TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HỒ CHÍ MINH

KHOA ĐIỆN ĐIỆN TỬ

------ꕥ✧ꕥ------

TIỂU LUẬN CUỐI KỲ

Môn học: Hệ Thống Nhúng

Giảng viên

Đậu Trọng Hiển

Nhóm thực hiện: Nhóm 9

+ Lý Trần Quốc Uy - MSSV: 20139001

+ Lê Phan Nguyên Đạt - MSSV: 20139037

+ Lượng Vũ Hải Ninh - MSSV: 20139083

+ Cao Quỳnh Mai - MSSV: 20139009

+ Nguyễn Hoài Tâm - MSSV: 20139012

Thành phố Hồ Chí Minh, 11/2022

MỤC LỤC

CHƯƠNG 1: PHẦN CỨNG HỆ THỐNG NHÚNG

1.1. Bộ xử lí đa dụng

1.1.1. Khái niệm

Bộ xử lí đa dụng (Genaral purpose processor (GPP)) là một hệ thống kỹ thuật số được lập trình để giải quyết nhiều bài toán trong nhiều ứng dụng lớn,giải quyết những vấn đề tính toán trong các ứng dụng đa dạng như: giao tiếp, tự động hoá và các hệ thống nhúng công nghiệp.

Bộ xử lý đa đụng còn được gọi là CPU (Central Processing Unit)

1.1.2. Đặc điểm

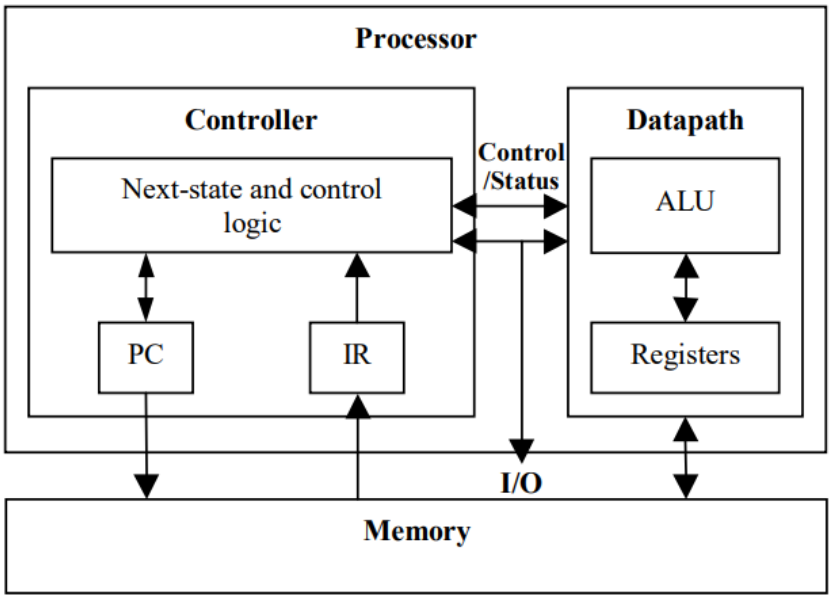
- Có thể lập trình để sử dụng trong nhiều ứng dụng.
- Cấu trúc phần cứng gồm có:

+ Datapath: Sử dụng các đường dẫn chung cùng với các thanh ghi và ALU (Arthmetic Logic Unit).

+ Khối Controller: Gồm các mạch logic để điều hướng dữ liệu, bộ đếm chương trình PC và bộ điều khiển thanh ghi lệnh (IR).

+ Bộ nhớ: Gồm bộ nhớ chương trình và bộ nhớ dữ liệu.

+ Các thanh ghi: hỗ trợ phục vụ cho bộ xử lý đa dụng trong nhiều chức năng khác.



Hình 1: Cấu trúc bộ xử lí đa dụng

1.1.3. Ứng dụng

Được sử dụng rất nhiều trong máy tính như 1 bộ xử lý trung tâm CPU, các vi điều khiển, máy trạm, cổng USB.

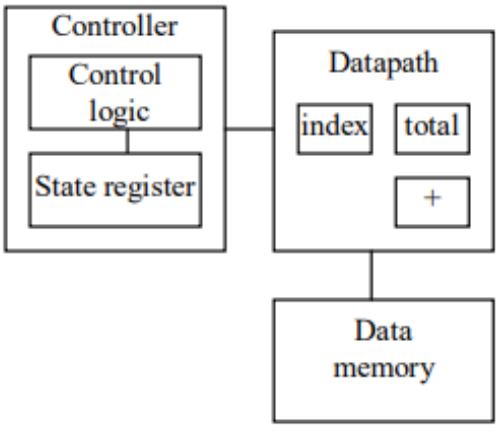
1.2. Bộ xử lí đơn dụng

1.2.1. Khái niệm

Bộ xử lý đơn dụng (Single Purpose Processors) là một hệ thống kỹ thuật số được thiết kế để thực hiện một chức năng nhất định để đáp ứng nhu cầu các chức năng chuyên biệt cho vô vàn các loại hệ thống nhúng. Một kỹ sư hệ thống nhúng thiết kế bộ xử lý đơn dụng bằng cách tạo ra một mạch số tùy chỉnh. Một vài cái tên thông dụng khác có thể được sử dụng như coprocessor và accelerator.

1.2.2. Đặc điểm

- Tốc độ hoạt động nhanh:
- Để các task được hoàn thành trong số chu kỳ máy thấp hơn, các thành phần datapath phải hoạt động song song, và dữ liệu giữa các datapath sẽ được truyền dẫn trực tiếp tới nhau mà không thông qua các thanh ghi trung gian, hoặc bỏ qua việc nạp bộ nhớ chương trình.
- Sử dụng các thành phần thực thi đơn giản, ít bộ ghép kênh hơn, và logic điều khiển đơn giản hơn.
- Sử dụng các công nghệ IC cho phép tùy chỉnh
- Kích thước nhỏ, không bắt buộc có bộ nhớ chương trình.
- Tập lệnh đơn giản
- Datapath và controller đơn giản hơn của bộ xử lý đa dụng.
- Tiêu thụ năng lượng thấp.
- Giá thành một thiết bị nhỏ khi chế tạo với số lượng lớn.
- Thời gian phát triển lâu, chi phí cho việc phát triển cao.
- Trong một vài trường hợp không tương thích với các bộ xử lý đa dụng.



Hình 2: Kiến trúc của một bộ xử lí đơn dụng

1.2.3. Ứng dụng

Ta thường xem các bộ xử lý đơn dụng như là các ngoại vi, vì chúng thường nằm ngoài CPU. Tuy nhiên, những vi điều khiển hiện nay thường tích hợp luôn cả các ngoại vi này vào trong chung với CPU, và thậm chí coi các thanh ghi ngoại vi này như một phần trong tập hợp các thanh ghi của CPU. Kết quả là ta có các "ngoại vi on-chip".

Các ứng dụng thường thấy của bộ xử lý đơn dụng:

+ Timer, Counter, và Watchdog timers.

+ Bộ điều khiển UART.

+ Bộ điều chế xung PWM.

+ Bộ điều khiển màn hình LCD.

+ Bộ điều khiển bàn phím.

+ Bộ điều khiển động cơ bước

+ Bộ chuyển đổi tương tự - số (ADC, DAC).

+ Real-time clock (RTC).

1.3. Bộ xử lí chuyên dụng

1.3.1. Khái niệm

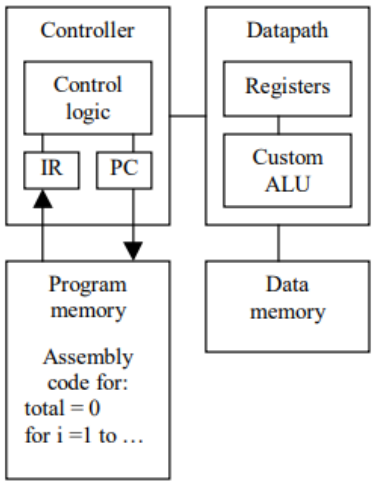
Khả năng linh hoạt thấp của bộ xử lý đơn dụng có thể khiến chúng không thể đáp ứng cho một số ứng dụng nhất định. Một bộ xử lý chuyên dụng (Application Specific Instruction set Processor (ASIP)) được sinh ra như là một giải pháp nhờ sự giao thoa giữa 2 dòng vi xử lý kể trên. Một ASIP được thiết kế cho một tập các ứng dụng cụ thể mà chia sẻ nhiều tính chất chung, ví dụ như các ứng dụng xử lý tín hiệu số, truyền thông, điều khiển nhúng,... Các kỹ sư thiết kế các bộ xử lý chuyên dụng này có thể thay đổi datapath cho từng lớp ứng dụng khác nhau, có thể thêm một vài thành phần tính năng đặc biệt, và loại bỏ các thành phần ít được sử dụng.

Sử dụng ASIP trong một hệ thống nhúng có thể mang lại nhiều lợi ích về mặt tùy biến trong khi vẫn đạt được cân bằng hiệu suất, năng lượng và kích thước. Tuy nhiên, các bộ xử lý này yêu cầu chi phí và thời gian phát triển rất lớn, và thậm chí là cần phát triển lại bộ compiler nếu nó chưa có sẵn. Do sự thiếu hụt về compiler cho các bộ xử lý quá đỗi đặc biệt này, nhiều thiết kế ASIP phải được lập trình bằng hợp ngữ.

1.3.2. Đặc điểm

Trong khi một phần được tùy biến để phục vụ các chức năng chỉ định, vẫn có nhiều phần được giữ lại để phục vụ cho việc tái lập trình.

Kiến trúc chung của ASIP được chỉ ra ở hình bên dưới, trong đó, datapath có thể được tùy chỉnh. Có thể có một thanh ghi tự động tăng giá trị, một đường dẫn cho việc cộng thanh ghi với một ô nhớ trong một lệnh, ít thanh ghi hơn, và bộ controller đơn giản hơn.



Hình 3: Kiến trúc của ASIP

1.3.3. Ứng dụng

Hai ứng dụng phổ biến của ASIP là vi đều khiển và bộ xử lý tín hiệu số

- Vi điều khiển:

Vi điều khiển là một vi xử lý được biến đổi để phù hợp cho phục vụ các ứng dụng nhúng. Các ứng dụng này thường là quản lý thông dụng và điều khiển các tín hiệu bit, tuy nhiên không yêu cầu thực hiện một số lượng lớn các phép tính toán phức tạp. Các vi điều khiển này có số đường dẫn dữ liệu ít hơn phục vụ cho việc đọc và xuất các dữ liệu ở cấp độ bit. Chúng thường được kết hợp với các ngoại vi thông dụng cho các ứng dụng điều khiển, ví dụ như giao tiếp truyền thông nối tiếp, timer, bộ đếm xung, và bộ chuyển đổi tương tự số.

- Bộ xử lý tín hiệu số (DSP):

DSP là một vi xử lý được thiết kế riêng để thực hiện các phép xủ lý trên các tín hiệu số, ví dụ như chuyển đổi sang số từ các tín hiệu tương tự như video và âm thanh. Các phép xử lý này gồm các tác vụ xử lý tín hiệu thông dụng như lọc tín hiệu, biến đổi (transformation) hay kết hợp (combination). Các phép xử lý này yêu cầu các xử lý toán học phức tạp, bao gồm các phép tính như nhân và cộng hay dịch và cộng. Để đáp ứng các phép tính này, DSP có thể có những thành phần datapath chuyên biệt như thành phần thực hiện phép nhân-tích lũy, có thể thực hiện phép tính phức tạo chỉ trong một lệnh. Vì các chương trình DSP thường thao tác trên một dãy dữ liệu rất lớn, DSP có thể bao gồm thêm cả phần cứng đặc biệt cho phép nạp vị trí bộ nhớ tuần tự song song với các hoạt động khác, nhằm tăng tốc độ thực thi chương trình.

