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: ThS. 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
TRƯỜNG ĐẠI HỌC BÁCH KHOA

----- ☆----

Số: _____ /BKĐT
Khoa: **Điện – Điện tử**
Bộ Môn: **Điện Tử**

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc.

----- ☆----

NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

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

2. NGÀNH: **ĐIỆN TỬ - VIỄN THÔNG** LỚP : DD16VT03
3. Đề tài: Thiết kế và xây dựng nền tảng IoT đa giao thức.
4. 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 Server.
5. Ngày giao nhiệm vụ luận văn: .....
6. Ngày hoàn thành nhiệm vụ: .....
7. 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 Bách Khoa! Suốt khoảng thời gian gần 4 năm qua đã giúp ta mạnh mẽ và trưởng thành hơn. Không muốn biết ta có được những gì, ta mất đi bao nhiêu, mà quan trọng tuổi trẻ của ta đã lớn lên cùng với Bách Khoa, đó là một khoảng thời gian thanh xuân không thể nào quên.

Không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Em xin gửi lời cảm ơn chân thành đến thầy Bùi Quốc Bảo đã tận tình chỉ dẫn, giúp đỡ và định hướng trong suốt thời gian làm đồ án và luận văn, tạo điều kiện thuận lợi để hoàn thành đề tài luận văn này. Bên cạnh đó xin cảm ơn quý thầy cô ở Khoa Điện – Điện Tử đã trang bị cho em những kiến thức nền tảng, bổ ích trong khoảng thời gian học tập tại trường.

Em xin gửi lời cảm ơn đến các anh, các bạn trong phòng thí nghiệm Bộ môn Điện tử đã hỗ trợ em rất nhiều trong thời gian qua.

Cuối cùng, xin gửi cảm ơn các thành viên trong Câu lạc bộ Nghiên cứu khoa học Khoa Điện – Điện tử đã truyền đạt cho em nhiều kiến thức và tinh thần khi em bắt đầu những bước chân đầu tiên vào lĩnh vực Điện tử.

Tp. Hồ Chí Minh, ngày tháng năm .

Sinh viên

TÓM TẮT LUẬN VĂN

Luận văn này trình bày về việc xây dựng nền tảng 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 của 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.

Firmware của Gateway và các thiết bị đầu cuối:

- Lập trình các chức năng và kết nối 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 ứng dụng các giao thức (TCP, MQTT,..) và các chuẩn đóng gói dữ liệu (JSON,…) để kết nối đến IoT Server.

Software:

- 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

DANH SÁCH HÌNH MINH HOẠ

DANH SÁCH BẢNG SỐ LIỆU

1. GIỚI THIỆU

1.1 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 của nó.

Hình 1: Mô hình IoT tổng quan.

1.2 Tình hình nghiên cứu trong và ngoài nước

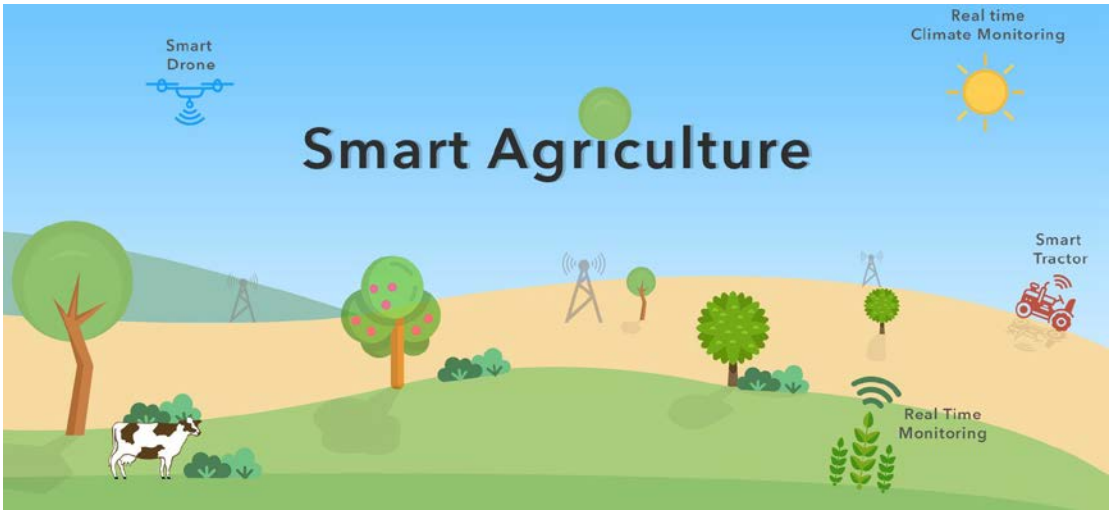
IoT hiện đang là động lực phát triển của hầu hết các lĩnh vực công nghệ. Các giải pháp IoT cho phép các doanh nghiệp cải thiến các hệ thống hiện có giúp nâng cao kết nối với khách hàng và đối tác. Một số ứng dụng IoT dẫn đầu xu hướng công nghệ hiện nay có thể kể đến như:

- 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 kỹ 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 2: Minh hoạ 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.

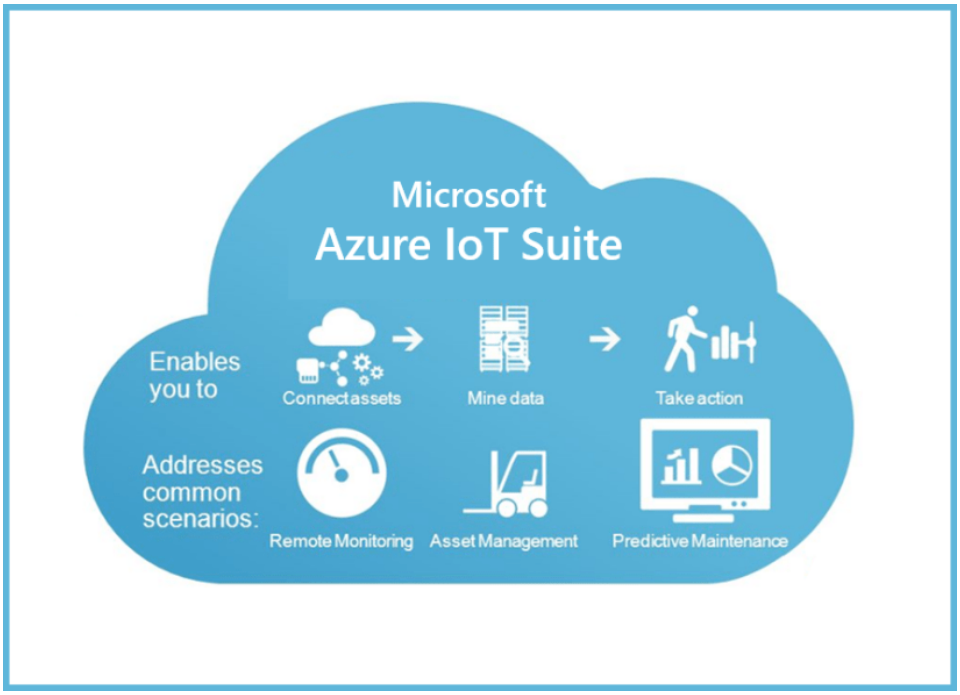


Hình 3: Minh họa nông nghiệp thông minh

- 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.

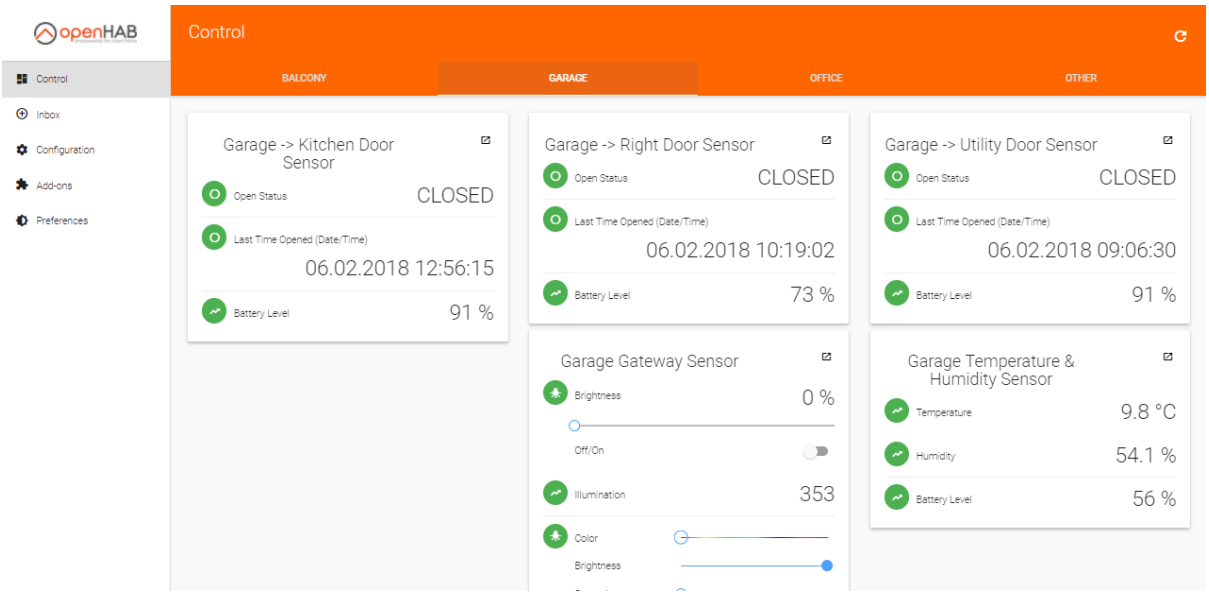
Hình 4: Minh họa vận chuyển thông minh.

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 5: Mô hình Microsoft Azure IoT.

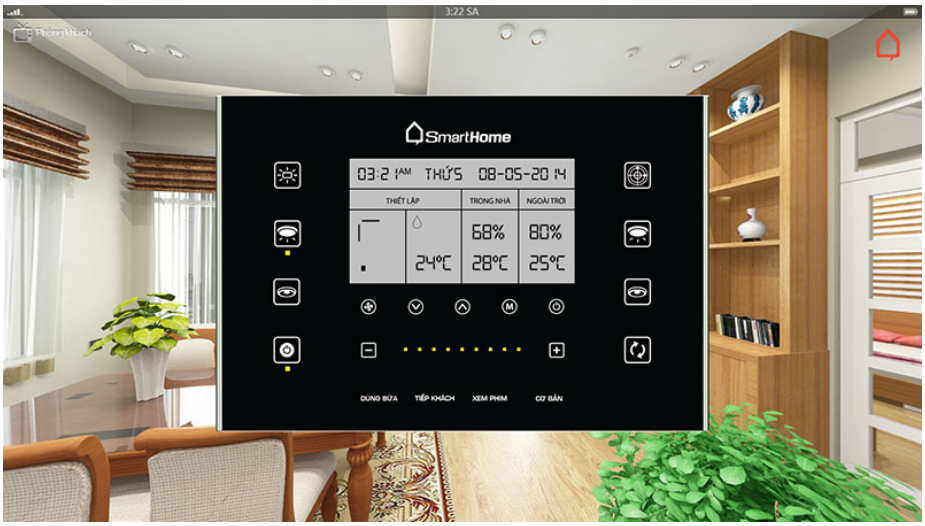
Một số nền tảng IoT mã nguồn mở như Kaa, ThingsBoard, OpenHAB thì không phải một nền tảng IoT hoàn chỉnh, mà chỉ tập trung vào giải pháp phần mềm hỗ trợ cho việc kết nối các thiết bị phần cứng.



Hình 6: 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,…

họ 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.



Hình 7: Smarthome by BKAV.

FPT Information System (FPT IS) là đơn vị về IoT trong các lĩnh vực giao thông, y tế, năng lượng… Trong đó, dự án Giao thông thông minh đang được triển khai thí điểm tại Thành phố Hồ Chí Minh là dự án tiêu biểu nhất ứng dụng IoT. Giải pháp kì vọng nâng cao năng lực quản lý, điều hành và chất lượng giao thông trên địa bàn các thành phố lớn, tiến tới phủ sóng trên cả nước. Nguồn dữ liệu từ các phương tiện giao thông công cộng, camera an ninh, … sau khi được phân tích sẽ giúp mô hình hóa giao thông thành phố, từ đó hỗ trợ các cơ quan trong việc quản lý, đánh giá, quy hoạch. Sở hữu nhiều nền tảng để xây dựng thành phố thông minh, như: Giao thông thông minh, eHospital, Hệ thống đo điện từ xa …, từ bệ phóng này, chiến lược kinh doanh của FPT IS là vận dụng những kinh nghiệm triển khai các dự án IoT thành công ở Việt Nam để tấn công thị trường quốc tế.

Qua do cho thấy, để đạt được tối ưu giá trị từ IoT, việc làm chủ một nền tảng IoT (IoT platform) là cần thiết. Nền tảng IoT 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. Như vậy, việc tự xây dựng một nền tảng IoT hoàn chỉnh, hỗ trợ từ phần cứng đến phần mềm, cho phép quản lý, điều khiển, và giám sát là mục tiêu cần thiết trong việc phát triển công nghệ IoT hiện nay.

1.3 Nhiệm vụ luận văn

Nhiệm vụ chính của luận văn là xây dựng một nền tảng IoT, trong đó sẽ chú trọng vào việc xây dựng các thiết bị đầu cuối, các Gateway thu thập dữ liệu theo các giao thức đặt ra và các phần mềm để kết nối các thiết bị đó với nền tảng ứng dụng trên Server (nền tảng ứng dụng trên Server được thực hiện bởi một sinh viên khác), cụ thể:

- Thiết kế và thi công phần cứng Gateway cho các giao thức Modbus Serial, LoRaWAN, Z-Wave.
- Thiết kế và thi công phần cứng các thiết bị đầu cuối cho từng giao thức Modbus Serial, LoRaWAN, Z-Wave.
- 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 nền tảng IoT trên Server.

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

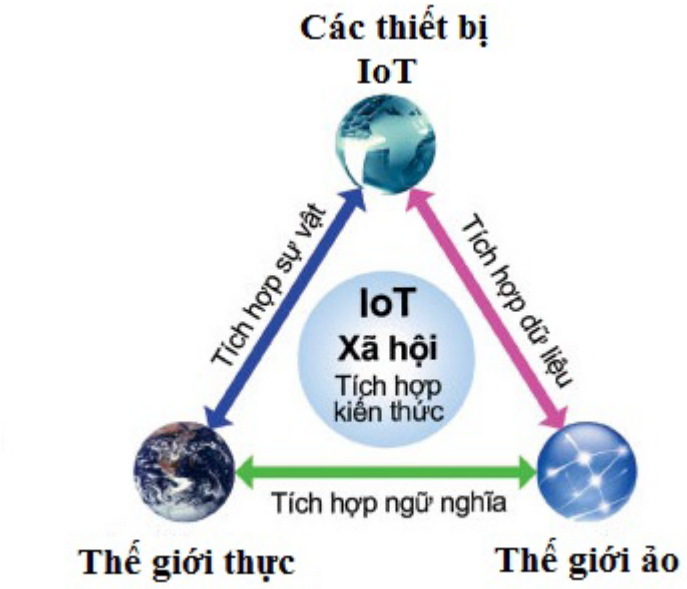
- Nội dung 1: Tìm hiểu lý thuyết về các chuẩn giao tiếp, cách hoạt động, cách thức quản lý, các thông số, ưu nhược điểm khi triển khai trên các hệ thống IoT cụ thể.
- Nội dung 2: Đề xuất, lựa chọn các giải pháp trong từng ứng dụng cụ thể.
- Nội dung 3: Phân tích, lựa chọn các thiết bị, linh kiện, cảm biến phù hợp cho hệ thống.
- Nội dung 4: Tìm hiểu, tham khảo tài liệu liên quan và 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 đa tác vụ. Từ đó phát triển chương trình nhúng cho các thiết bị trên nền tảng này.
- Nội dung 7: Tìm hiểu cách giao tiếp, điều khiển và trao đổi dữ liệu giữa các module truyền nhận (LoRaWAN, Z-Wave, 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 nền tảng IoT trên Server.
- Nội dung 9: Kiểm thử, đo đạc các thông số của hệ thống. Sửa lỗi phần cứng và phần mềm.

2. LÝ THUYẾT

2.1 Tìm hiểu tổng quan về IoT và các kết nối trong 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 [1] (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.

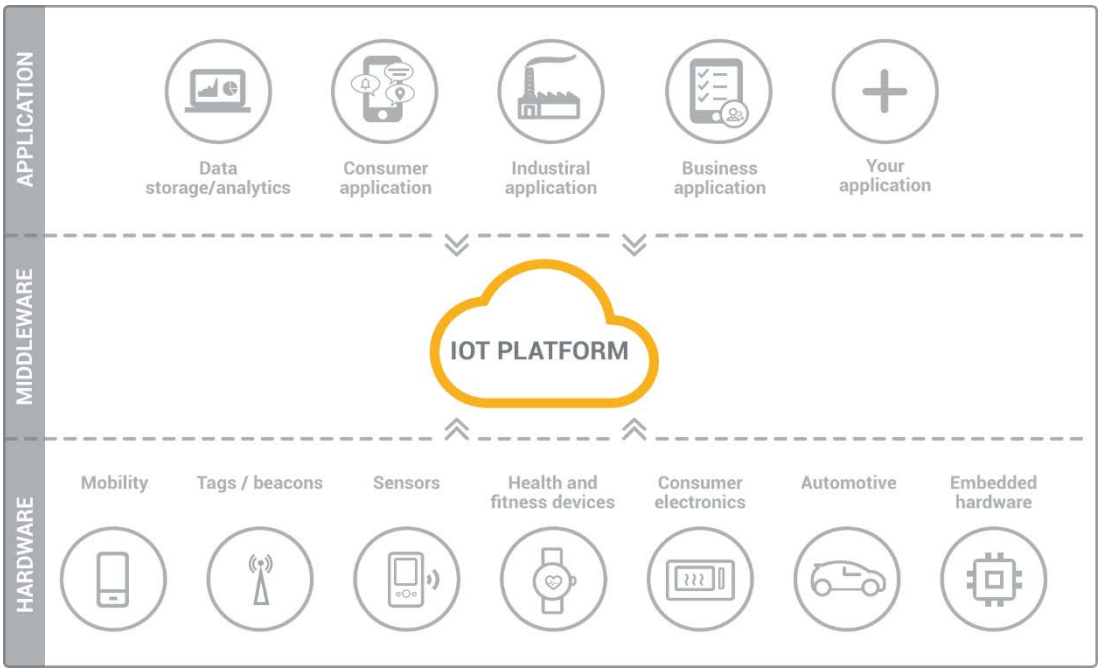


Hình 8: Mô tả tương tác của mạng lưới thiết bị kết nối Internet.

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.



Hình 9: Mô hình các lớp trong một nền tảng IoT.

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 tải 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.

Hình 10: Các giao thức phổ biến trong hệ thống IoT.

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 cây số mà không cần đến bộ khuếch đại. Ngoài ra, 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 đòi hỏi các thiết bị quản lý trên nền tảng mạng và việc triển khai giao thức này lên các thiết bị nhúng khá phức tạp về phần cứng và phần mềm so với RS485.
- 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.
- 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). Đây là công nghệ đang được chú ý và được các nhà sản xuất tập trung nhiều hơn.

Variable	Wi-Fi	Z-Wave	ZigBee	Thread	BLE
Year first launched in Market	1997	2003	2003	2015	2010
PHY/MAC Standard	IEEE 802.11.1	ITU-T G.9959	IEEE 802.15.4	IEEE 802.15.4	IEEE 802.15.1
Frequency Band	2.4 GHz	900 MHz*	2.4 GHz	2.4 GHz	2.4 GHz
Nominal Range (0 dBm)	100 m	30 – 100 m	10 – 100 m	10 – 100 m	30 m
Maximum Data Rate	54 Mbit/s	40-100 kbit/s	250 kbit/s	250 kbit/s	1 Mbit/s
Topology	Star	Mesh	Mesh	Mesh	Scatternet
Power Usage	High	Low	Low	Low	Low
Alliance	Wi-Fi Alliance	Z-Wave Alliance	ZigBee Alliance	Thread Group	Bluetooth SIG

Hình 11: So sánh một số loại kết nối thông dây.

Từ đó có thể thấy, mỗi giao thức đều có điểm mạnh và yếu riêng trong từng điều kiện môi trường. Việc thu thập dữ liệu cảm biến, tổng hợp và truyền tải dữ liệu với một giao thức đơn thuần không thể đáp ứng được sự đa dạng về kiến trúc và địa hình ở Việt Nam. Thiết bị Wifi tốc độ cao nhưng khoảng cách hạn chế và tiêu hao năng lượng lớn, thiết bị Zigbee ít tiêu hao năng lượng hơn nhưng tốc độ chậm và để truyền được xa thì phải triển khai mạng mesh nhiều thiết bị. Những thiết bị có khả năng truyền xa như LoRa thì bị giới hạn về tốc độ, băng

thông. Các thiết bị có khoảng cách rộng như GSM thì phải phụ thuộc vào phạm vi phủ sóng và độ ổn định của dịch vụ nhà mạng cung cấp. Do đó, cần phải có một nền tảng IoT hỗ trợ đa giao thức để đáp ứng tốt với các điều kiện môi trường khác nhau và thuận tiện cho người dùng trong việc lựa chọn thiết bị.

