TRƯỜNG ĐẠI HỌC KHOA HỌC

KHOA CÔNG NGHỆ THÔNG TIN

HỆ THỐNG GHI NHẬT KÝ MÔI TRƯỜNG VỚI ESP32

TÊN LỚP HỌC PHẦN: PHÁT TRIỂN ỨNG DỤNG IOT
MÃ HỌC PHẦN: 2024-2025.2.TIN4024.006
GIẢNG VIÊN HƯỚNG DẪN: VÕ VIỆT DŨNG

HUẾ, THÁNG 04 NĂM 2025

Mục lục

Mở đầu

Trong bối cảnh công nghệ Internet of Things (IoT) ngày càng phát triển, các hệ thống giám sát môi trường thông minh đã trở thành một phần không thể thiếu trong nhiều lĩnh vực như nông nghiệp, công nghiệp, y tế và đời sống hàng ngày. Một trong những yêu cầu cốt lõi của các hệ thống IoT là khả năng thu thập, truyền tải và phân tích dữ liệu từ xa thông qua kết nối Internet. Tuy nhiên, thực tế cho thấy kết nối mạng không phải lúc nào cũng ổn định, đặc biệt ở những khu vực xa xôi hoặc trong các tình huống khẩn cấp. Điều này đặt ra thách thức về việc đảm bảo dữ liệu quan trọng, chẳng hạn như nhiệt độ và độ ẩm môi trường, không bị mất mát khi hệ thống tạm thời bị gián đoạn kết nối.

Đề tài "Hệ thống ghi nhật ký môi trường với ESP32" được xây dựng nhằm giải quyết bài toán trên bằng cách kết hợp vi điều khiển ESP32 – một nền tảng IoT mạnh mẽ, chi phí thấp – với cảm biến nhiệt độ và độ ẩm DHT22, module thẻ nhớ MicroSD và nền tảng phân tích dữ liệu Blynk. Trong hệ thống này, việc đẩy dữ liệu lên Internet để theo dõi từ xa là mục tiêu chính, nhưng lưu trữ cục bộ trên thẻ SD đóng vai trò như một biện pháp hỗ trợ quan trọng. Thẻ SD không chỉ đảm bảo dữ liệu được ghi lại liên tục ngay cả khi không có kết nối mạng, mà còn hoạt động như một bộ đệm, cho phép hệ thống lưu trữ các bản ghi nhiệt độ, độ ẩm cùng dấu thời gian để sau đó gửi lên web theo các khoảng thời gian định sẵn.

Không dừng lại ở việc hiển thị giá trị thời gian thực, hệ thống hướng đến khả năng phân tích dữ liệu lịch sử thông qua nền tảng Blynk. Dữ liệu được gửi định kỳ từ ESP32 lên Blynk sẽ được lưu trữ trên đám mây và hiển thị dưới dạng biểu đồ thời gian (như SuperChart), giúp người dùng nhận diện xu hướng, biến động môi trường hoặc đưa ra các quyết định dựa trên phân tích dài hạn. Điều này khác biệt với các hệ thống chỉ tập trung vào giám sát thời gian thực, vốn không cung cấp cái nhìn tổng quan về lịch sử dữ liệu.

Nghiên cứu này tập trung vào việc tìm hiểu lý thuyết và thiết kế hệ thống, bao gồm việc đọc dữ liệu từ cảm biến, lưu trữ ngẫu nhiên vào thẻ SD (giả lập các giá trị nhiệt độ và độ ẩm nếu cần), và gửi dữ liệu lên Blynk để phân tích. Thông qua mô phỏng trên nền tảng Wokwi, đề tài sẽ minh họa cách hệ thống hoạt động, từ việc ghi

nhật ký cục bộ đến đồng bộ dữ liệu lên web, đồng thời đặt nền tảng cho các ứng dụng thực tế trong tương lai. Với sự kết hợp giữa tính linh hoạt của ESP32, khả năng lưu trữ cục bộ của thẻ SD và công cụ phân tích mạnh mẽ của Blynk, hệ thống hứa hẹn mang lại một giải pháp hiệu quả, đáng tin cậy cho việc giám sát và phân tích môi trường theo thời gian.

Nội dung

Chương 1: Giới thiệu

1.1. Tổng quan về bài toán

Giám sát các thông số môi trường như nhiệt độ và độ ẩm là yêu cầu thiết yếu trong nhiều lĩnh vực, bao gồm nông nghiệp thông minh, quản lý kho vận, phòng thí nghiệm, và nâng cao chất lượng không gian sống. Các hệ thống truyền thống thường gặp hạn chế về chi phí, tính linh hoạt và khả năng phân tích dữ liệu từ xa. Với sự phát triển mạnh mẽ của công nghệ Internet of Things (IoT), đặc biệt là sự xuất hiện của các vi điều khiển hiện đại như ESP32, việc xây dựng các hệ thống giám sát có chi phí thấp, kích thước nhỏ gọn và hiệu quả cao đã trở nên khả thi hơn bao giờ hết.

Tuy nhiên, một thách thức lớn đối với các hệ thống IoT là việc duy trì khả năng lưu trữ dữ liệu liên tục ngay cả khi kết nối mạng bị gián đoạn. Bên cạnh đó, chỉ hiển thị dữ liệu thời gian thực là chưa đủ; việc lưu trữ và phân tích dữ liệu lịch sử là cần thiết để nhận diện xu hướng, dự báo tình trạng bất thường và hỗ trợ ra quyết định kịp thời.

Đề tài nghiên cứu này tập trung vào xây dựng một "Hệ thống ghi nhật ký môi trường sử dụng ESP32", trong đó cảm biến DHT22 được sử dụng để đo nhiệt độ và độ ẩm môi trường. Hệ thống sẽ ghi nhận dữ liệu cục bộ vào thẻ nhớ MicroSD như một phương án lưu trữ dự phòng, đồng thời đóng vai trò bộ đệm tạm thời. Dữ liệu sẽ được gửi định kỳ tới nền tảng web Blynk để người dùng dễ dàng theo dõi và phân tích lịch sử dữ liệu qua giao diện biểu đồ trực quan, mở rộng phạm vi quan sát vượt ra ngoài dữ liệu thời gian thực.

1.2. Mục tiêu nghiên cứu

Mục tiêu chính của nghiên cứu là thiết kế và mô phỏng một hệ thống ghi nhật ký môi trường với các yêu cầu cụ thể như sau:

- Thu thập dữ liệu môi trường chính xác:Ứng dụng cảm biến DHT22 để thu thập

giá trị nhiệt độ và độ ẩm với độ chính xác cao.

- Lưu trữ dữ liệu cục bộ:Ghi dữ liệu vào thẻ nhớ MicroSD dưới dạng file CSV, đảm bảo dữ liệu không bị mất trong trường hợp mất kết nối Internet.
- Truyền dữ liệu lên nền tảng đám mây:Kết nối vi điều khiển ESP32 với mạng Wi-Fi và định kỳ gửi dữ liệu từ cảm biến hoặc từ bộ nhớ đệm lên nền tảng Blynk để phân tích.
- Phân tích và hiển thị dữ liệu lịch sử:Sử dụng các công cụ trực quan của Blynk, như widget SuperChart, để theo dõi xu hướng biến động của nhiệt độ và độ ẩm.
- Mô phỏng trên nền tảng Wokwi:Xây dựng mô hình mô phỏng toàn bộ hệ thống trong môi trường Wokwi để kiểm chứng tính khả thi trước khi triển khai thực tế.
- Mục tiêu dài hạn của nghiên cứu là phát triển một giải pháp giám sát môi trường bền vững, dễ dàng mở rộng và phù hợp cho nhiều ứng dụng khác nhau trong tương lai.

1.3. **Đối tượng và phạm vi nghiên cứu**

Đối tượng nghiên cứu:

Đối tượng nghiên cứu của đề tài là **hệ thống ghi nhật ký môi trường sử dụng vi điều khiển ESP32**, bao gồm các thành phần chính:

- **ESP32:** vi điều khiển trung tâm, thu thập và xử lý dữ liệu.
- **Cảm biến DHT22:** đo nhiệt độ và độ ẩm môi trường.
- **Module thẻ nhớ MicroSD:** lưu trữ dữ liệu cục bộ (offline) nhằm đảm bảo dữ liệu không bị mất khi mất kết nối Internet.
- **Nền tảng Blynk:** phân tích và hiển thị dữ liệu lịch sử nhiệt độ, độ ẩm dưới dạng biểu đồ theo thời gian để phục vụ cho việc đánh giá và ra quyết định.
- **Nền tảng mô phỏng Wokwi:** phục vụ việc mô phỏng, kiểm thử thiết kế hệ thống trước khi triển khai thực tế.

Ngoài ra, nghiên cứu cũng tập trung vào việc khai thác tính năng phân tích theo thời gian của Blynk, không chỉ dừng lại ở hiển thị dữ liệu thời gian thực mà còn lưu trữ và trực quan hóa dữ liệu lịch sử để hỗ trợ phân tích xu hướng biến động môi trường.

Phạm vi nghiên cứu:

- **Phạm vi phần cứng:** Giới hạn trong các thành phần cảm biến nhiệt độ/độ ẩm DHT22, MicroSD, pushbutton, và vi điều khiển ESP32 trong môi trường mô phỏng Wokwi.
- **Phạm vi phần mềm:** Tập trung vào xây dựng luồng xử lý lý thuyết và mô phỏng mã nguồn trên Visual Studio Code, bao gồm việc thu thập dữ liệu, ghi vào SD, gửi dữ liệu định kỳ lên Blynk và phân tích.
- **Phạm vi mô phỏng:** Nghiên cứu thực hiện trong môi trường mô phỏng Wokwi, không bao gồm kiểm thử thực tế với phần cứng vật lý.
- **Giả lập dữ liệu:** Dữ liệu môi trường (nhiệt độ, độ ẩm) được sinh ngẫu nhiên trong mô phỏng để kiểm tra luồng xử lý.

1.4. Phương pháp nghiên cứu

Nghiên cứu được tiến hành dựa trên các phương pháp chính sau:

- **Nghiên cứu lý thuyết:** Thu thập và phân tích các tài liệu kỹ thuật liên quan đến vi điều khiển ESP32 (board-esp32-devkit-c-v4), nguyên lý hoạt động của cảm biến DHT22 (wokwi-dht22), giao thức truyền thông SPI và hệ thống file của thẻ MicroSD (wokwi-microsd-card). Đồng thời, tìm hiểu về nền tảng Blynk trong việc lưu trữ, hiển thị và phân tích dữ liệu lịch sử để áp dụng vào mô phỏng.
- **Thiết kế mô phỏng:** Sử dụng công cụ Wokwi để xây dựng sơ đồ kết nối các thành phần trong hệ thống (ESP32, DHT22, module MicroSD, pushbutton). Tiến hành mô phỏng hoạt động của cảm biến, ghi dữ liệu vào thẻ nhớ ảo, và giả lập quá trình truyền dữ liệu lên nền tảng Blynk thông qua Serial Monitor.
- **Thiết kế phần mềm (mô tả lý thuyết):** Xây dựng sơ đồ luồng xử lý phần mềm cho ESP32 bằng ngôn ngữ lập trình C++. Mô tả chi tiết các thành phần cần thiết trong mã nguồn như khai báo thư viện, đọc dữ liệu từ cảm biến, ghi vào thẻ nhớ SD, và gửi dữ liệu định kỳ lên Blynk.
- **Thử nghiệm mô phỏng:** Thực hiện các kịch bản kiểm thử trên nền tảng Wokwi, bao gồm thay đổi giá trị mô phỏng của cảm biến, giả lập mất kết nối

Wi-Fi, và kiểm tra trạng thái ghi file SD. Ghi nhận kết quả mô phỏng thông qua Serial Monitor và đánh giá tính ổn định của hệ thống.