1.4. Chuẩn giao tiếp UART

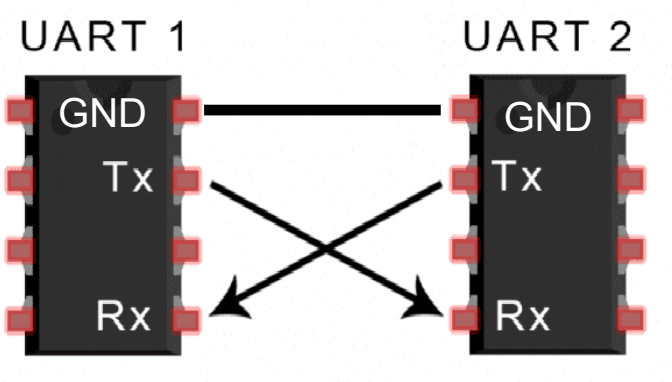
1.4.1. Đôi nét về chuẩn giao tiếp UART

UART (Universal Asynchronous Receiver Transmitter) là giao thức giao tiếp nối tiếp không đồng bộ. Đây là một trong những hình thức giao tiếp kỹ thuật số giữa thiết bị với thiết bị lâu đời nhất. Thường sử dụng hai dây:

+ Dây TX (Transmitter): là đường truyền dữ liệu

+ Dây RX (Receiver): là đường nhận dữ liệu

Trong một sơ đồ giao tiếp UART, chân TX (truyền) của một thiết bị kết nối trực tiếp với chân RX (nhận) của thiết bị kia và ngược lại. Chân GND của các thiết bị phải được đồng bộ với nhau.



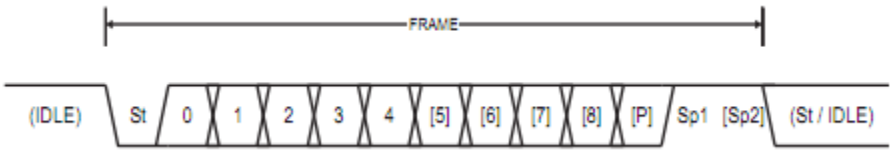
Hình 4: Sơ đồ kết nối UART giữa hai thiết bị

1.4.2. Một số khái niệm trong chuẩn truyền thông UART

- Baudrate: số đơn vị tín hiệu truyền trong 1 giây (baud/s). Trong UART, 1 đơn vị tín hiệu là 1Byte. Do UART không có đường xung clock để đồng bộ tốc độ truyền, nhận nên các thiết bị sẽ gửi dữ liệu dựa trên baudrate. Vậy nên baudrate thiết bị truyền phải bằng thiết bị nhận nếu không sẽ sinh ra lỗi.

Ví dụ: một thiết bị có baudrate là 115200 có thể truyền được 115200 x 8bit dữ liệu trong 1 giây.

Do UART truyền dữ liệu không đồng bộ, nên không có xung clock để đồng bộ thời gian truyền nhận của 1 bit. Nên trong chuẩn truyền thông UART, dữ liệu sẽ được truyền đi theo một định dạng khung truyền.



Hình 5: Khung truyền dữ liệu UART

- Start bit (bit ST): khi ở trạng thái chưa truyền thì đường truyền của UART giữ ở mức cao. Để bắt đầu truyền thì UART sẽ kéo đường truyền về mức thấp báo cho thiết bị nhận để bắt đầu nhận các bit trong khung dữ liệu.

- Data Frame (từ bit 0 đến bit [8]): Khung dữ liệu chứa dữ liệu để truyền đi. Nó có thể dài từ 8bit đến 9bit nhưng thường là 8bit. Dữ liệu được truyền với bit có trọng số bé nhất (LSB - Least Significant Bit) trước tiên.
- Parity Bit (bit [P]): Dùng để biết có bất kỳ dữ liệu nào bị thay đổi trong quá trình truyền dữ liệu hay không. Nếu bit Parity là mức thấp (0) thì tổng số bit cao trong khung dữ liệu truyền phải là số chẵn, nếu bit Parity là mức cao (1) thì tổng số bit cao trong khung dữ liệu truyền phải là số lẽ. Bit Parity là 0 hay 1 có thể do người chọn và cũng có thể không cần dùng bit Parity.
- Stop Bit(Bit kết thúc): Đưa đường truyền UART lên mức cao báo hiệu kết thúc truyền. Có thể có 1 hoặc 2 Stop bit.

1.5. Chuẩn giao tiếp I2C

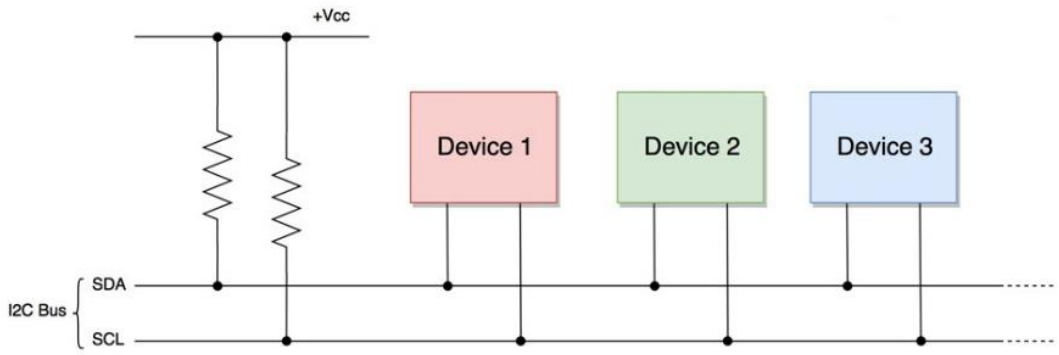
1.5.1. Đôi nét về chuẩn giao tiếp I2C

I2C (Inter-Integrated Circuit) là một giao thức giao tiếp nối tiếp đồng bộ. Được sử dụng để truyền nhận dữ liệu giữa các IC, chỉ sử dụng hai dây:

+ Dây SCL (Serial Clock Line): là đường xung clock để đồng bộ tốc độ truyền dữ liệu giữa các thiết bị

+ Dây SDA (Serial Data Line): là đường truyền, nhận dữ liệu.

Lưu ý khi thiết kế: Phải kéo Pull-Up hai dây SDA và SCL lên nguồn VCC

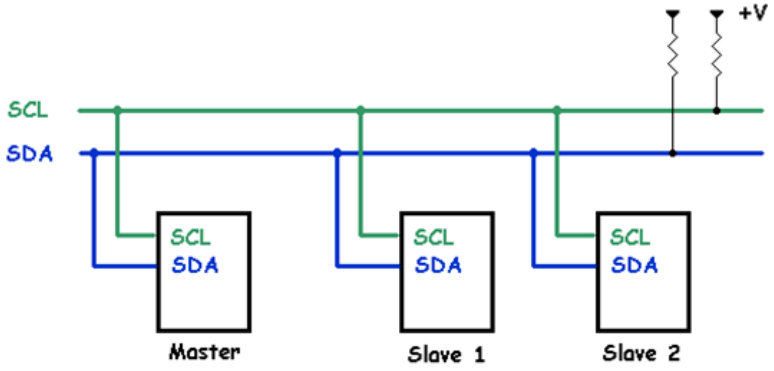


Hình 6: Sơ đồ kết nối I2C giữa các thiết bị

Khi 1 thiết bị kết nối với đường I2C thì SDA của thiết bị đó sẽ được nối vào dây SDA của Bus, còn chân SCL thì sẽ nối với dây SCL.

I2C cho phép nhiều thiết bị cùng kết nối vào một đường bus nên nếu không có cách quản lí phù hợp sẽ xảy ra chuyện nhầm lẫn dữ liệu giữa các thiết bị. I2C khắc phục điều đó bằng cách phân cấp thiết bị. Trên đường truyền I2C thường chỉ có một thiết bị

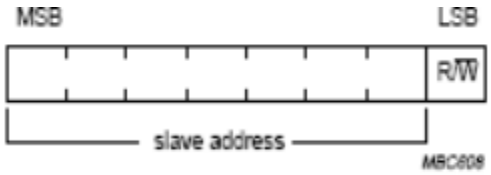
Master và nhiều thiết bị Slave. Mỗi thiết bị Slave chỉ có 1 địa chỉ duy nhất (gọi là địa chỉ I2C, gồm 7bit).



Hình 7: Phân cấp thiết bị trên đường truyền I2C

1.5.2. Khung truyền I2C

Khối địa chỉ I2C: Gồm 8 bit, trong đó 7bit đầu chứa địa chỉ I2C của Slave. Bit cuối là bit chọn chế độ đọc dữ liệu (R) hoặc ghi dữ liệu (W), trong đó 0 là chế độ W, 1 là chế độ R

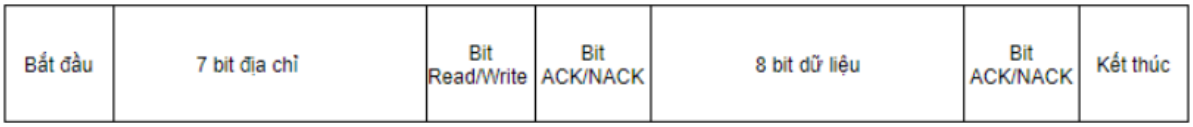


Hình 8: Khối địa chỉ I2C

Khi bắt đầu truyền dữ liệu I2C, thiết bị Master sẽ phát xung clock lên đường dây SCL, sau đó truyền khối địa chỉ I2C đi trên đường dây SDA. Nếu thiết bị Slave nhận được đúng địa chỉ thì sẽ phản hồi một bit ACK trên dây SDA để thông báo cho Master. Sau đó:

- Nếu bit R/W là chế độ Read thì Slave sẽ bắt đầu gửi dữ liệu về. Khi Master nhận đủ 8bit dữ liệu thì sẽ gửi bit ACK để thông báo.

- Nếu bit R/W là chế độ Write thì Master sẽ bắt đầu gửi dữ liệu xuống Slave. Khi Slave nhận đủ 8bit dữ liệu thì sẽ gửi bit ACK để thông báo.



Hình 9: Khung truyền dữ liệu I2C

1.6. Chuẩn giao tiếp SPI

1.6.1. Đôi nét về chuẩn giao tiếp SPI

SPI (Serial Peripheral Interface) là chuẩn truyền thông nối tiếp, có khả năng truyền dẫn song công toàn phần ở tốc độ cao. Được phát triển bởi hãng Motorola.

Sử dụng bốn dây:

+ Dây SCK (Serial Clock): là đường xung clock để đồng bộ tốc độ truyền nhận dữ liệu giữa các thiết bị

+ Dây MOSI (Master Output Slave Input): đường truyền dữ liệu từ master đến các slave

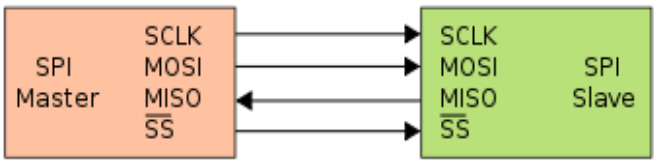
+ Dây MISO (Master Input Slave Output): đường nhận dữ liệu về master từ slave

+ Các dây CS (Chip Select): dùng để chọn thiết bị Slave mà Master muốn giao tiếp.

