|  |  |
| --- | --- |
| BAN CƠ YẾU CHÍNH PHỦ  **HỌC VIỆN KỸ THUẬT MẬT MÃ** | |
|  |  |
|  | |
|  | |
| **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC** | |
|  | |
| **ĐỀ TÀI**  **THIẾT KẾ VÀ CHẾ TẠO**  **THIẾT BỊ THÔNG MINH HỖ TRỢ CHĂM SÓC SỨC KHỎE** | |
| Sinh viên thực hiện: Nguyễn Văn Phương  Khóa: 4  Chuyên ngành: Hệ thống nhúng và điều khiển tự động  Mã ngành: 7520207  Người hướng dẫn: ThS. Phan Thị Thanh Huyền | |
| **HÀ NỘI – 2025** | |
| BAN CƠ YẾU CHÍNH PHỦ  **HỌC VIỆN KỸ THUẬT MẬT MÃ** | |
|  | |
|  | |
|  | |
| **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC** | |
|  | |
| **ĐỀ TÀI**  **THIẾT KẾ VÀ CHẾ TẠO**  **THIẾT BỊ THÔNG MINH HỖ TRỢ CHĂM SÓC SỨC KHỎE** | |
| Sinh viên thực hiện: Nguyễn Văn Phương  Khóa: 4  Chuyên ngành: Hệ thống nhúng và điều khiển tự động  Mã ngành: 7520207  Người hướng dẫn: ThS. Phan Thị Thanh Huyền | |
| **HÀ NỘI – 2025** | |

# LỜI CẢM ƠN

Sau quá trình học tập và rèn luyện tại Học viện Kỹ thuật mật mã, em lựa chọn thực hiện đề tài đồ án tốt nghiệp “**Thiết kế và chế tạo Thiết bị thông minh hỗ trợ chăm sóc sức khỏe**” dưới sự hướng dẫn của ThS. Phan Thị Thanh Huyền - Giảng viên Khoa Điện tử - Viễn thông. Em xin trân trọng cảm ơn cô vì sự hướng dẫn tận tình và hiệu quả trong quá trình thực hiện đồ án.

Em xin chân thành cảm ơn các giảng viên của Học viện Kỹ thuật mật mã đã nhiệt tình giảng dạy, truyền đạt kiến thức và chia sẻ kinh nghiệm, làm nền tảng cho việc thực hiện tốt đồ án tốt nghiệp.

Cuối cùng, em xin cảm ơn Ban Giám đốc, các phòng chức năng, hệ Quản lý sinh viên đã tạo mọi điều kiện thuận lợi và hỗ trợ em trong quá trình học tập tại Học viện; xin cảm ơn gia đình và người thân đã luôn động viên, khích lệ để em hoàn thành chương trình học tập một cách tốt nhất.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày tháng năm 2025*  **Sinh viên thực hiện**  (*Ký tên và ghi rõ họ tên*)  **Nguyễn Văn Phương** |

LỜI CAM ĐOAN

Em xin cam đoan bản đồ án này do em tự nghiên cứu dưới sự hướng dẫn của cô giáo ThS. Phan Thị Thanh Huyền.

Để hoàn thành đồ án này, em chỉ sử dụng những tài liệu đã ghi trong mục tài liệu tham khảo, ngoài ra không sử dụng bất cứ tài liệu nào khác không được ghi trong mục này.

Nếu sai, em xin chịu mọi hình thức kỷ luật theo quy định của Học viện.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày tháng năm 2025*  **Sinh viên thực hiện**  (*Ký tên và ghi rõ họ tên*)  **Nguyễn Văn Phương** |

# MỤC LỤC

Trang

[LỜI CẢM ƠN i](#_Toc184000008)

[LỜI CAM ĐOAN ii](#_Toc184000009)

[MỤC LỤC iii](#_Toc184000010)

[CÁC KÝ HIỆU, CHỮ VIẾT TẮT vi](#_Toc184000011)

[DANH MỤC BẢNG BIỂU vii](#_Toc184000012)

[DANH MỤC HÌNH ẢNH viii](#_Toc184000013)

[MỞ ĐẦU 1](#_Toc184000014)

[CHƯƠNG 1.](#_Toc184000015)

**TỔNG QUAN VỀ ĐỀ TÀI**

[1.1. Tính thực tiễn của đề tài 3](#_Toc184000016)

[1.2. Tổng quan về chỉ số sức khỏe con người 4](#_Toc184000017)

[1.2.1. Nhịp tim và chỉ số nhịp tim 4](#_Toc184000018)

[1.2.2. Nồng độ oxy trong máu. 8](#_Toc184000019)

[1.2.3. Chỉ số nhiệt độ cơ thể 9](#_Toc184000020)

[1.2.4. Huyết áp 10](#_Toc184000021)

[1.3. Thiết bị y tế theo dõi sức khỏe 11](#_Toc184000022)

[1.3.1. Giới thiệu 11](#_Toc184000023)

[1.3.2. Một số loại thiết bị theo dõi sức khỏe phổ biến 13](#_Toc184000024)

[1.3.3. Ưu điểm và nhược điểm 15](#_Toc184000025)

[1.3.4. Xu hướng phát triển 15](#_Toc184000026)

[1.4. Sơ lược về Internet vạn vật - IoT 17](#_Toc184000027)

[1.4.1. Giới thiệu 17](#_Toc184000028)

[1.4.2. Cấu trúc của hệ thống IoT 18](#_Toc184000029)

[1.4.3. Ứng dụng 19](#_Toc184000030)

[CHƯƠNG 2.](#_Toc184000031)

**CÔNG NGHỆ THIẾT KẾ**

[2.1. Giải pháp thiết kế 23](#_Toc184000034)

[2.1.1. Mục tiêu thiết kế 23](#_Toc184000035)

[2.1.2. Sơ đồ khối 23](#_Toc184000036)

[2.2. Công nghệ phần cứng 24](#_Toc184000037)

[2.2.1. ESP-WROOM-32 24](#_Toc184000038)

[2.2.2. Cảm biến MAX30102 28](#_Toc184000039)

[2.2.3. Cảm biến BME280 29](#_Toc184000040)

[2.2.4. Module SIM 4G A7680C 30](#_Toc184000041)

[2.2.5. Màn hình OLED 32](#_Toc184000042)

[2.2.6. Module hạ áp LM2596 34](#_Toc184000043)

[2.3. Các chuẩn giao tiếp 35](#_Toc184000044)

[2.3.1. Giao tiếp I2C 35](#_Toc184000045)

[2.3.2. Giao tiếp UART 37](#_Toc184000046)

[2.4. Công nghệ phần mềm 39](#_Toc184000047)

[2.4.1. Ngôn ngữ lập trình C 39](#_Toc184000048)

[2.4.2. Adruino IDE 40](#_Toc184000049)

[2.4.3. ThingSpeak 41](#_Toc184000050)

[CHƯƠNG 3.](#_Toc184000051)

**THIẾT KẾ VÀ THI CÔNG**

[3.1. Thiết kế phần cứng 45](#_Toc184000053)

[3.1.1. Sơ đồ nguyên lý của hệ thống 45](#_Toc184000054)

[3.1.2. Thi công thiết bị 47](#_Toc184000055)

[3.2. Thiết kế phần mềm 48](#_Toc184000056)

[3.2.1. Lưu đồ thuật toán và nguyên lý hoạt động 48](#_Toc184000057)

[3.2.2. Tạo tài khoản và giám sát hoạt động của thiết bị trên Thingspeak 50](#_Toc184000058)

[3.2.3. Cấu hình cho thiết bị 53](#_Toc184000059)

[3.3. Triển khai thử nghiệm và đánh giá hoạt động của sản phẩm 54](#_Toc184000060)

[3.3.1. Thử nghiệm hoạt động của thiết bị 54](#_Toc184000061)

[3.3.2. Thử nghiệm hoạt động của tài khoản ThinkSpeak 57](#_Toc184000062)

[3.3.3. Đánh giá hoạt động 58](#_Toc184000063)

[KẾT LUẬN 59](#_Toc184000064)

[TÀI LIỆU THAM KHẢO 60](#_Toc184000065)

[PHỤ LỤC 61](#_Toc184000066)

# CÁC KÝ HIỆU, CHỮ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Tiếng Anh** | **Tiếng Việt** |
| AC | Alternating Current | Dòng điện xoay chiều |
| ADC | Analog-to-Digital Converter | Bộ chuyển đổi tương tự sang số |
| BLE | Bluetooth Low Energy | Bluetooth năng lượng thấp |
| CLK | Clock | Xung clock |
| DAC | Digital-to-Analog Converter | Bộ chuyển đổi số sang tương tự |
| DC | Direct Current | Dòng điện 1 chiều |
| DMA | Direct Memory Access | Truy cập bộ nhớ trực tiếp |
| FPC | Flexible Printed Circuit | Mạch in linh hoạt |
| IC | Intergrated Circuit | Vi mạch tích hợp |
| IDE | Integrated Development Environment | Môi trường phát triển tích hợp |
| IoT | Internet of Things | Internet vạn vật |
| IP | Internet Protocol | Giao thức mạng |
| I2C | Inter-Integrated Circuit | Mạch liên kết giữa các mạch tích hợp |
| PSRAM | Pseudo Static Random Access Memory | Bộ nhớ truy cập ngẫu nhiên tĩnh giả |
| PWM | Pulse Width Modulation | Điều chế độ rộng xung |
| RAM | Random Access Memory | Bộ nhớ truy cập ngẫu nhiên |
| ROM | Read-Only Memory | Bộ nhớ chỉ đọc |
| RTC | Real-Time Clock | Đồng hồ thời gian thực |
| SRAM | Static Random-Access Memory | Bộ nhớ truy cập ngẫu nhiên tĩnh |
| SPI | Serial Peripheral Interface | Giao diện ngoại vi nối tiếp |
| UART | Universal Asynchronous Receiver-Transmitter | Bộ truyền nhận dữ liệu nối tiếp  bất đồng bộ |
| USB | Universal Serial Bus | Chuẩn giao tiếp nối tiếp đa năng |

# DANH MỤC BẢNG BIỂU

Trang

[Bảng 1.1. Nhịp tim của người trưởng thành 6](#_Toc184003580)

[Bảng 1.2. Chỉ số nhịp tim theo độ tuổi của nam và nữ 6](#_Toc184003581)

[Bảng 1.3. Bảng thông số huyết áp theo độ tuổi và giới tính 11](#_Toc184003582)

[Bảng 2.1. Bảng thông số kỹ thuật của module Sim A7680C 31](#_Toc184003583)

[Bảng 2.2. Chức năng chân của module Sim A7680C 32](#_Toc184003584)

[Bảng 2.3. Bảng tập lệnh cơ bản của module Sim A7680C 32](#_Toc184003585)

[Bảng 3.1. Bảng kết nối chân ESP-WROOM-32 với linh kiện và thiết bị 45](#_Toc184003586)

[Bảng 3.2. Bảng tổng hợp kết quả đo 56](#_Toc184003587)

# DANH MỤC HÌNH ẢNH

Trang

[Hình 1.1. Điện tâm đồ 7](#_Toc184004567)

[Hình 1.2. Đo nhịp tim bằng phương pháp hấp thụ quang học 8](#_Toc184004568)

[Hình 1.3. Thiết bị theo dõi sức khỏe thông minh 12](#_Toc184004569)

[Hình 1.4. Đồng hồ thông minh. 13](#_Toc184004570)

[Hình 1.5. Máy đo huyết áp tự động và máy đo nồng độ oxy trong máu 14](#_Toc184004571)

[Hình 1.6. Oura Ring 14](#_Toc184004572)

[Hình 1.7. Đồng hồ thông minh tích hợp trí tuệ nhân tạo 16](#_Toc184004573)

[Hình 1.8. Hệ thống IoT. 17](#_Toc184004574)

[Hình 1.9. Cấu trúc hệ thống IoT. 18](#_Toc184004575)

[Hình 1.10. IoT ứng dụng trong y tế 20](#_Toc184004576)

[Hình 1.11. Thiết bị IoT trong nhà thông minh giúp cuộc sống dễ dàng hơn. 22](#_Toc184004577)

[Hình 2.1. Sơ đồ khối 23](#_Toc184004578)

[Hình 2.2. Module ESP-WROOM-32 24](#_Toc184004579)

[Hình 2.3. Cấu hình của module ESP-WROOM-32 26](#_Toc184004580)

[Hình 2.4. Sơ đồ chức năng các chân của ESP32-WROOM-32 27](#_Toc184004581)

[Hình 2.5. Kích thước của ESP - WROOM - 32 28](#_Toc184004582)

[Hình 2.6. Cảm biến Max30102 28](#_Toc184004583)

[Hình 2.7. Cảm biến BME280 30](#_Toc184004584)

[Hình 2.8. Module Sim 4G A7680C 31](#_Toc184004585)

[Hình 2.9. Màn hình Oled 0.96 inch 33](#_Toc184004586)

[Hình 2.10. Module hạ áp LM2596 34](#_Toc184004587)

[Hình 2.11. Chuẩn giao tiếp I2C 35](#_Toc184004588)

[Hình 2.12. Truyền UART 37](#_Toc184004589)

[Hình 2.13. Ngôn ngữ lập trình C. 39](#_Toc184004590)

[Hình 2.14. Giao diện bắt đầu của Adruino IDE 40](#_Toc184004591)

[Hình 2.15. Giao diện phần mềm Adruino IDE 41](#_Toc184004592)

[Hình 2.16. ThingSpeak 41](#_Toc184004593)

[Hình 2.17. ThingSpeak và Matlab 42](#_Toc184004594)

[Hình 3.2. Sơ đồ nguyên lý của khối nguồn 45](#_Toc184004595)

[Hình 3.3. Sơ đồ nguyên lý của khối xử lý trung tâm 46](#_Toc184004596)

[Hình 3.4. Sơ đồ nguyên lý của khối cảm biến 46](#_Toc184004597)

[Hình 3.5. Sơ đồ nguyên lý của khối SIM 4G 46](#_Toc184004598)

[Hình 3.6. Mạch điều khiển hoàn thiện 47](#_Toc184004599)

[Hình 3.7. Lắp đặt mạch in vào vỏ sản phẩm 47](#_Toc184004600)

[Hình 3.8. Sản phẩm hoàn chỉnh 48](#_Toc184004601)

[Hình 3.9. Lưu đồ thuật toán hệ thống. 49](#_Toc184004602)

[Hình 3.10. Giao diện nhập thông tin đăng ký tài khoản Thingspeak 50](#_Toc184004603)

[Hình 3.11. Hoàn tất đăng ký tài khoản Thingspeak 50](#_Toc184004604)

[Hình 3.12. Đăng ký và đăng nhập tài khoản thành công 51](#_Toc184004605)

[Hình 3.13. Giao diện tạo kênh 51](#_Toc184004606)

[Hình 3.14. Thông tin của kênh dữ liệu 51](#_Toc184004607)

[Hình 3.15. Đồ thị theo dõi dữ liệu 52](#_Toc184004608)

[Hình 3.16. Thiết lập chia sẻ dữ liệu của kênh với cộng đồng 52](#_Toc184004609)

[Hình 3.17. Thông tin API Keys 52](#_Toc184004610)

[Hình 3.18. Cấu hình wifi cho thiết bị 53](#_Toc184004611)

[Hình 3.19. Khai báo mã URL và các trường trong mã nguồn chính 53](#_Toc184004612)

[Hình 3.20. Kiểm tra kết nối và truyền dữ liệu giữa thiết bị và ThingSpeak 53](#_Toc184004613)

[Hình 3.21. Hiển thị dữ liệu trên ThingSpeak 54](#_Toc184004614)

[Hình 3.22. Thiết bị khi thông số ổn định 54](#_Toc184004615)

[Hình 3.23. Thử nghiệm chức năng cảnh báo và gọi điện thoại cảnh báo 55](#_Toc184004616)

[Hình 3.24. So sánh kết quả đo nhịp tim của thiết bị với đồng hồ thông minh 55](#_Toc184004617)

[Hình 3.25. So sánh kết quả đo SpO2 của thiết bị với đồng hồ thông minh 55](#_Toc184004618)

[Hình 3.26. So sánh kết quả đo thân nhiệtcủa thiết bị với nhiệt kế 56](#_Toc184004619)

[Hình 3.27. Đồ thị nhịp tim 57](#_Toc184004620)

[Hình 3.28. Đồ thị nồng độ oxy trong máu SpO2 57](#_Toc184004621)

[Hình 3.29. Đồ thị nhiệt độ cơ thể 57](#_Toc184004622)

[Hình 3.30. Cập nhật thông số lên ThingSpeak theo thời gian thực 58](#_Toc184004623)

[Hình 3.31. Xuất file dữ liệu 58](#_Toc184004624)

# MỞ ĐẦU

**Trong những năm gần đây, với sự phát triển mạnh mẽ của cách mạng công nghiệp 4.0, Internet vạn vật, cùng với** cuộc sống ngày càng được nâng cao, **việc ứng dụng khoa học kỹ thuật và công nghệ đã trở thành xu hướng tất yếu, đang diễn ra một cách nhanh chóng và ngày càng trở nên quan trọng đối với các quốc gia trên thế giới nói chung và Việt Nam nói riêng. Có thể nhận thấy rằng các sản phẩm của khoa học công nghệ ngày càng được hoàn thiện về chức năng và phong phú về chủng loại, đáp ứng được những tiêu chuẩn khắt khe của nhiều ngành nghề, nhiều lĩnh vực trong sản xuất và đời sống. Công nghệ hiện đại không chỉ được ứng dụng trong các ngành sản xuất để phát triển kinh tế, nâng cao năng suất, giải phóng sức lao động mà còn được ứng dụng để phục vụ những nhu cầu thiết thực trong cuộc sống hàng ngày của con người như nhu cầu tự theo dõi sức khỏe một cách chủ động và thường xuyên tại nhà. Điều này là hết sức cần thiết, đặc biệt là trong bối cảnh cuộc sống ngày càng nhiều áp lực và hối hả khiến nhiều người không thể thu xếp được thời gian cho việc thăm khám sức khỏe định kỳ, môi trường sống ô nhiễm nghiêm trọng và ngày càng phát sinh nhiều dịch bệnh nguy hiểm, cùng với đó là tình trạng các hệ thống y tế đang phải đối mặt với những thách thức về yêu cầu ngày càng tăng của người dân đối với chất lượng phải được nâng cao cũng như chi phí của dịch vụ y tế hiện nay.**

**Xuất phát từ hình hình thực tế trên, đồng thời với mong muốn vận dụng những kiến thức đã được trang bị trong quá trình học tập tại Học viện để thiết kế và chế tạo thiết bị thật để phục vụ cuộc sống thực tế, em lựa chọn thực hiện đề tài “Thiết kế và chế tạo thiết bị giám sát thông số sức khỏe tại nhà” để làm đồ án tốt nghiệp.**

**Mục tiêu của đồ án là thiết kế và chế tạo một thiết bị thu thập và hiển thị một số thông số sức khỏe của con người, đồng thời kết hợp với công nghệ IoT, sử dụng nền tảng sẵn có là ThingSpeak để lưu trữ, hiển thị dữ liệu theo thời gian thực nhằm tạo điều kiện thuận lợi cho người sử dụng có thể tra cứu thông tin sức khỏe của bản thân và người thân khi cần thiết. Nội dung của đồ án trình bày cơ sở lý thuyết, giải pháp và công nghệ thiết kế, cũng như quy trình thiết kế, chế tạo thiết bị và được cấu trúc thành 3 chương chính:**

* Chương 1. Tổng quan về đề tài.
* Chương 2. Công nghệ thiết kế.
* Chương 3. Thiết kế và thi công.

Sau thời gian khoảng hơn ba tháng thực hiện đồ án, do vẫn còn hạn chế về kiến thức và kinh nghiệm thực tiễn, thời gian thực hiện đồ án tương đối ngắn nên chắc chắn đồ án không tránh khỏi thiếu sót. Em rất mong nhận được sự góp ý của thầy cô để nội dung của đồ án hoàn thiện hơn.

# CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

## 1.1. Tính thực tiễn của đề tài

Đối với mỗi người, sức khỏe là một tài sản vô giá, không có sức khỏe chúng ta không thể làm được bất kì thứ gì. Trong xã hội hiện đại, sức khỏe đóng vai trò quan trọng không thể phủ nhận. Nó là nền tảng đối với hạnh phúc cá nhân, sự phát triển của mỗi người, gia đình và xã hội. Cuộc sống được coi là trọn vẹn chỉ khi chúng ta có một một tâm trí an lành bên trong một **cơ thể khỏe mạnh.** Thế nhưng, nhịp sống bận rộn, hối hả, quá nhiều lo toan và đầy áp lực của xã hội hiện đại đã khiến cho nhiều người vô tình có thái độ thờ ơ với sức khỏe của chính bản thân mình, việc khám sức khỏe định kỳ tại các cơ sở y tế bị xem như một việc làm xa xỉ, lãng phí, tốn thời gian, dẫn đến tình trạng bỏ qua cơ hội “vàng” để phát hiện sớm và điều trị thành công các bệnh lý nguy hiểm như bệnh lao, tim mạch, tiểu đường, các rối loạn chức năng hô hấp, cao huyết áp, ... Chính vì vậy, việc tự chăm sóc giữ gìn sức khỏe cần bắt đầu càng sớm càng tốt để phòng ngừa nguy cơ ốm bệnh, nâng cao chất lượng cuộc sống. Nếu như trước đây, người Việt quan tâm đến việc chữa bệnh hơn phòng bệnh, không có thói quen chủ động chăm sóc và theo dõi sức khỏe định kỳ thì trong xã hội hiện đại ngày nay, khi điều kiện kinh tế khá hơn, việc chăm sóc sức khỏe chủ động đã được chú trọng hơn rất nhiều, một bộ phận lớn người dân đã bắt đầu quen với xu hướng chăm sóc sức khỏe chủ động đã phổ biến trên thế giới từ rất lâu. Chăm sóc sức khỏe chủ động hướng đến việc tự chăm sóc bản thân, tăng cường sức khỏe để giảm nguy cơ mắc bệnh và theo dõi sức khỏe sát sao để sớm phát hiện các nguy cơ bệnh lý thay vì tập trung vào điều trị khi đã mắc bệnh. Điều này không chỉ giúp nâng cao sức khỏe, cải thiện chất lượng cuộc sống của mỗi người mà còn có tác dụng giảm gánh nặng y tế cho xã hội và không trở thành gánh nặng cho quỹ Bảo hiểm y tế Nhà nước.

Hiện nay, cùng với sự phát triển của xã hội, cuộc sống ngày càng được nâng cao thì việc áp dụng công nghệ khoa học kỹ thuật vào đời sống, công việc ngày càng trở nên cần thiết. Với sự phát triển của các ngành khoa học kỹ thuật, công nghệ kỹ thuật điện tử mà trong đó đặc biệt là kỹ thuật tự động đóng một vai trò quan trọng trong mọi lĩnh vực khoa học kỹ thuật, quản lý, công nghiêp, nông nghiệp, đời sống, quản lý thông tin, ... ngày càng được ứng dụng rộng rãi, trong đó có cả lĩnh vực y tế và chăm sóc sức khỏe. Thực tế cho thấy, các thiết bị y tế ứng dụng công nghệ ngày càng đa dạng, phù hợp với nhiều môi trường khác nhau, từ quy mô lớn như bệnh viện, các cơ sở y tế cho đến các gia đình hoặc cá nhân. Hơn thế nữa, trong hoàn cảnh môi trường sống ngày càng ô nhiễm, tiềm ẩn nhiều dịch bệnh nguy hiểm, cộng thêm những áp lực từ cuộc sống thì việc trang bị các thiết bị y tế, [**t**hiết bị chăm sóc sức khỏe](https://www.dienmayxanh.com/thiet-bi-cham-soc-suc-khoe) tại nhà là điều vô cùng cần thiết, giúp xử lý kịp thời các tình huống khẩn cấp. Việc chủ động theo dõi sức khỏe tại nhà cũng giúp phát hiện sớm các bất thường trong cơ thể và theo dõi tình trạng bệnh lý để có biện pháp điều trị kịp thời.

Với những lý do trên, đồ án lựa chọn thiết kế và chế tạo một thiết bị giám sát thông số sức khỏe tại nhà với thao tác sử dụng đơn giản, kích thước gọn nhẹ. Ngoài ra, thiết bị được ứng dụng công nghệ IoT, kết nối với nền tảng ThingSpeak để phục vụ cho việc lưu lại thông số đo và thuận tiện trong việc chia sẻ dữ liệu với người thân hoặc cơ sở y tế khi cần thiết.

## 1.2. Tổng quan về chỉ số sức khỏe con người

Chỉ số sức khỏe là các thông số thể hiện tình trạng sức khỏe của một người ở thời điểm hiện tại. Dưới đây là một số chỉ số sức khỏe quan trọng nên quan tâm.

### 1.2.1. Nhịp tim và chỉ số nhịp tim

Nhịp tim là một trong các chỉ số quan trọng giúp bác sĩ chẩn đoán được các bất thường trong cơ thể của một người. Chính vì vậy mà nhịp tim tăng cao hay giảm xuống đột ngột có thể là dấu hiệu nhận biết các bệnh lý nguy hiểm màmoojt người đang mắc phải.

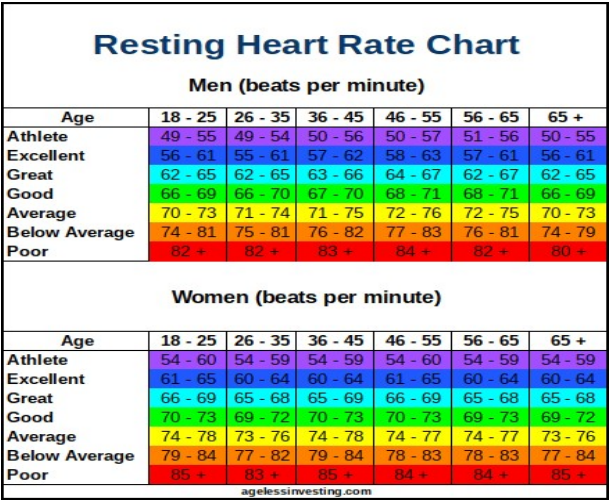
Nhịp tim là tốc độ nhịp tim đo bằng số lần co thắt (nhịp đập) của tim trong mỗi phút (BPM - beat per minute). Trung bình một đời người, tim sẽ đập khoảng 3 tỉ lần. Mỗi chu kỳ tim thực hiện co bóp một lần trong tất cả các buồng tim, khoảng thời gian giữa 2 lần co bóp của 1 buồng tim được coi là 1 chu kỳ, thường nhịp co bóp của tâm thất trái được xác định để tính chu kỳ nghe nhịp tim. Trong quá trình đó, tim sẽ đưa máu có chứa oxy đi tới các cơ quan trong cơ thể và vận chuyển máu thiếu oxy trở lại phổi.  Nhịp tim có thể thay đổi theo nhu cầu thể chất của cơ thể, bao gồm cả nhu cầu hấp thu oxy và bài tiết carbon dioxit. Các hoạt động có thể tạo ra thay đổi nhịp tim bao gồm tập thể dục, ngủ, lo lắng, căng thẳng, bệnh tật và khi uống thuốc. [Nhịp tim](https://medlatec.vn/tin-tuc/tu-van-nhu-the-nao-la-nhip-tim-binh-thuong-s175-n18696) của người bình thường cũng sẽ thay đổi theo tuổi tác. Nói chung, bất kỳ thay đổi lớn nào trong trạng thái thể chất hoặc tinh thần của một người thường dẫn đến thay đổi mạch. Ngoài ra, nhịp tim còn chịu tác động từ 4 yếu tố:

* Nhiệt độ không khí: Nhiệt độ tăng cao thường sẽ khiến tim bơm máu mạnh hơn và nhịp tim tăng 5-10 nhịp/phút.
* Tư thế khi đo: Yếu tố này chỉ ảnh hưởng đáng kể trong 15-20 giây đầu tiên.
* Cân nặng: Những người bị bệnh béo phì, thừa cân thường có nhịp tim cao hơn mức thông thường.
* Sử dụng thuốc: Các loại thuốc có chứa chất ức chế beta sẽ khiến nhịp tim chậm lại, trong khi thuốc chữa trị tuyến giáp lại khiến nhịp tim tăng lên.

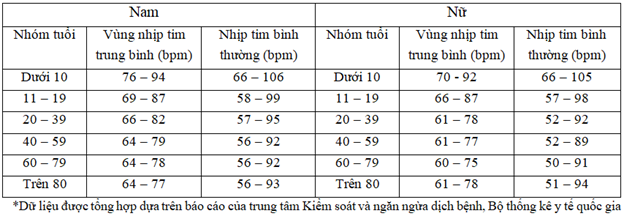
Chỉ số nhịp tim có vai trò quan trọng trong việc nhận biết các dấu hiệu bất thường và chẩn đoán một số vấn đề sức khỏe. Chỉ số nhịp tim bình thường là khác nhau giữa các cá thể, phụ thuộc vào giới tính, độ tuổi, tình trạng sức khỏe, …. Sự thay đổi của chỉ số nhịp tim có thể là dấu hiệu cho thấy một sự thay đổi của trạng thái tim, qua đó có thể phản ánh tình trạng sức khỏe cơ thể. Chỉ số nhịp tim nhắm đến khi cơ thể ở trạng thái hoạt động như làm việc, tập thể dục, vận động cơ thể,… Khi đó, tim cần hoạt động co bóp mạnh để cung cấp đủ lượng máu nuôi các cơ quan. Chỉ số nhịp tim thường được sử dụng để theo dõi trong quá trình luyện tập, bảo đảm các bài tập không quá sức. Vì vậy, theo dõi chỉ số nhịp tim của một người trong thời gian dài sẽ cho ta biết được phần nào tình trạng sức khỏe, thói quen hàng ngày, chất lượng cuộc sống của họ, qua đó có những biện pháp cải thiện kịp thời. Bảng 1.1 và 1.2 thể hiện chỉ số nhịp tim theo từng độ tuổi của nam và nữ.

Theo Hội Tim mạch Hoa Kỳ (America Heart Association - AHA), với hầu hết người trưởng thành, nhịp tim khi nghỉ ngơi bình thường là 60 - 100 nhịp mỗi phút. [Nhịp tim](https://tamanhhospital.vn/nhip-tim-binh-thuong/) dưới 60 lần/phút hoặc trên 100 lần/phút được xem là bình thường trong một số trường hợp như khi ngủ (nhịp tim có thể giảm xuống dưới 60 lần/phút), khi chạy bộ quá sức, vận động mạnh hoặc căng thẳng (nhịp tim đập nhanh trên 100 lần/phút). Còn nhịp tim cao hoặc thấp bất thường được coi là đáng lo ngại khi xảy ra trong nhiều giờ hoặc nhiều ngày và đi kèm với các triệu chứng khác như nhịp tim quá nhanh, kéo dài liên tục trên 100 nhịp mỗi phút, kèm các triệu chứng như chóng mặt, khó thở, đau ngực hoặc quá chậm (dưới 40 lần/phút).

Nhịp tim của người trưởng thành



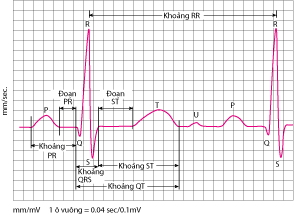
Chỉ số nhịp tim theo độ tuổi của nam và nữ



Nhịp tim có thể được đo chính xác bằng nhiều kỹ thuật. Sau đây là một số phương pháp đo phổ biến:

* Đo bằng tín hiệu điện:

Điện tâm đồ (ECG) đo nhịp tim bằng cách sử dụng các miếng dán điện cực nhỏ dán lên da ngực, cánh tay và chân. Bởi vì các tín hiệu điện tâm đồ là tuần hoàn, nhịp tim có thể được suy ra từ khoảng sóng R đến sóng R (RR) của các tín hiệu tuần hoàn này.

