**ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH**

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA ĐIỆN – ĐIỆN TỬ

**BỘ MÔN ĐIỆN TỬ**

---------------o0o---------------

****

**LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC**

**THIẾT KẾ VÀ XÂY DỰNG NỀN TẢNG IOT   
ĐA GIAO THỨC**

**GVHD: Bùi Quốc Bảo**

**SVTH: Nguyễn Thanh Tâm**

**MSSV: 1613058**

**TP. HỒ CHÍ MINH, THÁNG 7 NĂM 2020**

ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

TRƯỜNG ĐẠI HỌC BÁCH KHOA Độc lập – Tự do – Hạnh phúc.

-----✩----- -----✩-----

Số: \_\_\_\_\_\_ /BKĐT

Khoa: **Điện – Điện tử**

Bộ Môn: **Điện Tử**

N**HIỆM VỤ LUẬN VĂN TỐT NGHIỆP**

1. HỌ VÀ TÊN : Nguyễn Thanh Tâm MSSV: 1613058

1. NGÀNH: **ĐIỆN TỬ - VIỄN THÔNG** LỚP : DD16VT03
2. Đề tài: Hệ
3. Nhiệm vụ (Yêu cầu về nội dung và số liệu ban đầu):

* Tìm hiểu các giao thức truyền tải thông dụng trong các giải pháp IoT
* Thiết kế phần cứng và thi công phần cứng IoT Gateway.
* Thiết kế phần cứng và thi công các thiết bị đầu cuối.
* Lập trình kết nối IoT Gateway và các thiết bị đầu cuối.
* Lập trình kết nối IoT Gateway vào IoT Platform trên Server.

1. Ngày giao nhiệm vụ luận văn: ...............................
2. Ngày hoàn thành nhiệm vụ: ...................................
3. Họ và tên người hướng dẫn: Phần hướng dẫn

Thầy Bùi Quốc Bảo .....................................

................................................................. .....................................

Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

*Tp.HCM, ngày…... tháng….. năm 20*

**CHỦ NHIỆM BỘ MÔN NGƯỜI HƯỚNG DẪN CHÍNH**

**PHẦN DÀNH CHO KHOA, BỘ MÔN:**

Người duyệt (chấm sơ bộ):.......................

Đơn vị:......................................................

Ngày bảo vệ : ...........................................

Điểm tổng kết: .........................................

Nơi lưu trữ luận văn: ...............................

***LỜI CẢM ƠN***

“Đi qua những năm tháng Bách Khoa, ta mới biết tuổi trẻ đáng trân trọng như thế nào. Trân trọng, không hẳn là vì có những lúc khó khăn tưởng chừng như gục ngã, không hẳn là vì ta biết mình trưởng thành đến đâu mà đơn giản là vì ta đã làm tất cả những điều đó cùng ai”

Cảm ơn cha mẹ

Cảm ơn thầy hướng dẫn….

Cảm ơn các bạn trong phòng thí nghiệm 209B3 đã tận tình giúp đỡ mình….

Cảm ơn CLB….

Cảm ơn Đảng và nhà nước….

Cảm ơn Bách Khoa, suốt khoảng thời gian 4 năm đã làm ta bao lần vấp ngã, làm bao….

*Tp. Hồ Chí Minh, ngày tháng năm .*

**Sinh viên**

**TÓM TẮT LUẬN VĂN**

Luận văn này trình bày về việc xây dựng hệ thống IoT để thu thập dữ liệu và điều khiển thiết bị thông qua các giao thức: Z-wave, LoraWan, Modbus Serial. Luận văn gồm ba phần chính:

**Hardware Gateway và các thiết bị đầu cuối:**

* Thiết kế và thi công phần cứng các thiết bị đầu cuối có hỗ trợ các giao thức kết nối ở trên, bao gồm: Van nước, Công tắc điện không dây, Node thu thập dữ liệu cảm biến,
* Thiết kế và thi công phần cứng các gateway cho từng giao thức.
* Tìm hiểu, thiết kế, kiểm thử các thông số công suất, độ lợi, nhiễu trên các loại anten.

**Firmware Gateway và các thiết bị đầu cuối:**

* Lập trình các interface cho các giao thức trên nền tảng FreeRTOS cho các thiết bị chạy vi điều khiển.
* Tìm hiểu, triển khai trình điều khiển cho các thiết bị chạy trên nền tảng máy tính tính nhúng.
* Tìm hiểu, lập trình các giao thức (TCP, MQTT,..) và các chuẩn đóng gói dữ liệu (JSON,…) để kết nối đến IoT Platform ).

**Software trên máy tính:**

* Lập trình giao diện Windows Form trên máy tính để cài đặt và kiểm tra trạng thái của gateway và các thiết bị đầu cuối.
* Triển khai LoraWan Network Server ở mạng cục bộ.

**MỤC LỤC**

[1. GIỚI THIỆU 1](#_Toc44451632)

[1.1 Tổng quan 1](#_Toc44451633)

[1.2 Tình hình nghiên cứu trong và ngoài nước 2](#_Toc44451634)

[1.3 Nhiệm vụ luận văn 3](#_Toc44451635)

[2. LÝ THUYẾT 4](#_Toc44451636)

[2.1. Tìm hiểu về các kết nối trong hệ thống IoT thực tế 4](#_Toc44451637)

[3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG 4](#_Toc44451638)

[4. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM (NẾU CÓ) 5](#_Toc44451639)

[5. KẾT QUẢ THỰC HIỆN 5](#_Toc44451640)

[6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 6](#_Toc44451641)

[6.1 Kết luận 6](#_Toc44451642)

[6.2 Hướng phát triển 6](#_Toc44451643)

[7. TÀI LIỆU THAM KHẢO 6](#_Toc44451644)

[8. PHỤ LỤC 7](#_Toc44451645)

DANH SÁCH HÌNH MINH HỌA

[Hình 5‑1 Kết quả thi công 3](#_Toc310380287)

[Hình 5‑2 Kết quả mô phỏng 3](#_Toc310380288)

**DANH SÁCH BẢNG SỐ LIỆU**

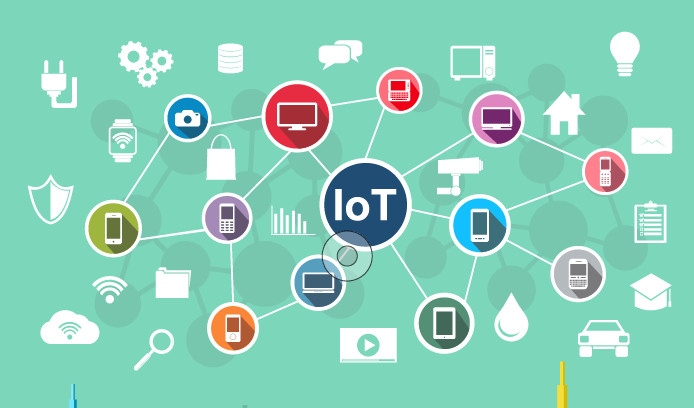
[Bảng 1 Thông số hệ thống 3](#_Toc310380293)

# GIỚI THIỆU

## Tổng quan

IoT- Internet Vạn Vật đã và đang là động lực phát triển của mọi lĩnh vực công nghệ. Trong năm 2018: theo Stantista, thị trường nhà thông minh toàn cầu đạt gần 60 tỷ USD; theo Gartner, chi tiêu cho an ninh IoT đạt 547 triệu USD; theo IDC, chi tiêu trên toàn thế giới về IoT đạt 772,5 tỷ USD. McKinsey ước tính có 127 thiết bị mới kết nối với Internet mỗi ngày, IoT có tác động kinh tế hàng năm từ 3.9 nghìn tỷ USD đến 11.1 nghìn tỷ USD vào năm 2025. Trong công nghiệp, Accenture ước tính Internet Công nghiệp có thể mang đến 14.2 nghìn tỷ USD cho nền kinh tế toàn cầu vào năm 2030. Trong vận tải, Business Insider ước tính rằng các nhà sản xuất toàn cầu sẽ đầu tư 70 tỷ USD vào năm 2020. Từ đó, ta có thể thấy rõ tầm ảnh hưởng lớn của IoT, cùng với những lợi ích và sự phát triển không ngừng nghỉ của nó.

Để đạt được tối ưu giá trị từ IoT, việc làm chủ một nền tảng IoT (IoT platform) là cần thiết. Nền tảng IoT là tập hợp những ứng dụng thực hiện việc liên kết giữa các thiết bị IoT và trung tâm dữ liệu từ đó tạo ra một hệ thống cung cấp đầy đủ thông tin được thu thập từ các thiết bị IoT (cảm biến, thiết điện tử tiêu dùng, thiết bị di động…) khác nhau để tạo và quản lý ứng dụng; lưu trữ và phân tích dữ liệu cũng như điều khiển từ xa các thiết bị IoT



## Tình hình nghiên cứu trong và ngoài nước

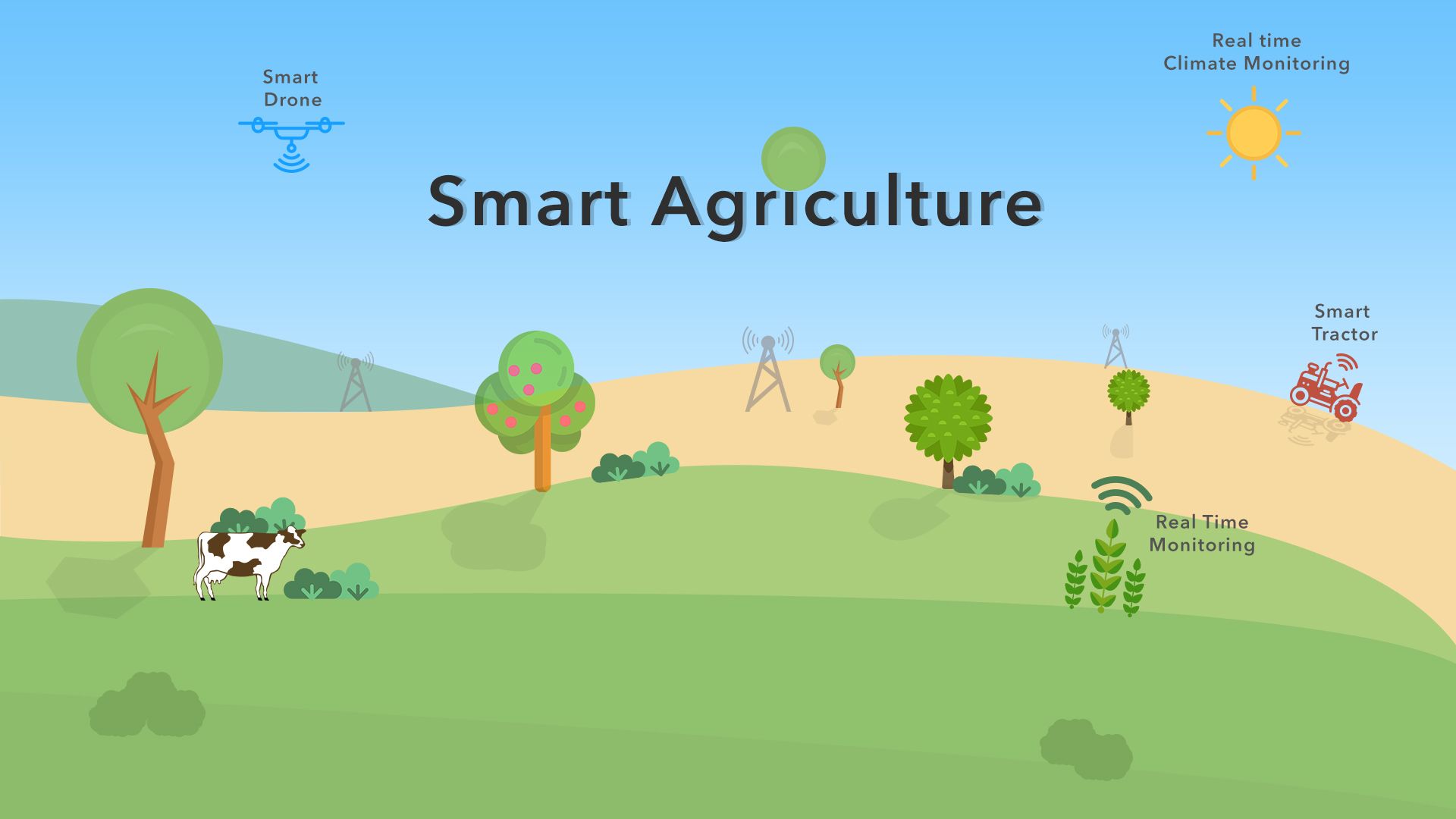
IoT hiện đang là động lực phát triển của hầu hết các lĩnh vực công nghệ. Các giải pháp IoT cho phép các doanh nghiệp cải thiến các hệ thống hiện có giúp nâng cao kết nối với khách hàng và đối tác. Một số ứng dụng IoT dẫn đầu xu hướng công nghệ hiện nay có thể kể đến như:

* Công nghiệp thông minh (Industrial IoT): Một số máy móc công nghiệp có thể hư hỏng hoặc quá hạn sử dụng, cần sự can thiệp từ các kĩ sư. Việc tích hợp các cảm biến và các kết nối không dây cho phép các máy móc có thể tự động phản hồi thông tin kịp thời về cho người quản lý.



Hình 1.2.1-1: Minh họa công nghiệp thông minh

* Nông nghiệp thông minh (Farming IoT): Với việc tích hợp các hệ thống cảm biến trên các cánh đồng, nông dân có thể nhận được các dữ liệu cần thiết để dự đoán về mùa màng, hoặc các hệ thống tự quản lý việc tưới tiêu, cung cấp ánh sáng.



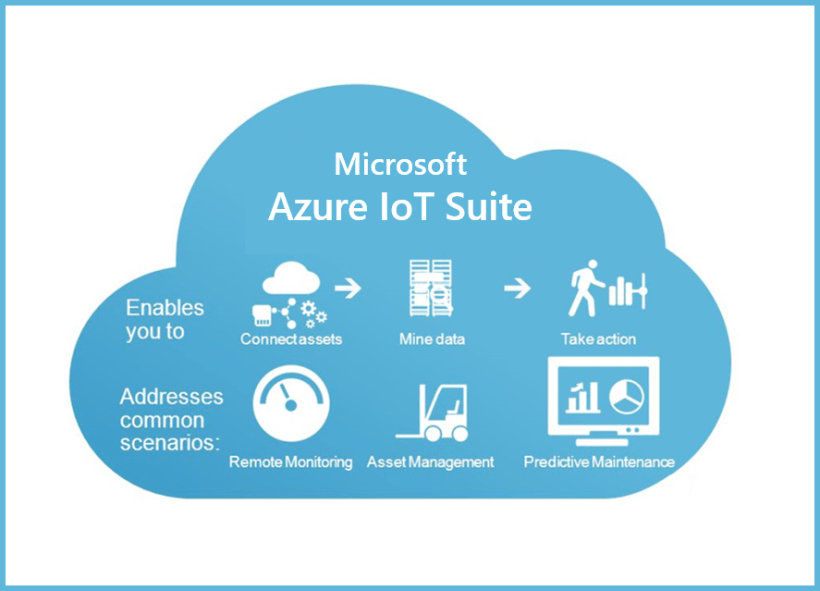
Hình 1.2.1-2: Minh họa nông nghiệp thông minh

* Vận chuyển thông minh (Logistics and transportation IoT): một trong những ứng dụng phổ biến là gán nhãn cho các xe container bằng cách sử dụng RFID. Điều này cho phép các công ty theo dõi quá trình vận chuyển hàng hóa, chọn ra đường đi tốt nhất.