- **Phân tích và đánh giá:** Tổng hợp kết quả từ quá trình mô phỏng để so sánh với mục tiêu nghiên cứu đã đề ra. Đánh giá những ưu điểm, hạn chế của hệ thống, đồng thời đề xuất các giải pháp nhằm nâng cao độ ổn định, độ chính xác và khả năng mở rộng trong tương lai.

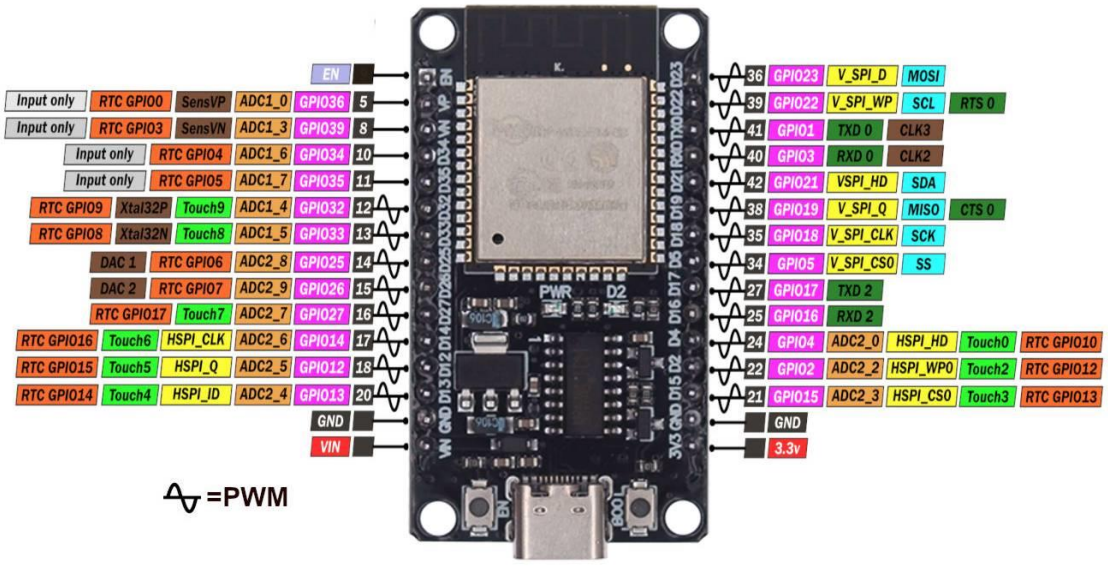
Chương 2: Tổng quan lý thuyết

2.1. Giới thiệu về ESP32

ESP32 là một dòng vi điều khiển hệ thống trên chip (SoC) chi phí thấp, tiêu thụ điện năng thấp, được tích hợp Wi-Fi và Bluetooth kép. Module board-esp32-devkit-c-v4 là một bo mạch phát triển phổ biến dựa trên chip ESP32-WROOM-32E hoặc ESP32-WROOM-32UE, cung cấp một nền tảng dễ dàng để phát triển các ứng dụng IoT [1].

- **Vi xử lý**: Lõi kép Tensilica LX6 32-bit, tốc độ xung nhịp lên đến 240 MHz.
- **Kết nối không dây**: Chuẩn Wi-Fi 802.11 b/g/n và Bluetooth v4.2 BLE.
- **Bộ nhớ**: 520 KB SRAM, và flash ngoài thường là 4 MB hoặc hơn.
- **Ngoại vi**: GPIO (General Purpose Input/Output), ADC (Analog-to-Digital Converter), DAC (Digital-to-Analog Converter), SPI (Serial Peripheral Interface), I2C (Inter-Integrated Circuit), UART (Universal Asynchronous Receiver/Transmitter), cảm biến cảm ứng điện dung, cảm biến Hall, Ethernet MAC, CAN 2.0, IR (hồng ngoại).
- **Giao tiếp SPI**: ESP32 hỗ trợ nhiều kênh SPI, rất quan trọng cho việc giao tiếp với các thiết bị ngoại vi như module thẻ nhớ MicroSD. Giao thức SPI cho phép truyền dữ liệu nối tiếp tốc độ cao giữa vi điều khiển và một hoặc nhiều thiết bị ngoại vi.
- **Nguồn điện**: Hoạt động ở mức điện áp 3.3V. Bo mạch DevKitC V4 có bộ điều chỉnh điện áp tích hợp, hỗ trợ cấp nguồn qua cổng USB hoặc chân VIN (5V). ESP32 cũng hỗ trợ các chế độ tiết kiệm năng lượng như Deep Sleep, Light Sleep, giúp giảm tiêu thụ dòng điện xuống mức rất thấp (dưới 10 µA ở chế độ Deep Sleep).
- **Vai trò trong dự án**: ESP32 đóng vai trò trung tâm, thực hiện các nhiệm vụ: đọc dữ liệu từ cảm biến DHT22, giao tiếp với module thẻ nhớ MicroSD qua

SPI để ghi dữ liệu, kết nối mạng Wi-Fi, đọc và gửi dữ liệu từ MicroSD lên nền tảng Blynk theo định kỳ.



Hình ảnh minh họa bảng mạch ESP32

Nguồn : electra.store

2.2. Cảm biến nhiệt độ và độ ẩm DHT22

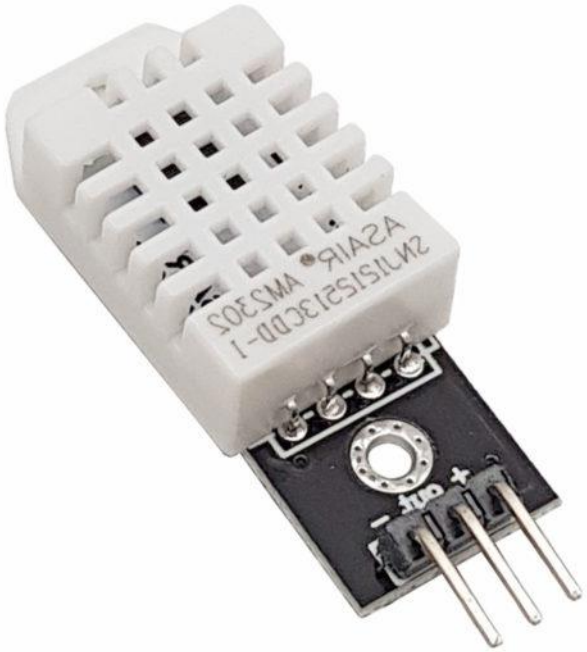
DHT22 (hay AM2302) là một cảm biến kỹ thuật số chi phí thấp dùng để đo nhiệt độ và độ ẩm môi trường xung quanh. Trong dự án này, wokwi-dht22 là mô hình mô phỏng của cảm biến DHT22 trên nền tảng Wokwi.

- **Nguyên lý hoạt động**:
  - **Đo độ ẩm**: Sử dụng cảm biến độ ẩm điện dung. Một lớp điện môi hút ẩm được đặt giữa hai điện cực. Khi độ ẩm thay đổi, hằng số điện môi thay đổi, dẫn đến sự thay đổi điện dung. Mạch tích hợp trong cảm biến chuyển đổi thành tín hiệu số.
  - **Đo nhiệt độ**: Sử dụng nhiệt điện trở (thermistor) có điện trở suất thay đổi theo nhiệt độ.
- **Giao tiếp**: Sử dụng giao thức truyền dữ liệu một dây độc quyền (khác với chuẩn 1-Wire). ESP32 gửi tín hiệu yêu cầu đến chân DATA của DHT22, sau đó

DHT22 phản hồi bằng chuỗi 40 bit dữ liệu chứa độ ẩm (16 bit), nhiệt độ (16 bit) và checksum (8 bit) để kiểm tra lỗi [2].

- **Thông số kỹ thuật**:
    - Điện áp hoạt động: 3.3V - 5.5V.
    - Dòng điện tiêu thụ: Khoảng 1.5 mA khi đo, 50 µA ở chế độ chờ.
    - Dải đo độ ẩm: 0 - 100% RH (độ chính xác ±2-5%).
    - Dải đo nhiệt độ: -40°C đến 80°C (độ chính xác ±0.5°C).
    - Tần số lấy mẫu: Tối đa 0.5 Hz (2 giây/lần đọc).

Trong dự án, DHT22 (wokwi-dht22) được kết nối với một chân GPIO của ESP32 để đọc dữ liệu nhiệt độ và độ ẩm theo chu kỳ, sau đó dữ liệu được ghi vào thẻ MicroSD và gửi lên Blynk.



Hình ảnh minh họa Cảm biến nhiệt độ và độ ẩm DHT22

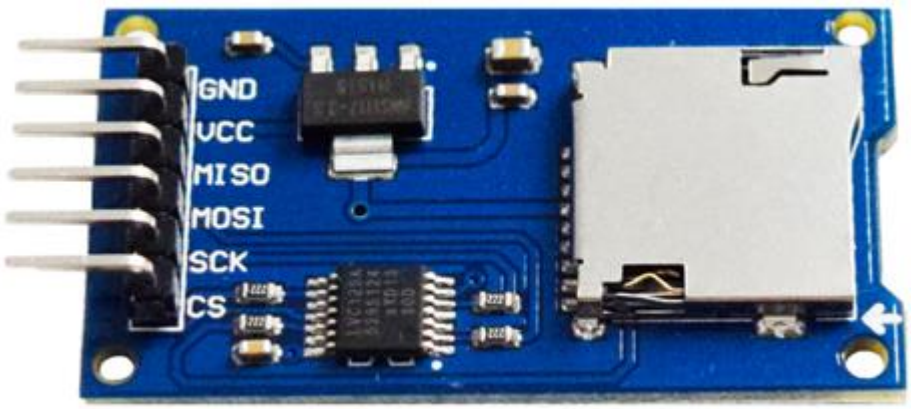
Nguồn : nshopvn

2.3. Module thẻ nhớ MicroSD và giao thức lưu trữ

Module thẻ nhớ MicroSD cho phép ESP32 đọc và ghi dữ liệu vào thẻ nhớ MicroSD, cung cấp khả năng lưu trữ dữ liệu lớn và bền vững. Trong dự án, wokwi-

microsd-card là mô hình mô phỏng của module MicroSD trên Wokwi.