1.6.2. Các dạng sơ đồ kết nối SPI

Chuẩn SPI có ba dạng sơ đồ kết nối thường gặp:

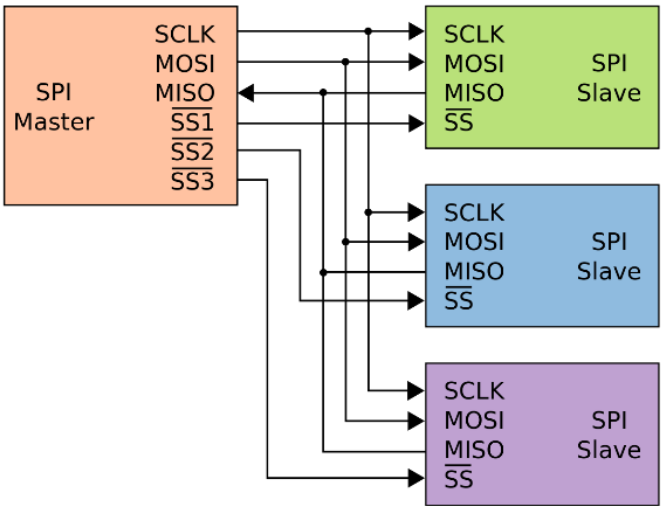
+ Sơ đồ kết nối một thiết bị Master và một thiết bị Slave: Trong sơ đồ này thì các chân SCK(hay SCLK), MOSI, MISO của cả hai thiết bị được nối với nhau. Một chân GPIO bất kì của thiết bị Master sẽ được cấu hình làm chân SS (hay CS) và nối trực tiếp với chân SS của thiết bị Slave.



Hình 10: Sơ đồ kết nối một thiết bị Master với một thiết bị Slave

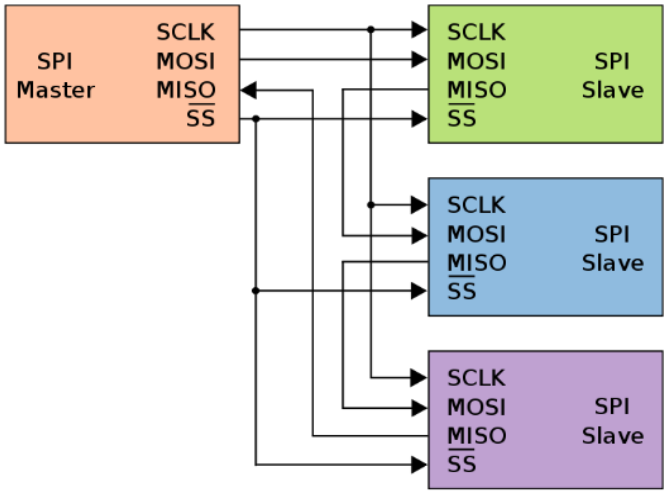
+ Sơ đồ kết nối một thiết bị Master và nhiều Slave độc lập: Trong sơ đồ này thì các chân SCK(hay SCLK), MOSI, MISO của các thiết bị được nối với nhau. Chân SS

của mỗi Slave sẽ kết nối với một GPIO bất kì của thiết bị Master. Mỗi lần chỉ có một thiết bị Slave hoạt động để gửi và nhận dữ liệu với Master.



Hình 11: Sơ đồ kết nối một thiết bị Master và nhiều Slave độc lập

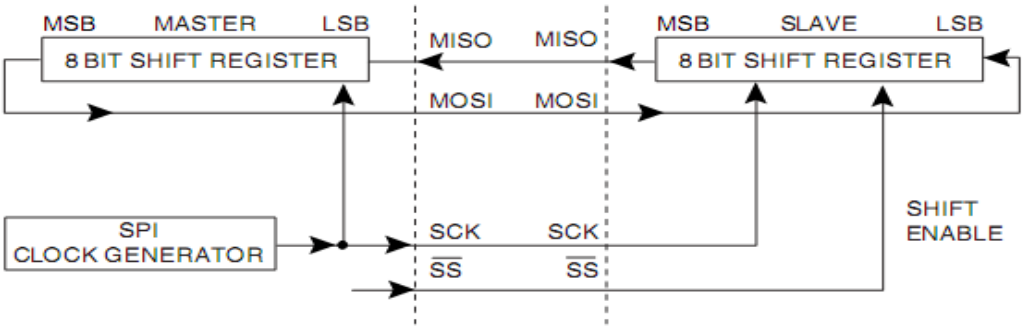
+ Sơ đồ kết nối một thiết bị Master và nhiều thiết bị Slave ở chế độ dây chuyền Daisy: Trong sơ đồ này thì các chân SCK (hay SCLK) của các thiết bị được nối với nhau. Một chân GPIO bất kì của thiết bị Master sẽ được cấu hình làm chân SS (hay CS) và kết nối với các chân SS của tất cả các thiết bị Slave. Chân MOSI của thiết bị Master sẽ được kết nối với chân MOSI của thiết bị Slave thứ nhất. Chân MISO của Slave thứ nhất sẽ kết nối với MOSI của thiết bị Slave thứ hai. Chân MISO của Slave tiếp theo sẽ được kết nối tương ứng và chân MISO của Slave cuối cùng sẽ kết nối với chân MISO của thiết bị Master.



Hình 12: Sơ đồ kết nối một thiết bị Master và nhiều thiết bị Slave ở chế độ dây chuyền Daisy

1.6.3. Khung truyền SPI

Khi bắt đầu truyền dữ liệu, thiết bị Master sẽ phát xung clock lên đường dây SCK. Tất cả các thiết bị SPI đều có một thanh ghi dịch 8bit. . Dữ liệu được truyền theo xung của dây SCK, mỗi xung truyền được 1bit. Khi Master gửi dữ liệu đi trên dây MISO thì cũng đồng thời nhận về 1bit dữ liệu từ Slave trên dây MOSI. Chính vì thế SPI có thể kiểm soát được khung dữ liệu và có thể truyền mà không cần bit bắt đầu, kết thúc.



Hình 13: Khung truyền SPI

1.7. Chuẩn mạng LAN

1.7.1. Đôi nét về chuẩn mạng LAN

Mạng LAN (Local Network Area) hay còn gọi là mạng cục bộ. Mạng LAN được hiểu là sự kết hợp của nhiều thiết bị được kết nối lại với nhau trong một hệ thống mạng tại một khu vực nhất định.

Hệ thống mạng giúp bạn chia sẻ, kết nối, lưu trữ các dữ liệu giữa các máy tính với nhau.

Hệ thống mạng LAN giúp các kết nối máy tính với máy in một cách an toàn và bảo mật.

Có thể sử dụng một máy tính bất kỳ để điều khiển các máy tính còn lại.

Sử dụng hệ thống mạng LAN để kết nối các thiết bị điện tử như máy tính bảng, laptop, điện thoại di động.

Giúp các công ty, tổ chức, doanh nghiệp muốn quản lý, sao lưu một cách an toàn dữ liệu của mình.

1.7.2. Các thành phần trong một mạng LAN

Máy tính đóng vai trò là thiết bị đầu cuối trong mạng, có nhiệm vụ gửi và nhận dữ liệu.

Kết nối: Kết nối cho phép dữ liệu truyền từ điểm này đến điểm khác trong mạng. Kết nối bao gồm các thành phần sau:

+ NIC: Các Card mạng (Network Interface Card) có nhiệm vụ chuyển đổi dữ liệu do máy tính tạo ra thành một định dạng có thể được truyền qua mạng LAN.

Thiết bị mạng: Mạng LAN yêu cầu các thiết bị mạng sau:

+ Hubs: Các Hub cung cấp các thiết bị tổng hợp hoạt động ở Lớp 1 trong mô hình tham chiếu OSI. Tuy nhiên, các hub đã được thay thế trong chức năng này bằng các thiết bị chuyển mạch, và rất hiếm khi thấy các trung tâm trong bất kỳ mạng LAN nào ngày nay.

+ Bộ chuyển mạch Ethernet (Ethernet Switches): Bộ chuyển mạch Ethernet tạo thành điểm tổng hợp cho các mạng LAN. Bộ chuyển mạch Ethernet hoạt động ở Lớp 2 trong mô hình tham chiếu OSI và cung cấp khả năng phân phối thông minh các khung trong mạng LAN.

+ Bộ định tuyến (Routers): Bộ định tuyến, đôi khi được gọi là cổng, cung cấp một phương tiện để kết nối các phân đoạn mạng LAN. Các bộ định tuyến hoạt động ở Lớp 3 của mô hình tham chiếu OSI.

1.7.3. Các giao thức trong mạng LAN

Giao thức chi phối cách dữ liệu được truyền qua mạng LAN và bao gồm:

+ Giao thức Ethernet

+ Giao thức Internet (IP)

+ Giao thức Internet (IPv4, IPv6)

+ Giao thức phân giải địa chỉ (ARP) và Giao thức phân giải địa chỉ ngược (RARP)

+ Giao thức cấu hình IP tự động (DHCP)

Mạng LAN thường được sử dụng để kết nối các máy tính trong gia đình, trong một văn phòng hay trong các tòa nhà của trường học, cơ quan tổ chức.

Mạng LAN giới hạn trong phạm vi có bán kính khoảng 100m.

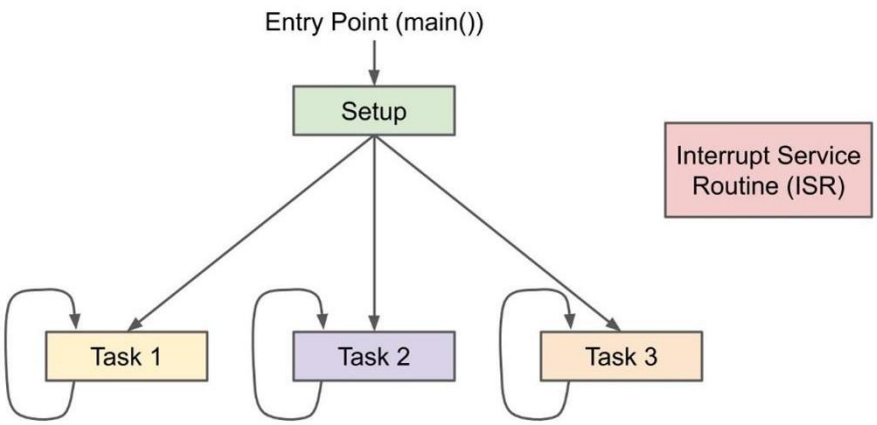
CHƯƠNG 2: PHẦN MỀM HỆ THỐNG NHÚNG

2.1. Hệ điều hành nhúng RTOS

RTOS (Real-time operating system) là một hệ điều hành được thiết kế cho các board mạch nhúng nhằm phục vụ các ứng dụng thời gian thực. Nó hoạt động bằng các chia chương trình ra thành các task. Mỗi task là một tập hợp các câu lệnh cần được thực thi.

Lấy ví dụ khi có ba task A, B, C có cùng mức ưu tiên thì sau khi thực hiện xong câu lệnh thứ nhất của task A, RTOS sẽ nhảy qua câu lệnh thứ nhất của task B và tiếp theo là của task C. Sau đó quay về câu lệnh thứ hai của task A. Còn nếu các task khác mức ưu tiên thì sau khi thực hiện xong một câu lệnh, RTOS sẽ nhảy tới task tiếp theo có mức ưu tiên lớn hơn và đã tới sự kiện.

Do RTOS chỉ là cải tiến về mặt phân bố phần mềm nên khi có sự kiện ngắt (Interrupt) xảy ra thì các chương trình ngắt vẫn được thực hiện trước tiên.



