

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
TRƯỜNG ĐIỆN – ĐIỆN TỬ



## ĐỒ ÁN NGHIÊN CỨU CỦ NHÂN

**THIẾT KẾ HỆ THỐNG ĐỊNH VỊ TRONG NHÀ ỨNG DỤNG  
CÔNG NGHỆ ULTRA WIDEBAND**

**Ngành Kỹ thuật điều khiển và tự động hóa**

**Chuyên ngành Kỹ thuật đo và tin học công nghiệp**

**Sinh viên:** Đỗ Anh Dũng

**Lớp:** ĐK&TĐH 10 K63

**Giảng viên hướng dẫn:** PGS.TS. Hoàng Sỹ Hồng

Chữ ký của GVHD

**Khoa** Tự động hóa

**Hà Nội, 8-2022**

NHIỆM VỤ  
ĐỒ ÁN TỐT NGHIỆP

Họ và tên sinh viên: .....

Khóa..... Trường: Điện- Điện tử      Ngành: KT ĐK & TĐH

1. *Tên đề tài:*

.....

2. *Nội dung đề tài:*

.....

.....

.....

.....

.....

.....

.....

.....

.....

3. *Thời gian giao đề tài:*.....

4. *Thời gian hoàn thành:*.....

Ngày..... tháng ..... năm 2022

CÁN BỘ HƯỚNG DẪN

## **Lời cảm ơn**

Đầu tiên em xin gửi lời cảm ơn đến PGS. TS Hoàng Sĩ Hồng, người đã trực tiếp hướng dẫn cho em thực hiện đề tài này. Thầy đã giúp em định hướng suy nghĩ và hướng đi trong suốt thời gian em thực hiện đề tài. Em xin chân thành cảm ơn các thầy, cô giáo đã trực tiếp giảng dạy, chia sẻ những kiến thức cả về học thuật lẫn đời sống cho em trong những năm tháng tại Bách Khoa Hà Nội. Lời cảm ơn cuối cùng em gửi tới các anh tại Trung tâm vi điện tử và tin học, Viện Ứng Dụng Công Nghệ Nacentech và tập thể Lab MANDevices.

## **Tóm tắt nội dung đồ án**

Số hóa vạn vật đang là một trong những xu thế của cuộc cách mạng công nghệ 4.0, trong đó các hệ thống quản lý hàng hóa đang được các công ty, nhà máy, xí nghiệp rất quan tâm và đầu tư, phát triển số hóa hệ thống. Một trong những dữ liệu đầu vào cực kỳ quan trọng cho hệ thống quản lý đó là dữ liệu vị trí. Đã có rất nhiều công nghệ được đưa ra để đáp ứng nhu cầu của từng bài toán cụ thể, ứng với đó là các hệ thống từ hiện trường, gateway, server hay cloud. Qua thời gian nghiên cứu, thu thập thông tin, em đã chọn đề tài “Thiết kế hệ thống định vị trong nhà ứng dụng công nghệ Ultra Wideband” để có cơ hội tiếp cận đến các bài toán đó.

Kết quả đồ án cơ bản hoàn thiện được mục tiêu đề ra. Đề tài hướng tới ứng dụng thực tế ngay tại hiện trường, yêu cầu cao về khả năng làm việc ổn định và lâu dài. Thiết bị được thiết kế hướng tới tiềm cản sản phẩm thương mại.

Các kiến thức và kỹ năng đạt được:

- Kiến thức về Bluetooth Low Energy
- Giao tiếp giữa ESP32 và các module trên thiết bị
- Kiến thức về MQTT, TLS, giao tiếp với cloud qua MQTT
- Kiến thức về WiFi, Ethernet, Smartconfig cho thiết bị WiFi

Sinh viên thực hiện

Ký và ghi rõ họ tên

# MỤC LỤC

<b>MỤC LỤC .....</b>	i
<b>DANH MỤC HÌNH VẼ.....</b>	iii
<b>DANH MỤC BẢNG BIỂU.....</b>	i
<b>DANH MỤC TỪ VIẾT TẮT .....</b>	ii
<b>CHƯƠNG 1. TÌM HIỂU CHUNG VỀ HỆ THỐNG ĐỊNH VỊ TRONG NHÀ .....</b>	1
1.1 <i>Bài toán định vị trong nhà.....</i>	1
1.2 <i>Công nghệ Ultra Wide-Band .....</i>	5
1.2.1 Nguyên lý các phương pháp đo .....	5
1.2.2 Tổng quan .....	7
1.3 <i>Google Cloud IoT Core.....</i>	10
1.3.1 Tổng quan .....	10
1.3.2 Cấu trúc .....	12
1.4 <i>MQTT .....</i>	14
1.5 <i>TLS .....</i>	16
1.6 <i>Kết luận chương.....</i>	18
<b>CHƯƠNG 2. THIẾT KẾ HỆ THỐNG .....</b>	20
2.1 <i>Mục tiêu.....</i>	20
2.2 <i>Thiết kế tổng thể.....</i>	20
2.3 <i>Module Ultra wideband .....</i>	21
2.3.1 Microchip ATA835x.....	22
2.3.2 NXP SR150.....	22
2.3.3 Decawave DW1000 .....	23
2.4 <i>Mô hình mạng hệ thống dựa trên module DWM1001.....</i>	24

2.4.1 Mô tả hoạt động của mạng UWB .....	26
2.4.2 Các mô hình mạng cho DWM1001 .....	27
2.4.3 Công nghệ Bluetooth Low Energy.....	29
<b>2.5 Thiết kế Gateway.....</b>	<b>35</b>
2.5.1 Yêu cầu hệ thống.....	35
2.5.2 Tổng quan hệ thống.....	37
2.5.3 Thiết kế khói nguồn .....	37
2.5.4 Thiết kế khói xử lý trung tâm.....	41
2.5.5 Khối lưu trữ dữ liệu.....	44
2.5.6 Thiết kế khói Ethernet.....	46
2.5.7 Thiết kế khói hiển thị, thông báo .....	49
<b>2.6 Thiết kế Firmware.....</b>	<b>51</b>
2.6.1 Quản lý Node .....	53
2.6.2 MQTT .....	58
2.6.3 Smartconfig.....	61
2.6.4 Ethernet .....	63
<b>CHƯƠNG 3. KẾT QUẢ THỬ NGHIỆM VÀ ĐÁNH GIÁ.....</b>	<b>64</b>
<b>3.1 Kết quả thiết kế, chế tạo.....</b>	<b>64</b>
<b>3.2 Kết quả thử nghiệm .....</b>	<b>65</b>
3.2.1 Kiểm tra hoạt động của thiết bị .....	65
3.2.2 Thử nghiệm tính năng của hệ thống.....	69
<b>CHƯƠNG 4. KẾT LUẬN .....</b>	<b>74</b>
<b>4.1 Kết luận chung.....</b>	<b>74</b>
<b>4.2 Hướng phát triển .....</b>	<b>75</b>
<b>TÀI LIỆU THAM KHẢO.....</b>	Error! Bookmark not defined.

## **DANH MỤC HÌNH VẼ**

Hình 1.1 Biểu đồ dựa trên độ chính xác và độ bao phủ .....	3
Hình 1.2 Tương quan giữa độ chính xác và bước sóng mang .....	3
Hình 1.3 So sánh tổng quan các công nghệ định vị trong nhà .....	4
Hình 1.4 Phương pháp đo ToA/ToF .....	6
Hình 1.5 TDoA trong định vị GNSS .....	6
Hình 1.6 Phương pháp Two-way Ranging .....	7
Hình 1.7 Mô hình mẫu giữa Cloud IoT Core với các dịch vụ khác của GCP....	11
Hình 1.8 Google Cloud IoT Core .....	12
Hình 1.9 Giao thức MQTT .....	14
Hình 1.10 Cách hoạt động của chứng chỉ TLS .....	16
Hình 1.11 Mã hóa đối xứng .....	17
Hình 1.12 Quá trình trao đổi session key trong TLS.....	18
Hình 2.1 Kit thử nghiệm ATA835 .....	22
Hình 2.2 Kit phát triển dựa trên SR150.....	23
Hình 2.3 Module DWM1001 .....	24
Hình 2.4 Kit phát triển DWM1001-DEV .....	24
Hình 2.5 Cấu trúc của siêu khung truyền .....	27
Hình 2.6 Mô hình sử dụng bridge node.....	28
Hình 2.7 Mô hình sử dụng gateway BLE.....	29
Hình 2.8 BLE Stack.....	30
Hình 2.9 Cấu trúc bản tin Advertising.....	31
Hình 2.10 Quá trình scanning của thiết bị BLE .....	32
Hình 2.11 Profiles, Services và Characteristics.....	34
Hình 2.12 IC TEA1721AT .....	39
Hình 2.13 Khối chuyển đổi điện áp AC sang DC .....	40
Hình 2.14 Khối chuyển đổi điện áp cao sang 5V .....	40
Hình 2.15 Khối hạ áp 3.3V.....	41

Hình 2.16 ESP32 Wroom-32.....	42
Hình 2.17 ESP32 Wroom-32 No Shield.....	42
Hình 2.18 Schematic khói xử lý trung tâm .....	43
Hình 2.19 Khối lưu trữ dữ liệu .....	44
Hình 2.20 IC Flash W25Q128JV .....	45
Hình 2.21 Cấu trúc bộ nhớ Flash theo Block .....	45
Hình 2.22 Cấu trúc một block .....	45
Hình 2.23 Ghi dữ liệu vào một page trong flash .....	46
Hình 2.24 IC W5500 Ethernet.....	47
Hình 2.25 Schematic khói Ethernet.....	47
Hình 2.26 Schematic IC W5500.....	48
Hình 2.27 Các chế độ của IC W5500 .....	49
Hình 2.28 Schematic cổng RJ45 HR911105A .....	49
Hình 2.29 Thời gian định nghĩa cho bit 0, 1 .....	50
Hình 2.30 Hình dạng xung định nghĩa cho bit 0, 1 .....	50
Hình 2.31 Schematic khói LED RGB .....	50
Hình 2.32 Schematic khói thông báo .....	51
Hình 2.33 Các tính năng chính của Gateway .....	52
Hình 2.34 Lưu đồ thuật toán trạng thái Initialize .....	55
Hình 2.35 Lưu đồ thuật toán trạng thái Running .....	56
Hình 2.36 Bộ lập lịch kết nối .....	57
Hình 2.37 Ví dụ quá trình quản lý kết nối .....	58
Hình 2.38 Lưu đồ thuật toán quá trình kết nối MQTT.....	60
Hình 2.39 Thêm public key khi khởi tạo thiết bị .....	61
Hình 2.40 Lưu đồ thuật toán quá trình cấu hình thiết bị .....	63
Hình 3.1 Mặt trên mạch PCB .....	64
Hình 3.2 Mặt dưới mạch PCB .....	64
Hình 3.3 Kết quả đóng hộp của thiết bị.....	65
Hình 3.4 Kiểm tra hoạt động khói nguồn .....	66
Hình 3.5 Kiểm tra hoạt động monitor, nạp, xóa firmware qua cổng microUSB	66

Hình 3.6 Giao diện APP Smartconfig.....	67
Hình 3.7 Thiết bị smartconfig và kết nối WiFi.....	68
Hình 3.8 Thiết bị nhận gói tin cấu hình project.....	68
Hình 3.9 Thiết bị nhận gói tin key và rootCA.....	68
Hình 3.10 Thử nghiệm độ chính xác của hệ thống DWM1001 .....	69
Hình 3.11 Không gian thử nghiệm .....	70
Hình 3.12 Thiết bị Gateway và 6 Tag.....	71
Hình 3.13 Thử nghiệm hệ thống tại hiện trường .....	73
Hình 3.14 Kết quả thử nghiệm hệ thống tại hiện trường.....	73

## **DANH MỤC BẢNG BIỂU**

Bảng 1.1 Các công nghệ định vị trong nhà .....	2
Bảng 2.1 Các phần tử linh kiện chính .....	37
Bảng 2.2 Mức tiêu thụ tối đa của hệ thống .....	38
Bảng 2.3 Năng lượng tiêu thụ khói xử lý trung tâm .....	39
Bảng 2.4 Thông số Module ESP32 Wroom-32 .....	42
Bảng 2.5 Giải thích thành phần khói xử lý trung tâm .....	43
Bảng 2.6 Thông số IC W5500 .....	47
Bảng 2.7 Các phân vùng bộ nhớ của thiết bị.....	52
Bảng 2.8 Các Characteristic sử dụng trong giao tiếp BLE với DWM1001 .....	53
Bảng 2.9 MQTT Topic của thiết bị.....	58
Bảng 2.10 Các payload thiết bị sử dụng.....	59
Bảng 2.11 Các URI của HTTP Server.....	62
Bảng 3.1 Kết quả thử nghiệm tính năng hệ thống DWM1001.....	69

## **DANH MỤC TỪ VIẾT TẮT**

Từ viết tắt	Tiếng anh	Tiếng việt
ATT	Attribute Protocol	
BLE	Bluetooth Low Energy	
ETSI	European Telecommunications Standards Institute	Viện tiêu chuẩn viễn thông Châu Âu
FCC	Federal Communications Commission	Ủy ban truyền thông liên bang
GAP	Generic Access Profile	
GATT	Generic Attribute Profile	
GCIC	Google Cloud IoT Core	
GCP	Google Cloud IoT Platform	
GNSS	Global Navigation Satellite System	Hệ thống định vị toàn cầu
HTTP	Hypertext Transfer Protocol	
IoT	Internet of Things	Kết nối vạn vật
JWT	JSON Web Token	
LoS	Light of Sight	
MCU	Microcontroller Unit	
MQTT	Message Queuing Telemetry Transport	
NLoS	None Light of Sight	
PoA	Phase of Arrival	

QoS	Quality of Service	
RFID	Radio Frequency Identification	
RTLS	Realtime Locating System	
RTT	Round Trip Time	
SoC	System on Chip	
TDMA	Time Division Multiple Access	
TDoA	Time Difference of Arrival	
TLS	Transport Layer Security	
ToA	Time of Arrival	
ToF	Time of Flight	
TWR	Two Way Ranging	
UUID	Universal Unique Identifier	
UWB	Ultra Wideband	



# **CHƯƠNG 1. TÌM HIỂU CHUNG VỀ HỆ THỐNG ĐỊNH VỊ TRONG NHÀ**

## **1.1 Bài toán định vị trong nhà**

Kể từ những ngày đầu tiên các công nghệ định vị bằng vệ tinh xuất hiện vào năm 60 của thế kỉ trước, giải pháp cho việc định vị đã liên tục có những biến chuyển lớn để đáp ứng nhu cầu của con người và xã hội. Việc biết được vị trí của con người, của vật luôn luôn là mối quan tâm hàng đầu. Trong lĩnh vực vận chuyển, vị trí của xe hàng, tàu thủy, máy bay ... , nếu có thể biết được, sẽ làm tăng hiệu quả vận chuyển, là yếu tố quan trọng cho bài toán logistic đang rất “hot” trong những năm gần đây. Trong lĩnh vực bán hàng ở các trung tâm, siêu thị, ... , việc biết được số lượng khách hàng lui tới các khu vực quầy bán hàng nào, thời gian khách hàng lưu lại quầy đó là bao lâu, sẽ là dữ liệu quan trọng liên quan tới các chiến dịch bán hàng. Trong các nhà xưởng, kho bãi, ... có thể biết được loại hàng hóa, mã sản phẩm nào đang nằm ở đâu hoặc nhân viên đang ở vị trí nào trong xưởng, sẽ làm tăng hiệu quả sản xuất, đưa quy trình sản xuất, xuất nhập kho bãi số hóa đúng như mong muốn của các công ty lớn hiện nay. Đó chỉ là một trong những bài toán và giải pháp trong một “bể” những nhu cầu của thị trường và xã hội trong thời kì số hóa đang chuyển mình cực mạnh.

Gốc rễ của những giải pháp đó đều quy về một điểm đó là biết được vị trí của vật thể. Để có thể biết được yếu tố nghe tưởng chừng đơn giản đó, các nhà khoa học đã đưa ra rất nhiều công nghệ khác nhau, với độ hiệu quả khác nhau và phù hợp cho các bài toán rất khác nhau. Công nghệ định vị bằng vệ tinh GNSS với các chùm vệ tinh nổi tiếng như GPS của Mỹ, GLONASS của Nga, ... đã có thể đạt đến độ chính xác 1cm với thuật toán RTK. Với độ chính xác cao như vậy, nó hoàn toàn có thể đáp ứng cho các bài toán liên quan đến điều khiển từ xa. Tuy nhiên GNSS chỉ thực sự mạnh khi ở không gian bên ngoài trời, còn khi ở trong nhà, GNSS có độ chính xác rất thấp, thậm chí là tín hiệu không đủ mạnh để có các thông tin về vị trí. Từ đó, các công nghệ không dây có thể hoạt động trong nhà và đã có sẵn như WiFi, Bluetooth, Bluetooth Low Energy được xét đến. Bảng dưới đây là so sánh giữa các công nghệ

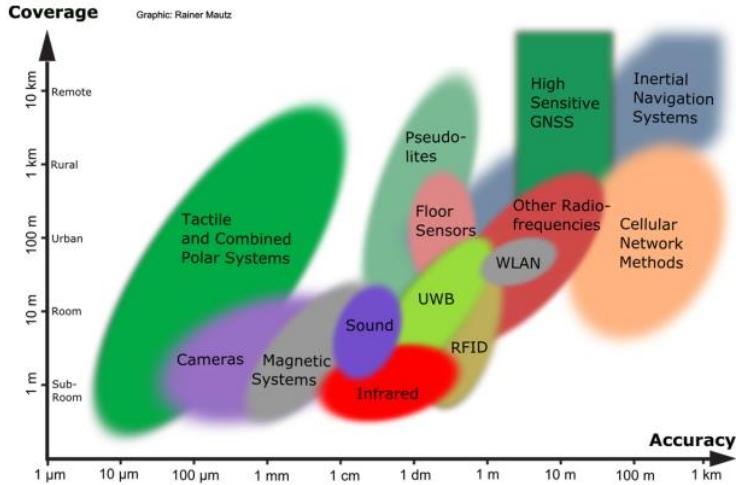
định vị hiện có cùng với độ chính xác, độ bao phủ và các ứng dụng phổ biến phù hợp cho ứng dụng đó.

