

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

KHOA ĐIỆN ĐIỆN TỬ

BỘ MÔN KỸ THUẬT MÁY TÍNH - VIỄN THÔNG



**ĐỒ ÁN 2**

**MẠCH QUANG BÁO THU THẬP VÀ HIỂN THỊ THÔNG TIN MÔI TRƯỜNG TẠI CÁC KHU CÔNG NGHIỆP, KHU DÂN CƯ**

NGÀNH CÔNG NGHỆ KỸ THUẬT MÁY TÍNH

Sinh viên: **NGUYỄN VĂN KIÊN 21119371**

**NGUYỄN TẤN LÂM 21119372**

Giảng viên hướng dẫn:

TP.HỒ CHÍ MINH - 12/2024

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

KHOA ĐIỆN ĐIỆN TỬ

BỘ MÔN KỸ THUẬT MÁY TÍNH - VIỄN THÔNG



**ĐỒ ÁN 2**

**MẠCH QUANG BÁO THU THẬP VÀ HIỂN THỊ THÔNG TIN MÔI TRƯỜNG TẠI CÁC KHU CÔNG NGHIỆP, KHU DÂN CƯ**

NGÀNH CÔNG NGHỆ KỸ THUẬT MÁY TÍNH

Sinh viên: **NGUYỄN VĂN KIÊN 21119371**

**NGUYỄN TẤN LÂM 21119372**

Giảng viên hướng dẫn: **PGS.TS.**

TP.HỒ CHÍ MINH - 12/2024

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

Đề tài: Mạch quang báo thu thập và hiển thị thông tin môi trường tại các khu công nghiệp, khu dân cư

Sinh viên thực hiện: Nguyễn Văn Kiên MSSV: 21119371

Nguyễn Tấn Lâm MSSV: 21119372

Giáo viên hướng dẫn: PGS.TS.

Nhận xét của giáo viên hướng dẫn:

…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….…………………………………………………………………………………….

……………………………………………………………………………………

**LỜI CẢM ƠN**

Người thực hiện xin gửi lời cảm ơn chân thành đến các thầy, cô hiện đang là giảng viên của **Khoa Điện – Điện Tử** đã giảng dạy, tạo điều kiện, môi trường học tập, đóng góp ý kiến và cung cấp cho người thực hiện những kiến thức cơ bản, cần thiết để nhóm có thể thực hiện quá trình nghiên cứu.

Đặc biệt, người thực hiện xin gửi lời cảm ơn đến **thầy** đã giúp đỡ về kiến thức và tài liệu tham khảo để hoàn thành tốt đồ án môn học 2 này. Trong suốt quá trình thực hiện, thầy đã tận tình chỉ dẫn để người thực hiện đi đúng hướng và giúp chỉnh sửa những thiếu sót trong quá trình nghiên cứu và thực hiện đồ án môn học này.

Trong quá trình thực hiện khó tránh khỏi sai sót rất mong Thầy, Cô bỏ qua và có những góp ý, nhận xét giúp người thực hiện hoàn thiện hơn.

Người thực hiện xin chân thành cảm ơn!

**TÓM TẮT**

Đề tài"Mạch quang báo thu thập và hiển thị thông tin môi trường tại các khu công nghiệp, khu dân cư" nhằm xây dựng một hệ thống giám sát và cảnh báo môi trường, giúp người dân nhận biết các thông số như nhiệt độ, độ ẩm, độ sáng, độ ồn và chất lượng không khí. Hệ thống tích hợp cảm biến môi trường và dữ liệu thời tiết trực tuyến, hiển thị thông tin qua tấm LED P5 và ứng dụng chạy trên nền tảng Window, hỗ trợ theo dõi thời tiết và các biến đổi đột ngột. Ngoài ra, hệ thống có phần mềm giám sát qua Internet, gửi cảnh báo khi thông số vượt ngưỡng an toàn, đảm bảo cập nhật nhanh chóng và liên tục. Đề tài hướng đến tính chính xác, tin cậy và khả năng cảnh báo hiệu quả, góp phần nâng cao chất lượng sống tại các khu công nghiệp và khu dân cư.

# 

**MỤC LỤC**

CHƯƠNG 1: TỔNG QUAN 1

1.1 GIỚI THIỆU 1

1.2 MỤC TIÊU ĐỀ TÀI 2

1.3 GIỚI HẠN ĐỀ TÀI 2

1.4 PHƯƠNG PHÁP NGHIÊN CỨU 3

1.5 BỐ CỤC QUYỂN BÁO CÁO 4

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 5

2.1. MODULE LED P5 5

2.1.1 Cấu tạo của module LED P5 5

2.1.2 Nguyên lý hoạt động và phương pháp quét LED 8

2.1.3 Xây dựng bộ font, biểu tượng và hàm hiển thị 9

2.2 VI ĐIỀU KHIỂN STM32F407VET6 10

2.2.1 Tổng quan về dòng vi điều khiển STM32 10

2.2.2 STM32F411CEU6 11

2.3 ESP32 DEVKIT 13

2.4 CÁC PHẦN MỀM THIẾT KẾ VÀ LẬP TRÌNH PHẦN CỨNG 15

2.4.1 Phần mềm EasyEDA 15

2.4.2 Phần mềm Keil C 16

2.4.3 Phần mềm STM32CubeMX 17

2.4.4 Arduino IDE 18

2.5 CÁC CẢM BIẾN THU THẬP THÔNG TIN 19

2.5.1 Cảm biến nhiệt độ, độ ẩm DHT11 19

2.5.2 Cảm biến khí MQ2 20

2.5.3 Cảm biến ánh sáng quang trở 22

2.5.4 Microphone đa hướng 23

2.6 NỀN TẢNG ĐÁM MÂY FIREBASE 24

2.7 CÁC THƯ VIỆN LẬP TRÌNH VÀ CÔNG CỤ BIÊN DỊCH PHẦN MỀM 26

2.7.1 Thư viện Kivy 26

2.7.2 Thư viện Requests 28

2.7.3 Thư viện Pyrebase 28

2.7.4 Công cụ PyInstaller 28

2.8 Giao thức truyền thông UART 29

CHƯƠNG 3: THIẾT KẾ HỆ THỐNG 32

3.1 YÊU CẦU CỦA HỆ THỐNG 32

3.2 ĐẶC TẢ HỆ THỐNG 33

3.2.1 Sơ đồ khối của hệ thống 33

3.2.2 Chức năng của từng khối 33

3.3 THIẾT KẾ VÀ THI CÔNG HỆ THỐNG 34

3.3.1 Thiết kế chức năng 34

3.3.2 Thiết kế mạch bộ điều khiển trung tâm 36

3.3.3 Lưu đồ giải thuật 38

3.3.4 Thiết kế giao diện 42

CHƯƠNG 4: KẾT QUẢ - NHẬN XÉT - ĐÁNH GIÁ 44

4.1 KẾT QUẢ 44

4.1.1 Mach xử lý trung tâm 44

4.1.2 Hoạt động của bảng LED 45

4.1.3 Giao diện người dùng 46

4.2. NHẬN XÉT 47

4.3 ĐÁNH GIÁ 48

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 49

5.1 KẾT LUẬN 49

5.2 Hướng phát triển 49

PHỤ LỤC 51

TÀI LIỆU THAM KHẢO 60

# CHƯƠNG 1: TỔNG QUAN

## 1.1 GIỚI THIỆU

Đề tài "Mạch quang báo thu thập và hiển thị thông tin môi trường tại các khu công nghiệp, khu dân cư" được thực hiện với mục đích xây dựng một hệ thống giám sát và cảnh báo môi trường, cung cấp các thông tin hữu ích và kịp thời cho người dân tại các khu công nghiệp và khu dân cư. Hệ thống này sẽ thu thập các thông số môi trường quan trọng như nhiệt độ, độ ẩm, độ sáng, độ ồn và chất lượng không khí từ các cảm biến môi trường, giúp người dân dễ dàng nhận biết được tình trạng hiện tại của không gian xung quanh.

Ngoài việc thu thập các thông số từ cảm biến, hệ thống còn tích hợp khả năng thu thập thông tin về thời tiết từ các nguồn dữ liệu trực tuyến qua Internet. Thông tin này sẽ được hiển thị lên tấm LED P5, giúp người dân nhận biết tình trạng thời tiết trong khu vực. Điều này sẽ rất hữu ích, đặc biệt trong các trường hợp thay đổi đột ngột của thời tiết như mưa, gió mạnh hay ô nhiễm không khí.

Hệ thống còn được tích hợp với phần mềm giám sát, nơi người dùng có thể theo dõi dữ liệu môi trường và nhận cảnh báo qua Internet. Khi các thông số môi trường vượt qua ngưỡng an toàn, hệ thống sẽ tự động gửi cảnh báo cho người dân hoặc các cơ quan chức năng, giúp họ có biện pháp kịp thời. Việc cập nhật và truyền tải dữ liệu này thông qua Internet đảm bảo rằng thông tin được cập nhật liên tục và nhanh chóng, giúp người dân chủ động trong việc bảo vệ sức khỏe và an toàn của mình.

Với mục tiêu phát triển một hệ thống hoàn chỉnh và hiệu quả, đề tài này không chỉ tập trung vào việc thu thập và hiển thị dữ liệu, mà còn đảm bảo tính chính xác, độ tin cậy và khả năng cảnh báo kịp thời, góp phần nâng cao chất lượng sống cho cộng đồng tại các khu công nghiệp và khu dân cư.

# 

## MỤC TIÊU ĐỀ TÀI

* Mục tiêu chính: Xây dựng một hệ thống giám sát và cảnh báo môi trường tại các khu công nghiệp và khu dân cư, thu thập và hiển thị thông tin môi trường từ cảm biến, cùng với thông tin thời tiết từ Internet, cung cấp cảnh báo kịp thời cho người dân và các cơ quan chức năng qua tấm LED P5 và phần mềm giám sát.
* Mục tiêu cụ thể
  + Thiết kế mạch điện sử dụng STM32 để thu thập các thông số môi trường như nhiệt độ, độ ẩm, độ sáng, độ ồn, và chất lượng không khí từ cảm biến.
  + Tích hợp các cảm biến đo lường môi trường với mạch STM32 để thu thập và xử lý dữ liệu một cách chính xác.
  + Xây dựng hệ thống điều khiển tấm LED P5 để hiển thị thông tin môi trường và cảnh báo cho người dân.
  + Phát triển phần mềm giám sát trên máy tính hoặc thiết bị di động, giúp người dùng theo dõi dữ liệu môi trường và nhận cảnh báo từ hệ thống.
  + Sử dụng ESP32 để kết nối hệ thống với Firebase, giúp truyền tải và cập nhật dữ liệu lên phần mềm giám sát theo thời gian thực.
  + Tạo hệ thống cảnh báo tự động khi các thông số môi trường vượt ngưỡng an toàn, giúp bảo vệ sức khỏe và an toàn cho cộng đồng.
  + Lập trình giao tiếp hiệu quả giữa STM32 và ESP32 để thu thập và cập nhật dữ liệu

## 1.3 GIỚI HẠN ĐỀ TÀI

Đề tài này giới hạn trong việc thiết kế và xây dựng một hệ thống giám sát và cảnh báo môi trường ở mức thử nghiệm với một tấm Led P5, từ đó có thể phát triển lên các hệ thôngs lớn hơn, có thể ứng dụng trong thực tiễn. Hệ thống tập trung vào việc thu thập và hiển thị thông tin môi trường cơ bản như nhiệt độ, độ ẩm, độ sáng, độ ồn và chất lượng không khí, mà không bao gồm việc giám sát các yếu tố khác như khí hậu hoặc các thông số chuyên sâu của từng loại ô nhiễm. Mạch điện sử dụng STM32 chỉ được thiết kế cho việc thu thập và xử lý dữ liệu từ các cảm biến môi trường cơ bản, trong khi ESP32 sẽ sử dụng giao thức Firebase để cập nhật dữ liệu lên phần mềm giám sát. Hệ thống chỉ hoạt động hiệu quả tại các khu vực có kết nối Internet ổn định để truyền tải dữ liệu lên Firebase.

## 1.4 PHƯƠNG PHÁP NGHIÊN CỨU

Để thực hiện đề tài, phương pháp nghiên cứu chủ yếu sẽ là sự kết hợp giữa thiết kế phần cứng và lập trình phần mềm. Các bước nghiên cứu và phát triển cụ thể bao gồm:

**Nghiên cứu và thiết kế phần cứng:**

* Nghiên cứu và lựa chọn các cảm biến môi trường phù hợp để thu thập thông tin về nhiệt độ, độ ẩm, độ sáng, độ ồn và chất lượng không khí.
* Thiết kế mạch PCB sử dụng vi điều khiển STM32 để kết nối với các cảm biến, xử lý dữ liệu và điều khiển tấm LED P5 hiển thị thông tin.

**Lập trình vi điều khiển:**

* Lập trình STM32 để thu thập dữ liệu từ các cảm biến môi trường, xử lý và truyền dữ liệu đến hệ thống hiển thị LED và ESP32.
* Lập trình ESP32 để kết nối với STM32 và Firebase, giúp cập nhật dữ liệu lên phần mềm giám sát và gửi các cảnh báo qua Internet.

**Phát triển phần mềm giám sát:**

Xây dựng phần mềm giám sát trên máy tính hoặc thiết bị di động để người dùng có thể theo dõi dữ liệu môi trường theo thời gian thực và nhận cảnh báo về các vấn đề môi trường.

**Kiểm thử và đánh giá hiệu quả:**

* Kiểm tra hoạt động của hệ thống trong môi trường thực tế tại các khu vực đang sống.
* Đánh giá độ chính xác của các cảm biến, khả năng truyền tải và cập nhật dữ liệu qua Internet, cũng như hiệu quả của các cảnh báo môi trường.

Phương pháp nghiên cứu này sẽ giúp tạo ra một hệ thống cơ bản hoàn chỉnh, đáp ứng được yêu cầu giám sát và cảnh báo môi trường cho cộng đồng.

## 1.5 BỐ CỤC QUYỂN BÁO CÁO

* CHƯƠNG 1: TỔNG QUAN

Giới thiệu về nội dung nghiên cứu của đề tài, đưa ra mục tiêu và nhiệm vụ mà đề tài cần đạt được, xác định rõ đối tượng và phạm vi nghiên cứu cụ thể cho đề tài.

* CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

Trình bày khái quát về đối tượng nghiên cứu, các thuật toán được sử dụng và các kiến thức có liên quan đến quá trình thiết kế, huấn luyện hệ thống.

* CHƯƠNG 3: THIẾT KẾ HỆ THỐNG

Trình bày chi tiết chức năng từng khối làm việc, giải thích cụ thể các cải tiến được sử dụng phát triển hệ thống (chức năng của giao diện và phần mềm).

* CHƯƠNG 4: KẾT QUẢ - NHẬN XÉT - ĐÁNH GIÁ

Đưa ra các kết quả thực nghiệm đã đạt được chứng minh sự hoàn thành công việc của hệ thống.

* CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Tổng kết các vấn đề đã giải quyết đồng thời đưa ra các vấn đề còn tồn tại, từ đó đưa ra phương hướng giải quyết

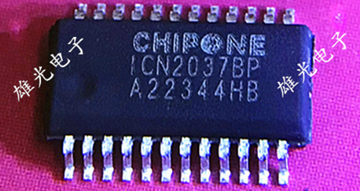
# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

## 2.1. MODULE LED P5

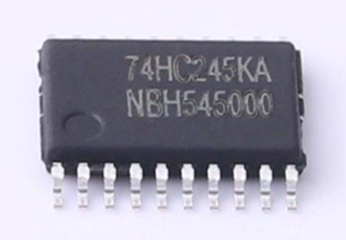
### 2.1.1 Cấu tạo của module LED P5

Cấu tạo của module LED P5 bao gồm:

- 24 IC thanh ghi dịch 16bit ICN2037BP.



*Hình 2.1: IC dịch 16bit ICN2037BP*

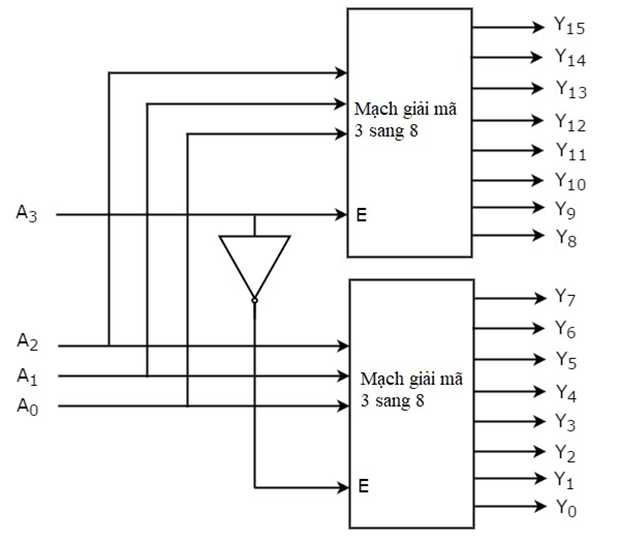
- 2 IC đệm dòng 74HC245KA.

*Hình 2.2: IC đệm dòng 74HC245KA*

- 2 IC giải mã 3 sang 8 74HC138:

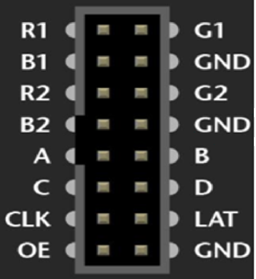


*Hình 2.3: IC giải mã 3 sang 8 74HC138*

Trên ma trận LED P5 full color, sử dụng hai bộ giải mã 3-to-8 để kết hợp thành một bộ giải mã 4-to-16, với bốn chân điều khiển đầu vào.

*Hình 2.4: Bộ giải mã 4 sang 16.*

Bên cạnh đó, tấm LED P5 còn được trang bị hai jumper kết nối 2x8, với một đầu connector làm đầu vào dữ liệu và một đầu connector làm đầu ra (để nối tiếp với tấm LED khác), sử dụng chuẩn HUB75-D.

*Hình 2.5: Bộ điều hợp HUB75-D* 

Bộ điều hợp HUB75-D bao gồm các thành phần chính như sau:

* OE: Đây là chân điều khiển của IC 138. Khi OE = 0, IC 138 không hoạt động, khiến tất cả các hàng LED đều tắt. Ngược lại, khi OE = 1, các tấm LED sẽ được phép sáng, tương tự như một công tắc tổng. Chân này có hai chức năng chính: quét các hàng LED và điều chỉnh độ sáng của bảng LED thông qua tín hiệu xung PWM.
* Các chân A, B, C, D: Những chân này giúp điều khiển việc sáng của các hàng LED, với chỉ hai hàng được bật sáng trong mỗi thời điểm.
* CLK: Chân này tạo xung để đẩy dữ liệu vào IC dịch.
* LAT: Chân này phát xung để xuất dữ liệu từ IC dịch ra ngoài.
* SCLK: Chân này phát xung để xuất dữ liệu từ thanh ghi dịch ra ngoài.
* R1, B1, G1, R2, G2, B2: Đây là các chân truyền tải dữ liệu màu sắc tới LED.

Cả jump DATA-IN và jump DATA-OUT đều có sơ đồ nối tương tự nhau, với các chân OE, A, B, C, D, CLK, LAT của hai jumper được nối chung. Sự khác biệt duy nhất là chân DATA. Jump DATA-OUT cho phép kết nối các module LED lại với nhau, thông qua việc nối DATA-OUT của module này với DATA-IN của module tiếp theo. Điều này khả thi vì các chân R1, G1, B1, R2, G2, B2 là các chân dữ liệu, chuyển thông tin từ dạng nối tiếp sang song song từ IC dịch. Vì thế, khi nối các jumper lại, thực chất là kết nối các IC dịch với nhau, cho phép dữ liệu được truyền liên tục giữa các module.

### 2.1.2 Nguyên lý hoạt động và phương pháp quét LED

Nguyên lý hoạt động:

● Tín hiệu OE khi tích cực mức cao sẽ cho phép xuất dữ liệu ra ở mỗi hàng.

● Tín hiệu chọn hàng A, B, C, D là 2 đường tín hiệu cho phép chọn hàng hiển thị.

● Tín hiệu LAT cho phép chốt dữ liệu ra.

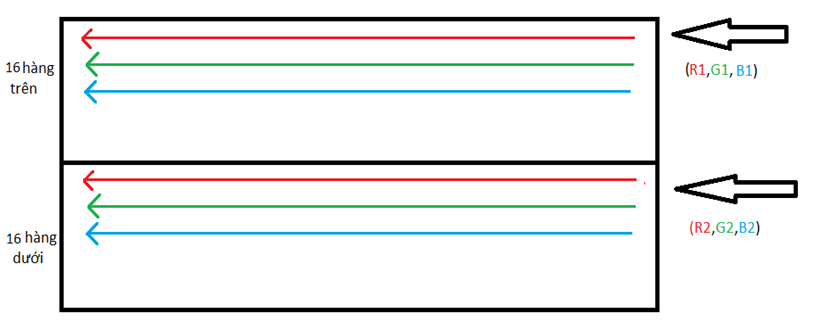
● Tín hiệu CLK tạo xung đưa dữ liệu ra các IC dịch.

● Tín hiệu Data đưa dữ liệu hiển thị ra bảng LED.

Phương pháp quét LED:

Module sử dụng 4 bit điều khiển A, B, C, D => tỉ số quét hàng là 1/16, mà số hàng của module là 32 hàng, vậy tại 1 thời điểm nó sẽ điều khiển 2 hàng

Ngoài ra chân đối với led P5 thì nó các chân data RGB chia làm 2 loại: loại 1 sẽ đảm nhiệm màu của nửa trên và loại 2 sẽ đảm nhiệm data màu cho nửa dưới:



*Hình 2.6: Mô tả nguyên lý quét led P5*

Trong dự án này, chúng ta sẽ lập trình để thực hiện hiển thị với 8 màu cơ bản là tổ hợp các mức thấp và cao của các màu R, G, B. Với 64 led chiều ngang và 32 led chiều dọc cùng với 3 thành phần màu RGB, mỗi pixel sẽ chiếm 3 bit, mỗi bit tương ứng 1 thành phần màu.

Với 1 bit màu và tỉ số scan là 16s, chúng ta cần 16 lần ngắt để quét hết 1 bảng led

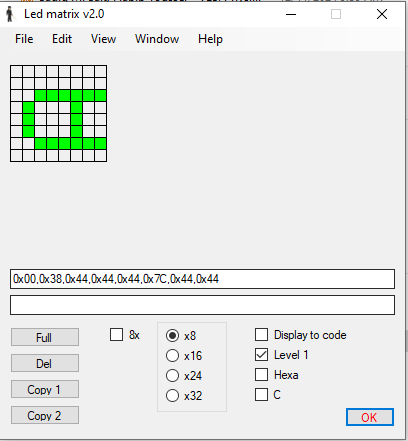
Trước hết ta tạo 1 buffer là 1 mảng chứa các dữ liệu 8 bit, với 3 chiều ứng với 3 màu R, G, B, mỗi chiều sẽ có 32x8 phần tử do mỗi phần tử là 8 bit ứng với 8 điểm ảnh do đó 32x8x8 sẽ tương ứng với 32x64 điểm ảnh.

Sau đó ta sẽ gọi hàm ngắt, mỗi lần ngắt sẽ dịch các bit màu R1, G1, B1, R2, G2, B2 của 2 hàng tương ứng, sau khi dịch hết 64 bit của hàn nửa trên và hàng nửa dưới sẽ gọi hàm chọn hàng và kích hoạt chân OE để Led sáng, thao tác này sẽ lặp lại 16 lần để quét hết 32 hàng LED

### 2.1.3 Xây dựng bộ font, biểu tượng và hàm hiển thị

Để hiển thị các thông tin lên bảng LED P5, ta cần xây dựng bộ font, biểu tượng và các hàm hiển thị để truyền dữ liệu vào mảng chứa dữ liệu, phục vụ cho quá trình quét LED

Với bộ font các ký tự, ta có thể sử dụng các phần mềm hỗ trợ tạo font, cụ thể là các phần mềm hỗ trợ chuyển hiển thị LED ma trận sang mã HEX.

*Hình 2.7: Phần mềm xây dựng mã hiển thị*

Trong dự án này, mỗi chữ được tạo thành từ 7 byte dữ liệu, tương ứng với một không gian 7x8 trên LED ma trận, các biểu tượng phục vụ cho việc trực quan hóa dữ liệu cũng được tạo theo cách tương ứng.

Sau đó xây dựng một hàm chọn vị trí, chọn màu sắc, gửi ký tự, chuỗi vào buffer, quá trình này thực chất đang set các bit màu tương ứng trong buffer.

## 2.2 VI ĐIỀU KHIỂN STM32F407VET6

### 2.2.1 Tổng quan về dòng vi điều khiển STM32

**STM32** là một trong những dòng vi điều khiển 32-bit phổ biến nhất hiện nay, được phát triển bởi STMicroelectronics. Dựa trên kiến trúc ARM Cortex-M, STM32 nổi bật với hiệu năng cao, khả năng tiêu thụ điện năng thấp và tích hợp đa dạng các ngoại vi. Nhờ đó, STM32 được ứng dụng rộng rãi trong nhiều lĩnh vực, từ IoT, công nghiệp tự động hóa, y tế đến ô tô. Ưu điểm vượt trội của STM32 bao gồm chi phí hợp lý, cộng đồng người dùng lớn và khả năng tương thích cao với các linh kiện khác. Với sự đa dạng về dòng sản phẩm và sự hỗ trợ mạnh mẽ từ nhà sản xuất, STM32 là một lựa chọn lý tưởng cho các dự án nhúng, đáp ứng được các yêu cầu ngày càng cao về hiệu suất, độ tin cậy và chi phí.



*Hình 2.8: Dòng vi điều khiển STM32*

### 2.2.2 STM32F411CEU6

STM32F407VET6 là một vi điều khiển 32-bit dựa trên kiến trúc ARM Cortex-M4, được sản xuất bởi STMicroelectronics. Nó được thiết kế để cung cấp hiệu năng cao, khả năng tiêu thụ điện năng thấp và tích hợp nhiều ngoại vi, đáp ứng nhu cầu của các ứng dụng đòi hỏi khắt khe về tốc độ xử lý và độ phức tạp.

*Hình 2.9: Chip STM32F407VET6*

**Đặc điểm nổi bật**

* **Lõi ARM Cortex-M4:** Cung cấp hiệu năng xử lý mạnh mẽ, hỗ trợ đơn vị điểm nổi (FPU) cho các phép tính phức tạp.
* **Bộ nhớ:** 512KB Flash và 192KB SRAM, đủ để lưu trữ chương trình và dữ liệu cho các ứng dụng lớn.
* **Tích hợp ngoại vi phong phú:**
  + ADC: Chuyển đổi tín hiệu tương tự thành số.
  + DAC: Chuyển đổi tín hiệu số thành tương tự.
  + UART, SPI, I2C: Giao tiếp với các thiết bị ngoại vi khác.
  + USB OTG: Kết nối với máy tính hoặc các thiết bị USB khác.
  + Ethernet: Kết nối mạng.
  + Timer, PWM: Điều khiển động cơ, tạo sóng vuông.
  + Và nhiều ngoại vi khác.
* **Tốc độ đồng hồ:** Lên đến 168 MHz, đảm bảo tốc độ xử lý nhanh.
* **Điện áp hoạt động:** 1.8V - 3.6V, linh hoạt trong nhiều ứng dụng.

**Ứng dụng**

STM32F407VET6 được ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm:

* **IoT:** Các thiết bị cảm biến, điều khiển từ xa, hệ thống nhà thông minh.
* **Công nghiệp tự động hóa:** Điều khiển động cơ, cảm biến, giao diện người máy.
* **Y tế:** Thiết bị y tế di động, thiết bị theo dõi sức khỏe.
* **Ô tô:** Hệ thống điều khiển động cơ, cảm biến.
* **Drone:** Điều khiển bay, xử lý hình ảnh.
* **Robot:** Điều khiển robot, xử lý dữ liệu cảm biến.

**Ưu điểm khi sử dụng STM32F407VET6**

* Hiệu năng cao
* Linh hoạt: Tích hợp nhiều ngoại vi, dễ dàng tùy biến.
* Cộng đồng lớn: Có nhiều tài liệu, thư viện và diễn đàn hỗ trợ.
* Chi phí hợp lý: So với các dòng vi điều khiển khác, STM32F407VET6 có mức giá cạnh tranh.

## 2.3 ESP32 DEVKIT

ESP32 là một vi điều khiển hệ thống trên chip (SoC) rất phổ biến, được sử dụng rộng rãi trong các dự án IoT. Nó nổi bật với khả năng kết nối không dây Wi-Fi và Bluetooth, cùng với một loạt các tính năng khác.

**Thông số kỹ thuật nổi bật**

* Bộ vi xử lý: Xtensa LX6 32-bit, có thể cấu hình lõi đơn hoặc lõi kép, chạy ở tốc độ lên đến 240 MHz.
* Bộ nhớ: 520KB SRAM, 448KB ROM, 16KB SRAM RTC.
* Kết nối: Wi-Fi 802.11 b/g/n, Bluetooth Classic và BLE, Ethernet (cần PHY bên ngoài), USB, SPI, I2C, UART, SD/SDIO/MMC.
* Các tính năng khác: ADC, DAC, PWM, GPIO, tăng tốc phần cứng mật mã, ...