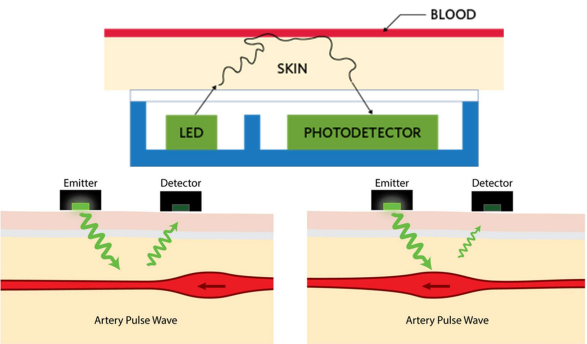


Hình 1.1. Điện tâm đồ

* Đo nhịp tim bằng phương pháp hấp thụ quang học:

Khi tim đập, máu sẽ được đẩy đi khắp cơ thể qua động mạch, tạo ra sự thay đổi về áp suất trên thành động mạch và lượng máu chảy qua động mạch. Vì vậy, ta có thể đo nhịp tim bằng cách đo những sự thay đổi đó. Khi hàm lượng máu trong thành động mạch thay đổi sẽ làm thay đổi mức độ hấp thụ ánh sáng của động mạch, do đó khi một tia sáng được truyền qua động mạch thì cường độ ánh sáng sau khi truyền qua sẽ biến thiên đồng bộ với nhịp tim. Khi tim giãn ra, lượng máu qua động mạch nhỏ nên hấp thụ ít ánh sáng, dẫn đến ánh sáng sau khi truyền qua động mạch sẽ có cường độ lớn; ngược lại khi tim co vào, lượng máu qua động mạch lớn hơn, ánh sáng sau khi truyền qua động mạch sẽ có cường độ nhỏ hơn.

Về nguyên tắc có thể đặt nguồn sáng và photodiode ở bất cứ nơi nào trên cơ thể có chứa động mạch. Vị trí đặt cảm biến hợp lý nhất là các đầu ngón tay, tuy động mạch ở vị trí này không quá lớn nhưng bề dày cơ thể ánh sáng phải truyền qua lại tương đối ít nên chỉ cần dùng 1 LED làm nguồn phát. Mặt khác, ở vị trí này cho mức độ biến thiên cường độ ánh sáng nhận được là khá lớn so với toàn bộ ánh sáng nhận được, tỉ số giữa biên độ tín hiệu với nền một chiều là đủ lớn để phần xử lý tín hiệu hoạt động đưa ra kết quả chính xác nhất.



* + - * 1. Đo nhịp tim bằng phương pháp hấp thụ quang học

### 1.2.2. Nồng độ oxy trong máu.

Lượng oxy trong máu hay độ bão hòa oxy trong máu (SpO2) là một thông số cực kỳ quan trọng, là thước đo lượng oxy mà tế bào máu của một người đang vận chuyển. Duy trì sự cân bằng máu giàu oxy là điều cần thiết cho sức khỏe.

Một phân tử Hemoglobin (Hb) có thể kết hợp với 4 phân tử oxy, khi đã gắn đủ 4 phân tử oxy được gọi là bão hòa oxy. Độ bão hòa oxy trong máu còn được gọi là chỉ số SpO2, biểu thị cho tỷ lệ Hemoglobin có oxy trên tổng lượng Hemoglobin trong máu. Nếu tất cả các phân tử Hemoglobin trong máu đều gắn với oxy thì độ bão hòa oxy là 100%. Chỉ số này được xem là một trong những dấu hiệu sinh tồn của cơ thể, bên cạnh các dấu hiệu như nhiệt độ, mạch, nhịp thở và huyết áp. Khi bị thiếu oxy máu, các cơ quan như tim, gan, não... sẽ chịu tác động tiêu cực rất nhanh. Vì vậy, cần theo dõi chỉ số nồng độ oxy trong máu thường xuyên để kịp thời can thiệp nếu xảy ra tình trạng nguy hiểm.

Hầu hết các phân tử Hb sẽ gắn với oxy khi chúng đi qua phổi. Thang đo chỉ số SpO2 tiêu chuẩn như sau:

* 97% - 99%: oxy trong máu tốt.
* 94% - 96%: oxy trong máu trung bình, cần cho thở thêm oxy.
* 90% - 93%: oxy trong máu thấp, nên có y tá hoặc bác sĩ theo dõi hoặc đến bệnh viện gần nhất.
* Dưới 92% không thở oxy hoặc dưới 95% có thở oxy: Đây là các dấu hiệu suy hô hấp rất nặng.
* Dưới 90% là biểu hiện của một ca cấp cứu trên lâm sàng.

Một số nguyên nhân dẫn đến sự bất thường của nồng độ oxy trong máu:

* Ngộ độc CO: CO thay thế oxy ở vị trí gắn vào sắt trên phân tử Hb gây ra ngộ độc CO, làm tăng COHb (CO gắn vào Hemoglobin) và giảm HbO2 (oxy gắn vào Hemoglobin). Hiện tượng này làm giảm độ bão hòa của oxy trong máu.
* Thiếu máu: Hemoglobin trong máu giảm thấp hơn bình thường. Khi không có tình trạng thiếu oxy máu, máy đo oxy dựa vào mạch đập sẽ cho kết quả chỉ số SpO2 chính xác khi nồng độ hemoglobin giảm xuống 2 - 3g/dL.
* Hoạt động trong môi trường bí khí, thiếu oxy như nhà máy, lò đốt, mỏ quặng...

### 1.2.3. Chỉ số nhiệt độ cơ thể

Nhiệt độ cơ thể là một chỉ số quan trọng trong y tế, giúp bác sĩ xác định tình trạng sức khỏe của một người. Chỉ số nhiệt độ (hay nhiệt độ cơ thể) của con người là một yếu tố quan trọng để đánh giá sức khỏe và sự hoạt động bình thường của cơ thể. Thân nhiệt hay nhiệt độ cơ thể là kết quả của sự cân bằng giữa sản sinh nhiệt và mất nhiệt trong cơ thể, là phép đo quan trọng để tránh nhiều yếu tố bị mất chức năng do nhiệt độ cao. Thông số này cũng phản ánh khả năng điều chỉnh nhiệt độ nội tại của cơ thể và giúp duy trì các chức năng sinh lý tối ưu.

Thân nhiệt chia thành hai loại: nhiệt độ trung tâm và nhiệt độ da của cơ thể. Nhiệt độ da thay đổi trong một phạm vi nhiệt độ rộng hơn nhiệt độ trung tâm, vì cơ chế điều nhiệt của cơ thể điều chỉnh nhiệt độ trung tâm. Nhiệt độ da ảnh hưởng bởi lưu thông máu và cũng liên quan đến nhịp độ và tốc độ trao đổi chất. Các yếu tố bên ngoài như lưu thông không khí, nhiệt độ môi trường và độ ẩm cũng đóng một vai trò quan trọng trong cơ chế điều hòa thân nhiệt này.

Nhiệt độ cơ thể người bình thường thường dao động trong một phạm vi hẹp và có vai trò quan trọng trong việc duy trì sự cân bằng nội môi. **Nhiệt độ cơ thể trung bình** của người trưởng thành dao động từ **36.10C đến 37.20C**. Tuy nhiên, mức nhiệt độ này có thể thay đổi tùy theo thời gian trong ngày, độ tuổi, mức độ hoạt động và tình trạng sức khỏe của từng cá nhân. Một số yếu tố khác có thể ảnh hưởng đến thân nhiệt:

* **Mức độ hoạt động**: Khi cơ thể vận động hoặc tham gia vào các hoạt động thể chất, nhiệt độ cơ thể có thể tăng lên.
* **Tuổi tác**: Người già và trẻ em có thể có sự thay đổi lớn hơn về nhiệt độ cơ thể so với người trưởng thành khỏe mạnh.
* **Thời gian trong ngày**: Nhiệt độ cơ thể thường dao động trong suốt 24 giờ.
* **Chế độ ăn uống và giấc ngủ**: Chế độ dinh dưỡng, lượng calo tiêu thụ, và các yếu tố khác cũng có thể tác động đến nhiệt độ cơ thể.
* **Điều kiện môi trường**: Nhiệt độ môi trường xung quanh có thể làm thay đổi nhiệt độ cơ thể thông qua quá trình tỏa nhiệt, hấp thu nhiệt hoặc qua việc đổ mồ hôi.

### 1.2.4. Huyết áp

Huyết áp được coi là thông số tim phổi quan trọng nhất, cho biết áp lực do máu tác động lên thành động mạch. Huyết áp cung cấp thông tin gián tiếp về lưu lượng máu khi tim co bóp (tâm thu) và thư giãn (tâm trương) và cũng có thể cho biết sự phân phối oxy qua tế bào. Huyết áp bị ảnh hưởng bởi một số đặc điểm sinh lý của con người như cung lượng tim; sức cản mạch ngoại vi; lượng máu và độ nhớt; và tính đàn hồi của thành mạch. Theo dõi huyết áp lưu động cho phép đo huyết áp nhiều lần trong ngày, lý tưởng để theo dõi huyết áp cao (tăng huyết áp), một trong những mối đe dọa lớn nhất đối với gánh nặng bệnh tật toàn cầu, cải thiện dự đoán bệnh tim mạch.

[**Huyết áp**](https://tamanhhospital.vn/huyet-ap/) không phải lúc nào cũng đứng yên mà sẽ thay đổi theo hoạt động của con người. **Huyết áp bình thường** là khi chỉ số huyết áp tâm trương và huyết áp tâm thu đo được nằm trong phạm vi khỏe mạnh, trong đó:

* [Huyết áp tâm thu](https://tamanhhospital.vn/huyet-ap-tam-thu/)**:** Trong khoảng từ 90-139 mmHg, đây là mức huyết áp cao nhất trong mạch máu, thể hiện áp lực của dòng máu lên lòng mạch ở thì tâm thu, phản ánh khả năng bơm máu của tim đi tới các hệ cơ quan trong cơ thể.
* [Huyết áp tâm trương](https://tamanhhospital.vn/huyet-ap-tam-truong/)**:** Trong khoảng từ 60-89 mmHg, là áp lực máu thấp nhất trong mạch máu xảy ra giữa các nhịp tim. Huyết áp tâm trương đo lường áp lực máu lên thành động mạch khi tim thả lỏng sau khi co bóp.

Thông số huyết áp bình thường theo độ tuổi được trình bày trong bảng 1.3.

Bảng thông số huyết áp theo độ tuổi và giới tính

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tuổi** | **Nam** | | **Nữ** | |
| ***Huyết áp tâm thu (mmHg)*** | ***Huyết áp tâm trương (mm Hg)*** | ***Huyết áp tâm thu (mmHg)*** | ***Huyết áp tâm trương (mm Hg)*** |
| 21-25 | 120.5 | 78.5 | 115.5 | 70.5 |
| 26-30 | 119.5 | 76.5 | 113.5 | 71.5 |
| 31-35 | 114.5 | 75.5 | 110.5 | 72.5 |
| 36-40 | 120.5 | 75.5 | 112.5 | 74.5 |
| 41-45 | 115.5 | 78.5 | 116.5 | 73.5 |
| 46-50 | 119.5 | 80.5 | 124 | 78.5 |
| 51-55 | 125.5 | 80.5 | 112.55 | 74.5 |
| 56-60 | 129.5 | 79.5 | 132.5 | 78.5 |
| 61-65 | 115.5 | 76.5 | 130.5 | 77.5 |
|  |  |  |  |  |

## 1.3. Thiết bị y tế theo dõi sức khỏe

### 1.3.1.Giới thiệu

Thiết bị theo dõi sức khỏe là các công cụ công nghệ được thiết kế để đo, giám sát và phân tích các chỉ số sức khỏe của người sử dụng, từ đó giúp họ duy trì lối sống lành mạnh và cải thiện sức khỏe. Những thiết bị này thường có khả năng theo dõi các yếu tố như hoạt động thể chất, nhịp tim, giấc ngủ, mức độ căng thẳng, huyết áp, và nhiều thông số sức khỏe khác. Các thiết bị theo dõi sức khỏe có thể giúp người dùng phát hiện sớm các vấn đề sức khỏe, từ đó đưa ra biện pháp cải thiện phù hợp.

Thiết bị theo dõi sức khỏe chia làm 2 loại là thông minh và thông thường. Thiết bị theo dõi sức khỏe thông thường thường có cấu tạo đơn giản, bao gồm các thành phần cơ bản phục vụ cho việc đo lường các chỉ số sức khỏe cơ bản. Đầu tiên, thiết bị sử dụng các cảm biến để thu thập dữ liệu, ví dụ như cảm biến đo nhịp tim, huyết áp, hoặc nhiệt độ cơ thể. Màn hình hiển thị giúp người dùng theo dõi kết quả đo lường, có thể là màn hình analog hoặc kỹ thuật số tùy thuộc vào loại thiết bị. Các thiết bị này có thể hoạt động nhờ nguồn năng lượng từ pin hoặc điện trực tiếp, giúp đảm bảo tính di động và tiện lợi. Một số thiết bị còn có cơ chế điều khiển đơn giản như nút bấm hoặc cần gạt để người dùng thao tác. Mặc dù không có nhiều tính năng phức tạp như các thiết bị thông minh, các thiết bị theo dõi sức khỏe thông thường vẫn có thể giúp người dùng theo dõi và quản lý các chỉ số sức khỏe cơ bản một cách hiệu quả và dễ dàng.



* + - * 1. Thiết bị theo dõi sức khỏe thông minh

Thiết bị theo dõi sức khỏe thông minh có cấu tạo phức tạp, tích hợp nhiều công nghệ tiên tiến để theo dõi và phân tích các chỉ số sức khỏe của người dùng. Đầu tiên, thiết bị sử dụng các cảm biến như cảm biến nhịp tim, gia tốc kế, con quay hồi chuyển, cảm biến oxy trong máu, huyết áp, nhiệt độ cơ thể, và điện tâm đồ để thu thập dữ liệu về các chỉ số sinh lý. Dữ liệu từ các cảm biến này sẽ được bộ vi xử lý trong thiết bị xử lý và chuyển đổi thành các thông tin hữu ích cho người dùng. Thiết bị thường được trang bị màn hình hiển thị, có thể là OLED hoặc LCD, để người dùng theo dõi kết quả đo lường. Ngoài ra, thiết bị còn có bộ nhớ để lưu trữ dữ liệu và kết nối không dây như Bluetooth hoặc wifi để đồng bộ hóa với các ứng dụng trên điện thoại. Nguồn năng lượng của thiết bị thường là pin sạc lithium-ion, cung cấp năng lượng cho các tính năng của thiết bị. Vỏ ngoài của thiết bị được làm từ vật liệu bền chắc như nhựa hoặc kim loại, và dây đeo thường bằng silicone hoặc vải để tạo sự thoải mái khi sử dụng. Ngoài các tính năng cơ bản, nhiều thiết bị còn tích hợp GPS để theo dõi hoạt động ngoài trời, tính năng nhận dạng giấc ngủ để theo dõi chất lượng giấc ngủ, và cảnh báo sức khỏe để thông báo người dùng về các vấn đề sức khỏe tiềm ẩn. Tất cả các thành phần này phối hợp chặt chẽ, giúp thiết bị cung cấp thông tin sức khỏe chính xác và hỗ trợ người dùng cải thiện sức khỏe một cách hiệu quả.

### 1.3.2. Một số loại thiết bị theo dõi sức khỏe phổ biến

#### 1.3.2.1. Đồng hồ thông minh - Smartwatches



* + - * 1. Đồng hồ thông minh.

Các đồng hồ thông minh như Apple Watch, Samsung Galaxy Watch hay Fitbit cung cấp nhiều tính năng theo dõi sức khỏe, bao gồm:

* Theo dõi nhịp tim: Đo lường nhịp tim liên tục hoặc trong các hoạt động thể thao.
* Theo dõi giấc ngủ: Phân tích chất lượng giấc ngủ, giúp người dùng cải thiện thói quen ngủ.
* Theo dõi hoạt động thể chất: Đo bước đi, quãng đường di chuyển, lượng calo tiêu thụ.
* Tính năng ECG (Electrocardiogram): Đo điện tâm đồ để phát hiện các vấn đề về tim mạch.

#### 1.3.2.2. Vòng đeo tay thể thao - Fitness Bands

Các vòng đeo tay như Fitbit Charge, Xiaomi Mi Band thường có thiết kế nhẹ nhàng, đơn giản hơn đồng hồ thông minh và tập trung vào các tính năng cơ bản:

* Đếm bước đi và đo quãng đường: Theo dõi mức độ vận động hàng ngày.
* Theo dõi giấc ngủ: Cung cấp báo cáo về các giai đoạn giấc ngủ.
* Đo nhịp tim: Giám sát nhịp tim trong suốt quá trình vận động.

#### 1.3.2.3. Máy đo huyết áp tự động

Thiết bị này cho phép người dùng đo huyết áp tại nhà một cách đơn giản. Các máy đo huyết áp tự động có thể kết nối với điện thoại thông minh để lưu trữ và phân tích dữ liệu theo thời gian.

#### 1.3.2.4. Máy đo nồng độ oxy trong máu - Pulse Oximeters

Thiết bị này đo mức độ oxy trong máu giúp phát hiện các vấn đề về hô hấp hoặc tim mạch. Thiết bị này rất quan trọng trong việc theo dõi sức khỏe của những người có bệnh lý liên quan đến tim và phổi.



* + - * 1. Máy đo huyết áp tự động và máy đo nồng độ oxy trong máu

#### 1.3.2.5. Thiết bị theo dõi giấc ngủ



* + - * 1. Oura Ring

Các thiết bị như Oura Ring hay tính năng giám sát giấc ngủ trên đồng hồ thông minh có thể phân tích các giai đoạn giấc ngủ. Thiết bị này cũng thu thập và phân tích dữ liệu sức khỏe chính xác và toàn diện, bao gồm nhịp tim, mức độ căng thẳng và nhiệt độ cơ thể, cung cấp những thông tin và gợi ý hữu ích để người dùng cải thiện sức khỏe.

### 1.3.3. Ưu điểm và nhược điểm

Ưu điểm:

* Quản lý sức khỏe chủ động: Giúp người dùng theo dõi và điều chỉnh các yếu tố như hoạt động thể chất, giấc ngủ, và tình trạng tim mạch.
* Dự đoán và phát hiện bệnh sớm: Các chỉ số như nhịp tim, huyết áp hay mức độ oxy trong máu có thể giúp phát hiện sớm các dấu hiệu bất thường, từ đó người dùng có thể can thiệp kịp thời.
* Cải thiện thói quen sống: Các báo cáo chi tiết về mức độ vận động và giấc ngủ giúp người dùng điều chỉnh lối sống, tránh các thói quen có hại cho sức khỏe.
* Tăng cường động lực luyện tập: Các tính năng như mục tiêu hàng ngày hoặc thử thách giúp người dùng duy trì động lực trong việc tập luyện thể thao.

Nhược điểm và thách thức:

* Độ chính xác: Mặc dù các thiết bị này ngày càng chính xác, nhưng không phải lúc nào kết quả đo lường cũng hoàn toàn giống với các thiết bị y tế chuyên dụng.
* Tiêu thụ năng lượng: Các thiết bị như đồng hồ thông minh thường cần phải sạc thường xuyên, đôi khi gây bất tiện.
* Tính khả dụng và đồng bộ hóa: Không phải tất cả các thiết bị đều tương thích với mọi hệ điều hành, và có thể gặp khó khăn trong việc đồng bộ hóa dữ liệu với các ứng dụng di động.

### 1.3.4. Xu hướng phát triển

Thiết bị theo dõi sức khỏe thông minh đang ngày càng phát triển mạnh mẽ nhờ vào sự tiến bộ nhanh chóng của công nghệ và nhu cầu ngày càng cao trong việc chăm sóc sức khỏe cá nhân. Các thiết bị này không chỉ giúp người dùng theo dõi sức khỏe mà còn giúp cải thiện chất lượng cuộc sống và tăng cường hiệu quả chăm sóc sức khỏe chủ động. Dưới đây là một số xu hướng nổi bật trong lĩnh vực thiết bị theo dõi sức khỏe:

* Tích hợp công nghệ trí tuệ nhân tạo: Tập trung phát triển theo hướng:
* Phân tích dữ liệu sức khỏe thông minh: Các thiết bị theo dõi sức khỏe đang ngày càng sử dụng trí tuệ nhân tạo (AI) để phân tích dữ liệu thu thập được từ người dùng và đưa ra các khuyến nghị về cách cải thiện sức khỏe. Ví dụ, AI có thể phân tích các mẫu giấc ngủ, nhận diện các xu hướng trong chỉ số tim mạch, từ đó cung cấp lời khuyên cá nhân hóa về dinh dưỡng, luyện tập và các thói quen sống.
* Dự đoán bệnh lý: AI có khả năng phân tích dữ liệu sức khỏe từ nhiều người dùng để nhận diện sớm các dấu hiệu của các bệnh lý tiềm ẩn như tim mạch, tiểu đường, hoặc bệnh về hô hấp.

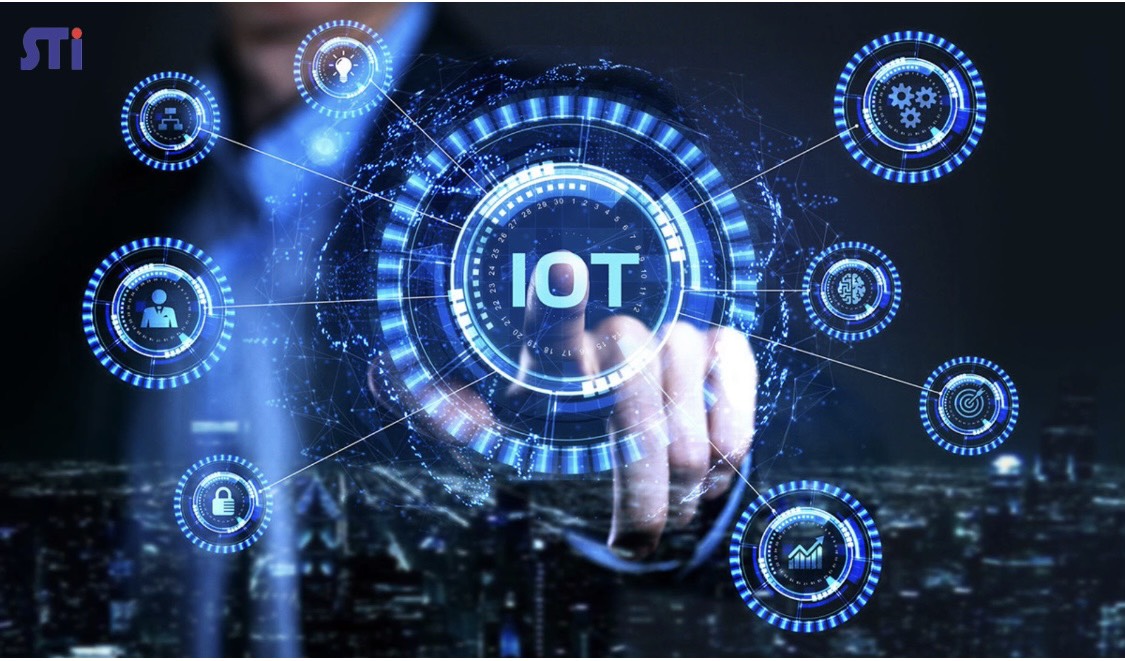


* + - * 1. Đồng hồ thông minh tích hợp trí tuệ nhân tạo
* **Tăng cường tính năng chăm sóc sức khỏe tâm lý:** Những năm gần đây, người ta đã chú trọng hơn đến sức khỏe tâm lý, đặc biệt là trong bối cảnh dịch bệnh và áp lực công việc ngày càng tăng. Các thiết bị theo dõi sức khỏe đang dần tích hợp khả năng đo lường và theo dõi mức độ căng thẳng, lo âu và các chỉ số tâm lý khác.
* **Ứng dụng công nghệ cảm biến sinh học (biometrics):** Các thiết bị như đồng hồ thông minh không chỉ đo nhịp tim mà còn có thể nhận diện các dấu hiệu tâm lý thông qua các thay đổi sinh học (như huyết áp, nhịp thở) và cung cấp các bài tập thư giãn hoặc thiền định để giúp người dùng giảm stress.

## 1.4. Sơ lược về Internet vạn vật - IoT

### 1.4.1. Giới thiệu

Internet vạn vật (Internet of Things - IoT) là mạng lưới vạn vật được kết nối với nhau bằng Internet, được đưa ra bởi Kevin Ashton vào năm 1999 - người sáng lập ra trung tâm Auto - ID tại đại học MIT. Thuật ngữ IoT được Liên minh Viễn thông Quốc tế (ITU) giới thiệu trong báo cáo Internet của ITU năm 2005. Đây là một hệ thống phức tạp, bởi nó bao gồm một lượng lớn các đường liên kết giữa máy móc, thiết bị và dịch vụ với nhau thông qua một mạng duy nhất mà không cần đến sự tương tác trực tiếp giữa người với người, hay người với máy tính.

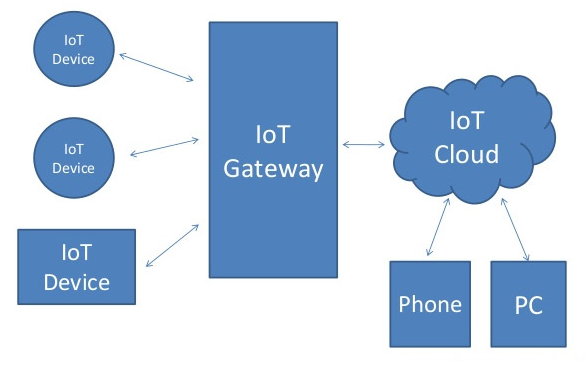


* + - * 1. Hệ thống IoT.

Các thiết bị IoT bao gồm những đồ vật được gắn thêm cảm biến để thu thập dữ liệu về môi trường xung quanh, thu phát tín hiệu, nhận dữ liệu và ra lệnh điều khiển các thiết bị khác được kết nối qua mạng Internet, Wifi, Bluetooth, mạng viễn thông băng rộng (3G, 4G, 5G), hồng ngoại.... Các thiết bị này có thể thực hiện việc trao đổi dữ liệu với nhau được kết nối với ứng dụng để điều khiển nó trực tiếp hoặc gián tiếp. Ngoài ra chúng còn có thể thu thập dữ liệu từ các thiết bị khác và xử lý dữ liệu ngay tại thiết bị hoặc gửi dữ liệu tới các máy chủ tập trung hay các ứng dụng dựa trên đám mây để xử lý những dữ liệu này. Người dùng có thể kiểm soát mọi đồ vật của mình qua mạng chỉ bằng một vài thao tác đơn giản trên thiết bị thông minh như điện thoại thông minh, máy tính bảng, máy tính cá nhân hay thậm chí là một chiếc đồng hồ thông minh nhỏ bé trên tay mà không bị giới hạn không gian hay thời gian.

Hiện nay, IoT không còn là một dự đoán nữa mà là một cuộc cách mạng ứng dụng công nghệ mới đang diễn ra như vũ bão trên toàn thế giới. Với khoảng 14 tỷ thiết bị IoT trên thế giới ở thời điểm hiện tại, ước tính đạt 27 tỷ thiết bị vào năm 2025, thế giới sẽ chứng kiến các thiết bị IoT được triển khai rộng rãi với nhiều công nghệ và nhiều ứng dụng mới trên nhiều mặt của đời sống kinh tế - xã hội. IoT đã trở thành một trong những công nghệ quan trọng, giúp mọi người sống và làm việc thông minh hơn, chủ động hoàn toàn trong cuộc sống. Có thể thấy, tiềm năng ứng dụng của IoT trải rộng trên nhiều lĩnh vực và có xu hướng đột phá trong tương lai, những ứng dụng của IoT ngày càng phong phú và thiết thực đối với cuộc sống trong xã hội hiện đại, mang lại trải nghiệm phong phú và tiện nghi cho người sử dụng.

### 1.4.2. Cấu trúc của hệ thống IoT



* + - * 1. Cấu trúc hệ thống IoT.

Một hệ thống IoT về cơ bản cấu tạo gồm 4 phần: Thiết bị (Things), trạm kết nối (Gateways), hạ tầng mạng và điện toán đám mây (Network and Cloud) và các lớp tạo cung cấp dịch vụ (Services and Solution Layers).

* Thiết bị (Things): Đây là những thiết bị điện tử, máy móc hoặc cảm biến mà bạn muốn kết nối vào mạng IoT. Chúng có thể là điện thoại thông minh, máy tính, máy chủ, cảm biến nhiệt độ, đèn, thiết bị đo lường, xe cộ và nhiều loại thiết bị khác được kết nối trực tiếp thông qua băng tầng mạng không dây và truy cập vào Internet. Giải pháp IoT giúp các thiết bị thông minh được sàng lọc, kết nối và quản lý dữ liệu một cách cục bộ, còn các thiết bị chưa thông minh thì có thể kết nối được thông qua các trạm kết nối.
* Trạm kết nối (Gateways): Một trong những trở ngại chính trong khi triển khai IoT đó là gần 85% các thiết bị phần cứng đã không được thiết kế để có thể kết nối với Internet và không chia sẻ dữ liệu với điện toán đám mây. Các trạm kết nối đóng vai trò là một trung gian trực tiếp, cho phép các vật dụng này kết nối với điện toán đám mây một cách được bảo mật và dễ dàng được quản lý.
* Hạ tầng mạng và điện toán đám mây (Network and Cloud): Internet là một hệ thống toàn cầu của nhiều mạng IP được kết nối với nhau và liên kết với hệ thống máy tính. Cơ sở hạ tầng mạng này bao gồm thiết bị định tuyến, trạm kết nối, thiết bị tổng hợp, thiết bị lặp và nhiều thiết bị khác có thể kiểm soát lưu lượng dữ liệu lưu thông và cũng được kết nối đến mạng lưới viễn thông và cáp được triển khai bởi các nhà cung cấp dịch vụ. Các công nghệ mạng được sử dụng trong IoT gồm wifi, Bluetooth, Zigbee, LTE, LoRaWAN và nhiều công nghệ khác. Các trung tâm dữ liệu và hạ tầng điện toán đám mây cung cấp lộ trình chia sẻ và lưu trữ dữ liệu IoT.
* Các lớp tạo và cung cấp dịch vụ (Services and Solution Layers): Lớp này gồm kiến trúc trung tâm dữ liệu cung cấp khả năng triển khai, quản lý và bảo vệ các ứng dụng, thông tin, cơ sở hạ tầng máy chủ một cách nhanh chóng, an toàn và tin cậy. Ngoài ra nó còn cung cấp các dịch vụ mở rộng cho cơ sở hạ tầng, đảm bảo sự liên kết các nguồn lực của trung tâm dữ liệu với các yêu cầu của ứng dụng một cách nhanh chóng và an toàn.

### 1.4.3. Ứng dụng

Tính linh hoạt của IoT làm cho nó trở thành công cụ đắc lực đối với rất nhiều doanh nghiệp, tổ chức, cơ quan chính phủ, ... Dưới đây là một số ví dụ rõ nét nhất về ứng dụng của Internet vạn vật:

* Ứng dụng IoT trong nông nghiệp thông minh

Đối với việc trồng trọt trong nhà kính, IoT biến việc giám sát, quản lý các điều kiện khí hậu vi mô thành hiện thực, từ đó giúp tăng sản lượng. Đối với việc trồng cây ngoài trời, thiết bị sử dụng công nghệ IoT có thể cảm nhận được độ ẩm, chất dinh dưỡng của đất, kết hợp với dữ liệu thời tiết giúp kiểm soát tốt hơn hệ thống tưới tiêu và bón phân thông minh.