**Bảng 1.1 Các công nghệ định vị trong nhà**

Công nghệ	Độ chính xác	Độ bao phủ (m)	Nguyên lý	Ứng dụng
Camera	0.1mm – dm	1 – 10	Các phép đo góc từ tập ảnh	Đo lường, dẫn đường robot
Infrared	cm – dm	1 – 5	Ảnh nhiệt	Xác định con người, theo dấu
Sound	cm	2 – 10	Khoảng cách từ thời gian truyền của tín hiệu	Định vị trong nhà, theo dấu
WLAN / Wi-Fi	m	20 – 50	RSSI	Định vị trong nhà
Ultra-Wideband	cm – m	1 – 50	Khoảng cách từ thời gian truyền của tín hiệu	Định vị trong nhà, hàm mỏ
Magnetic Systems	mm – cm	1 – 20	Tù trường	Định vị trong nhà, hàm mỏ

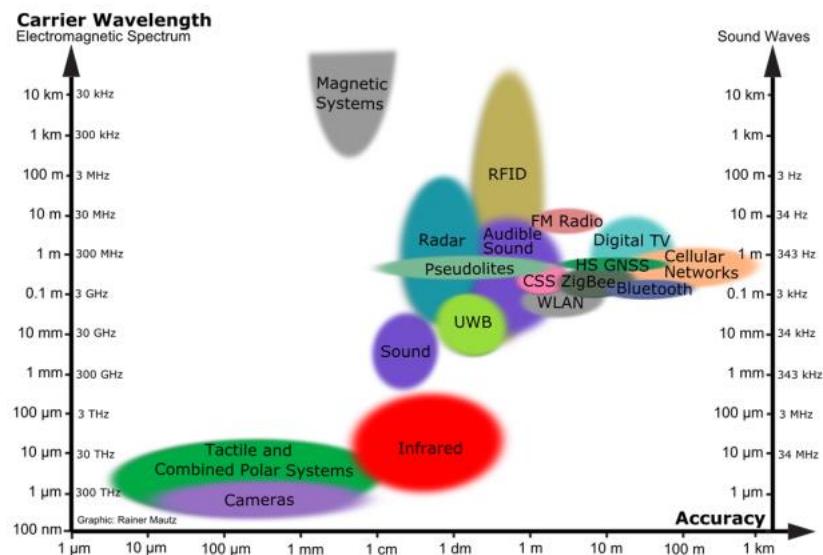
Mỗi công nghệ đều có điểm mạnh riêng, phù hợp với yêu cầu của từng ứng dụng cụ thể. Nhìn chung, các công nghệ không dây sử dụng sóng điện từ vẫn có lợi thế hơn về độ chính xác và giá thành. Công nghệ sử dụng camera với các thuật toán xử lý ảnh có độ chính xác cao, tuy nhiên lại có nhược điểm ở những nơi thiếu sáng, hoặc các tác động ngoại cảnh ảnh hưởng đến chất lượng ghi hình của camera, sẽ rất dễ gây ra sai số hệ thống. Giữa những công nghệ không dây, công nghệ sử dụng Ultra – Wideband đang nhận được sự quan tâm rất lớn và được đánh giá sẽ sớm trở thành xu thế trong tương lai gần nhờ vào độ chính xác cao, độ bao phủ rộng và giá thành hợp lí.

Biểu đồ dưới đây sẽ cho cái nhìn tổng quát hơn, so sánh giữa các công nghệ dựa trên hai yếu tố là độ chính xác và độ bao phủ



**Hình 1.1 Biểu đồ dựa trên độ chính xác và độ bao phủ**

Hầu hết các công nghệ đều dựa trên sóng điện từ và sóng âm. Có thể thấy, một phần lớn phổ điện từ được sử dụng cho bài toán định vị trong nhà. Các hệ thống với độ chính xác cao có xu hướng sử dụng các bước sóng ngắn hơn. Biểu đồ ở hình 1.2 sẽ cho thấy tương quan giữa độ chính xác và bước sóng mang.



**Hình 1.2 Tương quan giữa độ chính xác và bước sóng mang**

Nhìn chung, với không gian kích thước 50m x 50m, UWB đang chiếm lợi thế lớn so với các công nghệ khác. Và thực tế cũng đang chứng minh điều đó khi đang có rất nhiều các giải pháp sử dụng UWB để thu thập loại dữ liệu tưởng chừng đơn giản nhưng đóng vai trò rất quan trọng trong hệ thống là dữ liệu vị trí.

Technology	Accuracy	Range	Suitable for	Tracking	Transmitter power supply	Battery lifetime
Wi-Fi	< 15 m	< 150 m			or	
BLE 5.1	< 8 m	< 75 m				
	< 1 m with line-of-sight					
UWB	< 30 cm	< 150 m			or	
RFID	presence detection only	< 1 m			—	(passive RFID tag)

**Hình 1.3 So sánh tổng quan các công nghệ định vị trong nhà**

Hình 1.3 là tổng quan so sánh giữa 4 công nghệ được sử dụng nhiều nhất trong hệ thống định vị trong nhà, cùng với đó là các ứng dụng phù hợp cho từng công nghệ đó.

Hệ thống định vị trong nhà ngoài phần quan trọng nhất đo là xác định vị trí của người, vật, thì phần quan trọng không kém đó là đưa được dữ liệu đó ra bên ngoài, tức là có thể theo dõi được dữ liệu đó từ xa, có thể tại phòng điều khiển của nhà máy, hoặc là tại bất kỳ nơi đâu, chỉ cần mở điện thoại hoặc máy tính là có thể theo dõi được dữ liệu vị trí mà người dùng cần biết, hoặc cấu hình hệ thống từ xa mà không cần phải nạp lại firmware, software trực tiếp cho thiết bị. Với hệ thống sử dụng WiFi, việc đưa dữ liệu ra bên ngoài mạng và nhận dữ liệu cấu hình từ bên ngoài đã quá đơn giản, không cần phải thêm một thiết bị mới để có thể giao tiếp được với bên ngoài. Các hệ thống sử dụng các công nghệ như BLE, UWB, RFID đều cần một thiết bị chuyển tiếp, hay còn gọi là Gateway để có thể giao tiếp với bên ngoài.

Ở phía dưới hiện tường, Gateway sẽ là thiết bị quản lý dữ liệu về vị trí, hay cấu hình các thiết bị khi có dữ liệu gửi xuống từ server. Giao thức được Gateway ưu tiên sử dụng trong hệ thống để truyền nhận dữ liệu với server là MQTT hoặc HTTP. Dữ liệu vị trí là dữ liệu khá nhạy cảm, khi đó tầng SSL/TLS sẽ được thêm vào cho cả hai giao thức MQTT và HTTP.

Với tầng mạng phía trên của mô hình, như đã nói, server sẽ thường được ưu tiên sử dụng, dễ triển khai với các hệ thống nhỏ và vừa, hoặc đối với các nhà máy chỉ muốn lưu trữ dữ liệu về vị trí tại phòng điều khiển của nhà máy. Ngoài lựa chọn server để quản lý, lưu trữ thông tin, còn một mô hình khác đang rất được quan tâm và được nhiều hệ thống lớn sử dụng đó là lưu trữ tại đám mây (cloud). Ưu điểm rõ rệt nhất của

Cloud so với server vật lý hay server ảo truyền thông đó là độ ổn định cao và khả năng mở rộng hệ thống rất nhanh. Vì thế, cloud là lựa chọn lý tưởng cho các hệ thống định vị trong nhà. Trên thị trường đã có nhiều hệ thống cloud lớn và hiệu quả cao như của Amazone, Google,... Thêm vào đó, hệ thống cloud cũng cần được thiết kế để phù hợp với các ứng dụng IoT khi hỗ trợ các giao thức MQTT, HTTP với lớp bảo mật TLS, Broker cho các thiết bị sử dụng MQTT và các hệ thống topic, quản lý chật chẽ bởi các hệ thống IoT là các hệ thống có khả năng “scale” rất nhanh và mạnh, lượng dữ liệu có thể rất lớn, nếu không có quy hoạch và quản lý rõ ràng sẽ không thể đảm bảo được tính ổn định của hệ thống. Từ đó, thị trường đã xuất hiện các mô hình để có thể đáp ứng nhu cầu cho các ứng dụng IoT và nổi bật nhất là Google Cloud IoT Core, một dịch vụ con trong Google Cloud IoT Platform của Google.

## 1.2 Công nghệ Ultra Wide-Band

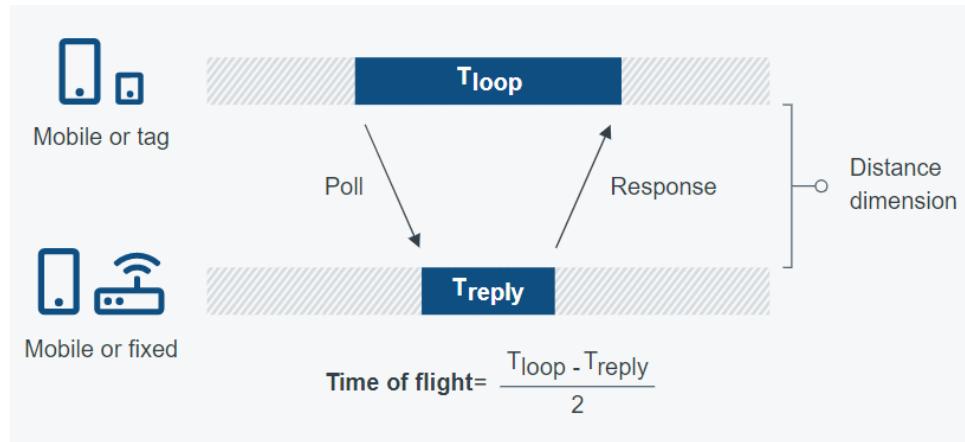
### 1.2.1 Nguyên lý các phương pháp đo

Các phương pháp đo dưới đây là các kỹ thuật phổ biến dành cho các bài toán về đo khoảng cách, đo góc

#### 1.2.1.1 Time of Arrival (ToA) / Time of Flight (ToF)

Nguyên lý của ToA dựa trên việc đo thời gian mà tín hiệu đi từ bộ truyền tới bộ nhận. Khoảng cách Euclide giữa 2 thiết bị bằng tích của thời gian tín hiệu truyền và vận tốc sóng. Bởi vì vận tốc sóng dựa trên các tính chất của môi trường lan truyền, nên cần phải chú ý đến đặc tính của vật liệu tín hiệu đi qua. Đối với các vật liệu xây dựng, tốc độ truyền phụ thuộc vào  $\sqrt{k}$  (trong đó k là hằng số điện môi).

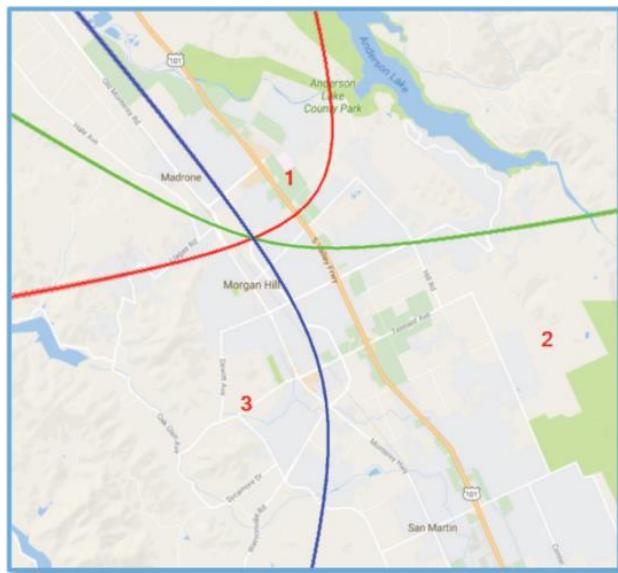
Vì ToA phụ thuộc vào thời gian truyền của tín hiệu, nên đồng bộ thời gian giữa bộ truyền và nhận cần rất chính xác, chỉ cần sai lệch 1 nano giây, khoảng cách có thể sai số thêm 30 cm nếu sử dụng các sóng radio.



**Hình 1.4 Phương pháp đo ToA/ToF**

#### 1.2.1.2 Time Difference of Arrival (TDoA)

TDoA là phương pháp định vị cho các nguồn sử dụng sóng RF. Phương pháp này yêu cầu cần có ít nhất ba bộ nhận tín hiệu. Mỗi bộ nhận cần được đồng bộ về thời gian một cách chính xác. Khi tín hiệu gửi từ một nguồn chưa rõ vị trí đến cả ba bộ nhận, khoảng cách sẽ được tính dựa trên sự khác nhau về thời gian nhận được tín hiệu của ba bộ và vị trí cố định của ba bộ đã được đặt từ trước. Với mỗi khoảng cách từ điểm phát đến điểm thu, sẽ tạo thành một hình hyperbol. Giao điểm của hình hyperbol đó chính là vị trí của điểm phát tín hiệu.



**Hình 1.5 TDoA trong định vị GNSS**

Để có thể sử dụng TDoA, cần hiểu rõ loại tín hiệu được sử dụng, sự phụ thuộc của kết quả đo dựa vào vị trí của phép đo thé nào, các nguồn gây nhiễu và cách để xử lý chúng. Phương pháp TDoA sẽ có thể tối ưu vấn đề về năng lượng so với các

phương pháp khác bởi quá trình tính toán sẽ nằm ở các thiết bị cố định đã biết trước vị trí, sử dụng nguồn điện lưới, khi đó các thiết bị mà cần biết vị trí sẽ không cần hoạt động quá nhiều, phù hợp cho các ứng dụng sử dụng pin.

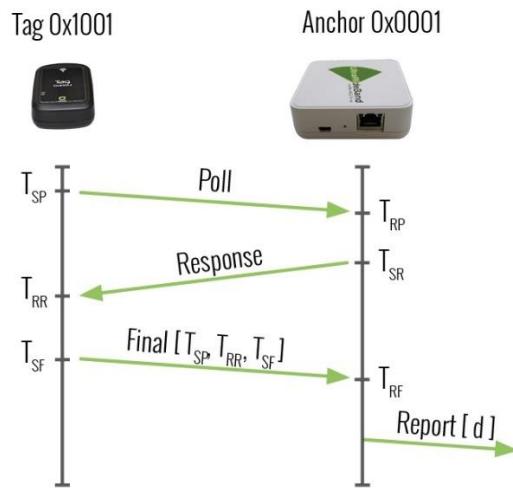
#### 1.2.1.3 Round Trip Time (RTT) / Roundtrip Time of Flight (RToF) / Two Way Ranging (TWR)

Phương pháp RTT, hay còn gọi là TWR dựa trên nguyên lý của phương pháp ToF. Để đo được khoảng cách, giữa hai thiết bị sẽ cần trao đổi ba loại bản tin. Theo hình 1.6, giả sử Tag là thiết bị cần biết vị trí và Anchor là thiết bị đã được đặt vị trí từ trước. Tag sẽ gửi bản tin Poll ở thời điểm  $T_{SP}$  đến cho Anchor và sẽ được Anchor đáp lại bằng một bản tin ở thời điểm  $T_{SR}$ . Khi Tag nhận được bản tin trả lời từ Anchor ở  $T_{RR}$ , bản tin cuối cùng sẽ được gửi từ Tag sang Anchor ở thời điểm  $T_{SF}$  và nhận được ở Anchor ở  $T_{RF}$ .

Cuối cùng, vị trí được tính bởi công thức:

$$\text{distance} = \text{ToF} \cdot v$$

$$\text{ToF} = [(T_{RR} - T_{SP}) - (T_{SR} - T_{RP}) + (T_{RF} - T_{SR}) - (T_{SF} - T_{RP})] / 4$$



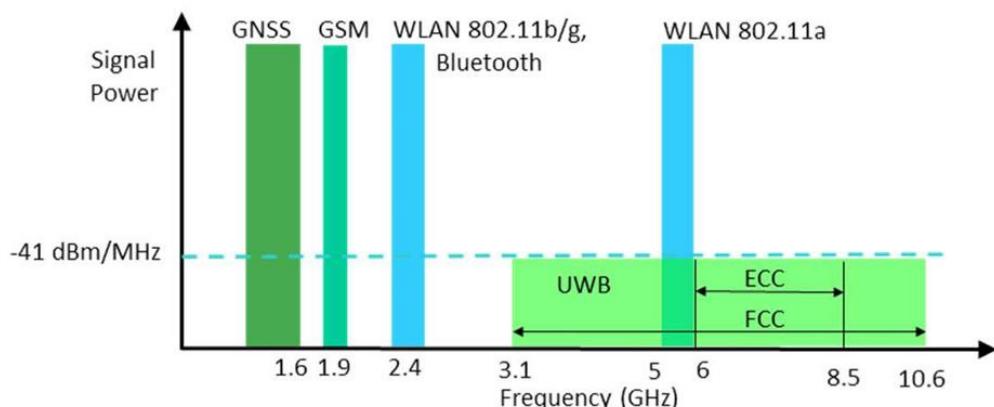
**Hình 1.6 Phương pháp Two-way Ranging**

#### 1.2.2 Tổng quan

Công nghệ UWB là công nghệ đang nhận được rất nhiều sự chú ý và là một chủ đề “hot” trong những năm gần đây. UWB mang đến quá nhiều tiện ích với những ứng

dụng trong việc kết nối không dây tốc độ cao của các thiết bị số trong nhà và văn phòng.

UWB là một công nghệ sử dụng sóng radio cho các ứng dụng truyền thông khoảng cách ngắn, băng thông rộng với khả năng chống nhiễu đa đường, phù hợp cho các ứng dụng ước lượng khoảng cách trong nhà, định vị và theo dấu. Một hệ UWB cụ thể có các bộ tạo sóng radio và các bộ nhận để thu các sóng lan truyền và phân tán. Ngược lại với hoạt động của các hệ băng thông hẹp, các sóng UWB chiếm băng thông khá lớn ( $> 500\text{MHz}$ ). Chính xác hơn, các sóng radio phát ra sẽ bị ảnh hưởng bởi UWB nếu băng thông của nó không vượt qua  $500\text{MHz}$  hoặc 20 % của tần số sóng mang. Để có thể giảm đi sự ảnh hưởng tới các dịch vụ radio khác, FCC đã hạn chế vùng sử dụng miễn phí (unlicensed) của UWB ở mật độ công suất bức xạ là  $-41.3\text{ dBm/MHz}$  và giới hạn băng thông tần số xuống  $3.1\text{ GHz} - 10.6\text{ GHz}$  ( $6.0\text{ GHz} - 8.5\text{ GHz}$  theo ECC). Hình 1.1 thể hiện phổ sóng UWB so với các chuẩn truyền thông sử dụng sóng radio hiện tại.



Các hạn chế về công suất phát của các ủy ban truyền thông đã làm giảm tầm hoạt động của xuống còn dưới 100m. Mặt khác, mật độ phổ năng lượng thấp cũng sẽ làm giảm đi các yếu tố gây hại tới cơ thể con người và hạn chế lại sự ảnh hưởng của UWB tới các bộ nhận băng thông hẹp khác. Với phần UWB được cấp phép sẽ hoạt động trong dải sóng của sóng điện từ, ở đó các phần tử tần số thấp trong phổ tín hiệu của UWB có khả năng xâm nhập các vật liệu xây dựng như bê tông, kính và gỗ. Đây là một đặc tính phù hợp cho bài toán định vị trong nhà, bởi nó cho phép tầm hoạt động dưới điều kiện không tầm nhìn (nLoS) và giúp cho việc định vị giữa các phòng trở nên dễ dàng hơn.

### 1.2.2.1 Đo khoảng cách sử dụng UWB

Ứng dụng tốt nhất của UWB là đo khoảng cách nhờ vào băng thông siêu rộng tạo nên miền thời gian với độ phân giải cao và miền khoảng cách được liên tục. Độ phân giải khoảng cách có thể đạt được xấp xỉ dựa trên công thức

$$rr \approx \frac{v}{2b'} \quad (1.1)$$

ở đó  $v$  là vận tốc của sóng và  $b'$  là băng thông. Ví dụ, theo giới hạn của FCC và độ lan truyền trong không gian (giả sử vận tốc của sóng bằng với vận tốc ánh sáng  $v = c0$ ),  $rr \approx 0.5 \times c0 / 7.5 \text{ GHz} = 2 \text{ cm}$  và theo giới hạn của FCC sẽ tương ứng là 6 cm.