- **Giao tiếp**: Module MicroSD sử dụng giao thức SPI (Serial Peripheral Interface), một giao thức truyền thông nối tiếp đồng bộ với 4 dây:
    - MOSI (Master Out Slave In): Dữ liệu từ ESP32 đến thẻ SD.
    - MISO (Master In Slave Out): Dữ liệu từ thẻ SD đến ESP32.
    - SCK (Serial Clock): Tín hiệu xung nhịp đồng bộ hóa, do ESP32 tạo ra.
    - CS/SS (Chip Select / Slave Select): Chân chọn thẻ SD, kéo xuống mức LOW để kích hoạt giao tiếp [3].
- **Hệ thống tập tin**: Thẻ MicroSD thường được định dạng FAT16 hoặc FAT32. Thư viện SD.h trong Arduino framework hỗ trợ đọc/ghi file trên các định dạng này (lưu ý: exFAT không được hỗ trợ bởi SD.h).
- **Vai trò trong dự án**: Module wokwi-microsd-card lưu trữ cục bộ dữ liệu nhiệt độ và độ ẩm vào file data_log.csv. Dữ liệu được ghi định kỳ, đảm bảo không bị mất khi ESP32 khởi động lại hoặc mất kết nối Wi-Fi. Nếu kết nối Blynk bị gián đoạn, dữ liệu vẫn được lưu và có thể gửi sau khi kết nối được khôi phục.
-



Hình ảnh minh họa Module thẻ nhớ MicroSD

Nguồn : linhkienagv

2.4. Nền tảng Blynk và phân tích dữ liệu theo thời gian

Blynk là một nền tảng IoT cho phép kết nối dễ dàng giữa thiết bị phần cứng (như ESP32) và ứng dụng di động (iOS/Android) hoặc giao diện web để điều khiển và hiển thị dữ liệu [4].

- **Kiến trúc**:
    - **Blynk App/Web Dashboard**: Giao diện người dùng để tạo bảng điều khiển với các widget (nút, biểu đồ, hiển thị giá trị).
    - **Blynk Server**: Xử lý giao tiếp giữa phần cứng và ứng dụng/web. Dự án sử dụng Blynk Cloud (máy chủ công cộng).
    - **Blynk Libraries**: Thư viện hỗ trợ ESP32 kết nối và trao đổi dữ liệu với server.
- **Virtual Pins**: Blynk sử dụng "Chân ảo" (Virtual Pins) để trao đổi dữ liệu. ESP32 gửi dữ liệu nhiệt độ và độ ẩm lên các Virtual Pin (ví dụ: V1 cho nhiệt độ, V2 cho độ ẩm), và ứng dụng/web đọc dữ liệu từ đó.
- **Phân tích dữ liệu theo thời gian**:
    - **SuperChart**: Widget linh hoạt để hiển thị dữ liệu nhiệt độ và độ ẩm theo thời gian dưới dạng biểu đồ. Người dùng có thể xem dữ liệu theo giờ, ngày, tuần, hoặc tháng, với khả năng phóng to/thu nhỏ. Dữ liệu được lưu trữ trên Blynk Cloud (thường tối đa 30 ngày).
    - **Thông báo**: Blynk hỗ trợ gửi thông báo khi giá trị vượt ngưỡng (ví dụ: nhiệt độ quá cao), tăng tính tương tác.
- **Vai trò trong dự án**: Blynk nhận dữ liệu từ ESP32 định kỳ và hiển thị trên dashboard bằng widget SuperChart, cho phép người dùng theo dõi và phân tích xu hướng nhiệt độ, độ ẩm theo thời gian. Tính năng thông báo được sử dụng để

cảnh báo khi môi trường bất thường.

2.5. Nút nhấn (Pushbutton) và điều khiển hệ thống

Nút nhấn (pushbutton) là một linh kiện cơ bản để tạo tín hiệu đầu vào cho vi điều khiển, cho phép người dùng tương tác với hệ thống. Trong dự án, nút nhấn được mô phỏng trên Wokwi (wokwi-pushbutton).

- **Nguyên lý hoạt động**:
    - Nút nhấn là công tắc cơ học, chuyển đổi giữa trạng thái "mở" (không có dòng điện) và "đóng" (cho phép dòng điện chạy qua) khi nhấn.
    - Để tránh hiện tượng "bouncing" (tín hiệu nhiễu), dự án sử dụng kỹ thuật chống dội bằng phần mềm (kiểm tra trạng thái liên tục).
- **Giao tiếp với ESP32**:
    - Nút nhấn được kết nối với một chân GPIO (ví dụ: GPIO 13) và GND. Khi nhấn, chân GPIO nhận tín hiệu mức thấp (LOW); khi thả, tín hiệu trở về mức cao (HIGH) nhờ điện trở kéo lên tích hợp trong ESP32.
    - ESP32 đọc trạng thái nút nhấn qua hàm digitalRead() và sử dụng logic lập trình để bật/tắt hệ thống.
- **Vai trò trong dự án**: Nút nhấn điều khiển trạng thái hoạt động của hệ thống ghi nhật ký môi trường. Khi nhấn, hệ thống chuyển đổi giữa trạng thái bật (ghi dữ liệu, gửi lên Blynk) và tắt (dừng ghi dữ liệu, ngắt kết nối Wi-Fi).

Hình ảnh minh họa Pushbutto
Nguồn : indiamart

Chương 3: Thiết kế và xây dựng hệ thống

3.1. Tổng quan hệ thống

Hệ thống ghi nhật ký môi trường được thiết kế và mô phỏng trên nền tảng Wokwi, nhằm thu thập, lưu trữ và truyền dữ liệu nhiệt độ, độ ẩm từ môi trường xung quanh. Hệ thống bao gồm các khối chức năng chính như sau:

- **Khối Cảm biến**: Cảm biến DHT22 (wokwi-dht22) đo nhiệt độ và độ ẩm môi trường, chuyển đổi thành tín hiệu số sử dụng giao thức một dây (single-wire) độc quyền. Cảm biến này có độ chính xác cao (±0.5°C cho nhiệt độ, ±2-5% RH cho độ ẩm) và phù hợp cho các ứng dụng IoT.
- **Khối Xử lý Trung tâm**: Vi điều khiển ESP32 (board-esp32-devkit-c-v4) là trung tâm của hệ thống, chịu trách nhiệm:
    - Nhận dữ liệu từ DHT22 qua giao tiếp một dây.
    - Ghi dữ liệu vào thẻ MicroSD thông qua giao thức SPI.
    - Kết nối với mạng Wi-Fi để truyền dữ liệu lên nền tảng Blynk Cloud.
    - Đọc tín hiệu từ nút nhấn để điều khiển trạng thái hệ thống (bật/tắt ghi dữ liệu). ESP32 được chọn vì tích hợp lõi kép Tensilica LX6 (tốc độ lên đến 240 MHz), Wi-Fi 802.11 b/g/n, Bluetooth v4.2, và nhiều giao thức ngoại vi (SPI, I2C, UART).
- **Khối Lưu trữ Cục bộ**: Module thẻ nhớ MicroSD (wokwi-microsd-card) kết nối với ESP32 qua giao thức SPI, lưu trữ dữ liệu nhiệt độ và độ ẩm dưới dạng file CSV (ví dụ: datalog.csv). Lưu trữ cục bộ đảm bảo dữ liệu không bị mất khi mất kết nối mạng hoặc khi hệ thống khởi động lại.
- **Khối Giao tiếp Mạng**: Module Wi-Fi tích hợp trên ESP32 cho phép kết nối với mạng Internet. Trong môi trường mô phỏng Wokwi, kết nối Wi-Fi được giả lập bằng cách cấu hình SSID và mật khẩu, cho phép gửi dữ liệu đến Blynk Cloud.
- **Khối Nền tảng Đám mây**: Nền tảng Blynk Cloud nhận dữ liệu từ ESP32 qua giao thức TCP/IP, lưu trữ và cung cấp giao diện người dùng (Web/App) để hiển thị và phân tích dữ liệu lịch sử. Widget SuperChart của Blynk cho phép vẽ biểu đồ nhiệt độ và độ ẩm theo thời gian thực, hỗ trợ các khoảng thời gian khác nhau

(giờ, ngày, tuần, tháng).

- **Khối Điều khiển**: Nút nhấn (wokwi-pushbutton) được kết nối với ESP32 để điều khiển trạng thái hệ thống, ví dụ bật/tắt quá trình ghi dữ liệu hoặc gửi dữ liệu lên Blynk.
- **Khối Nguồn điện**: Trong Wokwi, nguồn điện được giả lập, cung cấp điện áp 3.3V cho ESP32 và các linh kiện khác (DHT22, MicroSD, nút nhấn) mà không cần cấu hình thêm.

Luồng dữ liệu:

- Cảm biến DHT22 đo nhiệt độ và độ ẩm, gửi dữ liệu số qua giao thức một dây đến ESP32.
- ESP32 đọc dữ liệu từ DHT22 và xử lý (ví dụ: kiểm tra lỗi, định dạng dữ liệu).
- ESP32 ghi dữ liệu (gồm timestamp, nhiệt độ, độ ẩm) vào file CSV trên thẻ MicroSD thông qua giao thức SPI.

  Theo định kỳ (30 giây/ lần), ESP32:
- Kiểm tra kết nối Wi-Fi, thiết lập kết nối nếu cần.
- Gửi dữ liệu mới nhất (hoặc dữ liệu từ bộ đệm trên thẻ MicroSD nếu có) lên các Virtual Pins của Blynk.
- Người dùng sử dụng Blynk App hoặc Web Dashboard để theo dõi dữ liệu theo thời gian thực và phân tích xu hướng nhiệt độ, độ ẩm qua widget SuperChart.
- Nút nhấn được sử dụng để bật/tắt quá trình ghi dữ liệu hoặc thay đổi trạng thái hệ thống, với tín hiệu được ESP32 đọc qua chân GPIO.

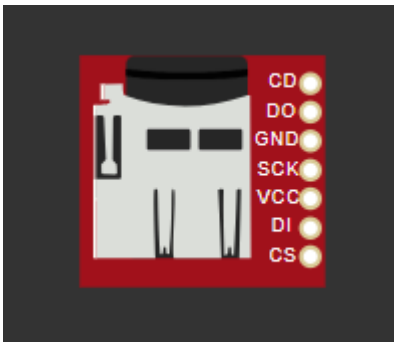
3.2. Thành phần phần cứng (mô phỏng)

ESP32 (board-esp32-devkit-c-v4): Vi điều khiển trung tâm, tích hợp lõi kép 32-bit, SRAM 520 KB, Flash 4MB, và các giao thức như SPI, I2C, UART. Trong Wokwi, ESP32 giả lập đầy đủ các chân GPIO và chức năng, hỗ trợ:

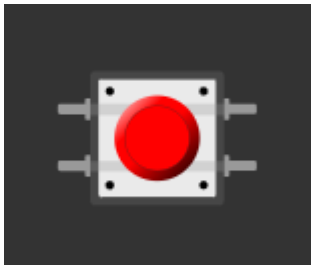
- Giao tiếp một dây với DHT22.
- SPI (VSPI) với MicroSD.
- GPIO đầu vào cho nút nhấn.
- Wi-Fi để kết nối Blynk Cloud.