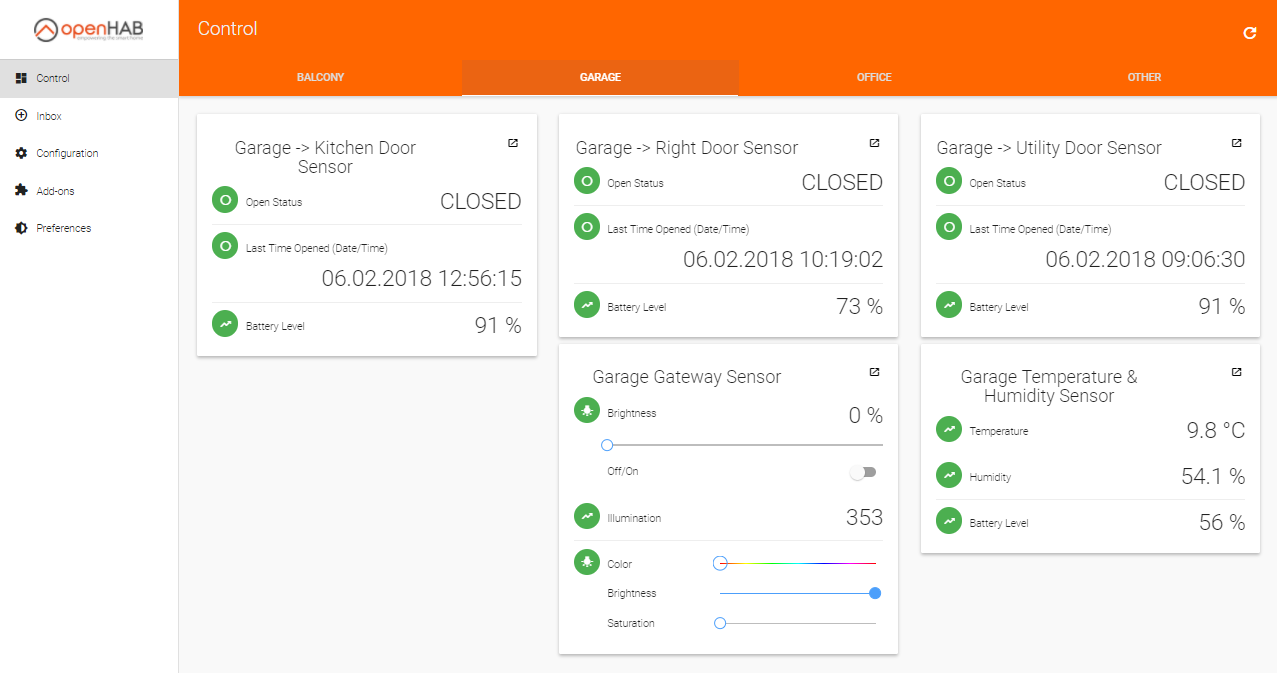
Hình 1.2.1-3: Minh họa vận chuyển thông minh

Các công ty dẫn đầu công nghiệp IoT trên thế giới hiện nay có thể kể đến như Google, Microsoft, Amazon, IBM, Oracle,... Trong đó, Google, Amazon và Microsoft chủ yếu cung cấp các nền tảng Cloud hỗ trợ lưu trữ dữ liệu, IBM và Oracle thiên về các thiết bị Gateway và phần cứng hoặc ứng dụng công nghệ AI, Block Chain...



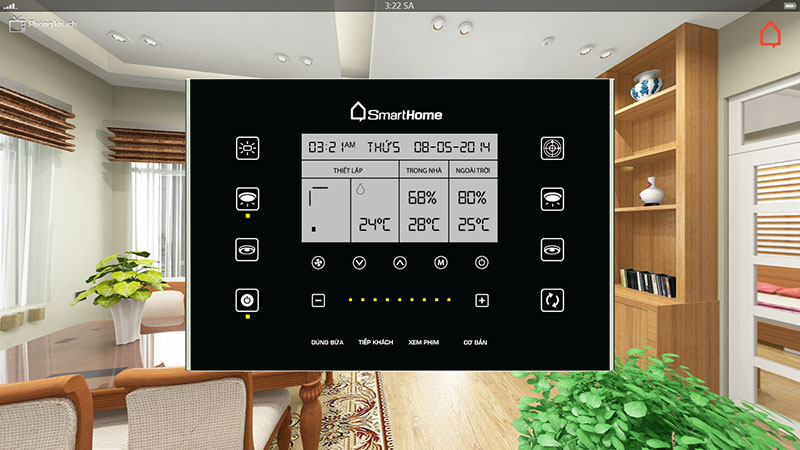
Hình 1.2.2-1: Microsoft Azure IoT

Một số nền tảng IoT mã nguồn mở như Kaa, ThingsBoard, OpenHAB thì không phải một nền tảng IoT hoàn chỉnh, mà chỉ tập trung vào giải pháp phần mềm hỗ trợ cho việc kết nối các thiết bị phần cứng.



Hình 1.2.2-2: Giao diện OpenHAB dashboard

Tại Việt Nam, nhu cầu về những giải pháp IoT đang rất lớn, để hỗ trợ cho các hệ thống IoT công-nông nghiệp như Nông nghiệp thông minh, Vận tải thông minh và Thành phố thông minh; tuy nhiên, những thương hiệu IoT lớn ở Việt Nam như Lumi, SmartHome by BKAV,… chỉ mới tập trung vào thiết kế và lắp đặt các thiết bị Nhà ở thông minh với những gói dịch vụ và sản phẩm đầu cuối nhất định, gây bất tiện cho người dùng trong việc nâng cấp và thiếu đa dạng trong việc chọn lựa. Bên cạnh đó, một vài cái tên trẻ như Inut, IoTmaker,.. đã xây dựng và phát triển nền tảng IoT của riêng họ với các giải pháp IoT cùng phần cứng. Tuy nhiên, các nền tảng này chỉ mới tập trung vào một giao thức truyền là Wifi với đặc trưng là công suất tiêu thụ lớn, nên khả năng ứng dụng vào các ứng dụng đòi hỏi năng lượng thấp, đặc biệt là nông nghiệp thông minh là không cao.



Hình 1.2.2-3: Smarthome by BKAV

Như vậy, việc tự xây dựng một nền tảng IoT hoàn chỉnh, hỗ trợ từ phần cứng đến phần mềm là mục tiêu cần thiết trong việc phát triển công nghệ IoT hiện nay.

## Nhiệm vụ luận văn

Các nhiệm vụ chính của luận văn:

* Thiết kế và thi công phần cứng gateway và các thiết bị đầu cuối cho từng giao thức .
* Lập trình giao tiếp giữa các gateway với các thiết bị đầu cuối tương ứng với từng giao thức.
* Lập trình cho các thiết bị đầu cuối ứng với từng chức năng đặt ra.
* Lập trình kết nối các gateway vào IoT Platform.

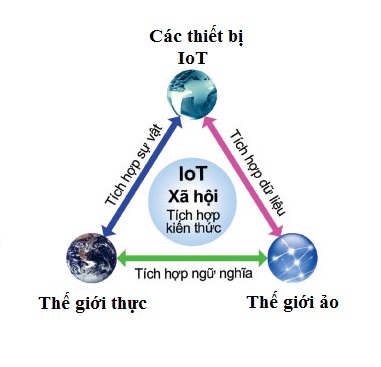
Từ các nhiệm vụ chính đó, cần phải thực hiện những nội dung:

* Nội dung 1: Tìm hiểu lý thuyết về các chuẩn giao tiếp, cách hoạt động, nghi thức quản lý, các thông số, ưu nhược điểm khi triển khai trên các hệ thống IoT cụ thể.
* Nội dung 2: Đề xuất, lựa chọn các giải pháp trong từng ứng dụng cụ thể.
* Nội dung 3: Phân tích, lựa chọn các thiết bị, linh kiện, cảm biến phù hợp cho hệ thống.
* Nội dung 4: Tìm hiểu, tham khảo các thiết kế có sẵn. Từ đó đưa ra sơ đồ chi tiết cho hệ thống.
* Nội dung 5: Tìm hiểu các quy tắc placement, layout mạch cho những thành phần đã đặt ra trên board. Sau đó thực hiện thi công mạch.
* Nội dung 6: Tìm hiểu về hệ điều hành FreeRTOS và cách triển khai hệ thống multi-tasking trên vi điều khiển. Từ đó phát triển firmware cho các thiết bị trên nền tảng Os này.
* Nội dung 7: Tìm hiểu cách giao tiếp, điều khiển và trao đổi dữ liệu giữa các modules (LoRaWAN, Zwave, Modbus, Ethernet, …) từ đó viết trình điều khiển cho từng module trên vi điều khiển và máy tính nhúng.
* Nội dung 8: Thống nhất và chuẩn hoá các loại dữ liệu thu được từ các gateway để giao tiếp với IoT Platform trên server.
* Nội dung 9: Kiểm thử, đo đạc các thông số của hệ thống. Sửa lỗi phần cứng và phần mềm.

# LÝ THUYẾT

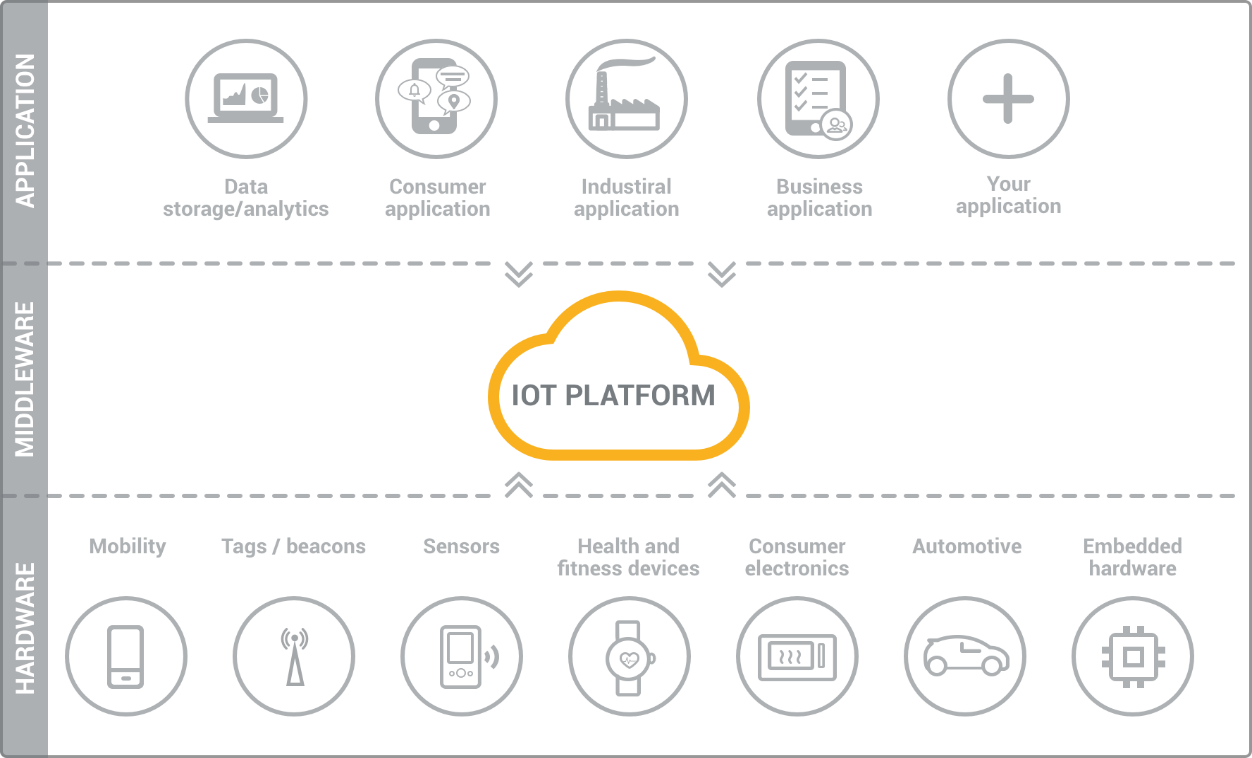
## 2.1. Tìm hiểu tổng quan về IoT và các kết nối trong một hệ thống IoT thực tế

### 2.1.1. Tổng quan về IoT

Internet Vạn Vật, hay cụ thể hơn là Mạng lưới vạn vật kết nối Internet hoặc là Mạng lưới thiết bị kết nối Internet (tiếng Anh: Internet of Things, viết tắt IoT) là một liên mạng, trong đó các thiết bị, phương tiện vận tải (được gọi là "thiết bị kết nối" và "thiết bị thông minh"), phòng ốc và các trang thiết bị khác được nhúng với các bộ phận điện tử, phần mềm, cảm biến, cơ cấu chấp hành cùng với khả năng kết nối mạng máy tính giúp cho các thiết bị này có thể thu thập và truyền tải dữ liệu. Hệ thống IoT cho phép vật được cảm nhận hoặc được điều khiển từ xa thông qua hạ tầng mạng hiện hữu, tạo cơ hội cho thế giới thực được tích hợp trực tiếp hơn vào hệ thống điện toán, hệ quả là hiệu năng, độ tin cậy và lợi ích kinh tế được tăng cường bên cạnh việc giảm thiểu sự can dự của con người. Khi IoT được gia tố cảm biến và cơ cấu chấp hành, công nghệ này trở thành một dạng thức của hệ thống ảo-thực với tính tổng quát cao hơn, bao gồm luôn cả những công nghệ như điện lưới thông minh, nhà máy điện ảo, nhà thông minh, vận tải thông minh và thành phố thông minh. Mỗi vật được nhận dạng riêng biệt trong hệ thống điện toán nhúng và có khả năng phối hợp với nhau trong cùng hạ tầng Internet hiện hữu.

Về cơ bản, một nền tảng IoT là hệ thống hoàn chỉnh, kết hợp các thiết bị phần cứng, các kết nối, phần mềm, các ứng dụng thông minh để tạo ra một giải pháp hiệu quả cho việc quản lý, cấu hình các thiết bị, thu thập và phân tích dữ liệu, kết nối đến các hệ thống cloud và server.