UWB hỗ trợ các kỹ thuật bao gồm Time of Arrival (ToA), Two Way Ranging (TWR), Time Difference of Arrival (TDoA). Các kỹ thuật đều dựa trên việc tính toán chênh lệch thời gian và được chia thành 3 loại dựa trên các nguyên lý dưới đây

- Continuous Waves: trong dải tần số, các tần số khác nhau được sử dụng tuần tự bằng cách quét hoặc nhảy. Tín hiệu được phân tích trong miền tần số dẫn đến độ phân giải thời gian thấp, làm giảm độ hiệu quả của các ứng dụng thời gian thực. Các sóng liên tục sẽ cho phép việc định vị chính xác hơn, tuy nhiên lại không thể sử dụng cho các thiết bị nhỏ như điện thoại thông minh vì công nghệ này yêu cầu kích thước anten khá lớn. Nếu dải tần số rất rộng, thì cần phải có kích thước anten khá lớn để đạt được đủ độ hiệu quả của anten.
- Impulse Radio: UWB Impulse Radio (UWB - IR) có cấu trúc khá đơn giản và được sử dụng cho các phép đo khoảng cách với yêu cầu tốc độ nhanh. Độ dài của các xung đều trong vùng nano giây, thậm chí còn ít hơn. So với phương pháp sóng liên tục (continuous waves), các xung siêu ngắn thường ít bị ảnh hưởng với các tín hiệu đi theo các con đường khác, cho phép độ phân giải khoảng cách tốt hơn và từ đó làm giảm đi ảnh hưởng của tín hiệu đa đường. Vì các sóng radio phải được cấp nguồn trong thời gian ngắn trước và trong khi tạo ra xung, UWB – IR có mức tiêu thụ điện năng thấp hơn so với các kỹ thuật UWB khác. Các hệ thống định vị dựa trên kỹ thuật xung UWB được sử dụng với tốc độ lặp lại tương đối thấp khoảng 1MHz đến 100MHz, trái ngược với các hệ thống truyền thông sử dụng UWB ở tốc độ 1GHz đến 100GHz. Các nhà nghiên cứu đã sử dụng kỹ thuật TWR để tạo ra các hệ

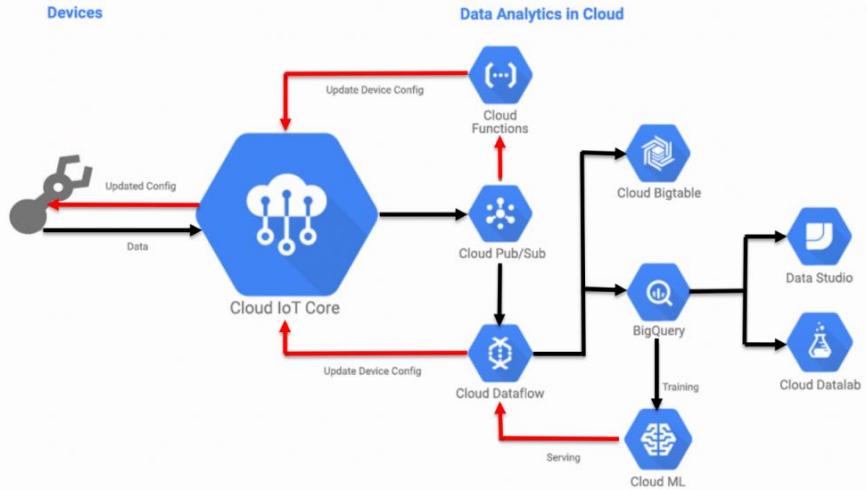
định vị với độ chính xác xấp xỉ 4cm, thậm chí đã xuống tới 1cm trong điều kiện LoS hoàn hảo trong phòng thí nghiệm.

## 1.3 Google Cloud IoT Core

### 1.3.1 *Tổng quan*

Internet of Things (Iots) vẫn đang phát triển vô cùng mạnh mẽ tạo ra một lượng dữ liệu khổng lồ, đặt ra bài toán về lưu trữ, xử lý dữ liệu và khai thác một cách hiệu quả nhất. Rất nhiều doanh nghiệp và công ty hiện tay đang chọn giải pháp xử lý dữ liệu trên các nền tảng đám mây thay vì xây dựng một lượng lớn máy chủ nội bộ bởi tính chất auto-scale cũng như những ưu thế về hạ tầng mà các dịch vụ đám mây đem lại. Với bề dày lịch sử về thu thập, lưu trữ và phân tích dữ liệu, không thể không nhắc đến các nền tảng và giải pháp mà Google mang lại trong các ứng dụng IoTs.

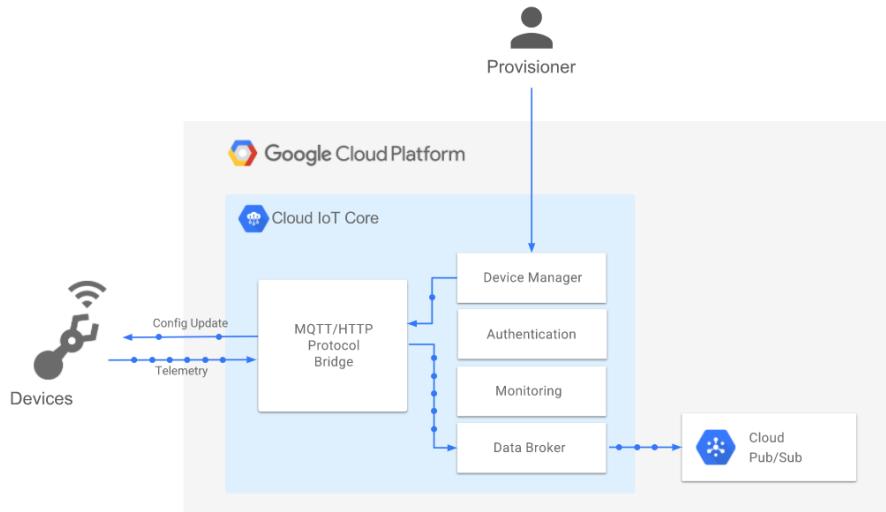
Dịch vụ Google Cloud IoT Core là một dịch vụ con nằm trong nền tảng Google Cloud Platform (GCP). Đây là một dịch vụ hữu ích cho phép chúng ta tạo kết nối hai chiều an toàn giữa các thiết bị IoT với nền tảng đám mây của Google thông qua các phương thức bảo mật tối ưu. Với nền tảng mà Google cung cấp, các nhà phát triển không cần phải bận tâm về cơ sở hạ tầng, thiết lập bảo mật, làm thế nào để trích xuất dữ liệu hay phân tích dữ liệu từ các thiết bị IoT sao cho hiệu quả. Vì bên cạnh dịch vụ GCIC, GCP còn cung cấp cho chúng ta rất nhiều dịch vụ con với các chức năng khác nhau như Cloud BigQuery, AutoML, DataLab,... Và Google Cloud IoT Core là dịch vụ nền tảng, đóng vai trò cầu nối quan trọng để các thiết bị IoT của chúng ta có thể dễ dàng tương tác với các dịch vụ đó.



**Hình 1.7 Mô hình mẫu giữa Cloud IoT Core với các dịch vụ khác của GCP**

Google Cloud IoT Core là một dịch vụ quản lý với hai thành phần chính là Device Management và Communication Broker:

- Device Management: Đây là thành phần cho phép chúng ta tổ chức việc kết nối và quản lý các thiết bị IoT với GCIC. Các thiết bị kết nối tới Google Cloud IoT Core sẽ được xem như là một “device”. Nhóm các “device” lại với nhau, chúng ta sẽ có một “registry” là ô chứa đại diện cho cả các “device” đó. Việc khởi tạo và quản lý “device” với “registry” sẽ phụ thuộc vào cách tổ chức của chúng ta.
- Communication Broker: đây là thành phần cung cấp các giao thức truyền thông để chúng ta có thể truyền nhận dữ liệu với GCIC, hiện GCIC hỗ trợ hai loại giao thức rất phổ biến đó là MQTT (sử dụng TLS) và HTTPS. Và một thành phần có tên Data Broker sẽ có nhiệm vụ chuyển tiếp, phân phối các luồng dữ liệu đến một dịch vụ khác để thực hiện lưu trữ xử lý có tên gọi là Cloud Pub/Sub.



**Hình 1.8 Google Cloud IoT Core**

### 1.3.2 Cấu trúc

#### 1.3.2.1 Các định nghĩa

- Thiết bị: là một đơn vị có thể kết nối internet và trao đổi dữ liệu với Cloud. Các thiết bị đó thường được gọi là thiết bị thông minh và giao tiếp thông qua hai kiểu dữ liệu: telemetry và state.
- Telemetry: là tất cả các dữ liệu dưới dạng sự kiện, ví dụ các phép đo thông số môi trường được gửi từ thiết bị tới cloud.
- State: một khối dữ liệu do người dùng tự định nghĩa mô tả trạng thái hiện tại của thiết bị.
- Registry: một tập hợp các thiết bị cùng chung các đặc điểm.

#### 1.3.2.2 Kết nối

Để một thiết bị kết nối tới GCIC, sẽ cần đảm bảo hai điều kiện là xác thực (authentication) và bảo mật TLS.

Với điều kiện xác thực, GCIC sử dụng xác thực không đối xứng với các public key:

- Thiết bị sử dụng một private key để đăng ký một JWT (JSON Web Token), với JWT là phần tử sử dụng cho việc xác thực ngắn hạn giữa các thiết bị và server. Với kết nối MQTT, JWT sẽ nằm ở phần password trong bản tin CONNECT, như một chứng nhận cho định danh của thiết bị

- Dịch vụ sử dụng một public key (được lưu sẵn trước khi JWT được gửi) để xác thực danh tính của thiết bị. Cặp public key và private key được tạo ra cùng một nguồn và có liên hệ mật thiết với nhau

Với bảo mật TLS, điều quan trọng nhất thiết bị cần đó là cần lưu sẵn chứng chỉ CA gốc để có thể kết nối đến MQTT Broker hay thiết lập kết nối HTTPS. Google hỗ trợ root CA có dung lượng rất nhỏ (< 1 KB), phù hợp với dung lượng lưu trữ của các thiết bị tại hiện trường, mà hầu hết là có dung lượng nhỏ.

#### 1.3.2.3 Topic MQTT

GCIC quy định hai loại topic để phục vụ các luồng thông tin từ device lên cloud bao gồm telemetry topic và state topic.

Hai loại topic đều quy định các tiền tố và các thiết bị phải tuân theo các topic đó để có thể tham gia vào luồng dữ liệu của mỗi registry.

- Telemetry: /devices/*DEVICE\_ID*/events
- State: /devices/*DEVICE\_ID*/state

Với mỗi loại sự kiện khác nhau, nội dung sẽ được quy định và phân biệt ở payload. Thường thì payload sẽ sử dụng cấu trúc JSON, và có các trường nhất định để phân biệt các sự kiện và trạng thái khác nhau.

Ví dụ với bản tin thiết bị gửi dữ liệu nhiệt độ lên cloud, topic và payload khi đó có thể có dạng như sau:

Topic: /devices/ESP\_3410656776/events/

Payload: {

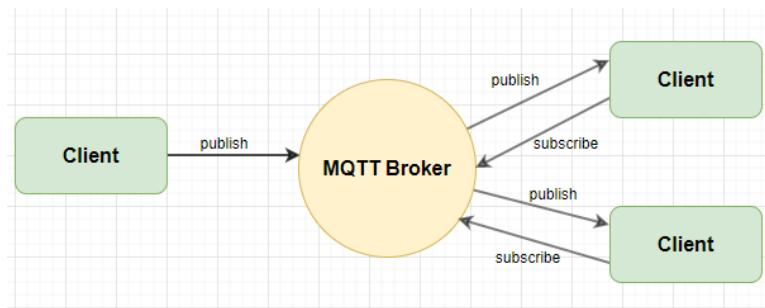
```

        “id”: 3410656776,
        “method”: “temperature”
        “unit”: “C”
        “value”: {
            “temperature”: 32
        }
    }
```

## 1.4 MQTT

MQTT là viết tắt của MQ Telemetry Transport. Đây là một giao thức theo dạng publis/subscribe, giao thức cực kì đơn giản và có dung lượng nhẹ, được thiết kế cho các mạng băng thông thấp, độ trễ cao. Nó là một giao thức lý tưởng cho các ứng dụng machine-to-machine (M2M) hoặc trên IoT, và cho các ứng dụng di động có băng thông và năng lượng pin ở mức cao.

Mô hình hoạt động của MQTT được mô tả như Hình 1.9. Trong hệ thống sử dụng MQTT, các node (client) được kết nối tới một điểm trung gian gọi là broker (server). Mỗi client sẽ đăng ký (subscribe) với một hoặc nhiều topic ví dụ như `/home/topic1`, `/home/topic2`. Ngoài ra các client cũng có thể gửi (publish) dữ liệu lên một hoặc một vài topic. Khi một client A publish dữ liệu vào `topic1` thì tất cả các client đã subscribe vào `topic1` đều nhận được dữ liệu đó.



Hình 1.9 Giao thức MQTT

Một vài khái niệm trong MQTT:

- MQTT Client (Publisher, Subscriber): Các node (có thể là node cảm biến, điện thoại, laptop) kết nối với MQTT Broker.
- MQTT Server (Broker): được coi như server, nơi sẽ lưu trữ, nhận các bản tin từ client và phân phối tới các client khác đã subscribe vào topic đó.
- Topic: là các hàng đợi chứa các message cho phép các client trao đổi thông tin và dữ liệu với nhau.
- Session: Một session được định nghĩa là kết nối từ client đến server. Tất cả các giao tiếp giữa client và server đều là 1 phần của session.
- Subscription: Không giống như session, subscription về mặt logic là kết nối từ client đến topic. Khi thực hiện subscribe đến topic, client có thể trao đổi messages với topic.

- Message: Message là các đơn vị dữ liệu được trao đổi giữa các clients.
- QoS: Các conneciton của MQTT được thực hiện dựa trên TCP/IP, do đó MQTT đưa ra 3 loại QoS (Qualities of Service) khi publish và subscribe là:
  - QoS0 – At most once: message được truyền nhận dựa hoàn toàn vào tính tin cậy của TCP/IP. Việc mất hoặc lặp message có thể xảy ra. Có thể ví dụ một trường hợp như trong môi trường sensor mà việc mất một gói dữ liệu tại một thời điểm không ảnh hưởng đến toàn bộ quá trình.
  - QoS1 – At least once: message được gửi với ít nhất một lần xác nhận từ đầu kia, nghĩa là có thể có nhiều hơn một lần xác nhận đã nhận được dữ liệu.
  - QoS2 – Exactly once: message được đảm bảo được gửi đi và bên nhận chỉ nhận được đúng một lần, quá trình này cần phải trải qua bốn bước bắt tay.
- Retain: Retain là một cờ (flag) được gắn cho một message của giao thức MQTT. Retain nhận 2 giá trị là 1 hoặc 0. Nếu một message có retain = 1, broker sẽ lưu lại message cuối cùng có retain = 1 đó của một topic kèm theo mức QoS tương ứng. Khi Client bắt đầu subscribe vào topic đó sẽ ngay lập tức nhận được message đó.

Giao thức MQTT mang lại nhiều lợi ích khi truy cập dữ liệu IoT, các ưu điểm của MQTT có thể kể đến đó là:

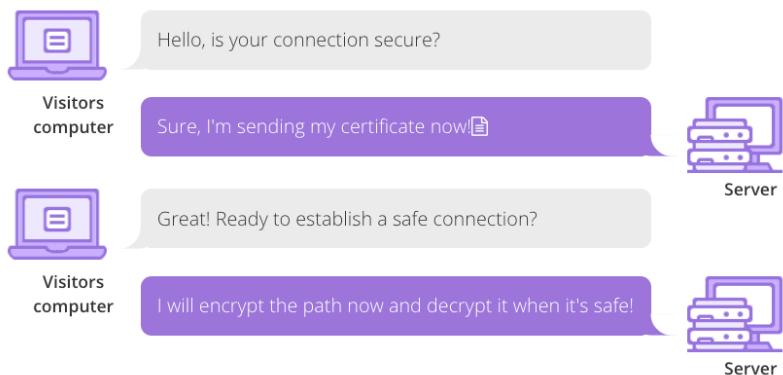
- Truyền thông tin hiệu quả hơn
- Tăng khả năng mở rộng
- Giảm đáng kể tiêu thụ băng thông mạng
- Rất phù hợp cho điều khiển và đo thám
- Chi phí thấp
- Rất an toàn, bảo mật
- Được sử dụng trong các ngành công nghiệp dầu khí, các công ty lớn như Amazon, Facebook,...
- Tiết kiệm thời gian phát triển
- Giao thức publish/subscribe thu thập nhiều dữ liệu hơn và tốn ít băng thông hơn so với giao thức cũ.

## 1.5 TLS

TLS là viết tắt của Transport Layer Security. Đây là một dạng giao thức bảo mật cung cấp mức độ riêng tư cao, cũng như tính toàn vẹn của dữ liệu khi giao tiếp bằng mạng và internet. TLS là tiêu chuẩn được sử dụng trong việc bảo mật các ứng dụng web và trang web trên khắp thế giới kể từ khi được giới thiệu vào năm 1999. Nó là sự kế thừa và thay thế cho hệ thống SSL.

Để có thể bảo mật được thông tin, TLS định nghĩa ra chứng chỉ TLS và các private key, public key để mã hóa đường truyền cũng như đảm bảo được người tham gia kết nối. Mục đích của TLS là bảo mật các thông tin nhạy cảm trong quá trình truyền trên internet như thông tin cá nhân, thông tin thanh toán, thông tin đăng nhập. Nó là giải pháp thay thế cho phương pháp truyền thông tin văn bản dạng plain text, văn bản loại này khi truyền trên internet sẽ không được mã hóa, nên việc áp dụng mã hóa vào sẽ khiến các bên thứ ba không xâm nhập được vào thông tin của bạn.

Chứng chỉ TLS hoạt động bằng cách tích hợp key mã hóa vào thông tin định danh. Nó sẽ giúp công ty mã hóa mọi thông tin được truyền mà không bị ảnh hưởng hoặc chỉnh sửa bởi các bên thứ ba.



**Hình 1.10 Cách hoạt động của chứng chỉ TLS**

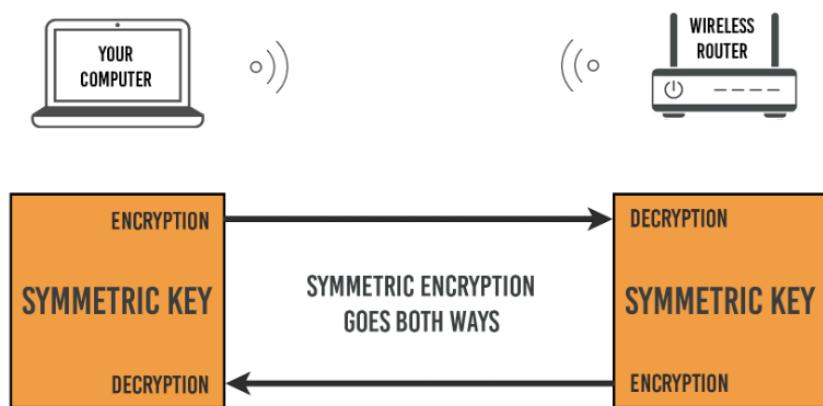
TLS hoạt động bằng cách sử dụng public và private key, đồng thời các khóa duy nhất của mỗi phiên kết nối (session). Trong phiên kết nối ban đầu, public và private key được dùng để tạo session key, vốn được dùng để mã hóa và giải mã dữ liệu được truyền. Session key được sử dụng trong một khoảng thời gian nhất định và chỉ có thể dùng trong phiên kết nối này.