Dựa trên đặc trưng của các giao thức trên, để đảm bảo một hệ thống cân bằng với các thông số **khoảng cách**, **năng lượng**, **tốc độ** và các **đặc trưng** về một nền tảng IoT, trong đề tài sẽ triển khai 3 loại kết nối chính:

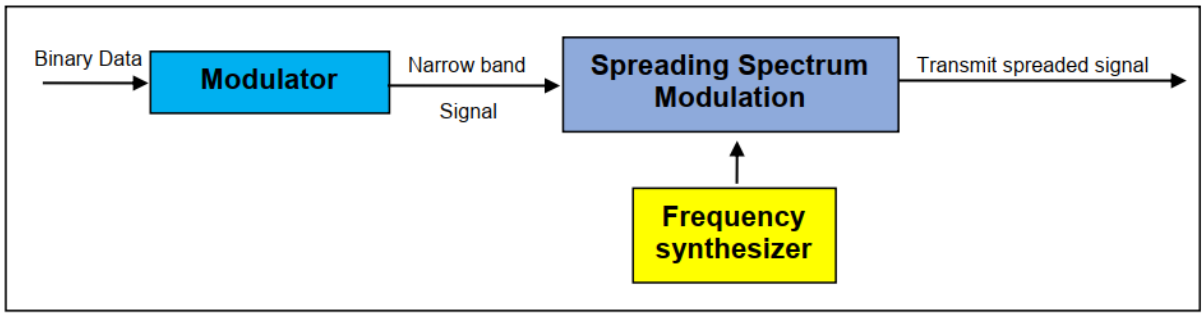
- RS485: thích hợp dùng trong các ứng dụng có độ trễ thấp, trong mạng nội bộ. Trong đề tài sẽ thực hiện triển khai giao thức Modbus Serial dựa trên nền tảng vật lý của RS485 để thuận tiện hơn trong việc quản lý, điều khiển và thu thập dữ liệu từ các thiết bị. Ngoài ra, có thể mở rộng mạng lưới các thiết bị nhờ vào khả năng tương thích với các thiết bị bên ngoài theo tiêu chuẩn Modbus.
- LoRa: thích hợp cho việc thu thập dữ liệu ở khoảng cách xa và không yêu cầu nhiều về băng thông và tốc độ, ngoài ra các ứng dụng LoRa đảm bảo được mức độ tiêu thụ năng lượng, có thể hoạt động lâu dài dựa trên nguồn pin. Trong đề tài sẽ thực hiện triển khai giao thức LoRaWAN (là một giao thức lớp trên của LoRa), đề tài sẽ tập trung vào các cơ chế quản lý đa thiết bị trong mạng, các đảm bảo về sự toàn vẹn của gói tin, bảo mật và cơ chế tiết kiệm năng lượng dựa theo các loại thiết bị của tiêu chuẩn LoRaWAN.
- Z-Wave: do có khoảng cách cơ bản lớn hơn Zigbee cũng như có thể triển khai mạng mesh để tăng khoảng cách truyền nhận, Z-Wave thích hợp với các ứng dụng có khoảng cách gần và vừa phải, có thể kế đến như nhà thông minh. Ngoài ra việc triển khai Z-Wave làm hệ thống có thể mở rộng dễ dàng hơn do có thể tương thích với các loại thiết bị bên ngoài hỗ trợ giao thức này từ nhiều nhà cung cấp khác nhau.

Trong đề tài, việc sử dụng các giao thức khác như Ethernet, Wifi trong việc liên kết các khối cũng được thực hiện.

2.2 Giới thiệu công nghệ LoRa và giao thức LoRaWAN

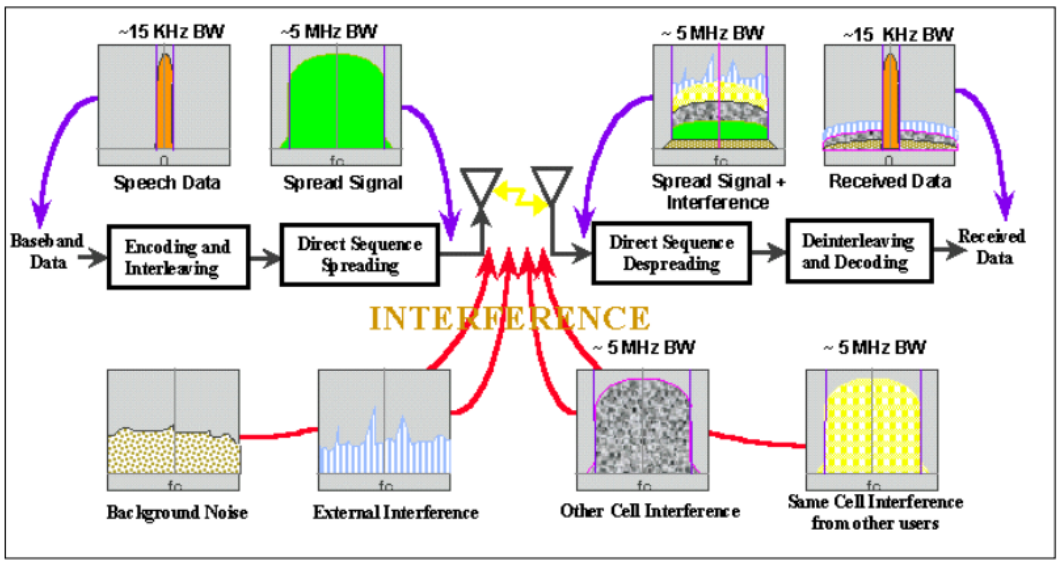
2.2.1 Giới thiệu công nghệ truyền thông không dây LoRa

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 [2].



Hình 12: Sơ đồ điều chế tại phía phát dùng công nghệ LoRa.

Công nghệ LoRa sử dụng sơ đồ điều chế trải phổ (spread spectrum modulation), công nghệ này bắt nguồn từ kỹ 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.

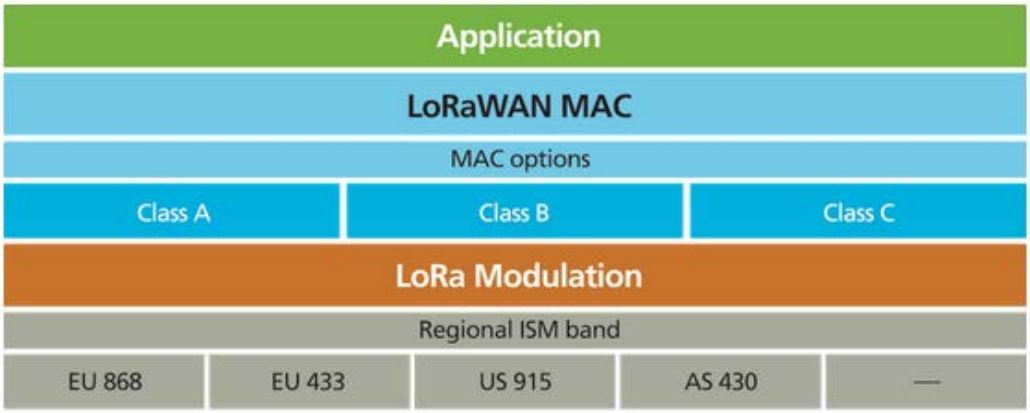


Hình 13. Mô phỏng hình dạng tín hiệu khi truyền và nhận trong mạng LoRa.

Sơ đồ khối trên thể hiện quá trình điều chế tín hiệu dùng kĩ thuật LoRa trước khi phát ra antenna. 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. 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.

2.2.2 Giới thiệu giao thức LoRaWAN

Giao thức LoRaWAN được phát triển dựa trên giao thức LPWA (Low Power, Wide Area), được đặt ra bởi LoRa Alliance [3]. LoRaWAN là giao thức điều khiển phương tiện (MAC) so với LoRa ở lớp vật lý (Physical), có thể xem LoRaWAN tương tự như lớp thứ 2 và 3 ở mô hình mạng OSI.

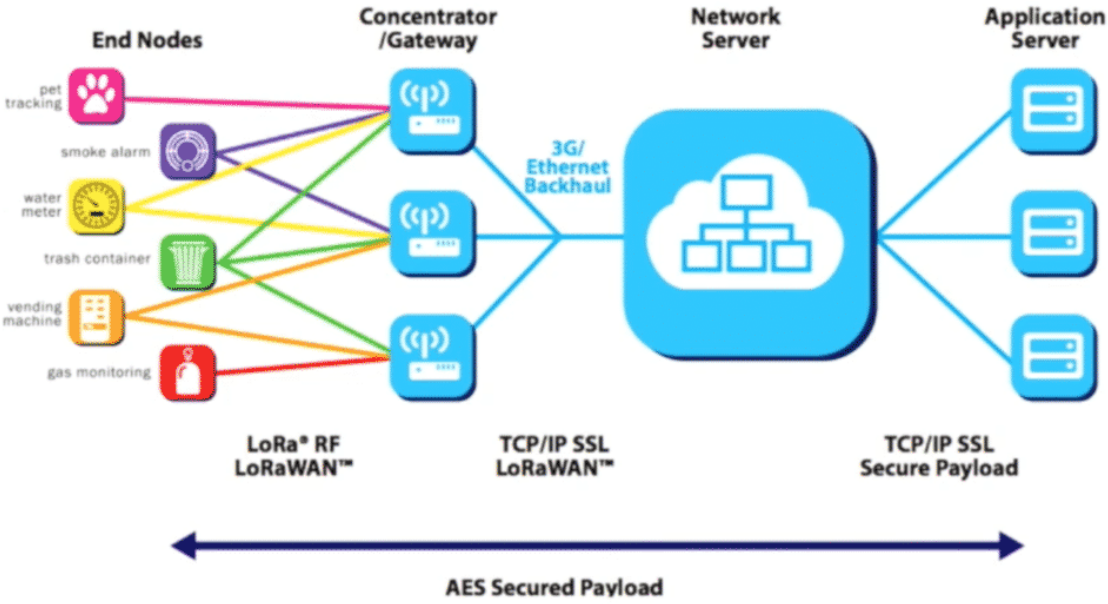


Hình 14: Mô hình phân lớp của LoRaWAN

Các thành phần của mạng LoRaWAN:

- End Node - các thiết bị đầu cuối: các thiết bị nhúng thu thập dữ liệu,
- Gateway: nhận các gói tin từ các thiết bị đầu cuối để gửi lên Network Server và ngược lại.
- Network Server: máy chủ quản lý các thiết bị, gói tin theo tiêu chuẩn của LoRaWAN. Ngoài ra trên Server còn các các ứng dụng, các API giúp tương tác với các chức năng của LoRaWAN.

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, …



Hình 15. Mô hình giải pháp LoRaWAN trong các ứng dụng.

Thiết bị LoRaWAN được chia thành 3 loại (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 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.

Các tính năng chính của LoRaWAN:

- Tầm xa: kết nối các thiết bị tầm xa lên đến 30 dặm tại các khu vực nông thôn và khả năng thâm nhập các môi trường dày đặt trong nhà hoặc đô thị.
- Năng lượng thấp: yêu cầu năng lượng tối thiểu, tuổi thọ pin có thể kéo dài lên đến 10 năm, giảm chi phí thay thế, bảo trì.
- Đảm bảo: tính năng mã hoá AES128 đầu cuối, xác thực lẫn nhau, bảo vệ tính toàn vẹn của gói tin và bảo mật.
- Chuẩn hoá: cung cấp khả năng tương tác và tính khả dụng của các thiết bị theo chuẩn LoRaWAN giúp triển khai nhanh chóng các thiết bị IoT.
- Ngoài ra còn có thể kể đến các lợi ích khác như: hoạt động trên miền tần số miễn phí (không cần giấy phép), tính di động và khả năng định vị do tiêu thụ năng lượng thấp, giá thành khi triển khai phạm vi rộng.

Bên cạnh đó, LoRaWAN cũng có một số hạn chế nhất định:

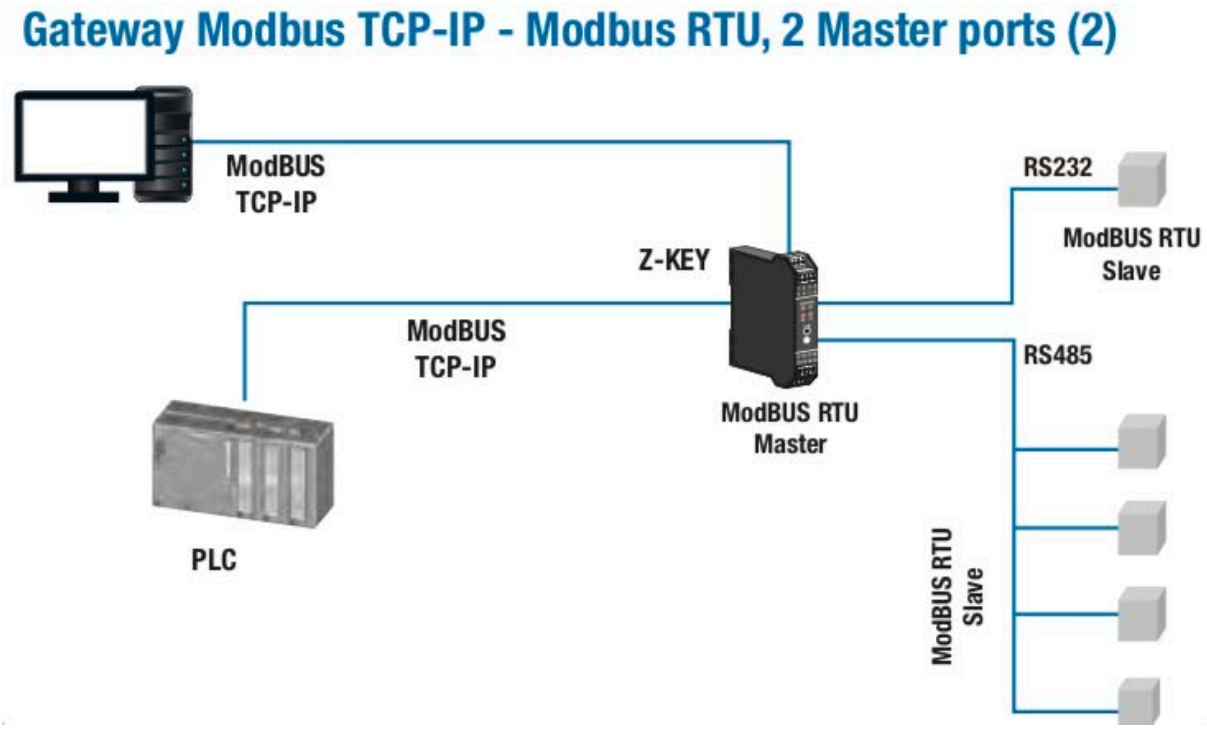
- Không dành cho các ứng dụng có nhu cầu dữ liệu lớn, việc gửi các gói tin và kích thước gói tin đều phải theo tiêu chuẩn giới hạn của LoRaWAN.
- Không thích hợp cho các ứng dụng đòi hỏi độ trễ thấp và các ràng buộc về thời gian thực.
- Tốc độ dữ liệu thấp, triển khai trên miền tần số không thu phí nên có khả năng bị nhiễu với các tần số lân cận và các ứng dụng có tần số gần đó.

2.3 Giới thiệu giao thức Modbus

Modbus do Modicon (hiện nay thuộc Schneider Electric) [4] 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.



Hình 16: Mô hình các thiết bị trong mạng Modbus.

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:

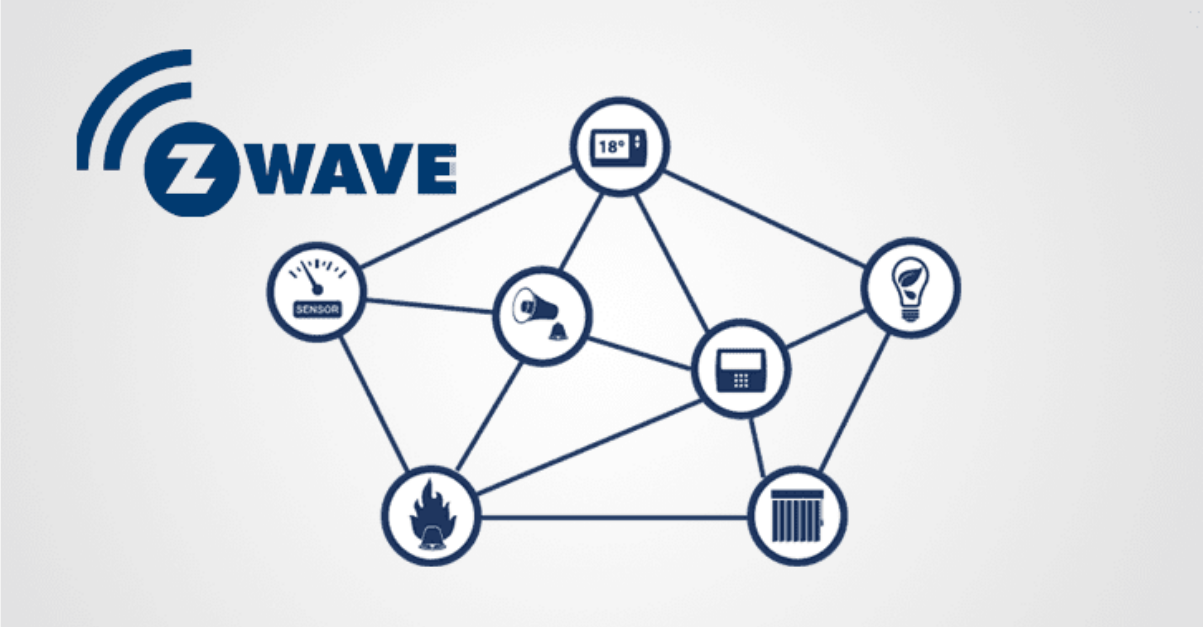
Modbus ASCII: Mọi thông điệp được mã hóa bằng hexadeci-mal, sử dụng đặc tính ASCII 4 bit. Đối với mỗi một byte thông tin, cần có 2 byte truyền thông, gấp đôi so với Modbus RTU hay Modbus TCP. Tuy nhiên, Modbus ASC II chậm nhất trong số 3 loại protocol, nhưng lại thích hợp khi modem điện thoại hay kết 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 115200 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.

2.4 Giới thiệu giao thức Z-Wave

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 [5].



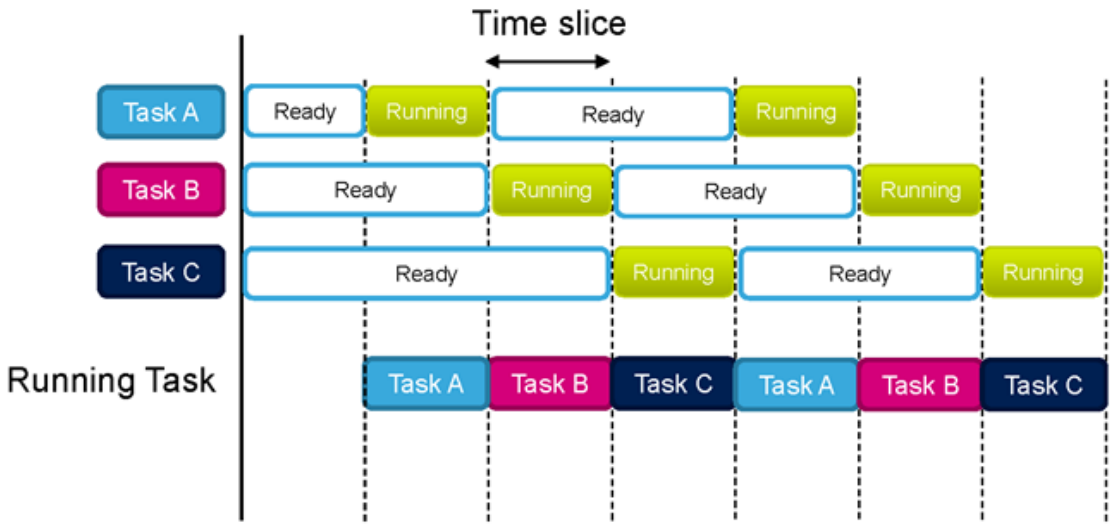
Hình 17: Mạng Z-Wave và các ứng dụng.

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.

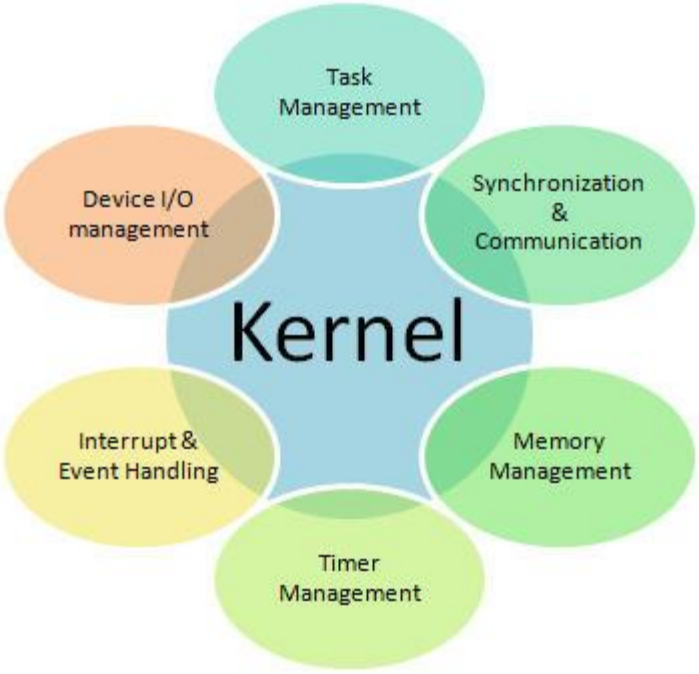
2.5 Giới thiệu về hệ điều hành thời gian thực RTOS

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.



Hình 18: Ví dụ về multi-tasking trong các hệ thống thời gian thực.

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 [6]. 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.



Hình 19: Các thành phần cơ bản của một RTOS.

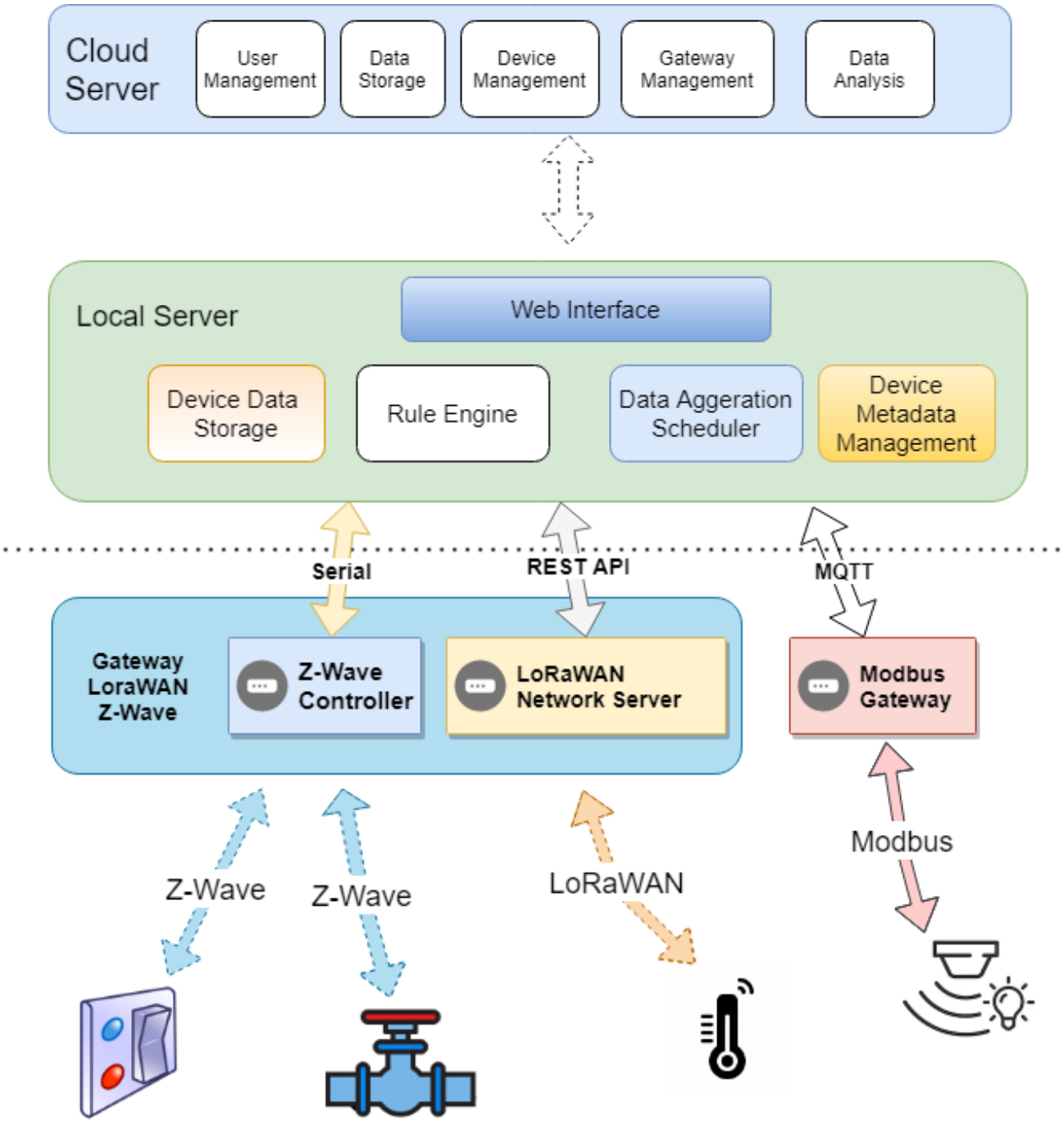
Các lợi ích của việc dùng RTOS:

- 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, việc theo dõi hoạt động của hệ thống dễ dàng hơn.