Các thành phần cấu tạo nên một hệ thống IoT bao gồm:

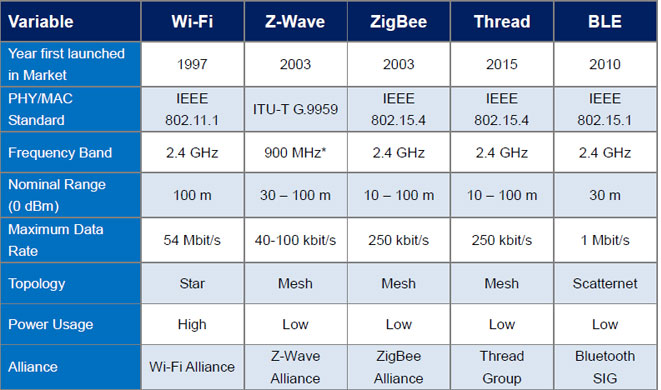
* Các thiết bị kết nối: là các máy móc, cảm biến, động cơ nhằm thực hiện các tác vụ được yêu cầu, ngoài ra còn có thể thu thập dữ liệu từ môi trường, kết nối, tương tác với các thiết bị khác, gửi dữ liệu về server,...
* Các kết nối không dây: đóng vai trò như một kênh giúp các thiết bị có thể giao tiếp và truyền dữ liệu qua lại, nhận lệnh từ server. Các yêu cầu đặt ra cho các chuẩn kết nối trong một nền tảng IoT là độ tin cậy, bảo mật, tốc độ và độ tiêu hao năng lượng.
* Phần mềm xử lý dữ liệu: có thể được chạy ở server hoặc cloud, có nhiệm vụ phân tích các dữ liệu thu thập được, đưa ra các yêu cầu cần thực hiện bên trong hệ thống IoT.
* Giao diện người dùng: cung cấp cho người dùng cái nhìn tổng thể về hoạt động của toàn bộ hệ thống, hỗ trợ người dùng tương tác với các thiết bị, quản lý hệ thống, hiển thị các thông tin dữ liệu một cách thân thiện với người dùng.
* 

### 2.1.2. Các kết nối trong hệ thống IoT thực tế

Kết nối IoT – kết nối vạn vật, trong đó trọng điểm là các giao thức kết nối để liên kết các thiết bị. Các ứng dụng IoT đều có các ràng buộc rõ ràng về phạm vi kết nối (không dây, có dây), điện năng tiêu thụ, tốc độ truyền tai dữ liệu, độ tin cậy và bảo mật. Giao thức truyền thông tin có rất nhiều với nhiều chủng loại thiết bị khác nhau dành cho các nhu cầu khác nhau. Những giao thức phổ biến trong các hệ thống IoT hiện nay:



Những giao thức phổ biến trong các hệ thống IoT hiện nay:

* Bluetooth: Bluetooth xuất hiện hầu hết ở các thiết bị như máy tính, điện thoại… Bluetooth hỗ trợ tốc độ truyền tải dữ liệu lên tới 70 Kbps trong phạm vi 10 – 100m và có công suất tiêu thụ lên đến 1W. Tuy nhiên, các chuẩn công nghệ Bluetooth mới hơn có hỗ trợ BLE – Bluetooth Low Energy giúp giảm công suất đáng kể (0.01 – 0.5W).
* LoRa: LoRa là công nghệ giao tiếp ở khoảng cách xa, có thể lên đến hàng kilomet mà không cần đến bộ khuếch đại. Hơn thế nữa, công suất tiêu thụ của các thiết bị sử dụng công nghệ LoRa là thấp, khoảng 0.1W. Công nghệ LoRa dễ bắt gặp trong các ứng dụng Sensor Network. Tuy nhiên, Tốc độ truyền thấp khoảng 0.3 – 50Kbps.
* Wifi: Wifi là giao thức thường được sử dụng để truyền hình ảnh, âm thanh nhờ băng thông lớn. Tuy nhiên, Wifi có phạm vi truyền sóng nhỏ và công suất tiêu thụ lớn.
* RS485: Có tốc độ truyền tương đối cao và ổn định, tuy nhiên, RS485 là giao thức có dây nên chỉ phù hợp để kết nối các trạm trong mạng cục bộ.
* Ethernet: là công nghệ truyền thông dùng để kết nối các mạng LAN cục bộ, có tốc độ cao hơn RS485, ít bị gián đoạn,độ bảo mật cao, tuy nhiên, giao thức này là giao thức có dây.
* Zigbee: Zigbee là một giao thức mở, sử dụng mạng lưới (mesh network) để mở rộng hệ thống. Zigbee phù hợp với hệ thống cần nhiều nút mạng. Tốc độ truyền khoảng 40-250Kbps. Cũng như các giao thức trên, công nghệ này cũng tiêu hao ít năng lượng. Nhược điểm của giao thức này là khoảng cách giữa các nút còn hạn chế, khó xuyên tường, đường truyền có độ tin cậy thấp, chưa thực sự thích hợp cho các dự án IoT thông minh ở Việt Nam.
* Z-Wave: Tương tự như Zigbee, Z-Wave cũng sử dụng mạng lưới để mở rộng phạm vi kết nối. Ngoài ra, Z-Wave có hầu hết các ưu điểm của Zigbee và cải thiện được khoảng cách truyền sóng giữa các nút (lên đến 100m), Z-Wave sử dụng tần số 900MHz – dải tần ít được các giao thức sử dụng nên tránh được các xung đột. Vì thế, giao thức Z-Wave có độ tin cậy cao hơn. Đây là công nghệ đang được chú ý và được các nhà sản xuát tập trung nhiều hơn.
* 

// Nếu thiếu sẽ extend khúc này ra + hình ảnh

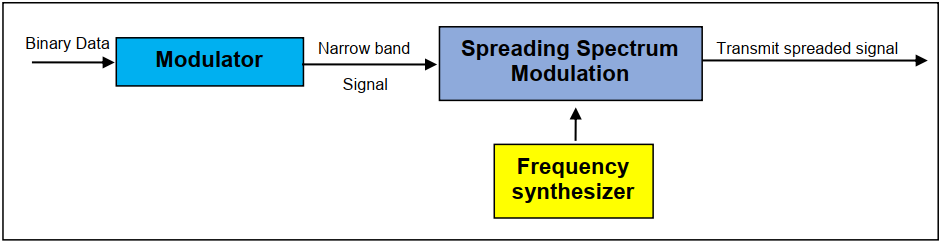
// Khúc này copy lý thuyết lora, lorwan Modbus. Zwave đông nghẹt vào,

## 2.2 Giới thiệu công nghệ LoRa và giao thức LoRaWAN

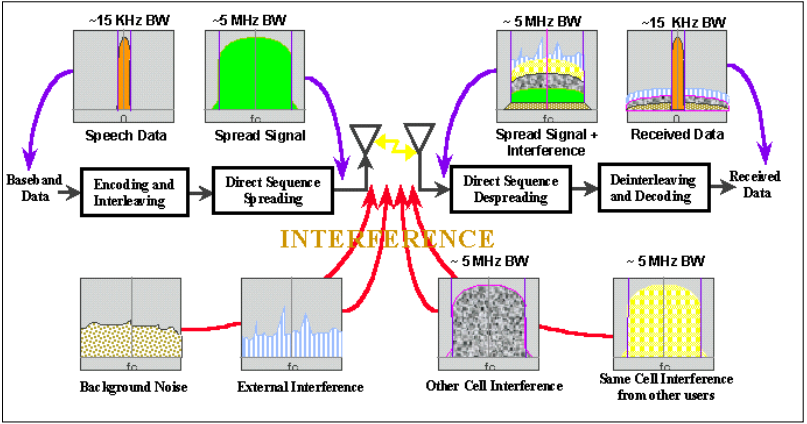
### 2.2.1 Giới thiệu công nghệ truyền thông không dây LoRa

Theo công ty Semtech (đơn vị sở hữu bản quyền công nghệ LoRa) thì LoRa là công nghệ truyền thông không dây, sử dụng năng lượng thấp, thông tin được bảo mật khi truyền và đang là giải pháp mới trong việc xây dựng mạng lưới kết nối các thiết bị điện tử trong khu vực rộng [1].

Công nghệ LoRa sử dụng sơ đồ điều chế trải phổ (spread spectrum modulation), công nghệ này bắt nguồn từ kĩ thuật điều chế Chirp Spectrum (CSS) nhưng điều chỉnh được tốc độ dữ liệu (data rate) để có được độ nhạy (sensitivity) tốt nhất trong điều kiện băng thông cố định (fixed channel bandwidth). Ngoài ra, trong quá trình mã hóa tín hiệu trước khi truyền, tín hiệu đi qua bộ “spreading factor” cùng với việc điều chỉnh được data rate nên hệ thống đáp ứng được việc truyền tín hiệu ở khoảng cách xa, tiết kiệm năng lượng. Với điểm mạnh trên, người dùng có thể tối ưu mạng lưới các thiết bị trong cùng điều kiện băng thông cố định.



Hình 2. Sơ đồ điều chế tại phía phát dùng công nghệ LoRa

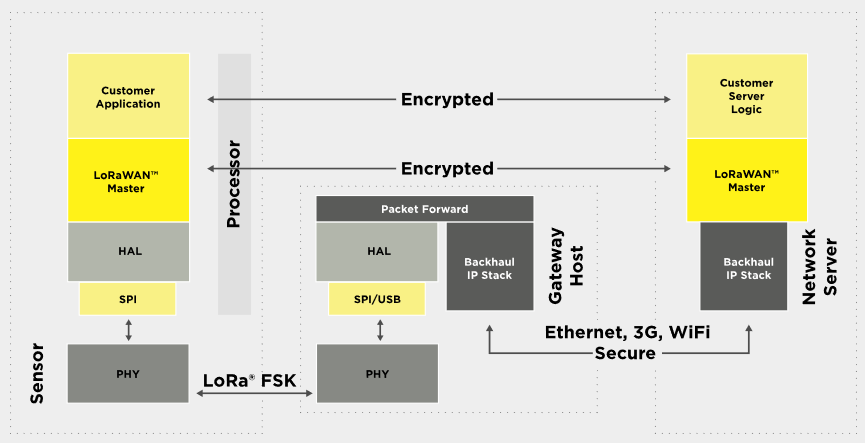


Hình 3. Mô phỏng hình dạng tín hiệu khi truyền và nhận trong mạng LoRa [2].

Sơ đồ khối trên thể hiện quá trình điều chế tín hiệu dùng kĩ thuật LoRa trước khi phát ra anten. Kỹ thuật điều chế Chirp Spread Spectrum được phát triển vào năm 1940 trong ứng dụng radar, trao đổi thông tin trong quân sự. Tuy nhiên, kỹ thuật này vẫn được phát triển để ứng dụng trong đời sống ngày nay. Kỹ thuật này dùng chuỗi sóng hình sin có tần số tăng hoặc giảm theo thời gian để mã hóa tín hiệu [3]. Thông số “Spreading factor” cho ta biết mỗi bit của thông tin cần truyền đi được mã hóa bởi chuỗi sóng sin dài hay ngắn.

### Giới thiệu giao thức LoRaWAN

Giao thức LoRaWAN được phát triển dựa trên giao thức LPWA (Low Power, Wide Area). Các thiết bị sử dụng giao thức LoRaWAN có dạng kiến trúc mạng hình sao – thiết bị gateway ở trung tâm có chức năng chuyển tiếp gói tin giữa thiết bị đầu cuối và server. Thiết bị gateway kết nối với server thông qua các cách như: Ethernet, 3G, Wifi [4].



Hình 4. Giao thức LoRaWAN trong các thiết bị

Giao thức LoRaWAN được chia thành 3 lớp (class) khác nhau. Tùy vào mục đích sử dụng, thiết bị đầu cuối sẽ được tích hợp class tương ứng. Mỗi class có cách thức hoạt động khác nhau.

#### Class A:

Tất cả các thiết bị đầu cuối đều được hỗ trợ lớp này. Khi thiết bị hoạt động ở lớp A, các gói tin truyền lên server được gửi đi tại mọi thời điểm. Ngay sau khi gửi gói tin, thiết bị mở cổng “downlink” để nhận gói tin về nhưng chỉ trong thời gian ngắn. Vì việc nhận gói tin phụ thuộc vào thời điểm truyền gói tin nên thiết bị đầu cuối hoạt động ở lớp A tiết kiệm năng lượng nhất.

#### Class B:

Khác với lớp A, các thiết bị ở lớp B được đồng bộ với mạng bằng những gói tin định kì. Việc mở cổng “downlink” cũng tuân theo lịch trình đã lên sẵn. Việc bổ sung này giúp thiết bị đầu cuối có thể nhận được tín hiệu từ server thông qua gateway thường xuyên. Tuy nhiên, các thiết bị hoạt động ở lớp B tiêu thụ năng lượng cao hơn lớp A.

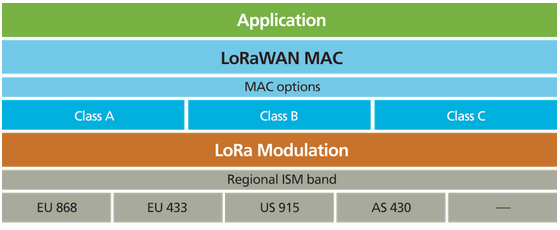
#### Class C:

Khác với sự hoạt động các thiết bị ở lớp A, thiết bị ở lớp C mở cổng “downlink” tại mọi thời điểm, khi đóng cổng “downlink” thì đồng thời với việc mở cổng truyền dữ liệu lên gateway. Các thiết bị hoạt động ở lớp C có độ trễ nhận, truyền gói tin cực kì thấp. Cũng vì thế, các thiết bị này không phù hợp với mục đích tiết kiệm năng lượng.

Ngày nay, để phù hợp với mục đích sử dụng thì nhiều người đã tích hợp lớp A và lớp C chung trong một thiết bị. Việc này giúp người dùng thoải mái tùy chỉnh để phù hợp mục đích sử dụng.



Hình 5. Mô hình hóa quá trình hoạt động của 3 class trong giao thức LoRaWAN.



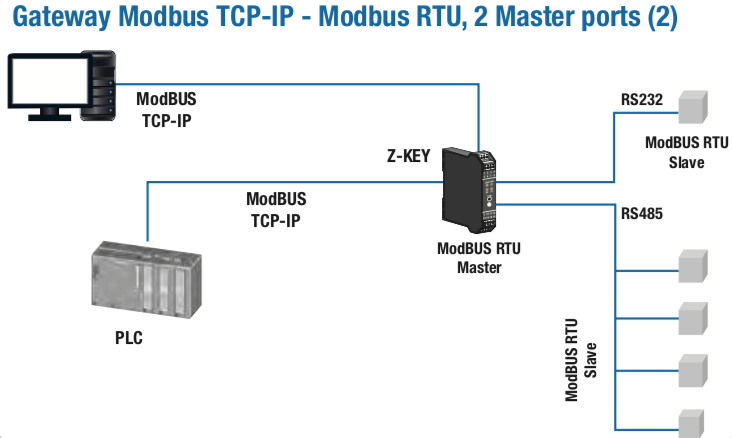
Hình 6. Mô hình giao thức LoRa chi tiết

## Giới thiệu giao thức Modbus

Modbus do Modicon (hiện nay thuộc Schneider Electric) phát triển năm 1979, là một phương tiện truyền thông với nhiều thiết bị thông qua một cặp dây xoắn đơn. Ban đầu, nó hoạt động trên RS232, nhưng sau đó nó sử dụng cho cả RS485 để đạt tốc độ cao hơn, khoảng cách dài hơn, và mạng đa điểm (multi-drop). MODBUS đã nhanh chóng trở thành tiêu chuẩn thông dụng trong ngành tự động hóa, và Modicon đã cho ra mắt công chúng như một protocol miễn phí.

Modbus là một hệ thống “Master - Slave”, “Master” được kết nối với một hay nhiều “Slave”. “Master” thường là một PLC, PC, DCS, hay RTU. “Slave” Modbus thường là các thiết bị hiện trường, tất cả được kết nối với mạng trong cấu hình multi-drop. Khi một chủ Modbus muốn có thông tin từ thiết bị, chủ sẽ gửi một thông điệp về dữ liệu cần, tóm tắt dò lỗi tới địa chỉ thiết bị. Mọi thiết bị khác trên mạng sẽ nhận thông điệp này nhưng chỉ có thiết bị nào được chỉ định mới có phản ứng.

Các thiết bị trên mạng Modbus không thể tạo ra kết nối, chúng chỉ có thể phản ứng. Nói cách khác, chúng “lên tiếng” chỉ khi được “nói tới”. Một số nhà sản xuất đang phát triển các thiết bị lai ghép hoạt động như các tớ MODBUS, tuy nhiên chúng cũng có “khả năng viết”, do đó làm cho chúng trở thành các thiết bị chủ ảo.



Hiện nay, có 03 chuẩn Modbus đang được sử dụng phổ biến trong công nghiệp - tự động hóa là: Modbus RTU, Modbus ASCII, Modbus TCP. Tất cả thông điệp được gửi dưới cùng một format. Sự khác nhau duy nhất giữa 3 loại Modbus là cách thức thông điệp được mã hóa. Cụ thể:

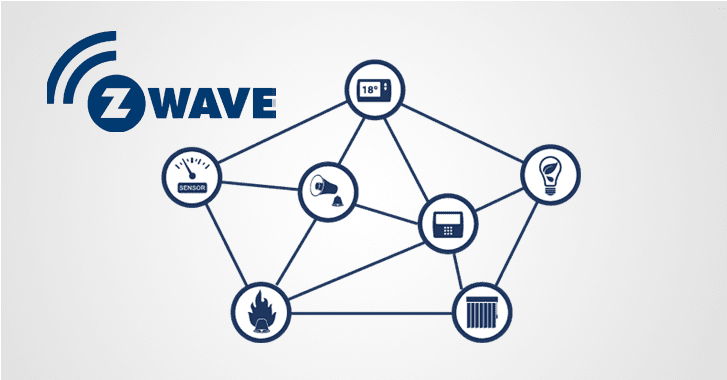
Modbus ASCII: Mọi thông điệp được mã hóa bằng hexadeci-mal, sử dụng đặc tính ASCII 4 bit. Đối với mỗi một byte thông tin, cần có 2 byte truyền thông, gấp đôi so với Modbus RTU hay Modbus TCP. Tuy nhiên, Modbus ASC II chậm nhất trong số 3 loại protocol, nhưng lại thích hợp khi modem điện thoại hay kết nối sử dụng sóng radio do ASCII sử dụng các tính năng phân định thông điệp. Do tính năng phân định này, mọi rắc rối trong phương tiện truyền dẫn sẽ không làm thiết bị nhận dịch sai thông tin. Điều này quan trọng khi đề cập đến các modem chậm, điện thoại di động, kết nối ồn hay các phương tiện truyền thông khó tính khác.