Hình 14: Sơ đồ hoạt động của RTOS

2.2. Hệ điều hành Android

Android là một hệ điều hành mã nguồn mở phát triển dựa trên nhân Linux, được thiết kế dành cho các thiết bị di động có màn hình cảm ứng. Android được thiết kế bởi Android Inc với hỗ trợ tài chính từ Google và được Google mua lại vào năm 2005.

2.3. Giới thiệu IDE Android Studio

Android Studio là môi trường phát triển tích hợp (hay IDE) dùng để phát triển các ứng dụng dành riêng cho Android. Được Google xây dựng, phát triển và ra đời vào tháng 5 năm 2013 nhằm thay thế cho phần mềm Ecilipse Android Development Tool (ADT) với nhiều tính năng, trình soạn thảo, giao diện và công cụ mạnh mẽ giúp người lập trình nâng cao năng suất trong xây dựng và phát triển ứng dụng Android. Android Studio xây dựng dựa trên IDE Jave Intellij của hãng JetBrains, có thể sử dụng trên các hệ điều hành trên Windows, Mac và Linux.

Android Studio hỗ trợ ngôn ngữ lập trình chính là Java với XML, sau này có thêm Kotlin.

2.4. Giới thiệu bộ SDK ESP-IDF

ESP-IDF là một bộ SDK (Software Development Kit) của công ty Espressif cung cấp cho các lập trình viên để lập trình cho dòng chip ESP32.

ESP32 là một series các SOC (System On Chip) tiêu thụ năng lượng thấp, hỗ trợ Wifi và Bluetooth Low Enregy (BLE) dựa trên bộ vi xử lý Tensilica Xtensa LX6.

ESP-IDF cung cấp các thư viện chứa các câu lệnh điều khiển phần cứng ESP32 và code mẫu tham khảo đính kèm. Các thư viện đã được biên dịch, tối ưu hoá và một trình biên dịch kèm theo (Xtensa GCC) để biên dịch các code của người dung.

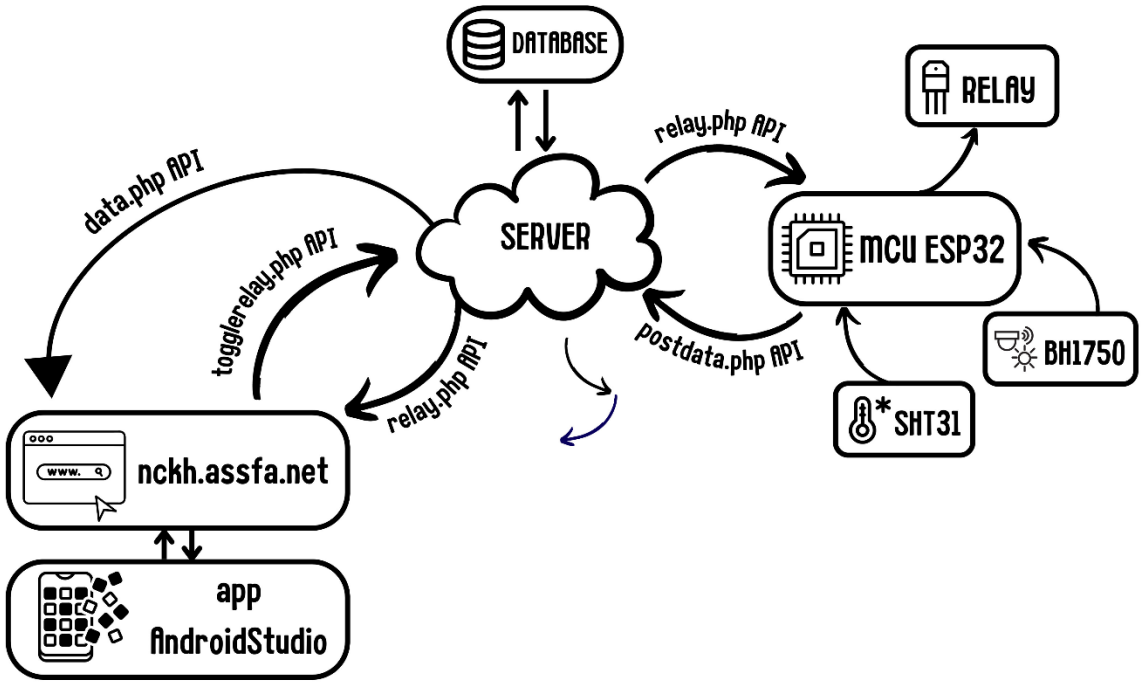
ESP-IDF hỗ trợ ngôn ngữ lập trình chính là C/C++.

CHƯƠNG 3: BÀI TẬP CUỐI KÌ

3.1. Nội dung thực hiện

Thiết kế hệ thống giám sát nhiệt độ, độ ẩm, ánh sáng và điều khiển thiết bị điện trong nhà.

Sơ đồ hệ thống:



Hình 15: Sơ đồ hệ thống

Nhóm sử dụng vi điều khiển ESP32, đọc dữ liệu từ các cảm biến SHT31 (nhiệt độ, độ ẩm) và BH1750 (ánh sáng) và gửi lên server thông qua API postdata.php. Và lấy trạng thái nút bấm từ server thông qua API relay.php để điều khiển một Relay bật tắt thiết bị điện trong nhà.

Phần app giám sát được viết bằng Android Studio theo dạng webview. Khi khởi động app sẽ load trang web https://nckh.assfa.net lên. Trên app sẽ có các chức năng giám sát trạng thái kết nối mạng. Khi mạng bị ngắt kết nối thì trên app sẽ hiển thị thông báo “Không có kết nối internet”.

Giao diện trên web bao gồm ba khung hiển thị nhiệt độ, độ ẩm, ánh sáng. Một biểu đồ cập nhật theo thời gian thực và một nút bấm để điều khiển bật tắt thiết bị điện.

Hình 16: Giao diện app

Hình 17: Thông báo khi không có kết nối mạng

3.2. Thiết kế phần cứng

3.2.1. Thông số vi điều khiển ESP32

ESP32 là một series các SOC (System On Chip) tiêu thụ năng lượng thấp, hỗ trợ Wifi và Bluetooth Low Enregy (BLE) dựa trên bộ vi xử lý Tensilica Xtensa LX6.

Trong hệ thống này, nhóm sử dụng vi điều khiển ESP-DOWD(V3) với các thông số:

+ Tần số hoạt động: 240MHz

+ Bộ nhớ: 520KB SRAM, 448KB ROM cho bootloader, 4MB Flash chương trình

Hình 18: Vi điều khiển ESP32-DOWD V3

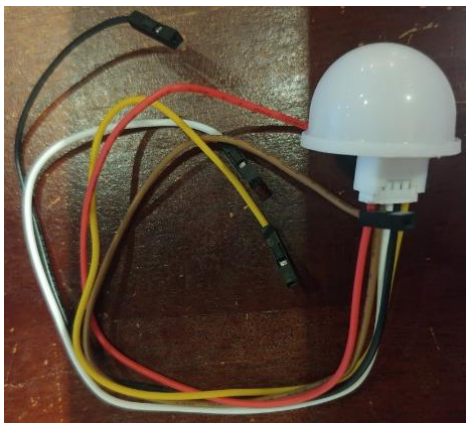
3.2.2. Thông số cảm biến SHT31

- Tầm đo:
  + Cảm biến nhiệt độ: -40 đến 125 (ºC)
  + Cảm biến độ ẩm: 0 đến 100 (%RH)
- Sai số:
  + Cảm biến nhiệt độ: ±0.3 (ºC)
  + Cảm biến độ ẩm: ±0.8 (%RH) (kiểm chứng tại 25ºC)
- Điện áp hoạt động: 2.4 đến 5.5 (V)
- Dòng tiêu thụ: 800 (µA)
- Giao thức giao tiếp: I2C

Hình 19: Cảm biến SHT31

3.2.3. Thông số cảm biến BH1750

- Tầm đo: 1 đến 65535 (lx)
- Sai số: ±4 (lx)
- Điện áp hoạt động: 2.4 đến 3.6 (V)
- Dòng tiêu thụ: 120 (µA)
- Giao thức giao tiếp: I2C

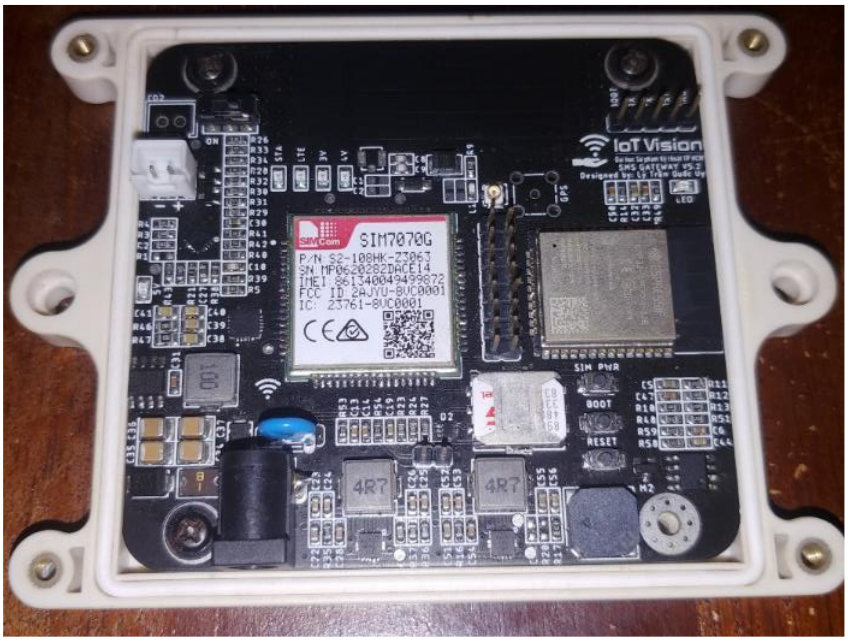


Hình 20: Cảm biến BH1750

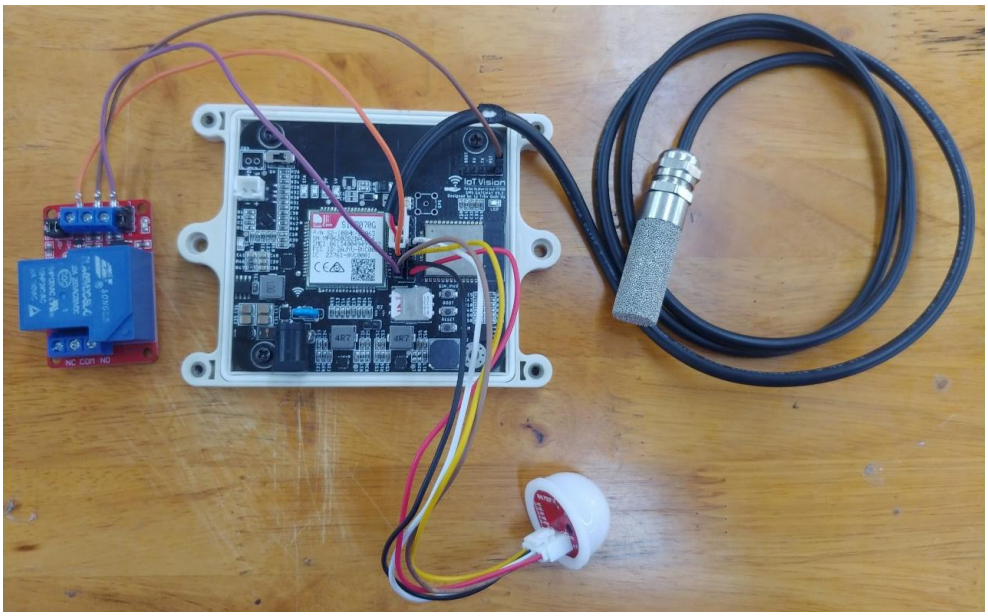
3.2.4. Board mạch chính

Board mạch chính được sử dụng do nhóm thiết kế

- Vi điều khiển chính: ESP32 WROOM-32E (lõi ESP32 DOWD V3)
- Nguồn cấp: Từ 9V đến 30VDC
- Tích hợp mạch sạc pin LIPO và hỗ trợ chạy bằng pin LIPO 1S
- Tích hợp module LTE SIM7070G, có hỗ trợ GPS