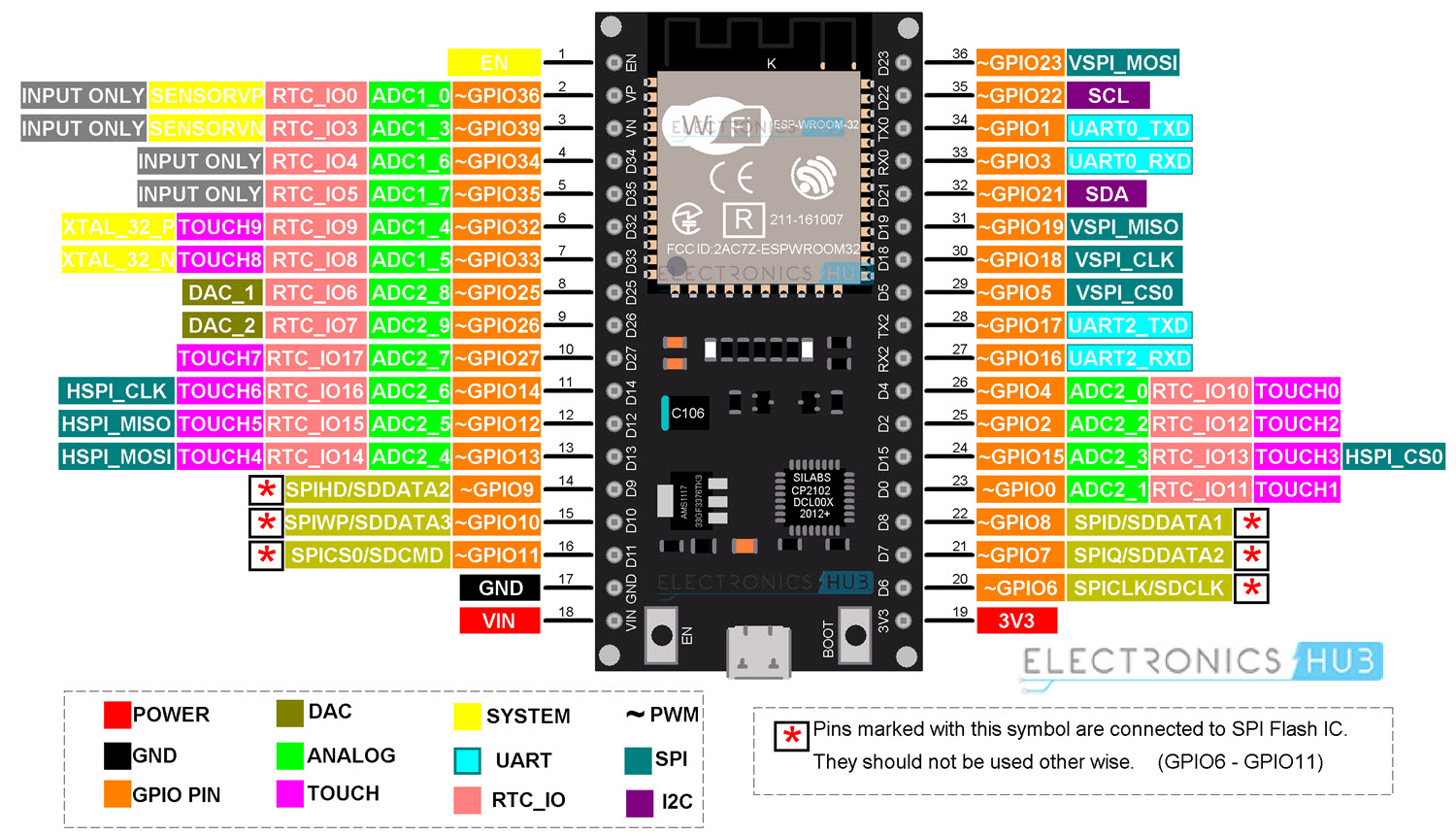
**Ưu điểm**

* Đa năng: Có thể kết nối với nhiều loại cảm biến, module và thiết bị ngoại vi khác.
* Hiệu năng cao: Khả năng xử lý nhanh, đáp ứng được nhiều ứng dụng khác nhau.
* Tiêu thụ điện năng thấp: Tiết kiệm năng lượng, phù hợp với các thiết bị chạy pin.
* Cộng đồng lớn: Có nhiều tài liệu, thư viện và hỗ trợ từ cộng đồng.

**Ứng dụng**

* IoT: Xây dựng các thiết bị thông minh, cảm biến không dây.
* Nhà thông minh: Điều khiển ánh sáng, nhiệt độ, an ninh.
* Robot: Điều khiển robot, thu thập dữ liệu.
* Dự án DIY: Thực hiện các dự án sáng tạo, tự động hóa.

**ESP32 DevKit**

****

*Hình 2.10: Sơ đồ chân ESP32 DevKit*

ESP32 DevKit là một bo mạch phát triển được thiết kế đặc biệt để làm việc với module ESP-WROOM-32. Nó cung cấp một nền tảng thuận tiện cho các nhà phát triển, sinh viên và những người đam mê điện tử để khám phá và xây dựng các dự án IoT.

**Cấu tạo của ESP32 DevKit**

* Module ESP-WROOM-32: Đây là trái tim của bo mạch, chứa chip ESP32, bộ nhớ flash, bộ dao động và các thành phần RF cần thiết.
* Các chân GPIO: Dùng để kết nối với các cảm biến, module và thiết bị ngoại vi khác.
* Cổng USB: Dùng để cấp nguồn và lập trình cho ESP32.
* Đèn LED: Sử dụng để báo hiệu trạng thái hoạt động của bo mạch.
* Cổng cắm chân: Dùng để mở rộng các chức năng của bo mạch.

**Ưu điểm của ESP32 DevKit**

* Dễ sử dụng: Giao diện đơn giản, dễ dàng kết nối với máy tính và bắt đầu lập trình.
* Linh hoạt: Có thể sử dụng để xây dựng nhiều loại dự án khác nhau.
* Cộng đồng lớn: Có nhiều tài liệu, thư viện và ví dụ mã nguồn hỗ trợ.
* Giá thành hợp lý: Phù hợp với các dự án cá nhân và giáo dục.

## 2.4 CÁC PHẦN MỀM THIẾT KẾ VÀ LẬP TRÌNH PHẦN CỨNG

### 2.4.1 Phần mềm EasyEDA

*Hình 2.11: Phần mềm EasyEDA*

**EasyEDA** là một phần mềm thiết kế mạch điện tử và PCB trực tuyến phổ biến, hỗ trợ cả người mới bắt đầu lẫn kỹ sư chuyên nghiệp. Phần mềm này cung cấp một môi trường làm việc thân thiện với các tính năng toàn diện, bao gồm:

* **Schematic Capture**: Vẽ sơ đồ nguyên lý với giao diện trực quan, đi kèm thư viện linh kiện phong phú từ nhà cung cấp và cộng đồng.
* **PCB Design**: Thiết kế mạch in với khả năng chỉnh sửa chi tiết, hỗ trợ nhiều lớp (multi-layer) và các công cụ tự động như kiểm tra lỗi (DRC).
* **Simulation**: Mô phỏng hoạt động của mạch điện tử, giúp kiểm tra và tối ưu hóa thiết kế trước khi sản xuất.
* **Collaboration**: Cho phép làm việc nhóm trực tuyến, chia sẻ thiết kế dễ dàng, phù hợp cho các dự án hợp tác.
* **Integration**: Tích hợp với các nhà sản xuất PCB, giúp đặt hàng và sản xuất trực tiếp từ phần mềm.

EasyEDA có thể sử dụng trên trình duyệt web hoặc thông qua ứng dụng cài đặt, với phiên bản miễn phí và trả phí tùy nhu cầu. Phần mềm này không chỉ hỗ trợ trong thiết kế mạch mà còn tối ưu hóa quy trình từ ý tưởng đến sản xuất, tiết kiệm thời gian và chi phí.

### 2.4.2 Phần mềm Keil C

*Hình 2.12: Phần mềm Keil C*

**Keil uVision 5** là một phần mềm phát triển tích hợp (IDE) phổ biến, được sử dụng rộng rãi để lập trình vi điều khiển. Với giao diện trực quan, dễ sử dụng, Keil uVision 5 hỗ trợ nhiều dòng vi điều khiển khác nhau và cung cấp các công cụ mạnh mẽ như biên dịch, gỡ lỗi, mô phỏng. Phần mềm này giúp các lập trình viên viết, kiểm tra và nạp chương trình vào vi điều khiển một cách nhanh chóng và hiệu quả. Nhờ khả năng tạo ra mã máy tối ưu và hỗ trợ nhiều tính năng nâng cao, Keil uVision 5 là một công cụ không thể thiếu cho những ai đang làm việc trong lĩnh vực nhúng.

### 2.4.3 Phần mềm STM32CubeMX



*Hình 2.13: Phần mềm STM32CubeMX*

STM32CubeMX là một công cụ trực quan, miễn phí do STMicroelectronics cung cấp, hỗ trợ người dùng cấu hình và tạo mã khởi tạo cho vi điều khiển STM32. Với giao diện đồ họa thân thiện, STM32CubeMX giúp đơn giản hóa quá trình phát triển phần mềm nhúng. Thay vì phải viết thủ công các đoạn mã khởi tạo phức tạp, người dùng chỉ cần lựa chọn các ngoại vi cần thiết và cấu hình các thông số trên giao diện, STM32CubeMX sẽ tự động sinh ra mã C tương ứng. Điều này giúp tiết kiệm thời gian, giảm thiểu lỗi và tăng tính hiệu quả trong quá trình phát triển sản phẩm. Không chỉ vậy, STM32CubeMX còn hỗ trợ nhiều dòng vi điều khiển STM32 khác nhau, tích hợp với các IDE phổ biến và được cập nhật thường xuyên để đáp ứng nhu cầu của người dùng

### 2.4.4 Arduino IDE

*Hình 2.14: Phần mềm Arduino IDE*

Arduino IDE là một môi trường phát triển tích hợp (IDE) rất phổ biến, được thiết kế để giúp việc lập trình vi điều khiển trở nên dễ dàng hơn, đặc biệt là đối với những người mới bắt đầu. Khi kết hợp với ESP32, một vi điều khiển mạnh mẽ và linh hoạt, Arduino IDE cung cấp một nền tảng tuyệt vời để thực hiện các dự án IoT và tự động hóa.

Ưu điểm khi sử dụng Arduino IDE cho ESP32:

* Dễ sử dụng: Giao diện trực quan, ngôn ngữ lập trình đơn giản, dễ học.
* Cộng đồng lớn: Có rất nhiều tài liệu, thư viện, và ví dụ mã nguồn sẵn có, giúp tìm kiếm giải pháp và học hỏi từ cộng đồng.
* Linh hoạt: ESP32 có nhiều tính năng và Arduino IDE cung cấp đủ các công cụ để khai thác hết tiềm năng của nó.
* Miễn phí và mã nguồn mở: hoàn toàn có thể tùy chỉnh và mở rộng Arduino IDE theo ý muốn.

Quy trình làm việc cơ bản:

1. Cài đặt: Tải về và cài đặt Arduino IDE, sau đó thêm thư viện ESP32 vào.
2. Chọn board: Trong Arduino IDE, chọn loại board ESP32 mà bạn đang sử dụng.
3. Viết code: Sử dụng ngôn ngữ lập trình Arduino để viết code điều khiển ESP32.
4. Nạp code: Kết nối ESP32 với máy tính và nạp code vào board.

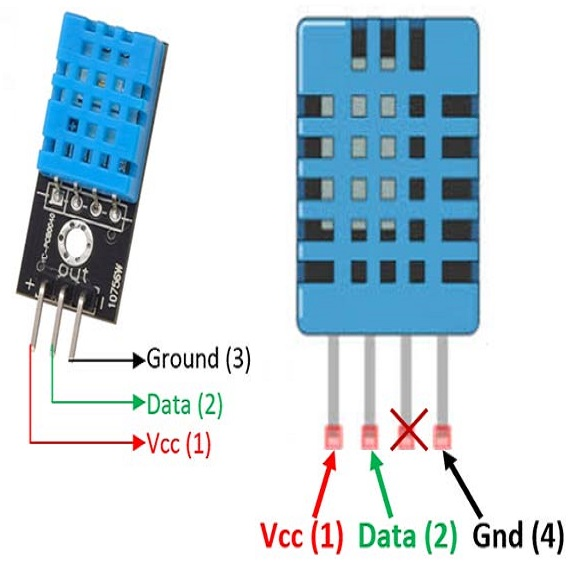
## 2.5 CÁC CẢM BIẾN THU THẬP THÔNG TIN

### 2.5.1 Cảm biến nhiệt độ, độ ẩm DHT11

DHT11 là một cảm biến nhiệt độ và độ ẩm kỹ thuật số giá rẻ, được sử dụng rộng rãi trong các dự án điện tử. Cảm biến này có khả năng đo nhiệt độ và độ ẩm tương đối của môi trường xung quanh và truyền dữ liệu về vi điều khiển qua giao tiếp 1-wire.

Thông số kỹ thuật:

* Điện áp hoạt động: 3.3V - 5V
* Dòng điện tiêu thụ: Ít hơn 2mA
* Khoảng đo nhiệt độ: 0°C - 50°C
* Độ chính xác nhiệt độ: ±2°C
* Khoảng đo độ ẩm: 20% - 90% RH
* Độ chính xác độ ẩm: ±5% RH
* Thời gian lấy mẫu: Khoảng 1 giây/lần

****Sơ đồ chân:

*Hình 2.15: Sơ đồ chân DHT11*

* VCC: Nguồn điện cấp cho cảm biến
* GND: Chân nối đất
* Data: Chân truyền dữ liệu

Ưu điểm:

* Giá rẻ, dễ sử dụng
* Tiêu thụ điện năng thấp
* Kích thước nhỏ gọn

Nhược điểm:

* Độ chính xác không cao
* Tốc độ lấy mẫu chậm

Ứng dụng:

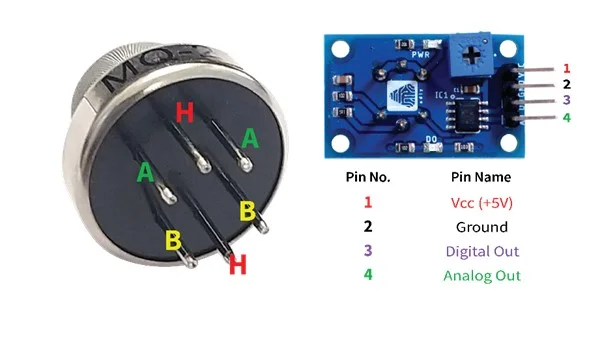
* Điều khiển nhiệt độ và độ ẩm
* Dự án IoT
* Trạm thời tiết mini

### 2.5.2 Cảm biến khí MQ2

**Cảm biến Khí MQ-2** là một loại cảm biến hóa học được thiết kế để phát hiện các loại khí dễ cháy như LPG, propane, metan, hydro và các loại khí khác. Nó cũng nhạy cảm với khói.

**Thông số kỹ thuật:**

* **Điện áp hoạt động:** 5V DC
* **Dòng điện tiêu thụ:** Khoảng 150-200mA
* **Nhiệt độ hoạt động:** 0-50°C
* **Độ nhạy:** Khác nhau tùy thuộc vào loại khí và nồng độ
* **Thời gian đáp ứng:** Vài giây đến vài chục giây

**Sơ đồ chân:**

*Hình 2.16: Sơ đồ chân MQ2*

* **Vcc:** Nguồn điện cấp cho cảm biến
* **GND:** Chân nối đất
* **A0:** Chân tín hiệu analog (độ dẫn điện của cảm biến)

**Nguyên lý hoạt động:**

Khi tiếp xúc với khí gas, độ dẫn điện của lớp vật liệu bán dẫn bên trong cảm biến thay đổi. Sự thay đổi này được chuyển đổi thành tín hiệu điện áp analog trên chân A0.