Modbus RTU: Dữ liệu được mã hóa theo hệ nhị phân, và chỉ cần một byte truyền thông cho một byte dữ liệu. Đây là thiết bị lí tưởng đối với RS232 hay mạng RS485 đa điểm, tốc độ từ 1200 đến 115 baud. Tốc độ phổ biến nhất là 9600 đến 19200 baud. Modbus RTU là protocol công nghiệp được sử dụng rộng rãi nhất.

Modbus TCP: Modbus TCP đơn giản là Modbus qua Ethernet. Thay vì sử dụng thiết bị này cho việc kết nối với các thiết bị tớ, do đó các địa chỉ IP được sử dụng. Với Modbus TCP, dữ liệu Modbus được tóm lược đơn giản trong một gói TCP/IP. Do đó, bất cứ mạng Ethernet hỗ trợ TCP/IP sẽ ngay lập tức hỗ trợ Modbus TCP.

## Giới thiệu công nghệ Zwave

Z-Wave là một giao thức truyền thông không dây được phát triển bởi Zensys vào năm 2001. 7 năm sau, Sigma Designs đã mua độc quyền công nghệ này.

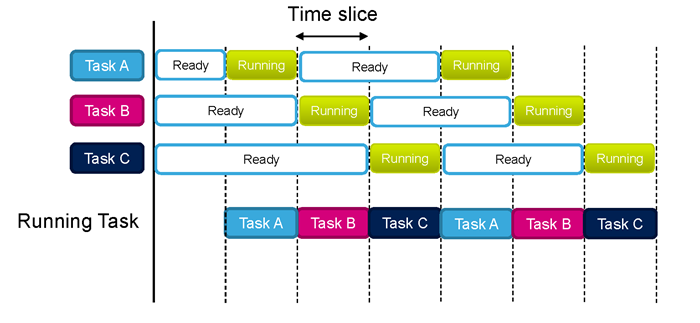


Z-Wave bao gồm một mạng lưới sử dụng sóng vô tuyến năng lượng thấp để giao tiếp. Chủ yếu được sử dụng để kết nối thiết bị thắp sáng tự động, tiện ích sưởi, công cụ bảo mật và các thiết bị thông minh khác. Z-Wave hoạt động ở tần số 800-900 MHz, Z-Wave cho phép các thiết bị tự kết nối với nhau và tạo thành một mạng mới. Thường thì sẽ có 1 thiết bị trung tâm đảm nhiệm việc kết nối với internet nhưng bản thân các thiết bị Z-Wave (bóng đèn, cảm biến…) không hề sử dụng đến Internet và chỉ sử dụng Z-Wave để giao tiếp với nhau cũng như là bộ xử lý trung tâm (cái kết nối với internet), có nghĩa là tín hiệu sẽ có thể nhảy từ thiết bị này qua thiết bị khác. Một mạng tới có thể kết nối tối đa 232 thiết bị.

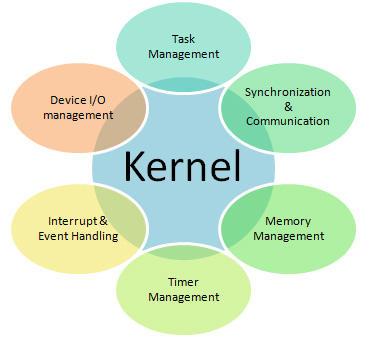
Z-Wave không phải là hệ thống mở và do đó, nó chỉ dành cho khách hàng của Zensys và Sigma Designs. Mặc dù điều này ban đầu có vẻ như là một hạn chế, nhưng nó thực sự là một trong những thế mạnh lớn nhất của giao thức này. Một trong những lợi thế quan trọng nhất của một hệ thống khép kín là an toàn. Mỗi mạng Z-Wave và các sản phẩm của mạng đều có ID duy nhất được sử dụng để liên lạc với trung tâm của bạn, và ID này bổ sung thêm một mức độ an toàn khác vượt qua mức mã hóa AES-128.

## 2.5 Giới thiệu về hệ điều hành thời gian thực RTOS

Hệ điều hành thời gian thực được thiết kế ra cho các nhiệm vụ đặc biệt. Các ứng dụng cần được thực thi với thời gian thật chính xác, các lỗi phát sinh cần được cô lập và xử lý nhanh chóng. Mọi sự chậm trễ, lỗi phát sinh không lường trước có thể khiến hệ thống bị đổ vỡ. Vì vậy, RTOS sử dụng trong những ứng dụng yêu cầu thời gian đáp ứng nhanh, chính xác về thời gian. Vi điều khiển có tài nguyên rất giới hạn. Do đó, hệ điều hành này chỉ tập trung vào một số ít các tính năng. Chúng cố gắng tối ưu tối đa số luồng, bộ lập lịch và các tác vụ (task) trên một hệ thống cỡ nhỏ.



FreeRTOS là một hệ điều hành nhúng thời gian thực (Real Time Operating System) mã nguồn mở được phát triển bởi Real Time Engineers Ltd, sáng lập và sở hữu bởi Richard Barry. FreeRTOS được thiết kế phù hợp cho nhiều hệ nhúng nhỏ gọn vì nó chỉ triển khai rất ít các chức năng như: cơ chế quản lý bộ nhớ và tác vụ cơ bản, các hàm API quan trọng cho cơ chế đồng bộ. Nó không cung cấp sẵn các giao tiếp mạng, drivers, hay hệ thống quản lý tệp (file system) như những hệ điều hành nhúng cao cấp khác. Tuy vậy, FreeRTOS có nhiều ưu điểm, hỗ trợ nhiều kiến trúc vi điều khiển khác nhau, kích thước nhỏ gọn (4.3 Kbytes sau khi biên dịch trên Arduino), được viết bằng ngôn ngữ C và có thể sử dụng, phát triển với nhiều trình biên dịch C khác nhau (GCC, OpenWatcom, Keil, IAR, Eclipse, …), cho phép không giới hạn các tác vụ chạy đồng thời, không hạn chế quyền ưu tiên thực thi, khả năng khai thác phần cứng. Ngoài ra, nó cũng cho phép triển khai các cơ chế điều độ giữa các tiến trình như: queues, counting semaphore, mutexes.



Các ứng dụng không cần dùng RTOS:

* Ứng dụng đơn (ứng dụng chỉ có 1 chức năng)
* Ứng dụng có vòng lặp đơn giản
* Ứng dụng <32kB

Nếu ứng dụng của bạn mà kích thước chương trình lớn dần và độ phức tạp tăng lên thì RTOS sẽ rất hữu dụng trong trường hợp này, lúc đó RTOS sẽ chia các ứng dụng phức tạp thành các phần nhỏ hơn và dễ quản lý hơn.

Các lợi ích của việc dùng RTOS:

* Chia sẻ tài nguyên một cách đơn giản: cung cấp cơ chế để phân chia các yêu cầu về bộ nhớ và ngoại vi của MCU
* Dễ debug và phát triển: Mọi người trong nhóm có thể làm việc một cách độc lập, các lập trình viên thì có thể tránh được các tương tác với ngắt, timer, với phần cứng (cái này mình không khuyến khích lắm vì hiểu được phần cứng vẫn sẽ tốt hơn nhiều)
* Tăng tính linh động và dễ dàng bảo trì: thông qua API của RTOS,…

# THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

## 3.1 Yêu cầu thiết kế

Từ các mục tiêu đặt ra của đề tài, phần cứng của hệ thống cần thực hiện bao gồm:

**Gateway Modbus Serial**:

* Có các cổng RS485, RS232, UART để giao tiếp với các thiết bị chuẩn Modbus có sẵn trên thị trường cũng như dễ dàng cài đặt, hiệu chỉnh.
* Có cổng Ethernet để triển khai mô hình Modbus TCP hoặc các dịch vụ trên lớp mạng để giao tiếp với IoT Platform trên server.
* Phần cứng vi điều khiển đủ mạnh để đảm bảo chạy các ứng dụng một cách ổn định.
* Đèn báo trạng thái, buzzer để thông báo cho người dùng.
* Hỗ trợ tối đa 4 cổng RS485 để quản lý được nhiều thiết bị.

**Gateway LoraWAN**:

* Tương thích với phần cứng máy tính nhúng.
* Khối nguồn để đảm bảo khối concentrator hoạt động ổn định.
* Kích thước nhỏ gọn, dễ lắp đặt.

**Gateway Z-wave:**

* Do Z-wave là mạng kết nối theo kiểu mess nên gateway khá đơn giản (phần cứng như một node) nên có thể tích hợp chung với Gateway LoraWAN.

**Các thiết bị đầu cuối:**

* Trong phạm vi đề tài sẽ thực hiện các bản thử nghiệm của một số thiết bị IoT thông dụng, trong đó gồm: Van nước điều khiển qua Z-wave, Z-wave node có sẵn các ngoại vi để tích hợp cho các thiết bị có sẵn (relay, cảm biến chuyển động), IoT Node dùng thu thập dữ liệu cảm biến thời tiết (ánh sáng, nhiệt độ, độ ẩm, …) hỗ trợ cả 2 giao thức là Modbus và LoraWAN để có thể triển khai trên các điều kiện khác nhau. Ngoài ra, đề tài cũng sử dụng thiết bị công nghiệp có sẵn (biến tần, plc, …) để thử nghiệm độ tương thích.
* Đảm bảo hỗ trợ ít nhất một trong ba giao thức của gateway để giao tiếp truyền tải dữ liệu.
* Hỗ trợ các ngoại vi cơ bản nhất (GPIO, OneWire, SPI, I2C, …) để điều khiển cũng như giao tiếp với các cảm biến thông dụng.
* Hỗ trợ các cổng debug để dễ dàng kiểm tra, hiệu chỉnh, sửa lỗi. Đèn led, nút nhấn để điều khiển bằng tay.
* Thiết kế khối nguồn hợp lý, linh kiện để đem lại mức tiêu thụ năng lượng thấp cho các thiết bị đầu cuối không dây, các thiết bị dùng nguồn pin.
* Nắm vững các quy tắc layout với các khối đặc thù như nguồn, vi điều khiển, cổng Ethenet, chuẩn RS485. Các quy tắc routing về đường dây, ăn-ten cho các thiết bị không dây.
* Sắp xếp linh kiện một cách hợp lý. Dễ dàng thay thế hoặc đo đạc trong quá trình kiểm tra hoạt động.

## Phân tích

Sau đây là các phương án cho từng thiết bị:

### 3.2.1 Gateway Modbus Serial

|  |  |  |  |
| --- | --- | --- | --- |
| Phương án | Mô tả hoạt động | Ưu điểm | Nhược điểm |
| Sử dụng STM32F407 làm vi điều khiển chính cho Gateway | Triển khai Modbus Protocol ở chế độ master.  Kết nối mạng nội bộ LAN để có thể gửi nhận dữ liệu tổng hợp được đến IoT Platform trên server  Các ngoại vi cơ bản để thông báo chức năng. | Cấu hình và ngoại vi cơ bản đáp ứng được yêu cầu.  Được dùng phố biến nên có sẵn các thư viện cơ bản, cộng đồng hỗ trợ động đảo. Dễ tìm thấy và mua ở thị trường trong nước Giá thành vừa phải khi so cùng cấu hình so với một số nhà sản xuất khác. | Giá thành cao |
| Sử dụng LAN8720A IC giao tiếp giữa vi điều khiển chính và Ethernet port | IC giao tiếp Ethernet, hỗ trợ tốc độ cao 10/100 Mbps. | Rẻ, nhỏ gọn, giao tiếp RMII dễ sử dụng.  Thư viện có sẵn cho STM32.  Đơn giản hoá trong việc xử lý các gói ở lớp PHY. | Cần vi điều khiển hỗ rợ, đủ mạnh để chạy các kết nối mạng. |
| MAX485, MAX3232 | IC chuyển tiếp UART-TTL sang RS485/RS232. | Rẻ. Chuẩn giao tiếp thích hợp cho truyền xa hơn TTL | Một số ic không chính hãng chạy không ổn định. |
| IC AP3211 | IC AP3211 có kích thước nhỏ gọn, dòng điện đầu ra tối đa 1.5 ampe. |  |  |

### Gateway Lora và Gateway Z-wave

|  |  |  |  |
| --- | --- | --- | --- |
| Phương án | Mô tả hoạt động | Ưu điểm | Nhược điểm |
| Sử dụng máy tính nhúng Rasberry Pi 3B+ làm khối điều khiển chính của gateway | Raspberry máy tính nhúng nhỏ được tích hợp nhiều phần cứng mạnh mẽ đủ khả năng chạy hệ điều hành và cài đặt được nhiều ứng dụng trên nó. Phiển bản Rasberry Pi 3+ sử dụng chip Broadcom BCM2837B0, quad-core @1.4Ghz, 1 GB RAM, 4xUSB 2.0, 10/100mb Ethernet, Wifi 802.11n và 40 GPIO pinout mở rộng. | Rẻ, cấu hình mạnh, dễ tìm và dễ mua ở thị trường trong nước. Được sử dụng phổ biến, cộng đồng hỗ trợ mạnh, nhiều thư viện, chạy được nhiều nền tảng hệ điều hành thông dụng như Linux, Raspbian nên dễ dàng trong việc triển khai các ứng dụng. | Không đủ nhiều ngoại vi như |
| Sử dụng RAK833, RAK831 Lora Concentrator làm thiết thu phát dữ liệu đến các thiết bị đầu cuối. | Lora Concentrator hỗ trợ nhận tín hiệu với 8 kênh đồng thời thu nhận tín hiệu. Do đó nó có thể làm một đảm bao thu nhận cho khoảng 500 thiết bị đầu cuối (theo nhà sản xuất). Hỗ trợ LoraWAN Stack 1.0.2 Hỗ trợ toàn bộ các băng tần của LoraWAN: 433MHz, 470MHz, 865MHz, 868MHz, 915MHz, 920MHz, 923MHz | Nhỏ gọn, giao tiếp SPI đơn giản, dễ lắp đặt.  Có sẵn thư viện từ Semtech, tương thích trên các nền tảng hệ điều hành phổ biến (OpenWRT, Raspbian,…) | Giá thành tương đối đắt và chưa phổ biến trong nước Cần Antenna nhạy để thu truyền được khoảng cách xa. |
| Sử dụng nguồn xung với IC AP3211. | IC AP3211 có kích thước nhỏ gọn, dòng điện đầu ra tối đa 1.5 ampe. Mạch thiết kế đơn giản. | Có chế độ bảo vệ quá dòng, bảo vệ quá áp, bảo vệ quá nhiệt độ trên IC AP3211(nhiệt độ vượt quá 1600C). Ngoài ra, chip tự động tắt hoàn toàn nếu điện áp vào dưới 3,8 volt và IC hoạt động nếu điện áp đầu vào trên 3.6 volt |  |
| Tích hợp ZM5202 Z-wave module | Chạy firmware Z-Wave controller giúp thiết lập mạng Z-Wave (thêm/xóa và điều khiển thiết bị Z-Wave) thông qua việc truyền nhận các gói tin serial với máy tính nhúng. Chạy firmware điều khiển các ngoại vi và giao tiếp với Z-Wave controller thông qua giao thức Z-Wave, đóng vai trò là thiết bị đầu cuối trong mạng. | Nhỏ gọn. Footprint không phức tạp, thuận tiện cho việc thiết kế. Số lượng GPIO đủ theo theo yêu cầu. Dễ dàng cấu hình và tùy chỉnh firmware. | Các pad giao tiếp được thiết kế nằm ở bụng chip gây khó khăn trong việc hàn thủ công cũng như tháo lắp gỡ lỗi. |