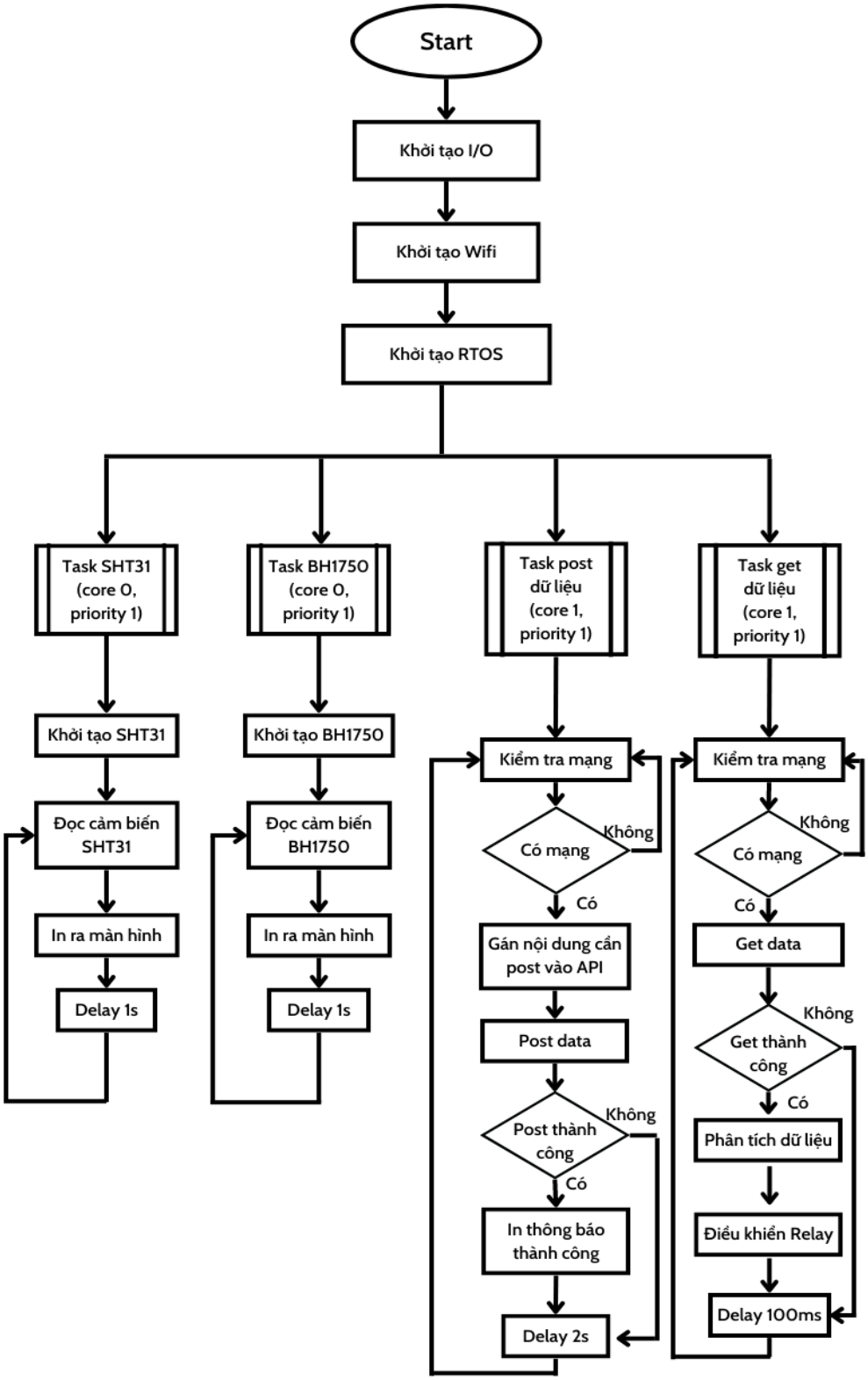
Hình 21: Board mạch chính



Hình 22: Kết nối các thành phần của hệ thống

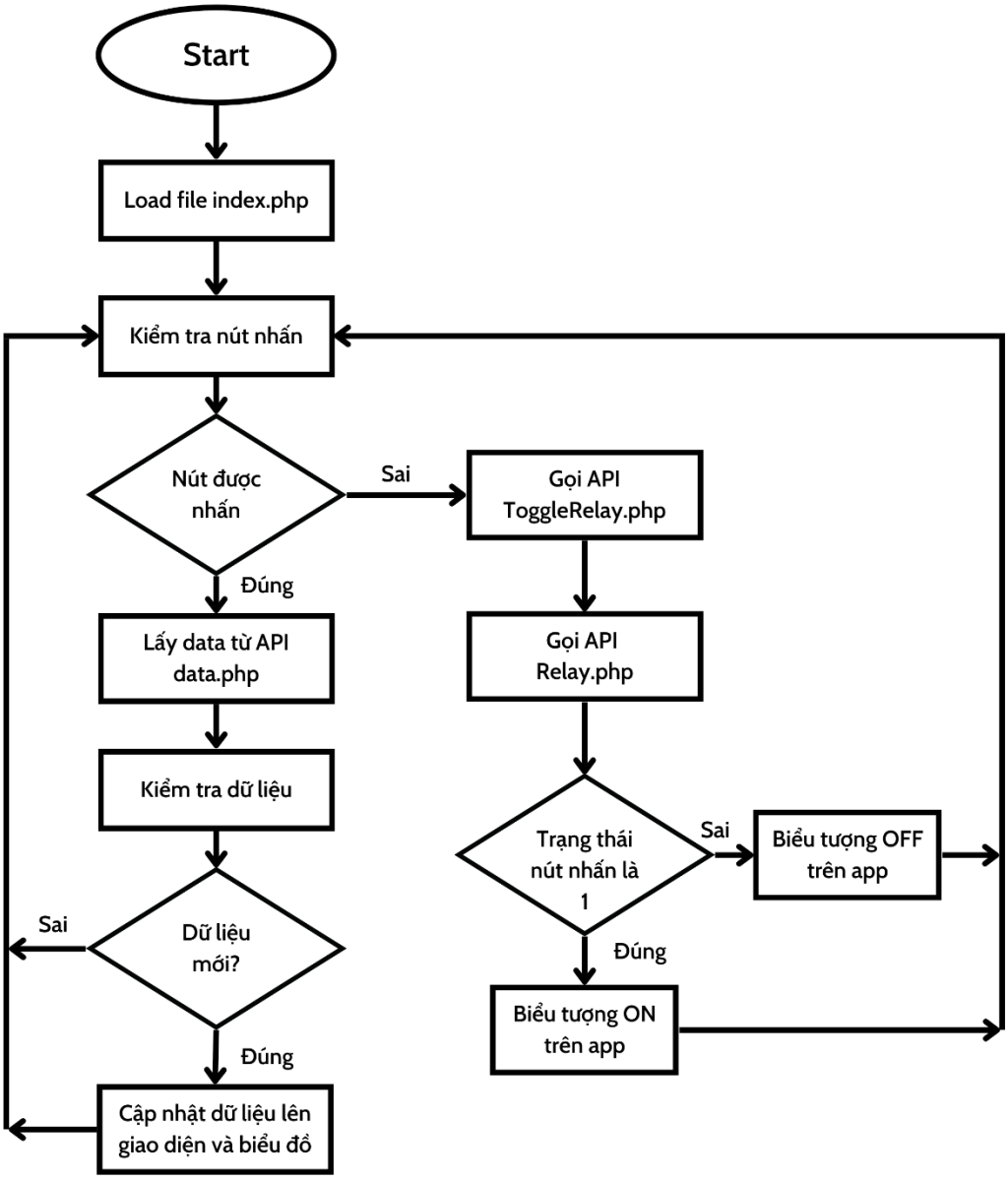
3.3. Lưu đồ giải thuật

3.3.1. Lưu đồ giải thuật thiết bị



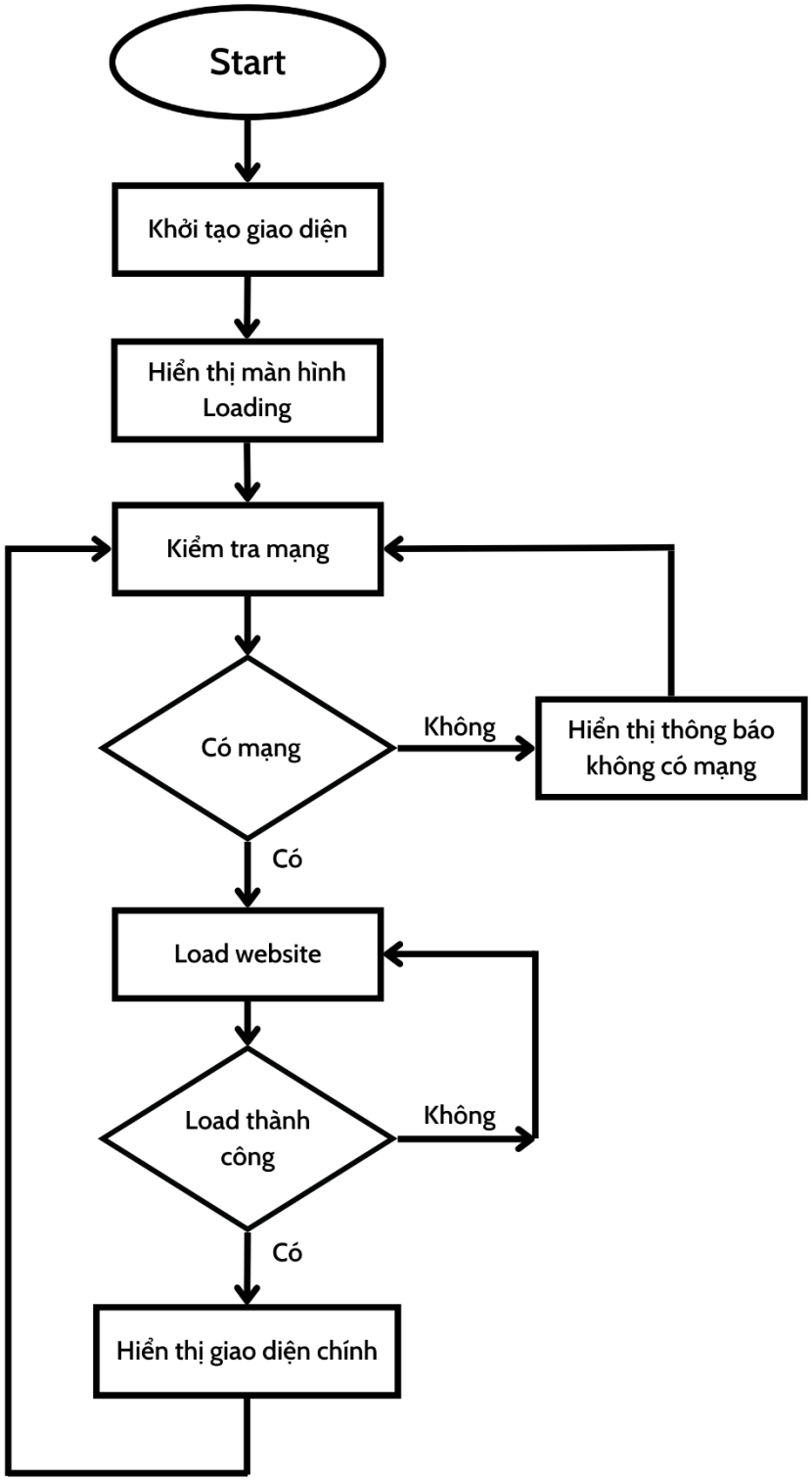
Hình 23: Lưu đồ giải thuật thiết bị

3.3.2. Lưu đồ giải thuật webserver



Hình 24: Lưu đồ giải thuật webserver

3.3.3. Lưu đồ giải thuật app Android Studio



Hình 25: Lưu đồ giải thuật app Android

3.4. Code hệ thống

Link code hoàn chỉnh của hệ thống:
https://drive.google.com/drive/folders/1ehQBp0zr8PXoVD1QtNE1D33bRBk7yEHE?usp=sharing

3.4.1. Code thiết bị

Phần code điều khiển ESP32 đọc dữ liệu cảm biến SHT31, BH1750 và gửi dữ liệu về server. Lấy trạng thái nút bấm từ server để điều khiển một Relay. Sử dụng hệ điều hành RTOS gồm bốn task: đọc cảm biến SHT31, đọc cảm biến BH1750, gửi dữ liệu cảm biến về server, lấy trạng thái nút bấm từ server.