* Ứng dụng IoT trong ngôi nhà thông minh

[Nhà thông minh](https://www.elcom.com.vn/nha-thong-minh-nhung-xu-huong-smart-building-trong-tuong-lai-1663723433) đảm nhận những việc như kích hoạt và kiểm soát môi trường để ngôi nhà luôn ở trạng thái thoải mái nhất cho người ở. Bữa tối được nấu từ xa, thức ăn đã sẵn sàng trước khi chủ nhân quay trở về. An ninh cũng dễ dàng hơn nhờ khả năng điều khiển thiết bị theo dõi, cảnh báo và hệ thống điện từ xa, đồng thời kích hoạt khóa thông minh, cho phép những người thích hợp ra vào nhà ngay cả khi họ không có chìa khóa. Ngoài ra, bằng việc ứng dụng IoT, các thiết bị đeo bao gồm điện thoại, đồng hồ, kính thực tế ảo, máy theo dõi sức khỏe, ... đã giúp người dùng cải thiện khả năng và phương thức giải trí, kết nối mạng, theo dõi sức khỏe và luyện tập thể chất.

* Ứng dụng IoT trong y tế



* + - * 1. IoT ứng dụng trong y tế

Đầu tiên và quan trọng nhất, thiết bị đeo IoT cho phép bệnh viện, chuyên gia theo dõi sức khỏe của bệnh nhân tại nhà. Việc đưa cảm biến IoT vào các thiết bị quan trọng khác cũng giúp giảm tình trạng hỏng hóc và tăng độ tin cậy khi khám chữa bệnh. Điều này góp phần giảm thời gian điều trị nội trú trong bệnh viện nhờ cung cấp thông tin theo thời gian thực, chính xác đến từng phút, đảm bảo bệnh nhân được chăm sóc tốt khi điều trị ngoại trú. Trong bệnh viện, giường thông minh thông báo cho nhân viên y tế về tình trạng sẵn có nhằm cắt giảm thời gian chờ đợi. Quá trình chăm sóc người bệnh, đặc biệt là người cao tuổi nhờ vậy mà trở nên thuận tiện, thoải mái hơn. Ngoài việc theo dõi thời gian thực tại nhà, cảm biến cũng có thể xác định bệnh nhân có gặp phải những vấn đề sức khỏe nghiêm trọng hơn như bị ngã hay bị đau tim hay không.

* Ứng dụng IoT trong sản xuất

Công nghệ RFID và GPS có thể giúp doanh nghiệp theo dõi sản phẩm từ khi bắt đầu sản xuất trên dây chuyền ở nhà máy đến vị trí bày bán tại cửa hàng đích, toàn bộ chuỗi cung ứng từ đầu đến cuối. Những cảm biến này thu thập thông tin về thời gian di chuyển, tình trạng sản phẩm và điều kiện môi trường mà sản phẩm phải chịu. Cảm biến được gắn vào thiết bị trong nhà máy cũng giúp xác định "nút thắt cổ chai" (Bottleneck) trong dây chuyền, do đó giảm thiểu thời gian sản xuất, tránh lãng phí. Một số cảm biến khác cũng có thể theo dõi hiệu suất của máy, dự đoán khi nào thiết bị sẽ cần bảo trì, do đó ngăn ngừa hỏng hóc nặng nề, gây tốn kém nhiều chi phí.

* Ứng dụng IoT trong bán lẻ

Công nghệ IoT có rất nhiều thứ để cung cấp cho thế giới bán lẻ. Số liệu bán hàng mua sắm trực tuyến hoặc tại cửa hàng được kiểm soát tự động. Phần lớn điều này phụ thuộc vào RFID, vốn đã được sử dụng rộng rãi trên toàn thế giới. Thêm vào đó, sự ra đời của mua sắm trực tuyến đã làm giảm nhu cầu đối với cơ sở bán lẻ truyền thống. Tuy nhiên, IoT có thể giúp phân tích lưu lượng truy cập của trung tâm mua sắm, các cửa hàng nằm trong trung tâm thương mại có thể thực hiện những điều chỉnh cần thiết, nhằm nâng cao trải nghiệm mua sắm của khách hàng, đồng thời giảm chi phí. Hơn nữa, IoT giúp nhà bán lẻ tiếp cận mục tiêu khách hàng dựa trên những lần mua hàng trước đây. Trang bị thông tin được cung cấp qua IoT, nhà bán lẻ có thể tạo chương trình khuyến mãi được cá nhân hóa cho khách hàng trung thành của họ, loại bỏ chương trình tiếp thị đại trà tốn kém mà không có nhiều cơ hội thành công. Phần lớn các chương trình khuyến mãi này có thể được thực hiện thông qua điện thoại thông minh của khách hàng.

* Ứng dụng IoT trong giao thông vận tải

Đến thời điểm này, những tiến bộ đạt được với [ô tô tự lái](https://www.elcom.com.vn/xe-tu-lai-tuong-lai-cua-giao-thong-thong-minh-1678862721) đã được nhiều người biết đến. Nhưng đó chỉ là một phần của tiềm năng to lớn trong lĩnh vực giao thông vận tải. GPS, một ví dụ khác của IoT, đang được sử dụng để giúp các công ty vận tải vạch ra tuyến đường nhanh hơn, hiệu quả hơn cho xe tải vận chuyển hàng hóa, nhờ đó đẩy nhanh thời gian giao hàng. Ngoài ra, nhà quy hoạch thành phố cũng có thể sử dụng dữ liệu GPS đó nhằm xác định mô hình giao thông, nhu cầu chỗ đậu xe cũng như xây dựng và bảo trì đường bộ.



* + - * 1. Thiết bị IoT trong nhà thông minh giúp cuộc sống dễ dàng hơn.

Tóm lại, IoT đã trở thành một trong những công nghệ quan trọng. IoT giúp mọi người sống và làm việc thông minh hơn, cũng như chủ động hoàn toàn trong cuộc sống. Do đó, IoT là một trong những công nghệ quan trọng nhất của cuộc sống hàng ngày với rất nhiều ứng dụng trong thế giới thực, trải dài trên nhiều lĩnh vực, hướng tới nhiều đối tượng từ cá nhân cho tới doanh nghiệp, nhà nước và đa quốc gia.

# CHƯƠNG 2. CÔNG NGHỆ THIẾT KẾ



## 2.1. Giải pháp thiết kế

### 2.1.1. Mục tiêu thiết kế

Thiết kế và chế tạo thiết bị thông minh hỗ trợ chăm sóc sức khỏa có các chức năng như sau:

* Thiết kế thiết bị thông minh hỗ trợ theo dõi các thông số sức khỏe cơ bản
* Theo dõi giấc ngủ và đưa ra gợi ý cải thiện chất lượng giấc ngủ
* Cảnh báo sớm khi phát hiện bất thường về sức khỏe và nhắc nhở vận động
* Kết hợp truyền tin không dây để định vị và hỗ trợ người dùng trong tình huống khẩn cấp

### 2.1.2. Sơ đồ khối

A diagram of a computer

AI-generated content may be incorrect.

* + - * 1. Sơ đồ khối

Sơ đồ khối của thiết bị được trình bày trên hình 2.1 với chức năng từng khối cụ thể như sau:

* **Khối xử lý trung tâm**: ESP32-S3-MINI-1-N8 (MCU lõi kép, tích hợp Wi-Fi + BLE).
* **Khối cảm biến:** Đo các chỉ số sức khỏe và chuyển động cơ bản.
* **Khối lưu trữ:** Lưu trữ thông tin khi không gửi thành công.
* **Khối hiển thị:** Hiển thị nhanh các thông số cơ bản tại chỗ.
* **Khối nguồn**: Cung cấp điện ổn định cho toàn bộ hệ thống từ pin hoặc nguồn ngoài.
* **Khối lưu trữ:** Ghi lại dữ liệu tạm thời hoặc khi offline.
* **Khối điều khiển:** Cho phép người dùng tương tác/cấu hình thiết bị tại chỗ.
* **Server:** Nhận MQTT, xử lý và chuyển tiếp vào Database.
* **Database:** Lưu trữ dài hạn dữ liệu sức khỏe của người dùng.
* **Website và Ứng dụng:** Giao diện để hiển thị các thông tin biểu đồ, lịch sử, cấu hình.

## 2.2. Công nghệ phần cứng

### 2.2.1. ESP-WROOM-32

2.2.1.1. Giới thiệu



* + - * 1. Module ESP32-S3-MINI-1-N8

Module ESP32-S3-MINI-1-N8 là phiên bản thu gọn, hiệu năng cao của dòng ESP32-S3 do Espressif Systems sản xuất. Được đóng gói với kích thước chỉ 18 × 25,5 mm, module này kết hợp vi xử lý lõi kép Tensilica® Xtensa® LX7 đạt tốc độ tối đa 240 MHz, Flash SPI 8 MB và SRAM 2 MB. Tích hợp anten PCB cùng khả năng kết nối Wi‑Fi 802.11 b/g/n và Bluetooth Low Energy 5.0 (hỗ trợ advertising extension, long range), ESP32-S3-MINI-1-N8 lý tưởng cho các thiết bị IoT đeo tay, cảm biến di động, gateway BLE-to-MQTT và các ứng dụng nhúng tiêu thụ thấp.

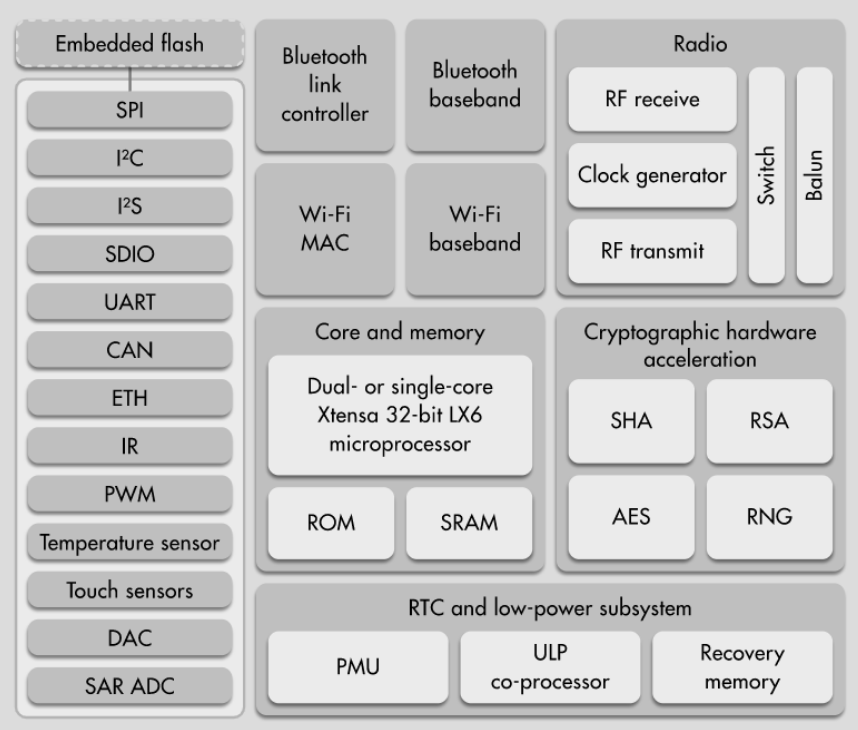
**Ưu điểm nổi bật:**

* **Đa nhiệm mạnh mẽ:** Lõi kép xử lý song song, chạy các tác vụ cảm biến, giao tiếp và xử lý dữ liệu mượt mà.
* **Tiết kiệm năng lượng:** Pin‑friendly với chế độ deep-sleep chỉ ~5 µA, kéo dài thời gian hoạt động của thiết bị.
* **Bộ nhớ dư dả:** Cho phép cài đặt firmware phức hợp, OTA, và lưu trữ tạm thời dữ liệu lớn.
* **Ngoại vi phong phú:** Hỗ trợ đa dạng giao tiếp (I²C, SPI, UART, I²S, RMT, CAN, SDIO) cùng ADC, DAC, touch sensor.
* **Thiết kế SMT dễ dàng:** Pad 0,5 mm, tương thích dây chuyền SMT, giảm chi phí sản xuất.

#### 2.2.1.2. Cấu hình và thông số kĩ thuật

Module ESP32-S3-MINI-1-N8 được thiết kế với hệ thống thông số kỹ thuật toàn diện, đáp ứng nhiều yêu cầu ứng dụng chuyên nghiệp:

* **Vi xử lý & Bộ nhớ:**
  + **CPU:** Dual-core Tensilica® Xtensa® LX7, tối đa 240 MHz cho mỗi lõi, hỗ trợ lệnh vector cho AI acceleration.
  + **Flash:** 8 MB SPI NOR flash, hỗ trợ secure boot và encrypted flash.
  + **SRAM:** 2 MB internal SRAM, đủ khả năng chạy các ứng dụng thời gian thực và giao thức mạng.
* **Kết nối không dây:**
  + **Wi‑Fi:** 802.11 b/g/n trên băng tần 2,4 GHz, hỗ trợ HT40, WPA2.
  + **Bluetooth:** BLE 5.0, quảng bá extended advertising, long range PHY, đảm bảo phủ sóng rộng và tiết kiệm năng lượng.
* **Giao tiếp ngoại vi:**
  + **GPIO:** Tối đa 23 chân, hỗ trợ PWM, input capture và output compare.
  + **ADC:** 5 kênh 12-bit, tốc độ sampling đến 20 kSPS, phù hợp đo tín hiệu analog từ cảm biến.
  + **DAC:** 2 kênh 8-bit, tạo tín hiệu analog điều khiển ngoại vi.
  + **I²C:** 2 bus, tốc độ chuẩn up to 400 kHz.
  + **SPI:** 3 bus, hỗ trợ chế độ full‑duplex, tốc độ up to 80 MHz.
  + **UART:** 2 cổng, hỗ trợ RTS/CTS hardware flow control, baud rate lên đến 5 Mbps.
  + **I²S, RMT, CAN, SDIO:** Hỗ trợ các ứng dụng âm thanh, remote control, công nghiệp và thẻ nhớ SD.
* **Nguồn & Tiêu thụ:**
  + **Điện áp hoạt động:** 3,0 – 3,6 V.
  + **Deep-sleep:** ~5 µA.
  + **Active-mode:** 80–240 mA tùy mức CPU và RF.
* **Cơ khí & Môi trường:**
  + **Kích thước:** 18 × 25,5 mm, pad pitch 0,5 mm.
  + **Nhiệt độ hoạt động:** –40 °C đến 85 °C.
  + **Độ ẩm:** 5 % – 95 % RH không ngưng tụ.



* + - * 1. Cấu hình của module ESP32-S3-MINI-1-N8

#### 2.2.1.3. Sơ đồ chân

Sơ đồ chân và kích thước dùng trong thiết kế mạch in của vi điều khiển được trình bày trên hình 2.4 và hình 2.5. Chức năng của các chân cụ thể như sau:

* Nguồn và Flash SPI: Chân VCC cung cấp điện áp 3,3 V ổn định và GND đảm bảo kết nối đất chung; các chân GPIO6–GPIO11 chỉ phục vụ giao tiếp với Flash SPI 8 MB, không dùng cho I/O khác.
* GPIO đa năng: GPIO0–GPIO5, GPIO12–GPIO15, GPIO17–GPIO27 và GPIO32–GPIO39 hỗ trợ digital I/O, PWM, ADC 12-bit, DAC 8-bit, cảm biến điện dung touch sense và ngắt ngoài (external interrupt).
* GPIO chuyên biệt: GPIO34–GPIO39 chỉ nhận tín hiệu (input-only) và không tích hợp pull-up/pull-down; GPIO17 và GPIO18 là hai kênh DAC để xuất tín hiệu analog.
* Giao tiếp I²C và SPI: Bus I²C mặc định trên SDA=GPIO21, SCL=GPIO22; bus SPI VSPI với SCLK=GPIO18, MOSI=GPIO23, MISO=GPIO19, CS=GPIO5 cho tốc độ lên đến 80 MHz.
* UART và các giao thức khác: UART0 (TX0=GPIO1, RX0=GPIO3) và UART1 (TX1=GPIO10, RX1=GPIO9) có thể cấu hình lại qua GPIO matrix; module còn hỗ trợ I²S, RMT, CAN và SDIO cho âm thanh, điều khiển từ xa, ứng dụng công nghiệp và thẻ nhớ SD.
* Touch sensor và RTC\_GPIO: 10 kênh touch sensor trên các chân GPIO0, 2, 4, 12–15, 27, 32–33; bất kỳ chân RTC\_GPIO nào cũng có thể dùng để đánh thức module từ chế độ deep-sleep.
* Khuyến cáo bố trí PCB: Đặt tụ lọc nguồn 0,1 µF kết hợp 10 µF sát chân VCC để ổn định; giữ khoảng cách ≥5 mm giữa anten PCB và chi tiết kim loại; sử dụng ground pour rộng xung quanh khu vực RF và duy trì các đường tín hiệu cao tốc có impedance kiểm soát.

A diagram of a computer

AI-generated content may be incorrect.

* + - * 1. Sơ đồ chức năng các chân của ESP32-S3-MINI-1-N8

A diagram of a computer component

AI-generated content may be incorrect.

* + - * 1. Kích thước của ESP32-S3-MINI-1-N8

### 2.2.2. Cảm biến MAX30102

#### 2.2.2.1. Giới thiệu



* + - * 1. Cảm biến Max30102

Cảm biến nhịp tim và oxy trong máu MAX30102 được sử dụng để đo nhịp tim và nồng độ Oxy trong máu, thích hợp cho nhiều ứng dụng liên quan đến y sinh. cảm biến sử dụng phương pháp đo quang phổ biến hiện nay với thiết kế và chất liệu mắt đo chuyên biệt từ chính hãng Maxim cho độ chính xác và độ bền cao, nó bao gồm đèn LED bên trong, bộ tách sóng quang, các bộ phận quang học và các thiết bị điện tử, có tiếng ồn thấp với khả năng loại bỏ ánh sáng xung quanh. Cảm biến sử dụng giao tiếp I2C với bộ thư viện sẵn có trên Arduino nên rất dễ sử dụng. Một số ưu điểm của cảm biến:

* Dòng tải cực thấp, 600μA (chế độ đo) và 0,7μA (chế độ chờ). Do đó, nó là một lựa chọn tuyệt vời để sử dụng trong các thiết bị đeo được như đồng hồ thông minh, v.v.
* Nó có khả năng tốc độ lấy mẫu cao cùng với khả năng xuất dữ liệu nhanh.
* Tích hợp tính năng khử ánh sáng xung quanh.
* Để giao tiếp với vi điều khiển, cảm biến sử dụng chân I2C SCL và SDA.
* Một tính năng khác của cảm biến này là nó sử dụng bộ đệm FIFO (First-In-First-Out) 32 mẫu để lưu trữ dữ liệu so với MAX30100 chỉ có 16 bộ đệm FIFO mẫu. Nói cách khác, nó tiếp tục làm giảm mức tiêu thụ điện năng vì nó đã giữ tối đa ba mươi hai giá trị nhịp tim và SpO2.
* MAX30102 cũng có thể được sử dụng với các ngắt, có thể được bật cho một số nguồn như sẵn sàng cấp nguồn, sẵn sàng dữ liệu mới, khử ánh sáng xung quanh, sẵn sàng nhiệt độ và FIFO gần đầy. Với việc tạo ra ngắt, vi điều khiển có thể thực hiện các sự kiện khác không xảy ra trong quá trình thực thi tuần tự chương trình trong khi cảm biến liên tục lấy các mẫu dữ liệu mới.

#### 2.2.2.2. Thông số kỹ thuật

* Cảm biến: MAX30102
* Thông số đo lường: Nhịp tim (BPM), Độ bão hòa oxy trong máu (SpO2)
* Điện áp hoạt động: 1,8V đến 3,3V DC
* Giao diện truyền thông: I2C
* Tiêu thụ dòng điện: Công suất thấp (thường là 0,8mA khi hoạt động)
* Nhiệt độ hoạt động: -400C đến +850C
* Kích thước: 1.9 cm x 1.4 cm x 0.3 cm

### 2.2.3. Cảm biến LIS2DH12TR

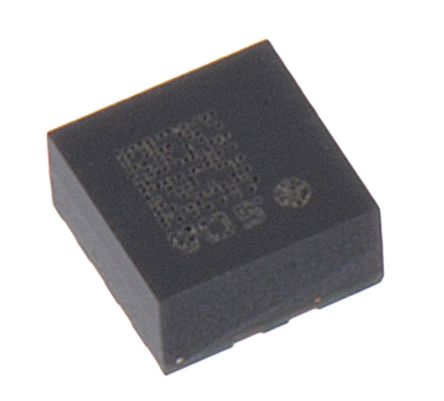
Cảm biến gia tốc MEMS LIS2DH12TR của STMicroelectronics là module 3 trục, đo gia tốc tĩnh và động với độ nhạy cao và tiêu thụ điện năng thấp, phù hợp cho các thiết bị đeo tay, điện thoại thông minh và ứng dụng IoT. Được đóng gói trong QFN-16 kích thước 3 × 3 mm, LIS2DH12TR trang bị vi điều khiển nội bộ để xử lý ngưỡng ngắt, FIFO và giao diện hướng đến ứng dụng MCU.

LIS2DH12TR cung cấp dải đo ±2/±4/±8/±16 g có thể cấu hình động, với độ phân giải 12-bit giúp phát hiện chuyển động tinh vi. Điện áp cung cấp từ 1,8 đến 3,6 V và độ tiêu thụ chỉ 2 µA ở chế độ low-power, hỗ trợ đa cấp tốc độ dữ liệu (ODR) từ 1 Hz đến 5 kHz. Tích hợp bộ lọc số và mạch FIFO 32 byte giảm tải cho MCU, cho phép ghi nhận 3 giá trị trục X, Y, Z liên tục và phát tín hiệu ngắt khi đầy FIFO hoặc khi vượt ngưỡng chuyển động đã định.

Module hỗ trợ giao tiếp I²C lên đến 400 kHz và SPI 4 dây max 10 MHz, với thanh ghi cấu hình linh hoạt để điều chỉnh full-scale, bandwidth, filter modes và threshold. Cảm biến còn cung cấp 2 chân ngắt (INT1, INT2) có thể lập trình cho các sự kiện như free-fall, tap/double-tap, activity/inactivity, zéro motion, slope detection, thuận tiện trong phát hiện sự kiện thời gian thực. Ngoài ra, LIS2DH12TR tích hợp pinpoint self-test, cho phép đánh giá tình trạng cảm biến tại chỗ.

Thông số kỹ thuật:

* Điện áp hoạt động: 1.8–3.6 V
* Dải đo: ±2/±4/±8/±16 g
* Độ phân giải: 12-bit (LSB=1 mg ở ±2 g)
* ODR: 1 Hz–5 kHz (tùy chế độ)
* Tiêu thụ: 2 µA (low-power), 8 µA–200 µA (ODR phụ thuộc)
* FIFO: 32 byte, 3 trục 16-bit giá trị
* Ngắt: 2 chân (INT1, INT2) cho free-fall, tap, activity/inactivity, zero motion, slope detection
* Giao tiếp: I²C έως 400 kHz, SPI 4 dây đến 10 MHz
* Nhiệt độ hoạt động: −40 °C đến 85 °C
* Package: QFN-16, 3 × 3 × 0.9 mm, pad pitch 0.5 mm.



* + - * 1. Cảm biến LIS2DH12TR

### 2.2.4. Module định vị GPS L80-R

Module L80-R của Quectel là một mô-đun GPS nhỏ gọn, tích hợp sẵn anten patch 15 mm × 15 mm × 4 mm và bộ khuếch đại nhiễu thấp (LNA), cung cấp hiệu suất định vị cao trong các ứng dụng công nghiệp và dân dụng. Với bộ xử lý GPS dựa trên nền tảng MTK, L80-R hỗ trợ 66 kênh tìm kiếm và 22 kênh theo dõi đồng thời, cho phép xác định vị trí nhanh chóng ngay cả trong các môi trường tín hiệu yếu. Kích thước module chỉ 16 × 16 × 6.95 mm, sử dụng kiểu chân LCC 12 chân, pitch 2.54 mm, thuận tiện trong việc hàn dán và tích hợp lên bo mạch chính.



* + - * 1. Module GPS L80-R

Một số công nghệ đặc biệt tích hợp như AIC (Active Interference Cancellation) giúp loại bỏ nhiễu RF từ các linh kiện khác, và EASY (Embedded Assist System) giúp giảm thời gian định vị TTFF thông qua việc tự động dự đoán dữ liệu quỹ đạo vệ tinh. Điều này làm cho L80-R đặc biệt phù hợp với các thiết bị định vị tiêu thụ năng lượng thấp như đồng hồ đeo tay, thiết bị theo dõi sức khỏe, xe đạp định vị, drone và thiết bị IoT.

Module L80-R cung cấp giao tiếp UART đơn giản, hỗ trợ cả chuẩn NMEA và giao thức riêng PMTK của MTK, tương thích tốt với các dòng vi điều khiển phổ biến. Module hoạt động ổn định trong dải nhiệt độ rộng từ –40 °C đến 85 °C và hoàn toàn tuân thủ tiêu chuẩn RoHS của EU.

##### 2.2.4.2. Thông số kỹ thuật

Bảng thông số kỹ thuật Module L80-R

|  |  |
| --- | --- |
| **Thông số** | **Giá trị** |
| Điện áp cấp (VCC) | 2.8 V – 4.3 V (typ. 3.3 V) |
| Điện áp backup (V\_BCKP) | 2.0 V – 4.3 V (typ. 3.3 V) |
| Dòng tiêu thụ | 25 mA (acquisition), 20 mA (tracking), 1 mA (standby), 7 µA (backup) |
| Dải tần | GPS L1 1575.42 MHz C/A code |
| Số kênh | 66 kênh tìm kiếm, 22 kênh theo dõi |
| Độ nhạy | –148 dBm (acquisition), –165 dBm (tracking) |
| TTFF (EASY enabled) | Cold: 15s, Warm: 5s, Hot: 1s @ –130 dBm |
| TTFF (EASY disabled) | Cold: 35s, Warm: 30s, Hot: 1s @ –130 dBm |
| Độ chính xác vị trí ngang | < 2.5 m CEP @ –130 dBm |
| Độ chính xác tốc độ | 0.1 m/s |
| Độ chính xác gia tốc | 0.1 m/s² |
| PPS Accuracy | ±10 ns, pulse width: 100 ms |
| Giao tiếp | UART (4800–115200 bps, mặc định 9600 bps) |
| Kích thước vật lý | 16 × 16 × 6.95 mm ±0.15 mm |
| Trọng lượng | Khoảng 6.0 g |
| Nhiệt độ hoạt động | –40 °C đến +85 °C |
| Nhiệt độ lưu trữ | –45 °C đến +125 °C |

##### 2.2.4.3. Sơ đồ chân

A close-up of a grey square

AI-generated content may be incorrect.

* + - * 1. Sơ đồ chân của L80-R

Module L80-R sử dụng 12 chân LCC với khoảng cách chuẩn 2.54 mm, được bố trí thuận tiện cho việc tích hợp lên PCB. Bảng dưới đây liệt kê tên chân, chức năng và mô tả điện áp:

Bảng sơ đồ chân L80-R

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Chân | Tên | I/O | Mô tả | Điện áp hoạt động |
| 1 | RXD1 | I | Nhận dữ liệu UART | VIH: 2.0–3.6 V |
| 2 | TXD1 | O | Gửi dữ liệu UART | VOH: 2.4–3.1 V |
| 3 | GND | - | Nối đất | - |
| 4 | VCC | I | Nguồn chính cho module | 2.8–4.3 V |
| 5 | V\_BCKP | I | Nguồn dự phòng RTC | 2.0–4.3 V |
| 6 | 1PPS | O | Xung định thời 1 giây | VOH: 2.4–3.1 V |
| 7 | RESERVED | - | Không sử dụng | - |
| 8 | RESERVED | - | Không sử dụng | - |
| 9 | NC | - | Không kết nối | - |
| 10 | RESET | I | Reset mức thấp (Low active) | VIL < 0.8 V |
| 11 | RESERVED | - | Không sử dụng | - |
| 12 | GND | - | Nối đất | - |

### 2.2.5. Màn hình IPS ST7789 1.3″ 240 × 240

#### 2.2.5.1. Giới thiệu

ST7789 1.3″ 240 × 240 Dot-Matrix Graphic TFT (7-pin SPI) là màn hình LCD IPS kích thước 1.3 inch với độ phân giải 240 × 240 pixel, được điều khiển bởi IC Sitronix ST7789. Bộ điều khiển này tích hợp sẵn các lệnh vẽ điểm ảnh, đường thẳng, hình chữ nhật và ký tự, giúp thuận tiện khi hiển thị giao diện đồ họa phong phú. Màn hình sử dụng giao tiếp SPI chuẩn 4 dây, cùng với 3 chân điều khiển (CS, DC, RST) và 1 chân nguồn LED nền, tổng cộng 7 chân đơn giản, phù hợp cho các bo mạch nhúng như ESP32, STM32 hoặc Arduino.

Tấm nền IPS (In-Plane Switching) của module mang lại góc nhìn rộng lên đến ~80°, màu sắc trung thực và độ tương phản cao. Bảng màu lên đến 65.536 màu (18-bit RGB nội bộ) cho phép hiển thị các hình ảnh và biểu đồ sắc nét, sống động. Tốc độ làm tươi tối đa 60 Hz cùng thời gian phản hồi nhanh phù hợp cho các ứng dụng đồ họa động và chơi game đơn giản trên thiết bị di động hoặc thiết kế giao diện người dùng trong các dự án IoT.

Về cơ bản, module hoạt động ở điện áp 2.4–3.3 V cho logic và nguồn LED nền, tiêu thụ khoảng 10 mA (không bao gồm LED nền) và ~20 mA khi bật đầy sáng. Kích thước tấm nền hoạt động (AA) là 23.4 × 23.4 mm, trong khi kích thước tổng thể module vào khoảng 29 × 29 mm, giúp dễ dàng gắn vào các thiết kế PCB nhỏ gọn. Thiết kế 7 chân tín hiệu đơn giản không chỉ tối ưu hóa lượng dây nối mà còn giảm thiểu tài nguyên GPIO trên vi điều khiển, đồng thời vẫn đáp ứng đầy đủ các chức năng điều khiển màn hình cơ bản.

A blue and black electronic device with a blue and red design

AI-generated content may be incorrect.