Dựa trên mục tiêu đề tài, sử dụng FreeRTOS giúp đơn giản việc xây dựng hệ thống chạy đa tác vụ để đảm bảo các tính ra. Bên cạnh đó, khả năng phát triển, kiểm tra gỡ lỗi nền vi điều khiển cũng được tiện lợi hơn.

3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

Từ việc nghiên cứu các nền tảng IoT đã nêu ở phần lý thuyết, đề tài đã xây dựng được mô hình cho một nền tảng IoT như sau:



Hình 20: Sơ đồ khối thiết kế hệ thống IoT trong đề tài.

Nền tảng IoT trên bao gồm hai phần chính như sau:

- Phần phía bắc (Northbound): có nhiệm vụ chính là thu thập, tổng hợp và phân tích dữ liệu. Phần này gồm hai khối là Local Server và Cloud Server:
  - Local Server đảm nhiệm việc thu thập và tổng hợp dữ liệu từ các thiết bị phần cứng ở biên (Edge) cũng như quản lý các thiết bị này. Đồng thời, Local Server đóng vai trò như cầu nối trung chuyển, sao lưu dữ liệu tại biên lên Cloud Server và truyền tải các yêu cầu và lệnh từ Cloud Server xuống các thiết bị phần cứng.
  - Cloud Server đảm nhiệm việc tổng hợp dữ liệu nhận được từ các Local Server ở biên (Edge) và vận hành các tác phụ phân tích và nghiên cứu trên tập dữ liệu này. Đồng thời, Cloud Server là trung gian cấp các giao diện để người dùng có thể điều khiển và quản lý các thiết bị phần cứng ở biến từ xa.
- Phần phía nam (Southbound): có nhiệm vụ chính là tương tác với thể giới thực, như là đo lường các thông số của điều kiện môi trường và điều khiển các thiết bị ở hiện trường. Phần này bao gồm các thiết bị cảm biến (Sensor), thiết bị thi hành (Actuator) và các thiết bị trung chuyển (Gateway):
  - Thiết bị Sesor thu thập giá trị của các điều kiện môi trường, chuyển đổi các giá trị này thành dữ liệu có thể đọc được và gửi về Local Server ở northbound thông qua các Gateway.
  - Thiết bị Actuator tương tác với môi trường và làm thay đổi trạng thái của các thiết bị ở thế giới thực. Việc tương tác này được điều khiển thông qua các lệnh từ Local Server gửi đến Actuator thông qua các thiết bị Gateway.
  - Thiết bị Gateway trung chuyển dữ liệu gửi từ Sensor, Actuator lên Local Server và các lệnh gửi từ Local Server xuống Actuator.

Tuy nhiên, nội dung của đề tài chỉ tập trung vào việc thiết kế, xây dựng phần phía nam (Southbound) gồm các thiết bị phần cứng Gateway, Sensor và Actuator cùng các chương trình nhúng trên các thiết bị này để có thể đo lường, tương tác với môi trường và gửi dữ liệu về Local Server ở phía bắc (Northbound). Phần phía bắc được thiết kế và xây dựng ở một đề tài khác. Cụ thể:

Các thiết bị đầu cuối (Actuator, Sensor):

Trong phạm vi đề tài sẽ thực hiện các bản thử nghiệm của một số thiết bị IoT thông dụng, trong đó gồm: **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. **Van nước** điều khiển qua Z-Wave, **Z-Wave Node** có các kết nối tích hợp cho các thiết bị có sẵn vào mạng Z-Wave (công tắc, cảm biến môi trường, cảm biến chuyển động, …). Ngoài ra, đề tài cũng sử dụng thiết bị công nghiệp (biến tần, PLC, …) và các thiết bị chuẩn Z-Wave từ các nhà cung cấp bên ngoài giải để thử nghiệm độ tương thích.

Các Gateway:

Có nhiệm vụ trung chuyển dữ liệu giữa các Node và Local Server. Bao gồm **Gateway Modbus Serial**, **Gateway LoRaWAN và Z-Wave**. Các Gateway là nơi tập trung và phân phối dữ liệu ứng với từng giao thức nó hỗ trợ.

3.1 Gateway Modbus Serial

Yêu cầu:

- Có các cổng RS485 để giao tiếp với các thiết bị chuẩn Modbus có sẵn trên thị trường, cổng RS232, UART để 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, còi để thông báo cho người dùng.
- Hỗ trợ tối đa 4 cổng RS485 để quản lý được nhiều thiết bị.

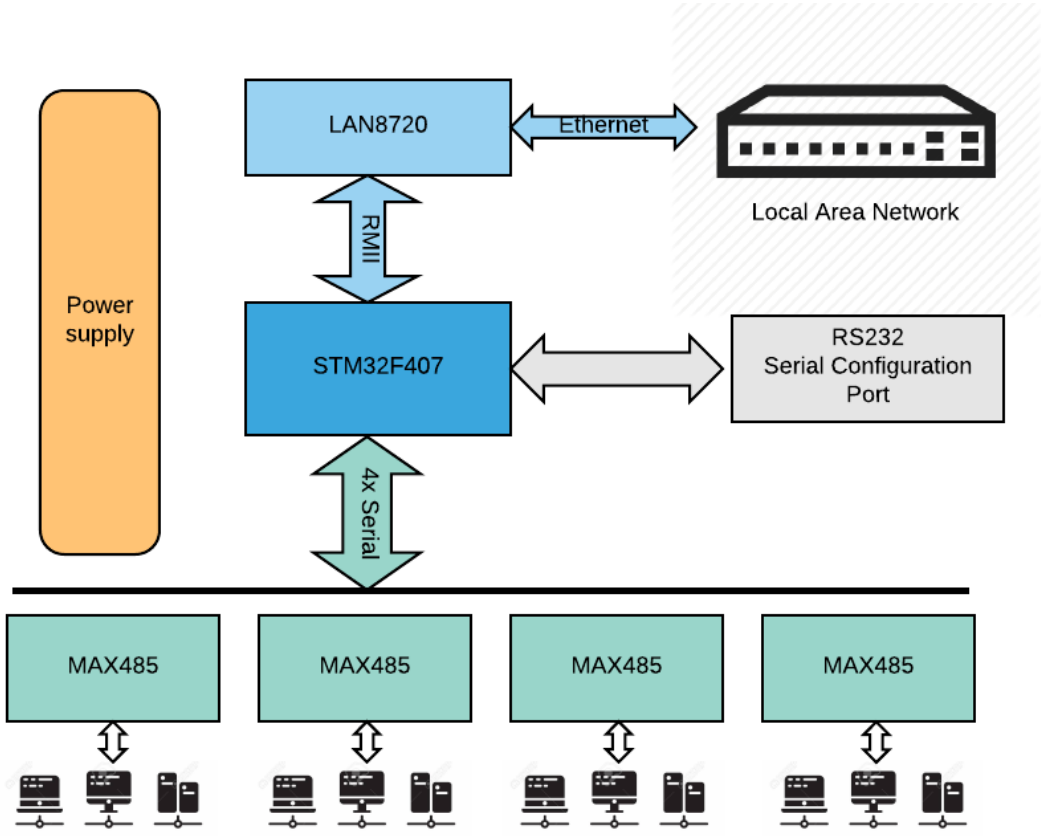
Phân tích:

Bảng 1: Phân tích các phương án cho 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 thông báo chức năng qua đèn Led, còi báo.	Cấu hình và ngoại vi cơ bản đáp ứng được yêu cầu. Được dùng phổ 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 IC LAN8720A	IC giao tiếp Ethernet, hỗ trợ tốc độ cao 10/100 Mbps.	Rẻ, nhỏ gọn, giao tiếp RMII dễ sử dụng. Đơn giản hoá trong việc xử lý các gói ở lớp PHY.	
Sử dụng IC đổi mức tín hiệu MAX485	IC chuyển tiếp UART-TTL sang RS485.	Chuẩn giao tiếp thích hợp cho tốc độ cao và truyền xa. Chuẩn giao tiếp hỗ trợ kết nối đa điểm	Một số IC không chính hãng hoạt động không ổn định.
Sử dụng IC AP3211 làm bộ điều chế nguồn cho board.	IC AP3211 có kích thước nhỏ gọn, dòng điện đầu ra tối đa 1.5 ampe.	Nguồn có hiệu suất chuyển đổi cao. Có chế độ bảo vệ quá dòng, bảo vệ quá áp, bảo vệ quá nhiệt độ trên IC AP3211 (nhiệt độ vượt quá 160^0C).	Thiết kế phức tạp hơn so với nguồn tuyến tính.

Sơ đồ khối tổng quát:



Hình 21: Sơ đồ khối tổng quát của Gateway Modbus Serial.

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

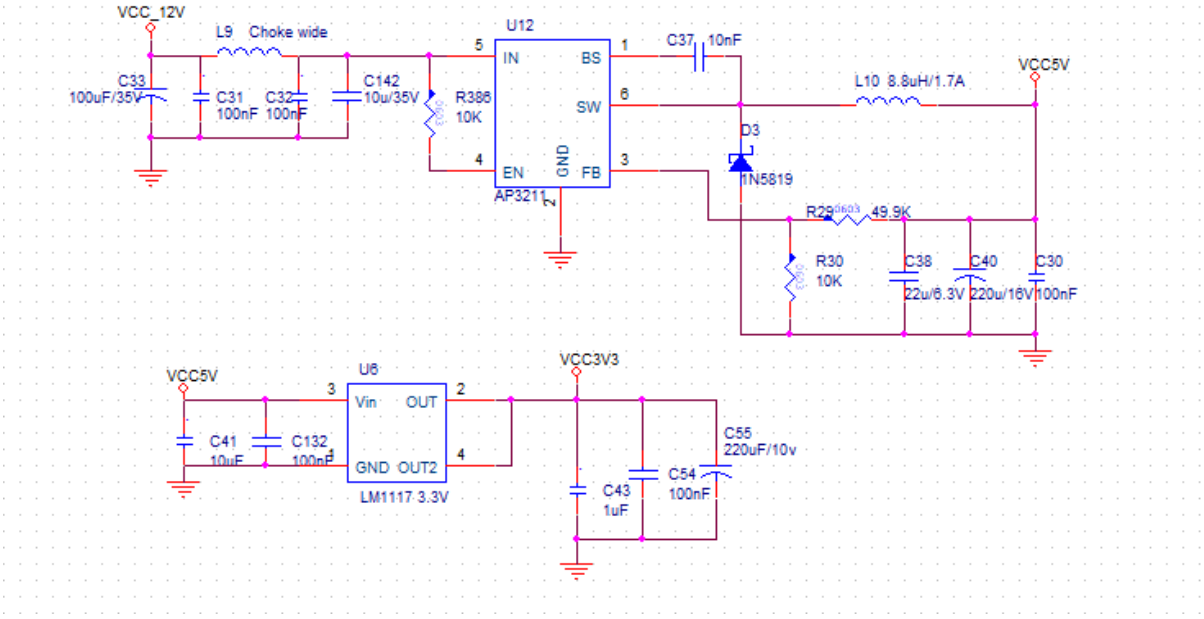
- Giao tiếp với MAX485 để tương tác với các thiết bị trọng mạng thông qua Modbus protocol.
- Giao tiếp với RS232 để cung cấp môi trường cài đặt với máy tính.
- Giao tiếp với LAN8720 để kết nối đến mạng nội bộ thông qua cổng giao tiếp Ethernet.

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 qua cổng Ethernet.

Sơ đồ mạch chi tiết:

Khối nguồn cung cấp:

Bộ nguồn của Gateway đảm bảo 2 mức điện áp chính là 5V và 3.3V với nguồn adapter 12V. Trong đó bộ nguồn xung (IC AP3211) được sử dụng để giảm áp xuống 5V và nguồn tuyến tính (IC AMS1117) được dùng để giảm 5V xuống 3.3V.

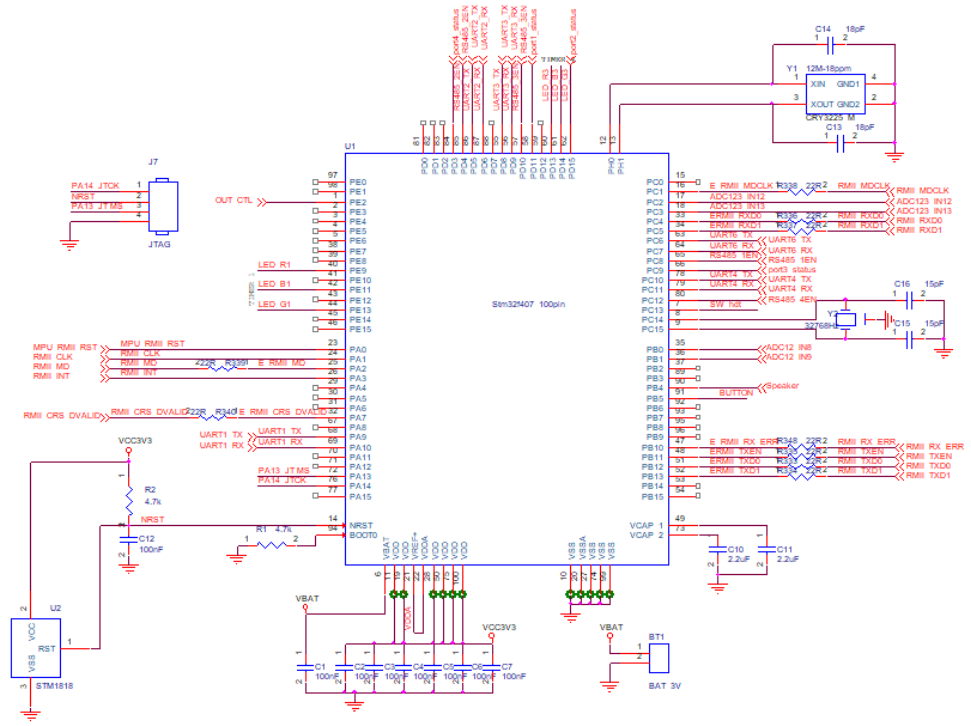


Hình 22: Sơ đồ mạch chi tiết bộ nguồn của Gateway Modbus Serial.

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 có thiết kế tương tự nhưng phải tính toán lại một số thông số về trở Feedback để thay đổi mức điện áp đầu ra. Cuộn dây và diode được tính toán và lựa chọn lại. Các tụ lọc nguồn, tụ by-pass, tụ bootstrap cũng cần được lựa chọn giá trị và điện áp hợp lý với từng chức năng. Bên cạnh đó việc placement các linh kiện và vẽ các shape dẫn điện cũng cần phải quan tâm để đạt được hiệu quả tốt nhất.

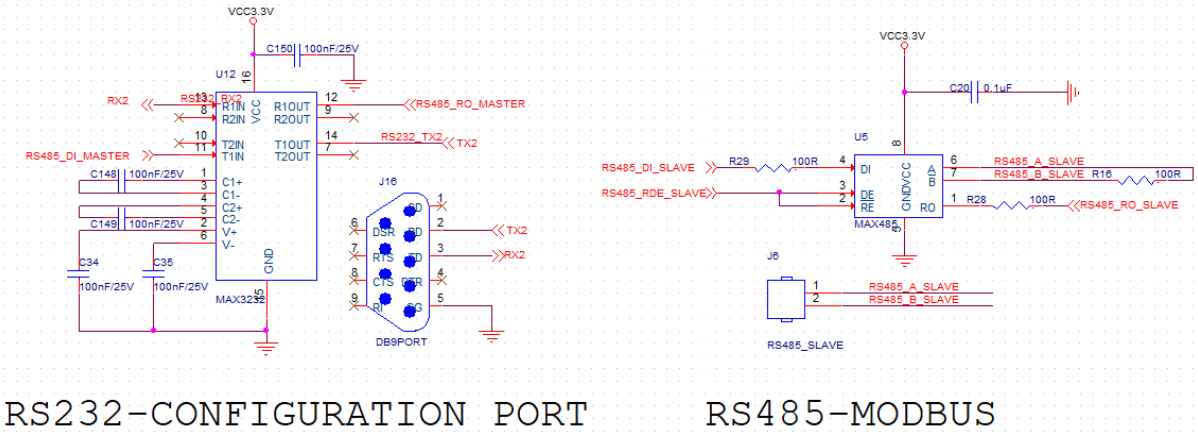
Khối vi điều khiển và các kết nối:

Khối vi điều khiển đảm bảo các yêu cầu xung quanh của STM32F407VET6, bao gồm: các tụ lọc nguồn, thạch anh ngoài 12 Mhz dành cho nguồn dao động chính và 32.768 Khz cho RTC, các điện trở kéo để thay đổi chế độ và mạch ra chân SWD cho nạp code và gỡ lỗi.



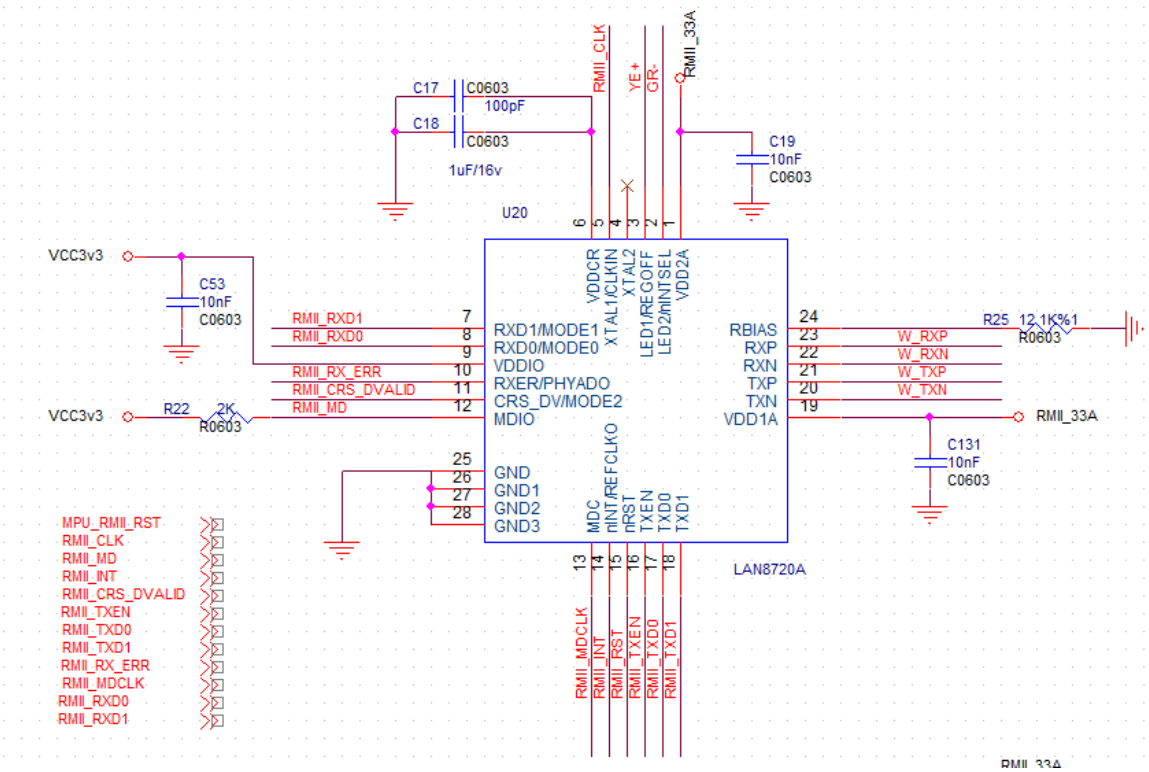
Hình 23: Sơ đồ mạch chi tiết khối vi điều khiển của Gateway Modbus Serial.

Khối Serial gồm một cổng RS232 để giao tiếp máy tính và 4 cổng RS485 dành cho Modbus ở chế độ 4 Master.



Hình 24: Khối Serial của Gateway Modbus Serial.

Khối giao tiếp Ethernet, sử dụng IC chuyển đổi LAN7820 sang giao diện RMII cho STM32. Việc kết nối các tín hiệu theo đúng các chân mà STM32 hỗ trợ đồng thời các luật routing cho các dây tin hiệu này cần được đảm bảo do tín hiệu có tần số cao nhất của giao diện RMII có thể lên đến 50Mhz.



Hình 25: Khối LAN8720 của Gateway Modbus Serial.

3.2 Gateway LoraWAN và Gateway Z-Wave

Yêu cầu:

- Hỗ trợ khe kết nối đến LoRa Concentrator và Z-Wave module.
- Tương thích với phần cứng máy tính nhúng Rasberry Pi.
- Khối nguồn đảm các khối chức năng hoạt động ổn định.
- Kích thước nhỏ gọn, sắp xếp linh kiện hợp lý để dễ lắp đặt.