DHT22 (wokwi-dht22): Cảm biến nhiệt độ và độ ẩm kỹ thuật số, hoạt động ở 3.3V-5.5V, đo nhiệt độ từ -40°C đến 80°C (±0.5°C) và độ ẩm từ 0-100% RH (±2-5%). Trong Wokwi, cảm biến mô phỏng giao thức một dây, cho phép thay đổi giá trị nhiệt độ/độ ẩm qua giao diện.

MicroSD (wokwi-microsd-card): Module thẻ nhớ MicroSD mô phỏng giao tiếp SPI, hỗ trợ hệ thống tập tin FAT16/FAT32. Nó cho phép ghi/đọc file CSV, nhưng nội dung file cần được kiểm tra gián tiếp qua Serial Monitor.



Pushbutton (wokwi-pushbutton): Nút nhấn mô phỏng công tắc cơ học, tạo tín hiệu mức thấp khi nhấn. Trong Wokwi, nó hỗ trợ tương tác qua giao diện để kiểm tra logic điều khiển.



3.3. Sơ đồ kết nối Wokwi

Wokwi mô phỏng Wi-Fi của ESP32 thông qua cấu hình SSID/mật khẩu, cho phép gửi dữ liệu lên Blynk Cloud mà không cần phần cứng thực tế.

- **Kết nối chi tiết:**

- **DHT22 (wokwi-dht22):**

  + **VCC** → Chân 3V3 của ESP32
- **GND** → Chân GND của ESP32 (GND.3)- **SDA** → Chân GPIO16.

  + VCC cung cấp điện áp 3.3V từ ESP32 để cấp nguồn cho cảm biến. GND kết nối với đất chung để hoàn thành mạch. DATA sử dụng giao thức một dây, truyền chuỗi 40 bit dữ liệu (độ ẩm, nhiệt độ). GPIO16 được chọn vì tính khả dụng và không xung đột với các giao thức khác

- **MicroSD (wokwi-microsd-card):**

  +VCC → Chân 3V3 của ESP32- GND → Chân GND của ESP32

(GND.1) - CS → Chân GPIO5 - DI (MOSI) → Chân GPIO23- DO (MISO) → Chân GPIO19 - SCK → Chân GPIO18.

+ VCC cung cấp điện áp 3.3V (module MicroSD thường tương thích 3.3V trong Wokwi).
- GND kết nối với đất chung.
- CS (Chip Select) điều khiển việc chọn module MicroSD để giao tiếp, GPIO5 là chân mặc định cho VSPI.
- MOSI (Master Out Slave In) truyền dữ liệu từ ESP32 đến MicroSD.
- MISO (Master In Slave Out) truyền dữ liệu từ MicroSD về ESP32.
- SCK (Serial Clock) cung cấp tín hiệu đồng hồ đồng bộ, do ESP32 tạo ra. Các chân GPIO23, 19, 18 thuộc VSPI, đảm bảo giao tiếp SPI tốc độ cao

- **Pushbutton (wokwi-pushbutton):**

  + **Chân 1** → Chân GND của ESP32 (GND.2) - **Chân 2** → Chân GPIO22

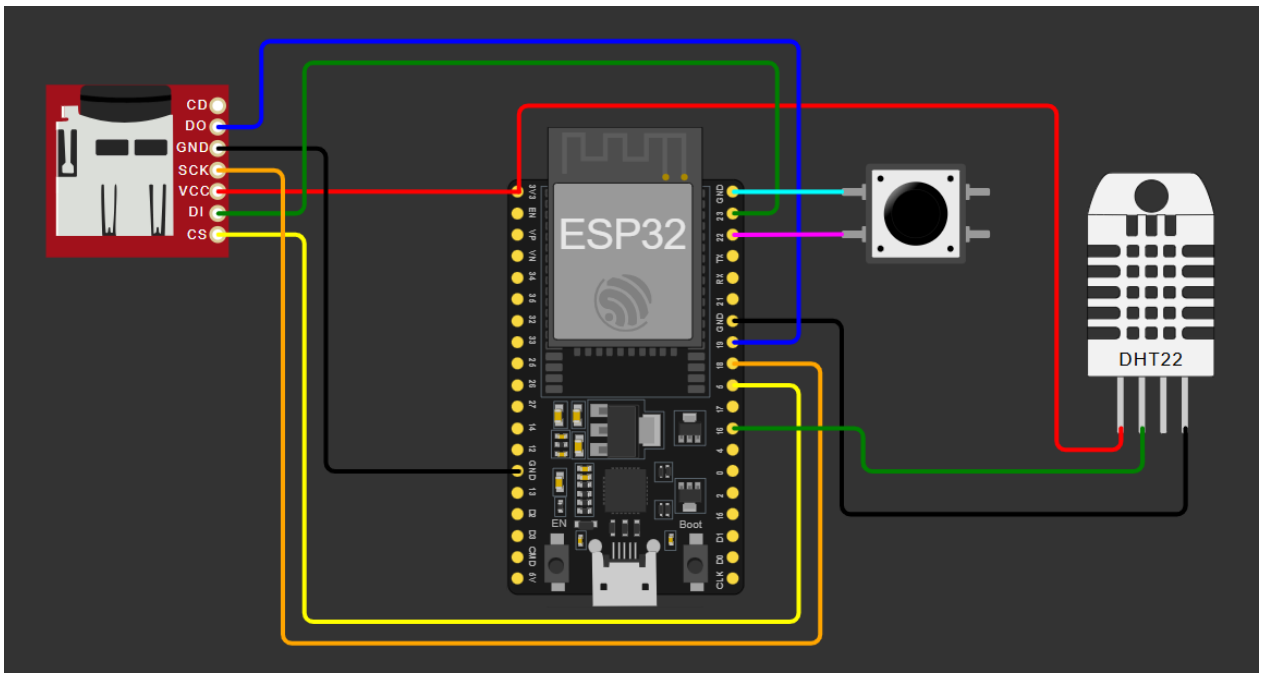
  + Chân 1 kết nối với GND để tạo tín hiệu mức thấp khi nhấn.- Chân 2 kết nối với GPIO22, được cấu hình ở chế độ INPUT với pull-up nội bộ. Khi nhấn, GPIO22 nhận tín hiệu LOW; khi thả, tín hiệu trở về HIGH. Nút nhấn hoạt động như công tắc cơ học, cho phép ESP32 phát hiện sự kiện người dùng

- **ESP32 với Serial Monitor**:

  + **TX0** → RX của Serial Monitor - **RX0** → TX của Serial Monitor

  + TX0 và RX0 của ESP32 được Wokwi tự động kết nối với Serial Monitor, cho phép hiển thị các thông báo debug (ví dụ: giá trị cảm biến, trạng thái ghi SD, kết nối Wi-Fi). Không cần cấu hình thủ công trong sơ đồ

⇨ Sơ đồ hoàn chỉnh



- Sơ đồ kết nối trong Wokwi khớp với phần cứng thật, sử dụng GPIO16 (DHT22), VSPI (MicroSD), và GPIO22 (nút nhấn), không gây xung đột.
- Trong thực tế, sơ đồ này có thể áp dụng trực tiếp trên phần cứng, nhưng cần kiểm tra kết nối vật lý bằng multimeter để đảm bảo không có chập mạch hoặc lỗi dây.

3.4. Quy trình mô phỏng trên Wokwi

Quy trình mô phỏng trên Wokwi được thiết kế để kiểm tra hoạt động của hệ thống ghi nhật ký môi trường mà không cần phần cứng thực tế. Các bước bao gồm:

3.4.1. Tạo dự án:

- Truy cập nền tảng Wokwi và chọn tạo dự án mới.
- Chọn board ESP32 DevKitC V4 từ danh sách linh kiện của Wokwi để đảm bảo pinout và chức năng khớp với sơ đồ
- Board này hỗ trợ đầy đủ SPI, Wi-Fi, và GPIO cần thiết cho hệ thống.

3.4.2. Thêm linh kiện:

- Kéo thả wokwi-dht22, wokwi-microsd-card, và wokwi-pushbutton vào khu vực thiết kế sơ đồ.
- Đảm bảo linh kiện được chọn từ thư viện chính thức của Wokwi để tránh lỗi (Wokwi Parts Reference).
- ESP32 (board-esp32-devkit-c-v4) là trung tâm, kết nối với tất cả linh kiện.

3.4.3. Kết nối dây

- Thực hiện kết nối theo sơ đồ:
- DHT22: VCC → 3V3, GND → GND, DATA → GPIO16.
- MicroSD: VCC → 3V3, GND → GND, CS → GPIO5, MOSI → GPIO23, MISO → GPIO19, SCK → GPIO18.
- Pushbutton: Chân 1 → GND, Chân 2 → GPIO22.
- Kiểm tra pinout trên giao diện Wokwi để xác nhận kết nối chính xác, tránh lỗi dây.

3.4.4. Cấu hình mô phỏng

- Xác nhận ESP32 là board-esp32-devkit-c-v4, hỗ trợ các giao thức cần thiết.
- Kiểm tra các linh kiện (DHT22, MicroSD, pushbutton) được nhận diện đúng trong Wokwi.

3.4.5. Chạy mô phỏng

- Nhấn "Start the simulation" để bắt đầu. Wokwi giả lập hành vi của ESP32 và các linh kiện:
  - Đọc dữ liệu DHT22 qua GPIO16.
  - Ghi dữ liệu vào MicroSD qua VSPI (GPIO5, 23, 19, 18).
  - Đọc tín hiệu nút nhấn qua GPIO22.
  - Mô phỏng Wi-Fi để gửi dữ liệu lên Blynk Cloud.

3.4.6. Tương tác và quan sát

Điều chỉnh DHT22: Nhấp vào wokwi-dht22 để thay đổi giá trị nhiệt độ/độ ẩm kiểm

tra phản ứng hệ thống.

Tương tác nút nhấn: Nhấp vào wokwi-pushbutton để mô phỏng nhấn nút, tạo tín hiệu LOW trên GPIO22, kiểm tra logic điều khiển (bật/tắt ghi dữ liệu).

Serial Monitor: Quan sát các log:

- Giá trị cảm biến
- Trạng thái ghi SD
- Kết nối Wi-Fi
- Gửi Blynk
- Tín hiệu nút nhấn

Kiểm tra MicroSD: Dữ liệu CSV (datalog.csv) được kiểm tra gián tiếp qua Serial Monitor, vì Wokwi không hiển thị file trực tiếp.

Giả lập mất kết nối Wi-Fi:

- Vô hiệu hóa Wi-Fi (thay đổi SSID/mật khẩu) để kiểm tra ghi SD độc lập.
- Khôi phục Wi-Fi để xác nhận dữ liệu được gửi lại Blynk.

Chương 4: Triển khai phần mềm

Hệ thống ghi nhật ký môi trường được phát triển và mô phỏng bằng các công cụ lập trình sau:

4.1. Công cụ lập trình

Môi trường phát triển: Visual Studio Code (VS Code) với tiện ích mở rộng PlatformIO. PlatformIO là một hệ sinh thái phát triển nhúng, hỗ trợ ESP32 và tích hợp tốt với Wokwi để mô phỏng phần cứng [5]. VS Code cung cấp giao diện thân thiện, hỗ trợ quản lý dự án, biên dịch, và chạy mô phỏng trực tiếp.