**Ưu điểm:**

* Giá rẻ, dễ sử dụng
* Độ nhạy cao với nhiều loại khí
* Kích thước nhỏ gọn

**Nhược điểm:**

* Độ chính xác không cao
* Độ nhạy bị ảnh hưởng bởi nhiệt độ, độ ẩm
* Không thể phân biệt chính xác các loại khí

**Ứng dụng:**

* Báo cháy
* Điều khiển thông gió
* Giám sát chất lượng không khí
* Dự án DIY

### 2.5.3 Cảm biến ánh sáng quang trở

*Hình 2.17: Quang trở*

**Quang trở GL5528** là một linh kiện điện tử có khả năng thay đổi giá trị điện trở khi cường độ ánh sáng thay đổi. Khi ánh sáng mạnh, điện trở giảm, và ngược lại.

**Ứng dụng:**

* **Điều khiển đèn tự động:** Tự động bật/tắt đèn theo ánh sáng.
* **Cảm biến ánh sáng cho robot:** Giúp robot tránh chướng ngại vật hoặc theo dõi ánh sáng.
* **Mạch báo động:** Phát hiện sự thay đổi ánh sáng đột ngột.

**Cách sử dụng:**

* **Kết nối:** Nối quang trở với một điện trở và một chân analog của vi điều khiển (ví dụ: Arduino).
* **Đọc giá trị:** Đọc giá trị điện áp trên chân analog để biết cường độ ánh sáng.

### 2.5.4 Microphone đa hướng

*Hình 2.18: Microphone*

Micro 6x2.2mm là một loại micro siêu nhỏ, thường được dùng trong các thiết bị điện tử. Nó có kích thước rất nhỏ, chỉ khoảng 6mm x 2.2mm.

Ưu điểm:

* Nhỏ gọn: Dễ lắp vào các thiết bị nhỏ.
* Nhạy: Thu âm tốt cả âm thanh nhỏ.

Ứng dụng:

* Điện thoại, tai nghe, thiết bị đeo, đồ chơi thông minh,...

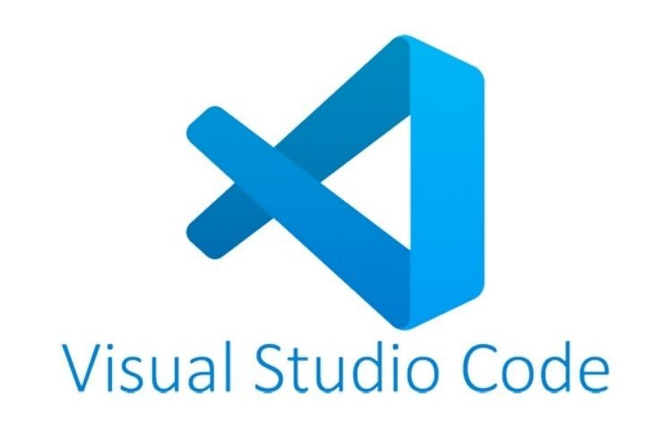
Nhược điểm:

* Dễ bị nhiễu
* Có thể bị méo tiếng ở âm lượng lớn

## 2.6 NỀN TẢNG ĐÁM MÂY FIREBASE

Firebase là một nền tảng phát triển ứng dụng di động và web được cung cấp bởi Google. Nó cung cấp một loạt các dịch vụ và công cụ để giúp các nhà phát triển xây dựng, quản lý và mở rộng ứng dụng của họ một cách dễ dàng và hiệu quả. Một số tính năng chính của Firebase bao gồm:

* **Authentication:** Cung cấp các phương thức xác thực người dùng như email/password, Google, Facebook, Twitter, và nhiều hơn nữa.
* **Realtime Database:** Một cơ sở dữ liệu NoSQL cho phép lưu trữ và đồng bộ dữ liệu trong thời gian thực giữa các người dùng.
* **Cloud Firestore:** Một cơ sở dữ liệu NoSQL linh hoạt và có thể mở rộng, cung cấp các tính năng mạnh mẽ hơn so với Realtime Database.
* **Cloud Storage**: Lưu trữ và chia sẻ tệp tin như hình ảnh, video, và các nội dung khác.
* **Cloud Functions:** Chạy mã backend mà không cần quản lý máy chủ, giúp mở rộng chức năng của ứng dụng.
* **Firebase Hosting:** Cung cấp dịch vụ lưu trữ tĩnh cho các ứng dụng web, với khả năng triển khai nhanh chóng và an toàn.
* **Firebase Analytics:** Cung cấp các công cụ phân tích mạnh mẽ để theo dõi và hiểu hành vi người dùng trong ứng dụng.
* **Firebase Crashlytics:** Giúp theo dõi và sửa lỗi ứng dụng một cách nhanh chóng và hiệu quả.

**2.7 PHẦN MỀM LẬP TRÌNH GIAO DIỆN**

*Hình 2.19:Logo phần mềm Visual Studio Code*

**Visual Studio Code** (VS Code) là một **trình chỉnh sửa mã nguồn đa nền tảng**, được phát triển bởi Microsoft. Phần mềm này được thiết kế để hỗ trợ các lập trình viên viết mã, xây dựng, và gỡ lỗi phần mềm một cách hiệu quả. Với giao diện trực quan, nhiều tính năng mạnh mẽ, và khả năng tùy chỉnh cao, VS Code đã trở thành một trong những công cụ phổ biến nhất dành cho lập trình viên trên toàn thế giới.

Một số tính năng của Visual Studio Code

**Hỗ trợ nhiều ngôn ngữ lập trình bao gồm:** JavaScript, Python, C++, Java, PHP, HTML/CSS, TypeScript,...

**Hỗ trợ IntelliSense:** Tính năng gợi ý mã thông minh, tự động hoàn thành code dựa trên ngữ cảnh và cú pháp.

**Debugger tích hợp:** Hỗ trợ gỡ lỗi trực tiếp trên trình chỉnh sửa, giúp lập trình viên phát hiện và sửa lỗi nhanh chóng.

**Terminal tích hợp:** Tích hợp sẵn terminal bên trong VS Code, hỗ trợ các shell như PowerShell, Bash, hoặc CMD.Giúp chạy các lệnh nhanh mà không cần chuyển đổi cửa sổ.

**Phát triển web:** Với sự hỗ trợ tốt cho HTML, CSS, JavaScript và các framework như React, Angular, hoặc Vue.js.

**Phát triển phần mềm:** Hỗ trợ viết mã bằng nhiều ngôn ngữ lập trình phổ biến.

**Data Science:** Cung cấp môi trường tốt cho Python, Jupyter Notebook, và các công cụ phân tích dữ liệu khác.

**Quản lý DevOps:** Tích hợp tốt với GitHub, Docker, và các công cụ CI/CD.

**Live Share (Cộng tác theo thời gian thực):** Hỗ trợ làm việc nhóm bằng cách chia sẻ code, debug session, hoặc terminal trong thời gian thực.

**Git integration:** Tích hợp sẵn với Git, quản lý mã nguồn dễ dàng.

## 2.7 CÁC THƯ VIỆN LẬP TRÌNH VÀ CÔNG CỤ BIÊN DỊCH PHẦN MỀM

### 2.7.1 Thư viện Kivy

Kivy là một thư viện mã nguồn mở được sử dụng để phát triển giao diện người dùng (UI) và ứng dụng đa nền tảng. Nó được viết bằng Python và hỗ trợ triển khai ứng dụng trên nhiều hệ điều hành như Windows, macOS, Linux, Android, và iOS. Kivy đặc biệt mạnh mẽ khi phát triển các ứng dụng có giao diện người dùng hiện đại, phức tạp và hỗ trợ cảm ứng.

**Đặc điểm chính của Kivy:**

* **Đa nền tảng:** Kivy cho phép viết mã một lần và chạy trên nhiều nền tảng khác nhau, bao gồm desktop, mobile và các thiết bị cảm ứng.
* **Hỗ trợ giao diện cảm ứng:** Kivy được thiết kế để hỗ trợ cảm ứng đa điểm (multi-touch) một cách tự nhiên, rất phù hợp cho các ứng dụng chạy trên màn hình cảm ứng.
* **Hiệu năng cao:** Kivy sử dụng OpenGL ES 2 để render đồ họa, giúp tạo ra giao diện người dùng mượt mà và trực quan.
* **Tùy chỉnh linh hoạt:** Hỗ trợ định nghĩa giao diện và logic tùy chỉnh thông qua các file KV Language (file .kv) hoặc viết trực tiếp bằng Python.
* **Mã nguồn mở:** Được phát hành dưới giấy phép MIT, Kivy cho phép các nhà phát triển tự do sử dụng, chỉnh sửa và đóng góp vào cộng đồng.

**Các thành phần chính của Kivy:**

* **Widget:** Thành phần cơ bản để xây dựng giao diện trong Kivy. Ví dụ:

Button: Nút bấm.

Label: Nhãn văn bản.

TextInput: Ô nhập văn bản.

Slider: Thanh trượt.

* **Layouts:** Các lớp bố cục giúp tổ chức vị trí của các widget. Một số layout phổ biến:

BoxLayout: Sắp xếp widget theo hàng hoặc cột.

GridLayout: Sắp xếp widget theo lưới.

FloatLayout: Vị trí widget theo toạ độ tự do.

AnchorLayout: Căn chỉnh widget ở các góc hoặc trung tâm.

* **KV Language:** Ngôn ngữ định nghĩa giao diện trong Kivy, giúp tách biệt giao diện và logic của ứng dụng.

Ví dụ:

Button:

text: "Click Me!"

size\_hint: 0.5, 0.5

pos\_hint: {"center\_x": 0.5, "center\_y": 0.5}

* **Animation:** Hỗ trợ tạo hoạt ảnh mượt mà cho các widget như thay đổi kích thước, vị trí, màu sắc.
* **Events:** Hệ thống sự kiện mạnh mẽ để xử lý các thao tác như nhấp chuột, cảm ứng đa điểm, hoặc thay đổi trạng thái.

**Một số công cụ và thư viện mở rộng:**

* **KivyMD:** Thư viện mở rộng dựa trên Kivy, hỗ trợ các thành phần giao diện theo Material Design.
* **Buildozer:** Công cụ giúp đóng gói ứng dụng Kivy thành file APK (Android) hoặc IPA (iOS).
* **Pyjnius:** Hỗ trợ tích hợp với API Java khi phát triển ứng dụng Android.

### 2.7.2 Thư viện Requests

Requests là một thư viện HTTP đơn giản và thanh lịch cho Python, được thiết kế để làm cho việc gửi các yêu cầu HTTP trở nên dễ dàng hơn. Nó hỗ trợ các phương thức HTTP như GET, POST, PUT, DELETE và nhiều hơn nữa. Một số tính năng chính của Requests bao gồm:

* Dễ dàng gửi các yêu cầu HTTP và xử lý phản hồi.
* Hỗ trợ các định dạng phản hồi khác nhau như text, binary, JSON.
* Quản lý cookies và session.
* Hỗ trợ xác thực HTTP và SSL.
* Xử lý lỗi và timeout.

### 2.7.3 Thư viện Pyrebase

Pyrebase là một thư viện Python đơn giản để tương tác với Firebase API. Nó cung cấp một giao diện Pythonic để làm việc với các dịch vụ của Firebase như Authentication, Database, Storage. Một số tính năng chính của Pyrebase bao gồm:

* Dễ dàng tích hợp Firebase vào ứng dụng Python.
* Hỗ trợ xác thực người dùng và quản lý token.
* Tương tác với cơ sở dữ liệu Firebase.
* Quản lý lưu trữ tệp trên Firebase.

### 2.7.4 Công cụ PyInstaller

PyInstaller là một công cụ mã nguồn mở giúp đóng gói các ứng dụng Python và tất cả các phụ thuộc của chúng vào một gói duy nhất. Người dùng có thể chạy ứng dụng đã đóng gói mà không cần cài đặt Python interpreter hoặc bất kỳ module nào khác. Một số tính năng chính của PyInstaller bao gồm:

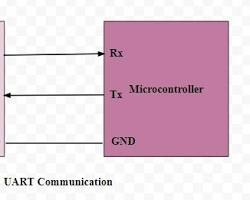
* **Đa nền tảng:** PyInstaller hỗ trợ Windows, macOS và GNU/Linux.
* **Tự động phát hiện phụ thuộc:** PyInstaller phân tích mã nguồn để tìm ra tất cả các module và thư viện cần thiết.
* **Đóng gói thành một file duy nhất:** Có thể chọn đóng gói ứng dụng vào một thư mục hoặc một file thực thi duy nhất.
* **Hỗ trợ nhiều phiên bản Python:** PyInstaller hoạt động tốt với các phiên bản Python từ 3.8 đến 3.13.
* **Tích hợp với các gói lớn:** PyInstaller có thể đóng gói các gói lớn như numpy, PyQt5, PySide2, wxPython và matplotlib mà không cần cấu hình thêm.

## 2.8 Giao thức truyền thông UART

UART (Universal Asynchronous Receiver-Transmitter) là một giao thức truyền thông nối tiếp được sử dụng rộng rãi để kết nối các thiết bị điện tử khác nhau. Nó đơn giản, linh hoạt và dễ triển khai, vì vậy được áp dụng trong nhiều ứng dụng từ vi điều khiển đến các hệ thống nhúng.

**Nguyên lý hoạt động**

UART truyền dữ liệu tuần tự, bit sau bit, trên hai đường dây chính:

* TX (Transmit): Đường truyền dữ liệu từ thiết bị này đến thiết bị khác.
* RX (Receive): Đường nhận dữ liệu từ thiết bị khác.

*Hình 2.20:Chân UART*

Không giống các giao thức đồng bộ, UART không sử dụng một xung đồng hồ chung để đồng bộ hóa việc truyền và nhận dữ liệu. Thay vào đó, nó sử dụng các bit bắt đầu và kết thúc để xác định bắt đầu và kết thúc của một byte dữ liệu.

Quá trình truyền dữ liệu:

1. Bit bắt đầu: Một bit có mức logic 0 được gửi đi để báo hiệu bắt đầu một byte dữ liệu.
2. Dữ liệu: Các bit dữ liệu được gửi theo sau, từ bit ít quan trọng nhất đến bit quan trọng nhất.
3. Bit chẵn lẻ (tùy chọn): Một bit được thêm vào để kiểm tra lỗi.
4. Bit dừng: Một hoặc nhiều bit có mức logic 1 được gửi đi để báo hiệu kết thúc một byte dữ liệu.

**Ưu điểm của UART**

* Đơn giản: Dễ dàng triển khai phần cứng và phần mềm.
* Linh hoạt: Có thể cấu hình tốc độ truyền, số bit dữ liệu, bit chẵn lẻ và số bit dừng.
* Chi phí thấp: Không yêu cầu nhiều thành phần điện tử.
* Phổ biến: Được hỗ trợ bởi hầu hết các vi điều khiển và vi xử lý.

**Nhược điểm của UART**

* Tốc độ truyền: Thường chậm hơn so với các giao thức khác như SPI hoặc I2C.
* Nhiễu: Dễ bị nhiễu do các yếu tố môi trường.
* Khoảng cách truyền: Hạn chế về khoảng cách truyền, thường chỉ vài mét.

**Ứng dụng của UART**

* Kết nối vi điều khiển với máy tính: Truyền dữ liệu từ vi điều khiển đến máy tính để hiển thị hoặc lưu trữ.
* Giao tiếp giữa các thiết bị nhúng: Truyền dữ liệu giữa các module trong một hệ thống.
* Điều khiển các thiết bị ngoại vi: Điều khiển động cơ, cảm biến, màn hình LCD,...
* Gỡ lỗi: Truyền dữ liệu để kiểm tra và gỡ lỗi các hệ thống nhúng.