### 3.2.3 IoT Node:

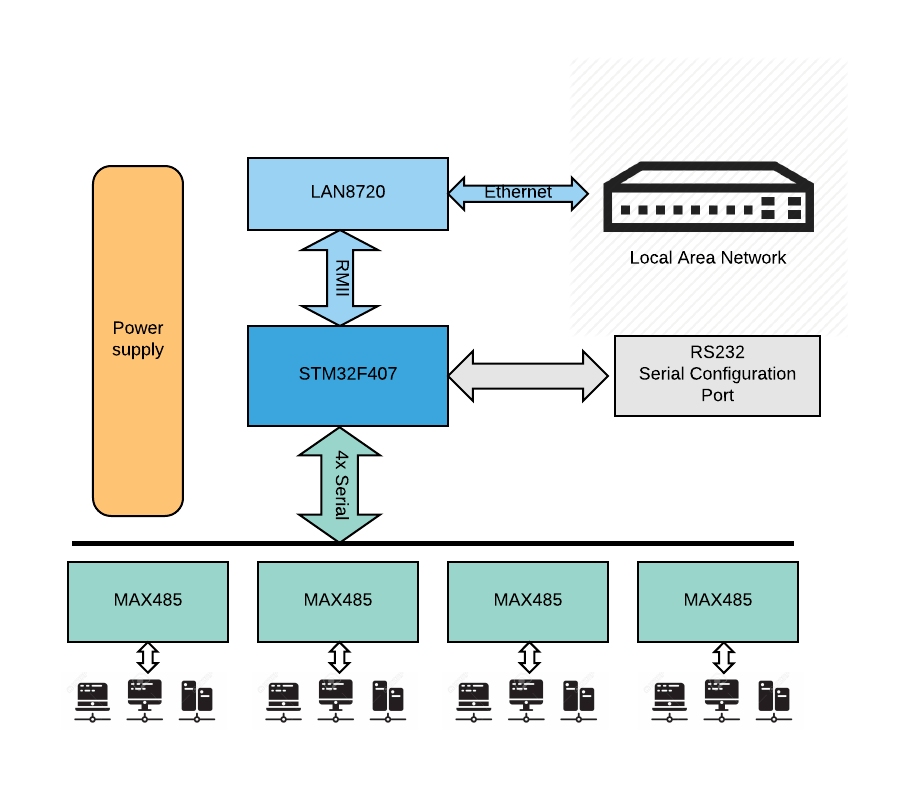
|  |  |  |  |
| --- | --- | --- | --- |
| Phương án | Mô tả hoạt động | Ưu điểm | Nhược điểm |
| Sử dụng STM32F103 làm vi điều khiển chính cho node | Thu thập dữ liệu từ 6 port IO có sẵn, bao gồm 3 Input coupler, 2 Analog và 1 Onewire port.  Triển khai Modbus Protocol ở chế độ slave.  Giao tiếp với module RAK để kết nối vào mạng LoraWAN như 1 node.  Các ngoại vi điều khiển bằng tay. | Cấu hình và ngoại vi cơ bản đáp ứng được yêu cầu.  Được dùng phố biến nên có sẵn các thư viện cơ bản, cộng đồng hỗ trợ động đảo. Dễ tìm thấy và mua ở thị trường trong nước Giá thành vừa phải khi so cùng cấu hình so với một số nhà sản xuất khác |  |
| Sử dụng EEPROM AT24C4 để lưu các lại các thông số tuỳ chỉnh | Lưu lại các giá trị tuỳ chỉnh, bao gồm các cài đặt liên quan đến từng port IO, các thông số của cổng Serial và các thông số của LoraWAN stack. | Giá thành rẻ, giao tiếp I2C đơn giản dễ sử dụng.  Số lần ghi xoá nhiều cũng như ổn định hơn so với việc dùng bộ nhớ Flash tích hợp sẵn trên vi điều khiển để lưu lại các giá trị |  |
| Sử dụng RAK811/ RAK4600 | Module LoraWAN cho phép kết nối vào mạng LoraWAN để truyền nhận dữ liệu. Trên module đã có sẵn LoraWAN stack 1.0.2. Trong đề tài sử dụng 2 loại module từ RAK, trong đó RAK4600 sẽ hỗ trợ thêm Bluetooth Low Energy. | Hỗ trợ AT Command thông qua serial để giao tiếp với vi điều khiển góp phần làm đơn giản hoá việc triển khai LoraWAN stack trên các Endnode device. | Firmware hoạt động chưa thực sự ổn định. Một vài tính năng liên quan đến LoraWAN vẫn còn đang phát triển. Một vài thông số liên quan đến năng lượng phát, thời gian truyền sóng, kích thước gói tin vẫn còn bị giới hạn và không thể mở khoá. |

1. Van nước

|  |  |  |  |
| --- | --- | --- | --- |
| Phương án | Mô tả hoạt động | Ưu điểm | Nhược điểm |
| Sử dụng STM32F103RCT6 làm vi điều khiển chính cho node | Vi điều khiển quản lý trạng thái và điều khiển 4 van nước thông qua Z-Wave. | Cấu hình và ngoại vi cơ bản đáp ứng được yêu cầu. |  |
| Tích hợp ZM5202 Z-wave module | Z-Wave module cung cấp các Command thông qua UART để kết nối Van nước vào mạng Z-Wave | (đã đề cập) |  |
| Sử dụng L293D làm driver điều khiển các Valve | IC cầu H điều khiển đóng mở van nước. | Kích thước nhỏ, chân cắm đơn giản cho việc tháo lắp và thay thế. | Giá thành còn cao, dòng tải tối đa nhỏ. |

## Sơ đồ khối tổng quát

### 3.3.1 Gateway Modbus Serial:

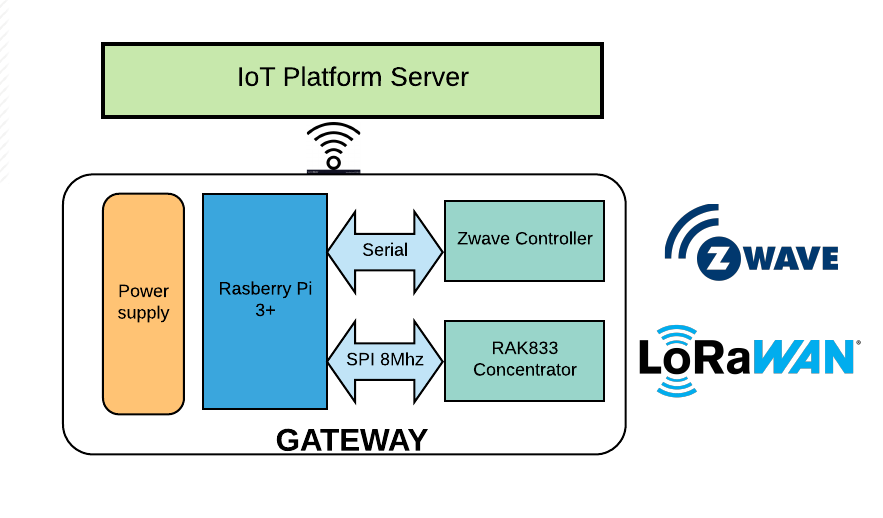


Khối vi điều khiển STM32F407VET6 đóng vai trò làm trung tâm xử lý dữ liệu, bao gồm các nhiệm vụ chính:

* Giao tiếp với MAX485 để tương tác với các thiết bị trọng mạng thông qua Modbus protocol.
* Giao tiếp với RS232 để cung cấp môi trường cài đặt với máy tính.
* Giao tiếp với LAN8720 để kết nối đến mạng nội bộ thông qua cổng giao tiếp Ethernet.

Các khối khác bao gồm: Khối nguồn để đảm bảo các mức điện áp ổn định cho các linh kiện hoạt động chính xác. MAX485 để chuyển tiếp qua lại tín hiệu UART và RS485. Khối LAN8720 để giao tiếp và xử lý các gói tin ở lớp PHY.

### 3.3.2 Gateway LoraWAN và Z-Wave



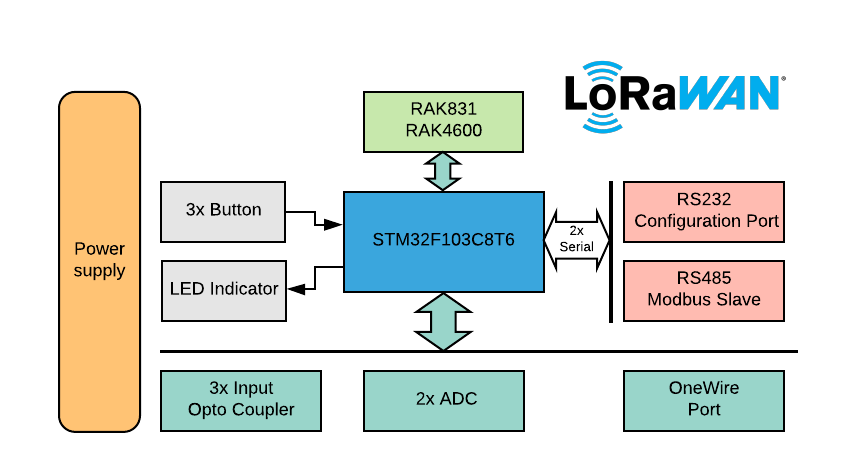
Khối trung tâm: máy tính nhúng Rasberry Pi 3+ có vài trò là trung tâm trung chuyển dữ liệuMáy tính nhúng sẽ chạy các driver để tương tác với các khối tập trung dữ liệu (bao gồm cả 2 giao thức Lora và Zwave) và các API cung cấp bởi IoT Platform Server để thực hiện các luồng dữ liệu từ các khối tập trung của từng giao thức lên IoT Server và ngược lại.

Zware Controller: là một module Zwave, có nhiệm vụ control các node trong mạng Zwave, kết nối với máy tính nhúng qua đường UART Serial.

RAK833 Concentrator: bộ tập trung dữ liệu của LoraWAN, có nhiệm vụ lắng nghe các gói tin Lora ở các dãy tần số và độ trải phổ cài đặt trước để gửi sang máy tính nhúng, đồng thời nhận gói tin từ máy tính nhúng để phát đến các node trong mạng.

Khối nguồn cung cấp: Cung cấp đúng mức điện áp và đảm bảo đủ dòng cần thiết cho các module, đặc biệt là khối RAK833 Concentrator. Theo yêu cầu từ nhà sản xuất, LoraWAN concentrator cần tối thiểu một bộ nguồn có thể tải điện áp 3V3 ở 1A.

### 3.3.3 IoT Node



Khối vi điều khiển giao tiếp

## 3.4 Sơ đồ mạch chi tiết:

### 3.4.2 Gateway LoraWAN và Z-Wave

#### Khối nguồn cung cấp:

Trong thiết kế đặt ra, board được sử dụng như một pack mở rộng để tương thích với khối GPIO Pinout của Rasperry. Do đó, việc sử dụng chung nguồn với Rasberry là một cách đơn giản và tối ưu cho thiết kế. Tuy nhiên cần phải xem xét các thông số:

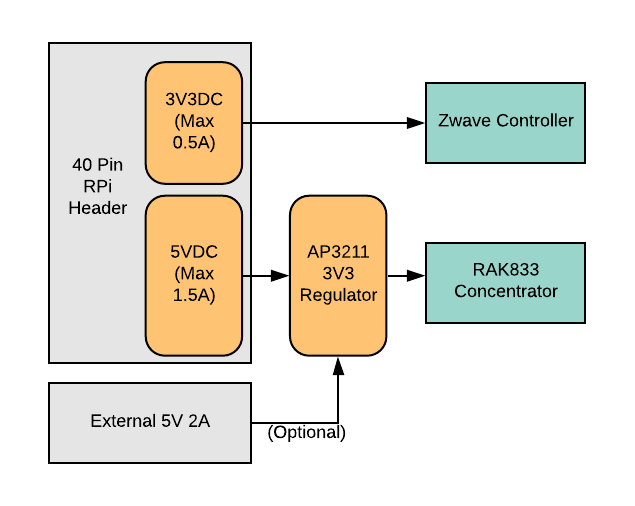
Đối với Rasberry:

* Pin 5V được nối trực tiếp với nguồn điện cung cấp chính của Rasberry và được bảo vệ bằng cầu chì tự phục hồi, dòng mong muốn có thể đạt đến 1.5A trong trường sử dụng nguồn đầu vào chuẩn của Rasberrypi.org (5V 2.5A).
* Pin 3V3 được hạ áp thông qua nguồn xung tích hợp trên board, có thể lên 0.5A. Tuy nhiên theo nhà sản xuất, Pin này không nên khai thác như một nguồn 3V3 chính cho các ứng dụng bên ngoài, thay vào đó các ứng dụng dùng 3V3 nên được hạ áp từ pin 3V3 bởi một mạch hạ áp ngoài.

Đối với RAK833 Concentrator: theo tài liệu tham khảo từ nhà sản xuất, cần tối thiểu một bộ nguồn 3V3/1A DC.

Đối với Zwave ZM5202: Dòng tối đa khi module thu phát tín hiệu là 48mA.

Sau khi thảm khảo những thiết kế có sẵn sử dụng trong phòng thí nghiệm. Cùng với những lựa chọn linh kiện trên, thiết kế của bộ nguồn cung cấp trong thiết bị Gateway có cấu trúc như sau:



* RAK833 Concentrator được cung cấp điện áp 3V3 từ nguồn xung của IC AP3211. Nguồn này sẽ được thiết kế để chịu tải tối đa 1,5A, được lấy đầu vào từ Pin 5V của Rasberry.
* Zwave Controller có dòng sử dụng thấp nên có thể sử dụng nguồn 3V3 cung cấp bởi Rasberry, phương án này góp phần đảm bảo nguồn của RAK833 không bị ảnh hưởng.

Dựa theo sơ đồ thiết kế mẫu của nhà sản xuất của AP3211 được lựa chọn làm IC giảm áp[7], bộ nguồn cung cấp trong thiết bị cũng có dạng thiết kế tương tự nhưng phải tính toán lại một số linh kiện để phù hợp hơn.

1. 

Điện áp đầu ra được điều chỉnh bằng cách dùng bộ chia áp lấy từ đầu vào và đưa vào chân “FB”. Bộ chia áp dùng trong sơ đồ mẫu là sử dụng điện trở để chia áp. Đây là phương pháp chia áp đơn giản và không khó khăn trong việc lựa chọn linh kiện nên ta áp dụng theo và điều chỉnh lại giá trị điện trở để có thông số thích hợp. Công thức tính điện áp đưa vào chân “FB”:

Theo datasheet của AP3211 thì điện áp trên chân “FB” có giá trị là 0.81. Từ đó, suy ra giá trị điện áp đầu ra được tính bằng công thức

Với VOUT mong muốn là 3.3 Volt. Ta suy ra được:

Trong thiết kế của bộ nguồn này, ta chọn R1 với giá trị 30 kΩ; ta chọn R2 với giá trị 10 kΩ. Vì cần thiết kế bộ nguồn có kích thước nhỏ gọn nên các linh kiện hầu hết kiểu chân dán (surface-mount technology – smt).