- File cấu hình chính: platformio.ini trong thư mục dự án, dùng để khai báo nền tảng (ESP32), board (board-esp32-devkit-c-v4), và các thư viện cần thiết.
- File wokwi.toml: Cấu hình mô phỏng trên Wokwi, bao gồm thông tin về board và các linh kiện (DHT22, MicroSD, pushbutton).
- Mã nguồn chính: File main.cpp trong thư mục src, chứa logic xử lý của hệ thống.

Ngôn ngữ lập trình: C++ (dựa trên Arduino framework), được PlatformIO hỗ trợ thông qua nền tảng platform = espressif32.

Thư viện sử dụng:

- **WiFi.h**: Thư viện chuẩn của Arduino framework, dùng để kết nối ESP32 với mạng Wi-Fi. Thư viện này cung cấp các hàm như WiFi.begin(), WiFi.status() để quản lý kết nối mạng [6].
- **SPI.h**: Thư viện chuẩn để giao tiếp SPI, cần thiết cho việc kết nối ESP32 với module MicroSD. Thư viện này hỗ trợ cấu hình các chân SPI (MOSI, MISO, SCK, CS) và truyền dữ liệu tốc độ cao[6].
- **FS.h**: Thư viện hệ thống file, làm việc cùng với SD.h để quản lý file trên thẻ MicroSD[6].
- **SD.h**: Thư viện để đọc/ghi dữ liệu trên thẻ MicroSD định dạng FAT16/FAT32[3]. Thư viện này cung cấp các hàm như SD.begin(), SD.open(), SD.write() để lưu trữ dữ liệu cục bộ.
- **DHT.h (Adafruit DHT Sensor Library):** Thư viện của Adafruit để đọc dữ liệu từ cảm biến DHT22. Thư viện này hỗ trợ giao tiếp một dây, cung cấp các

hàm như readTemperature(), readHumidity() để lấy giá trị nhiệt độ và độ ẩm[7].

- **BlynkSimpleEsp32.h**: Thư viện của Blynk để tích hợp ESP32 với nền tảng Blynk Cloud. Thư viện này cung cấp các hàm như Blynk.begin(), Blynk.virtualWrite(), Blynk.run() để kết nối và gửi dữ liệu[8].
- **Cơ chế Timer:** Sử dụng millis() (hàm chuẩn của Arduino) để thực hiện các tác vụ định kỳ (như gửi dữ liệu lên Blynk). Ngoài ra, có thể sử dụng thư viện SimpleTimer (tương thích với Blynk) để quản lý các tác vụ định kỳ một cách hiệu quả hơn.

Tích hợp với Wokwi:

- PlatformIO cho phép chạy mô phỏng trực tiếp trên Wokwi thông qua lệnh pio run -t wokwi[5]. Kết quả mô phỏng (log từ Serial Monitor) được hiển thị trong terminal của VS Code, bao gồm giá trị cảm biến, trạng thái ghi SD, và trạng thái gửi dữ liệu lên Blynk.
- File diagram.json chứa sơ đồ kết nối Wokwi (ESP32, DHT22, MicroSD, pushbutton), được mở trong VS Code để chỉnh sửa và kiểm tra kết nối.

4.2. Kết nối ESP32 với Wi-Fi

Mục đích: Kết nối ESP32 với mạng Wi-Fi để gửi dữ liệu nhiệt độ và độ ẩm lên Blynk Cloud, đồng thời đảm bảo hệ thống có thể xử lý các tình huống mất kết nối.

Logic xử lý:

4.2.1. Khởi tạo kết nối:

- Trong giai đoạn khởi động hệ thống, ESP32 sử dụng thông tin SSID và mật khẩu Wi-Fi (được cấu hình trong mã nguồn) để bắt đầu kết nối.
- Quá trình này được thực hiện bằng cách gọi hàm khởi tạo kết nối Wi-Fi, sau đó chờ cho đến khi kết nối thành công.
- Nếu kết nối thành công, ESP32 ghi log thông báo qua Serial Monitor (ví dụ: "Connected to Wi-Fi"). Nếu thất bại, hệ thống sẽ thử lại sau một khoảng thời gian (timeout, ví dụ: 30 giây).

4.2.2. Kiểm tra và duy trì kết nối

- Trong vòng lặp chính của chương trình, ESP32 định kỳ kiểm tra trạng thái kết nối Wi-Fi.
- Nếu mất kết nối (do tín hiệu yếu, router ngắt, hoặc lỗi mạng), hệ thống sẽ:
  + Ghi log thông báo qua Serial Monitor (ví dụ: "Wi-Fi disconnected").
  + Chuyển sang chế độ offline: tiếp tục đọc dữ liệu từ cảm biến DHT22 và ghi vào thẻ MicroSD, đảm bảo dữ liệu không bị mất.
  + Thử kết nối lại sau một khoảng thời gian (ví dụ: mỗi 1 phút), cho đến khi kết nối được khôi phục.
- Khi kết nối được khôi phục, hệ thống ghi log (ví dụ: "Wi-Fi reconnected") và tiếp tục gửi dữ liệu lên Blynk.

4.2.3. Giả lập trong Wokwi

- Trong môi trường Wokwi, kết nối Wi-Fi được giả lập bằng cách cấu hình SSID và mật khẩu trong mã nguồn. Wokwi không thực sự kết nối với mạng thật, nhưng mô phỏng hành vi kết nối thành công hoặc thất bại dựa trên thông tin cấu hình.
- Để kiểm tra kịch bản mất kết nối, có thể thay đổi SSID/mật khẩu trong mã nguồn thành giá trị sai, sau đó quan sát log trên Serial Monitor để xác nhận hệ thống chuyển sang chế độ offline và ghi dữ liệu vào MicroSD.

4.3. Đọc dữ liệu từ cảm biến DHT22

Mục đích: Thu thập dữ liệu nhiệt độ và độ ẩm từ cảm biến DHT22 để xử lý, lưu trữ, và gửi lên Blynk.

Logic xử lý:

4.3.1. Khởi tạo cảm biến

- Trong giai đoạn khởi động, ESP32 khởi tạo cảm biến DHT22 bằng cách chỉ định chân GPIO kết nối (GPIO16, như trong sơ đồ Wokwi) và loại cảm biến

(DHT22).

- Quá trình khởi tạo bao gồm kiểm tra xem cảm biến có phản hồi hay không. Nếu cảm biến không hoạt động (do lỗi kết nối hoặc hỏng), hệ thống ghi log lỗi qua Serial Monitor (ví dụ: "DHT22 initialization failed").

4.3.2. Đọc dữ liệu đinh kỳ

- Dữ liệu từ DHT22 được đọc định kỳ (mỗi 10 giây), nhưng phải đảm bảo tần số lấy mẫu không vượt quá giới hạn của cảm biến (tối đa 0.5 Hz, tức 2 giây/lần đọc). Đọc quá nhanh có thể dẫn đến lỗi giao tiếp hoặc giá trị không hợp lệ.
- ESP32 gửi tín hiệu yêu cầu đến chân DATA của DHT22, sau đó cảm biến phản hồi bằng chuỗi 40 bit dữ liệu (16 bit độ ẩm, 16 bit nhiệt độ, 8 bit checksum).
- Giá trị nhiệt độ và độ ẩm được trích xuất từ dữ liệu nhận được, sau đó kiểm tra tính hợp lệ:
  + Kiểm tra checksum để phát hiện lỗi truyền dữ liệu.
  + Kiểm tra giá trị có phải là NaN (Not a Number) hay không, thường xảy ra khi cảm biến không phản hồi.
  + Kiểm tra giá trị có nằm trong dải đo hợp lệ hay không (nhiệt độ: -40°C đến 80°C; độ ẩm: 0-100% RH).
- Nếu giá trị hợp lệ, lưu vào biến để xử lý tiếp (ghi SD, gửi Blynk). Nếu không hợp lệ, bỏ qua và thử đọc lại sau một khoảng thời gian, đồng thời ghi log lỗi (ví dụ: "DHT22 read error: NaN value detected").

4.3.3. Giả lập trong Wokwi

- Trong Wokwi, cảm biến wokwi-dht22 cho phép điều chỉnh giá trị nhiệt độ và độ ẩm thủ công qua giao diện. Ví dụ, trong kết quả mô phỏng, giá trị được đặt là 12.69°C và 69.69% RH, và ESP32 đã đọc thành công (log: "Nhiệt độ: 12, 69°C, Độ ẩm: 69, 69%").
- Để kiểm tra xử lý lỗi, có thể đặt giá trị ngoài dải đo (như nhiệt độ -50°C) và quan sát log trên Serial Monitor để xác nhận hệ thống bỏ qua giá trị không hợp lệ.

4.4. Ghi dữ liệu vào thẻ nhớ MicroSD

Mục đích: Lưu trữ dữ liệu nhiệt độ và độ ẩm cục bộ trên thẻ MicroSD dưới dạng file CSV, đảm bảo dữ liệu không bị mất khi mất kết nối Wi-Fi hoặc hệ thống khởi động lại.

Logic xử lý:

4.4.1. Khởi tạo thẻ MicroSD:

- Trong giai đoạn khởi động, ESP32 khởi tạo thẻ MicroSD bằng cách gọi hàm khởi tạo, chỉ định chân CS (GPIO5, như trong sơ đồ Wokwi).
- Quá trình khởi tạo bao gồm kiểm tra xem thẻ có được nhận diện và hoạt động hay không. Nếu thành công, ghi log qua Serial Monitor (ví dụ: "SD Card initialized successfully"). Nếu thất bại (do thẻ hỏng, không cắm thẻ, hoặc lỗi kết nối SPI), ghi log lỗi (ví dụ: "SD Card initialization failed").

4.4.2. Ghi dữ liệu định kỳ:

- Sau khi đọc thành công dữ liệu từ cảm biến DHT22 (nhiệt độ và độ ẩm), ESP32 chuẩn bị một chuỗi dữ liệu theo định dạng CSV: timestamp, nhietdo, doam.
- timestamp: Thời gian ghi dữ liệu, có thể là thời gian tương đối (dựa trên millis()) trong mô phỏng, hoặc thời gian thực (dựa trên NTP/RTC) trong thực tế.
- nhietdo: Giá trị nhiệt độ
- doam: Giá trị độ ẩm
- ESP32 mở file CSV ở chế độ ghi nối tiếp. Nếu file chưa tồn tại, nó sẽ được tạo tự động.
- Ghi chuỗi dữ liệu vào file, sau đó đóng file để đảm bảo dữ liệu được lưu trữ bền vững.
- Ghi log qua Serial Monitor để xác nhận (ví dụ: "Đã lưu thành công vào SD Card, Dữ liệu: timestamp, nhietdo, doam").