Phân tích:

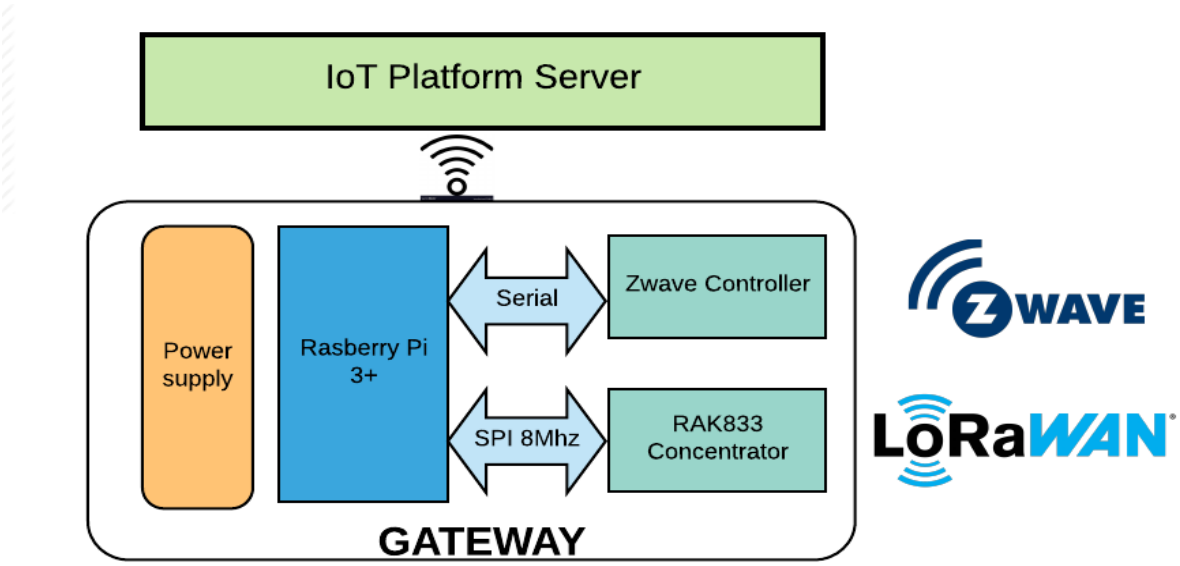
Bảng 2: Phân tích các phương án cho Gateway LoRaWAN và Z-Wave.

Phương án	Mô tả hoạt động	Ưu điểm	Nhược điểm
Sử dụng máy tính nhúng Rasberry Pi 3 B+ làm khối điều khiển	Raspberry là 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. Phiên bản Rasberry Pi 3+ sử dụng chip Broadcom BCM2837B0, quad-	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	Pinout mở rộng nhiều tuy nhiên chỉ hỗ trợ 1 cổng serial.

chính của Gateway.	core, tốc độ 1.4Ghz, 1 GB RAM, 4xUSB 2.0, 10/100mb Ethernet, Wifi 802.11n và 40 GPIO pinout mở rộng.	nhiều nền tảng hệ điều hành thông dụng như Linux, Raspbian nên dễ dàng trong việc triển khai các ứng dụng.	
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 công bố của 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 1.5A.	Nguồn có hiệu suất chuyển đổi cao. Có chế độ bảo vệ quá dòng, bảo vệ quá áp, bảo vệ quá nhiệt độ trên IC AP3211 (nhiệt độ vượt quá 160^0C).	Thiết kế phức tạp hơn so với nguồn tuyến tính.
Tích hợp ZM5202 Z-Wave module	Chạy firmware Z-Wave Controller giúp thiết lập mạng Z-Wave (thêm và xóa và điều khiển thiết bị Z-Wave) thông qua việc truyền nhận các gói tin serial với máy tính nhúng.	Nhỏ gọn, footprint không phức tạp, thuận tiện cho việc thiết kế. Dễ dàng cấu hình và tùy chỉnh firmware.	Các pad nằm ở bụng chip gây khó khăn trong việc hàn thủ công cũng

			như tháo lắp gỡ lỗi.

Sơ đồ khối tổng quát:



Hình 26: Sơ đồ khối tổng quát của LoraWAN và Z-Wave Gateway.

Khối trung tâm Gateway: máy tính nhúng Rasberry Pi có vài trò là trung tâm trung chuyển dữ liệu. Máy tính nhúng sẽ chạy các driver để tương tác với các khối tập trung dữ liệu (LoRa Concentrator và Z-Wave Controller) và các gói phần mềm ứng với từng giao thức. Từ đó cung cấp các interface cho IoT Server để thực hiện các luồng dữ liệu từ các khối tập trung của từng giao thức lên IoT Server và ngược lại.

Z-Wave Controller: là một Z-Wave module, được lập trình ở chế độ controller, có nhiệm vụ quản lý và điều khiển các Node trong mạng Z-Wave, kết nối với máy tính nhúng qua đường UART Serial.

RAK833 Concentrator: bộ tập trung dữ liệu của LoraWAN, có nhiệm vụ thu 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 LoRaWAN.

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.

Sơ đồ mạch chi tiết:

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:

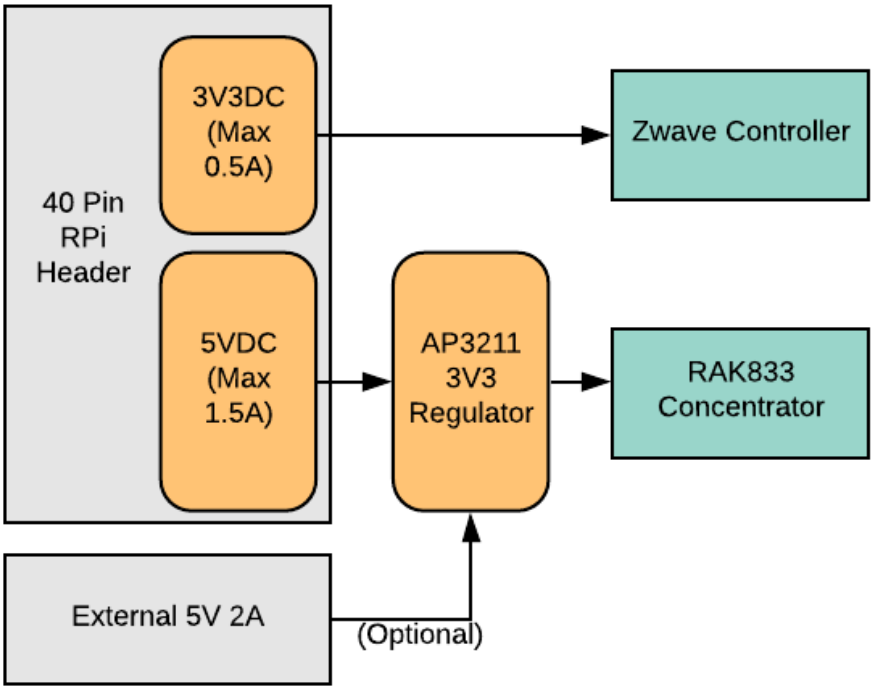
- Pin 5V được nối trực tiếp với nguồn điện cung cấp chính của Rasberry và được bảo vệ bằng cầu chì tự phục hồi, dòng mong muốn có thể đạt đến 1.5A trong trường sử dụng nguồn đầu vào chuẩn của Rasberrypi.org (5V 2.5A).
- Pin 3V3 được hạ áp thông qua nguồn xung tích hợp trên board, có thể lên 0.5A. Tuy nhiên theo nhà sản xuất, Pin này không nên khai thác như một nguồn 3.3V chính cho các ứng dụng, thay vào nếu các ứng dụng cần dòng điện tương đối đáng kể thì nên được cấp nguồn từ mạch một mạch hạ áp bên ngoài.

Đối với RAK833 Concentrator: theo tài liệu tham khảo từ nhà sản xuất, cần tối thiểu một bộ nguồn 3.3V/1A DC.

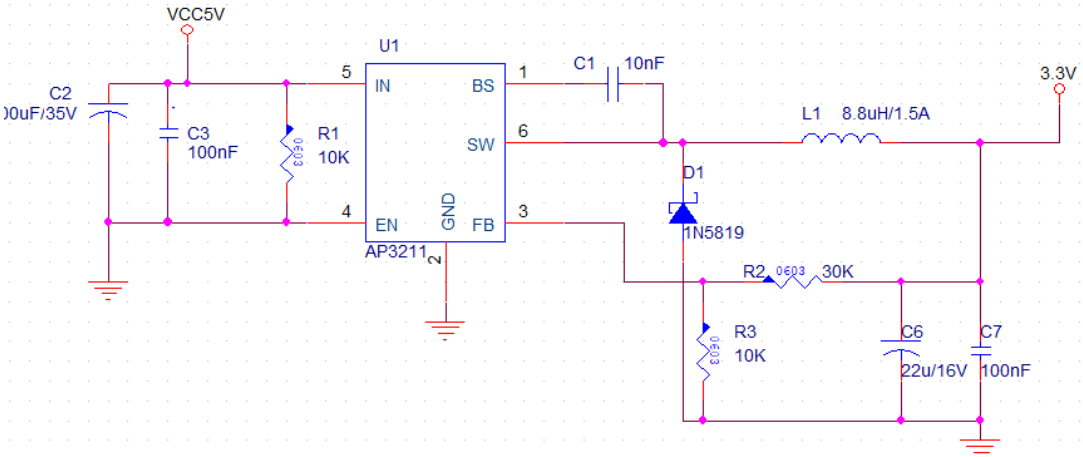
Đối với Zwave ZM5202: Dòng tối đa khi module thu phát tín hiệu là 48mA.

Sau khi tham khảo những thiết kế có sẵn sử dụng trong phòng thí nghiệm. Cùng với những lựa chọn linh kiện theo các phương án đặt ra, thiết kế của bộ nguồn cung cấp trong thiết bị Gateway:

- RAK833 Concentrator được cung cấp điện áp 3.3V từ nguồn xung của IC AP3211. Nguồn này sẽ được thiết kế để chịu tải tối đa 1A, được lấy đầu vào từ Pin 5V của Rasberry Pi.
- Z-Wave Controller có dòng sử dụng thấp nên có thể sử dụng nguồn 3.3V cung cấp bởi pin 3V3 trên Rasberry, phương án này góp phần đảm bảo nguồn của RAK833 không bị ảnh hưởng và tránh bị sự cố khi ảnh hưởng chung khi phần nguồn IC AP3211 có vấn đề.



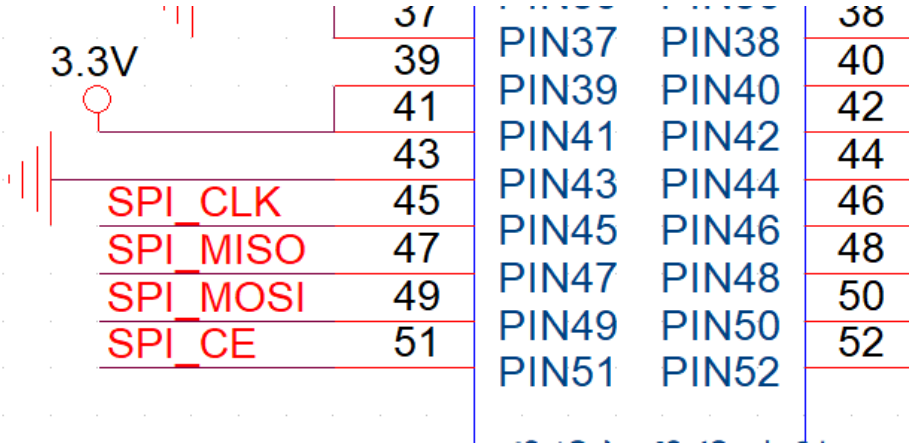
Hình 27: Sơ đồ khối nguồn của Gateway LoraWAN và Z-Wave.



Hình 28: Sơ đồ mạch chi tiết của khối nguồn xung dùng IC AP3211.

Các khối giao tiếp:

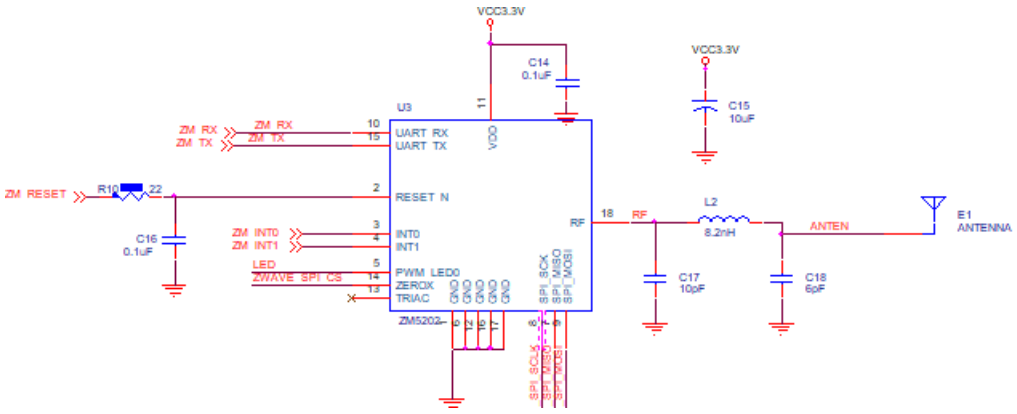
Khối cổng giao tiếp SPI của RAK833 Concentrator:



Hình 29: Sơ đồ mạch chi tiết khối giao tiếp SPI.

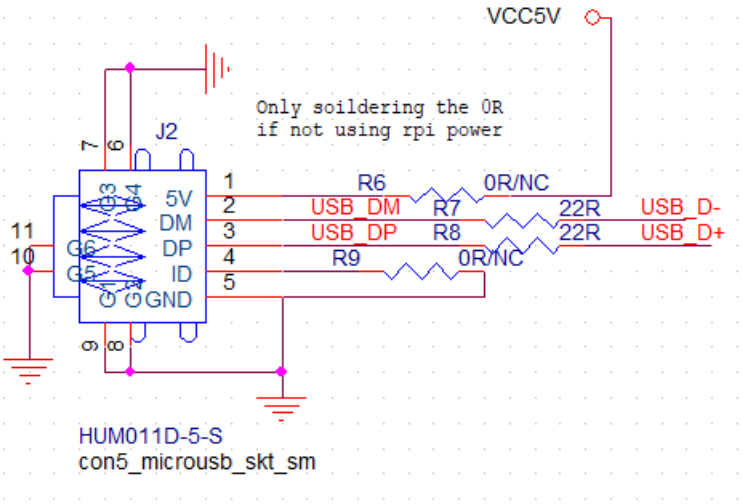
Module RAK833 được thiết kế dạng PCI Express Mini Card với nguồn và các pin chức năng được chỉ định tại một số pin chính trên khe PCI-E. Sau quá trình đọc tài liệu từ nhà sản xuất cũng như giao diện khe PCI-E, các chân quan trọng cần kết nối với máy tính nhúng dựa trên giao thức SPI bao gồm: SPI_MOSI, SPI_MISO, SPI_CLK, SPI_CE và một chân Reset cho module khi khởi động hoặc khởi động lại nếu có lỗi.

Khối cổng giao tiếp Zwave và Zwave Controller:



Hình 30: Sơ đồ mạch chi tiết khối Z-Wave Controller

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



Hình 31: Sơ đồ chi tiết khối USB.

3.3 IoT Node

Yêu cầu:

- Hỗ trợ thu thập dữ liệu từ bên ngoài, bao gồm 3 cổng input xung đầu vào có sử dụng cách ly, 2 cổng tín hiệu hiệu tương tự và một cổng hỗ trợ kết nối OneWire.
- Hỗ trợ giao thức Modbus trong trường hợp Node hoạt động ở chế độ kết nối trong cục bộ có dây.
- Hỗ trợ giao thức LoRaWAN trong trường hợp Node thu thập dữ liệu từ xa.
- Nguồn đầu vào 12V (khi sử dụng nguồn adapter) và 9V (khi sử dụng pin) để dễ dàng lắp đặt vào các hệ thống thực tế.
- Các chức năng cài đặt và lưu lại thông số, các điều khiển qua nút nhấn, các Led thông báo trạng thái.

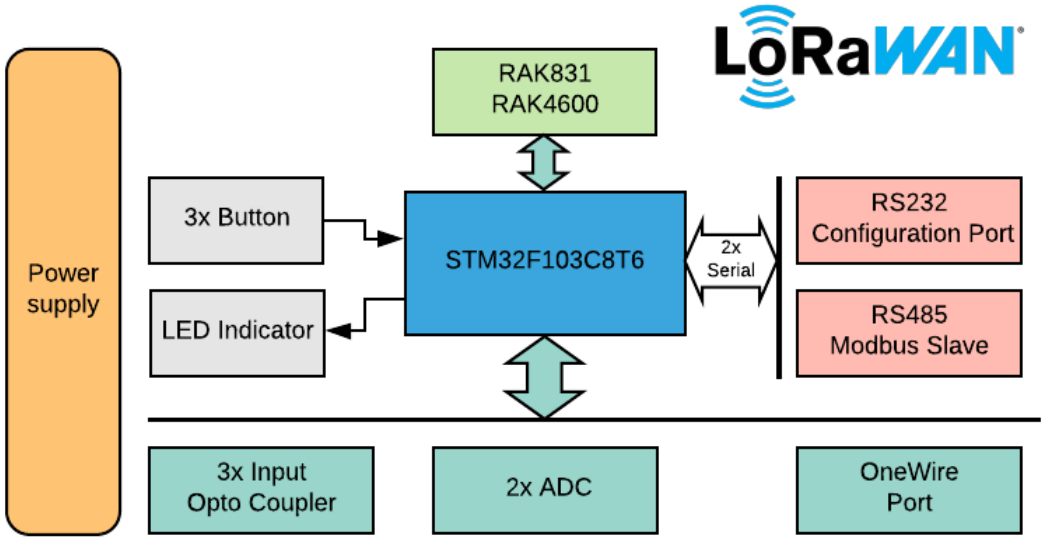
Phân tích:

Bảng 3: Phân tích các phương án cho 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.	Cấu hình và ngoại vi cơ bản đáp ứng được yêu cầu. Được dùng phổ biến nên có sẵn các thư viện cơ bản, cộng đồng hỗ trợ động đảo.	

	Triển khai Modbus Protocol ở chế độ slave. Giao tiếp với module RAK để kết nối vào mạng LoraWAN như 1 node. Các giao tiếp điều khiển bằng tay.	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 module RAK811/ RAK4600 cho việc giao tiếp với LoRaWAN	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 ổn định.

Sơ đồ khối tổng quát:



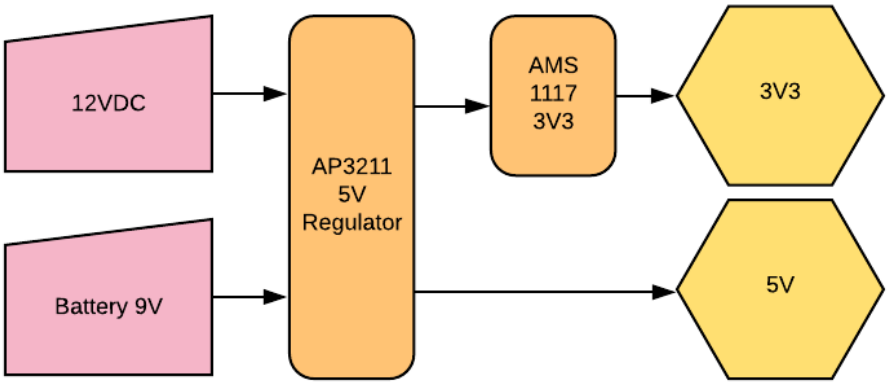
Hình 32: Sơ đồ khối tổng quát của IoT Node.

Khối vi điều khiển giao tiếp với các khối ngoại vi, bao gồm:

- Các khối Opto Input, Analog Input, OneWire để thu thập dữ liệu bên ngoài.
- Các khối giao tiếp Serial: RAK811, RAK4600 để kết nối với mạng LoRaWAN, RS232 để kết nối với cổng cài đặt và RS485 để kết nối cổng hoạt động của giao thức Modbus.
- Đèn Led, nút nhấn để điều khiển.
- Bộ nguồn cung cấp các mức điện áp cho các linh kiện và các cảm biến bên ngoài.

Sơ đồ mạch chi tiết:

Khối nguồn cung cấp:

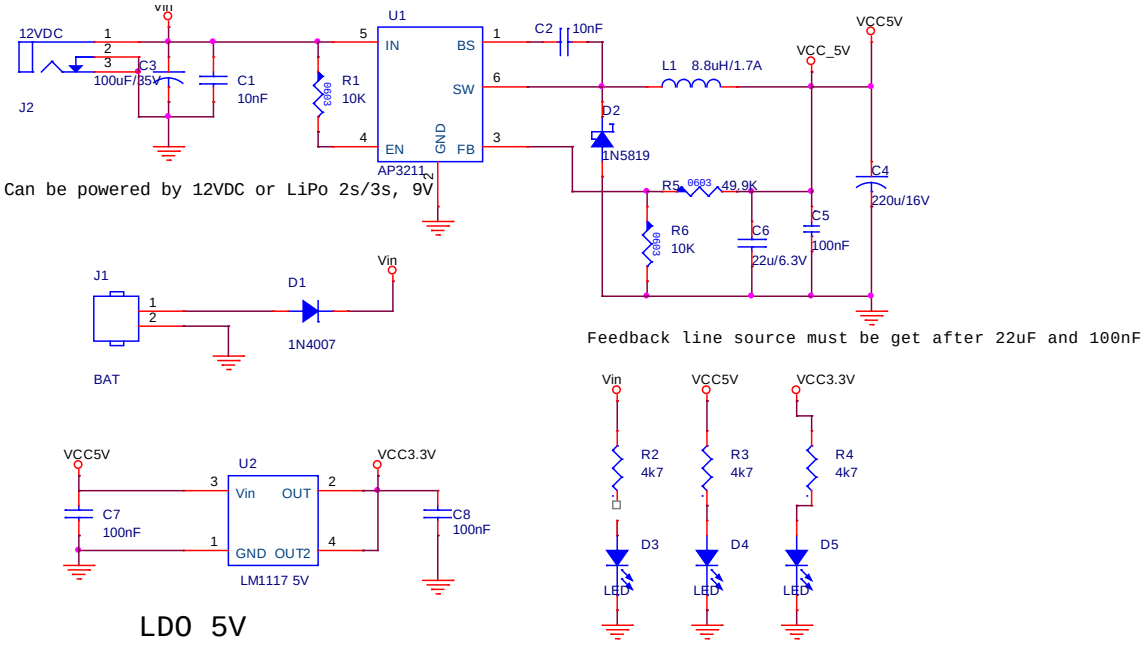


Hình 33: Sơ đồ khối nguồn của IoT Node.

Khối nguồn cần đảm bảo các mức điện áp:

- Điện áp 5V: Là đầu vào cho bộ nguồn 3.3V 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 đã được đề cập ở phía trên, các thông số về tụ, cuộn cảm và trở Feedback có phần thay đổi so với AP3211 ở chế độ áp ra 3.3V.