Cuộn cảm (inductor) có vai trò trong việc tinh chỉnh độ gợn sóng của dòng điện tại đầu ra và thường sử dụng trong nguồn xung. Giá trị của cuộn cảm phụ thuộc vào tần số hoạt động của nguồn xung (operating frequency), giá trị dòng điện (load current), độ gợn sóng của dòng điện mong muốn (ripple current), chu kì hoạt động (duty cycle). Giá trị cuộn cảm càng lớn thì càng giảm độ gợn sóng của dòng điện và điện áp tại đầu ra. Nhưng với giá trị càng lớn, kích thước thực tế của cuộn cảm cũng lớn. Vì vậy, có sự đánh đổi giữa việc chọn cuộn cảm cho nguồn xung. Trong hầu hết các ứng dụng, độ gợn sóng của dòng điện thường nằm trong khoảng giá trị từ 20% đến 30%. Và độ gợn sóng của dòng điện lớn nhất ở đầu ra khi nguồn hoạt động ở chế độ liên tục là 26%, từ đó giá trị của cuộn cảm được tính theo công thức sau:

Với VOUT = 3.3V; VIN = 5V; fSW = 1.4Mhz; IOUT = 1A. Ta suy ra được giá trị cuộn cảm là L1 = 3.08 . Tuy nhiên, giá trị của cuộn cảm này không phổ biến trong thực tế. Trong khi đó, cuộn cảm giá trị 3.3 phổ biến trên thị trường. Với giá trị cuộn cảm 3.3 và các thông số không đổi như: VOUT, VIN, fSW, và độ gợn sóng dòng điện tại đầu ra là 26%. Ta có dòng điện tối đa tại đầu ra là: 0.934A. Với giá trị này, bộ nguồn vẫn có thể cung cấp dòng điện cần thiết cho các trạng thái của thiết bị gateway.

Sau khi chọn lại giá trị của cuộn cảm, ta cần tính lại giá trị đỉnh của dòng điện để chắc chắn cuộn cảm đã chọn có thể chịu đựng được. Công thức tính giá trị đỉnh của dòng điện:

Với IOUT = 0.934A; VIN = 5V; VOUT = 3.3V; fSW = 1.4 Mhz; L1 = 3.3 . Ta suy ra được giá trị đỉnh của dòng điện là IPEAK ≈ 1.055A. Theo nhà sản xuất, giá trị dòng điện mà cuộc cảm chịu đựng gấp 1.5 lần giá trị IPEAK. Giá trị dòng điện lớn nhất mà cuộn cảm đã chọn có giá trị 1.75 A. Giá trị này hoàn toàn phù hợp với mức an toàn mà nhà sản xuất đưa ra.

Diode được sử dụng trong bộ nguồn này là diode Schottky. Có hai điểm chính khi lựa chọn diode Schottky: chọn diode có độ rơi áp thấp trên diode với điện áp thuận và có độ chuyển mạch nhanh (fast switching speed). Việc chọn diode theo hai tiêu chí này vì mục đích giảm mất mát công suất và tăng hiệu quả chuyển đổi. Ngoài ra, còn lưu ý đến các thông số như: điện áp đánh thủng của diode này phải lớn hơn điện áp đầu ra; giá trị điện áp trung bình qua diode lớn hơn giá trị đỉnh của dòng điện tại đầu ra (IPEAK). Trong tài liệu thiết kế bộ nguồn dùng IC AP3211 khuyến khích fchọn diode Schottly 1N5819. Các thông số đặt ra khi chọn diode đều thỏa mãn với diode 1N5819 [9].

Chức năng chính của các tụ bypass là để giảm gợn sóng và nhiễu của mạch DC.

Tụ bypass nối giữa VDD và GND làm các nhiễu tần số cao đi thẳng từ nguồn về đất.

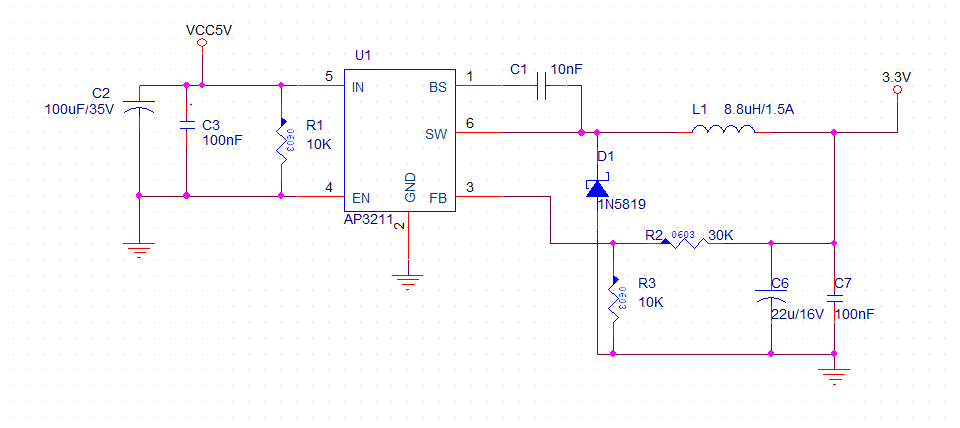
Tụ còn giúp bù điện áp khi có giảm áp đột ngột.

Một tụ lớn có giá trị khoảng 4.7uF đến 47uF để bù khi sụt áp, các nhiễu tần số thấp.

Một tụ nhỏ hơn, thường là 0.1uF để lọc những tần số cao hơn.

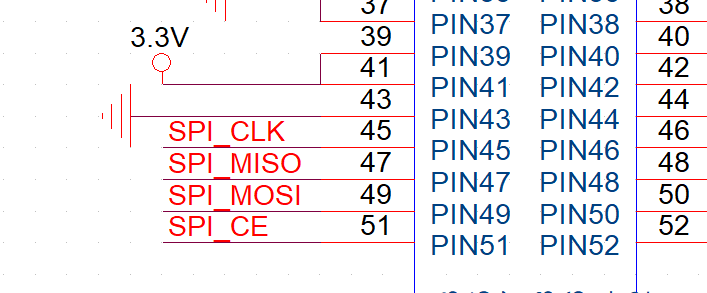
Và tụ nhỏ nữa khoảng 0.01uF để lọc tín hiệu tần số cao.

Ngoài ra, việc sử dụng nhiều tụ lọc còn giúp giảm điện trở nối tiếp ký sinh trong tụ điện (ESR – equivalent series resistance) do chúng được mắc song song với nhau.



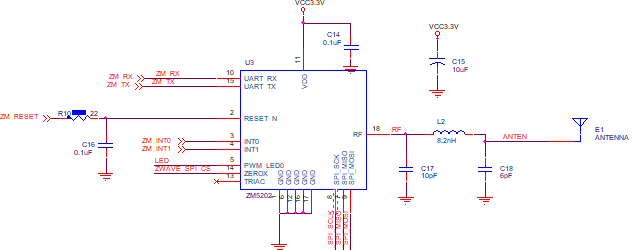
#### Các khối giao tiếp:

Khối cổng giao tiếp SPI của RAK833 Concentrator:

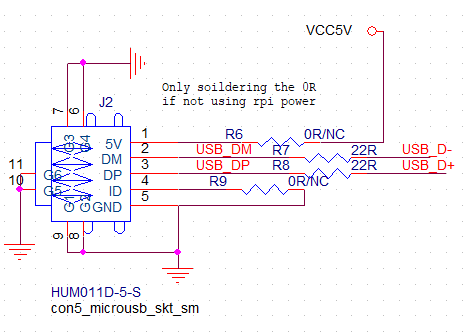


Module RAK833 được thiết kế dạng PCI Express Mini Card với nguồn và các pin chức năng được chỉ định tại một số pin chính trên khe PCI-E. Sau quá trình đọc tài liệu từ nhà sản xuất cũng như giao diện khe PCI-E, các chân quan trọng cần kết nối với máy tính nhúng dựa trên giao thức SPI bao gồm: SPI\_MOSI, SPI\_MISO, SPI\_CLK, SPI\_CE và một chân reset cho module khi khởi động.

Khối cổng giao tiếp UART của Zwave Controller:

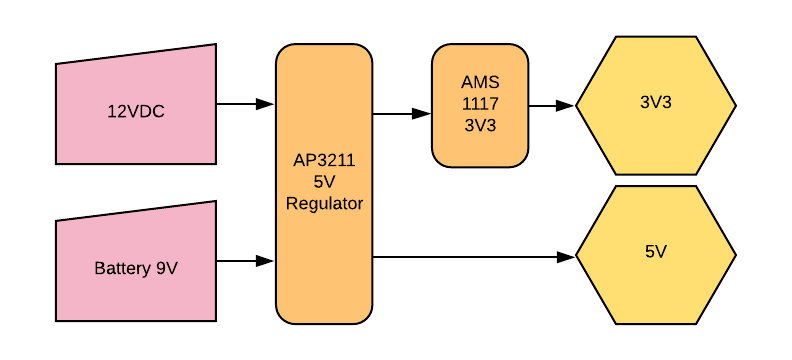


Ngoài ra còn có cổng USB để có thể dùng nguồn ngoài thay thế cho nguồn từ Rasberry, bên cạnh đó RAK833 hỗ trợ giao tiếp thư 2 thông qua USB ở chế độ device, phương án dự phòng khi giao tiếp thứ nhất qua SPI có vấn đề.



### 3.4.3 IoT Node

#### Khối nguồn cung cấp:



Khối nguồn cần đảm bảo các mức điện áp:

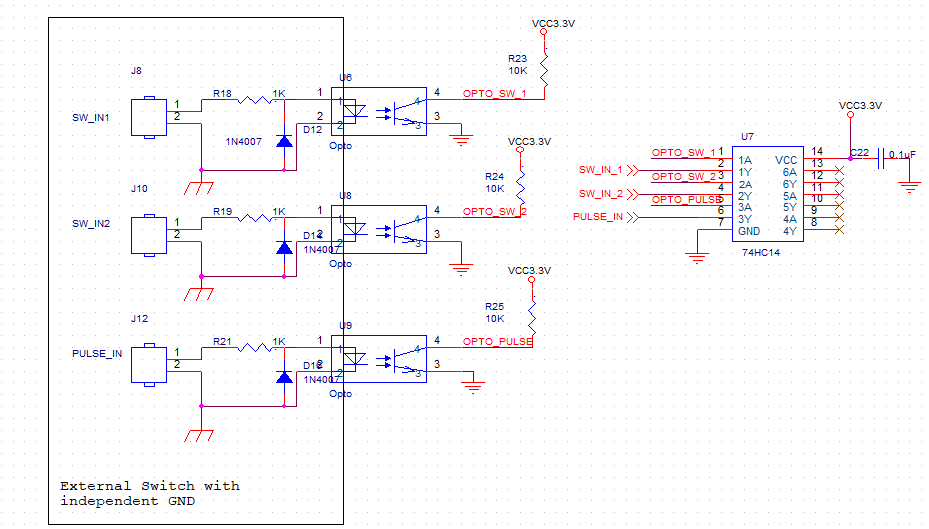
* Điện áp 5V: Là đầu vào cho bộ nguồn 3V3 và các cảm biến bên ngoài, có đầu vào là 12V từ adapter hoặc Pin, do mức độ chênh áp cao và dòng điện theo ước tính có thể lớn (0.5A-1A) tuỳ theo cảm biến bên ngoài nên sẽ được triển khai bằng bộ nguồn xung với IC chính là AP3211, các thông số về tụ, cuộn cảm và trở feedback có phần thay đổi so với AP3211 ở chế độ áp ra 3V3.
* Điện áp 3V3: là nguồn chính cung cấp cho vi điều khiển và các IC chức năng trên board (STM32, RAK811, RAK4600, EEPROM, MAX485, ...) có tổng dòng tiêu thụ không cao, tuy nhiên lại cần nguồn có độ ổn định nên sẽ được triển khai bằng AMS1117 3V3, IC này có dòng tối đa 0.8A và có mức drop out voltage nhỏ (đầu vào chấp nhận điện áp từ 4.75V – 15V), ngoài ra thiết kế đơn giản, chỉ yêu cầu 1 tụ lọc ở chân ouput để cải thiện chất lượng đầu ra nên rất thích hợp trong trường hợp này. Bên cạnh đó cần phải cân nhắc các thiết kế tụ lọc, tụ bypass, tụ decoupling cho các IC để đạt được trạng thái nguồn ổn định nhất.



#### Các cổng thu thập tín hiệu:

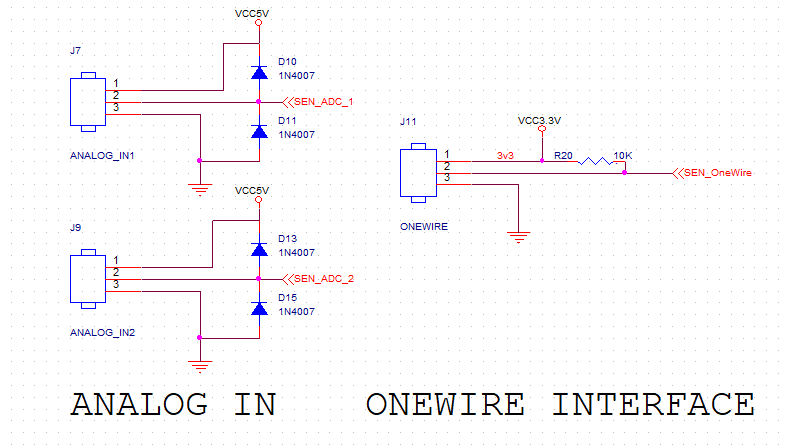
Trong yêu cầu đặt ra, mạch bao gồm:

* Các cổng input đầu vào cách ly bằng opto:



Các Opto được phân cực bởi các tín hiệu bên ngoài với độc lập với mức tín hiệu GND trong mạch, qua opto, các xung cao áp, các nguồn nhiễu, quá áp bên ngoài sẽ không làm ảnh hưởng đến mạch bên trong. Các Diode mắc ngược góp phần bảo vệ opto trong trường hợp mắc ngược đường tín hiệu. Ngoài ra để hạn chế các gai tín hiệu không mong muốn làm sai lệch trong quá trình đọc hoặc đo xung, IC chuyển mạch Smitt-Trigger 74HC14 được dùng //thông tin về nó xách qua phụ lục.

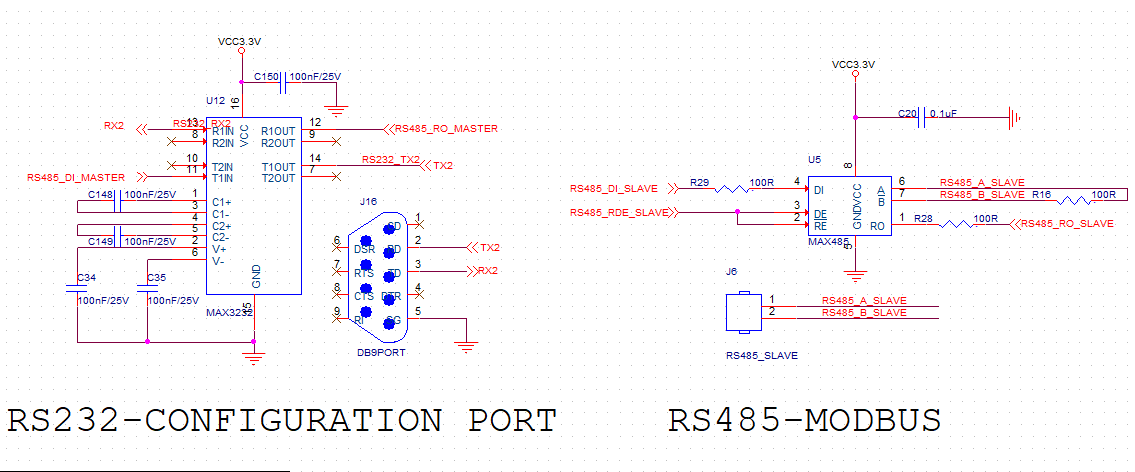
* Cổng đầu vào tín hiệu tương tự Analog và cổng dành cho OneWire:



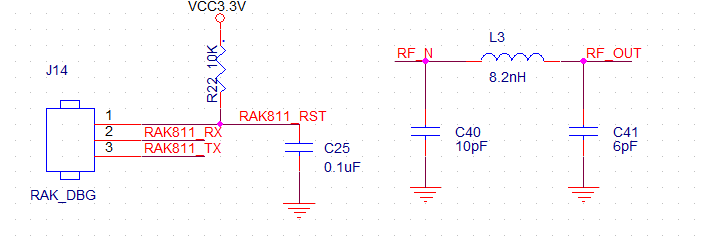
Mạch Analog in kèm bảo vệ quá áp và mạch cho các cảm biến onewire

#### Các khối giao tiếp:

Khối giao tiếp Serial cho RS485 chạy modbus và RS232 cho việc kết nối máy tính:



Đối với 2 module RAK, giao tiếp Serial được sử dụng để giao tiếp với vi điều khiển nên mạch tương đối đơn giản, ngoài ra cần chú trọng đến Antenna, mạch phối hợp trở kháng cho Antenna và các tụ lọc nguồn.