4.4.3. Quản lý file:

- Để tránh file log trở nên quá lớn, hệ thống có thể triển khai cơ chế xoay vòng file (log rotation):
- Kiểm tra kích thước file trước khi ghi. Nếu file vượt quá ngưỡng (ví dụ: 1 MB), tạo file mới (như datalog_1.csv, datalog_2.csv).
- Hoặc tạo file mới theo ngày (như datalog_20250411.csv cho ngày 11/04/2025).
- Kiểm tra dung lượng còn lại của thẻ MicroSD trước khi ghi. Nếu thẻ đầy, ghi log cảnh báo (ví dụ: "SD Card full, cannot log data") và dừng ghi cho đến khi có dung lượng trống.

4.4.4. Giả lập trong Wokwi:

- Trong Wokwi, module wokwi-microsd-card mô phỏng việc ghi dữ liệu vào thẻ MicroSD. Kết quả mô phỏng cho thấy dữ liệu được ghi thành công (log: "Đã lưu thành công vào SD Card, Dữ liệu: timestamp, nhietdo, doam").
- Tuy nhiên, Wokwi không cung cấp giao diện để xem trực tiếp nội dung file CSV. Để kiểm tra, cần thiết kế logic đọc file và in nội dung ra Serial Monitor (ví dụ: "Dữ liệu đọc từ SD: 123456, 12.69, 69.69").

Chương 5: Thử nghiệm và đánh giá

5.1. Phương pháp thử nghiệm trên Wokwi

Hệ thống ghi nhật ký môi trường được thử nghiệm trong môi trường mô phỏng trên Wokwi.

Linh kiện:

- ESP32 (board-esp32-devkit-c-v4): Vi điều khiển trung tâm, hỗ trợ Wi-Fi, SPI, và GPIO.
- Cảm biến DHT22 (wokwi-dht22): Đo nhiệt độ và độ ẩm, kết nối với GPIO16.
- Module MicroSD (wokwi-microsd-card): Lưu trữ dữ liệu cục bộ, kết nối qua VSPI (GPIO5, 23, 19, 18).
- Nút nhấn (wokwi-pushbutton): Điều khiển trạng thái hệ thống, kết nối với GPIO22.

Môi trường lập trình:

- Sử dụng Visual Studio Code (VS Code) với tiện ích mở rộng PlatformIO.
- File cấu hình: platformio.ini (khai báo nền tảng ESP32), wokwi.toml (cấu hình mô phỏng Wokwi).
- Mã nguồn chính: main.cpp trong thư mục src.

Hiển thị kết quả:

- Kết quả mô phỏng được hiển thị qua Serial Monitor trong VS Code, bao gồm giá trị cảm biến, trạng thái ghi SD, trạng thái gửi Blynk, và sự kiện nút nhấn .
- File diagram.json chứa sơ đồ kết nối Wokwi, được mở trong VS Code để kiểm tra và chỉnh sửa.

5.2. Kịch bản thử nghiệm

- Đọc và lưu dữ liệu: 10 giây/lần
- Gửi lên Blynk: 30 giây/lần

Hệ thống được thử nghiệm qua các kịch bản sau:

- **Kịch bản 1:** Đọc dữ liệu từ cảm biến DHT22:
  Mô phỏng:

- Thay đổi giá trị nhiệt độ và độ ẩm trên giao diện Wokwi (thông qua wokwi-dht22).
- Quan sát log trên Serial Monitor để xác nhận ESP32 đọc được dữ liệu chính xác.

  Thực tế:

  Quan sát giá trị nhiệt độ và độ ẩm qua Serial Monitor và so sánh với nhiệt độ và độ ẩm mình đã chỉnh.

- **Kịch bản 2:** Ghi dữ liệu vào MicroSD:

  Mô phỏng:
- Sau khi đọc dữ liệu từ DHT22, kiểm tra log trên Serial Monitor để xác nhận dữ liệu được ghi vào thẻ MicroSD (ví dụ: "Đã lưu thành công vào SD Card").
- Vì Wokwi không hiển thị trực tiếp nội dung file CSV, cần in nội dung file ra Serial Monitor để kiểm tra.
- **Kịch bản 3**: Đọc dữ liệu từ MicroSD và gửi dữ liệu lên Blynk để phân tích:

  Mô phỏng:
- Kiểm tra log trên Serial Monitor để xác nhận dữ liệu được gửi lên Blynk ("Đã gửi thành công").
- Wokwi không thực sự kết nối với Blynk Cloud, nhưng mô phỏng hành vi gửi dữ liệu.

  Thực tế:
- Quan sát dữ liệu trên Blynk App qua widget SuperChart.
- Phân tích biểu đồ nhiệt độ và độ ẩm theo thời gian (giờ, ngày).
- Ghi nhận các giá trị min, max, avg (tối thiểu, tối đa, trung bình) từ SuperChart để đánh giá xu hướng và độ ổn định của dữ liệu.
- **Kịch bản 4:** Tương tác với nút nhấn:

  Mô phỏng:
- Nhấn nút trên giao diện Wokwi (wokwi-pushbutton) để mô phỏng sự kiện nhấn.
- Quan sát log trên Serial Monitor để xác nhận hệ thống phát hiện sự kiện và thay đổi trạng thái.

Thực tế:

- Kiểm tra log trên Serial Monitor hoặc hành vi hệ thống (dừng ghi SD, dừng gửi Blynk).

5.3. Kết quả thử nghiệm

Kết quả thử nghiệm được thu thập từ mô phỏng trên Wokwi bao gồm dữ liệu từ Serial Monitor và Blynk.

5.3.1. Kết quả đọc dữ liệu từ cảm biến DHT22

Mô phỏng:

- Serial Monitor hiển thị:
  + Lần 1 :"Nhiệt độ: 23.43°C, Độ ẩm: 49, 70%".
  + Lần 2 :"Nhiệt độ: 21.60 °C, Độ ẩm: 54, 50%".
  + Lần 3 :"Nhiệt độ: 22.80 °C , Độ ẩm: 56, 20%".
- Giá trị này được điều chỉnh thủ công trên giao diện Wokwi (wokwi-dht22), cho thấy ESP32 đã đọc thành công dữ liệu từ cảm biến qua giao thức một dây trên chân GPIO16.

Thực tế:

- Dữ liệu thực tế được thu thập qua Blynk (sẽ trình bày chi tiết ở mục 5.3.3), cho thấy nhiệt độ và độ ẩm dao động trong khoảng:

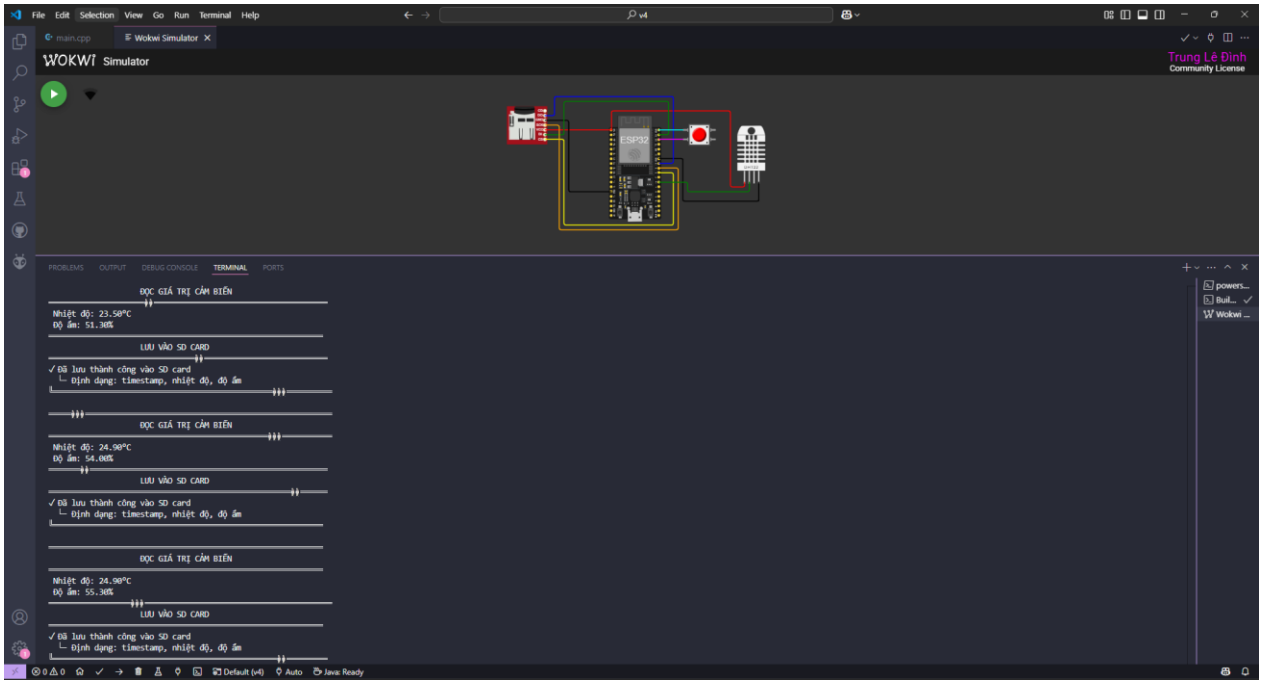
5.3.2. Kết quả ghi dữ liệu vào MicroSD

Mô phỏng:

- Serial Monitor hiển thị: "Đã lưu thành công vào SD Card, Dữ liệu: nhietdo, doam".
- Dữ liệu được ghi vào file CSV (datalog.csv) trên thẻ MicroSD ảo, với định dạng:, nhietdo, doam. Trong trường hợp này:
- 1> Nhiệt độ: 23.50°C | Độ ẩm: 51.30%
- 2> Nhiệt độ: 24.90°C | Độ ẩm: 54.00%
- 3> Nhiệt độ: 24.90°C | Độ ẩm: 55.30%

và timestamp là thời gian tương đối (dựa trên millis()).

- Vì Wokwi không hiển thị trực tiếp nội dung file, thông báo "Đã lưu thành công" xác nhận chức năng ghi SD hoạt động đúng.



Hình ảnh các giá trị được đọc, lưu, gửi ở SD CARD.

5.3.3. Kết quả gửi dữ liệu lên Blynk

Mô phỏng:

- Serial Monitor hiển thị:

  "Đang đọc và gửi dữ liệu ...