- 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 ổn định nên sẽ được triển khai bằng AMS1117 3V3, IC này có dòng tối đa 0.8A và có mức rớt áp nhỏ (đầu vào chấp nhận điện áp từ 4.75V – 15V), thiết kế đơn giản, chỉ yêu cầu 1 tụ lọc ở chân output để cải thiện chất lượng đầu ra nên rất thích hợp trong trường hợp này. Bên cạnh đó cần thiết kế tụ lọc, tụ bypass, tụ decoupling cho các IC để đạt được trạng thái nguồn ổn định nhất.

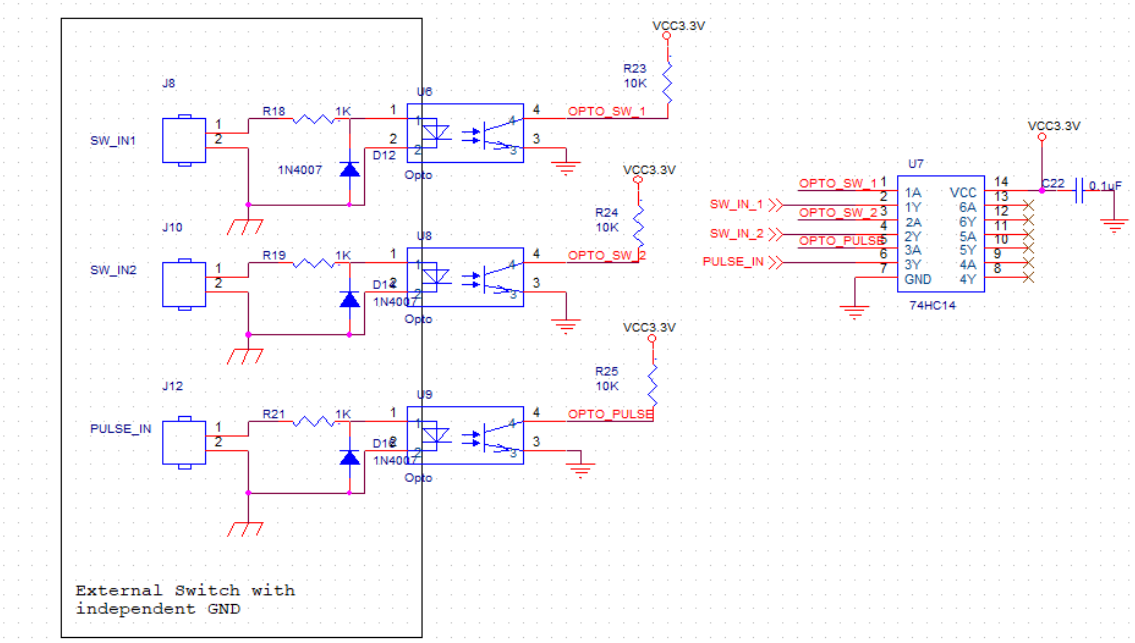


Hình 34: Sơ đồ mạch chi tiết khối nguồn của IoT Node.

Các cổng thu thập tín hiệu:

Trong yêu cầu đặt ra, mạch bao gồm:

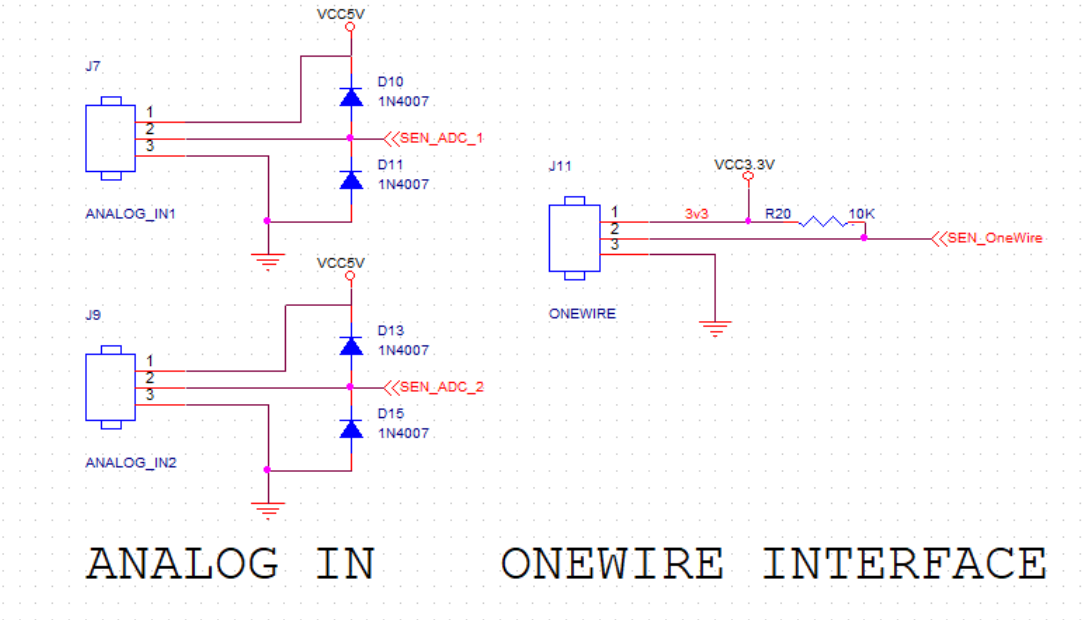
- Các cổng input đầu vào cách ly bằng opto:



Hình 35: Sơ đồ mạch chi tiết khối tín hiệu đầu vào của IoT Node.

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 [9] được dùng để chuyển đổi mức tín hiệu trước khi đi vào chân Input của vi điều khiển.

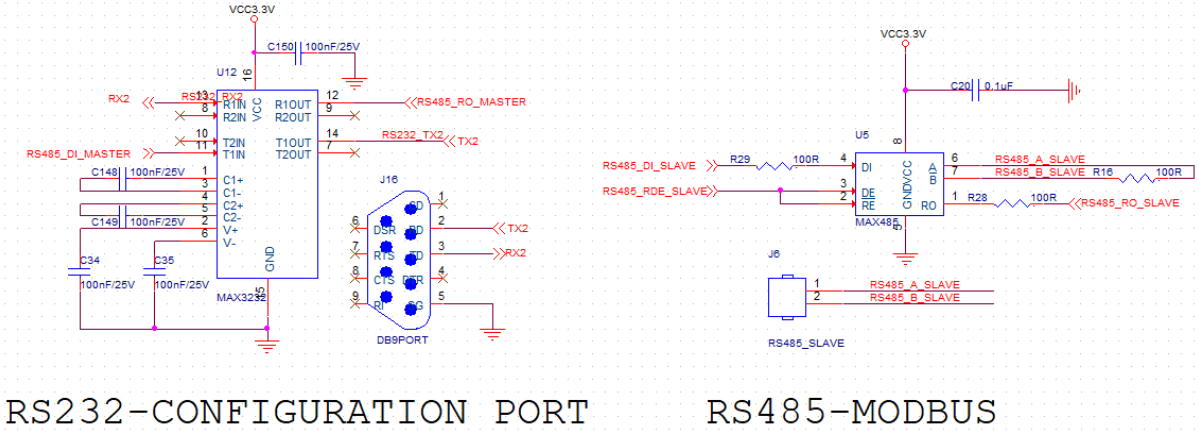
- Cổng đầu vào tín hiệu tương tự Analog và cổng dành cho OneWire:



Hình 36: Sơ đồ mạch chi tiết khối Analog và khối OneWire của IoT Node.

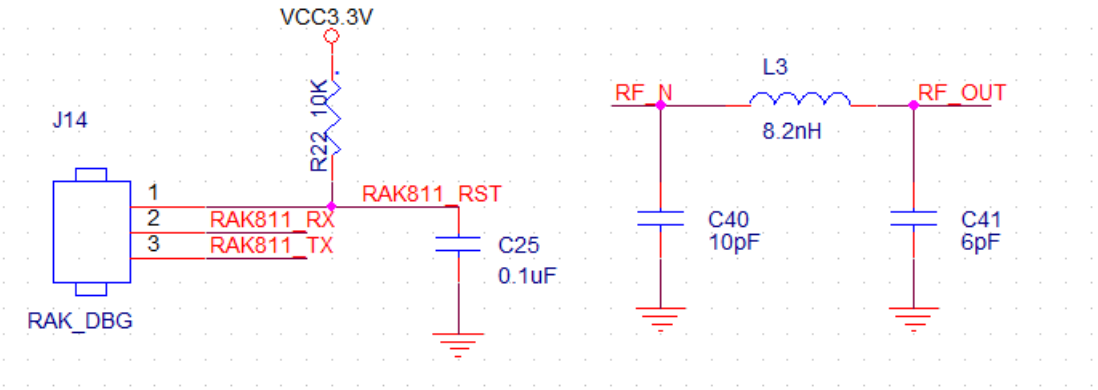
Các khối giao tiếp:

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



Hình 37: Sơ đồ mạch chi tiết khối Serial của IoT Node.

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.



Hình 38: Sơ đồ mạch chi tiết khối giao tiếp LoRaWAN và Antenna của IoT Node.

3.4 Z-Wave Node:

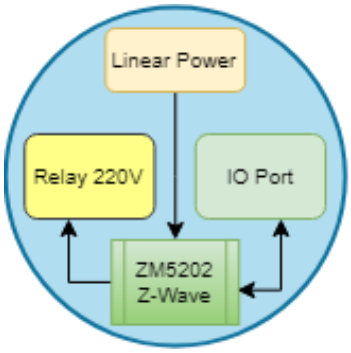
Yêu cầu:

- Hỗ trợ module Z-Wave để kết nối vào mạng Z-Wave.
- Board mạch nhỏ gọn, có nguồn đầu vào khoảng rộng để dễ lắp đặt lên các thiết bị.
- Có các cổng nhập xuất cơ bản để điều khiển và đọc tín hiệu cảm biến.

Phân tích:

Dựa trên các thiết kế từ trước và linh kiện có sẵn trong phòng thí nghiệm, Node sẽ được thiết kế dựa trên module ZM-5202 tương tự như Controller trên Gateway. ZM5202 sẽ điều khiển một công tắc relay 220V và thu nhận, điều khiển tin hiệu thông qua một số pin nhập xuất cơ bản. Bên cạnh đó để đơn giản về kích thước, nguồn của Node được cân nhắc dùng nguồn tuyến tính.

Sơ đồ khối tổng quát:



Hình 39: Sơ đồ khối tổng quát của Z-Wave node.

Sơ đồ mạch chi tiết:

Việc thiết kế Z-Wave node có tương tự trên Gateway, tuy nhiên cần chú ý đến thiết kế mạch phối hợp trở kháng và đầu ra Antenna để đạt được mức thu phát tín hiệu tốt nhất.

Từ việc tham khảo các thiết kế trên mạng cũng như các thiết kế từ trước. Giải pháp Antenna trong Z-Wave sẽ được thực hiện bằng một sợi dây đồng trục, kích thước khoảng 8-10cm (bằng ½ bước sóng của Z-Wave).

3.5 Van nước điều khiển qua Z-Wave

Yêu cầu:

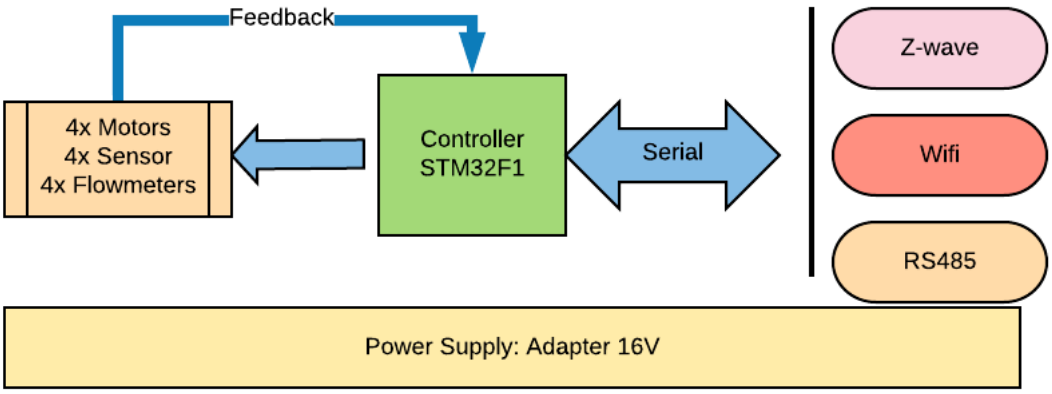
- Hỗ trợ các mức điện áp thông dụng của van nước điện có mặt trên thị trường.
- Hỗ trợ khả năng vận hành đóng mở các van nước điện và các cảm biến bảo vệ.
- Hỗ trợ giao thức Z-Wave để điều khiển van nước điện
- Các nút nhấn điều khiển, đèn Led thông báo.

Phân tích:

Bảng 4: Phân tích các phương án cho van nước điều khiển qua Z-Wave.

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.	Dễ dàng giao tiếp và tích hợp và hệ thống.	
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:



Hình 40: Sơ đồ khối của Van nước điều khiển qua Z-Wave.

Các khối chính bao gồm vi điều khiển điều khiển và quản lý trạng thái của van nước thông qua driver L293D và được Feedback trạng thái về thông qua các cảm biến dừng của của van nước khi đạt đến điểm biên. Van nước được điều khiển thông qua Z-Wave theo đường Serial, đồng thời để tăng tính mở rộng, board còn được tích hợp thêm module Wifi và RS485.

Khối nguồn được cung cấp bởi adapter 16V DC và được sử dụng IC AP3211 để hạ áp xuống các mức điện áp 12V và 5V mong muốn để điều khiển van nước.

Hệ thống driver điều khiển cũng như hệ thống cảm biến để kiểm soát trạng thái của van nước trong quá trình hoạt động được điều khiển bằng IC L293D, các cảm biến dừng được phản hồi trực tiếp vào vi điều khiển.

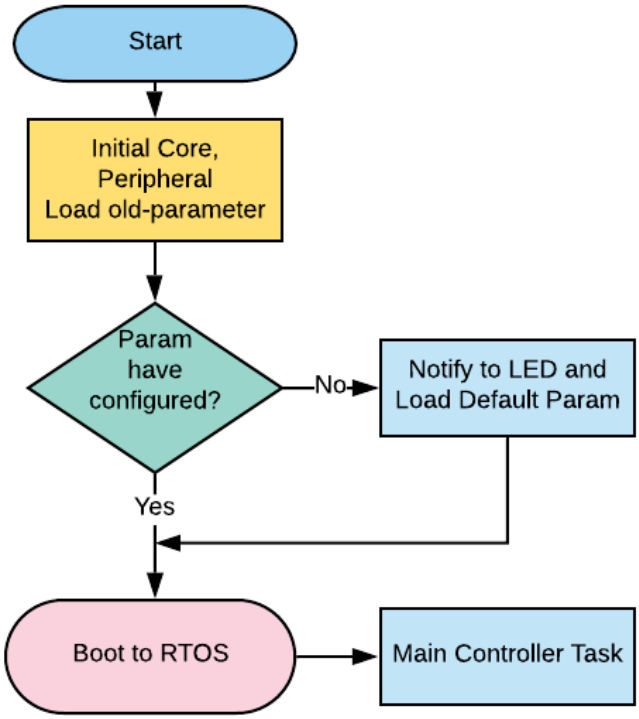
4. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM

4.1 Gateway Modbus Serial

Yêu cầu:

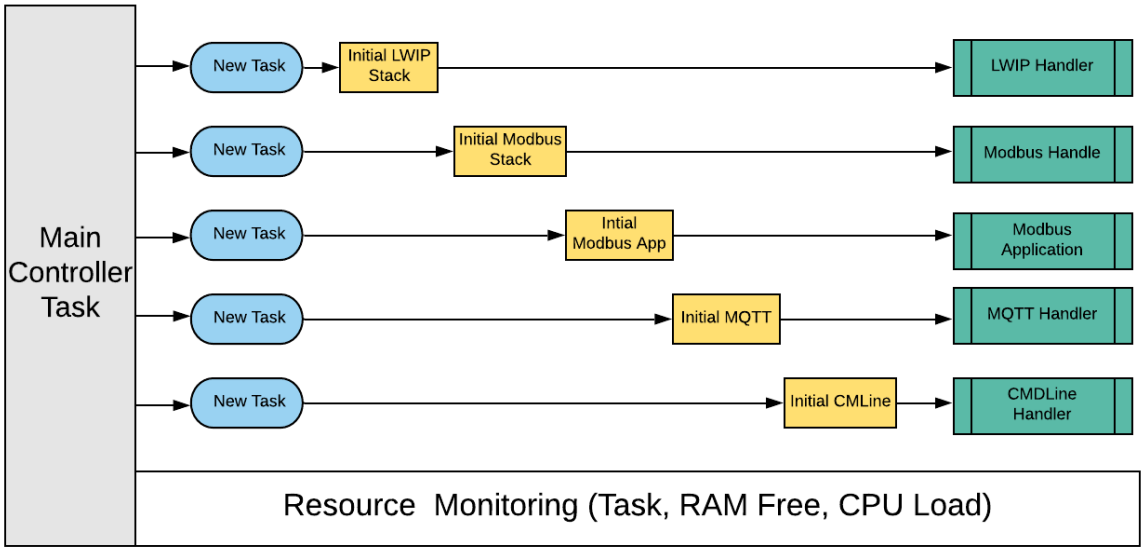
- Gateway phải chạy được Modbus Stack ở chế độ Master để thu thập, thay đổi, quản lý dữ liệu cho các thiết bị đầu cuối. Đồng thời để nâng cao tối đa số lượng thiết bị trong mạng, 4 cổng RS485 trên board đều phải được khai thác, có thể xem như Gateway quản lý 4 mạng Modbus riêng biệt.
- Gateway được kết 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 Server) 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 trang thái đường truyền và lỗi trong quá trình chạy thực nghiệm.

Lưu đồ giải thuật:



Hình 41: Quá trình khởi động của 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.



Hình 42: Các khối nhiệm vụ của Gateway Modbus Serial.

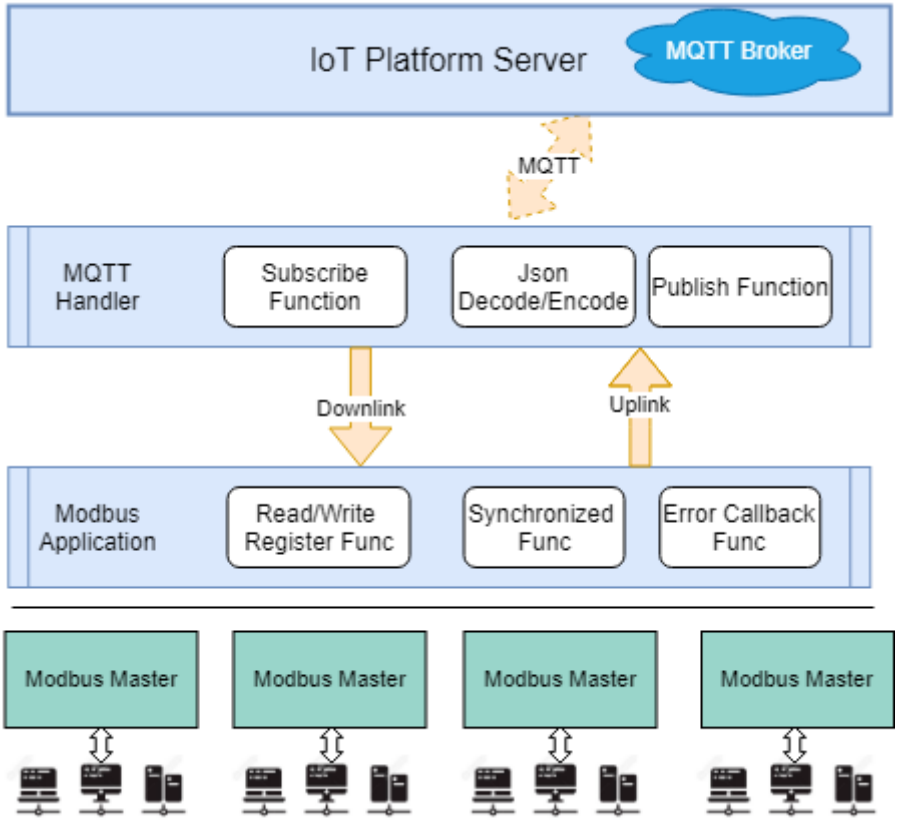
Sau khi boot thành công, Gateway sẽ tiến vào Main Controller Task – đây là task chính của Gateway, bao gồm các cơ chế tạo, quản lý các task sau này. Cũng trên Main Controller Task, các cơ chế monitoring tài nguyên hệ thống được triển khai và gửi lên người dùng thông qua Serial RS232.

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 và 1 Serial port), 300 bytes ram tuỳ theo các function được triển khai. 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.

Command Line: Giao diện command line thông qua Serial để cài đặt các thông số cho Gateway.



Hình 43: Luồng dữ liệu của Gateway Modbus Serial.

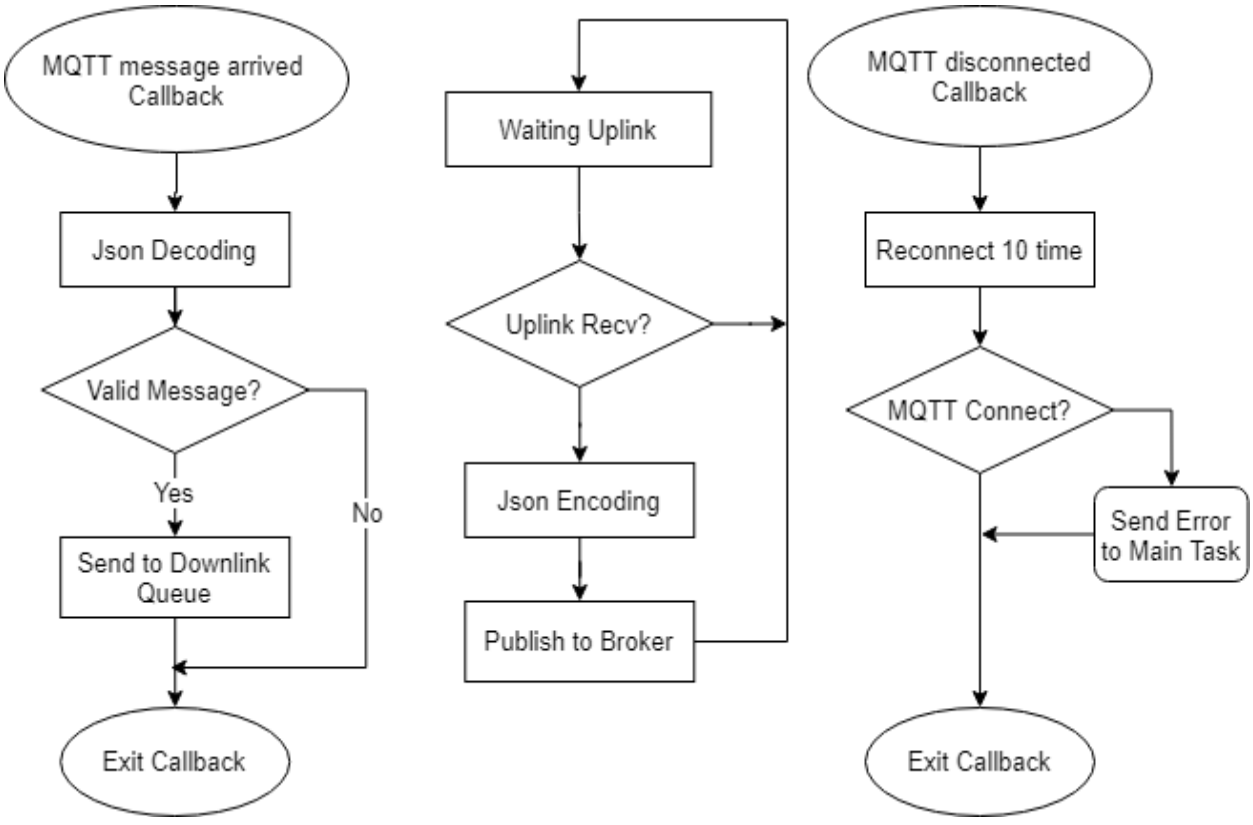
MQTT Handler: Trong quá trình khởi động, MQTT Handler sẽ khởi tạo dịch vụ MQTT dựa trên lớp TCP đã được cung cấp bởi thư viện LWIP khởi tạo trước đó. Sau đó, MQTT Handler sẽ tiến hành Subscribe vào Topic định danh trước (thống nhất với IoT Server và được lưu lại thông qua bộ nhớ Flash), dịch vụ MQTT sẽ tạo ra callback mỗi khi có gói tin được gửi đến Topic đã đăng ký, 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:

- Read Input Register (0x04)
- Read Holding Registers (0x03)
- Write Single Register (0x06)
- Write Multiple Registers (0x10)
- Read/Write Multiple Registers (0x17)
- Read Coils (0x01)
- Write Single Coil (0x05)
- Write Multiple Coils (0x0F)
- Read Discrete Inputs (0x02)
- 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.