**Cấu hình UART**

Khi sử dụng UART, bạn cần cấu hình các thông số sau:

* Tốc độ baud: Số bit truyền trong một giây.
* Số bit dữ liệu: Số bit trong một byte dữ liệu.
* Parity: Kiểu kiểm tra lỗi (chẵn, lẻ, không có).
* Số bit dừng: Số bit dừng sau một byte dữ liệu.

Trong dự án này, ta sẽ thực hiện truyền nhận thông tin giữa STM32 và ESP32 qua giao thức UART

# CHƯƠNG 3: THIẾT KẾ HỆ THỐNG

## 3.1 YÊU CẦU CỦA HỆ THỐNG

Để đáp ứng các mục tiêu đã đặt ra ở phần mục tiêu đề tài, nhóm xin đề xuất hệ thống bao gồm các yêu cầu như sau:

1. **Yêu cầu chức năng**

* Thông tin môi trường: Trực quan, kịp thời, chính xác.
* Giao diện người dùng: Dễ nhìn, dễ sử dụng, tiện lợi, trực quan.
* Hiển thị bảng Led: Đầy đủ nội dung, bố cục hợp lý.

1. **Yêu cầu phi chức năng**

* Hiệu năng: Phản hồi chính xác, khả năng mở rộng tốt.
* Khả năng sử dụng: Giao diện thân thiện, dễ sử dụng.

1. **Yêu cầu hệ thống**

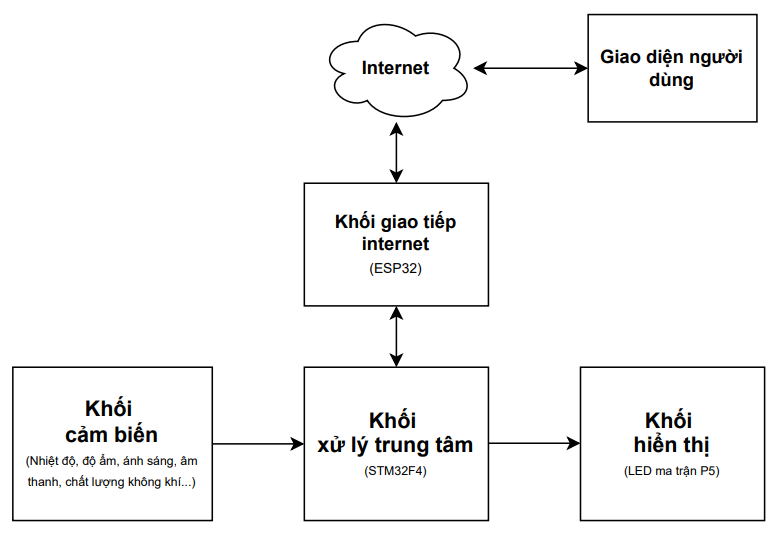
* Công nghệ: Vi điều khiển STM32F4,ESP32, ngôn ngữ lập trình C, Python.
* Cảm biến: Đảm bảo dữ liệu thu được có độ sai lệch nhỏ nhất.
* API tích hợp: API thời tiết để lấy dữ liệu trực tiếp của thời tiết trong vùng theo tọa độ, API Firebase dùng để lưu trữ dữ liệu.

1. **Kết nối và tích hợp**

* API thời tiết và API Firebase: Tương tác qua lại giữa hệ thống vật lý, giao diện người dùng và API thời tiết cùng với API Firebase.

## 3.2 ĐẶC TẢ HỆ THỐNG

### 3.2.1 Sơ đồ khối của hệ thống

****

*Hình 3.1 Sơ đồ khối của hệ thống*

### 3.2.2 Chức năng của từng khối

* **Khối cảm biến:** Khối này chịu trách nhiệm thu thập các dữ liệu môi trường như nhiệt độ, độ ẩm, ánh sáng, âm thanh và chất lượng không khí. Các cảm biến vật lý sẽ đo các thông số này và chuyển đổi chúng thành tín hiệu điện tử mà vi điều khiển có thể xử lý.
* **Khối xử lý trung tâm (STM32F4):** Đây là bộ não của hệ thống. Vi điều khiển STM32F4 nhận dữ liệu từ khối cảm biến, xử lý, tính toán và đưa ra các quyết định dựa trên dữ liệu thu thập được. Đồng thời nó thực hiện chức năng điều khiển hiển thịn LED ma trận
* **Khối hiển thị (LED ma trận P5):** Khối này hiển thị dữ liệu đã được xử lý hoặc thông tin khác cho người dùng. Màn hình LED ma trận P5 cho phép hiển thị thông tin một cách trực quan.
* **Khối giao tiếp internet (ESP32):** Khối này đảm nhiệm việc kết nối hệ thống với internet. ESP32 là một vi điều khiển có khả năng Wi-Fi, cho phép nó gửi dữ liệu đã được xử lý từ khối xử lý trung tâm lên đám mây hoặc một máy chủ. Nó cũng có thể nhận các lệnh điều khiển từ internet.
* **Giao diện người dùng:** Đây là phần giao tiếp giữa người dùng và hệ thống. Thông qua giao diện này, người dùng có thể xem dữ liệu, điều khiển hệ thống hoặc cấu hình các thông số. Giao diện này là một ứng dụng chạy trên Window.

## 3.3 THIẾT KẾ VÀ THI CÔNG HỆ THỐNG

### 3.3.1 Thiết kế chức năng

Dựa vào những yêu cầu về chức năng, nhóm thiết kế các chức năng sau:

**Chức năng thu thập dữ liệu môi trường**

* **Mô tả**: Thu thập thông tin từ các cảm biến đo các thông số môi trường như: Nhiệt độ, độ ẩm, độ sáng, mức độ ồn, chất lượng không khí (PM2.5, PM10, khí gas, v.v.)
* **Thiết bị sử dụng**:
  + Cảm biến DHT22/AM2302 (nhiệt độ và độ ẩm)
  + Cảm biến ánh sáng (BH1750 hoặc tương tự)
  + Microphone đo mức độ ồn
  + Cảm biến khí (MQ-2, MQ-135)
* **Chức năng cụ thể**:
  + Định kỳ đọc dữ liệu từ các cảm biến.
  + Xử lý và chuẩn hóa dữ liệu trước khi sử dụng.

**Chức năng hiển thị lên LED P5**

* **Mô tả**: Hiển thị thông tin môi trường hoặc các thông báo khác lên bảng LED P5 full color.
* **Thiết bị sử dụng**:
  + Bảng LED P5 (sử dụng giao tiếp HUB75-D).
  + Bộ vi điều khiển STM32 để điều khiển LED.
* **Chức năng cụ thể**:
  + Hiển thị dữ liệu dạng văn bản (nhiệt độ, độ ẩm, cảnh báo chất lượng không khí).
  + Hiển thị đồ họa hoặc biểu tượng (biểu tượng thời tiết, mức độ nguy hiểm).

**Chức năng cập nhật dữ liệu từ Internet**

* **Mô tả**: Thu thập thêm dữ liệu dự báo thời tiết Internet.
* **Thiết bị sử dụng**:
  + ESP32 kết nối WiFi.
  + Firebase hoặc API thời tiết (OpenWeatherMap, WeatherStack, v.v.).
* **Chức năng cụ thể**:
  + Gửi yêu cầu đến API để lấy dữ liệu dự báo thời tiết.
  + Đồng bộ hóa dữ liệu thời gian thực với bảng LED.

**Chức năng lưu trữ và cập nhật dữ liệu lên phần mềm máy tính**

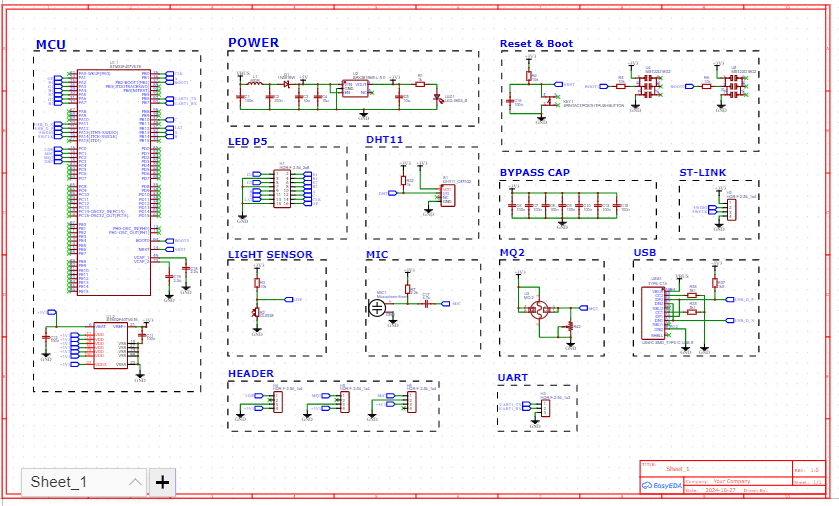
* **Mô tả**: Đồng bộ dữ liệu từ STM32 qua ESP32 để lưu trữ trên Firebase hoặc truyền về phần mềm quản lý trên máy tính.
* **Thiết bị sử dụng**:
  + Firebase hoặc cơ sở dữ liệu trên server (MongoDB, MySQL).
  + Giao tiếp WiFi giữa ESP32 và máy tính.
* **Chức năng cụ thể**:
  + Lưu trữ lịch sử dữ liệu môi trường.
  + Hiển thị dữ liệu và cảnh báo trên giao diện phần mềm máy tính.

**Chức năng cảnh báo môi trường**

* **Mô tả**: Cung cấp thông tin cảnh báo khi các thông số môi trường vượt ngưỡng cho phép.
* **Thiết bị sử dụng**:
  + Buzzer hoặc đèn báo hiệu.
* **Chức năng cụ thể**:
  + Phát tín hiệu cảnh báo khi phát hiện chất lượng không khí xấu hoặc độ ồn cao.
  + Gửi thông báo cảnh báo qua Internet đến phần mềm hoặc ứng dụng.

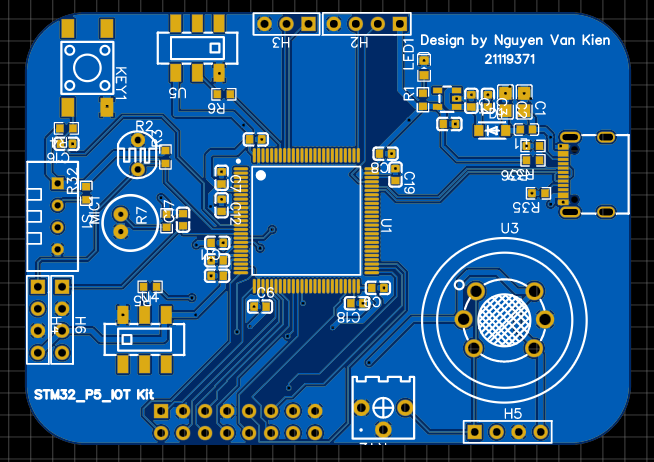
### 3.3.2 Thiết kế mạch bộ điều khiển trung tâm

**3.3.2.1 Thiết kế Schematic**

****

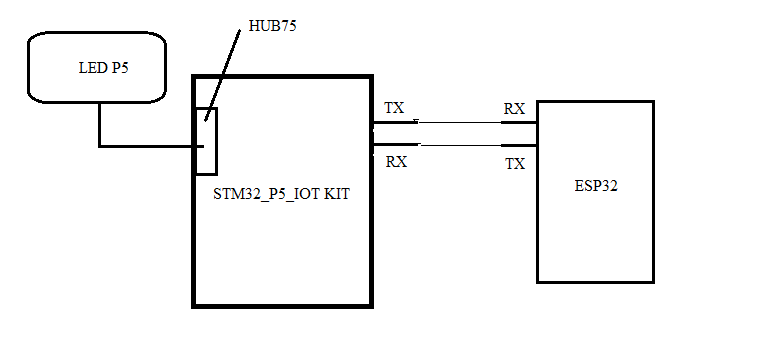
*Hình 3.2:Mạch Schematic bộ điều khiển trung tâm*

**3.3.2.2 Thiết kế mạch PCB**

****

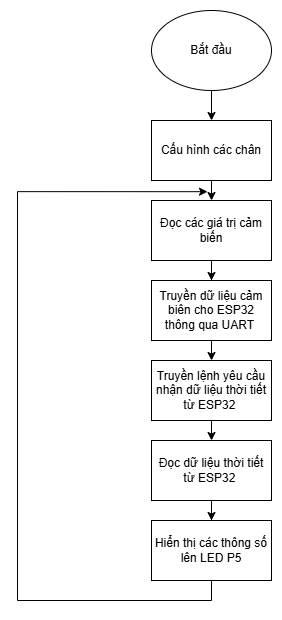
*Hình 3.3:Mạch PCB bộ điều khiển trung tâm*

**3.3.3 Sơ đồ kết nối**

****

*Hình 3.4:Sơ đồ kết nối*

### 3.3.3 Lưu đồ giải thuật

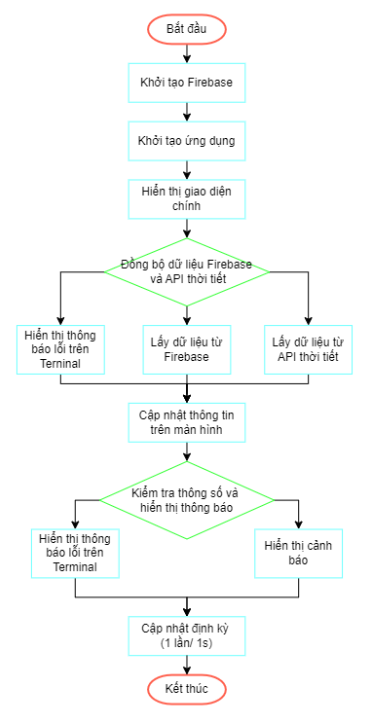
**3.3.1.1 Khối xử lý trung tâm**

*Hình 3.5:Lưu đồ giải thuật tại STM32*

**3.3.1.2 Khối giao tiếp Internet**

*Hình 3.6:Lưu đồ giải thuật tại ESP32*

**3.3.1.3 Khối giao diện người dùng**

****

*Hình 3.7: Lưu đồ giải thuật khối giao diện người dùng*

Giải thích lưu đồ giải thuật khối giao diện người dùng.

**Bắt đầu chương trình:**

* Chương trình bắt đầu bằng cách khởi chạy ứng dụng Kivy.

**Khởi tạo Firebase:**

* Kết nối với Firebase bằng cách sử dụng cấu hình được cung cấp (firebase\_config).
* Tạo kết nối với cơ sở dữ liệu Firebase để lấy và cập nhật dữ liệu.

**Khởi tạo App:**

* Tạo giao diện ứng dụng với các thành phần chính như InfoCard, WeatherCard, và ButtonCard.
* Định nghĩa các sự kiện (nhấn nút, cập nhật dữ liệu).

**Đồng bộ dữ liệu Firebase & API thời tiết:**

* **Lấy dữ liệu từ Firebase:**
  + Các thông số môi trường như tem, hum, atmos, noise, light, và time được lấy từ cơ sở dữ liệu Firebase.
* **Lấy dữ liệu từ API thời tiết:**
  + Gửi yêu cầu đến API thời tiết và nhận về thông tin thời tiết (gió, mây, UV, lượng mưa, tầm nhìn, v.v.).
  + Cập nhật thông tin thời tiết lên Firebase.