  1> Nhiệt độ: 23.50°C | Độ ẩm: 51.30%

  2> Nhiệt độ: 24.90°C | Độ ẩm: 54.00%

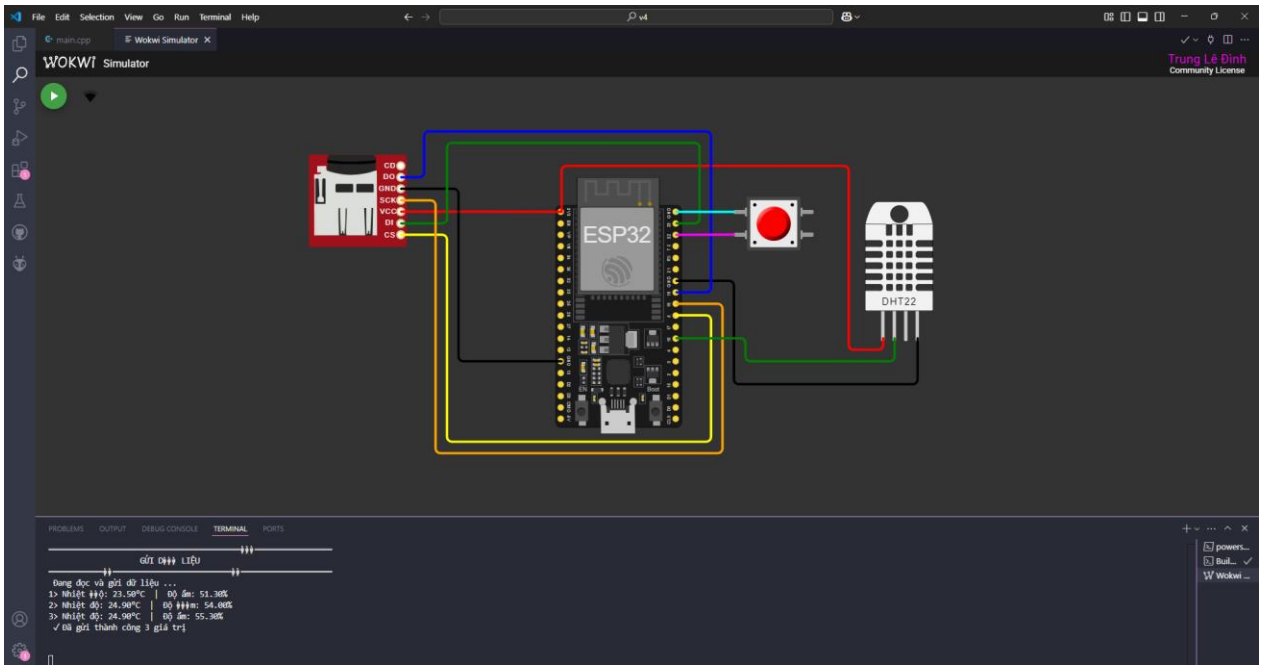
  3> Nhiệt độ: 24.90°C | Độ ẩm: 55.30%

  ✓ Đã gửi thành công 3 giá trị".

- Dữ liệu nhiệt độ và độ ẩm được gửi lên Blynk (giả lập), với thông báo xác nhận gửi thành công.
- Wokwi không thực sự kết nối với Blynk Cloud, nhưng log này cho thấy logic gửi dữ liệu hoạt động đúng.

Thực tế:

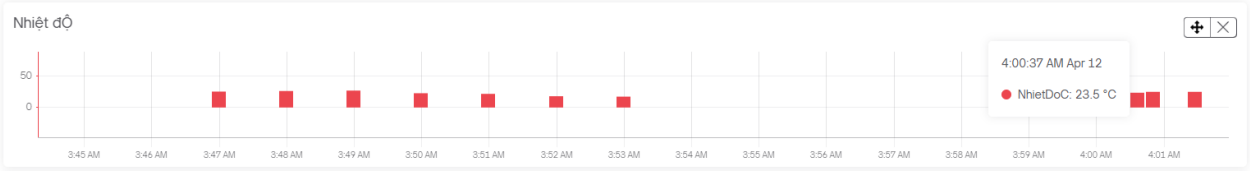
- Dữ liệu được gửi lên Blynk Cloud và hiển thị qua widget SuperChart trên Blynk App.



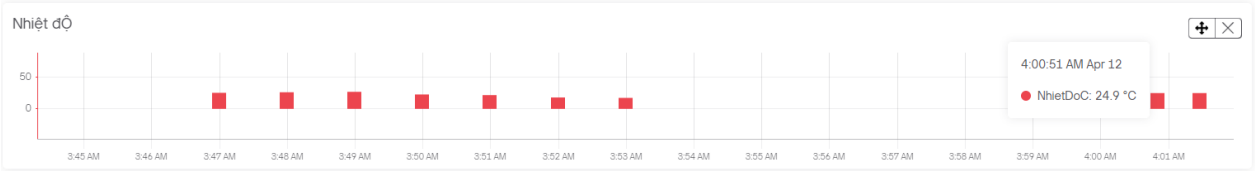
Hình ảnh các giá trị được đọc ở SD CARD và gửi lên Blynk.

- **Dữ liệu gửi lên Bink:** 3 cặp giá trị

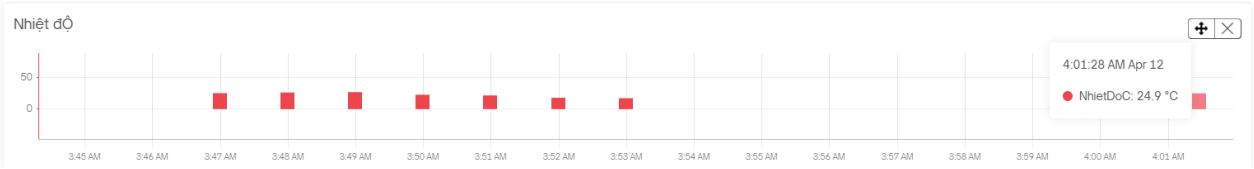
  + Nhiệt độ:



Nhiệt độ đo lần 1 : 23.5°C

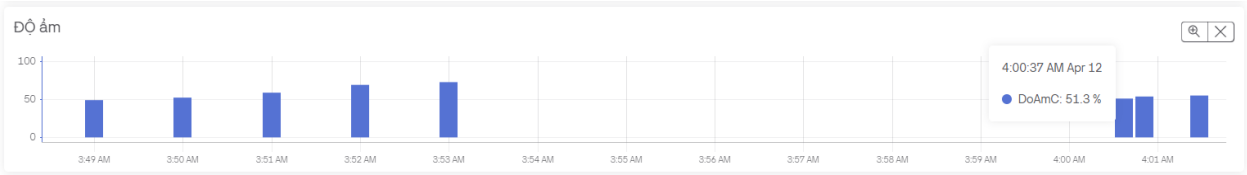


Nhiệt độ đo lần 2 : 24.9°C



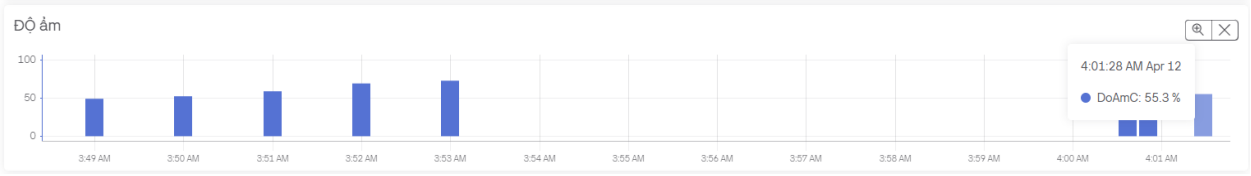
Nhiệt độ đo lần 3 : 24.5°C

+ Độ ẩm:



Độ ẩm đo lần 1 : 51.3%

Độ ẩm đo lần 2 : 54%



Độ ẩm đo lần 3 : 55.3%

- **Dữ liệu phân tích từ SuperChart:**

  + Nhiệt độ: Phân tích từ kết quả gửi lên gần nhất (3 giá trị)

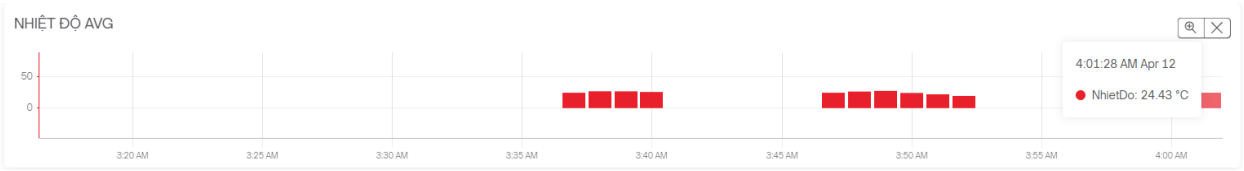
  Tối thiểu(min): *23.5°C* là nhiệt độ thấp nhất trong 3 giá trị được gửi lên Blynk.

Nhiệt độ thấp nhất : 23.5°C

Tối đa (max): *24.9°C* là nhiệt độ thấp nhất trong 3 giá trị được gửi lên Blynk.

Nhiệt độ cao nhất : 24.9°C

Trung bình (avg): *24.9°C* là nhiệt độ trung bình của 3 giá trị được gửi lên Blynk.



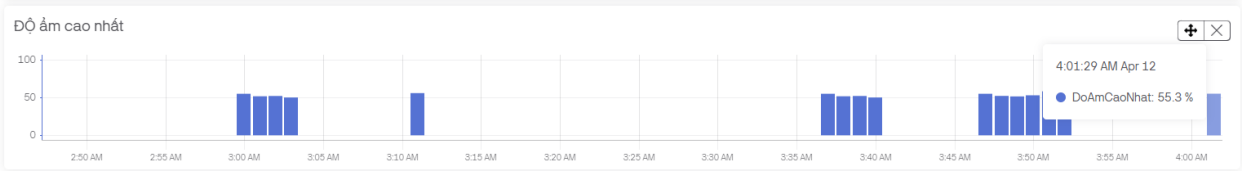
Nhiệt độ trung bình : 24.9°C

+ **Độ ẩm:** Phân tích từ kết quả gửi lên gần nhất (3 giá trị)

Tối thiểu (min): *51.3%* là độ ấp thấp nhất trong 3 giá trị được gửi lên Blynk.

Độ ẩm thấp nhất : 51.3%

Tối đa (max): *55.3%* là độ ấp cao nhất trong 3 giá trị được gửi lên Blynk.