* + - * 1. Màn hình IPS ST7789 1.3″ 240 × 240

#### 2.2.5.2. Sơ đồ chân và thông số kỹ thuật

Bảng thông số kỹ thuật màn hình IPS ST7789 1.3″ 240 × 240

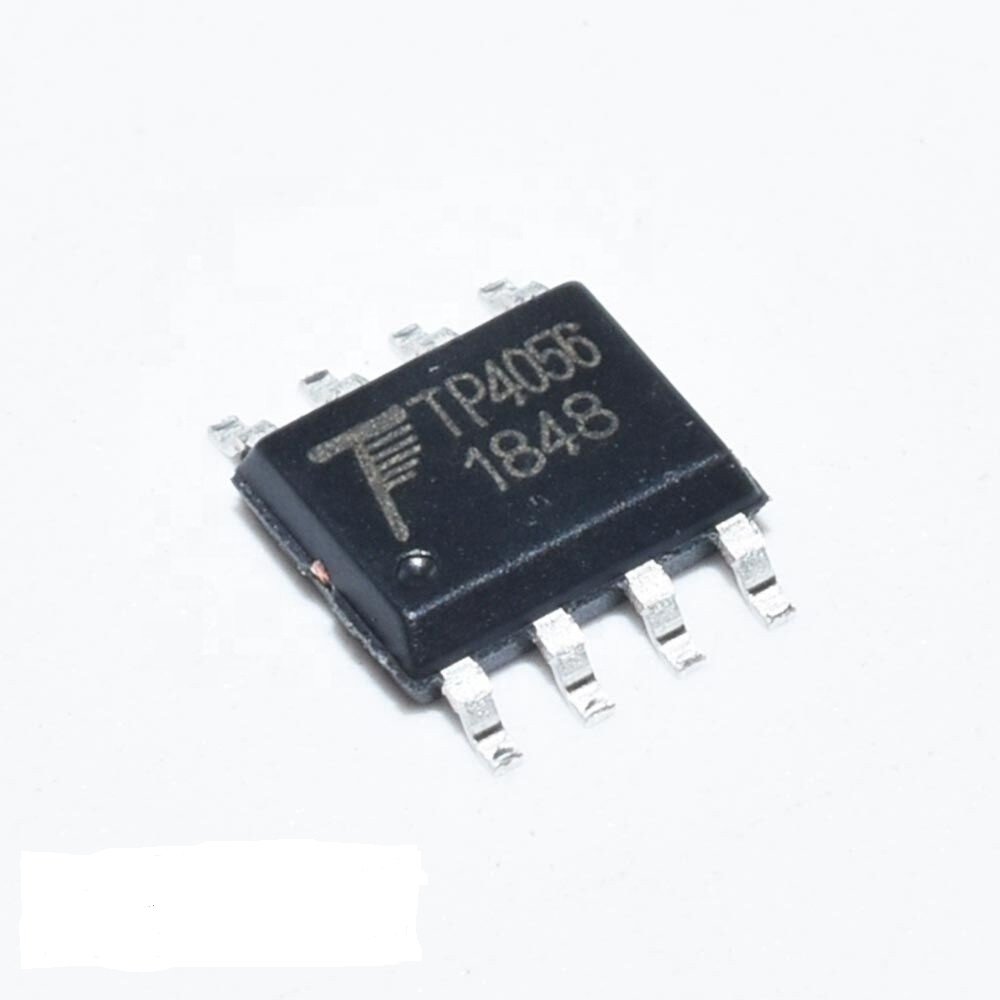
| **Thông số** | **Giá trị điển hình** |
| --- | --- |
| Bộ điều khiển (Driver IC) | Sitronix ST7789 |
| Kích thước hiển thị | 1.3 inch diagonal |
| Độ phân giải | 240 × 240 pixels |
| Khu vực hoạt động (AA) | 23.40 × 23.40 mm |
| Giao tiếp | 4-wire SPI |
| Điện áp nguồn (VDD) | 2.4 – 3.3 V |
| Điện áp I/O (VDDI) | 1.65 – 3.3 V |
| Màu hiển thị | 65.536 màu (RGB 18-bit nội bộ) |
| Góc nhìn | ≈ 80° (IPS) |
| Tần số làm tươi | Up to 60 Hz |
| Dòng tiêu thụ (không nền LED) | ~10 mA |
| Dòng nền LED | ~20 mA |
| Nhiệt độ hoạt động | –20 °C – +70 °C |
| Kích thước module | ~29 × 29 mm (bao gồm khung) |

Bảng chân tín kiệu màn hình IPS ST7789 1.3″ 240 × 240

| **Chân** | **Tên tín hiệu** | **Chức năng** |
| --- | --- | --- |
| 1 | VCC | Nguồn 3.3 V |
| 2 | GND | Mass |
| 3 | SCL (CLK) | SPI clock |
| 4 | SDA (DIN) | SPI MOSI (data input) |
| 5 | CS | Chip select, active-low |
| 6 | DC | Data/Command select (H=data, L=command) |
| 7 | RST | Reset module, active-low |

### 2.2.6. IC sạc TP4056 và IC LDO LP2985-33DBVR

*2.2.6.1. TP4056 – IC Sạc Li-Ion Linear 1 A*



* + - * 1. TP4056 – IC Sạc Li-Ion Linear 1 A (SOP-8)

TP4056 là một IC sạc tuyến tính (linear charger) chuyên dụng cho pin lithium-ion/lithium-polymer đơn cell, tích hợp đầy đủ mạch điều khiển sạc theo phương pháp Constant-Current/Constant-Voltage (CC/CV), bao gồm cả giai đoạn trickle charge cho pin đã xả sâu. IC này không yêu cầu MOSFET hay diode ngoài, cho phép thiết kế bo mạch đơn giản và chi phí thấp. TP4056 còn tích hợp chức năng bảo vệ quá dòng, bảo vệ quá áp, giám sát nhiệt độ qua chân TEMP, tự động tái sạc khi điện áp pin xuống dưới mức định sẵn và kết thúc sạc ở dòng C/10 để ngăn ngừa tình trạng overcharge.

Thông số kỹ thuật:

* Điện áp đầu vào (IN): 4,0 – 8,0 V (thích hợp nguồn USB 5 V)
* Điện áp sạc cố định (BAT): 4,200 V ± 1,5 %
* Dòng sạc tối đa: Lập trình qua R\_PROG, tối đa 1 000 mA (R\_PROG = 1,2 kΩ ↔ 1 A)
* Chế độ trickle-charge: Tự động sạc nhỏ (~140 mA) khi BAT < 2,9 V
* Kết thúc sạc (termination): Khi I\_CHG < C/10 (≈100 mA), dòng rò khi kết thúc ≤ 55 µA
* Soft-start nội: Giảm dòng đỉnh ban đầu, bảo vệ nguồn
* Chức năng bảo vệ: Bảo vệ quá dòng, quá áp; giám sát nhiệt độ qua chân TEMP
* Chỉ báo trạng thái: Hai chân STAT1/STAT2 điều khiển LED báo SẠC / HOÀN THÀNH
* Dòng tắt (shutdown): ≤ 150 µA
* Nhiệt độ hoạt động: –40 °C đến +85 °C
* Đóng gói: SOP-8, hỗ trợ tản nhiệt qua pad GND lớn

*2.2.6.2. LP2985-33DBVR – IC LDO 3.3 V 150 mA*

LP2985-33DBVR của Texas Instruments là bộ điều áp tuyến tính (LDO) cố định đầu ra 3,3 V, tối ưu cho cấp nguồn vi điều khiển và các mạch số tiêu thụ dòng không lớn (≤ 150 mA). IC này nổi bật với độ ồn rất thấp, PSRR cao, và khả năng chịu đựng dải điện áp đầu vào rộng (2,5–16 V), giúp bảo vệ mạch khỏi nhiễu nền do các nguồn chuyển mạch trước đó. LP2985-33DBVR cũng tích hợp chân Enable để dễ dàng điều khiển tắt/mở đầu ra, cùng các tính năng bảo vệ quá dòng và quá nhiệt đảm bảo an toàn cho ứng dụng.



* + - * 1. *LP2985-33DBVR – IC LDO 3.3 V 150 mA*

Thông số kỹ thuật

* Điện áp đầu vào (VIN): 2,5 – 16 V
* Điện áp đầu ra cố định (VOUT): 3,300 V ± 1 % (A-grade)
* Dòng tải tối đa: 150 mA
* Dòng tiêu thụ không tải (IQ): ~95 µA
* Dòng rò ở chế độ shutdown: ≤ 0,01 µA
* Ổn áp soft-start: Hạn chế dòng inrush khi khởi động
* Bảo vệ: Giới hạn dòng quá tải và ngắt khi quá nhiệt
* PSRR: > 70 dB @ 1 kHz; > 40 dB @ 1 MHz
* Độ ồn đầu ra: ~30 µV RMS (với tụ bypass 10 nF)
* Chân Enable (EN): Điều khiển bật/tắt LDO
* Nhiệt độ hoạt động: –40 °C đến +125 °C
* Đóng gói: SOT-23-5 (2,9 × 1,6 mm)

### 2.2.7. Các linh kiện khác

Dưới đây là bảng liệt kê các linh kiện khác:

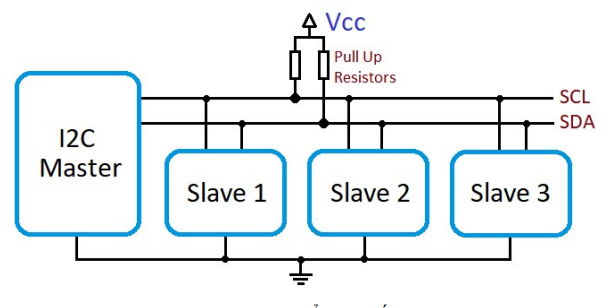
Bảng các linh kiện khác được sử dụng trong dự án.

| **STT** | **Tên linh kiện** | **Mô tả** | **Hình ảnh** |
| --- | --- | --- | --- |
| 1 | R và C | Điện trở và tụ điện kích thước 0603 chân dán | 0603 Resistors - SMD 1 MΩ 1 % 0603 Thick Film Resistors - SMD 100KPF 0603 Capacitors - SMD/SMT Multilayer Ceramic MLCC |
| 2 | BC817 | Transistor NPN BC817 | BC817 Transistor SMD Pinout, Datasheet, Equivalent, Circuit & Specs |
| 3 | BSS138 | MOSFET N-kênh BSS138 cho chuyển mạch logic | BSS138 Fairchild SOT-23 N-ch 50V/0.22A MOSFET |
| 4 | CH340C | IC chuyển USB ↔ UART, chip CH340C | Buy CH340C SMD IC online in India from DNA Solutions, Nashik |
| 5 | Micro\_SD\_Card | Socket thẻ nhớ Micro SD | Buy online MICRO SD MEMORY CARD SOCKET (TF-07F) in India at low cost ... |
| 6 | SW\_Push | Nút nhấn (push-button) | Nút Nhấn 3x4mm Cao 2.5mm 2 Chân SMD 12VDC 0.1A |
| 7 | USB\_C\_Plug\_USB2.0 | Cổng USB-C 2.0 |  |

## 2.3. Các chuẩn giao tiếp

### 2.3.1. Giao tiếp I2C

Đầu năm 1980 Phillips đã phát triển một chuẩn giao tiếp nối tiếp 2 dây được gọi là I2C. I2C là tên viết tắt của cụm từ Inter-Intergrated Circuit. Đây là đường Bus giao tiếp giữa các IC với nhau. I2C mặc dù được phát triển bởi Phillips, nhưng nó đã được rất nhiều nhà sản xuất IC trên thế giới sử dụng. I2C trở thành một chuẩn công nghiệp cho các giao tiếp điều khiển, có thể kể ra đây một vài tên tuổi ngoài Phillips như: Texas Instrument(TI), MaximDallas, analog Device, National Semiconductor, vv. Bus I2C được sử dụng làm bus giao tiếo ngoại vi cho rất nhiều loại IC khác nhau như các loại Vi điều khiển 8051, PIC, AVR, ARM, vv chip nhớ như: RAM tĩnh (Static RAM), EEPROM, bộ chuyển đổi tương tự số (ADC), số tương tự (DAC), IC điều khiển LCD, LED, vv. I2C sử dụng hai đường truyền tín hiệu: • Một đường xung clock (SCL) chỉ do Master phát đi (thông thường ở 100kHz và 400kHz. Mức cao nhất là 1MHz và 3.4MHz). • Một đường dữ liệu (SDA) theo 2 hướng.

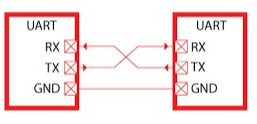


* + - * 1. Chuẩn giao tiếp I2C

Có rất nhiều thiết bị có thể cũng được kết nối vào một bus I2C, tuy nhiên sẽ không xảy ra chuyện nhầm lẫn giữa các thiết bị, bởi mỗi thiết bị sẽ được nhận ra bởi một địa chỉ duy nhất với một quan hệ master/slave tồn tại trong suốt thời gian kết nối. Mỗi thiết bị có thể hoạt động như là thiết bị nhận hoặc truyền dữ liệu hay có thể vừa truyền vừa nhận. Hoạt động truyền hay nhận còn tùy thuộc vào việc thiết bị đó là master hay slave. Một thiết bị hay một IC khi kết nối với bus I2C, ngoài một địa chỉ (duy nhất) để phân biệt, nó còn được cấu hình là thiết bị master hay slave. Đó là vì trên một bus I2C thì quyền điều khiển thuộc về thiết bị master hay slave. Thiết bị master nắm vai trò tạo xung clock đồng bộ cho toàn hệ thống, khi giữa hai thiết bị master – slave giao tiếp thì thiết bị master có nhiệm vụ tạo xung clock đồng bộ và quản lý địa chỉ của thiết bị slave trong suốt quá trình giao tiếp. Thiết bị master giữ vai trò chủ động, còn thiết bị slave giữ vai trò bị động trong việc giao tiếp. Về lý thuyết lẫn thực tế I2C sử dụng 7 bit để định địa chỉ, do đó trên một bus có thể có tới 27 địa chỉ tương ứng với 128 thiết bị có thể kết nối, nhưng chỉ có 112, 16 địa chỉ còn lại được sử dụng vào mục đích riêng. Bit còn lại quy định việc đọc hay ghi dữ liệu (1 là write, 0 là read). Có một lưu ý nhỏ về xung clock. Bản chất của I2C là dữ liệu trên đường SDA chỉ được ghi nhận ở sườn lên của chân CLK. Do vậy xung clock có thể không cần chính xác tốc độ là 1MHz hay 3.4MHz. Lợi dụng điểm này có thể sử dụng 2 chân GPIO để làm chân giao tiếp I2C mềm mà không nhất thiết cần một chân CLK tạo xung với tốc độ chính xác. Quá trình truyền dữ liệu giữa 2 thiết bị chế độ Master – Slave dùng bus I2C diễn ra như sau: Thiết bị master tạo xung START (tức là đường SDA thay đổi từ mức cao xuống mức thấp và đường SCL đang ở mức cao) để bắt đầu quá trình giao tiếp. Thiết bị master gửi địa chỉ của thiết bị slave muốn giao tiếp cùng với bit R/W và đợi xung ACK phản hồi. Địa chỉ của thiết bị slave được định nghĩa bởi 7 bit, ngoài ra có thể đánh địa chỉ dưới dạng 10 bit nếu sau lệnh START gửi chuỗi 11110 ra đường SDA. Bit R/W dùng để điều khiển hướng truyền- bit “0” là truyền từ master tới slave, ngược lại bit “1” là truyền từ slave tới master. Việc thiết lập bit R/W do thiết bị master quy định. Khi nhận được xung báo ACK từ thiết bị slave xác nhận đúng địa chỉ thì bắt đầu thực hiện truyền dữ liệu. Dữ liệu được gửi theo từng byte. Mỗi byte gồm 8 bit và sau mỗi byte đều bắt buộc có một xung ACK để đảm bảo quá trình truyền nhận diễn ra chính xác. Số lượng byte là không giới hạn. Xung ACK được định nghĩa là SDA kéo xuống mức thấp. Ngoài ra, khi không nhận được đúng địa chỉ hay khi muốn kết thúc quá trình giao tiếp thiết bị nhận sẽ gửi một xung NACK tức là SDA ở mức cao. Kết thúc quá trình truyền, thiết bị master sẽ gửi xung STOP hoặc thiết bị slave gửi xung NACK để báo hiệu kết thúc. Xung STOP tương tự như xung START nhưng trạng thái của đường SDA thay đổi từ mức thấp lên mức cao. Ngoài ra, trong một quá trình giao tiếp có thể lặp lại xung START kèm theo một địa chỉ mới để bắt đầu một giao tiếp khác. Điểm mạnh của I2C chính là hiệu suất và sự đơn giản của nó: một khối điều khiển trung tâm có thể điều khiển cả một mạng thiết bị mà chỉ cần hai lối ra điều khiển.

### 2.3.2. Giao tiếp UART

UART chuyển đổi giữa dữ liệu nối tiếp và song song. Một chiều UART chuyển đổi dữ liệu song song bus hệ thống ra dữ liệu nối tiếp để truyền đi. Một chiều khác UART chuyển đổi dữ liệu nhận được dạng dữ liệu nối tiếp thành dạng dữ liệu song song cho CPU có thể đọc vào bus hệ thống.

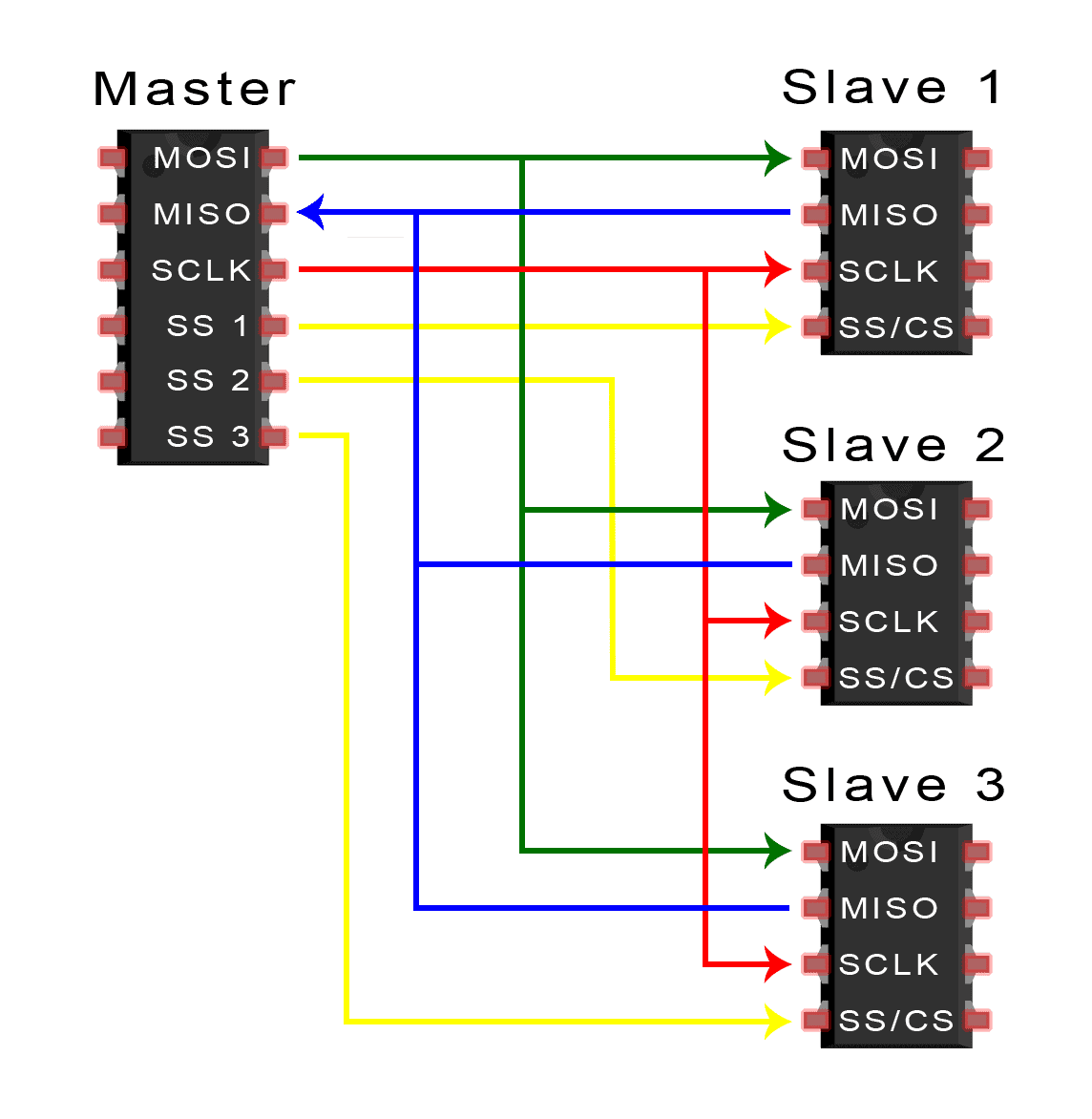


* + - * 1. Truyền UART

UART hỗ trợ cả hai kiểu giao tiếp là giao tiếp đồng thời và giao tiếp không đồng thời. Giao tiếp đồng thời tức là UART có thể gửi và nhận dữ liệu vào cùng một thời điểm. Còn giao tiếp không đồng thời là chỉ có một thiết bị có thể chuyển dữ liệu vào một thời điểm, với tín hiệu điều khiển hoặc mã sẽ quyết định bên nào có thể truyền dữ liệu. Giao tiếp không đồng thời được thực hiện khi mà cả 2 chiều chia sẽ một đường dẫn hoặc nếu có 2 đường nhưng cả 2 thiết bị chỉ giao tiếp qua một đường ở cùng một thời điểm. Thêm vào đường dữ liệu, UART hỗ trợ bắt tay chuẩn RS232 và tín hiệu điều khiển như RTS, CTS, DTR, DCR, RT và CD. Để thuận tiện, các chương trình gửi và nhận dữ liệu trong định dạng không đồng bộ đơn giản hơn. PC và nhiều vi xử lý khác có một bộ phận gọi là UART (universal asynchronous receiver/transmitter: truyền /nhận không đồng bộ chung) vì thế có thể vận dụng phần lớn những chi tiết truyền và nhận dữ liệu. Trong PC, hệ điều hành và ngôn ngữ lập trình hỗ trợ cho lập trình liên kết nối tiếp mà không cần phải hiểu rõ chi tiết cấu trúc UART. Để mở liên kết, ứng dụng lựa chọn một tần số dữ liệu hoặc là thiết lập khác hoặc cho phép truyền thông tại các cổng. Để gửi 1 byte, ứng dụng ghi byte này vào bộ đệm truyền của cổng được lựa chọn, và UART gửi dữ liệu này, từng bit một, trong định dạng yêu cầu, thêm bit Start, bit Stop, bit chẵn lẻ khi cần. Trong một cách đơn giản, byte nhận được tự động được lưu trữ trong bộ đệm. UART có thể dùng nhanh một ngắt để báo cho CPU và các ứng dụng biết dữ liệu đang nhận được và các sự kiện khác. Một vài vi điều khiển không bao gồm UART, và thỉnh thoảng bạn cần nhiều hơn các UART mà vi xử lý có. Trong trường hợp này, có 2 lựa chọn: thêm UART ngoài, hoặc mô phỏng UART trong mã chương trình. Basic Stamp của Parallax là một ví dụ của chip với một UART bổ sung trong mã chương trình. UART là một thiết bị đơn giản hỗ trợ tốt cả hai kiểu truyền thông đồng bộ và không đồng bộ. Các thông số cơ bản của chuẩn truyền: - Baud rate (tốc độ Baud): Khi truyền nhận không đồng bộ để hai module hiểu được nhau thì cần quy định một khoảng thời gian cho 1 bit truyền nhận, nghĩa là trước khi truyền thì tốc độ phải được cài đặt đầu tiên. Theo định nghĩa thì tốc độ baud là số bit truyền trong một giây. - Frame (khung truyền): Do kiểu truyền thông nối tiếp này rất dễ mất dữ liệu nên ngoài tốc độ, khung truyền cũng được cài đặt từ ban đầu để giảm bớt sự mất mát dữ liệu này. Khung truyền quy định số bit trong mỗi lần truyền, các bit thông báo như start, stop, các bit kiểm tra như parity, và số bit trong một data. - Bit Start: Là bit bắt đầu trong khung truyền. Bit này nhằm mục đích báo cho thiết bị nhận biết quá trình truyền bắt đầu. Trên AVR bit Start có trạng thái là 0. - Data: Dữ liệu cần truyền data không nhất thiết phải 8 bit có thể là 5, 6, 7, 8, 9. Trong UART bit LSB được truyền đi trước, bit MSB được truyền đi sau. - Parity bit: Là bit kiểm tra dữ liệu. Có 2 loại parity: chẵn (even parity), lẻ (old parity). Parity chẵn là bit parity thêm vào để số số 1 trong data + parity = chẵn. Parity lẻ là bit parity thêm vào để số số 1 trong data + parity = lẻ. Bit parity là không bắt buộc nên có thể dùng hoặc không.

### 2.3.3. Giao thức SPI

SPI (Serial Peripheral Interface) là một chuẩn giao tiếp nối tiếp đồng bộ, được phát triển bởi Motorola vào giữa thập niên 1980. SPI được thiết kế để truyền dữ liệu tốc độ cao giữa một vi điều khiển (master) và một hoặc nhiều thiết bị ngoại vi (slave). Với kiến trúc đơn giản, hiệu suất cao và dễ triển khai, giao thức SPI hiện đang là một trong những giao thức truyền thông phổ biến nhất trong các hệ thống nhúng, đặc biệt trong các ứng dụng yêu cầu truyền tải dữ liệu nhanh như giao tiếp với bộ nhớ flash, cảm biến tốc độ cao, màn hình LCD, mô-đun Wi-Fi hoặc các IC chuyên dụng.



* + - * 1. Truyền SPI

SPI hoạt động theo mô hình **master–slave**, trong đó một thiết bị điều khiển chính (master) sẽ khởi tạo quá trình truyền dữ liệu, tạo xung đồng bộ (clock) và điều khiển chọn thiết bị (chip select – CS). Các thiết bị ngoại vi (slave) sẽ chỉ phản hồi khi được master chủ động chọn thông qua chân CS riêng biệt.

SPI sử dụng tối thiểu 4 đường dây truyền tín hiệu:

1. **MOSI (Master Out – Slave In):** Dây truyền dữ liệu từ master sang slave.
2. **MISO (Master In – Slave Out):** Dây truyền dữ liệu từ slave sang master.
3. **SCLK (Serial Clock):** Tín hiệu xung đồng hồ do master tạo ra để đồng bộ dữ liệu.
4. **CS / SS (Chip Select hoặc Slave Select):** Tín hiệu chọn thiết bị slave để truyền dữ liệu. Tín hiệu này có mức logic thấp (active-low).

Ngoài ra, một số hệ thống có thể sử dụng thêm chân **RESET** hoặc **IRQ (Interrupt Request)** để nâng cao độ tin cậy hoặc hỗ trợ ngắt dữ liệu.

SPI là giao thức truyền dữ liệu song công hoàn toàn (full-duplex), tức là master và slave có thể truyền dữ liệu đồng thời trên hai đường MOSI và MISO trong mỗi chu kỳ xung SCLK. Tuy nhiên, trong thực tế, nhiều thiết bị chỉ sử dụng một chiều dữ liệu (half-duplex) nhằm đơn giản hóa phần cứng.

SPI có thể được cấu hình linh hoạt theo nhiều thông số quan trọng:

* **Tần số xung clock (SCLK):** Có thể cấu hình linh hoạt từ vài kHz đến hàng chục MHz. Nhiều vi điều khiển hiện đại hỗ trợ tần số SPI lên tới 40–80 MHz.
* **Độ dài dữ liệu truyền:** SPI thường sử dụng khung dữ liệu 8 bit, nhưng nhiều thiết bị cho phép truyền khung 16 bit, 32 bit hoặc tùy chỉnh khác.
* **Thứ tự bit:** Mặc định là MSB first (bit cao trước), tuy nhiên có thể chọn LSB first nếu thiết bị slave hỗ trợ.
* **Chế độ pha và cực (SPI Mode):** Giao thức SPI hỗ trợ 4 chế độ hoạt động dựa trên hai thông số:
  + **CPOL (Clock Polarity):** Xác định mức logic mặc định của xung SCLK khi nhàn rỗi (0: thấp, 1: cao).
  + **CPHA (Clock Phase):** Xác định cạnh xung đồng hồ (rising hoặc falling edge) sẽ được sử dụng để lấy dữ liệu.

Bảng các chế độ của SPI

| **Mode** | **CPOL** | **CPHA** | **Nhận dữ liệu tại cạnh** |
| --- | --- | --- | --- |
| 0 | 0 | 0 | Cạnh lên (rising edge) |
| 1 | 0 | 1 | Cạnh xuống (falling edge) |
| 2 | 1 | 0 | Cạnh xuống |
| 3 | 1 | 1 | Cạnh lên |

Master và slave phải cùng cấu hình đúng chế độ SPI thì mới có thể truyền nhận chính xác. Một số thiết bị phần cứng cho phép tự động nhận dạng chế độ, nhưng phần lớn yêu cầu lập trình viên cấu hình thủ công.

Ưu điểm và hạn chế của SPI:

**Ưu điểm:**

* **Tốc độ truyền cao:** SPI có thể hoạt động ở tốc độ rất cao (lên đến hàng chục MHz), vượt trội so với các giao thức như I²C hoặc UART trong các ứng dụng đòi hỏi băng thông lớn.
* **Truyền dữ liệu song công:** Có thể gửi và nhận dữ liệu đồng thời.
* **Cấu trúc phần cứng đơn giản:** Dễ dàng triển khai trên các vi điều khiển phổ biến như STM32, AVR, ESP32, hoặc Raspberry Pi.
* **Không yêu cầu địa chỉ:** Mỗi thiết bị slave được chọn thông qua chân CS riêng biệt, tránh xung đột địa chỉ như trong giao thức I²C.

**Hạn chế:**

* **Số lượng dây tín hiệu tăng theo số lượng slave:** Mỗi thiết bị slave yêu cầu một chân chip select riêng, điều này làm tăng số lượng chân GPIO sử dụng trên master.
* **Không hỗ trợ giao tiếp đa điểm thực sự:** Chỉ có một master và một hoặc nhiều slave – không có nhiều master như I²C.
* **Không có cơ chế xác thực hoặc kiểm tra lỗi tích hợp:** SPI không có các bit parity hay CRC như một số giao thức khác. Việc kiểm tra dữ liệu phải được lập trình riêng.