Lưu đồ giải thuật chi tiết của MQTT Handler:

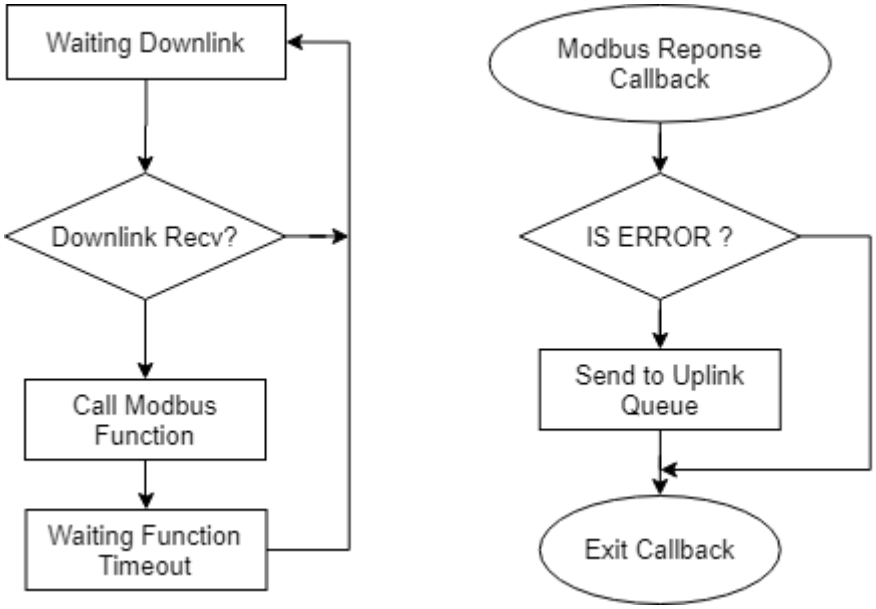


Hình 44: Lưu đồ giải thuật của khối MQTT Handler.

Giải thích:

- Khối MQTT Handler thiết lập kết nối đến Broker và đăng ký các Topic cần thiết. Từ đó tạo ra các luồng callback khi thoả điều kiện: nhận được gói tin MQTT và lỗi trong kết nối MQTT.
- Khi nhận được, gói tin sẽ được giải mã theo chuẩn JSON và kiểm tra tính hợp lệ. Nếu thoả điều kiện, các yêu cầu của gói tin sẽ được lấy ra và gửi xuống tầng Modbus App thông qua cơ chế hàng đợi của FreeRTOS.
- MQTT Handler sẽ chờ các phản hồi từ Modbus App và gửi lên MQTT Broker.

Lưu đồ giải thuật chi tiết của Modbus App:



Hình 45: Lưu đồ giải thuật của khối Modbus App.

Giải thích:

- Modbus App sẽ đợi yêu cầu từ MQTT Handler qua cơ chế hàng đợi, nếu nhận được yêu cầu, nó sẽ gọi các hàm tương ứng để Modbus Stack xử lý.
- Khi Modbus Stack phản hồi gói tin, dữ liệu sẽ được gửi lên MQTT Handler.

4.2 Gateway LoRaWAN và Z-Wave

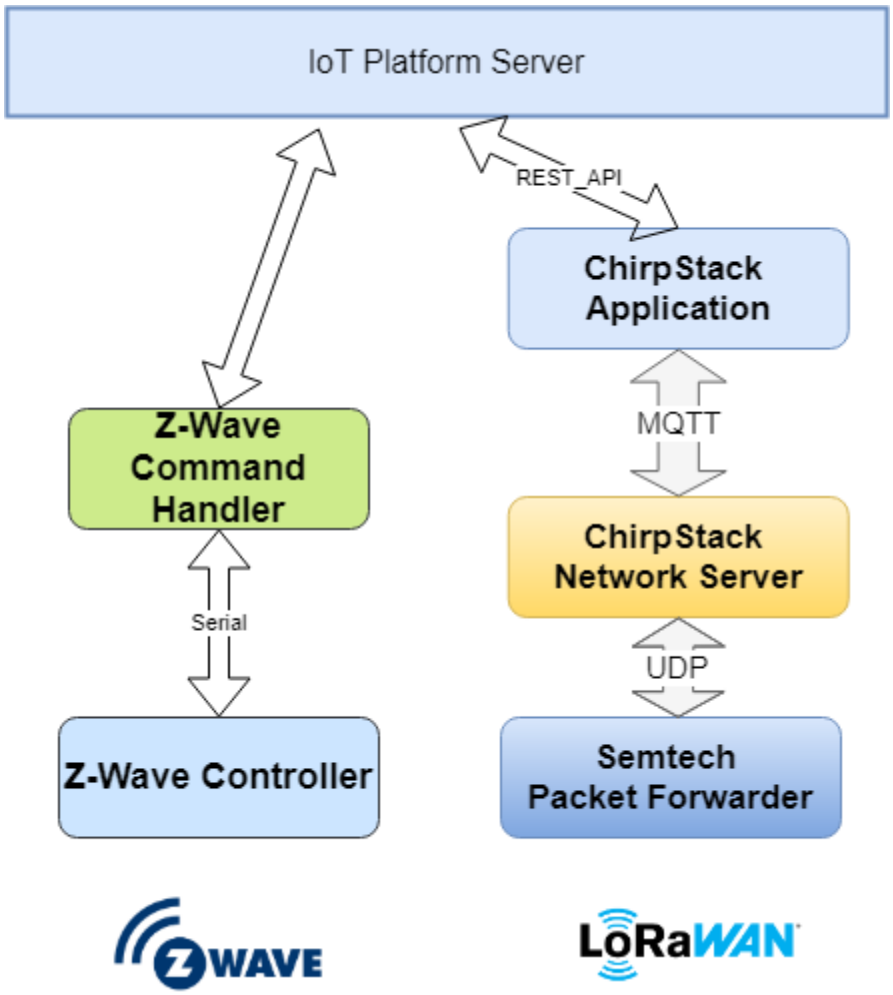
Yêu cầu:

Gateway sẽ thực hiện như một trạm trung chuyển, trong đó các luồng dữ liệu từ các thiết bị đầu cuối lên Server và ngược lại được đảm bảo.

Lưu đồ giải thuật:

Phần mềm trên Gateway đảm bảo luồng dữ liệu của 2 giao thức:

- LoRaWAN: triển khai LoRaWAN Network Server trên Gateway, thông qua đó Gateway sẽ quản lý lớp giao thức LoRaWAN, bao gồm việc thêm, quản lý các thiết bị và truyền nhận các gói tin. Gateway sẽ cung cấp API để Server có thể sử dụng các ứng dụng của LoRaWAN.
- Z-Wave: luồng dữ liệu từ Z-Wave sẽ qua đường nối tiếp Serial để kết nối đến Server.



Hình 46: Lưu đồ giải thuật tổng quát của Gateway LoRaWAN và Z-Wave.

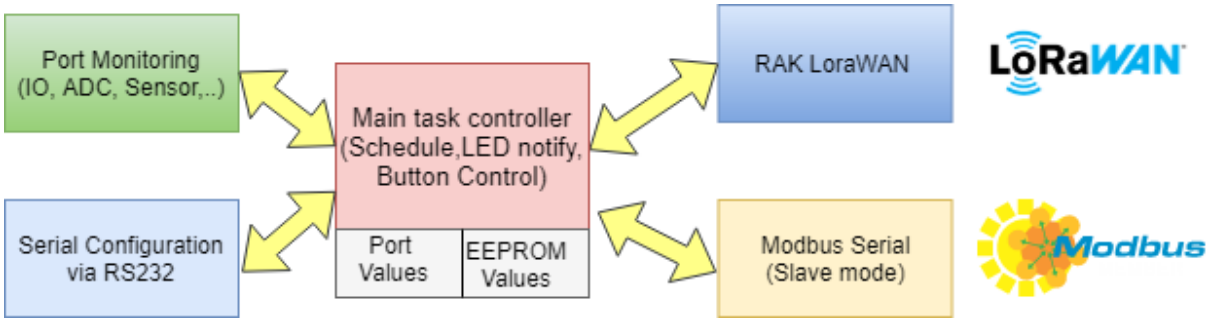
4.3 IoT Node

Yêu cầu:

- 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 chỉnh sửa cũng như quản lý node.

Lưu đồ giải thuật:

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.

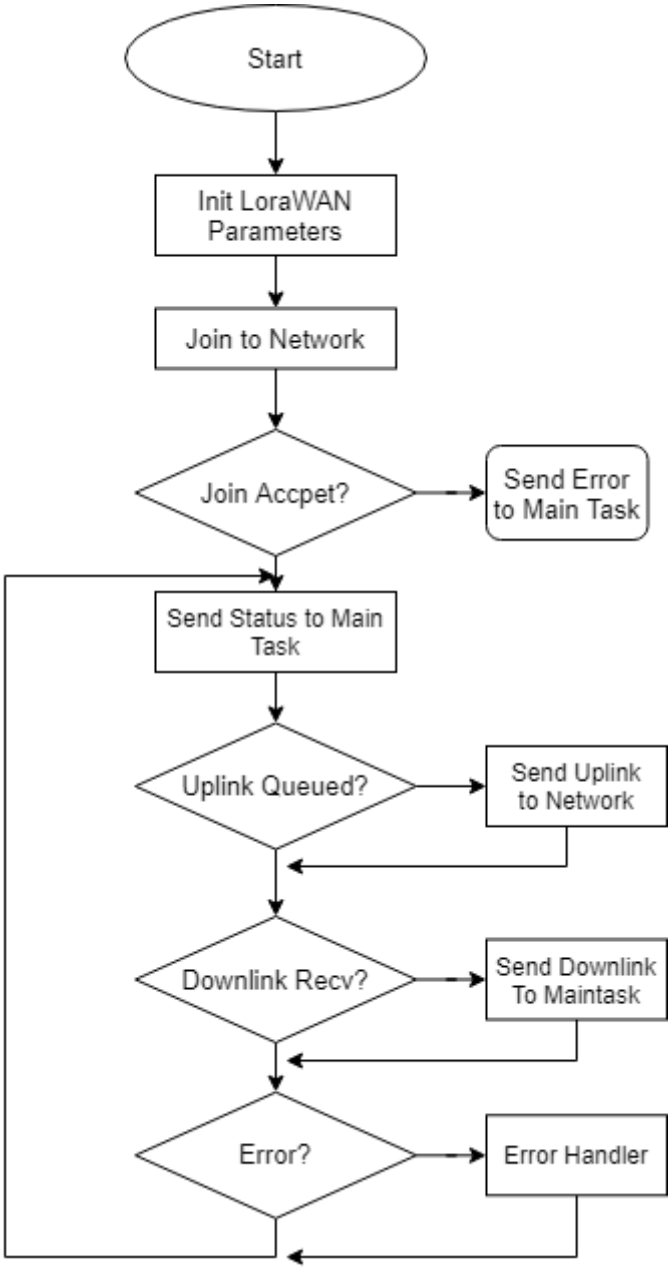


Hình 47: Các khối nhiệm vụ của Gateway Modbus Serial.

Các tác vụ chính trên IoT Node:

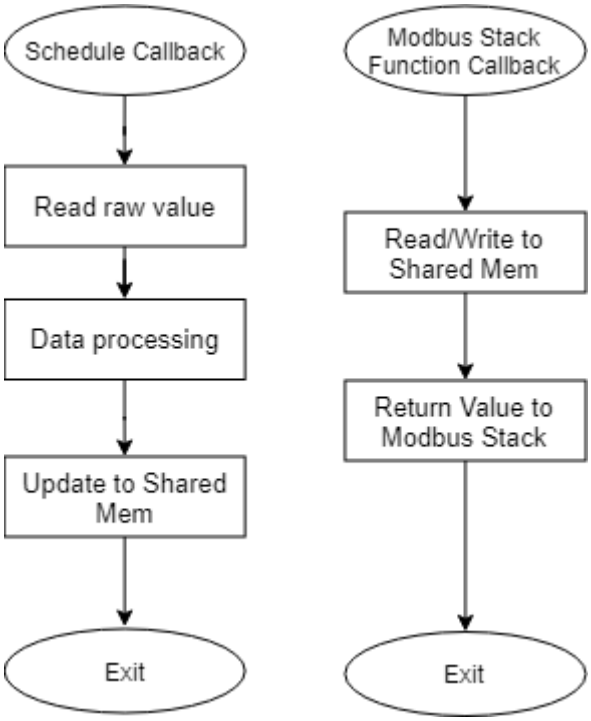
- Main task controller: Bao gồm các cơ chế đặt lịch, cài đặt cho các task xung quanh, quản lý về bộ nhớ dùng chung để cung cấp các vùng thanh ghi lưu dữ liệu hoạt động của node mà các task khác như Modbus và RAK có thể truy cập. Main Task còn cung cấp các giao diện tương tác bên ngoài như Led và nút nhấn.
- RAK LoraWAN: Đây là Task quản lý các truyền nhận dữ liệu thông qua giao thức LoraWAN mà cụ thể 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.



Hình 48: Lưu đồ giải thuật của khối RAK LoRaWAN.

- 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.
- 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.



Hình 49: Lưu đồ giải thuật của Port Monitoring và Modbus App.

4.4 Z-Wave Node

Z-Wave node hoạt động như một node độc lập: sử dụng các gói phát triển có sẵn được cung cấp bởi Silicon Labs để viết thêm tính năng, bao gồm các nhập xuất cơ bản, điều chế độ rộng xung.

Việc xây dựng các ứng dụng của Z-Wave Node được thực hiện bởi một sinh viên khác.

4.5 Van nước điều khiển qua Z-Wave

Yêu cầu:

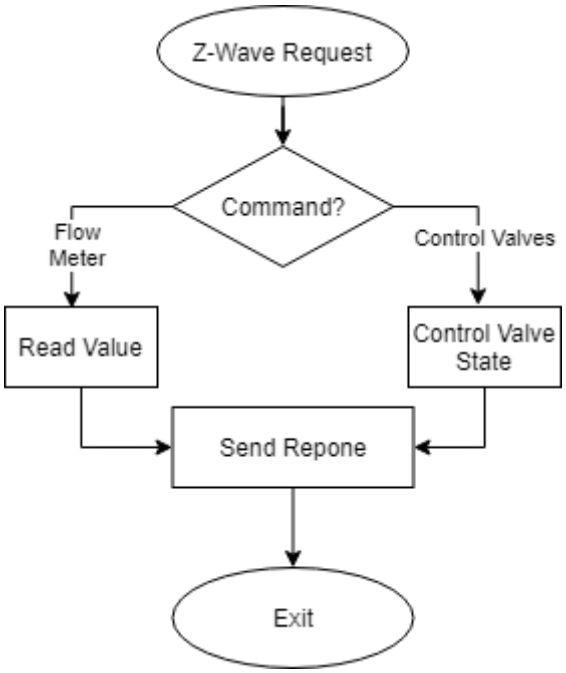
- Các chức năng đặc trưng của việc quản lý van nước, bao gồm đóng mở và quản lý trạng thái của các van nước điện, tốc độ dòng chảy.
- Giao tiếp với Z-Wave thông qua Command Handler, các tính năng kết nối vào mạng cũng như chấp hành và phản hồi trạng thái từ phía điều khiển (IoT Server).

Lưu đồ giải thuật:

Board van nước sẽ tiến hành kết nối vào mạng Z-Wave trong quá trình khởi động, đồng thời cho các van nước đã kết nối vào trạng thái đóng (trạng thái mặc định).

Khi kết nối thành công, Board có thể được điều khiển qua Z-Wave, trong trường hợp thất bại, Board vẫn có thể điều khiển thông qua các nút nhấn có sẵn.

Tác vụ chính của hệ thống sẽ chờ các Command yêu cầu từ mạng Z-Wave và xử lý đúng theo mục đích tương ứng: Điều khiển các trạng thai van nước, phản hồi trạng thái và trả về các giá trị liên quan.



Hình 50: Lưu đồ giải thuật chính của Board van nước điều khiển qua Z-Wave.

4.6 Phần mềm trên máy tính:

Yêu cầu:

- Cung cấp giao diện kết nối đến IoT Node thông qua cổng RS232 Serial Configuration.
- Cung cấp giao diện hiển thị thông tin và thay đổi các giá trị cài đặt trên Node.

Lưu đồ giải thuật:

Phần mềm đặt ra 2 luồng dữ liệu chính:

- Các lệnh Get, Set để lấy các thông tin từ IoT Node hiển thị lên giao diên và gửi các thông tin từ giao diện xuống Node.
- Các Reponse phản hồi từ Node tương ứng.

Hình 51: Luồng dữ liệu của phần mềm trên máy tính.

5. KẾT QUẢ THỰC HIỆN

5.1 Gateway Modbus Serial:

Phần cứng:

Dựa trên phần cứng có sẵn từ trước, các khối chức năng cần thiết hoạt động bình thường, bao gồm:

- Khối Ethernet có thể kết nối vào mạng giúp board có thể tuyển nhận dữ liệu qua Ethernet.
- Các cổng RS485 cho giao thức Modbus hoạt động ở chế độ haft-duplex.

Phần mềm:

Các kết quả đạt được:

- Tích hợp được hệ điều hành thơi gian thực FreeRTOS và thư viện LwIP lên Gateway giúp cho việc triển khai multitasking và sử dụng các dịch vụ liên quan đến mạng dễ dàng hơn.
- Tích hợp và tuỳ chỉnh FreeModbus ở chế độ 4 Master thành công lên nền tảng FreeRTOS.
- FreeModbus ở chế độ RTU, triển khai các Function chính: Read/Write Single/Multiple Register. Các cơ chế báo lỗi: lỗi gói tin, lỗi quá thời gian chờ, lỗi sai mã chức năng, …
- Tích hợp thư viện MQTT, kết nối thành công với Broker MQTT ở các QoS khác nhau.
- Triển khai cầu chuyển đổi giữa giao thức MQTT và Modbus, bao gồm 2 luồng dữ liệu chính là Downlink Request từ phía Server và Uplink Reponse từ phía Gateway.Các gói tin được chuẩn hoá theo kiểu JSON để dễ dàng tương tác với IoT Platform Server.
- Giao diện RS232 Serial Configuration để có thể thay đổi các giá trị khi cài đặt. Các giá trị sẽ được lưu vào vùng nhớ Flash của vi điều khiển.

5.2 Gateway LoRaWAN và Z-Wave

Phần cứng:

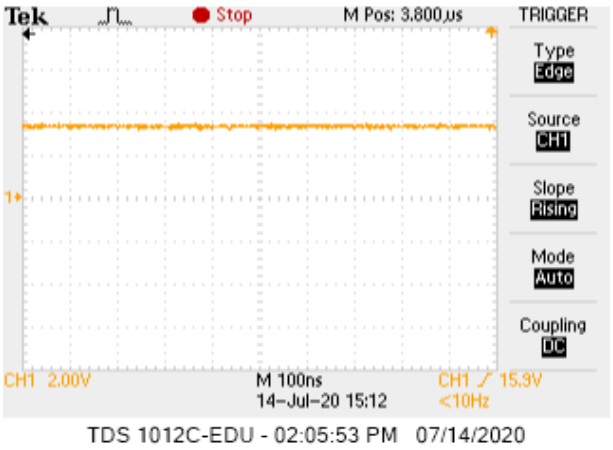
Thiết bị cơ bản chạy được các khối theo như yêu cầu đặt ra, bao gồm: Z-Wave Controller và LoRa Concentrator, trong quá trình lắp ráp lên board Rasberry Pi gặp một số vấn đề liên quan đến sai lệch chân Pin và đã được chỉnh sửa.

Hình 52: Kết quả thực hiện Gateway LoRaWAN và Z-Wave.

Khối nguồn cung cấp: các mức điện áp cho ra đúng với thiết kế đặt ra. Tiến hành khảo sát nguồn xung AP3211 cung cấp cho LoRa Concentrator ở mức điện áp 3V3.

Tiến hành khảo sát ở điều kiện:

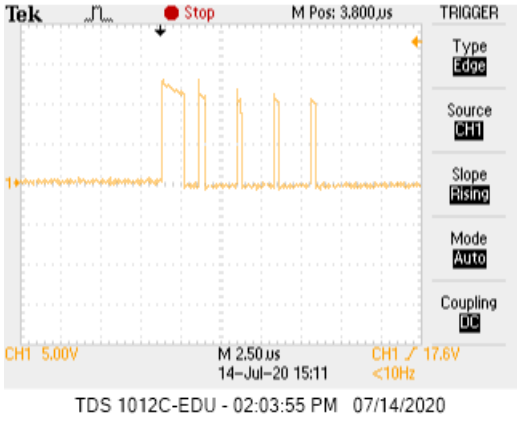
- Điện áp đầu vào 12VDC, điện áp đầu ra 3.3V
- Trở 3 Ohm. Từ đó suy ra dòng điện khoảng 1A.
- Thời gian: 60 phút.