Zwave node: Module Zwave ZM5202 nhỏ gọn, phần lõi bao gồm vi điều khiển có lõi là 8051 chạy stack Zwave, qua đó có thể phát triển ứng dụng dựa trên vi điều khiển giao tiếp với module Zwave hoặc phát triển các ứng dụng trực tiếp lên module

Theo những phân tích trên, trong giới hạn đề tài, sẽ thực hiện cả 2 trường hợp

Zwave node: Bao gồm một board mạch nhỏ gồm module Zwave và các cổng nhập xuất cơ bản, cổng Relay, board được thiết kế nhỏ gọn

Valve Controller:

# THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM (NẾU CÓ) => có

## 4.1 Yêu cầu

Dựa trên mục tiêu đề tài, cần xây dựng nền tảng phần mềm đảm bảo các tính năng đặt ra, cũng như khả năng phát triển, kiểm tra gỡ lỗi và tuỳ chỉnh cao. Bên cạnh đó việc sử dụng nền tảng FreeRTOS trong việc xây dựng phát triển phần mềm giúp đơn giản và dễ dàng triển khai đa tác vụ hơn trên nền vi điều khiển.

### 4.1.1 Gateway Modbus Serial:

Các yêu cầu cơ bản đặt ra cho Gateway:

* Gateway phải chạy được Modbus Stack ở chế độ Master để thu thập, thay đổi, quản lý dữ liệu cho các thiết bị đầu cuối. Đồng thời để nâng cao tối đa số lượng thiết bị trong mạng, 4 cổng RS485 trên board đều phải được khai thác, có thể xem như Gateway quản lý 4 mạng Modbus riêng biệt.
* Gateway được kết nối đến IoT Platform thông qua MQTT, trong đó mỗi gateway sẽ có địa chỉ ID định danh riêng trong mạng, đồng thời đảm bảo 2 luồng dữ liệu chính là downlink (gói tin yêu cầu từ IoT Platform) và uplink (gói tin phản hồi yêu cầu và gói tin đồng bộ).
* Gateway phải có chế độ tuỳ chỉnh, bao gồm: các thông số liên quan đến địa chỉ IP của Gateway trong mạng (Static IP, Netmask, Default Gateway), Gateway ID, MQTT Server IP và các thông số để điều chỉnh cho mạng Modbus: Baudrate, Stopbit, Parity, ...
* Các cơ chế thông báo lỗi, đèn LED và Buzzer thông báo lưu lương và lỗi trong quá trình chạy thực nghiệm.

### Gateway LoraWAN và Z-Wave

Các yêu cầu cơ bản đặt ra cho Gateway:

* Gateway LoraWAN làm trạm trung chuyển các gói tin LoraWAN từ các LoraWAN node đến LoraWAN Server.
* Gateway Zwave giao tiếp với máu tính nhúng thông qua serial, qua đó luồng dữ liệu của Zwave sẽ được quản lý bởi Z-Wave Command Handler trên IoT Platform.
* Các cơ chế đảm xử lý, báo lỗi khi các giao thức bị lỗi.

### IoT Node

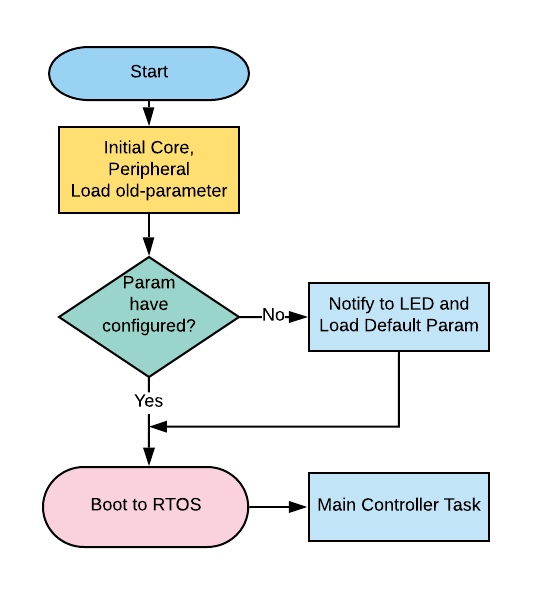
Các yêu cầu cơ bản đặt ra cho IoT Node:

* Xây dựng node thu thập dữ liệu môi trường, node phải hỗ trợ các kết nối ngõ vào cho các cảm biến thu thập thông dụng như GPIO Input, Analog Converter, Onewire. Node xử lý và chuẩn hoá các giá trị và thông tin liên quan để có thể gửi về trạm Gateway.
* Trong nhiều trường hợp, các cổng kết nối sẽ thu thập dữ liệu từ các cảm biến khác nhau, do đó phải có các profile riêng và cơ chế cài đặt cho từng cảm biến.
* Hỗ trợ Modbus Serial ở chế độ Slave, chuẩn hoá các giá trị thu thập được trên những thanh ghi đã được định danh trước.
* Hỗ trợ các Command để giao tiếp với module RAK, bao gồm cài đặt, kết nối và gửi nhận dữ liệu vào mạng LoraWAN.
* Hỗ trợ các command để kết nối với App trên máy tính để dễ dàng thay đổi chính sửa cũng như quản lý node.

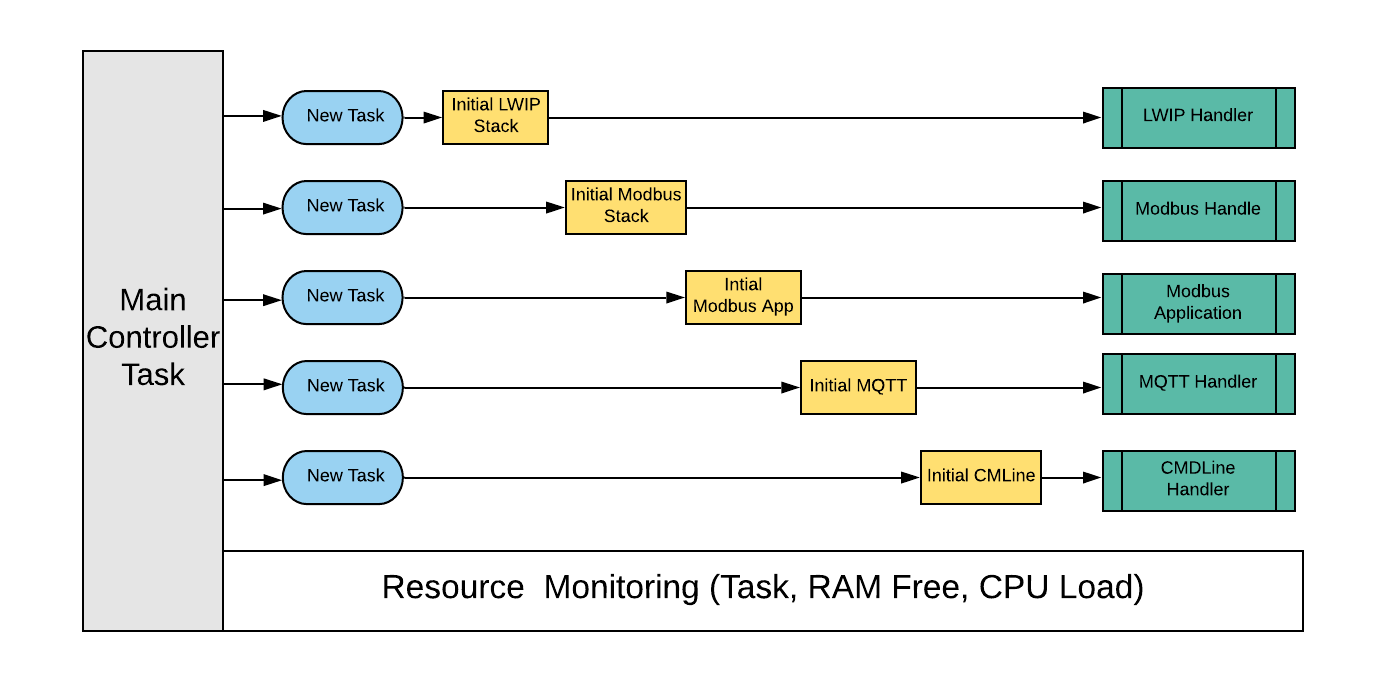
Valve Controller// ít quá thì thêm vào

# Phân tích

### 4.2.1 Phần mềm trên Gateway Modbus Serial



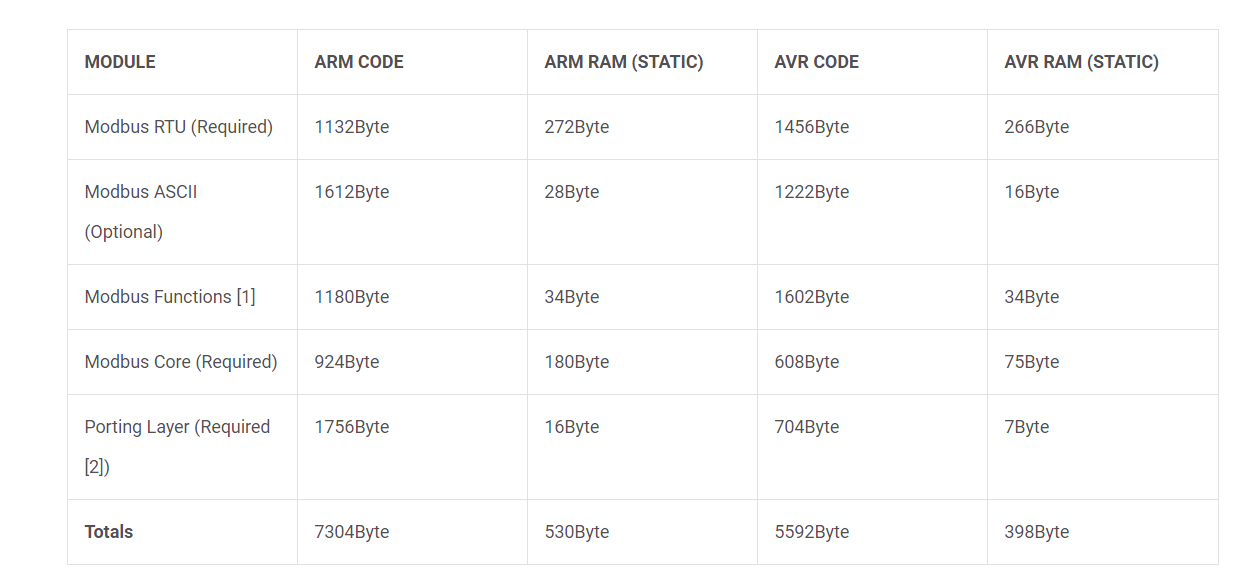
Quá trình khởi động của Gateway: vi điều khiển STM32 sẽ cài đặt các thông số hoạt động cơ bản và khởi tạo các ngoại vi, sau đó sẽ tìm đến vùng nhớ lưu trữ thông số cài đặt (trên vùng nhớ Flash hoặc EEPROM). Nếu các thông số đã được thiết đặt, gateway sẽ tiến hành load các cài đặt và bắt đầu boot vào kernel của RTOS, ngược lại sẽ báo trạng thái chưa cài đặt trên LED và tiến hành load các thông số mặc định.



Sau khi boot thành công, gateway sẽ nhảy vào Main Controller Task – đây là task chính của gateway, bao gồm các cơ chế tạo, quản lý các Task sau này. Cũng trên Main Controller Task, các cơ chế monitoring tài nguyên hệ thống được triển khai và gửi lên người dùng thông qua Serial RS232.

Để đảm bảo việc khởi động các protocol và các application một cách chính xác (MQTT chỉ chạy khi LWIP đã kết nối mạng, Modbus Application sẽ chạy khi Modbus Stack sẵn sàng,..), các Task phải được khởi tạo một cách có tuần tự. Các task mới chỉ được khởi động khi được cho phép bởi Main Controller Task cũng như phải reponse trạng thái khởi động để Main Controller Task quyết định các Task tiếp theo được khởi tạo.

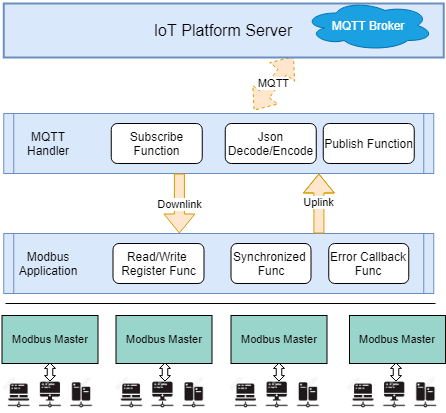
Modbus Stack: Trong đề tài, modbus stack được triển khai dựa trên một thư modbus có sẵn là freemodbus, đây là một gói thư viện miễn phí, hỗ trợ các chế độ RTU/ASCII và TCP đối với các thiết bị nhúng kết nối được mạng, hỗ trợ nhiều nền tảng vi điều khiển nên việc porting không quá phức tạp, bên cạnh đó, freemodbus còn được thiết kế để chạy trên các hệ điều hành thời gian thực thông dụng, việc triển khai freemodbus không tốn nhiều tài nguyên (1 timer + 1serial port), 300bytes ram và 5kb tuỳ theo các function được triển khai.



1. Tuy nhiên, trong yêu cầu đặt ra, cần tuỳ biến thư viện để có thể chạy tổng cộng 4 master để có thể đạt được khoảng 1000 thiết bị.
2. Command Line: Giao diện command line thông qua Serial để cài đặt các thông số cho gateway.

Lưu đồ giải thuật chung của modbusmqtt gateway

Thêm task Message giữa các task



MQTT Handler: Trong quá trình khởi động, MQTT Handler sẽ khởi tạo dịch vụ MQTT dựa trên lớp TCP đã được cung cấp bởi thư viện LWIP khởi tạo trước đó. Sau đó, MQTT Handler sẽ tiến hành Subscribe vào Topic định danh trước (thống nhất với IoT Platform Server và được lưu lại thông qua bộ nhớ flash), dịch vụ MQTT sẽ tạo ra callback mỗi khi có gói tin được gửi đến Topic, trong đề tài, gói tin đó sẽ là gói tin Downlink Request từ phía Server để yêu cầu Gateway trả về các thông tin từ các thiết bị mà nó quản lý, gói tin được chuẩn hoá theo JSON, gói tin yêu cầu này sẽ được gửi qua Modbus App qua cơ chế hàng đợi Queue của FreeRTOS. Ngoài ra, MQTT Handler sẽ chờ các phản hồi Uplink Reponse từ Modbus App để tiến hành gửi các giá trị của thiết bị được yêu cầu thông tin ngược lại lên Server.