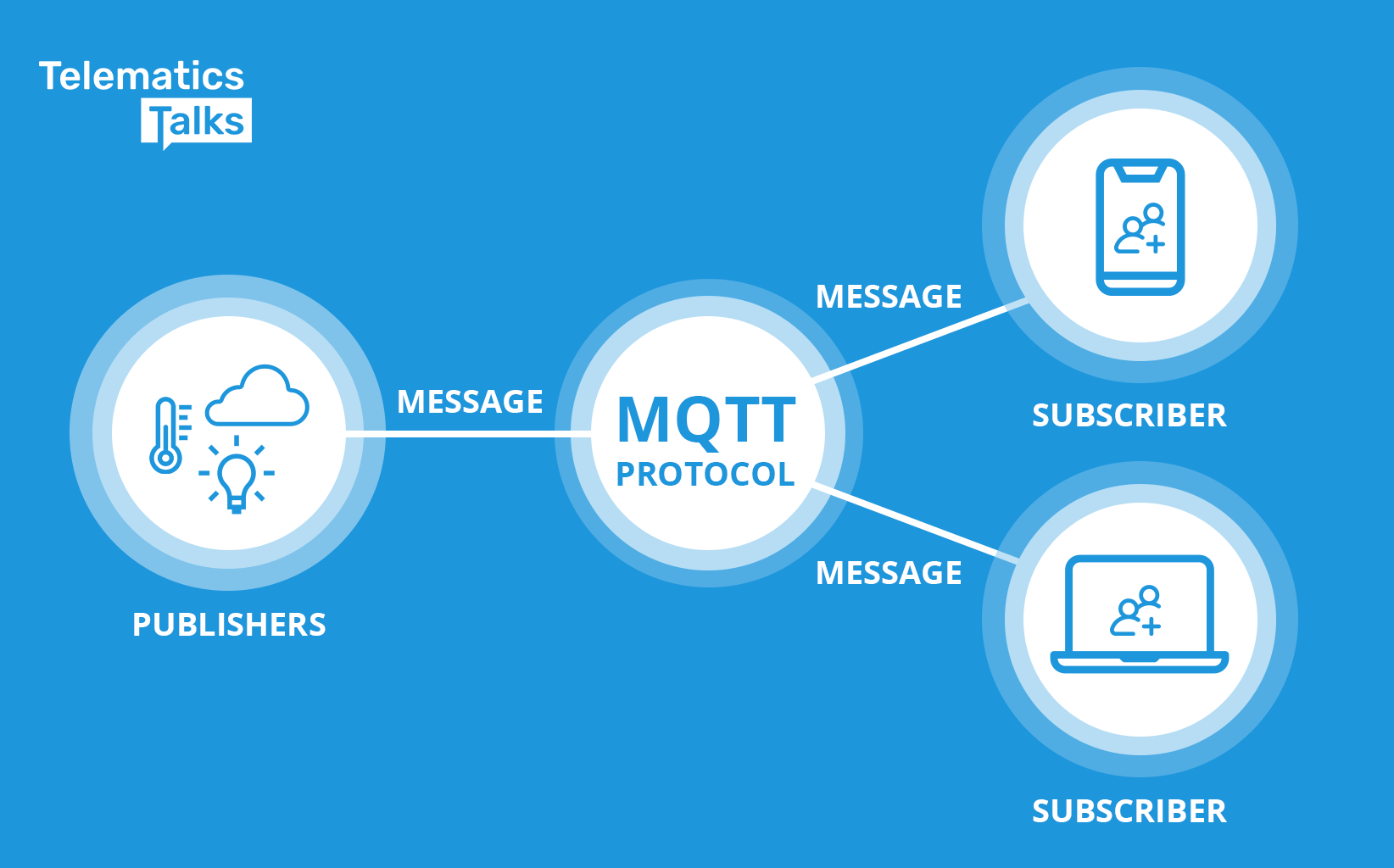
SPI được ứng dụng rộng rãi trong các hệ thống nhúng, điện tử công nghiệp và tiêu dùng nhờ tốc độ truyền cao, khả năng hoạt động ổn định và cấu trúc đơn giản. Một số ứng dụng phổ biến của SPI bao gồm:

* **Giao tiếp với bộ nhớ ngoại vi:** Flash NOR, EEPROM SPI, SRAM.
* **Kết nối với cảm biến tốc độ cao:** Gia tốc kế (accelerometer), con quay hồi chuyển (gyroscope), cảm biến áp suất hoặc nhiệt độ chuyên dụng.
* **Điều khiển hiển thị:** LCD TFT, OLED, màn hình cảm ứng, LED ma trận.
* **Kết nối module mở rộng:** Wi-Fi, Bluetooth, Ethernet, radio LoRa, thẻ nhớ SD.
* **Kết nối vi điều khiển với vi mạch đồng xử lý:** Trong các hệ thống phức tạp cần xử lý đồng thời nhiều tác vụ.

SPI là một trong những giao thức truyền thông nối tiếp mạnh mẽ và đáng tin cậy nhất trong lĩnh vực hệ thống nhúng. Nhờ tốc độ truyền cao, cấu trúc đơn giản và khả năng truyền song công, SPI trở thành lựa chọn lý tưởng trong nhiều ứng dụng yêu cầu truyền tải dữ liệu nhanh chóng và hiệu suất cao. Tuy nhiên, khi thiết kế hệ thống với nhiều thiết bị ngoại vi, cần lưu ý đến việc quản lý chân chip select và đồng bộ chế độ SPI giữa master và slave để đảm bảo hệ thống hoạt động ổn định và chính xác.

### 2.3.4. Giao thức không dây MQTT

MQTT (**Message Queuing Telemetry Transport**) là một giao thức truyền thông điệp dạng publish/subscribe nhẹ, tiết kiệm băng thông, được thiết kế dành riêng cho các ứng dụng IoT (Internet of Things). Giao thức này ra đời vào năm 1999, được phát triển bởi IBM với mục tiêu ban đầu là truyền tải dữ liệu cảm biến từ xa trong môi trường có băng thông hạn chế, độ trễ cao và không ổn định, điển hình là các ứng dụng giám sát từ xa thông qua vệ tinh. Với đặc điểm gọn nhẹ, đơn giản và hiệu quả cao, MQTT nhanh chóng được ứng dụng rộng rãi vào các hệ thống cảm biến không dây, các hệ thống giám sát sức khỏe, giám sát môi trường, hệ thống điều khiển tự động và đặc biệt là các thiết bị IoT hiện đại.



* + - * 1. Mô hình MQTT

Giao thức MQTT hoạt động dựa trên mô hình truyền nhận thông điệp bất đồng bộ theo cấu trúc client-server. Trong mô hình này, một thành phần trung tâm được gọi là Broker (máy chủ môi giới) sẽ giữ vai trò quản lý luồng thông điệp giữa các client, đảm bảo việc truyền tải thông tin diễn ra an toàn, nhanh chóng và tin cậy. Các client trong hệ thống MQTT có thể hoạt động ở hai vai trò chính: Publisher (bên gửi thông điệp) và Subscriber (bên nhận thông điệp). Các publisher sẽ gửi thông điệp lên broker thông qua các chủ đề cụ thể (topics), broker tiếp nhận và chuyển tiếp các thông điệp này đến các subscriber đã đăng ký nhận thông tin từ các topic tương ứng. Cách thức này giúp tách biệt hoàn toàn giữa người gửi và người nhận, giảm sự phụ thuộc trực tiếp giữa các thiết bị, từ đó đơn giản hóa việc phát triển và mở rộng hệ thống.

MQTT được thiết kế đặc biệt để tối ưu hóa băng thông truyền dẫn và giảm thiểu điện năng tiêu thụ cho các thiết bị sử dụng pin hoặc hoạt động trong môi trường tài nguyên hạn chế. Thông điệp MQTT có cấu trúc đơn giản, gọn nhẹ, bao gồm các trường cơ bản như topic, payload (dữ liệu), QoS (Quality of Service – mức độ đảm bảo chất lượng truyền tải) và Retain (lưu giữ trạng thái cuối cùng). MQTT cung cấp ba mức QoS cơ bản nhằm cân bằng giữa hiệu suất truyền tải và độ tin cậy của thông điệp:

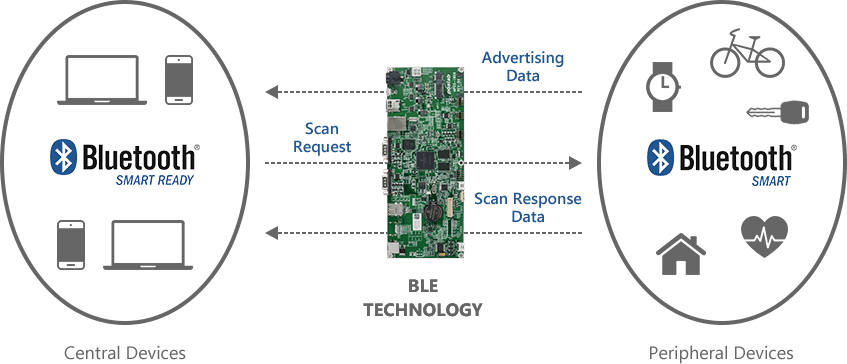
* QoS 0 (At most once): gửi thông điệp tối đa một lần, không đảm bảo chắc chắn thông điệp tới được nơi nhận, nhưng đạt tốc độ cao nhất.
* QoS 1 (At least once): đảm bảo thông điệp chắc chắn được gửi đến bên nhận ít nhất một lần, đôi khi có thể gây trùng lặp.
* QoS 2 (Exactly once): đảm bảo chắc chắn thông điệp chỉ được gửi đi chính xác một lần duy nhất, đảm bảo độ tin cậy tuyệt đối nhưng đòi hỏi nhiều tài nguyên và thời gian xử lý hơn.

Giao thức MQTT hoạt động tốt trên nền TCP/IP, hỗ trợ bảo mật qua việc sử dụng các chuẩn SSL/TLS, xác thực người dùng bằng username/password và có thể mã hóa dữ liệu truyền tải nhằm đảm bảo an toàn thông tin. Nhờ cấu trúc nhẹ, việc triển khai giao thức này trên các hệ thống nhúng với vi điều khiển công suất thấp như ESP32, ESP8266, STM32 và Arduino trở nên rất thuận tiện và hiệu quả. Bên cạnh đó, việc MQTT hỗ trợ cơ chế kết nối liên tục hoặc kết nối ngắt quãng định kỳ (keep-alive), giúp tối ưu đáng kể thời lượng pin và tài nguyên mạng trong các ứng dụng IoT di động hoặc từ xa.

Với các đặc điểm vượt trội kể trên, MQTT đã trở thành lựa chọn phổ biến cho các ứng dụng IoT trong nhiều lĩnh vực, từ nông nghiệp thông minh, tự động hóa công nghiệp, quản lý và giám sát năng lượng, cho tới các ứng dụng thiết bị đeo tay thông minh và y tế cá nhân hiện đại. Giao thức này giúp giảm thiểu phức tạp trong việc truyền tải dữ liệu cảm biến, đồng thời tối ưu hóa độ tin cậy, bảo mật và khả năng mở rộng, đáp ứng tốt các yêu cầu khắt khe của hệ thống IoT trong thực tế.

### 2.3.5. Giao thức không dây BLE

Bluetooth Low Energy (BLE), còn được gọi là Bluetooth Smart, là một công nghệ giao tiếp không dây tiết kiệm năng lượng, được phát triển nhằm phục vụ cho các ứng dụng đòi hỏi thời lượng pin dài và tốc độ truyền dữ liệu thấp hoặc trung bình. Chuẩn giao tiếp BLE được giới thiệu lần đầu tiên bởi tổ chức Bluetooth SIG (Bluetooth Special Interest Group) vào năm 2010 như một phần của chuẩn Bluetooth 4.0. Với đặc tính nổi bật là khả năng tiết kiệm năng lượng vượt trội và tương thích rộng rãi với hầu hết các thiết bị di động hiện đại như smartphone, máy tính bảng và máy tính cá nhân, BLE đã nhanh chóng trở thành một trong những chuẩn giao tiếp phổ biến nhất trong các ứng dụng Internet of Things (IoT), thiết bị đeo tay thông minh, các hệ thống theo dõi sức khỏe cá nhân và thiết bị nhà thông minh.



* + - * 1. Mô hình BLE

Cấu trúc hoạt động của BLE dựa trên mô hình giao tiếp không đối xứng giữa hai loại thiết bị chính: **Central** và **Peripheral**. Trong đó, thiết bị Central thường là các thiết bị chủ động như điện thoại thông minh hoặc máy tính, chịu trách nhiệm quét và thiết lập kết nối tới các thiết bị Peripheral. Ngược lại, thiết bị Peripheral là các cảm biến, thiết bị đeo tay hay các thiết bị đầu cuối, thường hoạt động ở chế độ tiêu thụ năng lượng thấp, phát các gói tin quảng bá (advertising packets) để các thiết bị Central có thể phát hiện và kết nối. Khi kết nối được thiết lập, BLE cho phép hai thiết bị trao đổi dữ liệu thông qua một cấu trúc dữ liệu đặc biệt gồm các dịch vụ (Services) và các đặc tính (Characteristics), mỗi dịch vụ và đặc tính được nhận dạng thông qua các định danh UUID (Universally Unique Identifier).

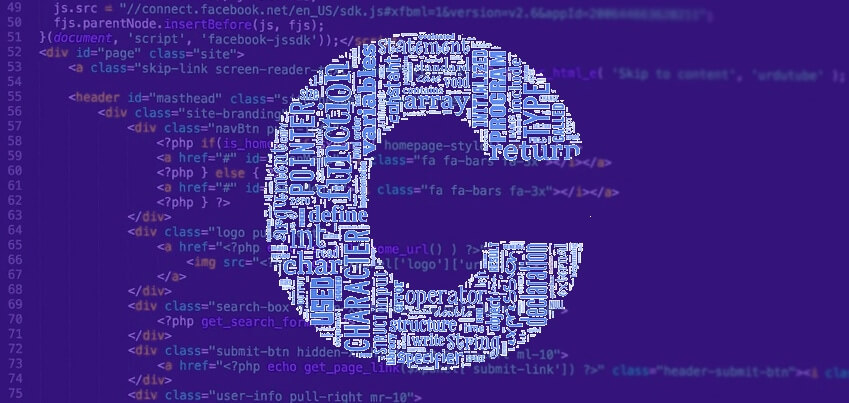
BLE có một cơ chế đặc biệt giúp tiết kiệm điện năng đáng kể so với các chuẩn không dây khác. Điều này được thực hiện thông qua việc quản lý chặt chẽ chu kỳ kết nối, giảm thời gian phát sóng radio và tối ưu các gói dữ liệu truyền tải có kích thước nhỏ gọn (thường không vượt quá vài chục byte mỗi lần). Chính vì vậy, BLE có thể hoạt động hiệu quả ngay cả khi sử dụng các nguồn năng lượng rất nhỏ như pin đồng xu (coin-cell battery). Một ưu điểm nổi bật nữa là khả năng hỗ trợ nhiều chế độ phát quảng bá như Advertising mode, Scanning mode, và Connected mode. Chế độ quảng bá cho phép các thiết bị liên tục phát tín hiệu để dễ dàng được nhận biết, trong khi Connected mode cho phép giao tiếp hai chiều ổn định, tin cậy và bảo mật giữa thiết bị Peripheral và thiết bị Central.

BLE cung cấp ba mức bảo mật khác nhau dựa trên việc sử dụng mã hóa AES-CCM 128-bit, xác thực, ghép đôi (pairing), và bonding nhằm đảm bảo an toàn dữ liệu truyền tải giữa các thiết bị. Các thiết bị BLE có thể hoạt động trên băng tần 2,4 GHz ISM (Industrial, Scientific, Medical) với tổng cộng 40 kênh, trong đó 3 kênh dành riêng cho quảng bá và 37 kênh còn lại sử dụng cho các phiên kết nối thực sự. Việc sử dụng thuật toán nhảy tần thích nghi (Adaptive Frequency Hopping - AFH) giúp giảm thiểu tối đa các vấn đề nhiễu sóng và xung đột với các thiết bị khác trên cùng băng tần, tăng cường sự ổn định và tin cậy của kết nối.

Với cấu trúc nhẹ và khả năng tiêu thụ năng lượng thấp vượt trội, giao thức BLE rất phù hợp cho các hệ thống cảm biến thông minh, thiết bị y tế cá nhân như đo nhịp tim, SpO₂, theo dõi chuyển động và giấc ngủ; ngoài ra còn ứng dụng rộng rãi trong các thiết bị nhà thông minh, beacon định vị trong nhà, và nhiều hệ thống IoT công nghiệp khác. Hiện nay, BLE đã được tích hợp rộng rãi vào các nền tảng vi điều khiển phổ biến như ESP32, STM32, Nordic nRF, cũng như có sẵn các thư viện hỗ trợ cho các nền tảng phần mềm như Android, iOS, và Windows, giúp cho việc triển khai và phát triển ứng dụng BLE trở nên dễ dàng và thuận tiện hơn bao giờ hết.

## 2.4. Công nghệ phần mềm

### 2.4.1. Ngôn ngữ lập trình C



* + - * 1. Ngôn ngữ lập trình C.

Ngôn ngữ lập trình C là một ngôn ngữ lập trình cấp trung, được phát triển vào năm 1972 bởi Dennis Ritchie tại Bell Labs với mục đích ban đầu là viết hệ điều hành UNIX. Ngôn ngữ C kết hợp những đặc điểm của cả ngôn ngữ cấp thấp và cấp cao, giúp lập trình viên vừa có thể kiểm soát phần cứng hiệu quả, vừa dễ dàng phát triển các ứng dụng phức tạp.

C là một ngôn ngữ có hiệu suất cao nhờ khả năng tối ưu hóa tài nguyên hệ thống và quản lý bộ nhớ trực tiếp. Điều này khiến nó trở nên lý tưởng cho các ứng dụng yêu cầu tốc độ và hiệu quả, chẳng hạn như hệ điều hành, trình điều khiển thiết bị, phần mềm nhúng và các ứng dụng thời gian thực. Khả năng điều khiển bộ nhớ bằng con trỏ và cấp phát động trong C cho phép lập trình viên tự quản lý việc sử dụng bộ nhớ, điều mà không nhiều ngôn ngữ lập trình khác cung cấp.

Một trong những điểm mạnh của C là tính di động cao, nghĩa là mã nguồn viết bằng C có thể biên dịch và chạy trên nhiều nền tảng khác nhau mà không cần chỉnh sửa nhiều. Điều này làm cho C trở thành ngôn ngữ phổ biến trong phát triển phần mềm hệ thống, các ứng dụng nhúng, và thậm chí là phát triển các ngôn ngữ lập trình khác.

Ngôn ngữ C có ảnh hưởng lớn đến nhiều ngôn ngữ lập trình hiện đại như C++, Java, C#, và Python. Các hệ điều hành lớn như UNIX, Linux, và Windows đều được phát triển phần lớn bằng C, và nó vẫn là lựa chọn hàng đầu trong lập trình hệ thống nhúng cũng như các ứng dụng yêu cầu tốc độ và khả năng tối ưu hóa cao.

Với cú pháp rõ ràng, mạnh mẽ và khả năng kiểm soát chi tiết hệ thống, ngôn ngữ lập trình C đã trở thành nền tảng của nhiều hệ thống công nghệ quan trọng và vẫn được sử dụng rộng rãi cho đến ngày nay.

### 2.4.2. ESP-IDF Framework



* + - * 1. Logo ESP-IDF Framework

ESP-IDF (**Espressif IoT Development Framework**) là một bộ khung phát triển phần mềm chính thức được phát triển bởi Espressif Systems dành riêng cho dòng chip ESP32 và ESP32-S series, hỗ trợ đầy đủ các chức năng cần thiết cho các ứng dụng IoT hiện đại. ESP-IDF cung cấp một môi trường lập trình đầy đủ, tích hợp sẵn hệ điều hành thời gian thực FreeRTOS, thư viện giao tiếp ngoại vi, quản lý bộ nhớ linh hoạt, hỗ trợ mạnh mẽ các kết nối không dây như Wi-Fi, Bluetooth Classic và Bluetooth Low Energy (BLE), cùng khả năng bảo mật nâng cao. Tất cả các tính năng này giúp nhà phát triển có thể nhanh chóng xây dựng, thử nghiệm và triển khai các ứng dụng nhúng một cách chuyên nghiệp và đáng tin cậy.

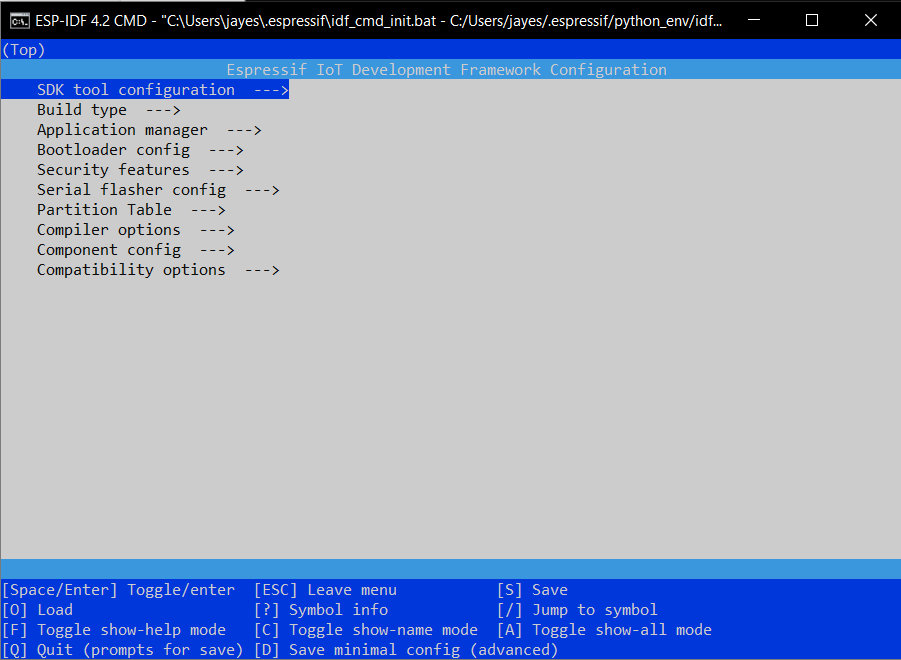
ESP-IDF được xây dựng dựa trên ngôn ngữ lập trình C và C++, đảm bảo hiệu suất hoạt động cao, thời gian đáp ứng nhanh và khả năng tùy biến mạnh mẽ. Framework này cung cấp một kiến trúc module rất rõ ràng và trực quan, giúp người dùng dễ dàng quản lý các chức năng riêng biệt như giao tiếp ngoại vi UART, I²C, SPI; giao thức mạng TCP/IP, MQTT, HTTP; hệ thống tập tin ảo SPIFFS, FATFS và đặc biệt là khả năng cập nhật phần mềm OTA (Over-The-Air) một cách dễ dàng. Bên cạnh đó, ESP-IDF còn tích hợp các công cụ mạnh mẽ hỗ trợ debug và profiling hiệu suất như ESP32 OpenOCD và ESP SystemView, giúp quá trình phát triển và kiểm thử ứng dụng trở nên thuận lợi và chính xác hơn.

Một trong những ưu điểm nổi bật nhất của ESP-IDF là khả năng quản lý và phân chia tài nguyên hệ thống một cách linh hoạt và hiệu quả nhờ vào FreeRTOS được tích hợp sẵn. FreeRTOS cho phép tạo nhiều tác vụ song song (multi-tasking), đồng bộ hóa các tác vụ bằng semaphore, mutex và event groups, quản lý bộ nhớ heap một cách linh hoạt, giúp tối ưu hóa hiệu năng và độ ổn định trong các ứng dụng đòi hỏi tính thời gian thực nghiêm ngặt. Bên cạnh đó, ESP-IDF hỗ trợ khả năng quản lý năng lượng thông minh, cho phép chip ESP32 chuyển đổi linh hoạt giữa nhiều chế độ vận hành như active mode, modem-sleep mode, light-sleep mode và deep-sleep mode, giúp tối ưu đáng kể điện năng tiêu thụ trong các ứng dụng chạy pin dài ngày.

Khả năng bảo mật cũng là một ưu điểm quan trọng của ESP-IDF. Framework này tích hợp sẵn các công nghệ bảo mật hàng đầu như mã hóa TLS/SSL để bảo vệ dữ liệu khi truyền tải qua mạng, mã hóa flash AES-256 để bảo vệ firmware và dữ liệu lưu trữ cục bộ, hỗ trợ các cơ chế xác thực dựa trên chứng chỉ số X.509, JWT (JSON Web Token), và Secure Boot để đảm bảo tính toàn vẹn và xác thực của firmware trong quá trình khởi động. Những tính năng bảo mật nâng cao này giúp ESP-IDF trở thành lựa chọn tối ưu cho các ứng dụng IoT yêu cầu bảo mật dữ liệu cao như hệ thống giám sát sức khỏe, nhà thông minh, hay các ứng dụng công nghiệp nhạy cảm.

Ngoài ra, Espressif Systems còn cung cấp một cộng đồng phát triển đông đảo và hỗ trợ liên tục cập nhật, sửa lỗi và bổ sung các tính năng mới cho ESP-IDF, giúp người dùng luôn được tiếp cận những công nghệ mới nhất và có sự hỗ trợ kịp thời từ nhà sản xuất. ESP-IDF cũng tích hợp sâu với các công cụ phổ biến như Eclipse, VS Code, PlatformIO và Espressif IDE, tạo điều kiện thuận lợi để lập trình viên dễ dàng phát triển, quản lý và triển khai dự án trong nhiều môi trường khác nhau.

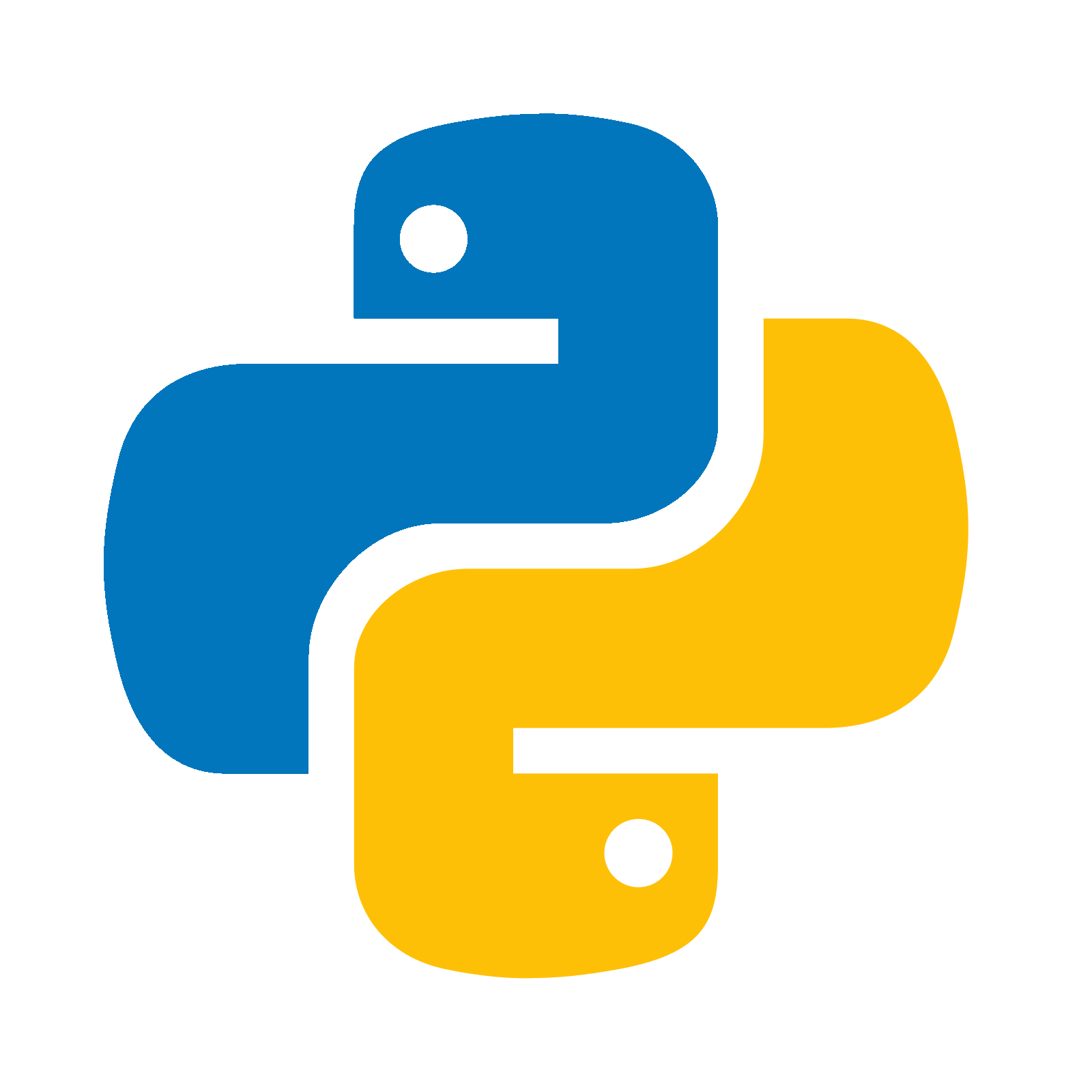
Nhìn chung, với những ưu điểm nổi bật như cấu trúc module rõ ràng, hỗ trợ FreeRTOS mạnh mẽ, khả năng bảo mật cao, quản lý năng lượng hiệu quả và cộng đồng hỗ trợ đông đảo, ESP-IDF thực sự là lựa chọn lý tưởng và chuyên nghiệp cho các nhà phát triển muốn xây dựng các hệ thống IoT hiện đại, ổn định và bảo mật dựa trên nền tảng vi điều khiển ESP32.



* + - * 1. Giao diện menuconfig của ESP-IDF.

### 2.4.3. Ngôn ngữ lập trình Python

Python là một ngôn ngữ lập trình bậc cao, đa mục đích, được Guido van Rossum giới thiệu lần đầu vào năm 1991 với tiêu chí cốt lõi là đơn giản hóa cú pháp và tăng cường tính dễ đọc cho mã nguồn. Với đặc trưng là ngôn ngữ lập trình thông dịch, Python giúp người lập trình nhanh chóng kiểm thử các ý tưởng và phát triển các ứng dụng phần mềm một cách hiệu quả và trực quan. Hiện nay, Python trở thành một trong những ngôn ngữ phổ biến nhất trên thế giới, được sử dụng rộng rãi trong nhiều lĩnh vực khác nhau, từ các hệ thống nhúng IoT, trí tuệ nhân tạo, phân tích dữ liệu, tự động hóa quy trình đến các ứng dụng web và desktop.



* + - * 1. Logo ngôn ngữ lập trình Python

Một trong những ưu điểm lớn nhất của Python là cú pháp ngắn gọn, dễ hiểu, gần với ngôn ngữ tự nhiên, nhờ vậy Python giảm thiểu đáng kể thời gian cần thiết để học tập và làm quen, ngay cả đối với những lập trình viên mới bắt đầu. Việc quản lý bộ nhớ tự động cùng khả năng xử lý linh hoạt các kiểu dữ liệu động cho phép người lập trình giảm đáng kể thời gian viết mã, hạn chế lỗi xảy ra trong quá trình phát triển. Bên cạnh đó, Python sở hữu một hệ sinh thái thư viện cực kỳ đa dạng và mạnh mẽ, như NumPy, Pandas, Matplotlib trong xử lý và phân tích dữ liệu, TensorFlow, PyTorch trong lĩnh vực trí tuệ nhân tạo, Django và Flask trong phát triển web, giúp người dùng dễ dàng mở rộng và áp dụng vào đa dạng các dự án và lĩnh vực chuyên biệt.

Ngoài ra, Python đặc biệt phù hợp với các hệ thống IoT và ứng dụng nhúng nhờ khả năng dễ dàng kết nối và giao tiếp với phần cứng thông qua các thư viện có sẵn. Ví dụ như thư viện PySerial cho phép Python tương tác với các thiết bị ngoại vi thông qua giao tiếp UART, hoặc thư viện Paho MQTT cung cấp khả năng tích hợp nhanh chóng với các hệ thống sử dụng giao thức MQTT. Với các bo mạch nhúng thông dụng như Raspberry Pi hay ESP32, Python giúp đơn giản hóa việc lập trình, giảm thời gian phát triển, đồng thời tối ưu hóa hiệu quả sử dụng tài nguyên của hệ thống nhúng.

Một ưu điểm quan trọng nữa của Python chính là khả năng tương thích đa nền tảng. Các ứng dụng viết bằng Python có thể hoạt động một cách ổn định và nhất quán trên nhiều nền tảng hệ điều hành khác nhau như Windows, Linux, macOS và thậm chí trên các nền tảng nhúng. Python còn được hỗ trợ mạnh mẽ bởi cộng đồng phát triển vô cùng đông đảo và năng động trên toàn thế giới, điều này giúp các nhà phát triển nhanh chóng tiếp cận được những giải pháp mới nhất, nhận hỗ trợ nhanh chóng khi gặp vấn đề và cập nhật liên tục các phiên bản nâng cấp cũng như bản vá bảo mật.

Như vậy, với cú pháp đơn giản, dễ hiểu, hệ thống thư viện đồ sộ, khả năng tương thích đa nền tảng và sự hỗ trợ rộng rãi từ cộng đồng, Python là lựa chọn tối ưu không chỉ trong việc phát triển nhanh các ứng dụng phần mềm mà còn đặc biệt hữu ích trong việc xây dựng và triển khai các hệ thống IoT và ứng dụng nhúng hiện đại, giúp tiết kiệm đáng kể thời gian phát triển, tối ưu hóa hiệu suất hoạt động và nâng cao chất lượng sản phẩm cuối cùng.