Khi thực hiện kết nối, máy tính sẽ kiểm tra chứng chỉ và xác nhận những điểm sau đây:

- Certificate là đúng tổ chức có chứng nhận phát hành hay không
- Server đang gửi tin có khớp với thông tin server đang được mô tả trong chứng chỉ hay không. Sau khi xác nhận là đúng server thì an tâm bắt đầu thực hiện truyền tin.

TLS được sử dụng phổ biến nhất với HTTPS, ví dụ rõ nhất đó là mỗi khi sử dụng các trình duyệt web để truy cập một trang web nào đó, các trang sử dụng HTTPS, cụ thể hơn là TLS được đánh dấu bảo mật. Các trang sử dụng HTTP thường được cảnh báo hoặc hạn chế nội dung bởi trình duyệt.

Lấy HTTPS để làm ví dụ cho cách TLS hoạt động. Nguyên lý đầu tiên để có thể bảo mật là tất cả dữ liệu sẽ bị mã hóa, và để giải mã, mỗi bên sẽ dùng một chìa khóa đặc biệt để giải mã.Thêm vào đó, khi trao đổi thông tin liên quan đến chìa khóa giải mã, thông tin đó cũng sẽ phải cần mã hóa hoặc có một cách để có thể bảo vệ thông tin về chìa khóa đó. Đó là toàn bộ quá trình về mặt lý thuyết của TLS. Khi cả hai bên đều sử dụng cùng một chìa khóa để giải mã, mã hóa, nó được gọi là mã hóa đối xứng. Người dùng sẽ chỉ có một chìa khóa, hay còn gọi là “mật khẩu”, ví dụ như trong WiFi, bạn chỉ cần đưa chìa khóa đó vào cả hai thiết bị router và máy tính của bạn, hai bên sẽ kết nối với nhau và trao đổi dữ liệu mã hóa dựa trên chìa khóa đó.

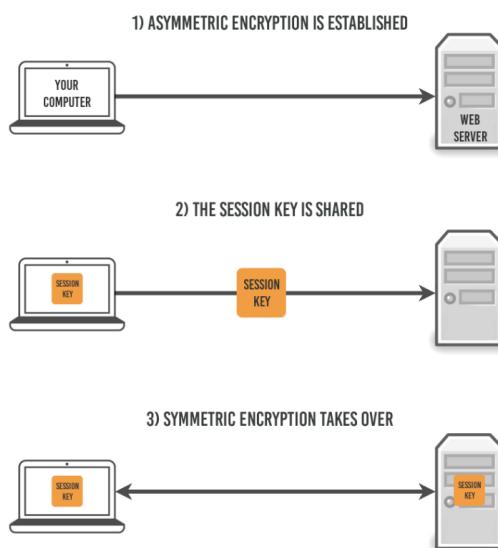


**Hình 1.11 Mã hóa đối xứng**

Tuy nhiên, câu chuyện sẽ trở nên khó hơn khi kết nối với Internet. Mã hóa đối xứng sẽ không thể dùng được bởi điểm kết nối cuối cùng của người dùng hoàn toàn không được biết trước, vậy làm thế nào để có thể chia sẻ được chìa khóa với bên còn

lại để những hacker ở giữa đường truyền từ người dùng sang máy chủ không thể can thiệp vào nội dung đó. Vấn đề được giải quyết với mã hóa không đối xứng, sử dụng hai chìa khóa hoàn toàn khác nhau, một chìa khóa để mã hóa, một chìa khóa để giải mã. Tuy là khác nhau về nội dung, nhưng hai chìa đều được tạo ra cùng một lúc và từ một nguồn, có mối liên hệ với nhau. Khi một bên sử dụng một chìa để mã hóa, dữ liệu nhận được ở bên kia sẽ chỉ có thể dùng chìa còn lại để giải mã. Hai chìa được định nghĩa là public key và private key.

Khi hai bên bắt đầu trao đổi thông tin kết nối, phía server sẽ gửi sang gói thông tin bao gồm public key. Sau đó thông tin được phía client gửi sang sẽ là session key được tạo ra bởi trình duyệt, được mã hóa bởi public key và khi nhận được ở phía server, được giải mã bởi private key. Sau khi đã có session key, cả hai bên đều đã có chung một chìa khóa và quá trình trao đổi dữ liệu bắt đầu, tương tự như ở mã hóa đối xứng.



**Hình 1.12 Quá trình trao đổi session key trong TLS**

## 1.6 Kết luận chương

Ở chương này, đồ án đã mô tả tổng quan hệ thống định vị thời gian thực trong nhà, cùng với đó là lý thuyết về công nghệ ultra wideband và các ứng dụng của công nghệ đó ở hiện tại cũng như tương lai. Từ đó thấy được một hệ thống định vị trong nhà gồm ba phần chính:

- Hệ thống định vị tại hiện trường
- Gateway quản lý hệ thống định vị
- Cloud lưu trữ thông tin và Web hiển thị

Ở chương tiếp theo, đồ án sẽ đi vào trình bày tổng thể hệ thống, dựa vào các nguồn lực có sẵn, để từ đó đưa ra các phân tích, lựa chọn phần cứng, phần mềm để xây dựng được hệ thống định vị thời gian thực sử dụng công nghệ Ultra wideband.

## **CHƯƠNG 2. THIẾT KẾ HỆ THỐNG**

Trong Chương 2, đồ án sẽ trình bày về tổng thể hệ thống, lựa chọn phần cứng cho các tầng trong hệ thống và các giao thức truyền thông để hệ thống tại hiện trường có thể giao tiếp với cloud.

### **2.1 Mục tiêu**

Trong chương 1, đồ án đã mô tả lại hệ thống định vị trong nhà sử dụng công nghệ UWB và các công nghệ, hệ thống để có thể ứng dụng các hệ thống UWB vào các bài toán quản lý, theo dõi con người, vật thể như xe nâng, xe chở hàng bên trong các kho, xưởng. Mục tiêu của đồ án sẽ thiết kế một hệ thống gồm ba tầng: hiện trường, hệ thống quản lý mạng UWB tại hiện trường và kết nối tới tầng thứ ba là cloud. Ứng dụng của hệ thống sẽ dành cho các nhà xưởng, kho trong nhà, nơi mà sóng GNSS không thể đảm bảo được độ chính xác. Các thiết bị mà hệ thống có thể theo dõi là các xe nâng di chuyển hàng hóa, để từ đó có thể kiểm soát được vị trí của các pallet hàng. Hệ thống sẽ đảm bảo được tốc độ cập nhật từ 0.5 Hz – 5 Hz cho mỗi đối tượng cần định vị và khoảng cách của tối đa của một mạng để hoạt động ổn định là một khu vực với chiều dài, chiều rộng, chiều cao lần lượt là 30m, 30m và 10m.

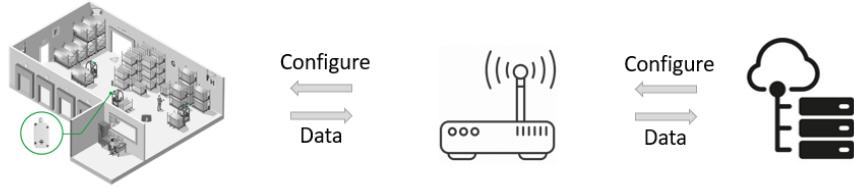
Với cloud, em sử dụng một hệ thống cloud được xây dựng sẵn dành cho các bài toán IoT, có mô hình giống như Google Cloud IoT, được hỗ trợ bởi team Phần mềm.

Tầng tiếp theo là tầng hệ thống quản lý mạng UWB tại hiện trường sẽ hướng tới một hệ thống đảm bảo được tính thời gian thực, quản lý được tất cả các thiết bị trong mạng UWB. Thiết bị đảm nhận vai trò quản lý đó cần được phân tích, thiết kế kĩ càng, hướng tới tiềm cận sản phẩm thương mại.

Tầng cuối cùng là tầng mạng UWB tại hiện trường cần đảm bảo các yêu cầu về chi phí, tần số sử dụng trong dải cho phép tại Việt Nam và độ chính xác đảm bảo trong khoảng từ 10 – 50cm.

### **2.2 Thiết kế tổng thể**

Hệ thống định vị trong nhà được mô tả như hình 2.1



**Hình 2.1 Tổng quan hệ thống**

Trong hệ thống trên, đồ án sẽ thiết kế hệ thống cho phần tại hiện trường, bao gồm hệ thống định vị bằng công nghệ UWB, phần tử Gateway là cầu nối giữa hệ thống UWB với cloud. Với hệ thống định vị sử dụng UWB, cần lựa chọn các mô hình mạng và thiết bị dễ lắp đặt, có độ ổn định cao cũng như có dải tần số phù hợp với dải tần số của Việt Nam quy định là từ 4.2 – 4.8 GHz. Cùng với đó, thiết bị UWB cũng cần hỗ trợ các mô hình mạng có các lựa chọn để dễ dàng xây dựng Gateway, hỗ trợ đầy đủ các thông tin cần thiết và có thể cấu hình các thông số cho thiết bị đó từ xa, thay vì việc phải trực tiếp nạp lại firmware cho các thiết bị đó.

Với luồng thông tin của Gateway và Cloud, giao thức được ưu tiên sử dụng sẽ là MQTT bởi thông tin của hệ thống tuy cần được cập nhật liên tục nhưng khối lượng dữ liệu trong mỗi bản tin là không lớn. Lựa chọn MQTT vẫn đảm bảo được tính real-time của hệ thống, mặt khác phần cứng sẽ có thể lựa chọn những bộ xử lý trung tâm nhỏ gọn hơn, giảm đi giá thành và kích thước của Gateway. Để có thể truy cập Cloud, Gateway cũng cần tích hợp thêm bảo mật SSL cho từng bản tin gửi đi.

Trong các phần tiếp theo, đồ án sẽ phân tích lựa chọn các thành phần dựa theo mô hình tổng thể, bao gồm phần cứng của hệ thống mạng UWB, phần cứng của Gateway cũng như mô hình mạng của thống định vị dựa trên phần cứng đã chọn.

### 2.3 Module Ultra wideband

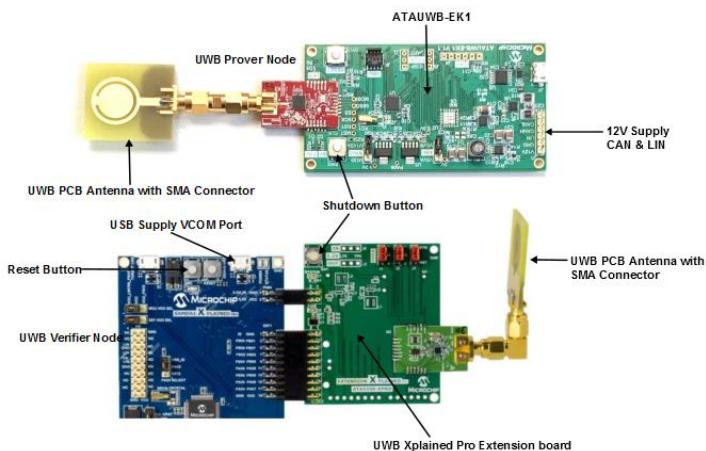
Thị trường Ultra wideband hiện tại đang được rất nhiều nhà sản xuất chip quan tâm và nghiên cứu, phát triển. Trong vòng 10 năm trở lại đây, hàng loạt module UWB đã được tung ra thị trường tuy nhiên chức năng, nguyên lý đều cơ bản giống nhau.

### 2.3.1 Microchip ATA835x

Dòng chip ATA835 của Microchip là một trong những dòng chip mới nhất, được thiết kế sử dụng năng lượng rất thấp với các tầng bảo mật cho các ứng dụng định vị, giao tiếp điểm - điểm. ATA835 có các thông số cơ bản như sau:

- Tần số hoạt động: 6.2 – 7.8 GHz
- Phù hợp với quy chuẩn UWB của ETSI và FCC
- Tốc độ truyền qua UWB đạt 246Kbps
- Giao tiếp với MCU qua SPI (20 Mbps)
- Sử dụng phương pháp đo ToF với độ phân giải 15cm
- Năng lượng tiêu thụ cho truyền nhận thấp (39mA khi nhận), phù hợp với các ứng dụng sử dụng pin Cell.

Microchip còn hỗ trợ các kit thử nghiệm cho dòng ATA835 như ở hình 2.1. Giá hiện tại của dòng chip này trung bì vào khoảng 9\$ / 1pcs.



Hình 2.1 Kit thử nghiệm ATA835

### 2.3.2 NXP SR150

SR150 là bộ điều khiển UWB được tích hợp các phương pháp đo TWR và TDoA, với độ chính xác đạt được lên tới 10cm.

- Tần số hoạt động: 6.24 – 8.24 GHz
- Công suất truyền lên tới 10 dBm, có thể lập trình được
- Hỗ trợ nhiều mức tốc độ truyền khác nhau: 850 kbps, 6.8Mbps, 7.8Mbps, 27Mbps và 31.2 Mbps

- Lõi xử lý Arm Cortex-M3
- Phù hợp cho các ứng dụng IOT, định vị trong nhà, các thiết bị Smarthome
- Công suất tiêu thụ tối đa: 920mW



**Hình 2.2 Kit phát triển dựa trên SR150**

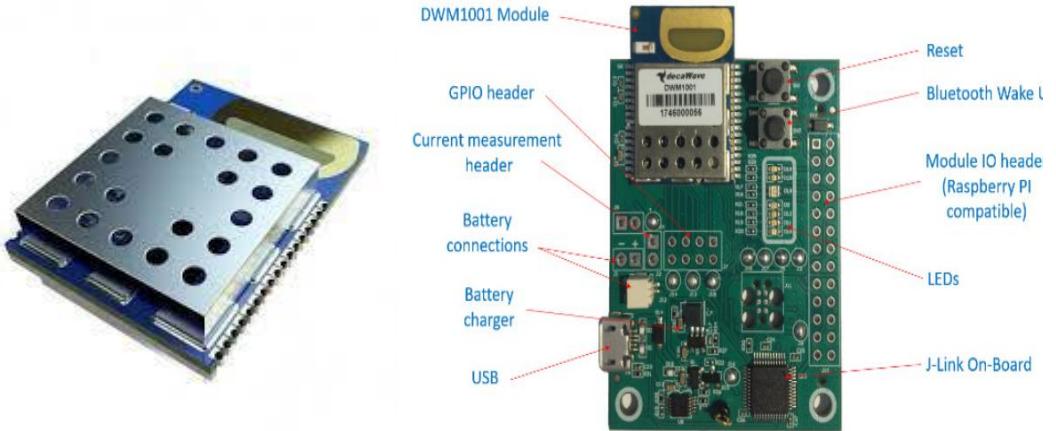
SR150 còn có một người anh em khác là SR100, được Samsung sử dụng để tích hợp vào Samsung Galaxy Note20 Ultra, là một trong những sản phẩm thương mại đầu tiên tích hợp Ultrawideband cùng với dòng chip U1 của Apple.

### 2.3.3 Decawave DW1000

Decawave là cái tên không xa lạ gì với công nghệ UWB, là một trong những nhà phát hành chip UWB đầu tiên và vẫn đang chiếm thị phần lớn trong thị trường chip UWB. Các sản phẩm ứng dụng công nghệ UWB hiện nay đều sử dụng các module phổ biến của Decawave như DW1000, DW3000 vì giá thành hợp lí và độ ổn định cao, tài liệu giao tiếp được hỗ trợ nhiều.

DW1000 là module được phát hành năm 2013, là một trong những chip thương mại đầu tiên hỗ trợ công nghệ UWB. Đến nay, DW1000 vẫn là một trong những lựa chọn hàng đầu của các sản phẩm UWB nhờ vào công suất tiêu thụ thấp, hỗ trợ hai thuật toán định vị TWR và TDoA cho độ chính xác lên tới 10cm.

- Tần số hoạt động: 3.5 – 6.5 GHz
- Mức tiêu thụ năng lượng thấp: 1 uA với chế độ SLEEP, 50nA với chế độ DEEPSLEEP.
- Phù hợp với quy định về phổ tần của FCC và ETSI
- Tốc độ truyền dữ liệu: 110kbps, 850 kbps, 6.8 Mbps
- Giao tiếp với MCU qua SPI
- Khoảng cách giao tiếp tối đa trong điều kiện không vật cản: 50m



**Hình 2.3 Module  
DWM1001**

**Hình 2.4 Kit phát triển DWM1001-  
DEV**

Hình 2.3 là module DWM1001, hỗ trợ chip DW1000, cùng với SoC NRF52832, bổ sung thêm giao tiếp qua Bluetooth Low Energy. Với BLE, thông tin về vị trí có thể được theo dõi từ xa, các cấu hình cho thiết bị cũng có thể được cấu hình từ xa mà không phải cấu hình trực tiếp qua việc thay đổi firmware. Thêm vào đó, BLE và BLE Mesh đang là các giao thức và mô hình mạng được sử dụng rất nhiều trong các ứng dụng về IoT, Smarthome, giúp cho việc tích hợp hệ thống định vị trong nhà và hệ thống IoT sẵn có trở nên dễ dàng hơn.

Kit phát triển DWM1001-DEV ở hình 2.4 được thiết kế sử dụng nguồn pin 4.2V, hoặc nguồn 5V từ cổng Micro USB, hỗ trợ cổng sạc pin 4.2V Micro USB, được ra sẵn các chân GPIO cho người dùng, cùng với đó là hàng chân IO tích hợp theo Raspberry Pi 3, giúp người dùng có thêm lựa chọn gateway cho hệ thống.

Với những ưu điểm cùng với giá thành hợp lý, dễ tìm kiếm trên thị trường, em đã lựa chọn module DWM1001, cụ thể là kit phát triển DWMM1001-DEV để có thể nghiên cứu và phát triển hệ thống mạng định vị thời gian thực trong nhà.

## 2.4 Mô hình mạng hệ thống dựa trên module DWM1001

Module DWM1001 được Decawave thiết kế rất nhiều mô hình mạng phục vụ cho cả mục đích thử nghiệm và thương mại. Nhà sản xuất hỗ trợ hai phương pháp TWR và TDoA. Như đã phân tích ở mục 1.2.1, phương pháp TDoA với ưu điểm tiết kiệm năng lượng cho các thiết bị di động, thường được ưu tiên cho các ứng dụng khắt khe về năng lượng. Tuy nhiên, với phương pháp TDoA, nhà sản xuất yêu cầu trả phí

khá cao để được hỗ trợ, và không mất phí với phương pháp TWR. Với giới hạn về mặt kinh tế của đồ án sinh viên, em sẽ chọn phương pháp TWR cho mô hình mạng của hệ thống.

Decawave hỗ trợ nhiều mô hình mạng phục vụ cho cả mục đích thử nghiệm và thực tế. Đồ án sẽ chỉ đề cập đến các mô hình thực tế.