Hình 53: Dạng sóng điện áp của nguồn xung AP3211 đo trên dao động ký Tektronix.

Kết quả: điện áp hoạt động ổn định, trong quá trình chạy không nhận thấy sự sụt áp hay gai nhiễu dựa trên tính năng trigger của máy đo, cuộn cảm có phần nóng lên do tải liên tục.

Tiến hành giảm trở tải, tăng dòng để kiểm tra tính năng Overcurrent detect, mạch chuyển sang chế độ ngắt, tần số hoạt động giảm xuống theo như datasheet nhà sản xuất, ở chân output của mạch có thể thấy được các xung detect.



Hình 54: Dạng sóng của điện áp của nguồn xung AP3211 khi bị ngắn mạch.

Khối LoRaWAN Concentrator: khối hoạt động bình thường khi cấp nguồn. Đèn nguồn sáng và có thể giao tiếp với Rasberry theo giao thức SPI. Trong quá trình hoạt động không thấy dấu hiệu bất thường nào.

Khối Z-Wave Controller: Khối hoạt động bình thường, có thể giao tiếp với Rasberry qua giao thức UART.

Cho đến thời điểm hiện tại, mạch hoạt động bình thường trong liên tục 8 tiếng.

Phần mềm:

Về RAK833/RAK831 Concentrator, phía Semtech đã đưa ra một bộ thư viện để máy tính nhúng có thể giao tiếp với Concentrator và thực hiện các tác vụ mong muốn như:

- util_sink: tạo giao diện lắng nghe và hiển thị các gói UDP.
- util_ack: tạo các gói phản hồi khi có các yêu cầu từ cổng UDP.
- util_tx_test: chạy thử tính năng gửi gói tin LoRa của Concentrator
- lora_pkt_fwd: gửi các gói tin LoRa nghe được từ Concentrator lên Server thông qua port UDP và ngược lại.

Trong các bài kiểm tra, về cơ bản máy tính nhúng có thể giao tiếp SPI và thu phát tín hiệu LoRa thông qua Concentrator. Thư viện cũng hỗ trợ các các cài đặt thông số cho Concentrator dưới dạng các file chuẩn JSON, việc triển khai hệ thống đơn giản hơn khi áp dụng các configuration có sẵn quy hoạch các thông tin về tần số, năng lượng, băng thông và các thông số liên quan đến việc triển khai LoRaWAN cho từng vùng.

Việc porting thư viện sang các máy tính nhúng cũng khá đơn giản, thư viện tương thích với các SPI Device driver trên Linux, do đó chỉ cần cung cấp cho thư viện Device file và các thông số liên quan đến SPI tương ứng. Thư viện chuẩn cung cấp bởi Semtech được cung cấp tại https://github.com/Lora-net/packet_forwarder.

Tương tự với Z-Wave Controller, cần mở Serial Port có sẵn trên Raspberry để lớp ứng dụng phía trên có thể tương tác với Controller. Trong các bài kiểm tra, Controller tương thích với các thiết bị bên ngoài từ các nhà sản xuất như Leviton, AEOTEC, … và các thiết bị Z-Wave tự thiết kế trong đề tài.

5.3 IoT Node

Phần cứng:

Sau khi đặt mạch in và thi công theo thiết kế đặt ra, nhìn chung mạch hoạt động được tất cả các khối.

- Khối nguồn: Hoạt động ở dãy điện áp từ 6V-18V. Các mức điện áp yêu cầu là 5V và 3.3V đúng với thiết kế đặt ra. Nguồn cung cấp điện áp cho các linh linh kiện hoạt động bình thường trong dãy điện áp này.
- Vi điều khiển: Hoạt động bình thường ở chế độ dùng thạch anh 12Mhz ngoài. Quá trình nạp, xoá, gỡ lỗi chip không thấy có vấn đề bất thường.
- Khối đầu vào Opto cách ly và đảo tín hiệu qua IC 74HC14 hoạt động đúng với thiết kế, qua khảo sát cho các tín hiệu vào thu đuọc tín hiệu đảo ngược ở trước ngõ vào vi điều khiển.
- Khối đầu vào tín hiệu tương tự cho kết quả đọc từ vi điều khiển đúng
- Khối
- Khối RAK LoRaWAN hoạt động bình thường, giao tiếp thành công với máy tính thông qua UART.
- Các khối RS232, RS485, nút nhấn, đèn Led hoạt động đúng với thiết kế.

Hình 55: IoT Node.

Phần mềm:

Việc xây dựng hệ thống trên nền tảng RTOS góp phần làm đơn giản hoá các chức năng đặt ra, các kết quả đạt được:

- Dữ liệu cảm biến bên ngoài được đọc và xử lý lưu lại vào vùng nhớ tuỳ theo profile đặt ra.
- Tích hợp FreeModbus ở chế độ Slave trên nền tảng FreeRTOS. Dữ liệu đặt ra trong các vùng nhớ bao gồm các giá trị cảm biến thu thập được
- Cơ chế thiết lập kết nối vào mạng LoRaWAN và gửi định thời các giá trị cảm biến.
- Giao diện RS232 Serial Configuration để có thể thay đổi các giá trị khi cài đặt. Các giá trị sẽ được lưu vào EPPROM.

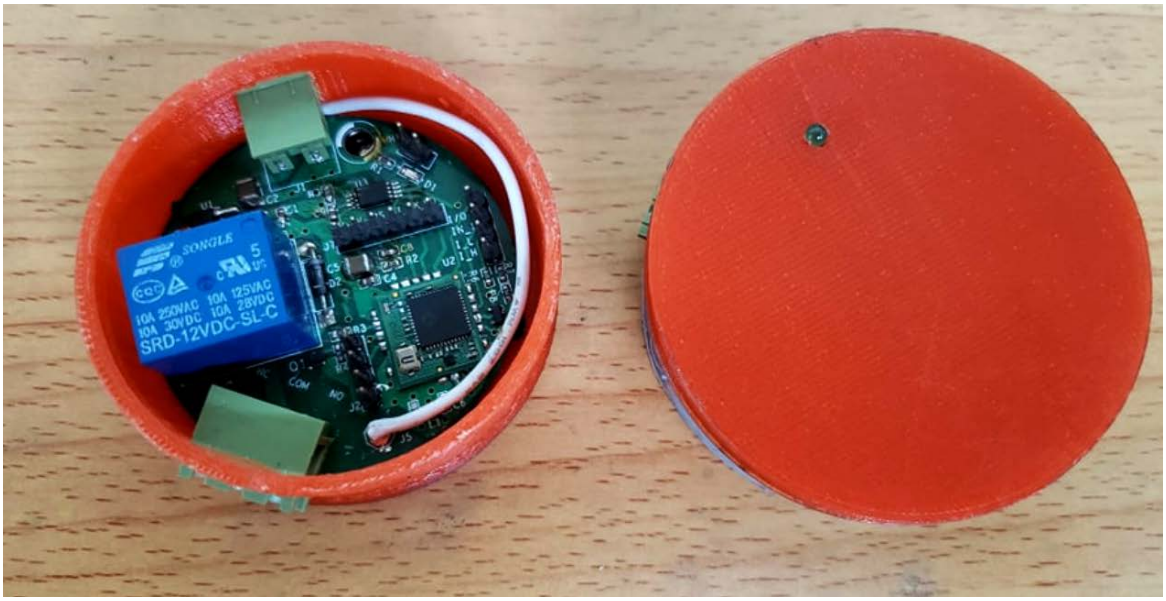
5.4 Z-Wave Node

Phần cứng:

Board mạch hoạt động ở mức điện áp từ 5V-12V. Do đó có thể dễ dàng tích hợp vào các thiết bị để kết nối các thiết bị vào mạng Z-Wave. Trong khảo sát, board mạch được kết nối vào một cảm biến chuyển động sau khi đã điều chỉnh thiết kế. Ngoài ra board mạch cũng có thể hoạt động như một node độc lập để điều khiển relay làm công tắc

Phần mềm:

- Board kết nối (include) thành công vào Controller đã thiết kế.
- Board có thể làm một công tắc điều khiển thiết bị 220V khi hoạt động như một công tắc không dây.
- Board có thể là một node thu thập các tín hiệu cảm biến, kết hợp các thiết bị có sẵn như một pack mở rộng kết nối Z-Wave.



Hình 56: Z-Wave node.

5.5 Van nước điều khiển qua Z-Wave

Phần cứng:

Board hoạt động được ở điện áp 12V-16V. Các khối chính bao gồm cầu H điều khiển van nước điện, Z-Wave và vi điều khiển hoạt động bình thường.

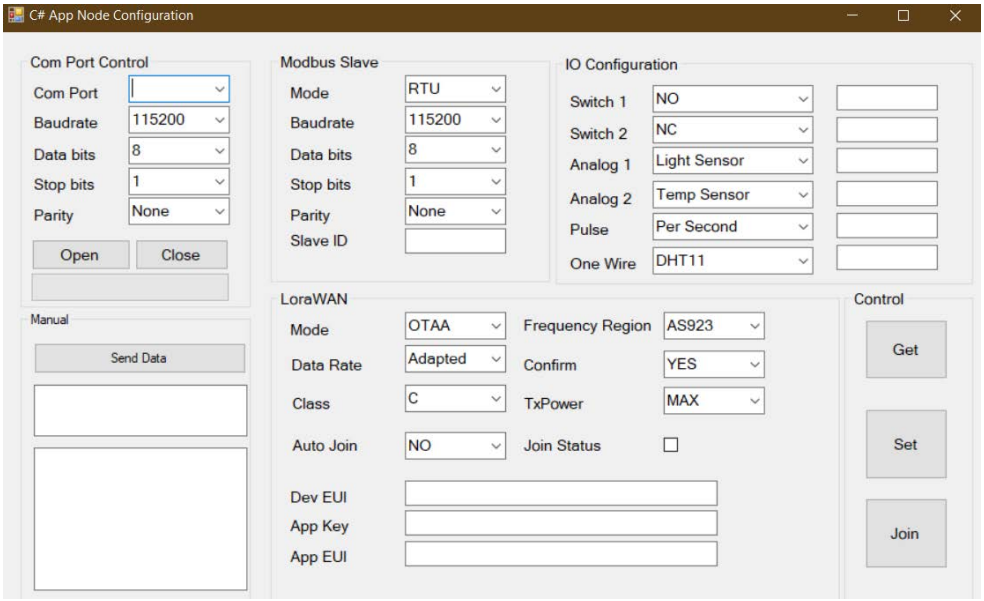
Hình 57: Board điều khiển van nước thông qua Z-Wave.

Phần mềm:

- Board điều khiển van nước có thể kết nối vào mạng Z-Wave Controller của Gateway.
- Board điều khiển van nước thực hiện được các chức năng cơ bản để quản lý và điều khiển toàn bộ 4 van thông qua Z-Wave.

5.6 App trên máy tính

Phần mềm hoạt động trên máy tính chạy đúng theo các yêu cầu đặt ra.



Hình 58: Giao diện App trên máy tính.

5.7 Các bài thử nghiệm, đánh giá

5.7.1 Giao thức Modbus

5.7.1.1 Modbus trên Gateway (Modbus Master)

Các bài thử nghiệm đặt ra:

Thử nghiệm tương thích với thiết bị Modbus có sẵn:

Trong phần này, Gateway sẽ được kết nối với biến tần Delta VDF EL [trích dẫn] có hỗ trợ Modbus và phần mềm Modbus Slave giả lập thiết bị modbus thông qua cổng Serial

Kết quả: Gateway có thể giao tiếp với biến tần thông qua các function: Read/Write Single/Multiple Holding Registers.

Thử nghiệm tương thích với các Slave Modbus trong đề tài:

Trong phần này, Gateway sẽ kết nối trực tiếp đến 8 thiết bị ở chế độ Slave Modbus (IoT Node ở chế độ Modbus). Việc thử nghiệm sẽ được thực hiện bằng cách Polling các thanh ghi giá trị từ tất cả các Node trong mạng.

Hình 59: IoT Node kết nối vào Gateway Modbus Serial.

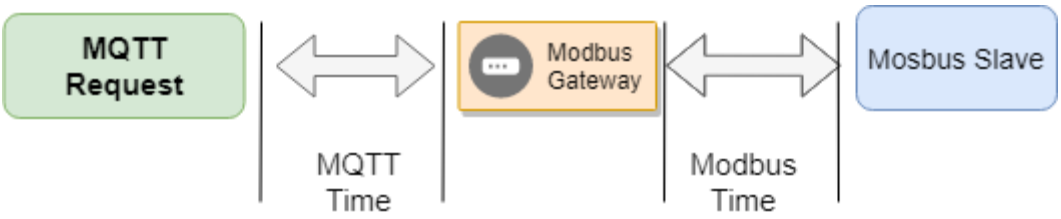
Kết quả: Mạng Modbus hoạt động ổn định, các Slave kết nối vào mạng với từng ID khác nhau và không có hiện tượng xung đột trong việc gửi nhận gói tin.

Thử nghiệm tốc độ tối đa:

Mô hình thời gian trễ của luồng dữ liệu:

- MQTT Time: Thời gian của giao thức MQTT, bao gồm thời gian gói tin Downlink Request được gửi đến Gateway thông qua trung gian Broker và thời gian gói tin Uplink Reponse phản hồi đến nơi yêu cầu. Trong bài kiểm tra, gói tin kiểm tra sẽ không được gửi xuống thiết bị Slave mà sẽ phản hồi ngay lập tức về Server khi đi đến Gateway.

- Modbus Time: Thời gian của giao thức Modbus từ khi bắt đầu nhận Request được yêu cầu đến khi trả về gói tin phản hồi từ Slave. Trong bài kiểm tra, thời gian này được tính bằng thời gian tổng gói tin đi trong mạng trừ MQTT Time.



Hình 60: Mô hình thời gian gói tin đi trong mạng Modbus.

Các thông số trong bài kiểm tra:

- Mô hình mạng LAN: 100Mbps, Asus Router RT-AC1300UHP.

- MQTT Broker: Mosquitto on Rasberry Pi, LAN 100Mbps.

- MQTT Client: Gateway Modbus with LAN8720A at 100Mbps

- Modbus: RTU Mode, Serial: 9600, 8, N, 1. Timeout: 3000ms.

Kết quả:

- Thời gian MQTT Time thu được:

No	QoS	Tổng số gói	Số gói lỗi	Latency (ms)
1	0	100	0	5968
2	1	100	0	24876
3	2	100	0	27563
4	0	1000	0	62000
5	1	1000	0	25223

6	2	1000	0	28642

Bảng 5: Bảng thống kê về thời gian trễ của giao thức MQTT.

Đánh giá: thời gian latency cao, việc thiết đặt QoS lên 1 và 2 làm tăng đáng kể độ trễ, nguyên nhân tạm thời có thể xem là do việc triển khai MQTT dưới vi điều khiển chưa thật sự là một giải pháp tốt. Bên cạnh đó, MQTT Broker của Mosquitto trên Rasberry Pi cũng gây ra độ trễ nhất định.

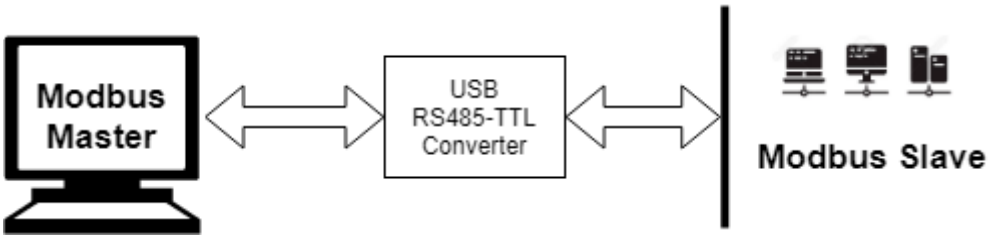
- Thời gian tổng của gói tin đi trong mạng:

Bảng 6: Bảng thống kê tổng thời gian gói tin đi trong mạng Modbus và MQTT.

No	QoS	Tổng số gói	Số gói lỗi	Latency (ms)
1	0	100	0	9604
2	0	100	2	1206
3	0	100	3	1488

5.7.1.2 Modbus trên Node (Modbus Slave)

Trong mục tiêu đặt ra, Node được xem như là Modbus Slave, để kiểm ra độ tương thích với tiêu chuẩn Modbus bên ngoài, phần mềm Modbus Poll được xem như là phần mềm giả lập Modbus Master chạy trên máy tính. Qua đó, việc kết nối với IoT Node thành công có giá trị để đánh giá độ tương thích của IoT Node với các thiết bị Modbus bên ngoài.



Hình 61: Mô hình kiểm tra tương thích của Modbus Slave.

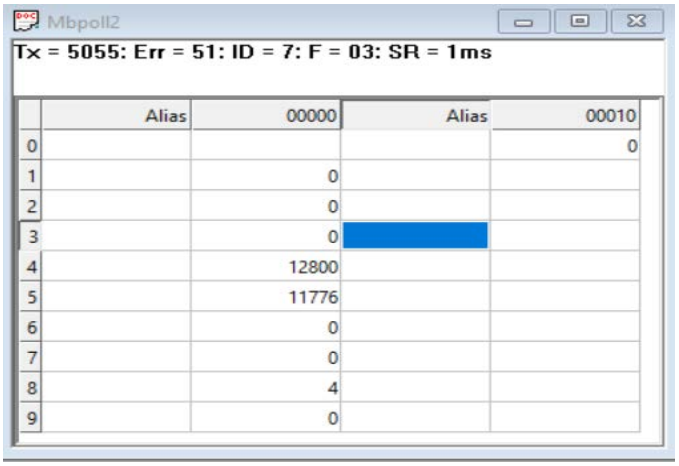
Thử nghiệm với mạng đơn thiết bị:

Thực hiện việc đọc và ghi liên tục các gói tin trong mạng Modbus. Trong bài kiểm tra, thời gian trễ của các gói tin phần lớn phụ thuộc vào tốc độ Baudrate đường truyền.

Các thông số bài kiểm tra:

- Modbus Slave: RTU Mode, Serial 9600, 8, N1. Time out: 2000ms.

- Function: Read Single/Multi Register. Write Single Resigter.



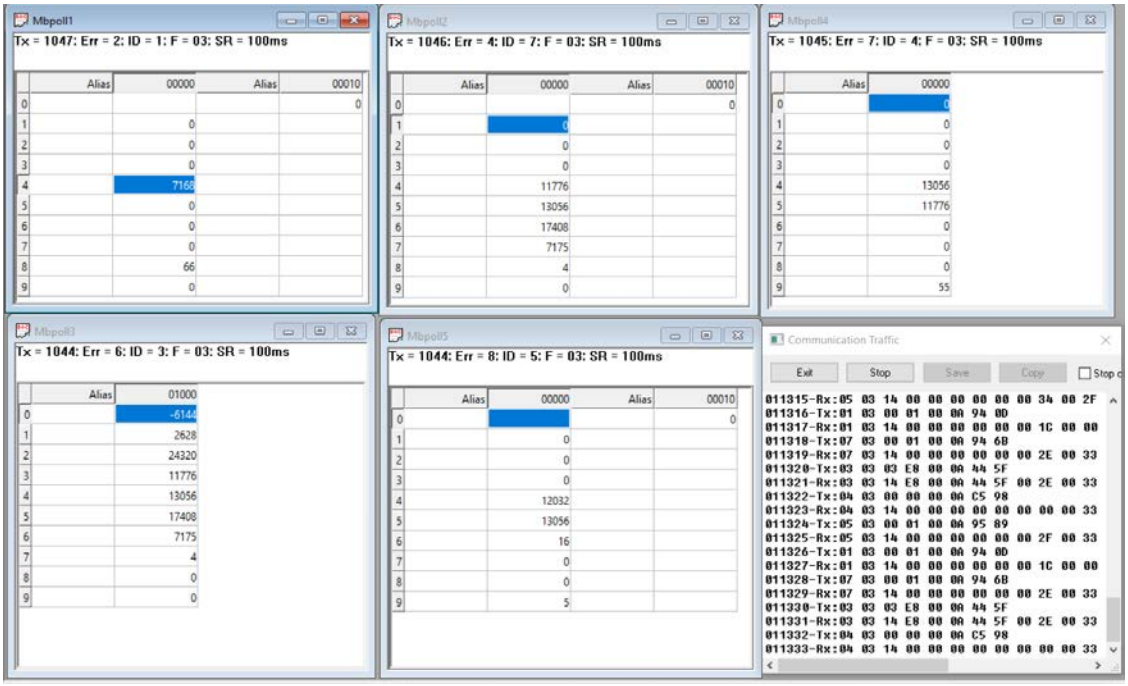
Hình 62: Giao diện Modbus Poll khi thử nghiệm với đơn thiết bị.

Kết quả thu được: Thiết bị tương thích với mạng Modbus được giả lập bởi phần mềm Modbus Poll. Tổng số gói yêu cầu là 5000 và lỗi ở 5 gói.

Thử nghiệm với mạng nhiều thiết bị:

Thực hiện đọc ghi lần lượt và liên tục tất cả thiết bị trọng mang Modbus. Các thông số bài kiểm tra:

- Modbus Slave: RTU Mode, Serial 9600, 8, N1.
- Function: Read Single/Multi Register. Write Single Resigter.



Hình 63: Giao diện Modbus Poll khi thử nghiệm với nhiều thiết bị.

Kết quả: nhìn chung các thiết bị hoạt động bình thường và không xung đột lẫn nhau, tuy nhiên vẫn có các gói bị lỗi.

Bảng 7: Bảng thống kê kết quả phản hồi của Modbus Slave trong mạng đa thiết bị.

ID	1	2	3	4	5
Total Request	1047	1016	1045	1044	1044
Total error	2	4	3	4	8
Error Type	Time Out	Time Out	Time Out	Time Out	Time Out

Nhận xét: qua các quá trình kiểm tra và tìm hiểu, các gói tin quá thời gian không phản hồi do bị lỗi thông qua đường truyền và bị loại khỏi hàng đợi bởi FreeModbus.

5.7.2 Giao thức LoraWAN

Mạng LoRaWAN được triển khai dưới các thiết bị phần cứng đã thi công, bên cạnh đó, để triển khai được mạng LoraWAN, cần phải có một server LoRaWAN để quản lý các nghi thức mà protocol của LoRaWAN Alliance đặt ra. Trong đề tài này, ChirpStack LoRaWAN Network Server stack [10] sẽ được triển khai ở mạng nội bộ.