Các ngôn ngữ lập trình được sử dụng: C/C++

Hardware.h

```
#ifndef __HARDWARE_H__
#define __HARDWARE_H__
#include "esp_log.h"
#include "driver/gpio.h"
#include <i2cdev.h>
#include <sht3x.h>
#include <bh1750.h>
#include <iostream>
using namespace std;

#define BUTTON                  GPIO_NUM_0
#define LED                     GPIO_NUM_23
#define RELAY                   GPIO_NUM_5
#define I2C_SDA_GPIO            GPIO_NUM_19
#define I2C_SCL_GPIO            GPIO_NUM_18

esp_err_t KhoiTaoIO(void);
esp_err_t BatLEDStatus(void);
esp_err_t TatLEDStatus(void);
#endif // __HARDWARE_H__
```

Hardware.cpp

```
#include "Hardware.h"
static const char *TAG = "Hardware";
esp_err_t KhoiTaoIO(void){
    esp_err_t ret = ESP_OK;
    ESP_LOGI(TAG, "Khởi tạo Led trạng thái");
    ret |= gpio_reset_pin(LED);
    ret |= gpio_set_direction(LED, GPIO_MODE_OUTPUT);
    ESP_LOGI(TAG, "Khởi tạo button");
    ret |= gpio_reset_pin(BUTTON);
    ret |= gpio_set_direction(BUTTON, GPIO_MODE_INPUT);
    ESP_LOGI(TAG, "Khởi tạo relay");
    ret |= gpio_reset_pin(RELAY);
    ret |= gpio_set_direction(RELAY, GPIO_MODE_OUTPUT);
    return ret;
}
esp_err_t BatLEDStatus(void){
    return gpio_set_level(LED, 1);
}
esp_err_t TatLEDStatus(void){
    return gpio_set_level(LED, 0);
}
```

main.cpp

```
#include <stdio.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "driver/gpio.h"
```

```
#include "esp_log.h"
#include "sdkconfig.h"
#include"string.h"
#include <cstring>

#include <iostream>
using namespace std;
static const char *TAG = "main";
#include "Hardware/Hardware.h"
#include "KiemTraInternet.h"
#include "cJSON.h"
#include "POSTGET.h"
extern "C"{
    void app_main(void);
}
char _IMEI[13];
void GetChipID(){
    uint8_t MAC[6];
    esp_read_mac(MAC, ESP_MAC_WIFI_STA);
    for (uint8_t i = 0; i < 6; i++){
        static int index = 0;
        index += snprintf(&_IMEI[index], 20 - index, "%02X", MAC[i]);
    }
    // Cập nhật SSID cho WiFi AP
    snprintf((char *)wifi_settings.ap_ssid, MAX_SSID_SIZE, "%s_%s",
DEFAULT_AP_SSID, _IMEI);
    ESP_LOGI(TAG, "MAC: %s",_IMEI);
}
uint8_t duration;
```

```
sht3x_t sht31_dev;
float _NhietDo, _DoAm;
void vTaskDocCamBienSHT31(void *pvParameters){
   memset(&sht31_dev, 0, sizeof(sht3x_t));
   sht3x_init_desc(&sht31_dev, 0, 0x44, I2C_SDA_GPIO, I2C_SCL_GPIO);
   sht3x_init(&sht31_dev);
   duration = sht3x_get_measurement_duration(SHT3X_HIGH);
   while(1){
      sht3x_start_measurement(&sht31_dev, SHT3X_SINGLE_SHOT,
SHT3X_HIGH);
      vTaskDelay(duration);
      sht3x_get_results(&sht31_dev, &_NhietDo, &_DoAm);
      ESP_LOGI(TAG, "Cảm biến SHT31: %.2f°C, %.2f%%", _NhietDo, _DoAm);
      vTaskDelay(pdMS_TO_TICKS(1000));
   }
   vTaskDelete(NULL);
}
i2c_dev_t BH1750_Dev;
uint16_t Lux_Data;
void vTaskDocCamBienBH1750(void *pvParameters){
   bh1750_init_desc(&BH1750_Dev, BH1750_ADDR_LOW, 0, I2C_SDA_GPIO,
I2C_SCL_GPIO);
   bh1750_setup(&BH1750_Dev, BH1750_MODE_CONTINUOUS,
BH1750_RES_HIGH);
   while(1){
      if (bh1750_read(&BH1750_Dev, &Lux_Data) != ESP_OK)
         ESP_LOGE(TAG, "Could not read lux data");
      else
         ESP_LOGW(TAG, "Cảm biến BH1750: %dlx", Lux_Data);
      vTaskDelay(pdMS_TO_TICKS(1000));
```

```
    }
    vTaskDelete(NULL);
}
char http_response_post[MAX_HTTP_OUTPUT_BUFFER];
void PostData(){
    char url_request[256];
    snprintf(url_request, 256,

"http://nckh.assfa.net/postdata.php?ThietBi=%s&NhietDo=%.2f&DoAm=%.2f&AnhSang=%d&CO2=480&Mua=1&ThoiTiet=1",_IMEI, _NhietDo, _DoAm, Lux_Data);
    cout << url_request << endl;
    HTTP_CODE_e http_code = http_get(url_request, http_response_post);
    if(http_code == HTTP_OK){
        ESP_LOGW(TAG, "Post Data OK");
    }
}
char http_response[MAX_HTTP_OUTPUT_BUFFER];
char http_response_cut[MAX_HTTP_OUTPUT_BUFFER];
int TrangThaiRelay;
void GetData(){
    HTTP_CODE_e http_code = http_get("http://nckh.assfa.net/relay.php", http_response);
    if(http_code == HTTP_OK){
        cout << http_response << endl;
        TrangThaiRelay = http_response[11] - 48;
        cout << "TrangThaiRelay: " << TrangThaiRelay << endl;
        gpio_set_level(RELAY, TrangThaiRelay);
    }
}
```

```
void vTaskGetData(void *pvParameters){
   while(1){
      if(KiemTraInternet() == CONNECTION_INTERNET_OK){
         GetData();
      }
      vTaskDelay(pdMS_TO_TICKS(100));
   }
   vTaskDelete(NULL);
}
void vTaskPostData(void *pvParameters){
   while(1){
      if(KiemTraInternet() == CONNECTION_INTERNET_OK){
         PostData();
      }
      vTaskDelay(pdMS_TO_TICKS(2000));
   }
   vTaskDelete(NULL);
}
void app_main(void){
   ESP_ERROR_CHECK(i2cdev_init());
   KhoiTaoIO();
   GetChipID();
   /* start the wifi manager */
   wifi_manager_start();
   wifi_manager_set_callback(WM_EVENT_STA_GOT_IP, &CheckIP);
   xTaskCreatePinnedToCore(vTaskDocCamBienSHT31,
"vTaskDocCamBienSHT31", 4096, NULL, 1, NULL, 0);
   xTaskCreatePinnedToCore(vTaskDocCamBienBH1750,
"vTaskDocCamBienBH1750", 4096, NULL, 1, NULL, 0);
```

```
  xTaskCreatePinnedToCore(vTaskGetData, "vTaskGetData", 4096, NULL, 1, NULL, 1);
  xTaskCreatePinnedToCore(vTaskPostData, "vTaskPostData", 4096, NULL, 1, NULL, 1);
  while (1) {
    if(KiemTraInternet() == CONNECTION_INTERNET_OK){
      TatLEDStatus();
    }
    else{
      BatLEDStatus();
    }
    vTaskDelay(pdMS_TO_TICKS(1000));
  }
}
```

3.4.2. Code Server

Phần code trên server tại subdomain: https://nckh.assfa.net.

Chức năng các API:

+ postdata.php: dùng để đưa dữ liệu vào database. Khi thiết bị gọi API này sẽ truyền vào số định danh thiết bị, các thông số nhiệt độ, độ ẩm,… API postdata.php sẽ đọc các dữ liệu và ghi vào database.

+ data.php: API này dùng để lấy một điểm dữ liệu cuối cùng phục vụ cho cập nhật dữ liệu trên web. Khi gọi API này sẽ lấy ra điểm dữ liệu cuối cùng từ database và xử lí thành chuỗi json và trả về kết quả.

+ togglerelay.php: Khi gọi, API này sẽ lấy giá trị Relay hiện tại lưu trong database và đảo trạng thái. Sau đó sẽ ghi lại giá trị đã đảo vào database.

+ relay.php: Khi gọi, API này sẽ lấy giá trị Relay hiện tại lưu trong database và xử lí thành chuỗi json và trả về kết quả.

Các ngôn ngữ lập trình được sử dụng: PHP và Javascript.