Dưới đây là các ưu điểm nổi bật của ngôn ngữ lập trình Python:

* Cú pháp đơn giản, dễ hiểu: Python có cú pháp rõ ràng, ngắn gọn, gần với ngôn ngữ tự nhiên, giúp người mới học dễ dàng tiếp cận và người phát triển dễ dàng bảo trì, nâng cấp mã nguồn.
* Ngôn ngữ thông dịch: Python cho phép thực thi từng dòng mã trực tiếp, giúp lập trình viên nhanh chóng kiểm thử, sửa lỗi và triển khai các tính năng mới một cách hiệu quả và nhanh chóng.
* Hệ sinh thái thư viện mạnh mẽ và đa dạng: Python sở hữu rất nhiều thư viện mã nguồn mở chất lượng cao, hỗ trợ đa dạng các lĩnh vực như trí tuệ nhân tạo (TensorFlow, PyTorch), xử lý dữ liệu (Pandas, NumPy), trực quan hóa dữ liệu (Matplotlib, Seaborn), lập trình web (Django, Flask), và lập trình hệ thống nhúng (PySerial, MicroPython).
* Khả năng tương thích đa nền tảng: Các chương trình viết bằng Python có thể dễ dàng chạy trên nhiều hệ điều hành khác nhau như Windows, Linux, macOS, Raspberry Pi OS, giúp tăng cường khả năng di động và dễ dàng triển khai trên nhiều nền tảng khác nhau.
* Quản lý bộ nhớ tự động: Python hỗ trợ cơ chế quản lý bộ nhớ tự động (garbage collection), giúp lập trình viên tránh các lỗi liên quan đến việc cấp phát và giải phóng bộ nhớ, từ đó giảm thiểu thời gian debug và nâng cao chất lượng phần mềm.
* Khả năng mở rộng và tích hợp linh hoạt: Python cho phép dễ dàng tích hợp với các ngôn ngữ lập trình khác như C, C++, Java thông qua các thư viện mở rộng như ctypes, Cython, JNI. Điều này giúp tăng hiệu suất xử lý cho các tác vụ đòi hỏi tốc độ tính toán cao mà Python thông thường khó đáp ứng.
* Cộng đồng phát triển lớn và tích cực: Python được hỗ trợ bởi một cộng đồng phát triển đông đảo trên toàn thế giới. Điều này mang lại nhiều lợi ích như sự hỗ trợ nhanh chóng khi gặp lỗi, các tài liệu hướng dẫn phong phú, cũng như các bản cập nhật thường xuyên về tính năng và bảo mật.
* Ứng dụng rộng rãi trong lĩnh vực IoT và hệ thống nhúng: Python có khả năng giao tiếp hiệu quả với phần cứng nhúng thông qua các thư viện chuyên dụng như PySerial, GPIO, và MicroPython. Đặc biệt phù hợp khi lập trình cho các bo mạch phổ biến như Raspberry Pi, ESP32.
* Hiệu quả cho các ứng dụng trí tuệ nhân tạo và khoa học dữ liệu: Python là ngôn ngữ được sử dụng nhiều nhất trong lĩnh vực AI và khoa học dữ liệu nhờ vào các thư viện chuyên biệt, khả năng xử lý dữ liệu lớn nhanh chóng, và cộng đồng chuyên gia đông đảo hỗ trợ phát triển các thuật toán tiên tiến.

Với những ưu điểm kể trên, Python xứng đáng là một lựa chọn hàng đầu cho việc phát triển các ứng dụng IoT, nhúng, trí tuệ nhân tạo và phần mềm ứng dụng tổng quát.

### 2.4.4. Firebase

#### 2.4.4.1. Giới thiệu



* + - * 1. Firebase

Firebase là nền tảng dịch vụ điện toán đám mây toàn diện, cung cấp giải pháp Backend-as-a-Service (BaaS), được phát triển bởi Firebase Inc. vào năm 2011, sau đó được Google mua lại vào năm 2014. Firebase ra đời nhằm hỗ trợ việc xây dựng nhanh chóng, triển khai dễ dàng và quản lý hiệu quả các ứng dụng web, di động, cũng như các thiết bị IoT. Thay vì phải tự quản lý toàn bộ cơ sở hạ tầng backend phức tạp, Firebase cho phép nhà phát triển tập trung vào phần front-end và trải nghiệm người dùng, giảm đáng kể chi phí phát triển, thời gian triển khai và các khó khăn trong vận hành.

Firebase cung cấp một hệ thống dịch vụ đồng bộ, tích hợp sâu rộng với các nền tảng phổ biến như Android, iOS, Web và thậm chí các ứng dụng nhúng. Nó tận dụng các API và SDK đơn giản nhưng mạnh mẽ, giúp người phát triển dễ dàng tích hợp các chức năng nâng cao như xác thực người dùng, phân tích dữ liệu, gửi thông báo push, và đồng bộ hóa dữ liệu thời gian thực mà không mất nhiều công sức lập trình. Điều này đặc biệt hữu ích cho các nhóm phát triển nhỏ hoặc các dự án startup với nguồn lực giới hạn.

Ngoài ra, Firebase sở hữu một cộng đồng hỗ trợ đông đảo và tích cực, với nhiều tài liệu hướng dẫn chi tiết, dễ hiểu cùng các diễn đàn thảo luận sôi nổi. Google cũng liên tục cập nhật và bổ sung thêm các tính năng mới vào Firebase nhằm đáp ứng tốt hơn nhu cầu ngày càng cao của các nhà phát triển và các ứng dụng đa nền tảng hiện đại. Vì thế, Firebase nhanh chóng trở thành một lựa chọn ưu tiên cho cả các ứng dụng nhỏ, đơn giản lẫn các hệ thống lớn, phức tạp cần quản lý hiệu quả trên quy mô toàn cầu.

#### 2.4.4.2. Các tính năng quan trọng

* Firebase nổi bật với hệ thống tính năng phong phú, toàn diện, và có khả năng đáp ứng hầu hết các nhu cầu phát triển ứng dụng hiện đại. Một số tính năng quan trọng và đặc biệt hữu ích bao gồm:
* Realtime Database: Lưu trữ và đồng bộ dữ liệu tức thời giữa các client kết nối thông qua WebSocket.
* Cloud Firestore: Cơ sở dữ liệu dạng NoSQL dựa trên tài liệu (document-based), hỗ trợ khả năng truy vấn phức tạp hơn và khả năng mở rộng cao hơn Realtime Database.
* Firebase Authentication: Cung cấp giải pháp xác thực linh hoạt, đa dạng phương thức như email/password, số điện thoại, Google, Facebook, Twitter, GitHub, giúp người dùng dễ dàng đăng nhập ứng dụng.
* Cloud Storage: Cho phép lưu trữ dữ liệu file lớn một cách an toàn, đáng tin cậy, hỗ trợ upload, download nhanh chóng và phân phối nội dung đa khu vực.
* Firebase Hosting: Cung cấp hosting nhanh, bảo mật, hỗ trợ CDN tích hợp sẵn để phục vụ nội dung web tĩnh và động với tốc độ truy cập cao.
* Cloud Functions: Viết mã backend dưới dạng hàm đơn giản chạy trực tiếp trên hạ tầng Google Cloud, được kích hoạt dựa vào sự kiện (event-driven) để tự động thực hiện các tác vụ backend mà không cần cấu hình máy chủ riêng.
* Firebase Analytics (Google Analytics for Firebase): Công cụ phân tích người dùng mạnh mẽ, cung cấp các báo cáo chuyên sâu về hành vi và tương tác người dùng với ứng dụng, giúp tối ưu hóa trải nghiệm và hiệu quả hoạt động.
* Firebase Cloud Messaging (FCM): Gửi thông báo push hiệu quả, hỗ trợ đầy đủ trên Android, iOS, Web, giúp giao tiếp với người dùng nhanh chóng và hiệu quả hơn.
* Firebase Remote Config: Cho phép điều chỉnh tham số ứng dụng từ xa, giúp thử nghiệm A/B, cá nhân hóa trải nghiệm người dùng, mà không cần cập nhật ứng dụng mới.
* Crashlytics và Performance Monitoring: Giúp giám sát lỗi ứng dụng và hiệu suất thực tế, phát hiện và khắc phục vấn đề nhanh chóng, duy trì chất lượng trải nghiệm người dùng ở mức cao nhất.

#### 2.4.4.3. Realtime Database

Realtime Database là dịch vụ nổi bật nhất của Firebase, đóng vai trò cơ sở dữ liệu NoSQL thời gian thực, sử dụng JSON làm định dạng lưu trữ chính, giúp đồng bộ nhanh chóng và hiệu quả giữa các client. Khác với các cơ sở dữ liệu truyền thống thường yêu cầu client chủ động truy vấn theo chu kỳ để cập nhật dữ liệu, Firebase Realtime Database cho phép tự động đồng bộ dữ liệu ngay lập tức giữa tất cả các client theo cơ chế event-driven, giảm thiểu độ trễ và nâng cao khả năng phản hồi của ứng dụng.



* + - * 1. Firebase Realtime Database

Dữ liệu được lưu trữ theo cấu trúc cây JSON, cung cấp một cách lưu trữ dữ liệu linh hoạt, rõ ràng, dễ dàng quản lý, đặc biệt phù hợp với các dữ liệu có cấu trúc phân cấp. Firebase cũng cung cấp hệ thống Rule-based Security linh hoạt, cho phép định nghĩa quyền truy cập một cách chi tiết, đảm bảo dữ liệu luôn an toàn và chỉ được truy cập bởi những client được cho phép.

Realtime Database đặc biệt thích hợp với các ứng dụng cần đồng bộ dữ liệu liên tục, cập nhật tức thời như các ứng dụng chat, mạng xã hội, game thời gian thực, và hệ thống giám sát IoT. Ngoài ra, khả năng hoạt động ngoại tuyến (offline mode) là một ưu điểm quan trọng, giúp người dùng tiếp tục sử dụng ứng dụng mà không gặp gián đoạn, ngay cả khi mất kết nối internet, với dữ liệu sẽ được đồng bộ trở lại khi kết nối được tái thiết lập.

#### 2.4.4.4. Cách thức hoạt động của Realtime Database

Firebase Realtime Database hoạt động trên mô hình client-server với cơ chế đồng bộ dữ liệu theo thời gian thực dựa trên các kết nối WebSocket. Khi một client kết nối vào Firebase, nó mở ra một đường truyền liên tục và trực tiếp đến server Firebase, giúp quá trình đồng bộ dữ liệu diễn ra ngay lập tức khi có sự thay đổi:

* Client thực hiện ghi (write) hoặc cập nhật (update) dữ liệu vào một node cụ thể trong cơ sở dữ liệu.
* Firebase server nhận thông tin cập nhật này và ngay lập tức ghi lại trạng thái mới của dữ liệu vào bộ nhớ của mình.
* Sau đó, Firebase server lập tức thông báo qua kết nối WebSocket đến tất cả các client khác đang “lắng nghe” (subscribe) vào node dữ liệu tương ứng.
* Các client khi nhận được thông báo sẽ tự động tải xuống các thay đổi này và cập nhật trạng thái ứng dụng mà không cần gửi yêu cầu truy vấn dữ liệu mới.
* Toàn bộ quá trình này diễn ra hoàn toàn tự động, tức thời, liên tục và không cần client phải liên tục gửi truy vấn, giảm thiểu đáng kể băng thông mạng và độ trễ.

Ngoài ra, Realtime Database có khả năng lưu trữ dữ liệu cục bộ trên client, giúp ứng dụng vẫn tiếp tục hoạt động ngay cả khi offline. Mọi thay đổi khi offline sẽ được ghi lại vào cache nội bộ và tự động đồng bộ lên server Firebase ngay khi kết nối internet được khôi phục.

Cơ chế đồng bộ thời gian thực này là nền tảng cho rất nhiều ứng dụng hiện đại, từ ứng dụng cộng tác làm việc nhóm, ứng dụng giám sát hệ thống IoT theo thời gian thực, ứng dụng chat tức thì, cho tới các hệ thống theo dõi và cập nhật trạng thái liên tục như các nền tảng thương mại điện tử, quản lý chuỗi cung ứng hay ứng dụng y tế cá nhân.



* + - * 1. Mô hình Firebase Realtime Database

### 2.4.5. Ngôn ngữ lập trình Dart và Framework Flutter

#### 2.4.5.1. Giới thiệu chung

**Ngôn ngữ lập trình Dart** là một ngôn ngữ lập trình hướng đối tượng, đa nền tảng, được thiết kế và phát triển bởi Google, chính thức công bố lần đầu tiên vào năm 2011. Ban đầu Dart được tạo ra nhằm giải quyết các hạn chế hiệu năng của JavaScript, đặc biệt là trong các ứng dụng web lớn. Tuy nhiên, sau khi được lựa chọn làm ngôn ngữ nền tảng cho framework Flutter, Dart ngày càng trở nên phổ biến và được sử dụng rộng rãi hơn trong việc phát triển ứng dụng di động và đa nền tảng.



* + - * 1. Ngôn ngữ lập trình Dart

Dart sở hữu cú pháp rõ ràng, dễ đọc, dễ hiểu và gần với các ngôn ngữ lập trình phổ biến như Java, JavaScript và C#. Điều này giúp lập trình viên nhanh chóng tiếp cận và giảm đáng kể thời gian học tập. Dart hỗ trợ biên dịch hai chế độ: biên dịch tức thời (Just-in-Time - JIT) giúp nhà phát triển nhanh chóng thử nghiệm ứng dụng ngay trong quá trình lập trình, và biên dịch trước (Ahead-of-Time - AOT) để tối ưu hóa hiệu suất thực thi trên thiết bị di động và các nền tảng khác. Dart còn cung cấp tính năng quản lý bộ nhớ tự động thông qua cơ chế Garbage Collection, giúp lập trình viên giảm thiểu các vấn đề về quản lý bộ nhớ và hạn chế lỗi runtime.

**Flutter** là một framework phát triển ứng dụng đa nền tảng, được xây dựng trên nền ngôn ngữ Dart, do Google phát hành vào năm 2017. Flutter hỗ trợ lập trình viên tạo ra các ứng dụng chất lượng cao trên nhiều nền tảng khác nhau như Android, iOS, Web, Windows, macOS và Linux chỉ từ một bộ mã nguồn duy nhất. Điều này không chỉ tiết kiệm thời gian, chi phí phát triển, mà còn tối ưu hóa hiệu quả quá trình bảo trì và cập nhật ứng dụng.



* + - * 1. Khung Flutter

Flutter sử dụng bộ engine đồ họa hiệu năng cao gọi là Skia để vẽ và hiển thị giao diện người dùng, hoàn toàn độc lập với các thành phần giao diện gốc của hệ điều hành. Chính vì vậy, Flutter mang lại hiệu suất đồ họa mượt mà, đồng nhất trên mọi thiết bị và nền tảng, loại bỏ được các rào cản về hiệu năng khi so sánh với các framework đa nền tảng khác. Ngoài ra, Flutter cung cấp một bộ thư viện widget đồ họa phong phú, hỗ trợ nhiều loại giao diện từ cơ bản đến phức tạp, dễ dàng tùy chỉnh và đáp ứng mọi nhu cầu thiết kế ứng dụng đa dạng.

2.4.5.2. Các đặc điểm nổi bật của ngôn ngữ Dart

Ngôn ngữ Dart có nhiều đặc điểm nổi bật giúp lập trình viên nhanh chóng xây dựng các ứng dụng chất lượng cao và dễ bảo trì, cụ thể như sau:

* **Cú pháp trực quan và dễ tiếp cận:** Dart cung cấp cú pháp rõ ràng, cấu trúc logic chặt chẽ, giúp lập trình viên dễ dàng hiểu và quản lý mã nguồn ngay cả với các ứng dụng có độ phức tạp cao.
* **Hỗ trợ lập trình hướng đối tượng đầy đủ:** Dart tuân thủ đầy đủ các nguyên lý lập trình hướng đối tượng như đóng gói (encapsulation), kế thừa (inheritance), đa hình (polymorphism), và trừu tượng (abstraction), giúp người lập trình dễ dàng quản lý và mở rộng mã nguồn.
* **Cơ chế biên dịch linh hoạt (JIT và AOT):** Dart có khả năng biên dịch JIT trong quá trình phát triển, cho phép lập trình viên nhanh chóng kiểm tra và chỉnh sửa mã nguồn. Ngược lại, AOT cung cấp hiệu suất thực thi tối ưu, giảm thời gian khởi động ứng dụng và tăng tốc độ xử lý.
* **Xử lý bất đồng bộ hiệu quả:** Dart hỗ trợ mạnh mẽ việc xử lý các tác vụ bất đồng bộ thông qua cú pháp async, await, và kiểu dữ liệu Future và Stream, giúp ứng dụng phản hồi nhanh hơn và quản lý tài nguyên hiệu quả hơn khi xử lý dữ liệu mạng, truy cập cơ sở dữ liệu và các tác vụ đòi hỏi thời gian chờ khác.
* **An toàn về kiểu dữ liệu (Type Safety):** Dart hỗ trợ kiểm tra kiểu dữ liệu tĩnh (static typing) và cả kiểu động (dynamic typing), giúp giảm thiểu lỗi runtime và tăng độ ổn định cho ứng dụng.
* **Quản lý bộ nhớ tự động (Garbage Collection):** Dart tích hợp trình quản lý bộ nhớ tự động, giúp giảm thiểu gánh nặng cho lập trình viên, hạn chế các vấn đề rò rỉ bộ nhớ và lỗi liên quan đến việc quản lý thủ công bộ nhớ.

#### 2.4.5.3. Framework Flutter và các ưu điểm chính

Flutter là một framework phát triển ứng dụng đa nền tảng tiên tiến và mạnh mẽ, do Google phát hành lần đầu vào năm 2017. Flutter được thiết kế để đáp ứng nhu cầu xây dựng nhanh chóng các ứng dụng chất lượng cao trên nhiều nền tảng như Android, iOS, Web, Windows, Linux và macOS chỉ với một cơ sở mã nguồn duy nhất. Nhờ đó, Flutter nhanh chóng trở thành một trong những công cụ phổ biến nhất trong lĩnh vực phát triển ứng dụng hiện đại, với mức độ phổ biến và tốc độ tăng trưởng ngày càng mạnh mẽ.

Flutter sử dụng ngôn ngữ lập trình Dart làm ngôn ngữ lập trình chính, giúp tận dụng tối đa khả năng biên dịch linh hoạt (AOT và JIT) để tối ưu hiệu suất ứng dụng và trải nghiệm phát triển nhanh chóng. Điểm đặc biệt của Flutter là cơ chế hoạt động dựa trên một engine đồ họa độc lập (Skia Graphics Engine), cho phép các ứng dụng được tạo ra luôn giữ được hiệu suất cao, mượt mà, và nhất quán trên tất cả các thiết bị và nền tảng hỗ trợ.

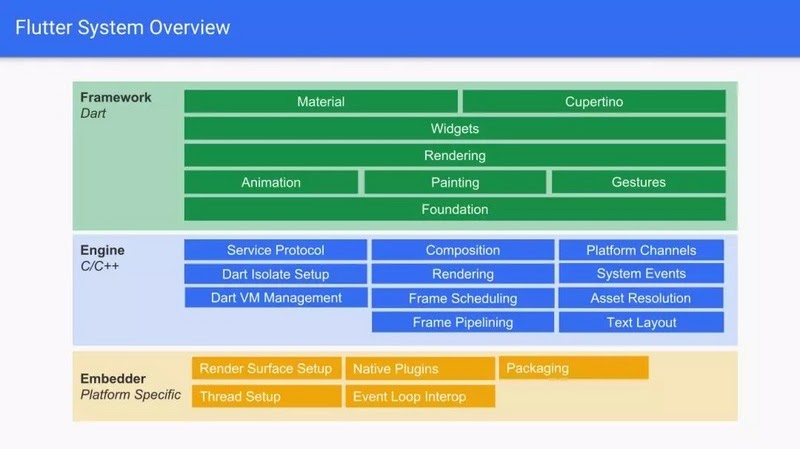
Dưới đây là những ưu điểm chính khiến Flutter trở nên đặc biệt hấp dẫn đối với cộng đồng phát triển:

* **Phát triển đa nền tảng hiệu quả:** Flutter cho phép các nhà phát triển viết một bộ mã nguồn duy nhất và triển khai đồng thời trên nhiều nền tảng khác nhau bao gồm Android, iOS, Web, Desktop (Windows, macOS, Linux), và các thiết bị nhúng. Điều này không chỉ giúp rút ngắn thời gian phát triển và thử nghiệm, mà còn giúp tối ưu hóa chi phí bảo trì và cập nhật ứng dụng. Flutter còn giúp đảm bảo tính nhất quán cao về giao diện và trải nghiệm người dùng giữa các nền tảng khác nhau.
* **Hiệu năng vượt trội nhờ kiến trúc độc đáo:** Khác với các framework đa nền tảng truyền thống dựa trên WebView hoặc JavaScript Bridge (ví dụ như React Native), Flutter sử dụng bộ engine đồ họa Skia được viết bằng C++ để vẽ trực tiếp giao diện lên màn hình thiết bị. Cách tiếp cận này giúp loại bỏ hoàn toàn các lớp trung gian và tối ưu hóa đáng kể hiệu suất đồ họa, giảm độ trễ, đảm bảo trải nghiệm người dùng mượt mà gần như ứng dụng native. Các animation, chuyển cảnh và tương tác phức tạp đều có thể được xử lý hiệu quả và nhanh chóng nhờ cơ chế này.
* **Hot Reload giúp phát triển nhanh chóng và trực quan:** Một trong những tính năng hấp dẫn nhất của Flutter là khả năng "Hot Reload", cho phép lập trình viên xem ngay lập tức kết quả thay đổi khi chỉnh sửa mã nguồn mà không cần khởi động lại ứng dụng. Hot Reload giúp rút ngắn đáng kể chu kỳ phát triển, tạo điều kiện thuận lợi cho các nhà thiết kế và lập trình viên thử nghiệm nhiều ý tưởng thiết kế, sửa lỗi nhanh chóng, và cải tiến ứng dụng liên tục trong thời gian ngắn nhất. Tính năng này đặc biệt hữu ích trong các dự án lớn, có yêu cầu cập nhật thường xuyên và cải tiến giao diện liên tục.
* **Thư viện widget đồ họa đa dạng, linh hoạt và mạnh mẽ:** Flutter cung cấp một bộ thư viện widget rất lớn và linh hoạt, được xây dựng dựa trên nguyên tắc "Everything is a widget" (mọi thứ đều là widget). Các widget của Flutter bao gồm từ các thành phần giao diện cơ bản như nút nhấn, văn bản, hình ảnh, đến các thành phần giao diện phức tạp như bảng điều khiển, thanh trượt, biểu đồ, hiệu ứng animation tùy chỉnh. Các widget này có khả năng tùy biến cao, dễ dàng sử dụng và có thể kết hợp để tạo ra các giao diện người dùng sáng tạo, đẹp mắt, trực quan, đáp ứng nhu cầu của cả người dùng lẫn nhà phát triển.
* **Cộng đồng phát triển đông đảo và hỗ trợ mạnh mẽ từ Google:** Flutter nhận được sự hậu thuẫn mạnh mẽ từ Google và sở hữu một cộng đồng phát triển rất đông đảo trên toàn thế giới. Google liên tục cập nhật, cải tiến các tính năng, sửa lỗi và tối ưu hiệu suất cho Flutter thông qua các bản cập nhật thường xuyên. Flutter còn có một hệ sinh thái rộng lớn với nhiều thư viện mã nguồn mở chất lượng cao, giúp dễ dàng tích hợp các dịch vụ bên ngoài như Firebase, RESTful API, SQLite, cũng như các công cụ phân tích và quản lý trạng thái phổ biến.
* **Khả năng mở rộng và tích hợp linh hoạt:** Flutter cho phép dễ dàng tích hợp các tính năng nền tảng đặc trưng thông qua các plugin native (Platform Channels). Điều này giúp các ứng dụng Flutter tận dụng tối đa sức mạnh từ các API nền tảng như camera, GPS, cảm biến, lưu trữ, notification, và các tính năng đặc biệt khác mà không gặp bất kỳ trở ngại đáng kể nào về hiệu suất hoặc trải nghiệm người dùng.
* **Quản lý trạng thái ứng dụng linh hoạt và hiệu quả:** Flutter cung cấp các phương pháp quản lý trạng thái ứng dụng hiện đại và hiệu quả, hỗ trợ nhiều mô hình quản lý trạng thái phổ biến như Provider, BLoC, Redux, MobX, GetX. Các mô hình này giúp quản lý trạng thái một cách rõ ràng, tách biệt logic nghiệp vụ khỏi giao diện người dùng, từ đó giúp ứng dụng dễ dàng mở rộng, bảo trì, và giảm thiểu các lỗi phát sinh trong quá trình phát triển.
* **Công cụ kiểm thử (Testing) toàn diện và mạnh mẽ:** Flutter cung cấp đầy đủ các công cụ và thư viện hỗ trợ kiểm thử tự động toàn diện từ unit test, widget test đến integration test. Điều này giúp các nhà phát triển dễ dàng đảm bảo chất lượng ứng dụng, phát hiện lỗi sớm và giảm đáng kể rủi ro trong quá trình triển khai sản phẩm thực tế.

Như vậy, với những ưu điểm nổi bật kể trên, Flutter trở thành một trong những lựa chọn hàng đầu trong phát triển ứng dụng đa nền tảng hiện đại. Framework này không chỉ giúp các nhà phát triển xây dựng nhanh chóng các ứng dụng đẹp, hiệu suất cao, mà còn tối ưu hóa mạnh mẽ hiệu quả kinh tế và thời gian phát triển, giúp các sản phẩm công nghệ có thể nhanh chóng tiếp cận người dùng một cách hiệu quả nhất.

#### 2.4.5.4. Kiến trúc và nguyên lý hoạt động của Flutter

Flutter là một framework đa nền tảng, có kiến trúc phân tầng rõ ràng, bao gồm ba lớp chính: **Framework Layer**, **Engine Layer**, và **Embedder Layer**, mỗi lớp đảm nhận một chức năng chuyên biệt nhằm tối ưu hóa hiệu suất, khả năng mở rộng và tương thích đa nền tảng.



* + - * 1. Kiến trúc của Flutter

**a) Framework Layer (Lớp ứng dụng - Dart)**

Đây là lớp trên cùng của Flutter, nơi mà lập trình viên trực tiếp tương tác để xây dựng giao diện và logic ứng dụng. Flutter Framework được viết hoàn toàn bằng ngôn ngữ Dart, cung cấp các công cụ mạnh mẽ và linh hoạt để thiết kế giao diện người dùng (UI). Framework này bao gồm các thư viện widget phong phú và đa dạng, giúp việc xây dựng các giao diện người dùng trực quan, đẹp mắt và dễ dàng tùy biến theo yêu cầu.

Các widget trong Flutter được tổ chức thành các cây widget (widget tree), mỗi widget đóng vai trò như một thành phần cấu tạo nên giao diện ứng dụng. Các widget này có thể là StatelessWidget (không thay đổi trạng thái) hoặc StatefulWidget (thay đổi trạng thái), cho phép quản lý hiệu quả trạng thái ứng dụng khi tương tác với người dùng.

Flutter Framework cũng bao gồm các hệ thống quản lý trạng thái mạnh mẽ như Provider, BLoC, Redux, GetX, giúp ứng dụng phản hồi tốt với sự thay đổi dữ liệu. Ngoài ra, Framework Layer cũng quản lý các vấn đề bất đồng bộ, xử lý sự kiện và định tuyến (routing) giữa các màn hình trong ứng dụng thông qua Navigator.

**b) Engine Layer (Lớp xử lý đồ họa và lõi - C++)**

Bên dưới Framework Layer là Flutter Engine, được viết chủ yếu bằng C++ nhằm đảm bảo hiệu suất xử lý đồ họa tối ưu. Flutter Engine sử dụng thư viện đồ họa Skia – một thư viện đồ họa mã nguồn mở mạnh mẽ, hiệu suất cao được Google sử dụng trong nhiều sản phẩm nổi bật như Chrome và Android – để vẽ và hiển thị giao diện ứng dụng lên màn hình thiết bị.

Engine chịu trách nhiệm biên dịch mã Dart từ Dart VM thành mã máy native hiệu suất cao (biên dịch Ahead-of-Time - AOT). Flutter Engine còn quản lý các tác vụ quan trọng khác như xử lý animation, hiệu ứng chuyển cảnh (transition), quản lý bộ nhớ hiệu quả, xử lý text rendering và layout phức tạp.

Ngoài ra, Engine Layer còn quản lý sự kiện đầu vào từ người dùng như thao tác cảm ứng, chuột, bàn phím, và chuyển những sự kiện này đến Framework Layer để xử lý logic tương tác. Điều này đảm bảo khả năng phản hồi nhanh và mượt mà trong quá trình tương tác người dùng với ứng dụng.

**c) Embedder Layer (Lớp tương tác nền tảng)**

Embedder là lớp thấp nhất của kiến trúc Flutter, chịu trách nhiệm tương tác trực tiếp với hệ điều hành của từng nền tảng cụ thể (Android, iOS, Web, Windows, Linux, macOS). Embedder đảm nhiệm việc khởi tạo ứng dụng, quản lý vòng đời ứng dụng (lifecycle events) như khởi động, tạm dừng, tiếp tục, kết thúc ứng dụng, và cung cấp các dịch vụ liên quan đến nền tảng như camera, bộ nhớ, truy cập tập tin, âm thanh, cảm biến.

Flutter cung cấp sẵn một bộ embedder riêng biệt cho từng nền tảng hỗ trợ, cho phép ứng dụng Flutter tận dụng tối đa hiệu suất và khả năng tương thích với nền tảng gốc. Embedder layer sử dụng các API native để tương tác trực tiếp với hệ điều hành, giúp ứng dụng Flutter có trải nghiệm tương đương ứng dụng native về hiệu suất và độ ổn định.

#### 2.4.5.5. Ứng dụng thực tế của Dart và Flutter

Ngôn ngữ lập trình Dart kết hợp với Framework Flutter ngày càng được ứng dụng rộng rãi trong nhiều lĩnh vực thực tế nhờ vào những ưu điểm vượt trội về hiệu suất, khả năng phát triển nhanh chóng và khả năng tương thích đa nền tảng mạnh mẽ.

**a) Ứng dụng di động (Mobile Applications)**

Flutter hiện được ứng dụng rộng rãi nhất trong việc phát triển ứng dụng di động đa nền tảng. Các doanh nghiệp lớn như Google, Alibaba, BMW, eBay, Grab, Reflectly đều lựa chọn Flutter để xây dựng các ứng dụng di động nhờ vào hiệu suất cao, thời gian phát triển nhanh, và khả năng duy trì tính nhất quán giao diện và trải nghiệm người dùng trên cả hai nền tảng Android và iOS. Flutter giúp các nhà phát triển tiết kiệm đáng kể thời gian và chi phí do chỉ cần duy trì một bộ mã nguồn duy nhất mà vẫn đảm bảo ứng dụng có hiệu suất gần như tương đương native.

**b) Ứng dụng web (Web Applications)**

Flutter Web cho phép tạo các ứng dụng web có giao diện phong phú, độ tương tác cao với trải nghiệm đồng nhất với ứng dụng di động. Thay vì sử dụng các framework web truyền thống như React hay Angular, Flutter Web cho phép tái sử dụng mã nguồn Dart từ ứng dụng di động để xây dựng nhanh chóng các ứng dụng web, giúp giảm đáng kể thời gian phát triển. Ứng dụng Flutter Web thích hợp với các dự án yêu cầu giao diện đẹp, hiệu ứng animation mượt mà như các trang landing page, dashboard quản lý, hay các hệ thống nội bộ doanh nghiệp.