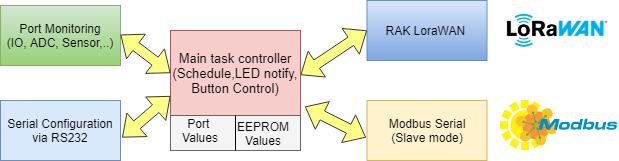
Modbus Application: Sau khi Modbus Stack đã sẵn sàng, Modbus Application sẽ được khởi tạo. Modbus App sẽ kiểm tra hàng đợi Queue từ khối MQTT handler và gọi các hàm cung cấp từ Modbus Stack để truy cập các thông tin liên quan đến giao thức modbus, bao gồm:

1. Read Input Register (0x04)
2. Read Holding Registers (0x03)
3. Write Single Register (0x06)
4. Write Multiple Registers (0x10)
5. Read/Write Multiple Registers (0x17)
6. Read Coils (0x01)
7. Write Single Coil (0x05)
8. Write Multiple Coils (0x0F)
9. Read Discrete Inputs (0x02)
10. Report Slave ID (0x11)

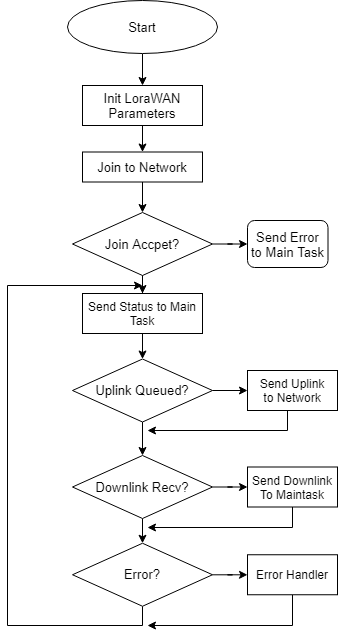
Các thông tin trả về sẽ được Modbus Stack trả về và gửi ngược lên đường Uplink Respose, đồng thời sẽ báo lỗi sang Main Controller Task nếu có bất kỳ gói tin nào lỗi.

ioT node:

Quá trình khởi động của IoT node tương tự như gateway, bằng các khởi động ở chế độ non OS, vi điều khiển sẽ tiến hành khởi tạo các ngoại vi cơ bản và load các cài đặt thông số đã lưu trữ trước, sau đó boot vào kernel của FreeRTOS để chạy chế độ multitasking.



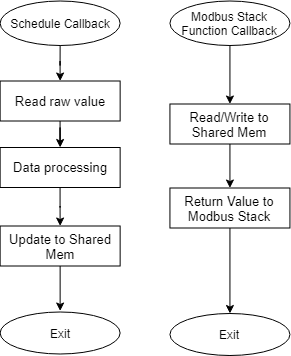
1. Main task controller: Bao gồm các cơ chế đặt lịch, cài đặt cho các task xung quanh, quản lý về bộ nhớ dùng chung để cung cấp các vùng thanh ghi lưu dữ liệu hoạt động của node mà các task khác như Modbus và RAK có thể truy cập. Main Task còn cung cấp các giao diện tương tác bên ngoài như Led và nút nhấn.
2. RAK LoraWAN: Đây là Task quản lý các truyền nhận dữ liệu thông qua giao thức LoraWAN mà trọng điểm chính là giao tiếp giữa vi điều khiển và module RAK qua giao diện AT Command đặt ra bởi RAK Wireless. Do đó việc chuyển đổi dùng 2 lựa chọn RAK4600 và RAK811 không làm thay đổi nhiều source code.



Do các yêu cầu về năng lượng ở trường hợp sử dụng pin cũng như đảm bảo các nguyên tắc của LoraWAN Limitation, [Trích dẫn vào] Nên =>> copy

//đang vẽ

1. Modbus Slave: như đã đề cập ở phần trên, thư viện FreeModbus cũng hỗ trợ stack Modbus cho các thiết bị ở chế độ Slave. Thông qua thư viện, khi nhận được yêu cầu từ Master, Stack sẽ gọi các hàm tương ứng để tương tác với các thanh ghi chức năng được cung cấp bởi Main Task.
2. Port Monitoring: Quản lý các port tương tác của Node sang các ngoại vi cảm biến bên ngoài, bao gồm 3 Input, 2 kênh đọc Analog và một Port giao thức One Wire. Task sẽ đọc các giá trị tuỳ theo khoảng thời gian đặt ra của từng loại cảm biến cũng như xử lý dữ liệu thu được tuỳ theo profile cho từng cảm biến.



Thiết kế thực hiện phần mềm trên máy tính

# KẾT QUẢ THỰC HIỆN

## Phần cứng

* + 1. LoraWan và Zwave Gateway

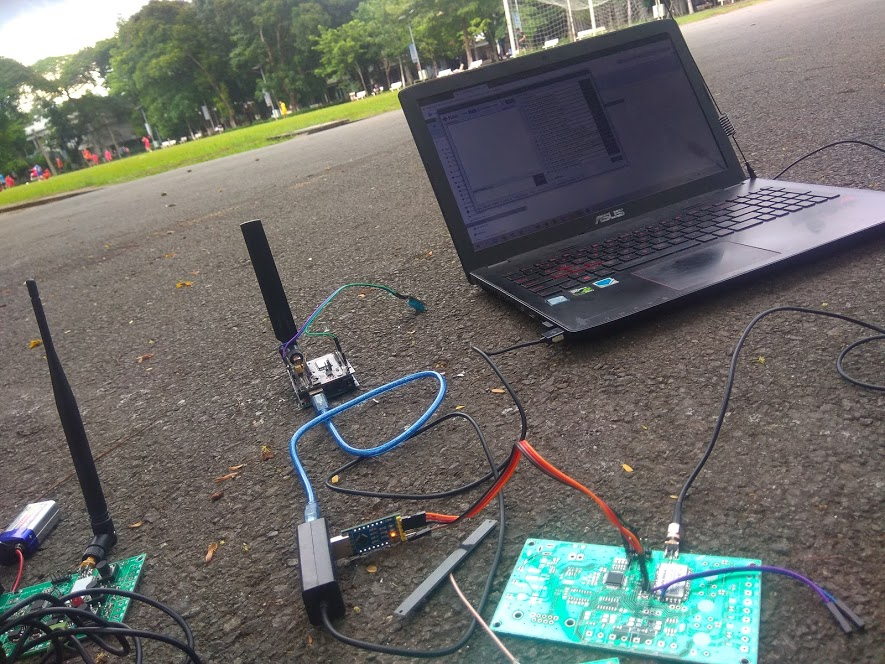


// Đo nguồn

//

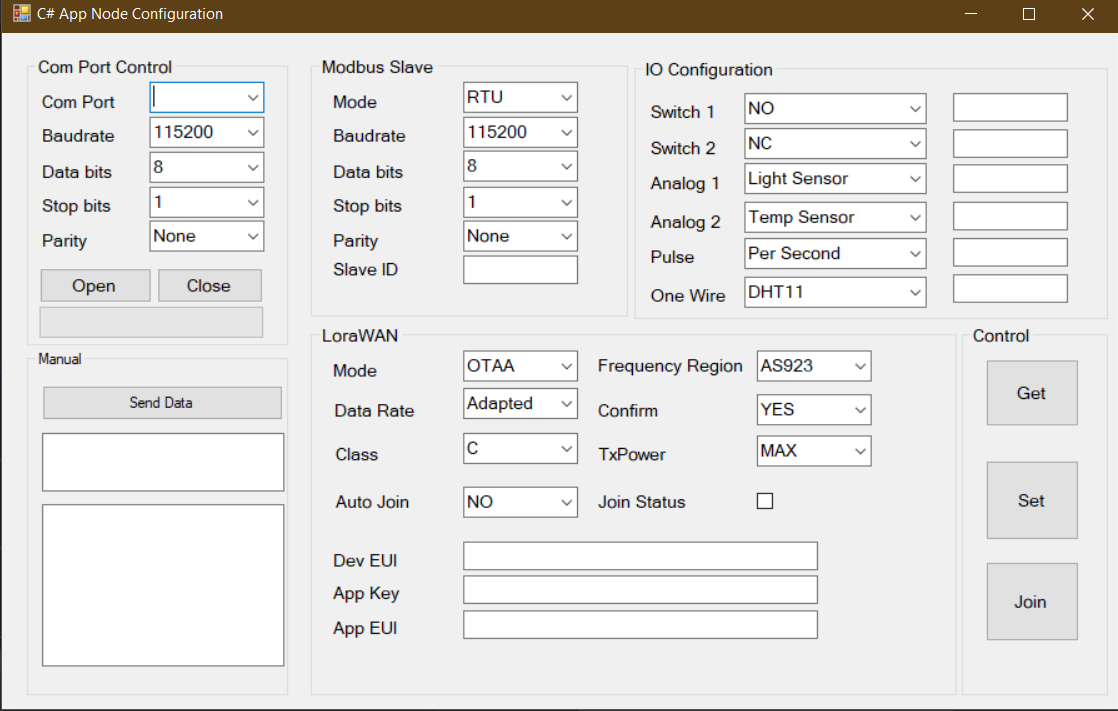
* + 1. IoT Node

Đo đạt khoảng cách



* + 1. Zwave node
    2. Valve nước

App



Trong phần này, sinh viên mô tả:

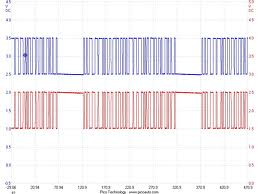
* Trình bày **cách thức đo đạc, thử nghiệm** 
  + Ghi rõ các thiết bị sử dụng và sơ đồ kết nối trong việc thử nghiệm
  + Ghi rõ các phần mềm sử dụng trong việc viết và thực thi chương trình
  + Ghi rõ cách bước tiến hành thử nghiệm (phần cứng và phần mềm)
* Trình bày số liệu đo đạc
  + Thực hiện thu thập số liệu trong nhiều trường hợp
  + Ghi rõ số liệu đo đạc thu được dưới hình thức bảng biểu, đồ thị …
* **Giải thích và phân tích về kết quả thu được**
  + Cần giải thích rõ ràng số liệu thu được trên các bảng biểu, đồ thị, dạng sóng …
  + Phân tích các số liệu để biết kết quả đã thực hiện là phù hợp, đạt yêu cầu

Nếu những bảng số liệu và kết quả mô phỏng quá nhiều, sinh viên có thể trình bày đưa vào phần Phụ Lục.

Ví dụ về hình minh họa: (dùng chức năng **Insert Caption** để tạo liên kết cho Danh sách hình minh họa)



Hình 5‑1 Kết quả thi công



Hình 5‑2 Kết quả mô phỏng

Ví dụ về Bảng số liệu

Bảng 1 Thông số hệ thống

|  |  |  |  |
| --- | --- | --- | --- |
| Thông số 1 | Thông số 2 | Thông số 3 | Thông số 4 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Sinh viên tóm tắt những điều rút ra được từ kết quả đề tài, những kinh nghiệm có được sau khi thực hiện đề tài. **Ưu và khuyết điểm** của kết quả nghiên cứu đề tài cũng được trình bày trong mục này. Sinh viên cần so sánh với mục tiêu đặt ra trong chương 1.

## Hướng phát triển

Sinh viên trình bày hướng phát triển và khả năng ứng dụng của đề tài

# TÀI LIỆU THAM KHẢO

Trong mục này, sinh viên liệt kê những tài liệu đã tham khảo khi thực hiện đề tài luận văn. Những nội dung trình bày ở mục trên có tham khảo tài liệu thì sinh viên cần ghi chú bằng chỉ số (ví dụ [1], [2]). Chỉ số này cần tương ứng danh mục tài liệu tham khảo. Sinh viên xem thêm hướng dẫn cách viết trích dẫn kiểu IEEE.

Ví dụ:

1. Tống Văn On, “Thiết kế mạch số với VHDL & Verilog”, Nhà xuất bản Lao động Xã Hội, 2007.
2. Altera Corp., “SDRAM Controller for Altera’s DE2/ DE1 boards”, [www.altera.com](http://www.altera.com)

# PHỤ LỤC

Trong phần này, sinh viên có thể trình bày:

* Những kết quả nghiên cứu bổ sung mà trong phần Kết quả luận văn chưa trình bày hết.
* Phần mã nguồn chương trình, sinh viên cũng có thể trình bày trong mục này. Để ngắn gọn, sinh viên chỉ đưa những mã nguồn chính vào phần Phụ lục.
* Sơ đồ toàn mạch chi tiết

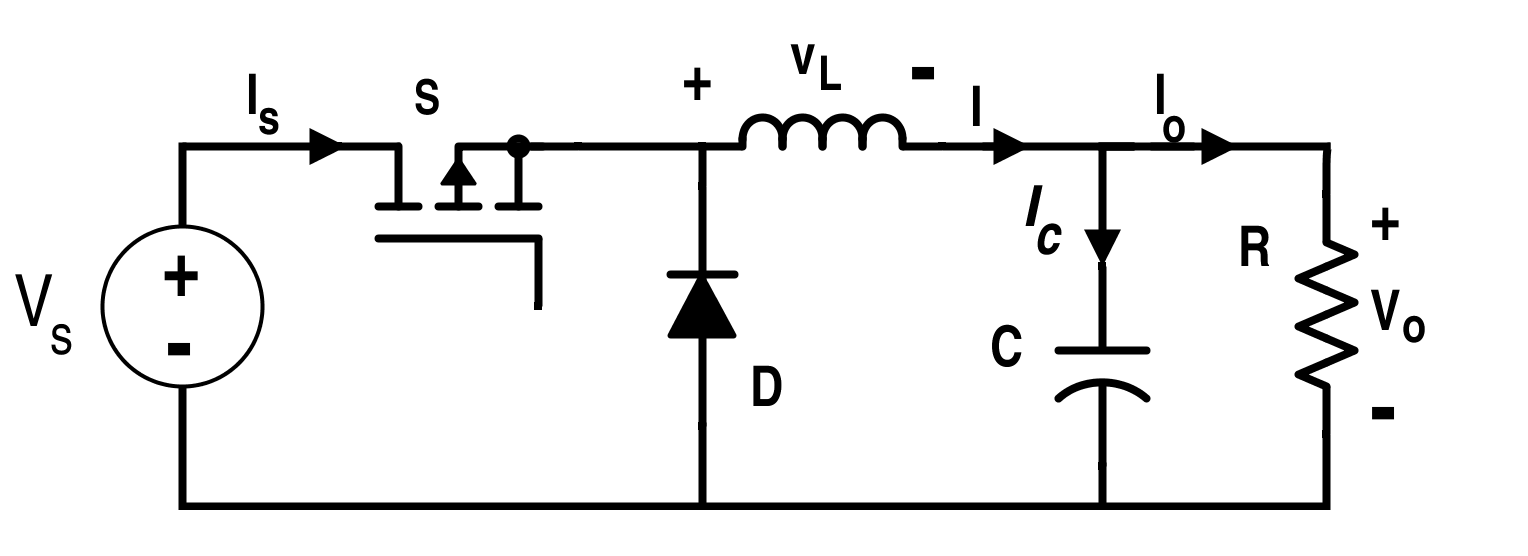
## 8.1 Bộ nguồn giảm áp – Buck Converter

Bất kỳ bộ nguồn Converter nào đều hoạt động ở một trong hai chế độ riêng biệt: chế độ liên tục và không liên tục. Ở chế độ liên tục, dòng điện luôn lớn hơn 0. Khi dòng điện trung bình chảy qua cuộn dây quá thấp do tải có điện trở lớn hoặc do tần số chuyển mạch thấp, bộ converter được xem hoạt động ở chế độ không liên tục.

Chế độ liên tục được sử dụng rộng rãi hơn vì có hiệu suất cao và tận dụng tốt những linh kiện chuyển mạch và linh kiện thụ động.

Ta giả sử rằng các cuộn dây và tụ điện đều là cuộn dây thuần và cuộn cảm thuần (tức không chứa điện trở ký sinh). Tuy nhiên, với điều kiện đó thì điện áp ngõ ra vẫn có những dao động nhỏ, gọi là ripple (gợn sóng). Các mạch converter tốt có gợn sóng rất nhỏ.

Để thuận tiện cho việc tính toán và phân tích mạch, chúng ta sẽ giả sử tải là thuần trở và điện áp ngõ ra không có gợn sóng, hay gọi là điện áp DC, có giá trị cố định.



Hình 55. Sơ đồ mạch Buck Converter đơn giản

Một mạch buck đơn giản gồm có:

- Nguồn Vs: ngõ vào, là nguồn DC hoặc nguồn AC đã được chỉnh lưu.

- Một Diode và một linh kiện chuyển mạch S (FET, BJT).

- Bộ lọc thông thấp LC

- Tải