Các ngôn ngữ bổ trợ: ngôn ngữ đánh dấu siêu văn bản HTML, ngôn ngữ định dạng CSS và các lệnh SQL.

connectdb.php

```
<?php
	$sever = "localhost";
	$user = "wseometc_admin";
	$pass = "Lytranquocuy@123";
	$db = "wseometc_nckh";
	$conn = new mysqli($sever, $user, $pass, $db);
	if (!$conn) {
		die("Ket noi db khong thanh cong!" . $conn->connect_error);
	}
	$conn->query("set names 'utf8'");
?>
```

postdata.php

```
<?php
	require('connectdb.php');
	$ThietBi = $_GET["ThietBi"];
	$NhietDo = $_GET["NhietDo"];
	$DoAm = $_GET["DoAm"];
	$AnhSang = $_GET["AnhSang"];
	$CO2 = $_GET["CO2"];
	$Mua = $_GET["Mua"];
	$ThoiTiet = $_GET["ThoiTiet"];
	date_default_timezone_set('Asia/Ho_Chi_Minh');
	$time_act = date('Y-m-d H:i:s'); // use actual date() format displayed in your table.
	$sql = "INSERT INTO DataIoTProject (ThietBi, NhietDo, DoAm, AnhSang, CO2, Mua, ThoiTiet, ThoiGian)
```

```
        VALUES ('" . $ThietBi . "','" . $NhietDo . "', '" . $DoAm . "' , '" . $AnhSang . "' , '" . $CO2 . "' , '" . $Mua . "', '" . $ThoiTiet . "', '" . $time_act . "')";
        $conn->query($sql);
        $conn->close();
?>
```

data.php

```
<?php
        require('connectdb.php');
        $sql = "SELECT * FROM `DataIoTProject` ORDER BY STT DESC LIMIT 1";
        if ($result = $conn->query($sql)) {
                while ($row = $result->fetch_assoc()) {
                        $ThietBi = $row["ThietBi"];
                        $NhietDo = $row["NhietDo"];
                        $DoAm = $row["DoAm"];
                        $AnhSang = $row["AnhSang"];
                        $CO2 = $row["CO2"];
                        $Mua = $row["Mua"];
                        $ThoiTiet = $row["ThoiTiet"];
                        $ThoiGian = $row["ThoiGian"];
                        echo '[{"ThietBi":"' . $ThietBi . '","NhietDo":' . $NhietDo . ',"DoAm":' . $DoAm . ',"AnhSang":' . $AnhSang . ',"CO2":' . $CO2 . ',"Mua":' . $Mua . ', "ThoiTiet":' . $ThoiTiet . ',"ThoiGian":"' . $ThoiGian . '"}]';
                }
                $result->free();
        }
        $conn->close();
?>
```

relay.php

```
<?php
	require('connectdb.php');
	$sql = "SELECT * FROM `DataModeIoTProject` WHERE 1";
	if ($result = $conn->query($sql)) {
		while ($row = $result->fetch_assoc()) {
			$Relay = $row["K1"];
			echo '[{"Relay1":' . $Relay . '}]';
		}
		$result->free();
	}
	$conn->close();
?>
```

togglerelay.php

```
<?php
	require('connectdb.php');
	$sql = "SELECT * FROM `DataModeIoTProject` WHERE 1";
	if ($result = $conn->query($sql)) {
		while ($row = $result->fetch_assoc()) {
			$Relay = $row["K1"];
			echo '[{"Relay1":' . $Relay . '}]';
			$Relay = 1 - $Relay;
			$sql1 = "UPDATE `DataModeIoTProject` SET `K1`=" .
$Relay . " WHERE 1";
			echo $sql1;
			$conn->query($sql1);
		}
		$result->free();
```

```
        }
        $conn->close();
?>
```

index.php

```
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
      <meta name="format-detection" content="telephone=no">
      <meta name="msapplication-tap-highlight" content="no">
      <meta name="viewport" content="user-scalable=no, initial-scale=1, maximum-scale=1, minimum-scale=1, width=device-width">
      <title>IoTProject</title>
      <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.1.1/jquery.js"></script>
      <script src="https://code.highcharts.com/highcharts.js"></script>
      <script src="https://code.highcharts.com/modules/exporting.js"></script>
      <script src="https://code.highcharts.com/modules/export-data.js"></script>
      <script src="https://code.highcharts.com/modules/accessibility.js"></script>
      <link rel="stylesheet" href="css/jquery.mobile-1.4.2.min.css" />
      <script src="js/jquery.js"></script>
      <script src="js/jquery.mobile-1.4.2.min.js"></script>
      <!-- <link rel="stylesheet" href="css/index.css"> -->
      <style>
         .highcharts-figure,
         .highcharts-data-table table {
         min-width: 320px;
         max-width: 800px;
```

```
margin: 1em auto;
}
#container {
    height: 400px;
    background-color: transparent;
}
.highcharts-data-table table {
font-family: Verdana, sans-serif;
border-collapse: collapse;
border: 1px solid #ebebeb;
margin: 10px auto;
text-align: center;
width: 100%;
max-width: 500px;
}
.highcharts-data-table caption {
padding: 1em 0;
font-size: 1.2em;
color: #555;
}
.highcharts-data-table th {
    font-weight: 600;
    padding: 0.5em;
}
.highcharts-data-table td,
.highcharts-data-table th,
.highcharts-data-table caption {
    padding: 0.5em;
}
```

```
.highcharts-data-table thead tr,
.highcharts-data-table tr:nth-child(even) {
    background: #f8f8f8;
}
.highcharts-data-table tr:hover {
    background: #f1f7ff;
}
.button {
    display: inline-block;
    padding: 15px 25px;
    font-size: 24px;
    cursor: pointer;
    text-align: center;
    text-decoration: none;
    outline: none;
    color: #fff;
    background-color: #4CAF50;
    border: none;
    border-radius: 15px;
    box-shadow: 0 9px #999;
}
.button:hover {background-color: #3e8e41}
.button:active {
    background-color: #3e8e41;
    box-shadow: 0 5px #666;
    transform: translateY(4px);
}
#backgroundimage
{
```

```
        height: 100%;
        left: 0;
        margin: 0;
        padding: 0;
        position: fixed;
        top: 0;
        width: 100%;
        z-index: -1;
    }
    #mainpage
    {
        background-image: url('/img/bg.png');
        background-size: 100vw 60vh;
        background-repeat: no-repeat;
        background-color: #e7f1fd
    }
    #PageHeader
    {
        background-color: transparent;
        border: 0;
    }
    table, th, td {
        border:0px solid black;
        border-spacing: 10px;
    }
    #TableChart
    {
        border-spacing: 0px;
    }
```

```
        </style>
    </head>
    <body align="center">
        <script src="js/index.js"></script>
        <!-- Khóa chức năng viewsource -->
        <script>
            document.addEventListener('contextmenu', event =>
event.preventDefault());
            document.addEventListener('keydown', function (event){
                if (event.ctrlKey){
                    event.preventDefault();
                }
                if(event.keyCode == 123){
                    event.preventDefault();
                }
            });
            node.addEventListener('contextmenu', function(e){
                e.preventDefault();
            });
        </script>
        <!-- Khóa chức năng viewsource -->
        <div id = "mainpage" data-role='page' style='display: inherit'>
            <div id = "PageHeader" data-role='header' style="color:white; width:100%;
text-align: center">
            <!-- <h1 style="color:white; font-size: 20px;">IoT Project</h1>     -->
            <h3 id="TenThietBi" style="color:white;">Thiết Bị</h3>
            </div>
            <div data-role="content">
                <table style="width:100%">
```

```
            <tr >
                <th style="background:white; border-radius: 20px; width:33%; height: 150px;box-shadow: 0px 8px 15px rgba(0, 0, 0, 0.1);">
                    <div style="width:100%; height: 100%;">
                        <div style="width:100%; height: 15px;"></div>
                        <div style="width:100%; height: 20px; font-size: 15px;">
                            Nhiệt Độ
                        </div>
                        <div style="width:100%; height: 50px; text-align:center;">
                            <img src="/img/nhietdo.png" width="50px" height="50px">
                        </div>
                        <div style="width:100%; height: 5px;"></div>
                        <div id="GiaTriNhietDo" style="width:100%; height: 20px; font-size: 18px;">0</div>
                        <div style="width:100%; height: 5px;"></div>
                        <div style="width:100%; height: 20px;">(°C)</div>
                    </div>
                </th>
                <th style="background:white; border-radius: 20px; width:33%; height: 150px;box-shadow: 0px 8px 15px rgba(0, 0, 0, 0.1);">
                    <div style="width:100%; height: 100%;">
                        <div style="width:100%; height: 15px;"></div>
                        <div style="width:100%; height: 20px; font-size: 15px;">
                            Độ Ẩm
                        </div>
                        <div style="width:100%; height: 50px; text-align:center;">
                            <img src="/img/doam.png" width="50px" height="50px">
                        </div>
                        <div style="width:100%; height: 5px;"></div>
```

```
                    <div id="GiaTriDoAm" style="width:100%; height: 20px; font-size: 18px;">0</div>
                    <div style="width:100%; height: 5px;"></div>
                    <div style="width:100%; height: 20px;">(%)</div>
                </div>
            </th>
            <th style="background:white; border-radius: 20px; width:33%; height: 150px;box-shadow: 0px 8px 15px rgba(0, 0, 0, 0.1);">
            <div style="width:100%; height: 100%;">
                <div style="width:100%; height: 15px;"></div>
                <div style="width:100%; height: 20px; font-size: 15px;">
                    Ánh Sáng
                </div>
                <div style="width:100%; height: 50px; text-align:center;">
                    <img src="/img/anhsang.png" width="50px" height="50px">
                </div>
                <div style="width:100%; height: 5px;"></div>
                <div id="GiaTriAnhSang" style="width:100%; height: 20px; font-size: 18px;">0</div>
                <div style="width:100%; height: 5px;"></div>
                <div style="width:100%; height: 20px;">(lx)</div>
            </div>
            </th>
        </tr>
    </table>
    <div style="background:white; border-radius: 20px; width:100%; height: 260px;">
        <div style="width:95%; height: 10px; margin-left: auto; margin-right: auto;"> </div>
        <div id="Chart1" style="background:black; width:95%; height: 240px; margin-left: auto; margin-right: auto; "> </div>
```

```
            <!-- <div id="Chart1" style="width:90%; height: 240px; position: absolute; top: 56%; left: 50%; transform: translate(-50%, -50%); "></div> -->
        </div>
        <br>
        <div style="background:white; border-radius: 30px;  width=100%; box-shadow: 0px 8px 15px rgba(0, 0, 0, 0.1)">
            <img id="ButtonRelay1" src="/img/IMGBtnOn.png" onclick="ToggleRelay()" width="100" height="100">
        </div>
    </div>
    <div data-role='footer' class='ui-footer ui-footer-fixed'>
    </div>
</div>
<script>
    Highcharts.chart('Chart1', {
        chart: {
            type: 'spline',
            animation: Highcharts.svg,
            marginRight: 10,
            events: {
                load: function () {
                    var seriesNhietDo = this.series[0];
                    var seriesDoAm = this.series[1];
                    var seriesAnhSang = this.series[2];
                    var lastx;
                    setInterval(function () {
                        var url = 'data.php';
                        $.getJSON(url).done(function(data) {
                            $.each(data, function (index, json) {
                                var thietbi = json.ThietBi;
```

```
            $("#TenThietBi").html("IMEI: " + thietbi);
            var giatrithoigian = json.ThoiGian;
            var date = new Date(giatrithoigian);
            var x = date.getTime();
            var giatrinhietdo = parseFloat(json.NhietDo);
            $("#GiaTriNhietDo").html(giatrinhietdo);
            var giatridoam   = parseFloat(json.DoAm);
            $("#GiaTriDoAm").html(giatridoam);
            var giatrianhsang = parseFloat(json.AnhSang);
            $("#GiaTriAnhSang").html(giatrianhsang);
            console.log(giatrithoigian);
            console.log(x);
            console.log(giatrinhietdo);
            console.log(giatridoam);
            console.log(giatrianhsang);
            if(x != lastx){
                seriesNhietDo.addPoint([x, giatrinhietdo], true, true);
                seriesDoAm.addPoint([x, giatridoam], true, true);
                seriesAnhSang.addPoint([x, giatrianhsang], true, true);
                lastx = x;
            }
        });
    });
    $.getJSON('relay.php').done(function(data) {
        $.each(data, function (index, json) {
            var relay = parseInt(json.Relay1);
            if(relay == 1){
                console.log("ON");
                // $("#ButtonRelay1").html("ON");
```

```
                        document.getElementById('ButtonRelay1').src =
'/img/IMGBtnOn.png';
                    }
                    else if(relay == 0){
                        console.log("OFF");
                        // $("#ButtonRelay1").html("OFF");
                        document.getElementById('ButtonRelay1').src =
'/img/IMGBtnOff.png';
                    }
                });
            });
        }, 1000);
    }
  }
},
time: {
  useUTC: false
},
title: {
  text: ''
},
accessibility: {
  announceNewData: {
    enabled: true,
    minAnnounceInterval: 15000,
    announcementFormatter: function (allSeries, newSeries, newPoint) {
      if (newPoint) {
        return 'New point added. Value: ' + newPoint.y;
      }
```

```
                return false;
            }
        }
    },
    xAxis: {
        type: 'datetime',
        tickPixelInterval: 40
    },
    yAxis: {
        title: {
            text: ''
        }
    },
    plotOptions: {
        line: {
            dataLabels: {
                enabled: false
            },
            enableMouseTracking: true
        }
    },
    tooltip: {
        fontSize: '14px',
        xDateFormat: 'Ngày: %d/%m/%Y <br> Thời gian: %H:%M:%S',
        shared: true
    },
    //Enable label
    legend: {
        enabled: true,
```

```
            itemStyle: {
                fontSize:'10px',
                // font: '20pt Trebuchet MS, Verdana, sans-serif',
                // color: '#A0A0A0'
            }
        },
        //Tắt dấu =
        exporting: {
            enabled: false
        },
        series: [{
            name: 'Nhiệt Độ(°C)',
            data: (function () {
            // generate an array of random data
                var data = [], time = (new Date()).getTime(), i;
                for (i = -19; i <= 0; i += 1) {
                    data.push({
                        x: time + i * 1000,
                        y: 0
                    });
                }
                return data;
            }())
        },{
            name: 'Độ Ẩm(%)',
            data: (function () {
            // generate an array of random data
                var data = [], time = (new Date()).getTime(), i;
                for (i = -19; i <= 0; i += 1) {
```

```
                    data.push({
                        x: time + i * 1000,
                        y: 0
                    });
                }
                return data;
            }())
        },{
            name: 'Ánh Sáng(lx)',
            data: (function () {
            // generate an array of random data
                var data = [], time = (new Date()).getTime(), i;
                for (i = -19; i <= 0; i += 1) {
                    data.push({
                        x: time + i * 1000,
                        y: 0
                    });
                }
                return data;
            }())
        }]
    });
  </script>
 </body>
</html>
```