**c) Ứng dụng desktop (Desktop Applications)**

Flutter hiện hỗ trợ phát triển ứng dụng desktop native trên các hệ điều hành như Windows, macOS, Linux. Điều này mở rộng đáng kể khả năng ứng dụng của Flutter, cho phép xây dựng các ứng dụng quản lý, các ứng dụng văn phòng, công cụ đa năng trên nhiều nền tảng khác nhau, đồng thời tái sử dụng hầu hết mã nguồn từ ứng dụng di động, giúp giảm đáng kể thời gian và chi phí bảo trì.

**d) Ứng dụng IoT và hệ thống nhúng**

Flutter và Dart cũng được ứng dụng vào các thiết bị IoT, hệ thống nhúng và bảng điều khiển thông minh. Nhờ giao diện linh hoạt, đẹp mắt, và khả năng tối ưu hóa tài nguyên tốt, Flutter cho phép tạo giao diện người dùng trực quan, dễ thao tác cho các thiết bị y tế thông minh, nhà thông minh, thiết bị đo đạc và giám sát môi trường. Một số ứng dụng cụ thể bao gồm các hệ thống điều khiển trung tâm thông minh, màn hình tương tác trên xe hơi, và các thiết bị giám sát sức khỏe cá nhân.

**e) Lĩnh vực khác**

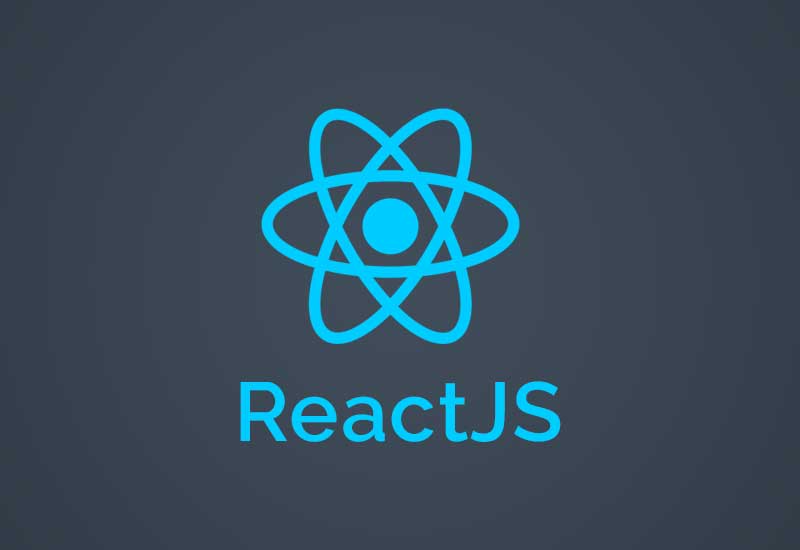
Flutter và Dart còn được sử dụng rộng rãi trong các ứng dụng trò chơi đơn giản, ứng dụng giáo dục tương tác, và các ứng dụng AR/VR nhờ khả năng xử lý đồ họa hiệu suất cao và khả năng tích hợp dễ dàng với các thư viện đồ họa mở rộng.

Với khả năng tương thích đa nền tảng vượt trội, hệ sinh thái rộng lớn và hiệu suất xử lý mạnh mẽ, Dart và Flutter đã và đang trở thành lựa chọn hàng đầu cho các doanh nghiệp và nhà phát triển ứng dụng trên toàn thế giới.

### 2.4.6. React.js

#### 2.4.6.1. Giới thiệu chung

**React.js** (gọi tắt là React) là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook vào năm 2013, với mục tiêu ban đầu là xây dựng giao diện người dùng (UI) hiệu quả, linh hoạt và dễ bảo trì. React nhanh chóng trở thành một trong những công cụ phát triển frontend phổ biến nhất trên thế giới nhờ vào kiến trúc hiện đại, khả năng tái sử dụng thành phần (component-based architecture) và hiệu suất xử lý mạnh mẽ.



* + - * 1. Logo React.js

Khác với các framework truyền thống như jQuery hoặc các thư viện DOM thuần, React giới thiệu khái niệm “Virtual DOM” – một bản sao ảo của cây DOM thực, cho phép tối ưu quá trình render bằng cách chỉ cập nhật những phần giao diện có thay đổi, thay vì làm mới toàn bộ. Điều này giúp nâng cao hiệu suất đáng kể, đặc biệt trong các ứng dụng web lớn, có số lượng phần tử giao diện và tương tác phức tạp.

React không phải là một framework hoàn chỉnh như Angular hay Vue.js mà là một **thư viện tập trung vào “view”** trong mô hình MVC (Model–View–Controller). Tuy nhiên, với sự hỗ trợ mạnh mẽ từ cộng đồng và hệ sinh thái phong phú, React có thể dễ dàng kết hợp với các thư viện khác để phát triển các ứng dụng web quy mô lớn, có tính năng đầy đủ và bảo trì dễ dàng.

#### 2.4.6.2. Cấu trúc và nguyên lý hoạt động

React xây dựng giao diện dựa trên mô hình **component** – mỗi component là một khối giao diện độc lập, có thể tái sử dụng và lồng ghép trong các component khác để tạo thành một cây giao diện phức tạp. Mỗi component có thể chứa logic riêng, trạng thái riêng (state), và có thể nhận dữ liệu từ bên ngoài thông qua props (properties).

Một trong những nguyên lý cốt lõi của React là tính **“unidirectional data flow”** (dòng dữ liệu một chiều). Dữ liệu đi từ component cha xuống component con thông qua props. Việc cập nhật giao diện sau khi dữ liệu thay đổi được quản lý tự động thông qua hệ thống lifecycle và Virtual DOM, giúp lập trình viên tránh được các lỗi phức tạp khi thao tác trực tiếp với DOM.

Khi state hoặc props thay đổi, React sẽ so sánh Virtual DOM mới với Virtual DOM cũ, xác định phần nào thay đổi (diffing algorithm) và chỉ cập nhật phần đó trên DOM thật, giúp ứng dụng phản hồi nhanh chóng và giảm tải đáng kể cho trình duyệt.

#### 2.4.6.3. Các tính năng nổi bật

React sở hữu nhiều tính năng nổi bật giúp quá trình phát triển frontend trở nên linh hoạt, mạnh mẽ và dễ quản lý:

* **Component-based architecture:** Cho phép chia nhỏ giao diện thành nhiều phần riêng biệt, dễ tái sử dụng, kiểm thử và bảo trì.
* **Virtual DOM:** Giảm thiểu chi phí cập nhật DOM thật, cải thiện hiệu suất đặc biệt trong ứng dụng có giao diện phức tạp.
* **JSX (JavaScript XML):** Cho phép viết code JavaScript pha trộn với cú pháp HTML, giúp dễ hình dung giao diện ngay trong mã nguồn.
* **Hooks API:** Được giới thiệu từ React 16.8, cho phép sử dụng state và các tính năng lifecycle trong các functional component mà không cần dùng class, đơn giản hóa cấu trúc và tăng khả năng tái sử dụng logic.
* **One-way data flow:** Dòng dữ liệu đi từ cha xuống con giúp quản lý dữ liệu rõ ràng, tránh lỗi không mong muốn do tương tác ngược dòng.
* **Server-side rendering (SSR):** Hỗ trợ render giao diện trên server, cải thiện hiệu năng tải trang và tối ưu hóa SEO (đặc biệt khi dùng với Next.js).
* **Concurrent Mode và Suspense (beta):** Giúp tối ưu trải nghiệm người dùng khi tương tác với dữ liệu bất đồng bộ, hỗ trợ “lazy loading” giao diện thông minh hơn.

#### 2.4.6.4. Ứng dụng thực tế

React được sử dụng rộng rãi trong nhiều loại ứng dụng từ nhỏ đến lớn nhờ vào tính mô-đun, dễ mở rộng và hiệu năng cao. Một số lĩnh vực và ví dụ ứng dụng thực tế của React bao gồm:

* **Mạng xã hội và ứng dụng tương tác:** Facebook, Instagram – nơi React được phát triển và áp dụng đầu tiên, sử dụng mạnh mẽ để xây dựng giao diện tương tác thời gian thực.
* **Thương mại điện tử:** Nhiều nền tảng như Shopify, Amazon (một phần), và các trang web thương mại điện tử lớn sử dụng React để xây dựng giao diện người dùng linh hoạt, đáp ứng nhanh.
* **Dashboard quản trị:** React rất phù hợp để xây dựng các giao diện quản lý phức tạp như admin panel, biểu đồ, bảng dữ liệu động, nhờ khả năng xử lý DOM hiệu quả và hỗ trợ thư viện biểu đồ như Chart.js, Recharts.
* **Ứng dụng SPA (Single Page Application):** React được thiết kế tối ưu cho SPA, cho phép chuyển đổi nội dung trên trang mà không cần tải lại toàn bộ trang, mang đến trải nghiệm liền mạch như ứng dụng gốc.
* **Ứng dụng di động (React Native):** React có thể kết hợp với React Native để xây dựng ứng dụng di động native cho Android và iOS từ cùng một nền tảng tư duy lập trình.

React.js là một công cụ mạnh mẽ và hiện đại cho việc xây dựng giao diện người dùng trong các ứng dụng web và mobile. Với kiến trúc component rõ ràng, hiệu suất vượt trội nhờ Virtual DOM, cú pháp JSX trực quan, và cộng đồng phát triển đông đảo, React cung cấp một giải pháp phát triển frontend vừa linh hoạt, vừa đáng tin cậy. Dù là một thư viện thay vì một framework toàn diện, nhưng nhờ hệ sinh thái phong phú và khả năng kết hợp với nhiều công nghệ khác như Redux, React Router, hoặc Next.js, React có thể đáp ứng tốt hầu hết nhu cầu phát triển ứng dụng hiện đại quy mô vừa và lớn. Chính vì vậy, React ngày càng được lựa chọn nhiều trong cả môi trường doanh nghiệp, startup lẫn giáo dục và nghiên cứu kỹ thuật.

# CHƯƠNG 3. THIẾT KẾ VÀ THI CÔNG



## 3.1. Thiết kế phần cứng

### 3.1.1. Sơ đồ nguyên lý của hệ thống

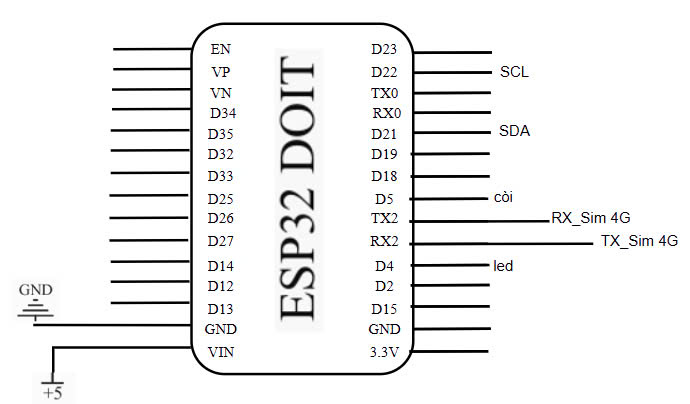
Sơ đồ nguyên lý của hệ thống được trình bày trên các hình từ hình 3.1 đến 3.4. Các chân của ESP-WROOM-32 được đấu nối với các chân của các linh kiện và thiết bị khác trong hệ thống theo bảng 3.1.

Bảng kết nối chân ESP-WROOM-32 với linh kiện và thiết bị

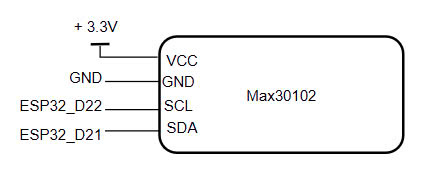
|  |  |  |
| --- | --- | --- |
| **ESP-WROOM-32** | **Chân kết nối** | **Linh kiện và thiết bị** |
| VIN | VCC | Nguồn + |
| GND | - | Nguồn - |
| IO5 | + | Còi |
| IO4 | + | Led |
| RX2 | TX | Module SIM |
| TX2 | RX | Module SIM |
| SCL | SCL | Max30102 BMP280 Oled |
| SDA | SDA | Max30102 BMP280 Oled |

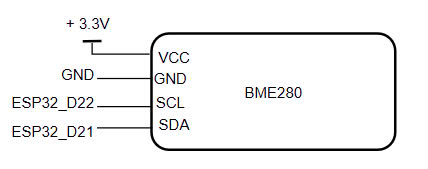


* + - * 1. Sơ đồ nguyên lý của khối nguồn

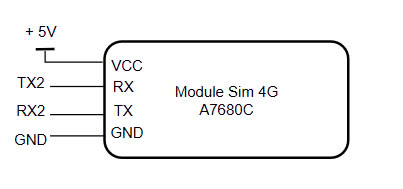


* + - * 1. Sơ đồ nguyên lý của khối xử lý trung tâm



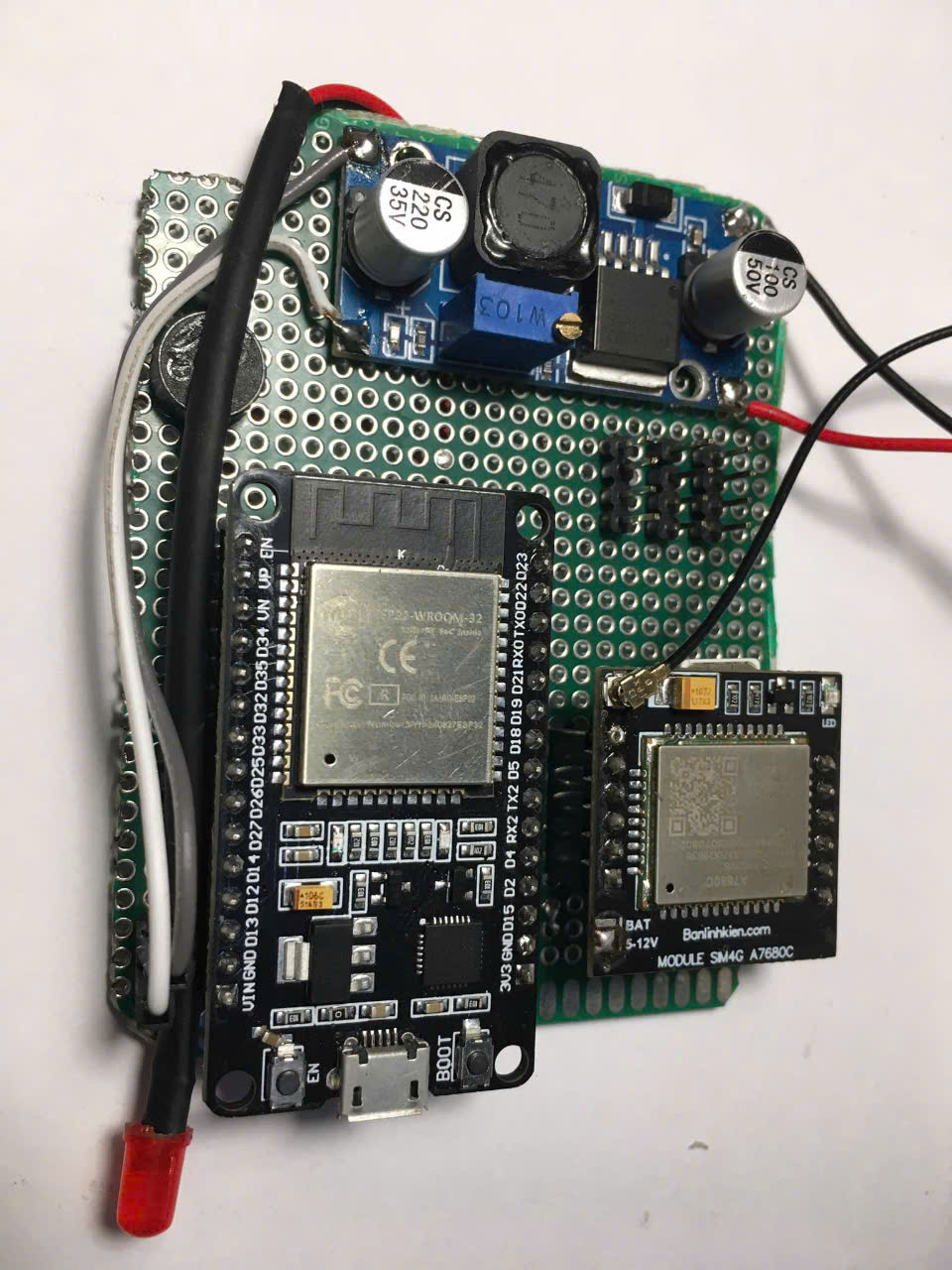


* + - * 1. Sơ đồ nguyên lý của khối cảm biến

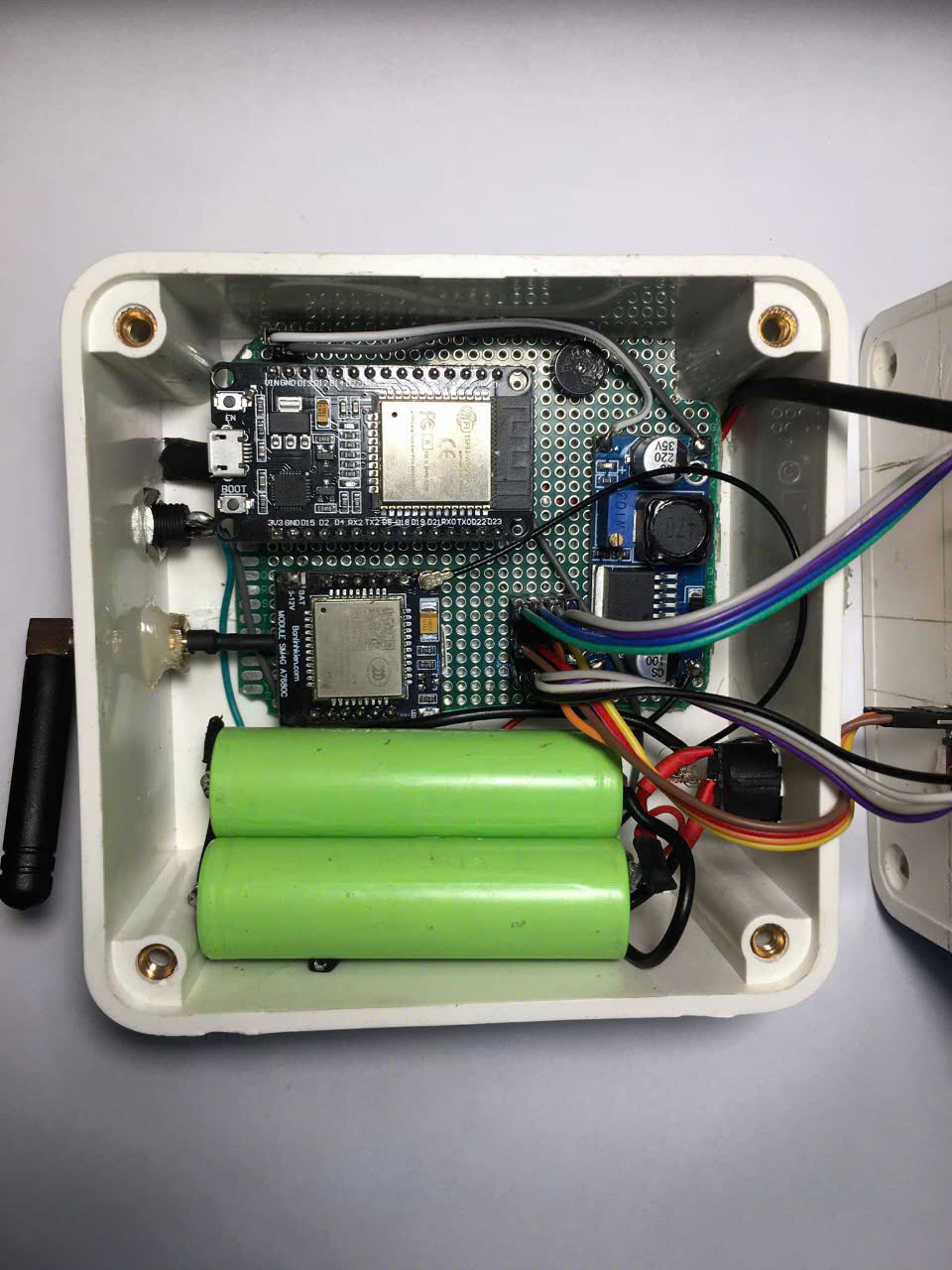


* + - * 1. Sơ đồ nguyên lý của khối SIM 4G

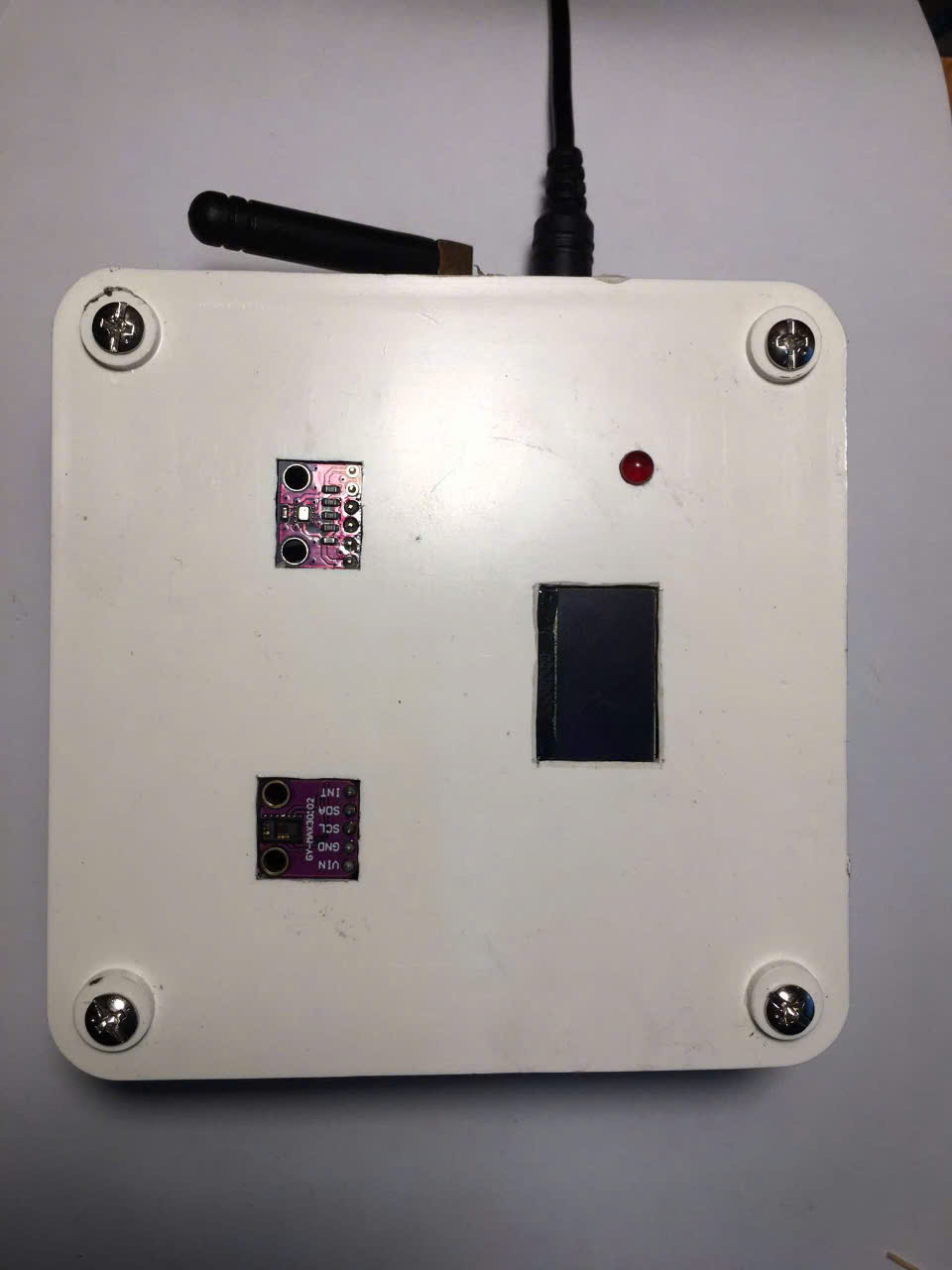
### 3.1.2. Thi công thiết bị



* + - * 1. Mạch điều khiển hoàn thiện



* + - * 1. Lắp đặt mạch in vào vỏ sản phẩm



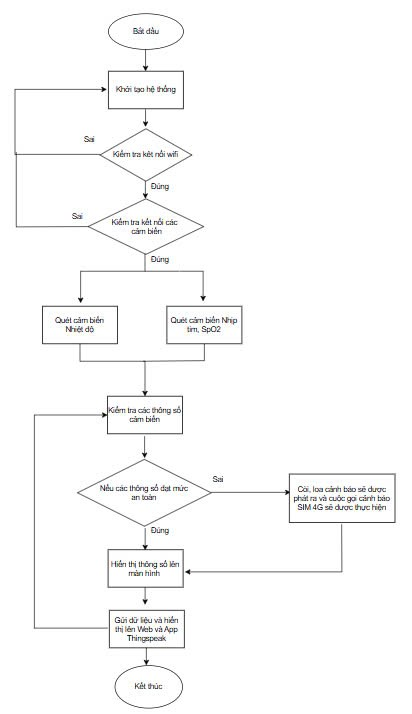
* + - * 1. Sản phẩm hoàn chỉnh

## 3.2. Thiết kế phần mềm

### 3.2.1. Lưu đồ thuật toán và nguyên lý hoạt động

Sau khi bật công tắc cấp nguồn, thiết bị sẽ tiến hành quá trình khởi tạo kết nối mạng không giây, module Sim, kết nối tới với các cảm biến và Thingspeak. Qúa trình đo thông số sẽ được bắt đầu khi người dùng đặt ngón tay lên các cảm biến. Các chỉ số sẽ hiển thị trên màn hình Oled của thiết bị và được cập nhật lên Thingspeak theo thời gian thực theo chu kỳ 10 giây 1 lần. Trong trường hợp thông số đạt đến giá trị cảnh báo thì led đỏ trên thiết bị sáng, loa cảnh báo sẽ được bật và đồng thời Module Sim được kích hoạt để thực hiện cuộc gọi đến số điện thoại đã được cài sẵn.

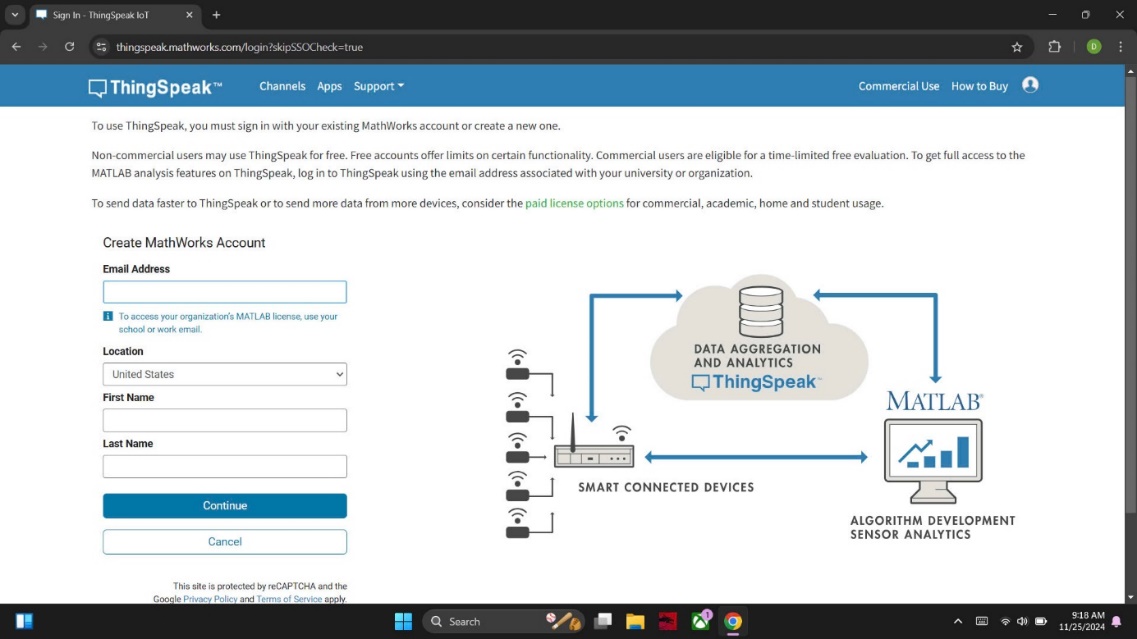
Trên Thinkspeak, các chỉ số được hiển thị dưới dạng đường đồ thị và chỉ số đang đo, dữ liệu được lưu trữ và xuất file khi cần thiết.



* + - * 1. Lưu đồ thuật toán hệ thống.

### 3.2.2. Tạo tài khoản và giám sát hoạt động của thiết bị trên Thingspeak

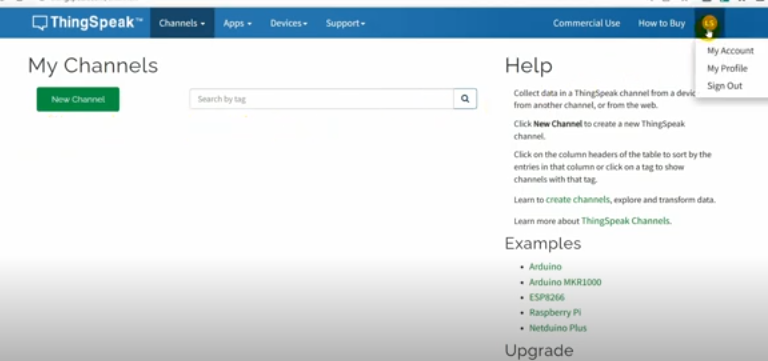
* Bước 1: Đăng ký tài khoản ThingSpeak:
* Truy cập vào địa chỉ <https://thingspeak.com> để mở giao diện trang chủ của Thinkspeak.
* Chọn Sign In ở góc trên, bên phải màn hình để mở giao diện đăng nhập, sau đó chọn “Create One” để mở giao diện tạo tài khoản mới.
* Điền các thông tin yêu cầu như địa chỉ email, khu vực, tên người dùng rồi chọn “Continue” (hình 3.9).
* Kiểm tra email và xác thực tài khoản bằng cách chọn “Verify Email”.



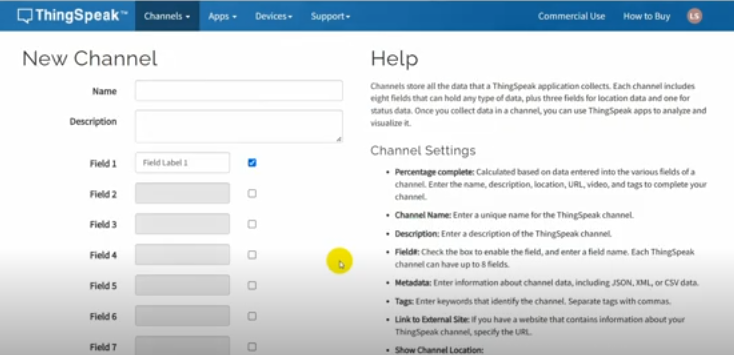
* + - * 1. Giao diện nhập thông tin đăng ký tài khoản Thingspeak

##### 

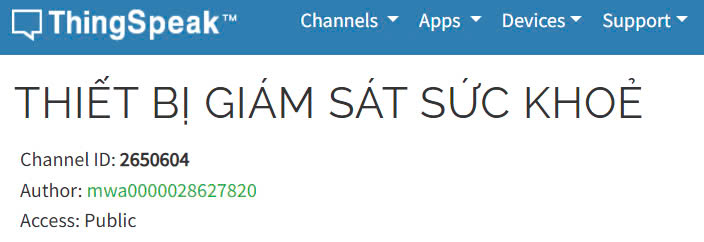
* + - * 1. Hoàn tất đăng ký tài khoản Thingspeak
* Bước 2: Hoàn tất việc đăng ký và đăng nhập vào tài khoản: Tại tab trình duyệt đã đăng ký tài khoản, chọn “Continue” để mở giao diện đặt mật khẩu cho tài khoản. Tại đây, tiến hành tạo mật khẩu theo hướng dẫn và chọn “I accept the Online Services Agreement”, rồi chọn “Continue” và “OK” để hoàn tất việc đăng ký tài khoản (hình 3.10).