**Cập nhật thông tin trên màn hình:**

* Thông tin từ Firebase và API thời tiết được hiển thị trên các InfoCard và WeatherCard.
* Thời gian hiện tại cũng được cập nhật theo chu kỳ.

**Kiểm tra thông số và hiển thị thông báo:**

* Kiểm tra xem các thông số như UV, lượng mưa, độ ô nhiễm, nhiệt độ có vượt ngưỡng hay không.
* Nếu có, hiển thị cảnh báo trên màn hình.

**Chờ cập nhật định kỳ:**

* Sử dụng Clock.schedule\_interval để cập nhật dữ liệu từ Firebase và API thời tiết mỗi giây.

**Kết thúc:**

* Ứng dụng chạy liên tục cho đến khi người dùng dừng.

### 3.3.4 Thiết kế giao diện

**a. Mục đích ứng dụng**

**Theo dõi thời tiết:** Ứng dụng cung cấp thông tin thời tiết hiện tại, bao gồm các thông số như gió, mây, tầm nhìn, UV và lượng mưa.

**Giám sát môi trường:** Ứng dụng cho phép người dùng theo dõi các chỉ số môi trường như nhiệt độ, độ ẩm, độ ô nhiễm, tiếng ồn và ánh sáng.

**b. Giao diện người dùng (UI)**

**Thiết kế thân thiện:** Giao diện được thiết kế đơn giản và dễ sử dụng, với các thẻ thông tin rõ ràng và dễ đọc.

**Thẻ thông tin:** Mỗi thông số môi trường được hiển thị trong các thẻ thông tin riêng biệt, giúp người dùng dễ dàng theo dõi và so sánh.

**Màu sắc hài hòa:** Sử dụng màu sắc nhẹ nhàng và hài hòa để tạo cảm giác thoải mái cho người dùng.

**c. Tính năng chính**

**Cập nhật thời tiết:** Thông tin thời tiết được cập nhật liên tục từ API thời tiết, đảm bảo người dùng luôn có thông tin mới nhất.

**Cảnh báo an toàn:** Ứng dụng cung cấp cảnh báo khi các thông số môi trường vượt quá ngưỡng an toàn, giúp người dùng có biện pháp phòng ngừa kịp thời.

**Thông báo chi tiết:** Khi nhấn vào nút "Xem thông báo", người dùng sẽ nhận được thông tin cụ thể về các thông số nào đã vượt ngưỡng an toàn.

**d. Tính năng tương tác**

**Nút chọn chế độ hiển thị:** Người dùng có thể chọn chế độ hiển thị thông tin khác nhau thông qua các nút vuông, giúp tùy chỉnh trải nghiệm sử dụng.

**Cập nhật thời gian thực:** Ứng dụng tự động cập nhật dữ liệu từ Firebase và thời gian hiện tại, đảm bảo thông tin luôn chính xác và kịp thời.

**e. Công nghệ sử dụng**

**Firebase:** Sử dụng Firebase để lưu trữ và quản lý dữ liệu môi trường, cho phép truy cập và cập nhật dữ liệu dễ dàng.

**Kivy:** Giao diện được xây dựng bằng Kivy, một framework mạnh mẽ cho phát triển ứng dụng di động và desktop.

**Pyrebase:** Thư viện cho phép kết nối với Firebase, giúp kết nối và quản lý dữ liệu trong Firebase Realtime Database

**Requests:** Thư viện cung cấp các mã nguồn để thực hiện các yêu cầu HTTP. Thư viện này giúp gửi và nhận dữ liệu từ các API một cách dễ dàng.

**PyInstaller:** Dùng để đóng gói ứng dụng Python thành các tệp thực thi độc lập cho nhiều hệ điều hành, bao gồm Windows, macOS và Linux.

**f. Lợi ích cho người dùng**

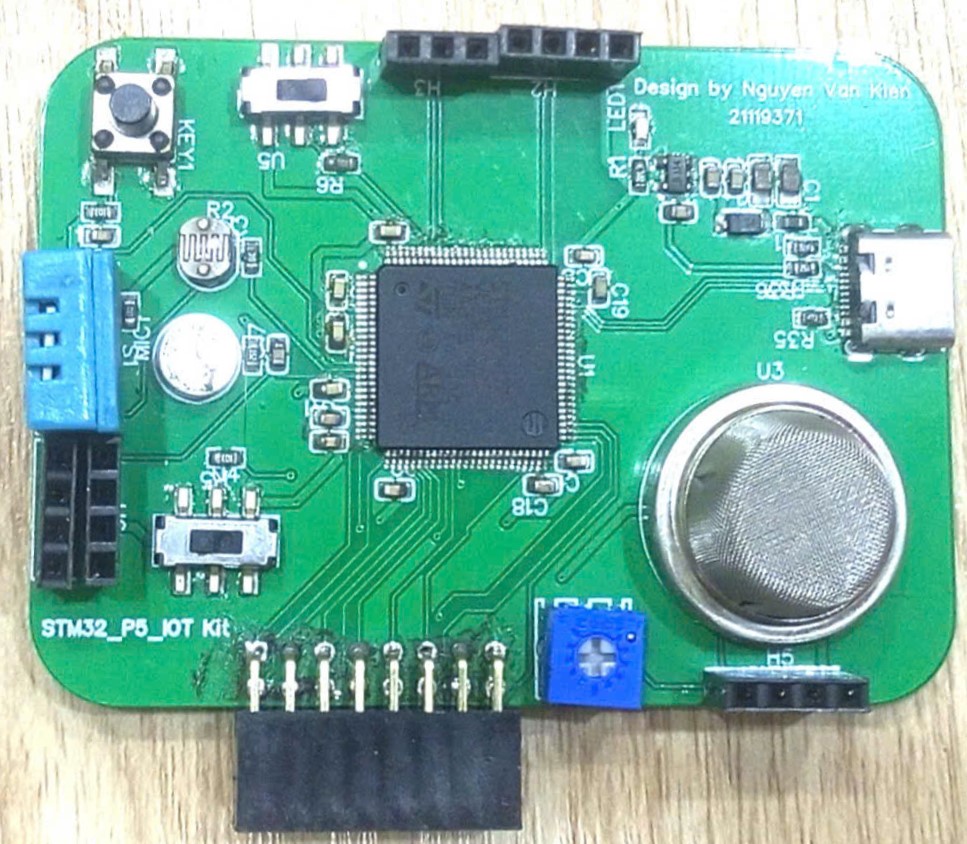
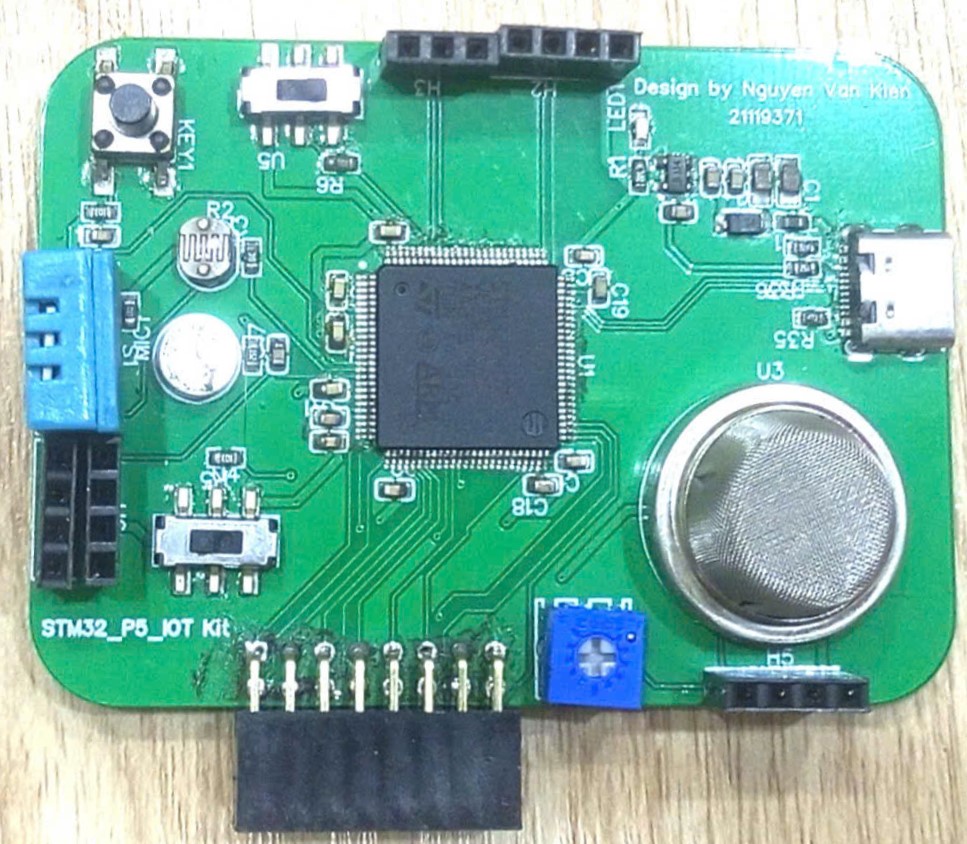
**Nâng cao nhận thức:** Giúp người dùng nhận thức rõ hơn về tình hình thời tiết và môi trường xung quanh.

**Quản lý sức khỏe:** Cung cấp thông tin cần thiết để người dùng có thể đưa ra quyết định hợp lý về sức khỏe và an toàn cá nhân.

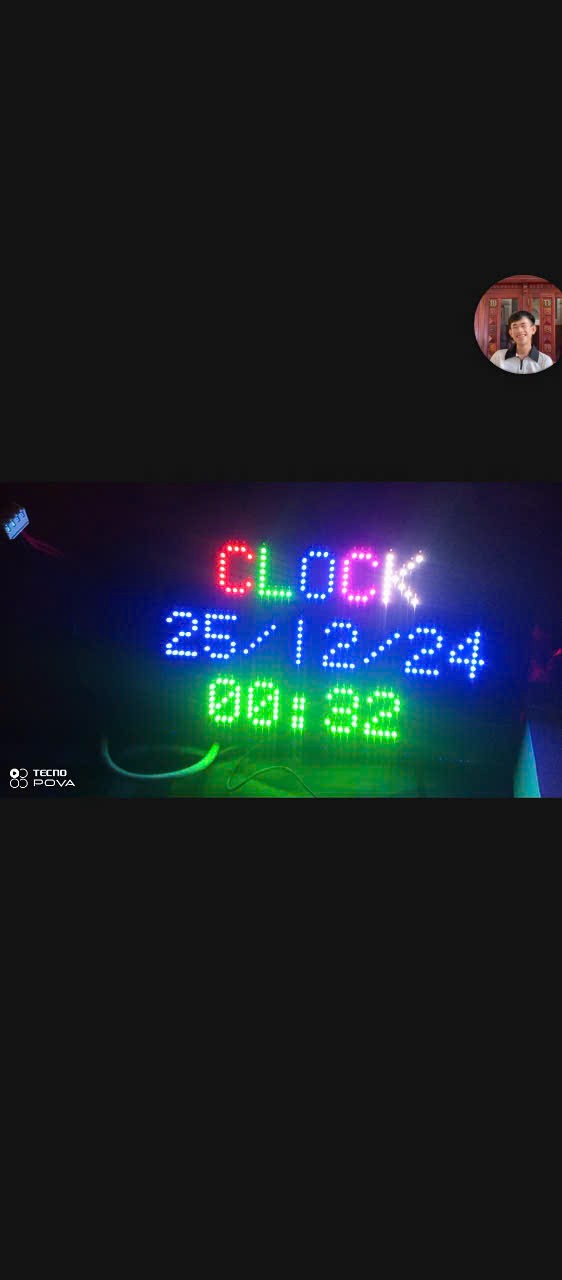
# CHƯƠNG 4: KẾT QUẢ - NHẬN XÉT - ĐÁNH GIÁ

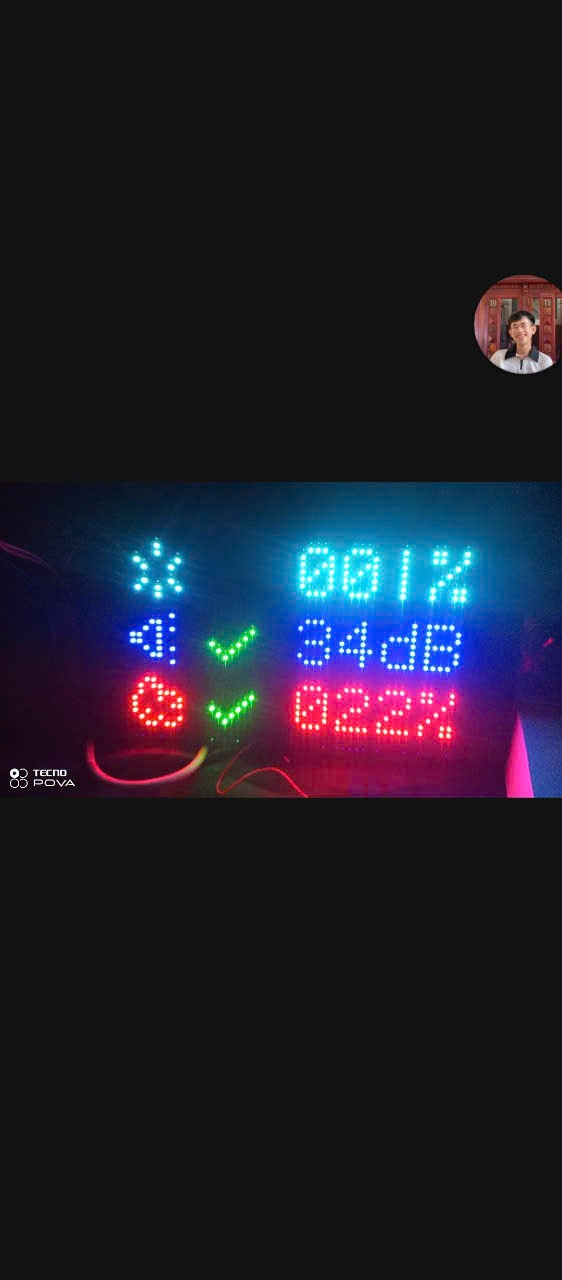
## 4.1 KẾT QUẢ

### 4.1.1 Mach xử lý trung tâm

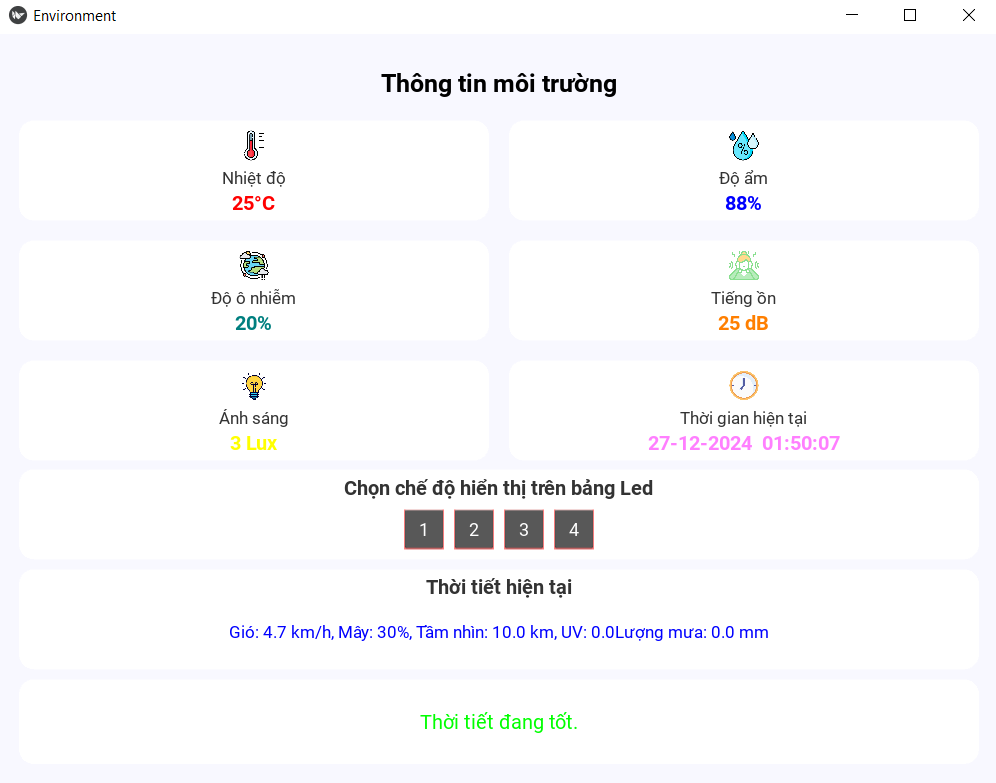
*Hình 4.1:Mạch PCB lúc chưa gắn linh kiện  
Hình 4.2:**Mạch PCB sau khi gắn linh kiện*