index.js

```
function ToggleRelay(){
```

```
    $.ajax({
        url: "toggleRelay.php",
        contentType: "application/json; charset=utf-8",
        type: "GET",
        complete: function () {
            console.log("Toggle OK");
            $.getJSON('relay.php').done(function(data) {
                $.each(data, function (index, json) {
                    var relay = parseInt(json.Relay1);
                    if(relay == 1){
                        console.log("ON");
                        document.getElementById('ButtonRelay1').src =
'/img/IMGBtnOn.png';
                    }
                    else if(relay == 0){
                        console.log("OFF");
                        document.getElementById('ButtonRelay1').src =
'/img/IMGBtnOff.png';
                    }
                });
            });
        },
        error: function (){
            console.log("Toggle Error");
        }
    });
}
```

3.4.3. Code app Android Studio

Ngôn ngữ lập trình được sử dụng: Java

```
package com.example.iotproject.ui.home;
import static java.lang.Thread.sleep;
import android.content.Context;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.http.SslError;
import android.os.Bundle;
import android.os.Handler;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.SslErrorHandler;
import android.webkit.WebResourceError;
import android.webkit.WebResourceRequest;
import android.webkit.WebView;
import android.webkit.WebViewClient;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;

import android.graphics.Bitmap;
import android.widget.ImageView;
import android.widget.TextView;
import com.bumptech.glide.Glide;
import com.example.iotproject.databinding.FragmentHomeBinding;

import java.util.Timer;
import java.util.TimerTask;
public class HomeFragment extends Fragment {
```

```
    private FragmentHomeBinding binding;
    String ShowOrHideWebViewInitialUse = "show";
    public WebView webView;
    private ImageView Icon;
    private ImageView Logo;
    private ImageView GifView;
    private TextView WarningText;
    boolean LoadError = false;
    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        HomeViewModel homeViewModel =
                new ViewModelProvider(this).get(HomeViewModel.class);
        binding = FragmentHomeBinding.inflate(inflater, container, false);
        View root = binding.getRoot();
        webView = binding.webView;
        Icon = binding.imageView2;
        Logo = binding.imageView3;
        GifView = binding.loadinggif;
        WarningText = binding.warning;
        Logo.setVisibility(View.GONE);

Glide.with(this).load("file:///android_asset/LoadingGif3.gif").into(binding.loadinggi
f);
        Timer timer = new Timer();
        timer.schedule(new TimerTask()
        {
            @Override
            public void run() {
            ConnectivityManager connectivityManager = (ConnectivityManager)
getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
            NetworkInfo wifi =
```

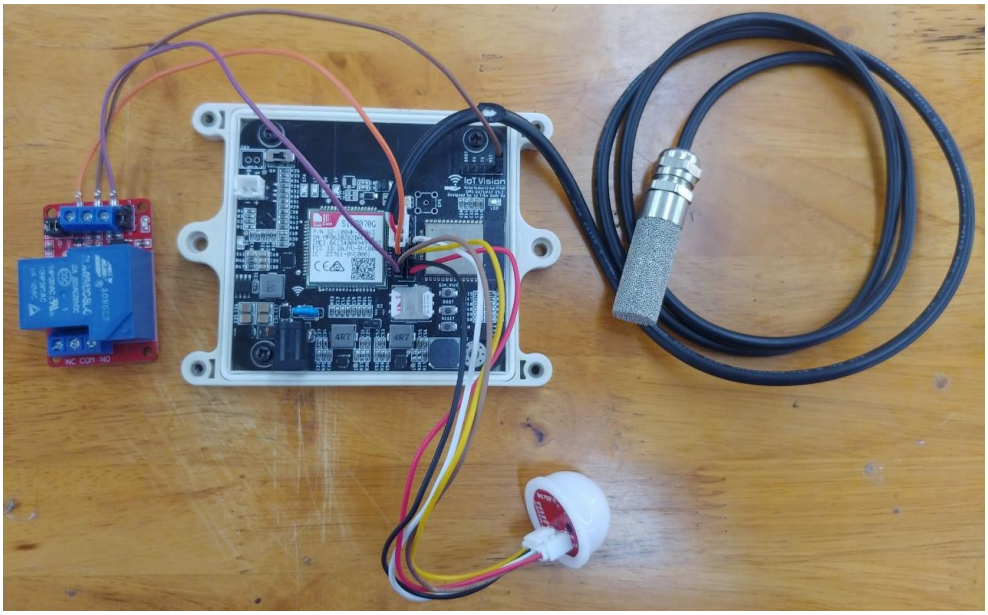
```
connectivityManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
        NetworkInfo mobileNetwork =
connectivityManager.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
        if(wifi.isConnected() || mobileNetwork.isConnected()){
            getActivity().runOnUiThread(()->{
                WarningText.setVisibility(View.GONE);
                if(LoadError){
                    LoadError = false;
                }
            });
        }
        else{
            getActivity().runOnUiThread(()->{
                WarningText.setVisibility(View.VISIBLE);
            });
        }
        }
    }, 0, 500);
    checkConnection();
    return root;
  }
  public void loadWebView(){
    webView.setVisibility(View.INVISIBLE);
    webView.setWebViewClient(new CustomWebViewClient());
    webView.getSettings().setJavaScriptEnabled(true);
    webView.getSettings().setDomStorageEnabled(true);
    webView.loadUrl("https://nckh.assfa.net");
  }
  public void checkConnection(){
    ConnectivityManager connectivityManager = (ConnectivityManager)
```

```
            getActivity().getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo wifi =
connectivityManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
        NetworkInfo mobileNetwork =
connectivityManager.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
        if(wifi.isConnected()){
            loadWebView();
        }
        else if (mobileNetwork.isConnected()){
            loadWebView();
        }
        else{
            webView.setVisibility(View.GONE);
            GifView.setVisibility(View.GONE);
            WarningText.setVisibility(View.VISIBLE);
        }
    }
    // This allows for a splash screen
    // (and hide elements once the page loads)
    private class CustomWebViewClient extends WebViewClient implements
com.example.iotproject.ui.home.CustomWebViewClient {
        @Override
        public void onPageStarted(WebView webview, String url, Bitmap favicon) {
            WarningText.setVisibility(View.GONE);
            webview.setVisibility(View.INVISIBLE);
        }
        @Override
        public void onPageFinished(WebView view, String url) {
            WarningText.setVisibility(View.GONE);
            Handler handler = new Handler();
            handler.postDelayed(new Runnable() {
```

```
            @Override
            public void run() {
                //do after delay
                Icon.setVisibility(View.GONE);
                Logo.setVisibility(View.GONE);
                GifView.setVisibility(View.GONE);
//                  WarningText.setVisibility(View.VISIBLE);
                view.setVisibility(View.VISIBLE);
            }
        }, 500);
        super.onPageFinished(view, url);
    }
    @Override
    public void onReceivedSslError(WebView view, SslErrorHandler handler,
SslError error) {
        handler.proceed();
    }
    @Override
    public void onReceivedError(WebView view, WebResourceRequest request,
WebResourceError error){
        WarningText.setVisibility(View.VISIBLE);
        LoadError = true;
    }
  }
  @Override
  public void onDestroyView() {
    super.onDestroyView();
    binding = null;
  }
}
```

CHƯƠNG 4: KẾT LUẬN

Nhóm đã thiết kế thành công một hệ thống giám sát nhiệt độ, độ ẩm, ánh sáng và điều khiển thiết bị điện trong nhà. Hệ thống bao gồm một phần cứng đọc cảm biến SHT31 (nhiệt độ, độ ẩm), BH1750 (ánh sang) và điều khiển một relay bật tắt thiết bị điện



Hình 26: Phần cứng hệ thống

Và một app giám sát với ba khung giám sát nhiệt độ, độ ẩm, ánh sáng, một biểu đồ vẽ giá trị theo thời gian thực và một nút bấm để điều khiển thiết bị.

Hình 27: Giao diện app giám sát

TÀI LIỆU THAM KHẢO

[1] Random Nerd Tutorials, “ESP32 Pinout Reference: Which GPIO pins should you use?” Available: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

[2] Android.com, “Build web apps in WebView” Available: https://developer.android.com/develop/ui/views/layout/webapps/webview

[3] Highcharts, “Highcharts Demos” Available: https://www.highcharts.com/demo

[4] Highcharts, “Live data from dynamic CSV” Available: https://www.highcharts.com/demo/live-data

[5] W3Schools, “JavaScript Tutorial” Available: https://www.w3schools.com/js/default.asp

[6] W3Schools, “HTML Tutorial” Available: https://www.w3schools.com/html/default.asp

[7] W3Schools, “PHP Tutorial” Available: https://www.w3schools.com/php/default.asp

[8] W3Schools, “Java Tutorial” Available: https://www.w3schools.com/java/default.asp

[9] Vina FE, “CHUẨN GIAO TIẾP I2C LÀ GÌ” Available: https://dientutuonglai.com/chuan-giao-tiep-i2c-la-gi.html

[10] Circuit Basics, “BASICS OF UART COMMUNICATION” Available: https://www.circuitbasics.com/basics-uart-communication/

[11] Piyu Dhaker, “Introduction to SPI Interface” Available: https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html

[12] Nguyễn Việt Hùng, Nguyễn Ngô Lâm, Nguyễn Văn Phúc, “Giáo trình kỹ thuật truyền số liệu”, 09/2011

[13] Nguyễn Trường Duy, Võ Đức Dũng, Nguyễn Thanh Hải, Nguyễn Duy Thảo, “Giáo trình kỹ thuật số”, 2020