Hệ thống dựa trên DWM1001 bao gồm các thiết bị được dán nhãn như sau:

- Anchor: là thiết bị đã có vị trí được đặt sẵn bởi người dùng từ trước. Vị trí của anchor bao gồm 3 trục x, y, z. Độ chính xác của hệ thống một phần phụ thuộc vào độ chính xác của vị trí đặt trước cho anchor và số lượng anchor trong mạng. Yêu cầu tối thiểu để đảm bảo độ chính xác 10cm là 4 anchor trong một mạng, hay nói cách khác, tag cần biết vị trí của tag so với ít nhất 4 anchor.
- Tag: là thiết bị được gắn trên người, vật cần biết vị trí. Với phương pháp đo TWR, tag sẽ là thiết bị chủ động gửi đi các bản tin REQUEST tới các anchor để bắt đầu quá trình tính toán vị trí từ tag đến anchor. Sau khi đã có vị trí của tag đến 4 anchor, tag sẽ tự tính toán vị trí của nó trong mạng và gửi vị trí đó ra bên ngoài thông qua BLE. Với DWM1001, Tag sẽ có hai chế độ hoạt động:
  - Chế độ chủ động: trong chế độ chủ động, tag chủ động trao đổi các gói tin. Tuy nhiên việc trao đổi gói tin sẽ phải phụ thuộc vào sự sắp xếp của hệ thống, vào những slot phù hợp. Chip DW1000 sẽ không vào trạng thái ngủ, mà chỉ chuyển sang trạng thái chờ khi không trong thời điểm gửi bản tin, vì điều khiển nRF52832 sẽ vào trạng thái ngủ nếu không còn nhiệm vụ gì cần xử lý. Chế độ này sẽ được ưu tiên sử dụng nếu hệ thống cần sử dụng Bluetooth
  - Chế độ năng lượng thấp: với chế độ năng lượng thấp, DW1000 sẽ chủ động vào chế độ ngủ khi không trong thời điểm gửi bản tin TWR, và sẽ thức dậy vào trong lần gửi bản tin tiếp theo. MCU cũng sẽ vào chế độ ngủ cùng với các ngoại vi khác trừ RTC và cảm biến gia tốc. Khi đó, DWM1001 sẽ ở trạng thái tiêu thụ năng lượng thấp nhất.

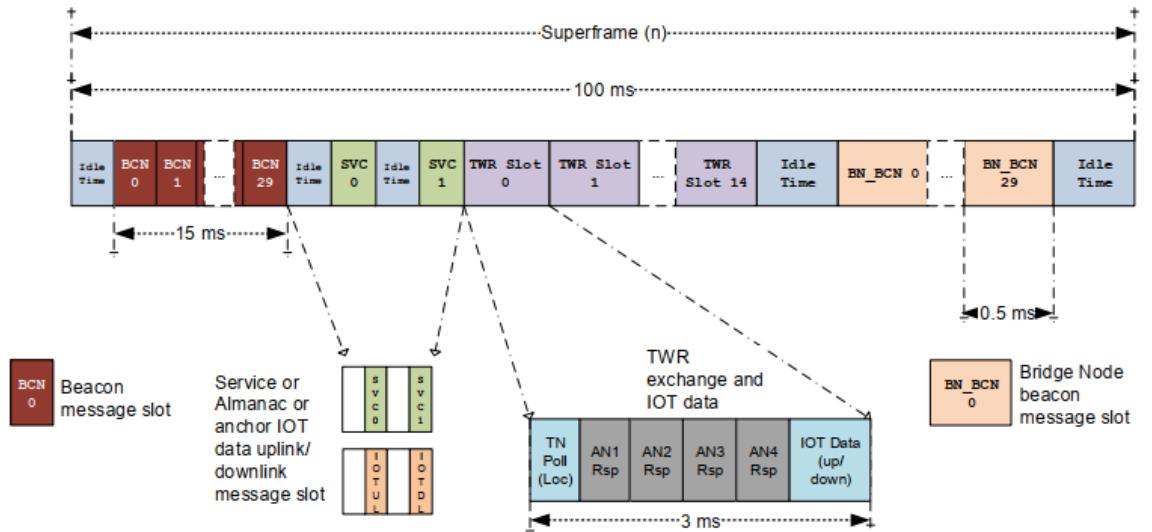
- Listener/Brigde: thiết bị lắng nghe các bản tin UWB từ mạng và chuyển tiếp ra bên ngoài, phục vụ các mục đích theo dõi, quan sát từ xa.
- Initiator: là một trong 4 anchor của mạng UWB, có vai trò là thiết bị khởi tạo mạng, bắt đầu cho phép trao đổi thông tin và có thể cập nhật firmware mới cho các node khác trong mạng qua UWB.

Module DWM1001 được lập trình để hỗ trợ cả ba vai trò trên, bằng cách cấu hình qua phần mềm của nhà sản xuất để nạp firmware hoặc cấu hình qua BLE.

#### **2.4.1 Mô tả hoạt động của mạng UWB**

Hệ thống UWB RTLS có mục đích chính là theo dõi vị trí của ác tag và có thể trao đổi dữ liệu của hệ thống ra bên ngoài, ví dụ như đưa lên cloud thông qua MQTT, sử dụng bridge node.

DRTLS sử dụng phương pháp truy nhập TDMA. Các node hoạt động sử dụng một siêu khung truyền có độ dài 100ms được mô tả ở hình 2.5. Initiator sẽ điều khiển khung thời gian và siêu khung truyền gồm 30 vị trí cho bản tin Beacon được gửi từ phía anchor. Tiếp theo là hai vị trí của bản tin Service (SVC), sử dụng cho trao đổi dữ liệu của bridge tới các anchor. Các vị trí tiếp theo là 15 vị trí cho quá trình trao đổi TWR, được sử dụng cho quá trình tính toán vị trí của tag lần lượt so với 4 anchor như đã mô tả ở mục 1.2.1.3. Cuối cùng là vị trí cho các bản tin của bridge để thông báo tới các tag nếu có dữ liệu cấu hình từ trên gửi xuống. Để có thể tham gia vào mạng, các phần tử phải đảm bảo tuân theo từng thời điểm của các vị trí trong siêu khung truyền, nếu không, dữ liệu nhận được có thể bị sai lệch, cũng như dữ liệu gửi đi sẽ không thể đến đúng vị trí cần nhận.



**Hình 2.5 Cấu trúc của siêu khung truyền**

Các tag sẽ giao tiếp với 4 anchor và tính được khoảng cách từ tag đến mỗi anchor đó, sau đó vị trí hiện tại của tag so với 4 anchor sẽ được tính dựa trên Location Engine đã được nạp sẵn trong mỗi thiết bị của Decawave. Sau đó vị trí sẽ có thể gửi qua Bluetooth hoặc gửi tới node bridge.

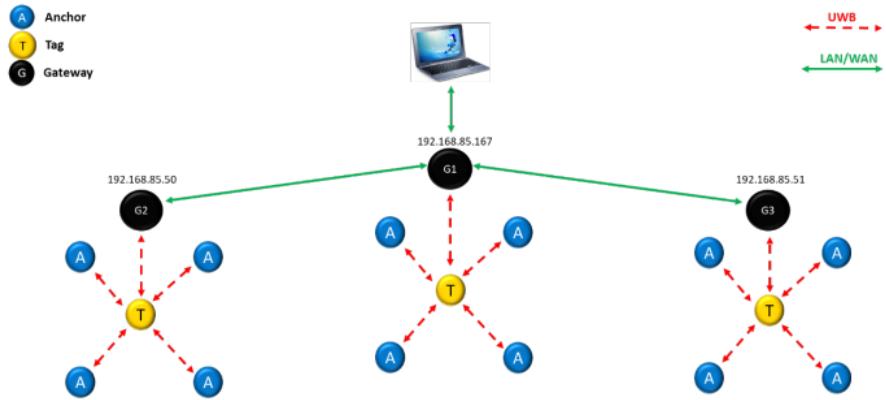
Một mạng tối đa sẽ có 30 anchor tương ứng với 30 vị trí trong superframe và không giới hạn tag. Tần số của hệ thống là 150Hz, khi đó với 15 tag, vị trí của tag sẽ cập nhật với tốc độ 10 Hz, 150 tag sẽ là 1 Hz và sẽ theo công thức sau:

$$\text{Tốc độ cập nhật vị trí} = \frac{150}{\text{số tag}}$$

#### 2.4.2 Các mô hình mạng cho DWM1001

##### 2.4.2.1 Mô hình sử dụng bridge node

Để phục vụ mục đích thử nghiệm cũng như giới thiệu các mô hình mạng cho DWM1001, Decawave đã đưa ra một mô hình mạng tận dụng hết được tài nguyên của một mạng, đó là sử dụng bridge node, cùng với đó bridge node cũng được kết nối với mạng local hoặc mạng Internet và đưa dữ liệu tới máy chủ hoặc cloud. Chắc chắn đây là một trong những mô hình mạng tiện dụng nhất cho các hệ thống RTLS bởi các thông tin của mạng đều được trao đổi qua UWB, đảm bảo được tốc độ cập nhật thông tin của mạng.



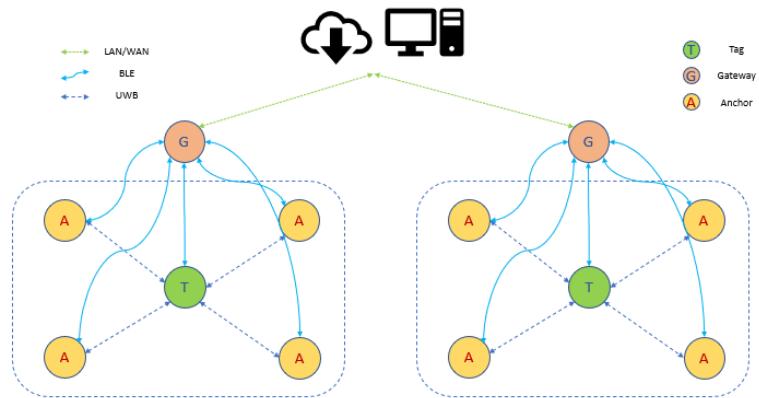
**Hình 2.6 Mô hình sử dụng bridge node**

Decawave hỗ trợ người dùng phần mềm của bridge node với lõi xử lý là Raspberry Pi 3. Module MDEK1001 đã được tích hợp sẵn hàng GPIO tương ứng với Raspberry Pi, người dùng chỉ cần cắm đúng vị trí là có thể sử dụng. Trước đó thì Raspberry Pi cần phải được nạp image được cung cấp từ Decawave, bên trong chứa chương trình của bridge node. Tuy nhiên, chương trình sử dụng cho bridge node đã được nhà sản xuất biên dịch và không được công khai. Cho nên để có thể tạo ra được một chương trình bridge node như nhà sản xuất đã làm, người dùng sẽ cần nghiên cứu lại từ đầu.

Với mô hình ở hình 2.6, mỗi mạng RTLS sẽ đặc trưng bởi một PANID, và sẽ có một bridge node để quản lý thông tin của mạng đó, đưa lên máy chủ hoặc nhận thông tin từ máy chủ gửi xuống qua mạng LAN hoặc WAN. Các bridge node đều có thể quản lý được tất cả anchor và tag trong mạng đó, cũng như tốc độ cập nhật thông tin cũng được đảm bảo bởi lõi xử lý mạng mẽ của Raspberry Pi.

Như đã nói ở trên, mô hình mạng này là mô hình tận dụng được hết khả năng của mạng UWB và khá dễ dàng để triển khai. Tuy nhiên, việc thiết kế lại bridge node là khá khó khăn, bởi để có thể đảm bảo tính đúng về thời gian của hệ thống, tuân theo các vị trí bản tin của siêu khung truyền (2.4.1) cần nhiều thời gian để nghiên cứu và thử nghiệm, thậm chí là có thể xây dựng lại quá trình giao tiếp trong mạng.

#### 2.4.2.2 Mô hình sử dụng gateway BLE



Hình 2.7 Mô hình sử dụng gateway BLE

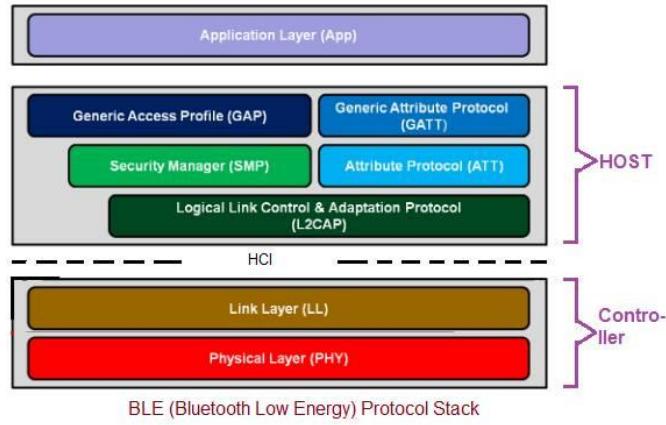
Dựa trên tính năng giao tiếp BLE của DWM1001, một mô hình khác được sử dụng ngoài mô hình sử dụng bridge node, là mô hình sử dụng gateway BLE. Về mặt quản lý, mô hình này khá giống với mô hình ở mục 2.4.2.1, chỉ có một điểm khác node gateway giao tiếp với các node trong mạng bằng BLE.

Mạng BLE vẫn đảm bảo được tốc độ với tốc độ theo lý thuyết với BLE 5.0 là 2MBps, khoảng cách của BLE tối đa lên đến 70 – 100m trong điều kiện LoS. Một ưu điểm khác của BLE đó là khả năng tích hợp và mở rộng, bởi trong 10 năm trở lại đây, BLE đang là một trong những giao thức được ưu tiên hàng đầu trong các hệ thống SmartHome cũng như các hệ thống cảm biến, chấp hành trong nhà. Nếu sử dụng mô hình mạng với gateway BLE, việc tích hợp vào các hệ thống sẵn có sử dụng BLE sẽ trở nên dễ dàng hơn.

Sau khi đưa ra so sánh và tính toán các ưu, nhược điểm của hai hệ thống, đồ án sẽ sử dụng mô hình với gateway BLE. Phản tiếp theo, đồ án sẽ trình bày về công nghệ BLE và cách sử dụng, giao tiếp trong chuẩn BLE.

#### 2.4.3 Công nghệ Bluetooth Low Energy

Công nghệ Bluetooth Low Energy là công nghệ dựa trên sóng radio 2.4GHz, được công bố lần đầu vào năm 2010 trong tên gọi Bluetooth 4.0. Đến nay chuẩn BLE thông dụng đã đạt đến Bluetooth 5.0 với tốc độ hỗ trợ lên tới 2MBps ở tầng PHY.



**Hình 2.8 BLE Stack**

Hình 2.8 mô tả cấu trúc của chuẩn BLE. Người lập trình sẽ sử dụng BLE để giao tiếp thông qua các giao thức được cung cấp bao gồm Generic Access Profile (GAP), Attribute Protocol (ATT) và Generic Attribute Profile (GATT).

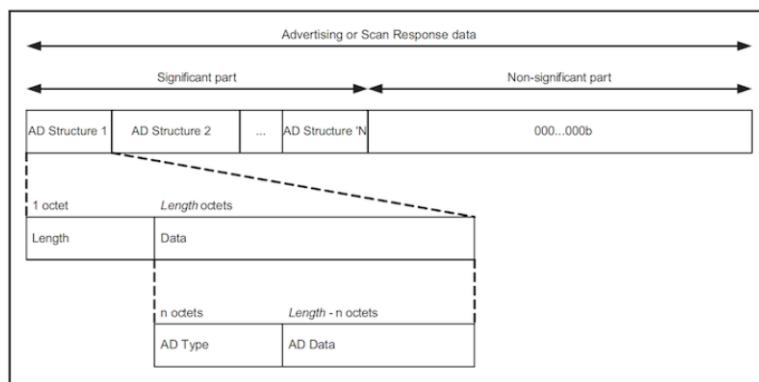
- GAP cung cấp cách để hai thiết bị BLE có thể giao tiếp với nhau bao gồm
  - Các chế độ và các vai trò của thiết bị BLE trong mạng
  - Các thông tin về quá trình advertising và scanning
  - Quá trình thiết lập kết nối
  - Bảo mật
- ATT quy định cách giao tiếp thông tin của hai thiết bị là theo mô hình server và client, cùng với đó định nghĩa cách để server gửi đi các thông tin tới client cũng như cấu trúc của các thông tin đó.
- GATT dựa trên ATT, định nghĩa ba nội dung quan trọng nhất trong BLE mà người lập trình sẽ tiếp xúc rất nhiều xuyên suốt quá trình thiết kế thiết bị BLE:
  - Services
  - Characteristics
  - Profiles

#### 2.4.3.1 GAP

GAP quy định cách để hai thiết bị có thể biết đến nhau, có được các thông tin cơ bản để quyết định xem có kết nối với nhau hay không. Quá trình sẽ bắt đầu khi một thiết bị phát ra các bản tin và thiết bị còn lại sẽ quát các bản tin đó. Thiết bị phát ra các bản tin sẽ được đặt tên là Peripheral và quá trình phát các bản tin được gọi là

advertising. Thiết bị quét các bản tin xung quanh có tên là Central và quá trình đó là quá trình scanning.

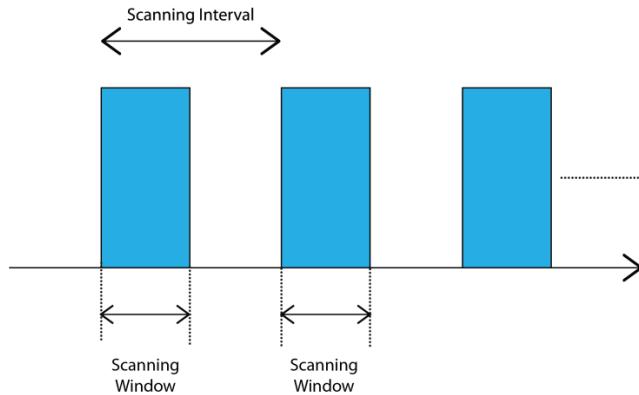
- Quá trình advertising sẽ có một số đặc trưng bao gồm:
  - Chu kì advertising: được quy định từ 20 mili giây đến 10.24 giây, với độ chia là 625 micro giây. Chu kì advertising cũng cần được lựa chọn kĩ càng, bởi nó sẽ ảnh hưởng trực tiếp đến tuổi thọ pin.
  - Bản tin advertising: Trong mỗi chu kì advertising, thiết bị sẽ gửi đi một bản tin với các thông tin phục vụ cho quá trình nhận biết giữa hai thiết bị. Cấu trúc bản tin được mô tả như ở hình 2.9. Dữ liệu được cấu trúc theo TLV (Type – Length - Value), tạm dịch là ba trường kiểu dữ liệu, độ dài và giá trị. Các kiểu dữ liệu phổ biến thường được thêm trong bản tin Advertising là:
    - Local name: chứa tên của thiết bị
    - Tx Power Level: mức năng lượng sử dụng cho việc truyền thông tin, đơn vị dBm
    - Flags: gồm các cờ biểu thị cho các chế độ của thiết bị
    - Service Solicitation: một danh sách của một hoặc nhiều mã UUID đặc trưng cho các service mà thiết bị hỗ trợ. Kiểu dữ liệu này giúp cho thiết bị Central biết được các service mà thiết bị Peripheral hỗ trợ trước khi thiết lập kết nối.
    - Appearance: định nghĩa chức năng của thiết bị ví dụ như điện thoại, cảm biến nhịp tim,... theo chuẩn của [Bluetooth SIG Assigned Numbers](#).