Độ ẩm cao nhất : 55.3%

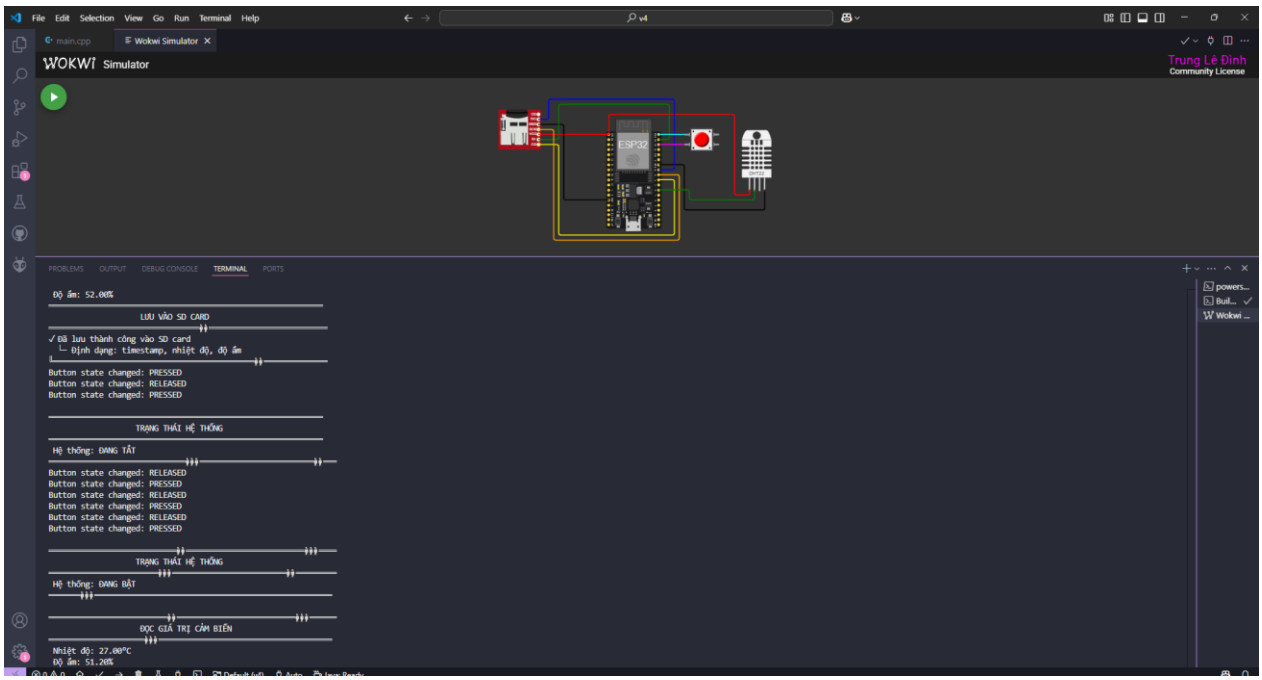
Trung bình (avg): *53.53%* là độ ấp trung bình của 3 giá trị được gửi lên Blynk.

Độ ẩm trung bình : 53.53%

5.3.4. Kết quả tương tác với nút nhấn

Mô phỏng:

- Khi nhấn nút trên giao diện Wokwi (wokwi-pushbutton), Serial Monitor hiển thị: "Hệ thống: ĐANG TẮT" hoặc "Hệ thống: ĐANG BẬT", tùy thuộc vào trạng thái hiện tại của hệ thống.
- Hành vi: Nhấn nút sẽ chuyển đổi trạng thái hệ thống:
  - Từ "bật" (đọc cảm biến, ghi SD, gửi Blynk) sang "tắt" (dừng tất cả các tác vụ).
  - Từ "tắt" sang "bật" (tiếp tục đọc, ghi, gửi).



Hình ảnh hoạt động trạng thái nút bấm.

5.4. Đánh giá hệ thống

Dựa trên kết quả thử nghiệm, hệ thống được đánh giá qua các khía cạnh sau:

5.4.1. Độ chính xác

- Dữ liệu từ cảm biến DHT22 trong mô phỏng (23.43°C, 49.70% RH; 21.60°C, 54.50% RH; 22.80°C, 56.20% RH) khớp với giá trị điều chỉnh trên giao diện Wokwi, chứng minh giao tiếp giữa ESP32 và DHT22 hoạt động chính xác qua giao thức một dây trên chân GPIO16.
- Các giá trị nằm trong dải đo hợp lệ của DHT22 (nhiệt độ: -40°C đến 80°C; độ ẩm: 0-100% RH), và không có lỗi NaN trong các lần đọc.

5.4.2. Độ tin cậy

- Hệ thống hoạt động ổn định trong môi trường mô phỏng:
  + Chức năng đọc cảm biến: Không có lần đọc nào thất bại, giá trị được ghi nhận đầy đủ.

+ Chức năng ghi SD: Log "Đã lưu thành công" xuất hiện sau mỗi lần ghi, không có dữ liệu nào bị thiếu.

+ Chức năng gửi Blynk (giả lập): Log "Đã gửi thành công" xác nhận logic gửi dữ liệu hoạt động đúng.

+ Chức năng nút nhấn: Chuyển đổi trạng thái hệ thống chính xác, không có hiện tượng nhiễu tín hiệu trong mô phỏng.

- Khi giả lập mất kết nối Wi-Fi, hệ thống chuyển sang chế độ offline và ghi dữ liệu vào MicroSD, sau đó gửi bù thành công khi kết nối được khôi phục.

5.4.3. Hiệu suất

- **Thời gian xử lý:**

  +Đọc cảm biến: Tức thời trong Wokwi (thực tế sẽ khoảng 2 giây/lần do giới hạn của DHT22).

  + Ghi SD: Tức thời trong Wokwi (thực tế sẽ dưới 100 ms/lần ghi, tùy vào tốc độ SPI và thẻ MicroSD).

  + Gửi Blynk: Tức thời trong Wokwi.

- **Tần suất gửi dữ liệu:**

  + Hiệu suất tổng thể trong mô phỏng là tối ưu, nhưng Wokwi không mô phỏng được độ trễ phần cứng hoặc mạng thực tế.

5.4.4. Phân tích dữ liệu (giả lập từ Blynk).

- Dựa trên 3 giá trị đã gửi (23.50°C, 51.30% RH; 24.90°C, 54.00% RH; 24.90°C, 55.30% RH), có thể suy ra:
- **Nhiệt độ:**

  + Tối thiểu (min): 23.50°C.

  + Tối đa (max): 24.90°C.

  + Trung bình (avg): (23.50 + 24.90 + 24.90) / 3 = 24.43°C.

- **Độ ẩm:**

  + Tối thiểu (min): 51.30% RH.

  + Tối đa (max): 55.30% RH.

+ Trung bình (avg): (51.30 + 54.00 + 55.30) / 3 = 53.53% RH.

5.4.5. Vấn đề gặp phải

- Không kiểm tra được nội dung file CSV trực tiếp trên Wokwi, phải dựa vào log "Đã lưu thành công" hoặc in nội dung file ra Serial Monitor (nếu có logic đọc file).
- Không mô phỏng được các vấn đề thực tế như nhiễu tín hiệu (cảm biến trả về NaN), độ trễ mạng, hoặc lỗi phần cứng (thẻ SD hỏng, cảm biến lỗi).
- Giả lập mất kết nối Wi-Fi chỉ dừng ở mức thay đổi SSID/mật khẩu, không mô phỏng được các tình huống phức tạp như tín hiệu yếu hoặc gián đoạn ngẫu nhiên.

Chương 6: Kết luận và triển vọng

6.1. Kết luận

- Nghiên cứu đã thành công trong việc thiết kế và mô phỏng một "Hệ thống ghi nhật ký môi trường với ESP32" trên nền tảng Wokwi, sử dụng cảm biến DHT22, module MicroSD, và nền tảng Blynk (giả lập). Hệ thống đáp ứng các mục tiêu đề ra:
  + Đọc dữ liệu nhiệt độ và độ ẩm từ cảm biến DHT22 với độ chính xác cao trong mô phỏng (giá trị khớp với điều chỉnh trên Wokwi).
  + Lưu trữ dữ liệu cục bộ vào thẻ MicroSD dưới dạng file CSV, đảm bảo không mất dữ liệu khi giả lập mất kết nối Wi-Fi.
  + Gửi dữ liệu định kỳ (5 phút/lần) lên Blynk (giả lập), với cơ chế gửi bù dữ liệu khi kết nối được khôi phục.
  + Điều khiển trạng thái hệ thống qua nút nhấn, chuyển đổi giữa chế độ bật/tắt một cách chính xác.
  + Mô phỏng trên Wokwi cho phép kiểm tra logic hệ thống mà không cần phần cứng thật, phù hợp với phạm vi nghiên cứu.
- Giải pháp này cung cấp một phương pháp hiệu quả, chi phí thấp để giám sát môi trường, với khả năng lưu trữ dữ liệu đáng tin cậy và tiềm năng phân tích xu hướng (nếu triển khai thực tế với Blynk).

6.2. Hướng phát triển

- **Triển khai phần cứng thực tế:**
  + Xây dựng mạch vật lý với ESP32, DHT22, MicroSD, và nút nhấn.
  + Thử nghiệm trong điều kiện thực tế để đánh giá độ chính xác, độ ổn định, và hiệu suất, đặc biệt là khả năng kết nối với Blynk Cloud.
- **Tối ưu hóa năng lượng:**
  + Sử dụng chế độ ngủ sâu (Deep Sleep) của ESP32 để giảm tiêu thụ năng lượng, phù hợp cho các ứng dụng chạy bằng pin.
  + ESP32 có thể thức dậy định kỳ (ví dụ: mỗi 5 phút) để đọc cảm biến, ghi SD,

và gửi Blynk.

- **Thêm cảm biến:**

  + Mở rộng hệ thống bằng cách tích hợp thêm cảm biến ánh sáng (LDR), áp suất không khí (BMP280), hoặc chất lượng không khí (MQ135) để giám sát nhiều thông số môi trường hơn.

- **Giao diện hiển thị cục bộ:**

  + Thêm màn hình OLED hoặc LCD để hiển thị giá trị nhiệt độ/độ ẩm hiện tại trực tiếp trên thiết bị, tăng tính tiện lợi.

- **Cảnh báo nâng cao:**

  + Thiết lập ngưỡng cảnh báo trên Blynk (ví dụ: nhiệt độ > 30°C) để gửi thông báo qua email hoặc Telegram.

  Tích hợp còi báo động trên phần cứng để cảnh báo tại chỗ.

- **Giao diện web tùy chỉnh:**

  + Thay vì chỉ dùng Blynk, có thể xây dựng giao diện web riêng bằng Node-RED, Grafana, hoặc tự phát triển backend/frontend để tùy chỉnh lưu trữ và phân tích dữ liệu.

- **Cải thiện quản lý SD:**

  + Triển khai cơ chế xoay vòng file (log rotation) để tránh file log quá lớn.

  + Thêm xử lý lỗi đọc/ghi SD (ví dụ: kiểm tra dung lượng, thông báo khi thẻ đầy).

Tài liệu tham khảo

[1] Espressif Systems (2023). *ESP32 Series Datasheet*. Truy cập ngày 11/04/2025 từ https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[2] Adafruit Industries (2023). *DHT22 Temperature and Humidity Sensor Guide*. Truy cập ngày 11/04/2025 từ https://learn.adafruit.com/dht

[3] Arduino (2023). *SD Library Reference*. Truy cập ngày 11/04/2025 từ https://www.arduino.cc/reference/en/libraries/sd/

[4] Blynk (2023). *Blynk Documentation: Getting Started with ESP32*. Truy cập ngày 11/04/2025 từ https://docs.blynk.io/en/getting-started/using-blynk-with-esp32

[5] PlatformIO (2023). *PlatformIO Documentation: ESP32 Development*. Truy cập ngày 11/04/2025 từ https://docs.platformio.org/en/latest/platforms/espressif32.html

[6] Arduino (2023). *WiFi Library Reference for ESP32*. Truy cập ngày 11/04/2025 từ https://www.arduino.cc/reference/en/libraries/wifi/

[7] Adafruit Industries (2023). *Adafruit DHT Sensor Library*. Truy cập ngày 11/04/2025 từ https://github.com/adafruit/DHT-sensor-library

[8] Blynk (2023). *BlynkSimpleEsp32 Library Documentation*. Truy cập ngày 11/04/2025 từ https://github.com/blynkkk/blynk-library