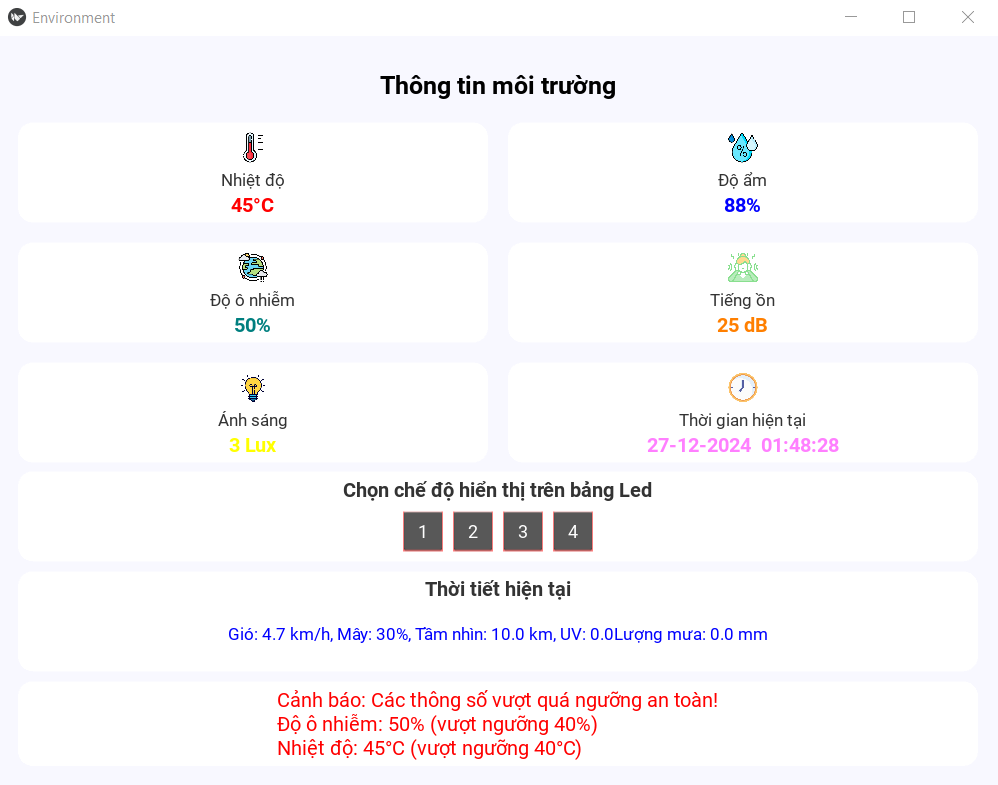
### 4.1.2 Hoạt động của bảng LED

*Hình 4.3:LED hiển thị thông tin thời tiết*

*Hình 4.4:LED hiển thị thời gian*   
*Hình 4.5:LED hiển thị thông tin môi trường*

### 4.1.3 Giao diện người dùng

*Hình 4.6: Giao diện của ứng dụng khi không có cảnh báo*

****

*Hình 4.7: Giao diện ứng dụng khi có cảnh báo về môi trường*

## 4.2. NHẬN XÉT

Ưu điểm:

* Hệ thống hoạt động ổn định, đảm bảo độ chính xác trong thu thập và hiển thị dữ liệu.
* Tích hợp thành công các chức năng cập nhật dữ liệu qua Internet và hiển thị thông tin thời gian thực.
* Giao diện LED P5 hiển thị rõ ràng, dễ theo dõi, phù hợp cho ứng dụng trong khu công nghiệp và khu dân cư.

Hạn chế:

* Tốc độ xử lý dữ liệu có thể bị ảnh hưởng khi số lượng cảm biến hoặc lượng dữ liệu tăng lên.
* Phạm vi kết nối Internet phụ thuộc vào chất lượng mạng tại khu vực triển khai.
* Chưa tối ưu chi phí sản xuất cho quy mô lớn.

## 4.3 ĐÁNH GIÁ

* **Mức độ hoàn thiện**:  
  Hệ thống đã cơ bản đạt được các mục tiêu đã đặt ra, bao gồm thu thập và hiển thị dữ liệu môi trường, cập nhật thông tin qua Internet, và cung cấp cảnh báo trực quan. Các chức năng chính hoạt động ổn định, đáp ứng yêu cầu kỹ thuật và vận hành thực tế.
* **Hiệu quả ứng dụng**:  
  Hệ thống có tiềm năng là cơ sở để phát triển và ứng dụng thực tế cao, đặc biệt tại các khu công nghiệp và khu dân cư, nơi cần giám sát môi trường để bảo vệ sức khỏe cộng đồng. Việc hiển thị thông tin trực quan trên tấm LED P5 giúp người dân dễ dàng tiếp cận thông tin, trong khi kết nối Internet đảm bảo dữ liệu được cập nhật liên tục và chia sẻ hiệu quả.

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1 KẾT LUẬN

Đề tài "Mạch quang báo thu thập và hiển thị thông tin môi trường tại các khu công nghiệp, khu dân cư" đã được thực hiện và hoàn thành với các kết quả nổi bật như sau:

* Thiết kế và triển khai thành công hệ thống thu thập dữ liệu từ các cảm biến môi trường, hiển thị trực quan lên tấm LED P5, và cập nhật dữ liệu qua Internet lên phần mềm.
* Hệ thống hoạt động ổn định, đáp ứng các mục tiêu ban đầu về tính chính xác, độ tin cậy, và khả năng ứng dụng thực tế.
* Sản phẩm đã chứng minh được hiệu quả trong việc cung cấp thông tin môi trường kịp thời và hữu ích cho người dân tại các khu công nghiệp và khu dân cư, góp phần nâng cao ý thức bảo vệ môi trường.

Tuy nhiên, một số hạn chế đã được nhận diện như:

* Chưa tối ưu hiệu suất xử lý dữ liệu khi hệ thống mở rộng quy mô.
* Cần cải thiện tính linh hoạt trong việc triển khai ở các khu vực khác nhau.

## 5.2 Hướng phát triển

Trong tương lai, hệ thống có thể được phát triển thêm theo các hướng sau:

* Mở rộng phạm vi giám sát: Tích hợp thêm các loại cảm biến mới như đo nồng độ bụi mịn PM2.5, khí độc, hoặc chất lượng nước.
* Cải thiện kết nối: Nâng cấp khả năng truyền dữ liệu bằng cách sử dụng các công nghệ mạng hiện đại như LoRa, 5G, hoặc Wi-Fi Mesh để đảm bảo kết nối ổn định ở mọi khu vực.
* Tăng cường khả năng hiển thị: Kết nối nhiều tấm LED để hiển thị thông tin trên quy mô lớn hơn, hoặc triển khai trên màn hình LED với độ phân giải cao.
* Phát triển ứng dụng IoT: Tích hợp ứng dụng di động để người dùng có thể nhận thông báo cảnh báo, theo dõi thông tin thời gian thực, và tùy chỉnh hệ thống từ xa.
* Tối ưu chi phí và năng lượng: Tập trung vào việc giảm chi phí sản xuất và tiêu thụ năng lượng để hệ thống phù hợp với triển khai quy mô lớn trong thực tế.

Hệ thống không chỉ mang lại lợi ích cho cộng đồng mà còn góp phần hỗ trợ chính quyền địa phương trong việc giám sát và quản lý môi trường, hướng đến mục tiêu phát triển bền vững.

# PHỤ LỤC

**Chương trình điều khiển LED P5**P5.h:

#ifndef P5\_H

#define P5\_H

#include "stm32f4xx\_hal.h"

#include "stdio.h"

#include "P5.h"

// Pin definitions

#define R1\_PIN 2

#define G1\_PIN 3

#define B1\_PIN 4

#define R2\_PIN 5

#define G2\_PIN 6

#define B2\_PIN 7

#define A\_PIN 1

#define B\_PIN 14

#define C\_PIN 10

#define D\_PIN 13

#define LAT\_PIN 12

#define OE\_PIN 1

#define CLK\_PIN 0

#define A\_SSET 1<<A\_PIN

#define B\_SSET 1<<B\_PIN

#define C\_SSET 1<<C\_PIN

#define D\_SSET 1<<D\_PIN

#define A\_RSET 1<<(A\_PIN+16)

#define B\_RSET 1<<(B\_PIN+16)

#define C\_RSET 1<<(C\_PIN+16)

#define D\_RSET 1<<(D\_PIN+16)

#define OE\_P GPIOA

#define xuat\_P GPIOB

#define clk\_P GPIOB

#define Control\_P GPIOB

#define data\_PORT GPIOA

// Color definitions

#define BLACK 0x00

#define WHITE 0x07

#define RED 0x01

#define GREEN 0x02

#define BLUE 0x04

#define MAGENTA 0x05

#define YELLOW 0x03

#define CYAN 0x06

#define R\_MASK 0x01

#define G\_MASK 0x02

#define B\_MASK 0x04

// Graphic dimensions

#define Graphic\_x 64

#define Graphic\_y 32

void ngatquetled();

void Matrix\_chonvitri(int x,int y);

void Matrix\_setpx(int32\_t x,int32\_t y,unsigned char color);

void Matrix\_guikitu(unsigned char txt,unsigned char color);

void Matrix\_guichuoi(const char \*s,unsigned char color);

void Matrix\_showIcon(int32\_t x, int32\_t y, int32\_t k, unsigned char color);

void Matrix\_clearArea(int32\_t x, int32\_t y, int32\_t width, int32\_t height) ;

void Matrix\_guichuoi\_MAKE\_COLOR(unsigned char \*s,unsigned char \*color);

#endif // P5\_H

P5.c:

**#include "P5.h"**

**unsigned char dem\_hang = 0;**

**unsigned char buffer\_display[3][32][8] = {0};**

**int MatrixX , MatrixY ;**

**unsigned char font[][7]=**

**{**

**0x00,0x00,0x00,0x00,0x00,0x00,0x00, // 32**

**0x10,0x38,0x38,0x10,0x10,0x00,0x10, //! 33**

**0x6C,0x6C,0x48,0x00,0x00,0x00,0x00, //" 34**

**0x00,0x28,0x7C,0x28,0x28,0x7C,0x28, //# 35**

**0x20,0x38,0x40,0x30,0x08,0x70,0x10, //$ 36**

**0x64,0x64,0x08,0x10,0x20,0x4C,0x4C, //% 37**

**0x20,0x50,0x50,0x20,0x54,0x48,0x34, //& 38**

**0x30,0x30,0x20,0x00,0x00,0x00,0x00, //' 39**

**0x10,0x20,0x20,0x20,0x20,0x20,0x10, //( 40**

**0x20,0x10,0x10,0x10,0x10,0x10,0x20, //) 41**

**0x00,0x28,0x38,0x7C,0x38,0x28,0x00, //\* 42**

**0x00,0x10,0x10,0x7C,0x10,0x10,0x00, //+ 43**

**0x00,0x00,0x00,0x00,0x00,0x30,0x30, //, 44**

**0x00,0x00,0x00,0x7C,0x00,0x00,0x00, //- 45**

**0x00,0x00,0x00,0x00,0x00,0x30,0x30, //. 46**

**0x00,0x04,0x08,0x10,0x20,0x40,0x00, /// 47**

**0x38,0x44,0x4C,0x54,0x64,0x44,0x38, //0 48**

**0x10,0x10,0x10,0x10,0x10,0x10,0x10, //1 49**

**0x38,0x44,0x04,0x18,0x20,0x40,0x7C, //2 50**

**0x38,0x44,0x04,0x38,0x04,0x44,0x38, //3 51**

**0x08,0x18,0x28,0x48,0x7C,0x08,0x08, //4 52**

**0x7C,0x40,0x40,0x78,0x04,0x44,0x38, //5 53**

**0x18,0x20,0x40,0x78,0x44,0x44,0x38, //6 54**

**0x7C,0x04,0x08,0x10,0x20,0x20,0x20, //7 55**

**0x38,0x44,0x44,0x38,0x44,0x44,0x38, //8 56**

**0x38,0x44,0x44,0x3C,0x04,0x08,0x30, //9 57**

**0x00,0x00,0x30,0x30,0x00,0x30,0x30, //: 58**

**0x00,0x00,0x30,0x30,0x00,0x30,0x30, //; 59**

**0x08,0x10,0x20,0x40,0x20,0x10,0x08, //< 60**

**0x00,0x00,0x7C,0x00,0x00,0x7C,0x00, //= 61**

**0x20,0x10,0x08,0x04,0x08,0x10,0x20, //> 62**

**0x38,0x44,0x04,0x18,0x10,0x00,0x10, //? 63**

**0x38,0x44,0x5C,0x54,0x5C,0x40,0x38, //@ 64**

**0x38,0x44,0x44,0x44,0x7C,0x44,0x44, //A 65**

**0x78,0x44,0x44,0x78,0x44,0x44,0x78, //B 66**

**0x38,0x44,0x40,0x40,0x40,0x44,0x38, //C 67**

**0x78,0x44,0x44,0x44,0x44,0x44,0x78, //D 68**

**0x7C,0x40,0x40,0x78,0x40,0x40,0x7C, //E 69**

**0x7C,0x40,0x40,0x78,0x40,0x40,0x40, //F 70**

**0x38,0x44,0x40,0x5C,0x44,0x44,0x3C, //G 71**

**0x44,0x44,0x44,0x7C,0x44,0x44,0x44, //H 72**

**0x38,0x10,0x10,0x10,0x10,0x10,0x38, //I 73**

**0x04,0x04,0x04,0x04,0x44,0x44,0x38, //J 74**

**0x44,0x48,0x50,0x60,0x50,0x48,0x44, //K 75**

**0x40,0x40,0x40,0x40,0x40,0x40,0x7C, //L 76**

**0x44,0x6C,0x54,0x44,0x44,0x44,0x44, //M 77**

**0x44,0x64,0x54,0x4C,0x44,0x44,0x44, //N 78**

**0x38,0x44,0x44,0x44,0x44,0x44,0x38, //O 79**

**0x78,0x44,0x44,0x78,0x40,0x40,0x40, //P 80**

**0x38,0x44,0x44,0x44,0x54,0x48,0x34, //Q 81**

**0x78,0x44,0x44,0x78,0x48,0x44,0x44, //R 82**

**0x38,0x44,0x40,0x38,0x04,0x44,0x38, //S 83**

**0x7C,0x10,0x10,0x10,0x10,0x10,0x10, //T 84**

**0x44,0x44,0x44,0x44,0x44,0x44,0x38, //U 85**

**0x44,0x44,0x44,0x44,0x44,0x28,0x10, //V 86**

**0x44,0x44,0x54,0x54,0x54,0x54,0x28, //W 87**

**0x44,0x44,0x28,0x10,0x28,0x44,0x44, //X 88**

**0x44,0x44,0x44,0x28,0x10,0x10,0x10, //Y 89**

**0x78,0x08,0x10,0x20,0x40,0x40,0x78, //Z 90**

**0x38,0x20,0x20,0x20,0x20,0x20,0x38, //[ 91**

**0x00,0x40,0x20,0x10,0x08,0x04,0x00, //\ 92**

**0x38,0x08,0x08,0x08,0x08,0x08,0x38, //] 93**

**0x10,0x28,0x44,0x00,0x00,0x00,0x00, //^ 94**

**0x00,0x00,0x00,0x00,0x00,0x00,0x7C, //\_ 95**

**0x30,0x30,0x10,0x00,0x00,0x00,0x00, //` 96**

**0x00,0x00,0x38,0x04,0x3C,0x44,0x3C, //a 97**

**0x40,0x40,0x78,0x44,0x44,0x44,0x78, //b 98**

**0x00,0x00,0x38,0x44,0x40,0x44,0x38, //c 99**

**0x04,0x04,0x3C,0x44,0x44,0x44,0x3C, //d 100**

**0x00,0x00,0x38,0x44,0x78,0x40,0x38, //e 101**

**0x18,0x20,0x20,0x78,0x20,0x20,0x20, //f 102**

**0x00,0x3C,0x44,0x44,0x3C,0x04,0x38, //g 103**

**0x40,0x40,0x70,0x48,0x48,0x48,0x48, //h 104**

**0x10,0x00,0x10,0x10,0x10,0x10,0x18, //i 105**

**0x08,0x00,0x18,0x08,0x08,0x48,0x30, //j 106**

**0x40,0x40,0x48,0x50,0x60,0x50,0x48, //k 107**

**0x20,0x20,0x20,0x20,0x20,0x20,0x38, //l 108**

**0x00,0x00,0x68,0x54,0x54,0x44,0x44, //m 109**

**0x00,0x00,0x70,0x48,0x48,0x48,0x48, //n 110**

**0x00,0x00,0x38,0x44,0x44,0x44,0x38, //o 111**

**0x00,0x78,0x44,0x44,0x44,0x78,0x40, //p 112**

**0x00,0x3C,0x44,0x44,0x44,0x3C,0x04, //q 113**

**0x00,0x00,0x58,0x24,0x20,0x20,0x70, //r 114**

**0x00,0x00,0x38,0x40,0x38,0x04,0x38, //s 115**

**0x00,0x20,0x78,0x20,0x20,0x28,0x10, //t 116**

**0x00,0x00,0x48,0x48,0x48,0x58,0x28, //u 117**

**0x00,0x00,0x44,0x44,0x44,0x28,0x10, //v 118**

**0x00,0x00,0x44,0x44,0x54,0x7C,0x28, //w 119**

**0x00,0x00,0x48,0x48,0x30,0x48,0x48, //x 120**

**0x00,0x48,0x48,0x48,0x38,0x10,0x60, //y 121**

**0x00,0x00,0x78,0x08,0x30,0x40,0x78, //z 122**

**0x18,0x20,0x20,0x60,0x20,0x20,0x18, //{ 123**

**0x10,0x10,0x10,0x10,0x10,0x10,0x10, //| 124**

**0x30,0x08,0x08,0x0C,0x08,0x08,0x30, //} 125**

**0x00,0x00,0x28,0x50,0x00,0x00,0x00, //~ 126**

**};**

**const unsigned char icon[7][8] = {**

**{0x30,0x36,0x30,0x36,0x30,0x48,0x48,0x30}, //Tem**

**{0x00,0x00,0x10,0x28,0x44,0x44,0x44,0x38}, //Hum**

**{0x08,0x08,0x41,0x22,0x00,0x22,0x49,0x08}, //Bri**

**{0x02,0x18,0x2A,0xC8,0xC8,0x2A,0x18,0x02}, //Noise**

**{0x30,0x58,0x4E,0x89,0x83,0xA0,0x49,0x36}, //Air**

**{0x00,0x42,0x24,0x18,0x18,0x24,0x42,0x00},**

**{0x00,0x00,0x02,0x04,0x88,0x50,0x20,0x00},**

**};**

**void scan16S(char so)**

**{**

**switch(so)**

**{**

**case 0: Control\_P->BSRR = A\_RSET | B\_RSET | C\_RSET | D\_RSET; return;**

**case 1: Control\_P->BSRR = A\_SSET | B\_RSET | C\_RSET | D\_RSET; return;**

**case 2: Control\_P->BSRR = A\_RSET | B\_SSET | C\_RSET | D\_RSET; return;**

**case 3: Control\_P->BSRR = A\_SSET | B\_SSET | C\_RSET | D\_RSET; return;**

**case 4: Control\_P->BSRR = A\_RSET | B\_RSET | C\_SSET | D\_RSET; return;**

**case 5: Control\_P->BSRR = A\_SSET | B\_RSET | C\_SSET | D\_RSET; return;**

**case 6: Control\_P->BSRR = A\_RSET | B\_SSET | C\_SSET | D\_RSET; return;**

**case 7: Control\_P->BSRR = A\_SSET | B\_SSET | C\_SSET | D\_RSET; return;**

**case 8: Control\_P->BSRR = A\_RSET | B\_RSET | C\_RSET | D\_SSET; return;**

**case 9: Control\_P->BSRR = A\_SSET | B\_RSET | C\_RSET | D\_SSET; return;**

**case 10: Control\_P->BSRR = A\_RSET | B\_SSET | C\_RSET | D\_SSET; return;**

**case 11: Control\_P->BSRR = A\_SSET | B\_SSET | C\_RSET | D\_SSET; return;**

**case 12: Control\_P->BSRR = A\_RSET | B\_RSET | C\_SSET | D\_SSET; return;**

**case 13: Control\_P->BSRR = A\_SSET | B\_RSET | C\_SSET | D\_SSET; return;**

**case 14: Control\_P->BSRR = A\_RSET | B\_SSET | C\_SSET | D\_SSET; return;**

**case 15: Control\_P->BSRR = A\_SSET | B\_SSET | C\_SSET | D\_SSET; return;**

**}**

**}**

**void chuyen(unsigned char byte\_r1, unsigned char byte\_g1, unsigned char byte\_b1,**

**unsigned char byte\_r2, unsigned char byte\_g2, unsigned char byte\_b2)**

**{**

**unsigned char i, mask;**

**for (i = 0; i < 8; i++)**

**{**

**mask = 0x80 >> i;**

**// Ði?u khi?n các chân R1, G1, B1, R2, G2, B2 b?ng giá tr? c?a byte tuong ?ng**

**if (byte\_r1 & mask) {**

**GPIOA->BSRR = (1 << R1\_PIN); // Set R1\_PIN HIGH**

**} else {**

**GPIOA->BSRR = (1 << (R1\_PIN + 16)); // Set R1\_PIN LOW**

**}**

**if (byte\_g1 & mask) {**

**GPIOA->BSRR = (1 << G1\_PIN); // Set G1\_PIN HIGH**

**} else {**

**GPIOA->BSRR = (1 << (G1\_PIN + 16)); // Set G1\_PIN LOW**

**}**

**if (byte\_b1 & mask) {**

**GPIOA->BSRR = (1 << B1\_PIN); // Set B1\_PIN HIGH**

**} else {**

**GPIOA->BSRR = (1 << (B1\_PIN + 16)); // Set B1\_PIN LOW**

**}**

**if (byte\_r2 & mask) {**

**GPIOA->BSRR = (1 << R2\_PIN); // Set R2\_PIN HIGH**

**} else {**

**GPIOA->BSRR = (1 << (R2\_PIN + 16)); // Set R2\_PIN LOW**

**}**

**if (byte\_g2 & mask) {**

**GPIOA->BSRR = (1 << G2\_PIN); // Set G2\_PIN HIGH**

**} else {**

**GPIOA->BSRR = (1 << (G2\_PIN + 16)); // Set G2\_PIN LOW**

**}**

**if (byte\_b2 & mask) {**

**GPIOA->BSRR = (1 << B2\_PIN); // Set B2\_PIN HIGH**

**} else {**

**GPIOA->BSRR = (1 << (B2\_PIN + 16)); // Set B2\_PIN LOW**

**}**

**clk\_P->BSRR=(1<<(CLK\_PIN+16));**

**clk\_P->BSRR=(1<<CLK\_PIN);**

**}**

**}**

**void ngatquetled()**

**{**

**unsigned char i;**

**//OE\_P->BSRR=(1<<(OE\_PIN+16));**

**OE\_P->BSRR=(1<<OE\_PIN);**

**// Truy?n d? li?u ra ma tr?n LED**

**for (i = 0; i < 8; i++) {**

**chuyen(**

**buffer\_display[0][dem\_hang][i], buffer\_display[1][dem\_hang][i], buffer\_display[2][dem\_hang][i],**

**buffer\_display[0][dem\_hang + 16][i], buffer\_display[1][dem\_hang + 16][i], buffer\_display[2][dem\_hang + 16][i]**

**);**

**}**

**OE\_P->BSRR=(1<<(OE\_PIN+16));**

**scan16S(dem\_hang);**

**Control\_P->BSRR=(1<<(LAT\_PIN+16));**

**Control\_P->BSRR=(1<<LAT\_PIN);**

**// Tang hàng dang quét**

**dem\_hang++;**

**if (dem\_hang == 16) dem\_hang = 0;**

**}**

**void Matrix\_chonvitri(int x,int y)**

**{**

**MatrixX=x;**

**MatrixY=y;**

**}**

**void Matrix\_setpx(int32\_t x,int32\_t y,unsigned char color)**

**{**

**if(x>Graphic\_x || y>Graphic\_y || x<0 || y < 0)return;**

**if((color&R\_MASK)!=0)buffer\_display[0][y][x/8] |= (0x80 >> (x%8)); //set R**

**else buffer\_display[0][y][x/8] &= ~(0x80 >> (x%8)); //clear R**

**if((color&G\_MASK)!=0)buffer\_display[1][y][x/8] |= (0x80 >> (x%8)); //set G**

**else buffer\_display[1][y][x/8] &= ~(0x80 >> (x%8)); //clear G**

**if((color&B\_MASK)!=0)buffer\_display[2][y][x/8] |= (0x80 >> (x%8)); //set B**

**else buffer\_display[2][y][x/8] &= ~(0x80 >> (x%8)); //clear B**

**}**

**void Matrix\_guikitu(unsigned char txt,unsigned char color)**

**{**

**int x,y;**

**for(y=MatrixY;y<7+MatrixY;y++)**

**{**

**for(x=MatrixX;x<6+MatrixX;x++)**

**{**

**if ( (font[txt-32][y-MatrixY] & (0x80>>(x-MatrixX))) != 0)Matrix\_setpx(x,y,color);**

**else Matrix\_setpx(x,y,0);**

**}**

**}**

**MatrixX+=6; //sau khi in xong thi tang con tro vi tri len**

**}**

**void Matrix\_guichuoi(const char \*s,unsigned char color)**

**{**

**while(\*s)**

**{**

**Matrix\_guikitu(\*s,color);**

**s++;**

**}**

**}**

**void Matrix\_guichuoi\_MAKE\_COLOR(unsigned char \*s,unsigned char \*color)**

**{**

**while(\*s)**

**{**

**Matrix\_guikitu(\*s,\*color);**

**s++;color++;**

**}**

**}**

**void Matrix\_showIcon(int32\_t x, int32\_t y, int32\_t k, unsigned char color) {**

**int32\_t i, j;**

**for (i = 0; i < 8; i++) {**

**for (j = 0; j < 8; j++) {**

**if (icon[k][i] & (0x80 >> j)) {**

**Matrix\_setpx(x + j, y + i, color);**

**} else {**

**Matrix\_setpx(x + j, y + i, 0); // Clear pixel**

**}**

**}**

**}**

**}**

**void Matrix\_clearArea(int32\_t x, int32\_t y, int32\_t width, int32\_t height) {**

**for (int32\_t i = y; i < y + height; i++) {**

**for (int32\_t j = x; j < x + width; j++) {**

**Matrix\_setpx(j, i, 0); // Clear pixel**

**}**

**}**

**}**

**Thư viện DHT11:** <https://github.com/mesutkilic/DHT11-STM32-Library>

**Full Project:** <https://github.com/KIEN20030806/STM32_P5_IOT>

# TÀI LIỆU THAM KHẢO

[1] Mike Driscoll, (2024), Build a Mobile Application With the Kivy Python Framework, *Real Python*, URL: [Build a Mobile Application With the Kivy Python Framework – Real Python](https://realpython.com/mobile-app-kivy-python/)

[2] Jim, (n.d.), Pyrebase, *GitHub*. URL: [3] requests, URL: [GitHub - thisbejim/Pyrebase: A simple python wrapper for the Firebase API.](https://github.com/thisbejim/Pyrebase)

[3] Alex Ronquillo, (2024), Python's Requests Library (Guide), *Real Python*, URL: [Python's Requests Library (Guide) – Real Python](https://realpython.com/python-requests/)

[4] PyInstaller Team, (n.d.), Using PyInstaller, *PyInstaller Documentation*, URL: [Using PyInstaller — PyInstaller 6.11.1 documentation](https://pyinstaller.org/en/stable/usage.html)

[6]<https://iot47.com/>

[7] <https://dientutuonglai.com/esp32-la-gi.html>

[8] <https://dientutuonglai.com/giao-tiep-uart-la-gi.html>

[9] STM32F407xx datasheet <https://www.st.com/resource/en/datasheet/stm32f405rg.pdf>

[10] DHT11 Humidity & Temperature Sensor - URL: <https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf?srsltid=AfmBOoojOa9Mptqj2H2_qZPx6wQgW3lYg5vwb_Tcf9GTuKYrSVVCVBpB>

[11] GL55 Series Photoresistor - URL: <https://docs.google.com/file/d/0B9cAzTdwG1EaOERyWG96c2x4NGc/edit?resourcekey=0-GqDKIpQgjh6nXIarvD2OgA>

[12] MQ2 datasheet - URL: <https://www.alldatasheet.vn/datasheet-pdf/pdf/1572279/HANWEI/MQ-2.html>