**Hình 2.9 Cấu trúc bản tin Advertising**

- Quá trình scanning định nghĩa các tham số sau:

- Scan Window: ứng với thời gian mà thiết bị trong trạng thái scanning
- Scan Interval: ứng với chu kỳ của thiết bị khi bắt đầu trạng thái scanning



**Hình 2.10 Quá trình scanning của thiết bị BLE**

- Kết nối hai thiết bị: Sau khi thiết bị Central nhận được thông tin từ Peripheral, hai bên sẽ bắt đầu trao đổi một số gói tin. Central sẽ gửi đi gói tin yêu cầu kết nối tới Peripheral. Sau khi Central nhận được gói tin phản hồi từ Peripheral, hai thiết bị sẽ vào trạng thái kết nối, khi đó Central được coi là master và Peripheral là slave. Master sẽ quản lý kết nối, các thông số kết nối và các sự kiện.
- Quá trình trao đổi dữ liệu trong kết nối: một trong những yếu tố giúp cho công nghệ BLE tiết kiệm năng lượng là ở cách hai thiết bị trao đổi dữ liệu. Quá trình trao đổi dựa trên một số tham số bao gồm:
  - Connection Event: là khoảng thời gian master và slave trao đổi dữ liệu cho tới khi cả hai bên không còn dữ liệu để gửi.
  - Connection Interval: được định nghĩa là khoảng thời gian giữa hai điểm bắt đầu của connection event, được giới hạn từ 7.5 mili giây đến 4 giây với độ chia 1.25 mili giây.

#### 2.4.3.2 Attribute Protocol (ATT)

ATT định nghĩa ra cấu trúc thông tin mà hai thiết bị cần trao đổi, để hệ thống hóa cũng như dễ dàng hơn trong việc lưu trữ, trao đổi thông tin. Hai thiết bị sử dụng ATT sẽ được gán hai vai trò:

- Server: là thiết bị chứa dữ liệu mà người dùng cần bao gồm cảm biến, vị trí, giá tốc, thời lượng pin .... Khi được yêu cầu thông tin, server sẽ gửi thông

tin đó đến các thiết bị yêu cầu ví dụ như các Gateway hoặc điện thoại. Ngoài ra, việc gửi thông tin còn có thể thông qua cách Server sẽ tự động gửi đi dữ liệu tới thiết bị đã đăng ký nhận thông báo khi dữ liệu thay đổi từ trước đó. Đó được định nghĩa là notification.

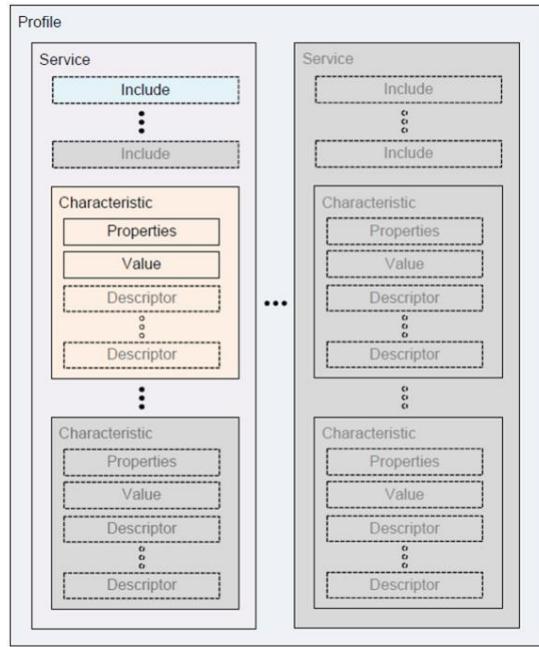
- Client: là thiết bị giao tiếp với server với mục đích đọc dữ liệu, cũng như điều khiển server khi cần thiết. Client cũng là thiết bị đăng ký nhận notification từ phía server từ trước để có thể nhận được dữ liệu gửi từ server sang khi chúng thay đổi.

Các attributes được định danh bởi các yếu tố sau:

- Attribute Type: Các attribute được định danh bởi các UUID (Universally Unique Identifier), có độ dài 16 bits (được quy định bởi Bluetooth SIG) hoặc 128 bits với các attribute do người dùng quy định.
- Attribute Handle: là một giá trị 16 bits mà server gán cho mỗi attribute, giống như một địa chỉ. Giá trị này được client sử dụng để tham chiếu các attributes.
- Attribute Permission: Quy định các hành động có thể làm với attribute đó, ví dụ như đọc, ghi hoặc notify.

#### 2.4.3.3 Generic Attribute Profile (GATT)

GATT là thuật ngữ sẽ gặp khá nhiều với các lập trình viên BLE, GATT cơ bản dựa trên ATT.



**Hình 2.11 Profiles, Services và Characteristics**

Hình 2.11 mô tả rõ nhất các cấu trúc dữ liệu của GATT gồm profile, service và characteristic. GATT dựa trên ATT, nên vẫn sẽ có hai vai trò là server và client. Tuy nhiên, một thiết bị có thể chạy song song hai vai trò server và client khi có thể đọc dữ liệu từ một server khác, sau đó gửi dữ liệu đó đi tới một client khác với vai trò một server.

- Service: bao gồm một hoặc nhiều attribute, một trong số đó có thể là characteristic để có thể đáp ứng các chức năng cụ thể của server. Ví dụ, service về pin của SIG bao gồm một characteristic là battery level. Ngoài ra, service còn chứa các attribute khác mà không phải là characteristic, phục vụ việc định nghĩa cũng như các quyền truy cập service.
- Characteristic: là một phần của một service, biểu diễn một phần thông tin mà server muốn chia sẻ tới các client. Characteristic chứa các attribute để định nghĩa các giá trị mà nó giữ:
  - Properties: được biểu diễn bởi một số các bit để định nghĩa các mà các giá trị characteristic có thể được sử dụng như đọc, ghi, ghi không phản hồi, thông báo,...
  - Descriptors: được sử dụng để chứa các thông tin liên quan đến giá trị của các characteristic như các trường sử dụng cho việc đăng ký nhận thông báo từ server, trường về định dạng và đơn vị của giá trị.

Sau khi đã hiểu rõ cách hai thiết bị BLE thiết lập kết nối, trao đổi dữ liệu và cấu trúc dữ liệu, việc lập trình sẽ dựa chủ yếu trên các hoạt động đọc ghi các service, characteristic và descriptor. Phần tiếp theo, đồ án đi vào thiết kế thiết bị gateway với hai phần chính là phần cứng và phần mềm.

## 2.5 Thiết kế Gateway

### 2.5.1 Yêu cầu hệ thống

Gateway là thiết bị quản lý trực tiếp với hệ thống định vị tại hiện trường, có chức năng là cầu nối, truyền tải dữ liệu mang thông tin vị trí, đồng thời cấu hình các thiết bị khi có yêu cầu từ cloud. Vai trò quản lý của Gateway hết sức quan trọng, luôn phải đảm bảo kiểm soát được trạng thái on/off của các node trong hệ thống và giữ được độ ổn định khi số lượng tag lớn. Hầu hết các chip BLE trên thị trường đều hỗ trợ giới hạn số lượng thiết bị kết nối BLE với chip, cụ thể là kết nối GATT tại cùng một thời điểm, nên việc xử lý ở phần mềm để có thể đảm bảo thông tin không bị trễ là một yêu cầu quan trọng.

Thông thường, Gateway sẽ sử dụng kết nối WiFi để đưa dữ liệu lên Cloud. Tuy nhiên với môi trường nhà máy, hệ thống mạng thường sử dụng chuẩn Ethernet. Vì thế, ngoài WiFi, Ethernet sẽ là một lựa chọn thứ hai để giao tiếp với Cloud của thiết bị Gateway. Các tiêu chí kỹ thuật cần có cho một thiết bị BLE Gateway được mô tả dưới đây:

- Khối nguồn: Do thiết bị Gateway là thiết bị cần chạy liên tục và ổn định, nên nguồn sẽ được lấy từ điện lưới 220VAC. Cùng với đó, do thiết bị có thể được lắp ở trong nhà, nhà máy, nên phải ưu tiên cho việc dễ lắp đặt. Vì thế, thiết bị sẽ được thiết kế phần nguồn trên mạch, thay vì sử dụng adapter AC – DC.
- Khôi lưu trữ: Mục đích chính của khôi lưu trữ là lưu lại các thông tin quan trọng của thiết bị cũng như hỗ trợ vào quá trình cập nhật firmware cho thiết bị. Khi thiết bị đã vào giai đoạn sản xuất và đến tay người dùng, sẽ gặp phải các lỗi mà người thiết kế không thể tính toán trước. Khi đó, người thiết kế cần cập nhật các phiên bản phần mềm mới có thể xử lý được các lỗi đó. Trên thiết bị nên được lưu trữ cả hai bản firmware cùng một lúc để phòng trừ trường hợp bản firmware mới bị lỗi. Khôi lưu trữ có thể sử dụng bộ nhớ

Flash với giá thành hợp lí và hỗ trợ các thư viện, tài liệu giao tiếp cũng như tốc độ đọc ghi rất cao.

- Khối Ethernet: Khối Ethernet cần đảm bảo hỗ trợ tốc độ ở tối thiểu 10Mbps, tốc độ tối đa có thể đạt tới 100Mbps. Thiết bị cần được lập trình để chuyển đổi giữa WiFi và Ethernet mà không ảnh hưởng quá nhiều quá trình hoạt động của thiết bị.
- Khối hiển thị, thông báo: mục đích chính của khối là để giúp người dùng dễ nhận biết các trạng thái của hệ thống như kết nối mạng thành công hay đang kết nối tới Cloud vì theo mô hình mạng, gateway được ưu tiên đặt ở vị trí cao nhất để tránh các vật cản. Vậy nên việc sử dụng các khối hiển thị như LED và khối thông báo sử dụng còi chíp sẽ giúp cải thiện sự tương tác với người dùng.
- Khối xử lý trung tâm: Sau khói nguồn, khói xử lý trung tâm là một trong những phần quan trọng nhất của thiết bị. Tất cả các thuật toán quản lý, truyền thông BLE, WiFi sẽ được khói xử lý trung tâm đảm nhận. Hiện nay trên thị trường đã có rất nhiều chip được tích hợp các chức năng trên, gọi là SoC. Các SoC có kích thước nhỏ gọn, có đầy đủ chức năng truyền thông và bộ xử lý mạnh, phù hợp cho các ứng dụng Gateway. Khối xử lý trung tâm cần được lựa chọn dựa trên những yêu cầu sau:
  - Lõi của MCU, xung nhịp lõi của SoC
  - Độ phổ biến của SoC
  - Các công cụ hỗ trợ phát triển (từ hãng và cộng đồng), khả năng hỗ trợ nếu phát sinh lỗi trong quá trình lập trình
  - Số lượng ngoại vi phải phù hợp với yêu cầu của bài toán
  - Giá cả hợp lý để tối ưu giá thành sản phẩm
  - Độ ổn định khi hoạt động lâu dài

Căn cứ vào các yêu cầu đặt ra của khói xử lý trung tâm và các khói ngoại vi, SoC cần hỗ trợ thêm các ngoại vi sau:

- Khối SPI: hai khói SPI hỗ trợ giao tiếp với Ethernet và Flash, đảm bảo tốc độ từ 40 – 80 MHz
- Khối GPIO: hỗ trợ từ 10 GPIO cho việc điều khiển khói hiển thị, thông báo và phục vụ cho việc nạp firmware, debug
- Khối UART: hỗ trợ việc debug trong quá trình phát triển

### **2.5.2 Tổng quan hệ thống**

Dựa vào yêu cầu của hệ thống, kết hợp với kinh nghiệm và giá, độ có sẵn của linh kiện, đồ án đã lựa chọn các phần tử chính của các khối như sau:

**Bảng 2.1 Các phần tử linh kiện chính**

	IC/SoC
Khối xử lý trung tâm	SoC ESP32 Wroom 32
Khối nguồn	Nguồn Flyback 5V 1A
Khối Ethernet	IC W5500
Khối hiển thị	LED RGB WS2812B
Khối thông báo	Còi chíp PS1240P02BT
Khối lưu trữ	IC Flash 16MB W25Q128JVSIQ

Các phần tiếp theo, đồ án sẽ đi vào phân tích lí do lựa chọn các phần tử linh kiện trên và kèm theo các chức năng của linh kiện đó trong mỗi khối tương ứng.

### **2.5.3 Thiết kế khối nguồn**

Đây là khối cung cấp năng lượng cho toàn bộ các ngoại vi của thiết bị. Nó được coi như trái tim của thiết bị, quyết định đến sự hoạt động và lâu dài của hệ thống nên cần được tính toán, chọn lựa cẩn thận, tính toán cụ thể để có thể đề phòng được các trường hợp quá tải. Bảng dưới đây là dòng hoạt động cụ thể của từng khối dựa trên các linh kiện đã chọn ở phần 2.5.2. Mức tiêu thụ được liệt kê ở đây là mức tiêu thụ tối đa của các khối và được tham khảo ở tài liệu của hãng sản xuất.

**Bảng 2.2 Mức tiêu thụ tối đa của hệ thống**

STT	Linh kiện chính của các khối	Tác vụ tiêu thụ	Điện áp sử dụng (V)	Mức tiêu thụ tối đa (mA)
1	ESP32-Wroom-32	Tắt cả các ngoại vi: radio (WiFi và BLE), UART, I2C,	3.3	500[td]
2	W5500	Truyền nhận dữ liệu, tốc độ tối đa 100Mbit/s	3.3	132[td]
3	WS2812B LED RGB	Thông báo các trạng thái tương ứng với các màu	5	50
4	Còi chíp PS1240P02BT	Thông báo các trạng thái tương ứng với tiếng còi	5	30
5	Flash 16MB W25Q128JVSIQ	Đọc ghi bộ nhớ	3.3	25
Tổng				≈ 700

Tính toán ở bảng 2.2 giúp cho việc thiết kế phần nguồn trở nên rõ ràng hơn, có thể dự đoán trước được các kịch bản xảy ra, ví dụ các thời điểm mức tiêu thụ đạt tối đa, để có thể giữ được độ ổn định của hệ thống. Khối tiêu thụ năng lượng nhiều nhất là khối xử lý trung tâm với năng lượng cần cấp là khoảng 500mA. Mức tiêu thụ lớn như vậy chỉ xảy ra ở thời điểm thiết bị kết nối tới WiFi và xảy ra ở thời điểm rất ngắn, còn năng lượng tiêu thụ thường xuyên sẽ ở mức 300mA với tác vụ chính là truyền nhận dữ liệu qua WiFi và BLE, cụ thể hơn là qua sóng RF 2.4GHz. Các mức tiêu thụ được liệt kê ở bảng 2.3.

**Bảng 2.3 Năng lượng tiêu thụ khói xử lý trung tâm**

Chế độ	Dòng tiêu thụ (mA)
Truyền dữ liệu chuẩn 802.11b, POUT = +19.5dBm	240
Truyền dữ liệu chuẩn 802.11g, POUT = +16dBm	190
Truyền dữ liệu chuẩn 802.11n, POUT = +14dBm	180
Nhận dữ liệu chuẩn 802.11b/g/n	100
Truyền dữ liệu BT/BLE	130
Nhận dữ liệu BT/BLE	100

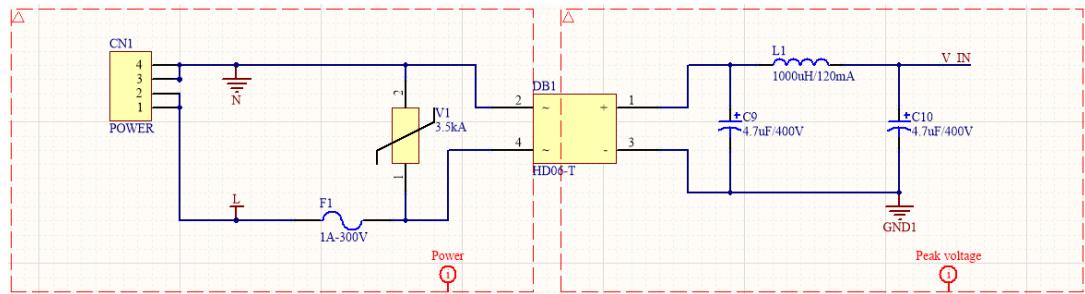
Với năng lượng tiêu thụ của toàn mạch ở khoảng 400 mA – 700mA, khói nguồn chuyển đổi AC – DC cần được thiết kế để đảm bảo công suất tối đa có thể đạt được là 1A ở mức 5V, tương đương với 5W. Khói nguồn cần chuyển đổi xuống 5V bởi có một số ngoại vi cần sử dụng mức điện áp 5V như đã liệt kê ở bảng 2.2. Tiếp theo đó, để có thể cung cấp năng lượng cho các khói sử dụng điện áp 3.3V, em sử dụng IC hạ áp thuộc dòng LDO, tiêu thụ năng lượng thấp và có khả năng cung cấp dòng điện tối đa ở mức 500mA.

Hiện tại đã có nhiều loại mạch chuyển đổi AC – DC trên thị trường, chủ yếu theo các nguyên lý: buck-boost và flyback, trong đó mạch flyback là mạch có độ hiệu quả cũng như chất lượng của năng lượng đầu ra rất tốt nếu được thiết kế đúng nguyên lý. Điểm mạnh hơn của mạch flyback so với buck-boost là ở cuộn biến áp. Nhờ vào cuộn biến áp, điện áp đỉnh đầu vào của mạch nguồn flyback có thể cao hơn mà không phải đòi hỏi tăng thêm quá nhiều ở phần tạo xung như ở mạch buck-boost.

Mạch nguyên lý flyback đồ án sử dụng có phần tử chính phụ trách công việc tạo xung (tương tự với phần tử FET trong mạch buck-boost) là IC TEA1721AT.

