể gửi dữ liệu lên máy tính hoặc IoT cloud (ví dụ:web nội bộ) để theo dõi từ xa.

-hà kính nông nghiệp 🌱

+Khi nhiệt độ cao → bật quạt thông gió.

+Khi độ ẩm thấp → bật máy phun sương/tưới cây

+Đảm bảo môi trường ổn định cho cây trồng.

-Hệ thống cảnh báo môi trường

+Cảnh báo khi độ ẩm quá cao (dễ nấm mốc, hỏng đồ đạc).

+Cảnh báo khi nhiệt độ vượt ngưỡng nguy hiểm cho sức khỏe.

-Ứng dụng trong IoT học tập và nghiên cứu

+Thường dùng để demo các dự án IoT: gửi dữ liệu lên server, hiển thị trên web/app.

+Giúp sinh viên dễ hiểu nguyên lý “cảm biến → xử lý → hiển thị/điều khiển thiết bị”.

4.Lập trình để đèn LED sáng khi giá trị cảm biến vượt một ngưỡng nào đó. Nếu giá trị giảm thì đèn tắt.

-Trong chương trình, sử dụng cảm biến DHT11 để đo nhiệt độ. Giá trị nhiệt độ được so sánh với một ngưỡng định trước là 30 °C (TEMP\_THRESHOLD = 30.0).

-Nguyên lý hoạt động:

+Khi nhiệt độ đo được cao hơn ngưỡng thì hệ thống sẽ kích hoạt LED cảnh báo. LED không sáng liên tục mà nhấp nháy theo chu kỳ để dễ nhận biết bằng mắt thường.

+Khi nhiệt độ giảm xuống bằng hoặc thấp hơn ngưỡng, LED sẽ tắt hoàn toàn, báo hiệu môi trường đã trở lại trạng thái an toàn.

-Cách lập trình:

+LED được khai báo ở chân GPIO13 (#define LED\_BTN1 13).

+Dùng biến thời gian (millis()) để điều khiển LED nhấp nháy non-blocking (không làm treo chương trình như khi dùng delay()).

+Trong hàm loop(), sau khi đọc nhiệt độ từ cảm biến, ta bổ sung đoạn kiểm tra điều kiện và điều khiển LED như sau:

*// ===== ĐÈN LED CẢNH BÁO =====*

*static unsigned long lastBlink = 0; // Lưu lần đổi trạng thái LED gần nhất*

*static bool ledState = LOW; // Trạng thái hiện tại của LED*

*unsigned long nowBlink = millis();*

*if (t > TEMP\_THRESHOLD) {*

*// Khi vượt ngưỡng -> LED nhấp nháy mỗi 400ms*

*if (nowBlink - lastBlink >= 400) {*

*lastBlink = nowBlink;*

*ledState = !ledState;*

*digitalWrite(LED\_BTN1, ledState);*

*}*

*} else {*

*// Khi nhỏ hơn hoặc bằng ngưỡng -> LED tắt hẳn*

*digitalWrite(LED\_BTN1, LOW);*

*ledState = LOW;*

*}*

-Kết quả thực tế:

Nếu nhiệt độ 32 °C → LED GPIO13 sẽ sáng liên tục.

Nếu nhiệt độ giảm còn 28–29 °C → LED sẽ tắt hẳn.

Việc điều khiển LED bằng millis() giúp hệ thống vẫn xử lý song song các chức năng khác như nút bấm BTN1/BTN2 và relay mà không bị gián đoạn.

5. Link video: - **Video trình bày, các hình ảnh linh kiện** [**https://www.youtube.com/watch?si=5Jrh0GAjzVuubY9-&v=fthzbg-XTLQ&feature=youtu.be**](mailto:mailtohttps://www.youtube.com/watch?si=5Jrh0GAjzVuubY9-&v=fthzbg-XTLQ&feature=youtu.be)

**-Trình bày bài toán yêu cầu và kết quả đạt được**

[**https://drive.google.com/drive/folders/193cal1KkoTtZPXLtbsVl3lJI\_aQwdscn?hl=vi**](mailto:https://drive.google.com/drive/folders/193cal1KkoTtZPXLtbsVl3lJI_aQwdscn?hl=vi)