Ngoài ra việc triển khai mạng LoRa cũng như LoRaWAN cần phải tuân theo quy hoạch tần số vô tuyến điện của Cục Tần Số cũng như các Limitation của LoRaWAN Alliance.y

Các thông số cài đặt LoRaWAN:

- Server: ChirpStack Network Server trên RasberryPi
- Gateway: RAK831 và RAK833 LoRa Concentrator.
- Node: IoT Node với RAK811 và RAK4600.
- LoRaWAN Version: 1.0.2
- Frequency: AS923
- Retry: 5.
- Max TxPower: 14dbm.
- Adaptive Datarate: On.
- Antenna: GSM 900Mhz 2db/6db gain, 915Mhz LoRa Antenna.
- Các giá trị liên quan khác được đặt mặc định theo LoRa Alliance ở vùng AS923.

Các bài thử nghiệm:

Thử nghiệm kết nối và gửi dữ liệu của Node vào mạng LoraWAN:

Từ lý thuyết đặt ra cũng như tiêu chuẩn của LoRaWAN Alliance, các thông số cần chuẩn bị để kết nối Node vào mạng bao gồm: DevEUI, AppKey, AppEUI ở mode OTAA và DevID, Network Session Key, AppSKey đối với mode APB. Các thông số này có thể lấy sẵn từ Node thông qua app giao diện trên máy tính đã viết và phải được đăng ký vào LoRaWAN server để Node được định danh và đảm bảo các gói tin được mã hoá, giải mã hợp lệ. Bên cạnh đó, cần cài đặt Node ở dãy tần số đúng với mô hình để các gói tin được truyền nhận chính xác.

Kết quả: trong quá trình thử nghiệm, các Node kết nối vào Server thành công theo cả 2 mode ABP và OTAA. Khảo sát 2 luồng dữ liệu chính:

Uplink:

- Gửi các gói tin gửi đến được Network Server.
- Các giới hạn về LoraWAN Limitation, bao gồm DwellTime, công suất phát, độ trải phổ làm ảnh hưởng đến khoảng cách.
- Có gói xác nhận từ phía Network Server khi nhận được gói và tính năng tự động thử gửi lại gói tin khi không nhận được xác nhận.

Downlink:

- Nhận thành công trên 2 loại thiết bị class A và class C
- Cơ chế gửi gói xác nhận từ node về Network Server không hoạt động.

Thử nghiệm 8 node trong mạng LoraWAN:

Theo công bố nhà sản xuất RAK Wireless, RAK831/RAK833 Concentrator có thể quản lý đồng thời 500 thiết bị trong mạng bán kính 1km^2. Tuy nhiên trong giới hạn về kinh phí cũng như khối lượng công việc, đề tài chỉ thực hiện 8 thiết bị tham gia đồng thời vào mạng. Mặc dù không đánh giá và kiểm định được khả năng quản lý số lượng lớn thiết bị của LoRaWAN, nhưng đây là cơ sở cơ bản nhất để thể hiện khả năng quản lý đa thiết bị của nó.

Kết quả: Quá trình đăng ký và kết nối vào mạng hoạt động bình thường, các gói gửi giá trị lên một cách ổn định.

Last seen	Device name	Device EUI	Device profile
a few seconds ago	iotNodeinbox	60c5a8fffe798363	device_profile_class
a few seconds ago	iotNodenobox	60c5a8fffe798377	device_profile_class
a few seconds ago	iot_node_rak4600_id5	60c5a8fffe754f08	device_profile_class
a few seconds ago	iot_node_rak4600_id6	60c5a8fffe754f0b	device_profile_class
a few seconds ago	iot_node_rak4600_id7	60c5a8fffe754665	device_profile_class
a few seconds ago	iot_node_rak460_id2	60c5a8fffe754f0a	device_profile_class
a few seconds ago	iot_node_rak811_id3	60c5a8fffe000002	device_profile_class
a few seconds ago	rak4600_newFW_test	60c5a8fffe754ee6	device_profile_class

Hình 64: Giao diện quản lý thiết bị trên ChirpStack Network Server.

Tính năng Adaptive Data Rate (ADR):

ADR là chức năng tự động tối ưu về tốc độ, thời gian truyền sóng, năng lương tiêu tốn của thiết bị trong mạng dựa theo chất lượng tín hiệu gửi và thu được.

Kết quả: trong quá trình thực nghiệm ở các khoảng cách khác nhau, các có sự thay đổi về tần số và độ trải phổ tuỳ theo khoảng cách và năng lượng phát từ khi kết nối đến khi thiết bị đã ổn định. Qua đó cho thấy tính năng ADR hoạt động trên hệ thống.

Tính năng mở rộng mạng bằng cách việc sử dụng nhiều Gateway:

Tất cả các gói tin được nghe bởi Gateway và gửi lên LoRaWAN Network Server, nếu không phải là gói từ các thiết bị mong muốn đã có trong mạng, Server sẽ loại bỏ gói. Server chấp nhận nhiều Gateway đăng ký vào nó, do đó mạng LoRaWAN có thể mở rộng nếu có nhiều Gateway kết nối vào Server. The Things Network [11] là một ví dụ mở rộng mạng LoRaWAN ở mức độ toàn cầu. Một gói tin từ thiết bị gửi lên có thể được thu bởi nhiều Gateway, tuy nhiên, gói tin phản hồi về thiết bị chỉ được gửi bởi Gateway có mức thu tín hiệu từ thiết bị tốt nhất.

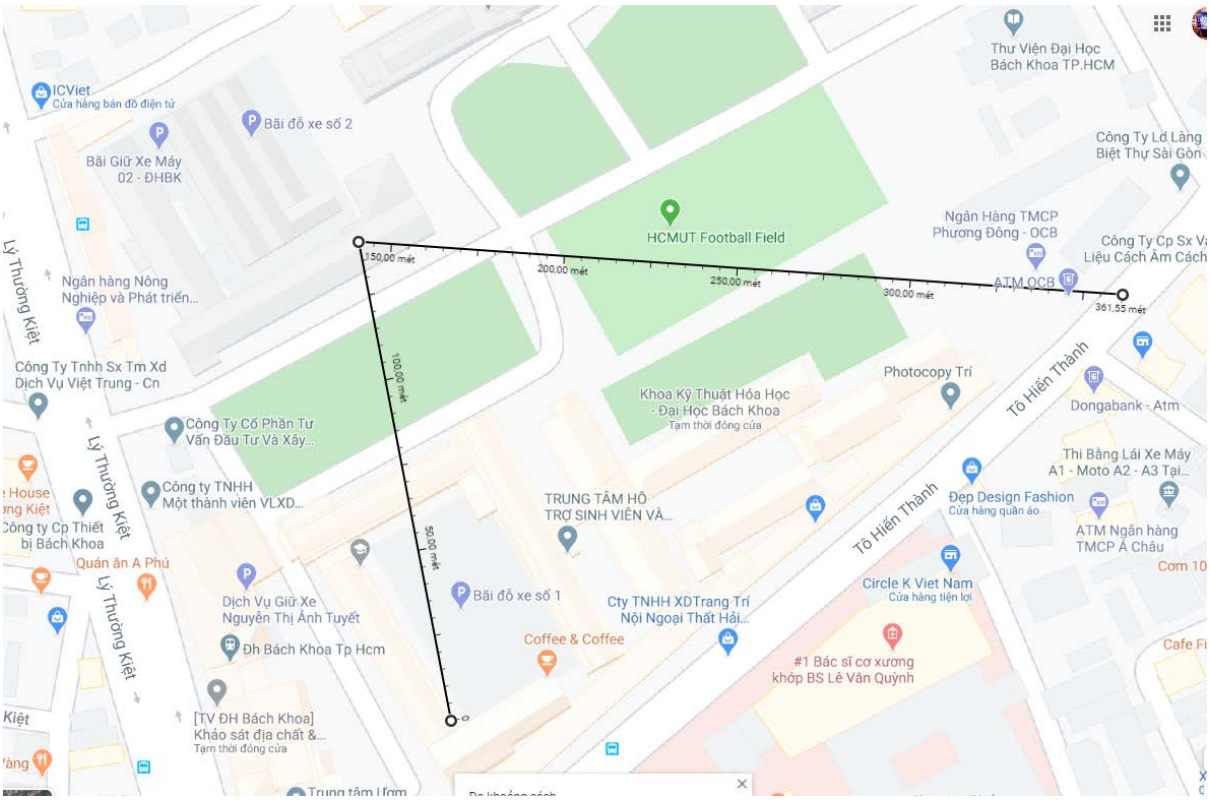
Kết quả: Trong đề tài sử dụng 2 Gateway, qua quá trình kiểm tra các gói tin nhận được từ ChirpStack cung cấp, có thể thấy được trường hợp gói tin được nhận bởi cả 2 Gateway, qua việc xử lý của Network Server, Gateway có tín hiệu tốt nhất sẽ được chọn để gửi gói tin phản hồi về Node.

Thử nghiệm về khoảng cách:

Các thông số đo đạc được trong môi trường bên ngoài có thể chênh lệch so với số liệu nhà sản xuất công bố. Trong quá trình thử nghiệm, đề tài tiến hành kiểm tra thử trong 2 loại môi trường thu phát khác nhau:

- Môi trường Building: tín hiệu có thể bị cản bởi các bức tường, toà nhà, điều kiện khảo sát tại khuôn viên trường Đại học Bách Khoa. Gateway đặt trong phòng thí

nghiệm ở độ cao khoảng 5 mét.



Hình 65: Kết quả khoảng cách thu được đo trên ứng dụng Google Map tại Đại Học Bách Khoa.

- Môi trường Line of sight: môi trường ít vật cản, thoáng, trong điều kiện của đề tài, thực hiện khảo sát ở vùng quê, Gateway được đặt cao 1 mét so với mặt đất.

Hình 66: Kết quả khoảng cách thu được đo trên ứng dụng Google Map tại vùng quê.

Theo chuẩn LoRaWAN, các gói tin được đảm bảo tính toàn vẹn thông qua các gói tin xác nhận. Nếu gói tin không được xác nhận, Node sẽ thử lại cho đến khi nhận được phản hồi hoặc quá số lần thử lại đã được cài trước. Trong bài thử nghiệm, số lần thử lại cài trong Node là 5, việc gói tin sau 5 lần thử lại và không có tín hiệu xác nhận được xem như gói tin bị lỗi và Node sẽ dừng việc gửi gói.

Bảng 8: Bảng thống kê kết quả đo đạc khoảng cách truyền nhận của LoRaWAN

Điều kiện	Khoảng cách	Tổng số gói	Số gói lỗi	RSSI	SNR
Building	100m	10	0	-80	4.2
Building	200m	10	2	-103	-8
Building	300m	10	6	-114	-9.8
Line of Sight	100m	10	0	-61	5.6
Line of Sight	300m	10	0	-75	2.2
Line of Sight	450m	10	0	-80	4
Line of Sight	600m	10	2	-94	-8
Line of Sight	900m	20	18	-110	-15

Nhận xét: khoảng cách thu được là tương đối còn thấp so với công bố nhà sản xuất cũng như các ứng dụng LoRa có trên thị trường. Tuy nhiên thông qua việc thực nghiệm và trao đổi trên cộng đồng:

- Việc sử dụng các thông số theo tiêu chuẩn AS923, bao gồm các dãy tần số từ 920Mhz đến 925Mhz, các giới hạn của LoRaWAN Limitation làm giảm một phần khoảng cách thu nhận vật lý cung cấp bởi lớp sóng LoRa.
- Giải pháp antenna hiện tại chưa hợp lý so với tần số triển khai, đồng thời các antenna đang dùng chưa được tính toán phối hợp trở kháng cũng như đo đạc hợp lý.

- Can nhiễu từ các ứng dụng khác khi ở dãy 920Mhz và 925Mhz, đặc biệt là GSM. Tần số 890Mhz-915Mhz là đường Uplink từ các thiết bị điện thoại phổ thông và từ 925Mhz-960Mhz từ trạm vô tuyến gửi Downlink, theo tham khảo, các tín hiệu này được phát với công suất 20W (gấp hơn 1500 lần công suất của các Node đang triển khai).

Hình 67: Dãy tần số dành cho GSM.

- Điều kiện đo đạc chưa thật sự tốt: đối với môi trường Line of Sight, cần khảo sát tại các vùng biển, vùng núi để có kết quả tốt nhất.

Dòng tiêu thụ của Node:

Khảo sát trên module LoRaWAN:

Bảng 9: Bảng thống kê dòng điện tiêu thụ của module RAK trên IoT Node

Vcc = 3.3V, Temp = 25°C	Tx	Class C Idle	Class A Idle	Sleep
RAK4600 (NRF52832)	122mA	27mA	7mA	3uA
RAK811 (STM32L1)	120mA	25mA	6mA	0.6uA

Khảo sát trên toàn Node:

Do giới hạn về thời gian, node chưa được tuỳ chỉnh để vào chế độ tiết kiệm pin, ở điện áp 12V, dòng trung bình của Node tiêu thụ 20mA. Qua đó cho thấy chưa thể triển khai ứng dụng dùng pin lâu dài.

5.7.3 Giao thức Z-Wave

Các thông số trong bài thử nghiệm: Z-Wave hoạt động, bài khảo sát thực hiện các tính năng gửi nhận dữ liệu binary từ các node về Controller và ngược lại.

Thử nghiệm độ tương thích của Z-Wave Controller.

Z-Wave Controller tương thích tốt với các sản phẩm bên ngoài thị trường (Leviton DZ15S, AEOTEC ZW112-A) và các sản phẩm thi công trong đề tài.

Thử nghiệm khoảng cách.

Giao thức Z-Wave là một mạng mesh kết nối nhiều thiết bị, do đó khoảng cách có thể được mở rộng nếu tăng số lượng node, trong đề tài sẽ tiến hành thử nghiệm khoảng cách giữa 2 node để đánh giá khoảng cơ bản có thể đạt được. Kết quả thu được cho thấy các Node hoạt động ổn định trong bán kính khoảng cách 50m trong điều kiện không vật cản.

Năng lượng tiêu thụ:

Trong quá trình thử nghiệm, các giá trị thu được:

Bảng 10: Bảng thống kê dòng điện tiêu thụ của Z-Wave Node.

Sate	Idle	Active	Sleep
Vcc	3.3v	3.3v	3.3v
I	1.8mA	15mA	0.4mA

6. Kết luận

6.1 Kết luận

Sau khi thực hiện đề tài này, sinh viện được rèn luyện nhiều hơn về kỹ năng deisgn cũng như đi layout cho mạch, kỹ năng hàn và sữa lỗi mạch, kỹ năng lập trình và debug các chương trình phần mềm, kỹ năng vẽ thiết kế cơ khí và gia công laser cũng như in 3D cơ bản. Tuy không đáng kể nhưng đây là một kinh nghiệm quý giá trong quá trình thực hiện xây dựng một hệ thống.

Ưu điểm:

- Tích hợp thành công các giao thức cơ bản đặt ra vào hệ thống, bao gồm LoRaWAN, Z-Wave, Modbus Serial. Qua đó góp phần tạo nên một nền tảng IoT có thể tuỳ biến mở rộng hay thu gọn dựa vào các giao thức mà nền tảng hỗ trợ.

- Việc triển khai thành công giao thức LoRaWAN góp phần mở rộng mạng lưới kết nối không dây và tăng thêm số lượng thiết bị có thể quản lý mà không cần quan tâm đến các nghi thức phức tạp bên dưới. Đây cũng là nền tảng để phát triển nhiều loại thiết bị trong một hệ thống IoT, đặc biệt là các thiết bị có yêu cầu về năng lượng và khoảng cách.
- Triển khai mạng lưới Z-Wave hướng tới ứng dụng nhà thông minh, các thiết bị được quy chuẩn nên tương thích với các thiết bị cung cấp từ nhiều hãng sản xuất bên ngoài.
- Triển khai thành công giao thức Modbus góp phần làm đơn giản hoá các quá trình quản lý dữ liệu trên mạng có dây nhiêu thiết bị cũng như tương tác được với các thiết bị hỗ trợ chuẩn Modbus bên ngoài.

Khuyết điểm:

Một vài mục tiêu đặt ra của đề tài vẫn chưa được hoàn thiện.

- LoRaWAN: luồng dữ liệu Downlink từ phía Server tạm thời không có tín hiệu xác nhận gửi thành công. Trong quá trình tìm hiểu, đây là lỗi của nhà sản xuất và sẽ được vá lỗi sau. Quá đó thấy được sự cần thiết khi làm chủ lớp giao thức này ở lớp thiết bị để không bị phụ thuộc và tăng tính tuỳ biến.
- LoRaWAN: Việc thiết kết mạch điện, antenna thu phát chưa tốt nên khoảng cách thu được chưa đạt như mong muốn. Ngoài ra một số ràng buộc về băng tần, tốc độ, thời gian tín hiệu truyền, công suất phát quy hoạch tại từng vị trí địa lý cũng ảnh hưởng phần nào kết quả thu được so với công bố nhà sản xuất.
- LoRaWAN: triển khai số lượng thiết bị trong mạng còn ít so với khả năng quản lý của mạng đặt ra.
- LoRaWAN: Server hiện tại vẫn dùng mã nguồn có sẵn từ ChirpStack, trong quá trình thử nghiệm đến thời điểm hiện tại vẫn chưa có lỗi không mong muốn xảy ra. Tuy nhiên, việc tích hợp nó vào hệ thống IoT Platform còn nhiều mặt hạn chế.
- Modbus: chỉ triển khai Modbus RTU và các chức năng đọc, ghi cơ bản nhất. Tuy có thể đáp ứng được các yêu cầu trong mạng thực tế nhưng vẫn chưa đủ hỗ trợ toàn bộ các chức năng của giao thức này.
- Việc xử lý lớp giao thức MQTT trên nền tảng vi điều khiển vẫn còn chậm do đó ảnh hưởng đến tốc độ của các Gateway Modbus.

- Z-Wave: khoảng cách đo đạc khoảng cách thực tế còn thấp so với nhà sản xuất, qua đó thấy được tính toán sai trong việc thiết kế antenna.

6.2 Hướng phát triển

Với phần cứng có sẵn và phần mềm sử dụng nền tảng có tính mở rộng cao, kết hợp với việc áp dụng nhiều giao thức tích hợp, mô hình có thể phát triển:

- Kết hợp thêm và chuẩn hoá các giao thức phổ biến để đa dạng hoá các lựa chọn trong một hệ thống IoT: Bluetooth, Wifi, Zigbee, …
- Xây dựng thêm các thiết bị đầu cuối trên từng giao thức, qua đó làm đa dạng hoá các giải pháp trong hệ thống IoT.

7. Tài liệu trích dẫn

[1] "Internet Vạn Vật," 27 6 2020 . [Online]. Available: https://vi.wikipedia.org/wiki/Internet_V%E1%BA%A1n_V%E1%BA%ADt.

[2] "SEMTECH," LoRa Gateways, [Online]. Available: https://www.semtech.com/lora/what-is-lora.

[3] L. Alliance, "lora-alliance.org," [Online]. Available: https://lora-alliance.org/about-lorawan.

[4] "Modbus," Modicon, 16 July 2020. [Online]. Available: https://en.wikipedia.org/wiki/Modbus.

[5] "Z-Wave, Safer, Smarter," Silicon Laboratories, 2020. [Online]. Available: https://www.z-wave.com/.

[6] "FreeRTOS™ Real-time operating system for microcontrollers," Real Time Engineers Ltd, 2020. [Online]. Available: https://www.freertos.org/.

[7] J. Jin, "BCD Semiconductor Manufacturing Limited," July 2012. [Online]. Available: https://www.diodes.com/assets/App-Note-Files/AN1063.pdf.

[8] DIODE, "DIODES 1N5819," [Online]. Available: https://www.diodes.com/assets/Datasheets/ds23001.pdf.

[9] "Diodes," https://www.diodes.com/assets/Datasheets/74HC14.pdf, January 2013. [Online]. Available: https://www.diodes.com/assets/Datasheets/74HC14.pdf.

[10] O. Brocaar, "ChirpStack open-source LoRaWAN® Network Server," 2020. [Online]. Available: https://www.chirpstack.io/.

[11] "The Things Network," The Things Industries, 2020. [Online]. Available: https://www.thethingsnetwork.org/.

8. PHỤ LỤC

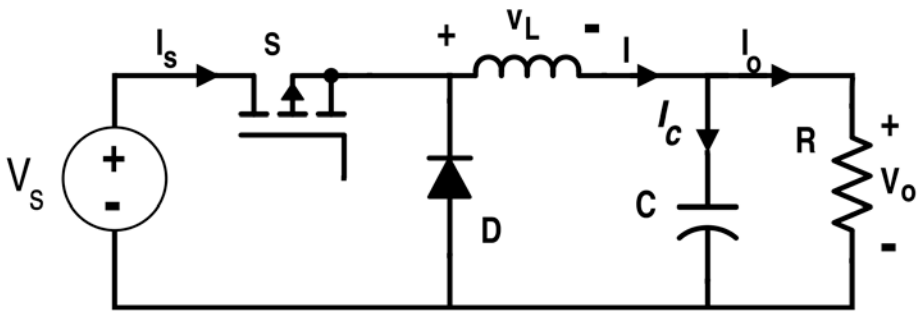
8.1 Bộ nguồn giảm áp – Buck Converter

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.

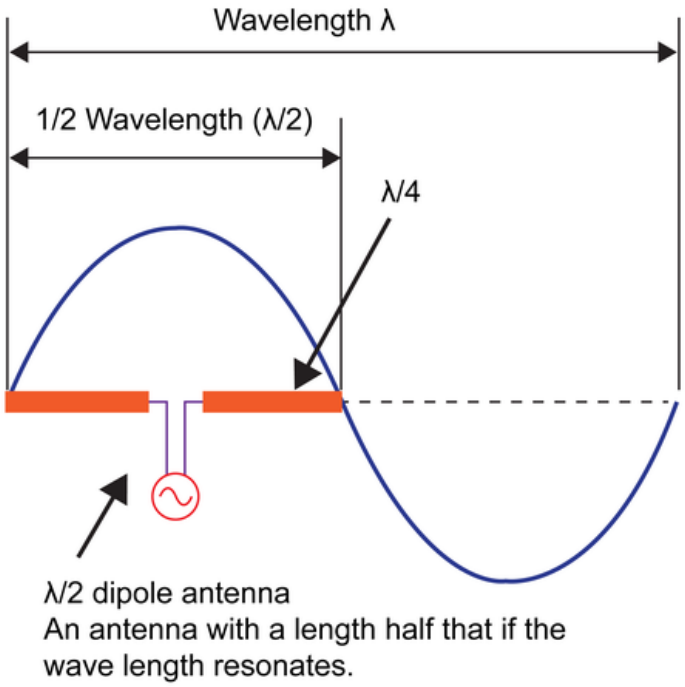


Hình 1. Sơ đồ mạch Buck Converter đơn giản

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
- Bộ điều khiển độ rộng xung dành cho linh kiện chuyển mạch.

8.2 Tham khảo thiết kế Antenna theo độ dài bước sóng.



Hình 68: Tham khảo thiết kế antenna theo độ dài bước sóng.

8.3 Mô hình demo hệ thống

Hình 69: Mô hình demo hệ thống.