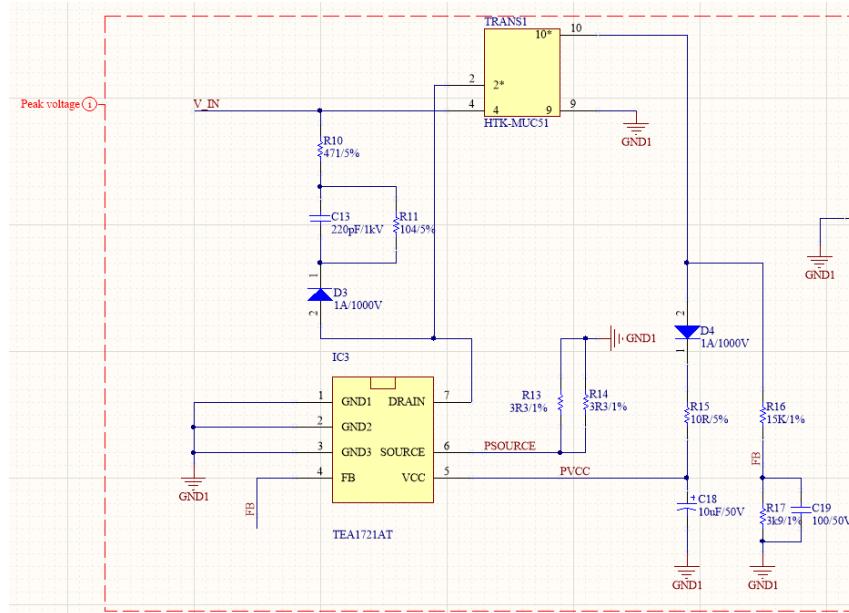


**Hình 2.12 IC TEA1721AT**



**Hình 2.13 Khối chuyển đổi điện áp AC sang DC**

Điện áp 220VAC đầu vào được chỉnh lưu qua cầu diode DB1, tạo ra điện áp DC khoảng 200V. Trước khi điện áp vào cầu chỉnh lưu sẽ đi qua một cầu chì giới hạn 300V 1A để bảo vệ toàn mạch, cùng với một phần tử quan trọng khác là tụ chống sét (varistor) để bảo vệ mạch khỏi những xung điện áp cao, xung gai. Phần tử được lựa chọn là V10E300P với điện áp đỉnh bảo vệ tối đa là 625V. Điện áp sau cầu chỉnh lưu sẽ có dạng sóng răng cưa với độ sụt lớn. Nếu điện áp đầu vào cho khói hạ áp phía sau được làm phẳng, chất lượng điện áp 5V đầu ra sẽ rất tốt. Cho nên, hai tụ hóa 4.7uF và cuộn cảm 1000H được thêm vào để có thể làm phẳng sóng răng cưa sau cầu chỉnh lưu.

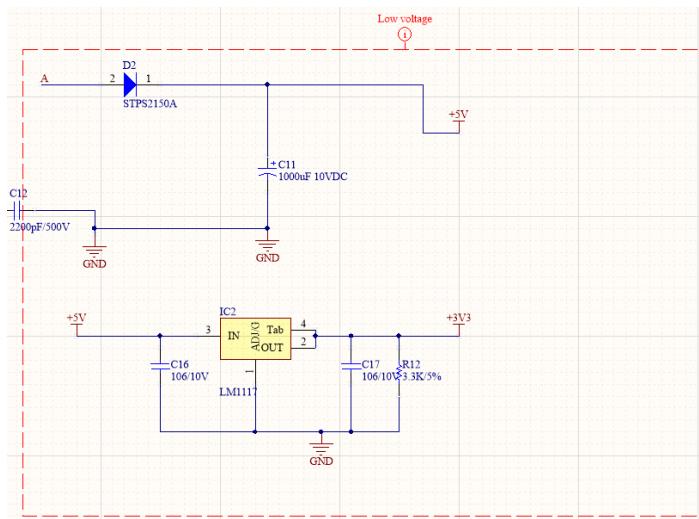


**Hình 2.14 Khối chuyển điện áp cao sang 5V**

Khối hạ áp chuyển đổi điện áp cao xuống điện áp 5V là phần mang đặc điểm chính của nguồn flyback với các phần tử chính là một cuộn biến áp và một IC FET.

Vai trò hoạt động của TEA1721AT là phần tử điều khiển của khối nguồn, có nhiệm vụ điều khiển cân bằng mức điện áp đầu ra ở 5V. Với điện áp ở chân FB được

lấy từ cuộn thứ cấp của biến áp xung, IC có thể tính toán và đưa ra điều chỉnh hợp lí ở đầu vào biến áp xung. Chế độ hoạt động chủ yếu của TEA1721AT là chế độ điện áp cố định. Chế độ bắt đầu khi IC được cấp một điện áp khởi động ở khoảng 17V. Sau đó tần số đóng cắt của IC sẽ bắt đầu ở 22.5kHz, và khi công suất ở phía đầu ra tăng lên, tần số đóng cắt của IC cũng sẽ tăng theo. Khi tần số đóng cắt đạt đến giá trị tối đa là 50.5 kHz, tương đương với công suất phát tối đa, IC sẽ ngừng hoạt động.



**Hình 2.15 Khối hạ áp 3.3V**

Sau khối điện áp 5V, mạch sử dụng IC ổn áp LDO LM1117 với dòng điện đầu ra tối đa là 1A và sụt áp 700mV. Theo tính toán ở bảng 2.2, dòng điện tối đa mà khối 3.3V sử dụng là 600- 700mA. Như vậy IC LM1117 có thể hoàn toàn đáp ứng được công suất đó.

#### 2.5.4 Thiết kế khói xử lý trung tâm

Khối xử lý trung tâm sử dụng module ESP32 Wroom-32 của nhà sản xuất Espressif với các thông số ở bảng 2.4 :



**Hình 2.16 ESP32 Wroom-32**

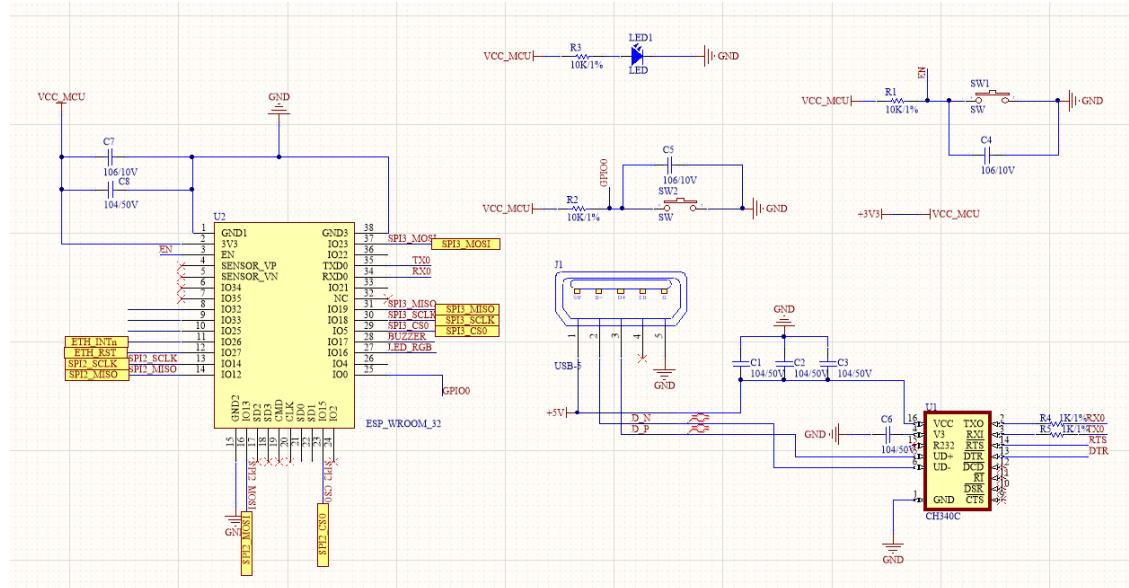


**Hình 2.17 ESP32 Wroom-32 No Shield**

**Bảng 2.4 Thông số Module ESP32 Wroom-32**

Chip xử lý (SoC)	ESP32-D0WDQ6 với 2 lõi Xtensa 32-bit LX6
Điện áp hoạt động	3.0V – 3.6V
Dòng điện hoạt động trung bình	80mA
Dòng điện tối thiểu cần đảm bảo từ nguồn cấp	500mA
Nhiệt độ hoạt động	-40°C ~ +85°C
Kích thước	18 mm x 25.5 mm x 3.1 mm
Bộ nhớ Flash	4MB
Bộ nhớ SRAM	520KB
Tần số thạch anh	40MHz
Ngoại vi	UART, 3 SPI, I2C, LED PWM, I2S, IR, GPIO, ADC, DAC, TWAI
WiFi	802.11 b/g/n (tốc độ tối đa 150 Mbps) Tần số: 2412 ~ 2484 MHz
Bluetooth	Bluetooth v4.2 BR/EDR và BLE

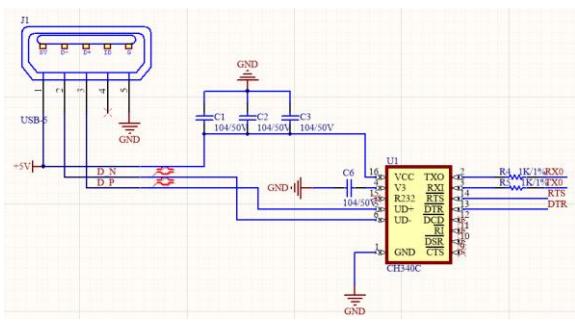
Ngoài ra, các ngoại vi của EPS32 được hỗ trợ mapping tới các chân bắt kì, giúp cho việc thiết kế trở nên dễ dàng hơn thay vì các ngoại vi cố định với các IO như ở các vi điều khiển khác. Sau khi đã nắm rõ được các thông số và ngoại vi của module, đồ án đi vào thiết kế schematic, được trình bày ở hình 2.18 và bảng 2.5.



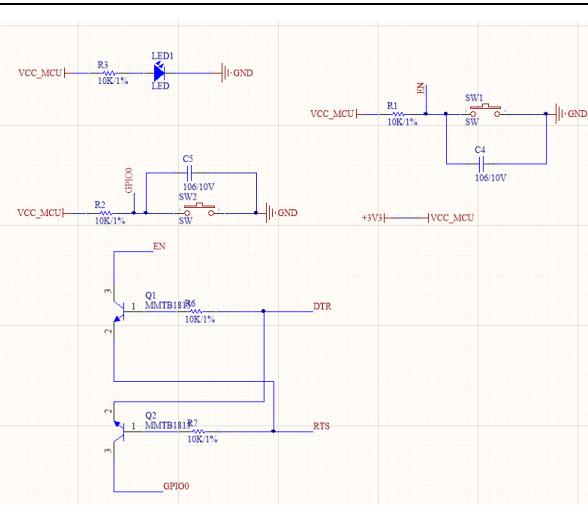
Hình 2.18 Schematic khối xử lý trung tâm

Bảng 2.5 Giải thích thành phần khối xử lý trung tâm

	<p>Module ESP32 Wroom-32 được cấp nguồn 3V3. Để đảm bảo nguồn cấp ổn định, các tụ bypass được thêm vào gần chân nguồn của module để lọc đi các nhiễu. Theo hướng dẫn của nhà sản xuất, 1 tụ 100nF/50V(104/50V) và 1 tụ 10uF/10V (106/10V) là đủ để đảm bảo độ ổn định của nguồn cấp.</p>
--	--



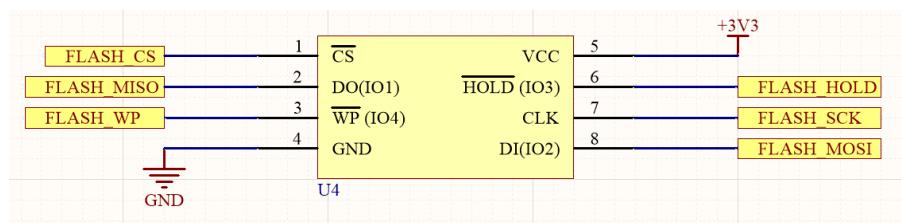
Để phục vụ quá trình nạp firmware và debug trong nghiên cứu và phát triển, một khối chuyển đổi UART sang USB và một cổng MicroUSB được sử dụng. Chip chuyển đổi UART to USB là CH340C, sử dụng nguồn 5V từ cổng USB và không sử dụng nguồn 5V từ mạch nên khi thiết bị sử dụng nguồn điện lưới, khói này sẽ không tiêu thụ năng lượng.



Đây là khói phụ trợ cho khói nạp firmware, có nhiệm vụ chính là đưa module ESP32 vào chế độ nạp firmware bằng cách lần lượt kéo chân EN và GPIO0 xuống GND. Hai nút bấm để reset module và để kéo GPIO0 xuống GND trong trường hợp khói nạp firmware hoạt động không đúng mong đợi.

### 2.5.5 Khối lưu trữ dữ liệu

Khối lưu trữ dữ liệu sử dụng IC Flash 16MB W25Q128FV giao tiếp qua bus SPI với tốc độ hỗ trợ tối đa của ESP32 là 80MHz khi sử dụng các chân SPI được nhà sản xuất khuyên dùng. Bus SPI của ESP32 hoàn toàn có thể sử dụng các chân khác, dựa trên ma trận GPIO mà nhà sản xuất hỗ trợ, tuy nhiên, tốc độ tối đa của bus khi đó sẽ giảm còn 40MHz, làm giảm đi tốc độ đọc ghi bộ nhớ.



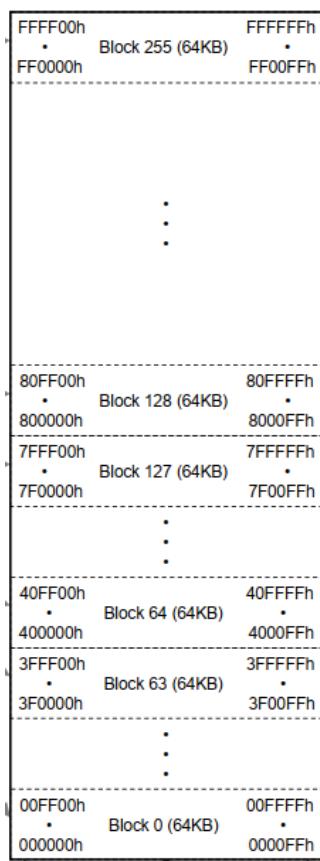
Hình 2.19 Khối lưu trữ dữ liệu

Đơn vị dữ liệu của Flash là byte. Ngoài ra, Flash còn hỗ trợ thêm các định dạng khói dữ liệu khác để giúp cho việc đọc ghi dữ liệu được hiệu quả hơn. Các định dạng

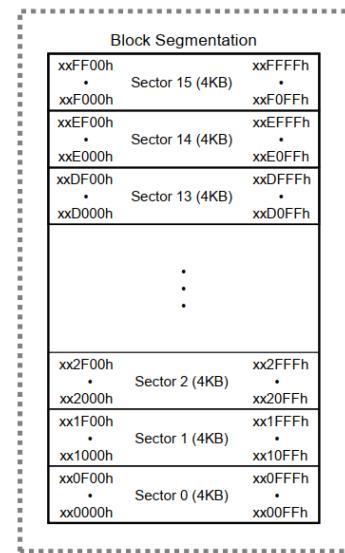
đó bao gồm page – 256 bytes, sector – 4 KB, block – 64K bytes. Khi đó, 1 sector sẽ có 16 page và 1 block sẽ gồm 16 sector.



**Hình 2.20 IC Flash W25Q128JV**

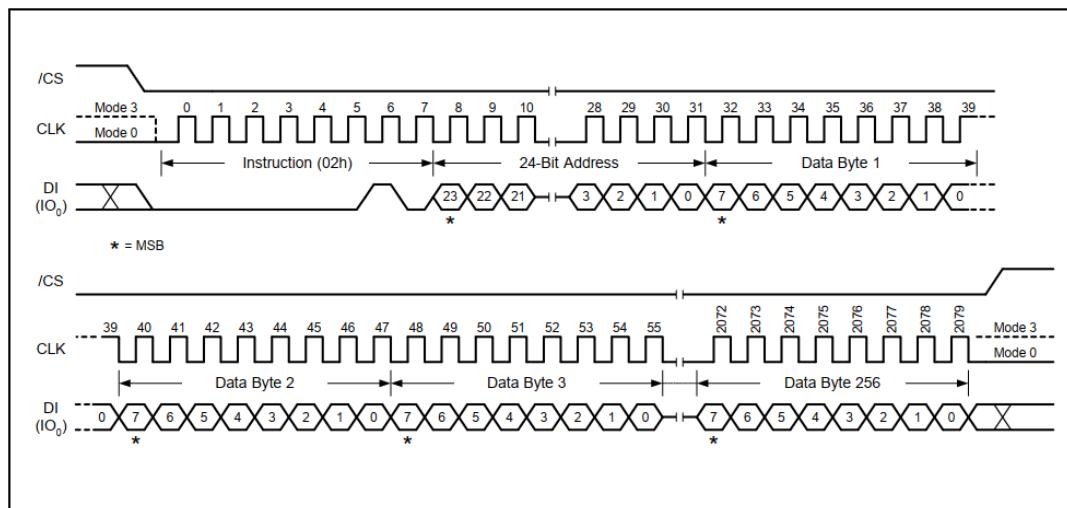


**Hình 2.21 Cấu trúc bộ nhớ Flash theo Block**



**Hình 2.22 Cấu trúc một block**

Để có thể đọc ghi dữ liệu, IC hỗ trợ tập lệnh và các hướng dẫn để tạo ra các khung truyền SPI như ở ví dụ ở hình 2.23 dưới đây



Hình 2.23 Ghi dữ liệu vào một page trong flash

Ví dụ ở đây với lệnh ghi dữ liệu vào một page trong flash. Khi bắt đầu quá trình, chân CS mức tích cực thấp cần được kéo xuống mức 0. Các byte dữ liệu lần lượt được truyền theo từng bit, dựa trên từng clock vào chân DI của flash. Byte đầu tiên là byte mã lệnh 02h, để báo cho flash biết rằng ngoại vi đang chuẩn bị ghi dữ liệu vào một page trong flash. Sau đó là 3 byte địa chỉ của page và ngay sau là 256 byte dữ liệu được gửi đến. Quá trình ghi dữ liệu được kết thúc khi chân CS được đưa lên mức 1. Flash còn hỗ trợ nhiều lệnh khác như đọc, ghi, xóa Sector, Block, xóa toàn bộ chip,...

### 2.5.6 Thiết kế khối Ethernet

Khối Ethernet sử dụng phần tử xử lý chính là IC W5500 của Wiznet, là một trong những IC Ethernet phổ biến được sử dụng trong các ứng dụng IoT. Các thông tin cơ bản của W5500 được mô tả ở bảng 2.6.