* + - * 1. Đăng ký và đăng nhập tài khoản thành công
* Bước 3: Tạo đồ thị hiển thị dữ liệu
* Chọn New Channel để mở giao diện tạo kênh mới.
* Điền thông tin cho kênh: Tên (Name) và mô tả về kênh (Description)
* Chọn số lượng trường dữ liệu (tối đa là 8) và đặt tên theo mục đích sử dụng rồi chọn “Save Channel”. Khi đó, giao diện với các đồ thị để hiển thị dữ liệu với đầy đủ các thông tin về kênh dữ liệu được mở ra (hình 3.13).



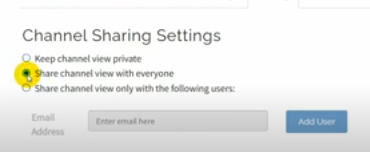
* + - * 1. Giao diện tạo kênh



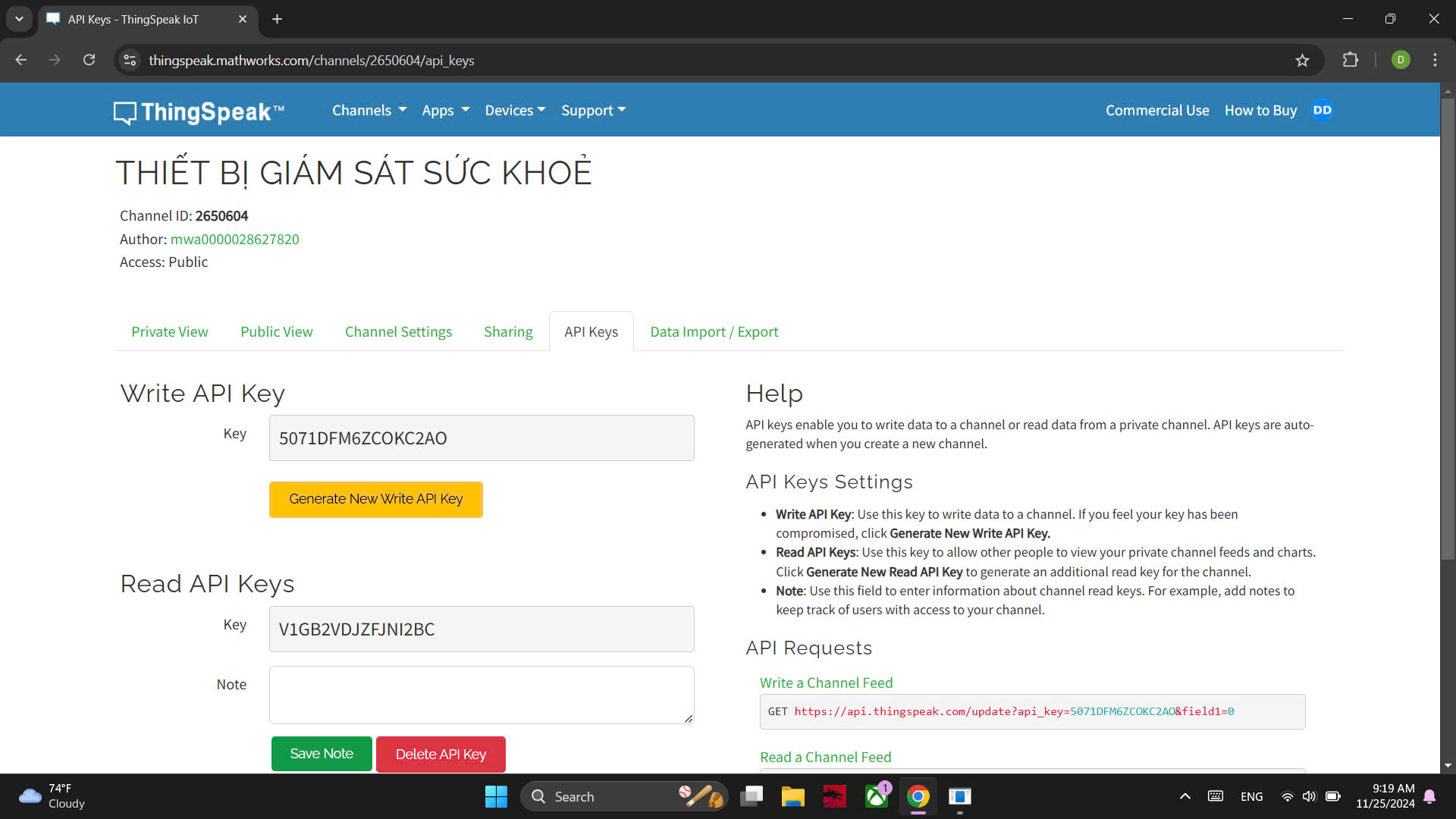
* + - * 1. Thông tin của kênh dữ liệu

##### 

* + - * 1. Đồ thị theo dõi dữ liệu
* Bước 4: Thiết lập một số tính năng khác cho kênh
* Chọn “Public View” nếu muốn dữ liệu hiển thị công khai.
* Chọn tab “Sharing” và chọn “Share channel view with everyone” (hình 3.15).



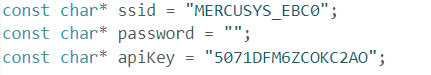
* + - * 1. Thiết lập chia sẻ dữ liệu của kênh với cộng đồng
* Bước 5: Lấy thông tin Keys API bằng cách tại tab “My Channels”, chọn “My channels” “API Keys” để mở giao diện quản lý API Keys (hình 3.16). Thông tin API Key cần được lưu lại để chuẩn bị cho việc lập trình, tải dữ liệu lên ThingSpeak hoặc xuất file dữ liệu khi cần thiết



* + - * 1. Thông tin API Keys

### 3.2.3. Cấu hình cho thiết bị

Cấu hình kết nối wifi cho thiết bị: Đoạn mã cấu hình wifi cho thiết bị được trình bày trên hình ảnh 3.17. Bằng cách sử dụng thư viện Wifi.h tích hợp vào sản phẩm, việc cấu hình kết nối wifi trở nên dễ dàng hơn, giúp người dùng quản lý các kết nối wifi một cách linh hoạt. Người dùng sẽ kết nối sản phẩm với mạng wifi bằng tên mạng được khai báo trong “ssid” và mật khẩu “pass”.



* + - * 1. Cấu hình wifi cho thiết bị

Cấu hình kết nối ứng dụng ThingSpeak với thiết bị: Để thiết bị có thể giao tiếp với ứng dụng ThingSpeak, người dùng cần khai báo đoạn mã URL và khai báo các trường tương ứng với các thông số trong mã nguồn chính (hinhf 3.18). Sau khi hoàn tất cấu hình wifi và kết nối với ứng dụng Thingspeak, tiến hành kiểm tra việc kết nối và truyền dữ liệu giữa thiết bị và ứng dụng bằng đoạn chương trình như trên hình 3.19.

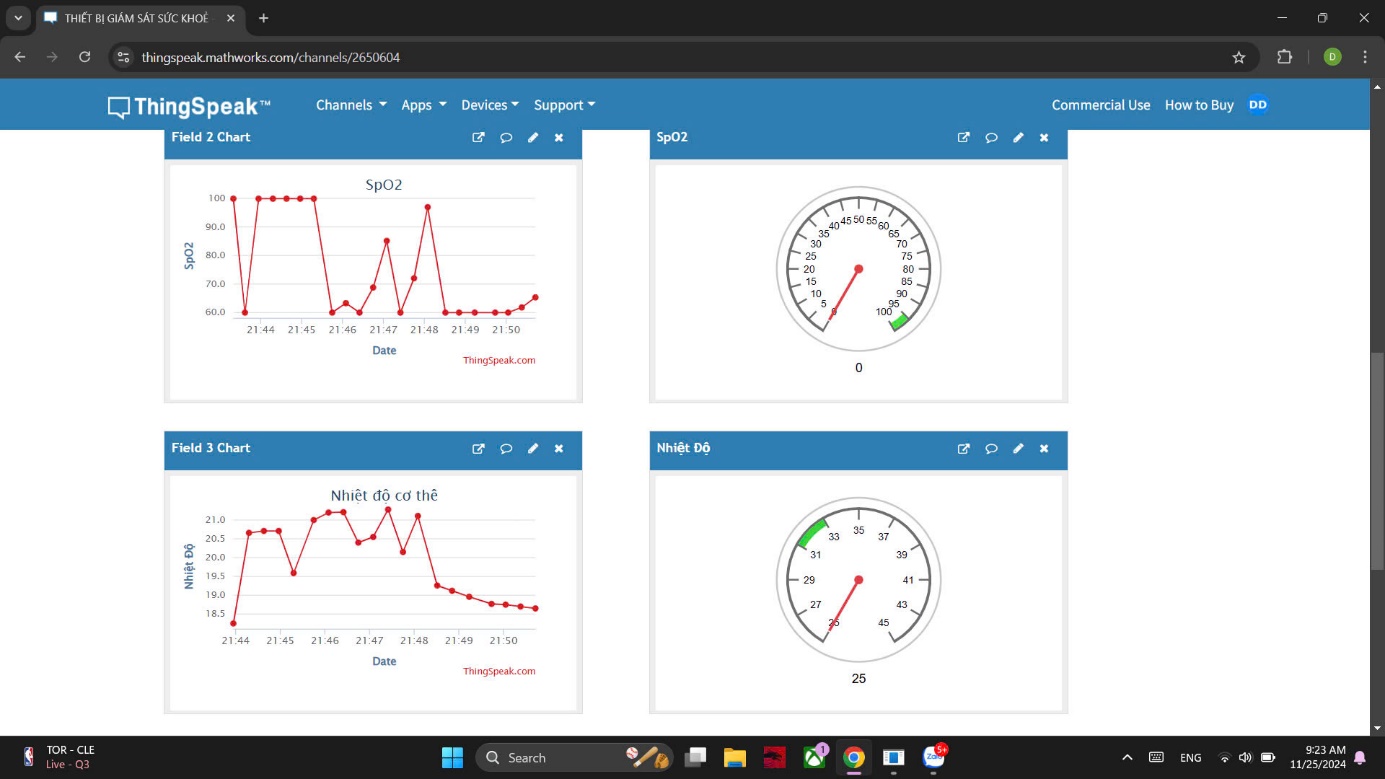


* + - * 1. Khai báo mã URL và các trường trong mã nguồn chính



* + - * 1. Kiểm tra kết nối và truyền dữ liệu giữa thiết bị và ThingSpeak

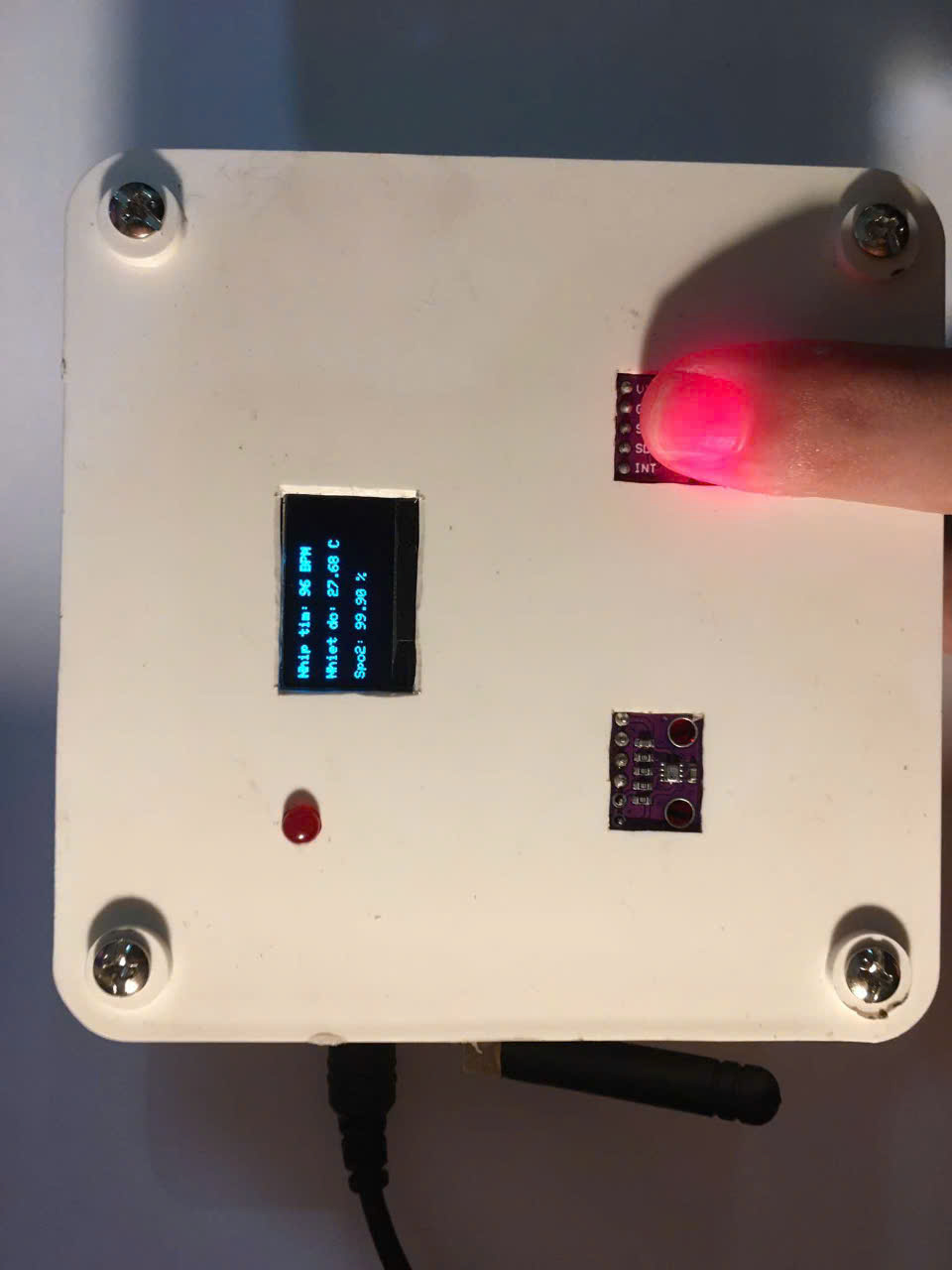
Sau khi kết nối thành công và iến hành nạp chương trình đã lập trình vào thiết bị, các dữ liệu thu được từ cảm biến sẽ được gửi lên ThingSpeak và người dùng có thể xem được đồ thị tại tab Private View (hình 3.20).



* + - * 1. Hiển thị dữ liệu trên ThingSpeak

## 3.3. Triển khai thử nghiệm và đánh giá hoạt động của sản phẩm

### 3.3.1. Thử nghiệm hoạt động của thiết bị



* + - * 1. Thiết bị khi thông số ổn định



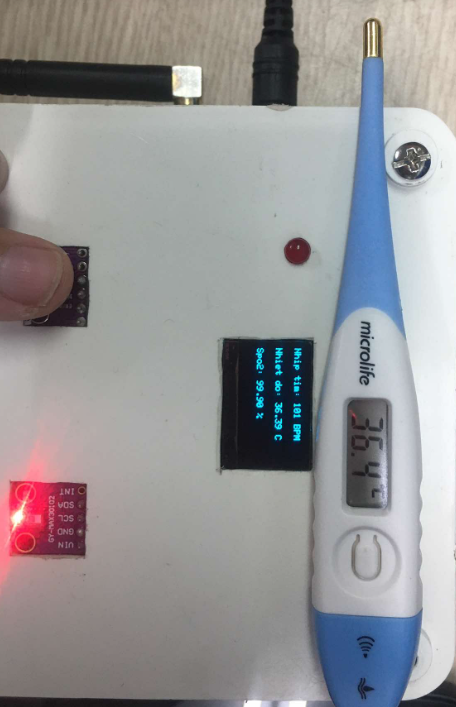
* + - * 1. Thử nghiệm chức năng cảnh báo và gọi điện thoại cảnh báo



* + - * 1. So sánh kết quả đo nhịp tim của thiết bị với đồng hồ thông minh



* + - * 1. So sánh kết quả đo SpO2 của thiết bị với đồng hồ thông minh

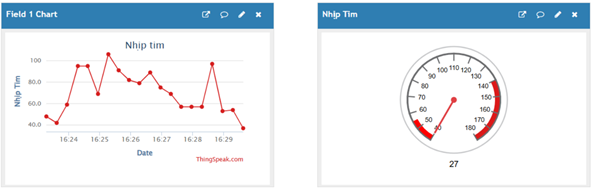


* + - * 1. So sánh kết quả đo thân nhiệtcủa thiết bị với nhiệt kế

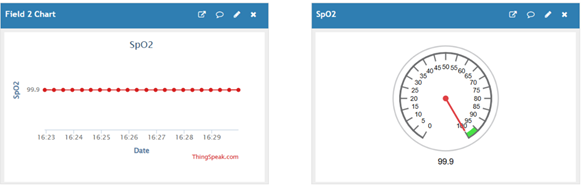
Bảng tổng hợp kết quả đo

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Lần đo** | **Nhịp tim (BMP)** | | **SpO2 (%)** | | **Nhiệt độ cơ thể (0C)** | |
| ***Thiết bị*** | ***Đồng hồ thông minh*** | ***Thiết bị*** | ***Đồng hồ thông minh*** | ***Thiết bị*** | ***Nhiệt kế truyền thống*** |
| 1 | 83 | 84 | 99.9 | 97 | 36.4 | 36.5 |
| 2 | 96 | 98 | 99.9 | 95 | 36.3 | 36.7 |
| 3 | 100 | 99 | 98.8 | 99 | 36.4 | 36.4 |
| 4 | 87 | 87 | 99.9 | 96 | 37.0 | 36.8 |
| 5 | 89 | 90 | 99.9 | 98 | 37.2 | 36.9 |
| 6 | 72 | 75 | 99.9 | 99 | 37.1 | 37.2 |
| 7 | 78 | 78 | 98.9 | 99 | 37.3 | 37.5 |
| 8 | 95 | 96 | 97.8 | 96 | 37.3 | 36.9 |
| 9 | 80 | 83 | 99.9 | 98 | 36.6 | 36.8 |
| 10 | 99 | 101 | 99.9 | 95 | 36.5 | 36.7 |

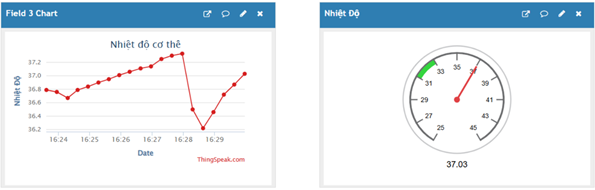
### 3.3.2. Thử nghiệm hoạt động của tài khoản ThinkSpeak



* + - * 1. Đồ thị nhịp tim



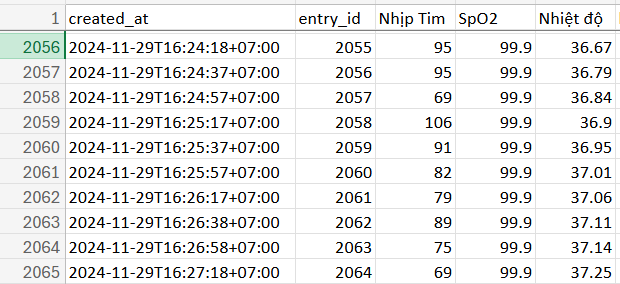
* + - * 1. Đồ thị nồng độ oxy trong máu SpO2



* + - * 1. Đồ thị nhiệt độ cơ thể



* + - * 1. Cập nhật thông số lên ThingSpeak theo thời gian thực



* + - * 1. Xuất file dữ liệu

### 3.3.3. Đánh giá hoạt động

* Thiết bị giám sát thông số sức khỏe hoạt động và thực hiện được các chức năng như thiết kế ban đầu đề ra.
* Các thông số sức khỏe đo bằng thiết bị do sinh viên thiết kế và chế tạo hiển thị rõ ràng trên thiết bị và có giá trị khá tương đồng với các thiết bị thương mại khác (đồng hồ thông minh và nhiệt kế).
* Dữ liệu được cập nhật và hiển thị trên tài khoản ThinkSpeak theo thời gian thực với độ trễ thấp, có thể chấp nhận được.

# KẾT LUẬN

**Những nhiệm vụ đã thực hiện được:**

* Tìm hiểu tổng quan về thông số sức khỏe và Internet vạn vật.
* Tìm hiểu các đặc tính kĩ thuật của linh kiện để phục vụ cho việc thiết kế mạch và lắp ráp sản phẩm.
* Tìm hiểu ngôn ngữ lập trình C trên nền tảng Arduino, nền tảng Thingspeak để phục vụ cho việc thiết kế phần mềm của sản phẩm.
* Thiết kế, lắp ráp, thử nghiệm và đánh giá hoạt động của thiết bị. Kết quả cho thấy thiết bị hoạt động và thực hiện được các chức năng như thiết kế ban đầu đề ra, nền tảng ThingSpeak trực quan, sử dụng dễ dàng, các thông số đo được khá chính xác, cập nhật và hiển thị lên nền tảng kịp thời.
* Tích luỹ kinh nghiệm chế tạo phần cứng và thiết kế phần mềm, làm cơ sở định hướng nghề nghiệp tương lai.

**Hạn chế của sản phẩm:**

* Tính thẩm mỹ của sản phẩm chưa cao, chức năng chưa đa dạng.
* Các linh kiện được sử dụng chỉ phù hợp với dự án của học sinh, sinh viên, chưa đảm bảo chất lượng nên đôi khi mô hình hoạt động không ổn định.
* Hoạt động của thiết bị phụ thuộc vào độ ổn định của wifi/4G.

**Hướng phát triển đề tài:**

* Phát triển thêm chức năng đo huyết áp.
* Cải thiện khả năng truyền nhận dữ liệu, giảm thiểu tối đa độ trễ và nhiễu.
* Cải thiện lại thiết kế cho tối ưu, thẩm mỹ nhất.

# TÀI LIỆU THAM KHẢO

**Tiếng Việt:**

1. TS. Lê Ngọc Bích, KS. Phạm Quang Huy, *Vi xử lý và Vi điều khiển*, Nhà xuất bản Thanh niên, 2020.
2. KS. Đặng Văn Hải, Lại Hồng Nhung, *Giáo trình Kỹ thuật cảm biến*, 2021.
3. TS. Nguyễn Tất Bảo Thiện, KS. Phạm Quang Huy*, Lập Trình IoT Với Arduino Esp8266 và Xbee,* Nhà xuất bản Thanh niên, 2018.

**Danh mục các Website tham khảo:**

1. [IoT là gì? Lợi ích và ứng dụng công nghệ Internet vạn vật](https://www.elcom.com.vn/iot-la-gi-loi-ich-va-cach-thuc-internet-of-things-van-hanh-1651741849), truy cập 19/9/2024.
2. [esp32-wroom-32\_datasheet\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf), truy cập 23/9/2024.
3. [Cảm biến Max30102](https://mecsu.vn/ho-tro-ky-thuat/may-do-oxy-xung-max30102-va-cam-bien-nhip-tim-voi-arduino.Db4?srsltid=AfmBOoou-9e0elnIxsjj5tZ93OMMKkdnfBNenkQm9J5zy_GoIuqeS4S_), truy cập 23/9/2024.
4. [Cảm biến BME280](https://dientu360.com/cam-bien-nhiet-do-do-am-ap-suat-khong-khi-bme280), truy cập 23/9/2024.
5. [Tiêu chuẩn nồng độ Oxi trong máu và nhịp tim của con người](https://nhathuoclongchau.com.vn/bai-viet/tieu-chuan-nong-do-oxy-trong-mau-va-nhip-tim-cua-nguoi-binh-thuong-la-bao-nhieu-48705.html), truy cập 20/9/2024.
6. [Tìm hiểu về chỉ số nhịp tim và huyết áp](https://medlatec.vn/tin-tuc/tim-hieu-ve-chi-so-nhip-tim-va-huyet-ap-s63-n30781), truy cập 30/9/2024.
7. [Bảng chỉ số huyết áp bình thường là bao nhiêu theo từng độ tuổi?](https://tamanhhospital.vn/huyet-ap-binh-thuong/), truy cập 20/10/2024.

# PHỤ LỤC

#include <Adafruit\_GFX.h> // OLED libraries

#include <Adafruit\_SSD1306.h> // OLED libraries

#include "MAX30105.h" // MAX3010x library

#include "heartRate.h" // Heart rate calculating algorithm

#include <Wire.h>

#include <SPI.h>

#include <Adafruit\_BMP280.h>

#include <WiFi.h>

#include <HTTPClient.h>

MAX30105 particleSensor;

const char\* ssid = "MERCUSYS\_EBC0"; // Replace with your WiFi SSID

const char\* password = ""; // Replace with your WiFi password

const char\* apiKey = "5071DFM6ZCOKC2AO"; // Replace with your ThingSpeak API key

unsigned long lastUpdate = 0; // Biến lưu thời gian lần gửi dữ liệu cuối cùng

const long updateInterval = 10000; // 10 giây

// Tính toán nhịp tim

const byte RATE\_SIZE = 4; // Số lượng trung bình

byte rates[RATE\_SIZE]; // Mảng nhịp tim

byte rateSpot = 0;

long lastBeat = 0; // Thời gian nhịp tim cuối

float beatsPerMinute;

int beatAvg;

// Tính toán SpO2

double avered = 0;

double aveir = 0;

double sumirrms = 0;

double sumredrms = 0;

double SpO2 = 0;

double ESpO2 = 60.0; // Giá trị khởi tạo

double FSpO2 = 0.7; // Hệ số lọc cho SpO2 ước tính

double frate = 0.95; // Lọc thấp cho giá trị IR/đỏ LED

int i = 0;

int Num = 30; // Lấy mẫu 30 lần để tính toán 1 lần

#define FINGER\_ON 7000 // Giá trị tối thiểu IR (xác định ngón tay có trên cảm biến)

#define MINIMUM\_SPO2 60.0 // Giá trị tối thiểu SpO2

// Cài đặt OLED

#define SCREEN\_WIDTH 128 // Chiều rộng OLED

#define SCREEN\_HEIGHT 64 // Chiều cao OLED

#define OLED\_RESET -1 // Pin reset

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET); // Khai báo tên màn hình

// Cấu hình cho BMP280

Adafruit\_BMP280 bmp; // I2C

int Tonepin = 4; // Chân của loa

void setup() {

Serial.begin(115200);

Serial.println("System Start");

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C); // Bắt đầu màn hình OLED

display.display();

delay(3000);

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

Serial.println("Không tìm thấy MAX30102");

while (1);

}

particleSensor.setup(0x7F, 4, 2, 800, 215, 16384);

particleSensor.enableDIETEMPRDY();

particleSensor.setPulseAmplitudeRed(0x0A);

particleSensor.setPulseAmplitudeGreen(0);

Serial.println(F("BMP280 test"));

if (!bmp.begin(0x76)) {

Serial.println(F("Không tìm thấy BMP280, kiểm tra kết nối!"));

while (1) delay(10);

}

bmp.setSampling(Adafruit\_BMP280::MODE\_NORMAL, Adafruit\_BMP280::SAMPLING\_X2, Adafruit\_BMP280::SAMPLING\_X16, Adafruit\_BMP280::FILTER\_X16, Adafruit\_BMP280::STANDBY\_MS\_500);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.print(".");

}

Serial.println("Connected to WiFi");

}

void loop() {

long irValue = particleSensor.getIR();

float temperature = bmp.readTemperature();

if (irValue > FINGER\_ON) {

if (checkForBeat(irValue) == true) {

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0, 5);

display.print("Nhip tim: ");

display.print(beatAvg); display.print(" BPM");

display.setCursor(0, 25);

display.print("SpO2: ");

if (beatAvg > 30) display.print(String(ESpO2) + "%");

else display.print("---- %");

display.display();

long delta = millis() - lastBeat;

lastBeat = millis();

beatsPerMinute = 60 / (delta / 1000.0);

if (beatsPerMinute < 255 && beatsPerMinute > 20) {

rates[rateSpot++] = (byte)beatsPerMinute;

rateSpot %= RATE\_SIZE;

beatAvg = 0;

for (byte x = 0; x < RATE\_SIZE; x++) beatAvg += rates[x];

beatAvg /= RATE\_SIZE;

}

}

uint32\_t ir, red;

double fred, fir;

particleSensor.check();

if (particleSensor.available()) {

i++;

ir = particleSensor.getFIFOIR();

red = particleSensor.getFIFORed();

fir = (double)ir;

fred = (double)red;

aveir = aveir \* frate + (double)ir \* (1.0 - frate);

avered = avered \* frate + (double)red \* (1.0 - frate);

sumirrms += (fir - aveir) \* (fir - aveir);

sumredrms += (fred - avered) \* (fred - avered);

if ((i % Num) == 0) {

double R = (sqrt(sumirrms) / aveir) / (sqrt(sumredrms) / avered);

SpO2 = -23.3 \* (R - 0.4) + 120;

ESpO2 = FSpO2 \* ESpO2 + (1.0 - FSpO2) \* SpO2;

if (ESpO2 <= MINIMUM\_SPO2) ESpO2 = MINIMUM\_SPO2;

if (ESpO2 > 100) ESpO2 = 99.9;

sumredrms = 0.0; sumirrms = 0.0; SpO2 = 0;

i = 0;

}

particleSensor.nextSample();

}

unsigned long currentMillis = millis();

if (currentMillis - lastUpdate >= updateInterval) {

lastUpdate = currentMillis;

String heartRate = String(beatAvg);

String oxygenLevel = String(ESpO2);

String temperatureStr = String(temperature);

String updateURL = "https://api.thingspeak.com/update?api\_key=";

updateURL += apiKey;

updateURL += "&field1=" + heartRate;

updateURL += "&field2=" + oxygenLevel;

updateURL += "&field3=" + temperatureStr;

HTTPClient http;

http.begin(updateURL);

int httpCode = http.GET();

if (httpCode > 0) {

String payload = http.getString();

Serial.println("ThingSpeak response: " + payload);

} else {

Serial.println("Error sending data to ThingSpeak");

}

http.end();

}

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE);

display.setCursor(0, 5);

display.print("Nhip tim: ");

display.print(beatAvg); display.print(" BPM");

display.setCursor(0, 25);

display.print("SpO2: ");

if (beatAvg > 30) display.print(String(ESpO2) + "%");

else display.print("---- %");

display.setCursor(0, 45);

display.print("Nhiet do: ");

display.print(temperature);

display.println(" C");

display.display();

} else {

for (byte rx = 0; rx < RATE\_SIZE; rx++) rates[rx] = 0;

beatAvg = 0; rateSpot = 0; lastBeat = 0;

avered = 0; aveir = 0; sumirrms = 0; sumredrms = 0;

SpO2 = 0; ESpO2 = 90.0;

display.clearDisplay();

display.setTextSize(2);

display.setTextColor(WHITE);

display.setCursor(30, 5);

display.println("Finger");

display.setCursor(30, 35);

display.println("Please");

display.display();

}

}