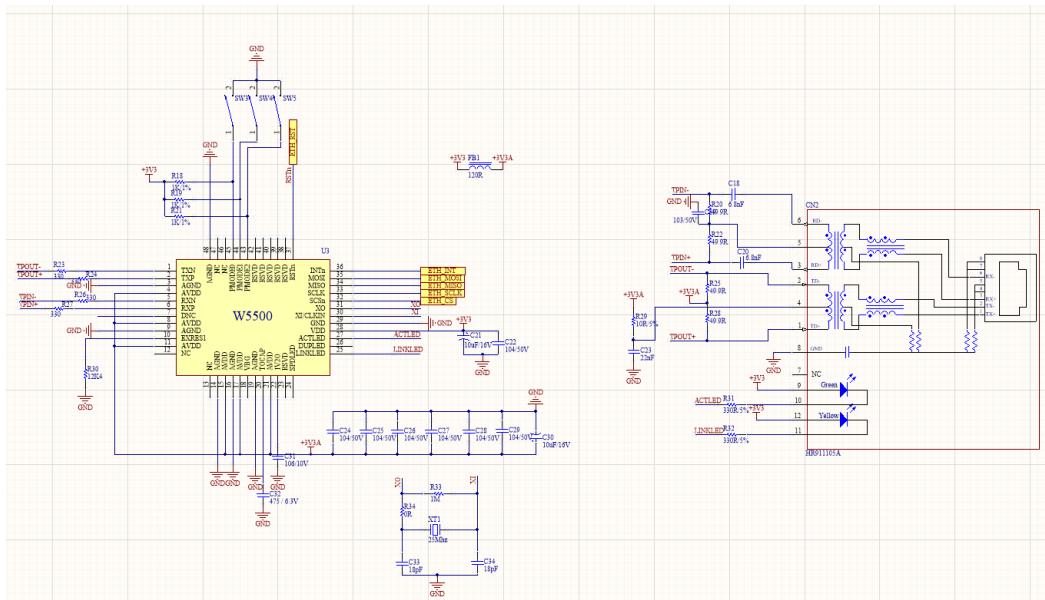


**Hình 2.24 IC W5500 Ethernet**

Bảng 2.6 Thông số IC W5500

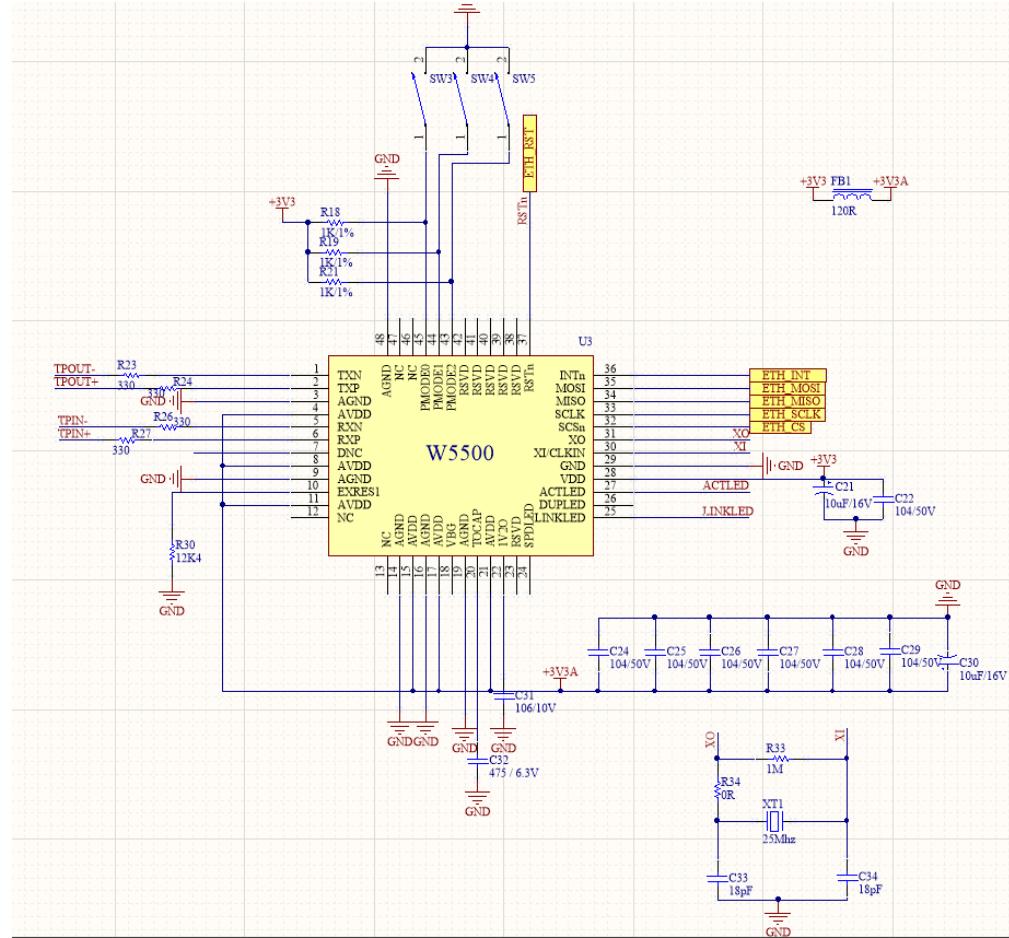
Thông số	Mô tả
Điện áp hoạt động	2.07 – 3.63V
Giao thức giao tiếp	SPI
Tốc độ giao tiếp SPI	33.3 – 80MHz
Tốc độ giao tiếp Ethernet tối đa	100Mbs

Schematic của khối Ethernet cần tuân theo các yêu cầu kỹ thuật của hãng để đảm bảo tín hiệu tránh được các nhiễu do tín hiệu Ethernet là tín hiệu có tốc độ cao, rất dễ bị ảnh hưởng bởi nhiễu.



## Hình 2.25 Schematic khối Ethernet

Khối Ethernet được chia làm hai phần chính, phần 1 gồm W5500 và các phần tử đi kèm, phần 2 là cổng RJ45 HR911105A.



Hình 2.26 Schematic IC W5500

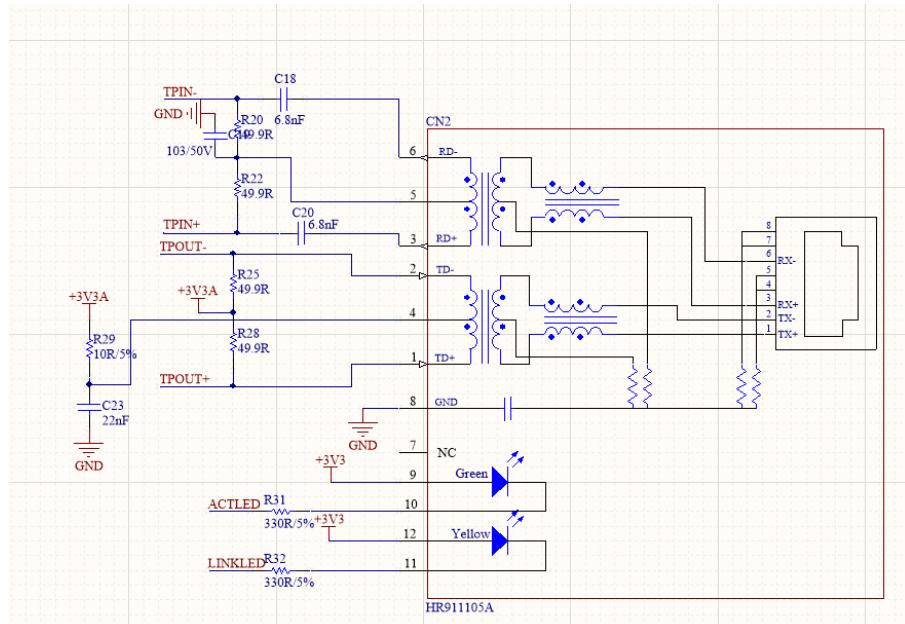
Các chân cấp nguồn cho IC cũng như các chân cấp nguồn cho khối analog trong IC cần được đảm bảo nguồn ổn định, nên các tụ bypass sẽ được thêm vào gần các chân analog để lọc đi các sóng nhiễu. Trước đó, nguồn 3.3V cấp cho IC được lọc qua một cuộn ferrite bead có tác dụng lọc đi các nhiễu cao tần.

Các chân TXN, TXP và RXN, RXP là các chân tín hiệu tốc độ cao, cho nên điện trở có giá trị khoảng  $33\Omega$  sẽ được thêm vào để lọc đi các nhiễu EMI theo tính toán của nhà sản xuất. IC W5500 còn cho phép người dùng cấu hình lựa chọn tốc độ và chế độ sử dụng bằng 3 bit tương ứng với 3 chân PMODE0, PMODE1, PMODE2. Các chế độ tương ứng với 3 bit được mô tả ở hình 2.27. Nếu không cấu hình thêm gì, với schematic như trên, W5500 sẽ ở chế độ auto-negotiation enabled.

PMODE [2:0]			Description
2	1	0	
0	0	0	10BT Half-duplex, Auto-negotiation disabled
0	0	1	10BT Full-duplex, Auto-negotiation disabled
0	1	0	100BT Half-duplex, Auto-negotiation disabled
0	1	1	100BT Full-duplex, Auto-negotiation disabled
1	0	0	100BT Half-duplex, Auto-negotiation enabled
1	0	1	Not used
1	1	0	Not used
1	1	1	All capable, Auto-negotiation enabled

Hình 2.27 Các chế độ của IC W5500

Hình 2.27 mô tả cổng RJ45 cùng các phần tử đi kèm gồm các điện trở  $49.9 \Omega$  để phối hợp trở kháng theo tính chất của các dây cáp ethernet hiện tại, các tụ  $6.8nF$  để có thể lọc đi các tín hiệu xoay chiều do cổng RJ45 đè án sử dụng có tích hợp biến áp bên trong, nên các sóng nhiễu xoay chiều là không tránh khỏi.



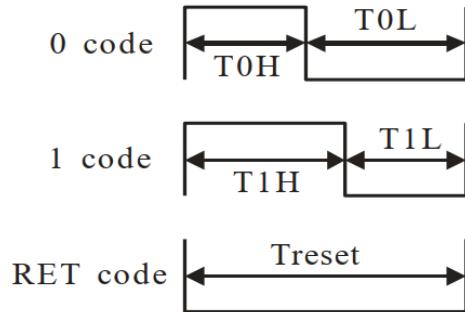
Hình 2.28 Schematic cổng RJ45 HR911105A

### 2.5.7 Thiết kế khối hiển thị, thông báo

Khối hiển thị sử dụng 3 LED RGB WS2812B, điện áp 5V. Điều đặc biệt với WS2812B là các LED có thể nối tiếp các chân dữ liệu với nhau, cho nên, chỉ cần một chân dữ liệu duy nhất để có thể đưa dữ liệu ra tất cả các LED. Ngoài ra, do không sử dụng chân clock, nên các bit 0, 1 của dữ liệu sẽ được định nghĩa tương ứng với các khoảng thời gian khác nhau được định nghĩa ở hình 2.29 và 2.30.

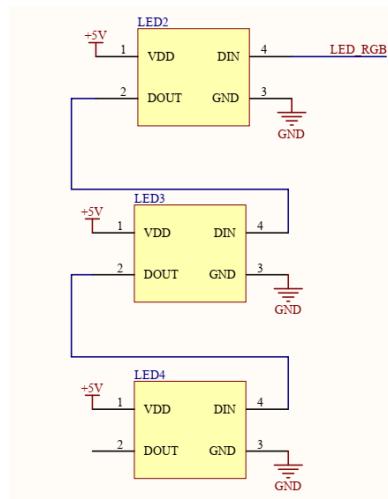
T0H	0 code ,high voltage time	0.4us	$\pm 150\text{ns}$
T1H	1 code ,high voltage time	0.8us	$\pm 150\text{ns}$
T0L	0 code , low voltage time	0.85us	$\pm 150\text{ns}$
T1L	1 code ,low voltage time	0.45us	$\pm 150\text{ns}$
RES	low voltage time	Above 50 $\mu\text{s}$	

**Hình 2.29 Thời gian định nghĩa cho bit 0, 1**



**Hình 2.30 Hình dạng xung định nghĩa cho bit 0, 1**

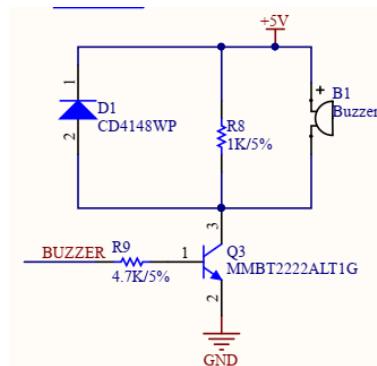
Dữ liệu truyền đi sẽ gồm 24 bit trong đó mỗi màu R, G, B là 8 bit. Như vậy để cấp dữ liệu đủ cho 3 LED ở hình 2.31, sẽ phải cần truyền 72 bit và giữa mỗi 24 bit sẽ là khoảng thời gian cho reset code, vào khoảng hơn 50 micro giây như định nghĩa ở hình 2.29.



**Hình 2.31 Schematic khối LED RGB**

Với khói thông báo, mạch sử dụng còi chíp PS1240P02BT, là loại còi sử dụng hiệu ứng áp điện, cho nên cần cấp xung ở một tần số nhất định để còi có thể hoạt động. Với loại còi này, xung cần cấp là 4kHz, với duty cycle là 50%. Nếu không đúng các

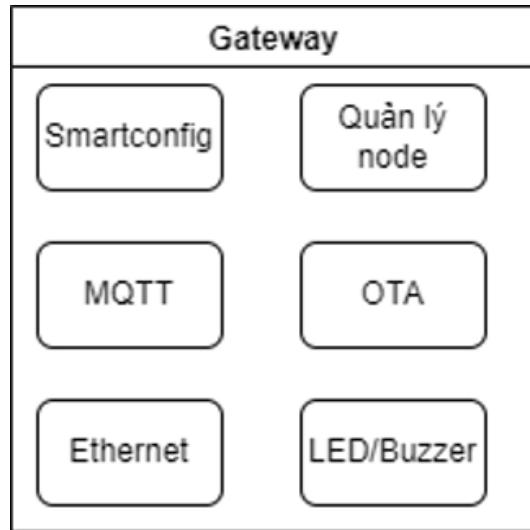
thông số như trên, còi sẽ không hoạt động. Phần mạch lực được thiết kế theo hướng dẫn của nhà sản xuất, với phần diode và điện trở mục đích chính là nạp, xả cho còi.



**Hình 2.32 Schematic khối thông báo**

## 2.6 Thiết kế Firmware

Module ESP32 Wroom-32 sử dụng framework ESP-IDF được nhà sản xuất Espressif phát triển và có cộng đồng hỗ trợ rất mạnh. Ngôn ngữ sử dụng để lập trình là ngôn ngữ C, phù hợp với các kiến trúc vi điều khiển.Thêm vào đó, framework của nhà sản xuất sử dụng hệ điều hành RTOS là FreeRTOS, giúp cho việc quản lý chương trình dễ dàng hơn và tăng tốc độ xử lý của chương trình khi phải xử lý nhiều tính năng phức tạp được mô tả ở hình 2.33. Chương trình khi bắt đầu sẽ đọc RESET\_REASON, được lưu trong bộ nhớ khi thiết bị reset. Nếu thiết bị là thiết bị mới, lý do reset của thiết bị sẽ là để cấu hình, được mô tả ở module Smartconfig. Hoặc khi thiết bị nhận ra có yêu cầu sử dụng Ethernet, thiết bị cũng sẽ reset là lý do sẽ là cho kết nối Ethernet. Ngoài ra còn nhiều RESET\_REASON khác và với mỗi lý do, chương trình khi khởi tạo sẽ thực hiện các công việc khác nhau, tuy nhiên vẫn sẽ hướng đến một điểm cuối là module quản lý node, và cũng là chức năng chính của thiết bị.



**Hình 2.33 Các tính năng chính của Gateway**

Bộ nhớ chương trình 4MB được chia thành các phân vùng khác nhau với các mục đích khác nhau được mô tả ở bảng 2.7

**Bảng 2.7 Các phân vùng bộ nhớ của thiết bị**

Tên phân vùng	Kiểu phân vùng	Địa chỉ	Mô tả
NVS	Data	0x9000 – 0xD000	Nonvolatile Storage – Vùng nhớ lưu trữ dữ liệu quản lý mạng
otadata	Data	0xD000-0xF000	Phân vùng lưu trữ dữ liệu OTA
phy_init	Data	0xF000 – 0x10000	Phân vùng lưu trữ dữ liệu tầng vật lý, thuộc về SDK
program	App	0x1000 – 0x2CC000	Phân vùng lưu trữ firmware chính của thiết bị
log_data	Data	0x2CC000 – 0x3CA000	Lưu trữ dữ liệu trạng thái của thiết bị trong lúc

			hoạt động
user_param	Data	0x3CA000 – 0x3CC000	Lưu trữ dữ liệu private key cho việc tạo JWT, rootCA cho bảo mật TLS

Khi được chia phân vùng và có chức năng, mục đích của từng phân vùng như trên giúp cho việc quản lý chương trình, lưu trữ các dữ liệu trạng thái trở nên dễ dàng. Ví dụ với phân vùng log\_data, khi thiết bị hoạt động sẽ được quy định rất nhiều trạng thái, như reset, cập nhật firmware, kết nối thành công tới cloud, kết nối thất bại,... Mục đích chính là để người phát triển có thể theo dõi được trạng thái của thiết bị từ xa khi thiết bị đã đến tay người dùng, giúp cho việc theo dõi, bảo trì từ xa dễ dàng hơn.

### 2.6.1 Quản lý Node

Quản lý node là tính năng quan trọng nhất của thiết bị Gateway. Để có thể thu thập được các thông tin cho quá trình quản lý, thiết bị sử dụng giao tiếp qua các Service của GATT, trong đó thiết bị đóng vai trò là GATT Client và các node là GATT Server. Các node DWM1001 sử dụng một vendor service (service tự thiết kế của nhà sản xuất) và được cung cấp thông tin đầy đủ phục vụ mục đích giao tiếp, phát triển của người dùng.

Thông tin được cung cấp bao gồm các characteristic mang thông tin về node như chế độ hoạt động, thông tin về vị trí, thông tin ID mạng,... được mô tả ở bảng

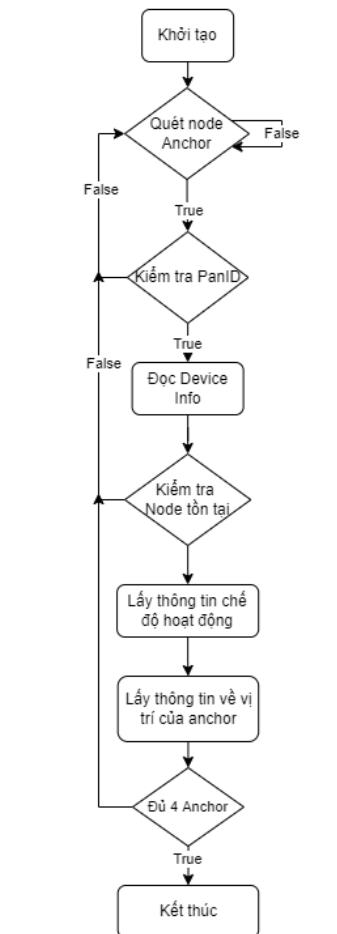
**Bảng 2.8 Các Characteristic sử dụng trong giao tiếp BLE với DWM1001**

STT	Characteristic	Kích thước	Mô tả
1	Operation Mode	2 Bytes	Các thông tin cơ bản về chế độ hoạt động của thiết bị

			<table border="1"> <thead> <tr><th colspan="2">1st byte (bit 7 down to 0)</th></tr> <tr><th>Bit</th><th>value</th></tr> </thead> <tbody> <tr><td>7</td><td>tag (0), anchor (1)</td></tr> <tr><td>6 - 5</td><td>UWB - off (0), passive (1), active (2)</td></tr> <tr><td>4</td><td>firmware 1 (0), firmware 2 (1)</td></tr> <tr><td>3</td><td>accelerometer enable (0, 1)</td></tr> <tr><td>2</td><td>LED indication enabled (0, 1)</td></tr> <tr><td>1</td><td>firmware update enable (0, 1)</td></tr> <tr><td>0</td><td>reserved</td></tr> <tr><th colspan="2">2nd byte (bit 7 down to 0)</th></tr> <tr><th>Bit</th><th>value</th></tr> <tr><td>7</td><td>initiator enable, anchor specific (0, 1)</td></tr> <tr><td>6</td><td>low power mode enable, tag specific (0, 1)</td></tr> <tr><td>5</td><td>location engine enable, tag specific (0, 1)</td></tr> <tr><td>4 - 0</td><td>reserved</td></tr> </tbody> </table>	1st byte (bit 7 down to 0)		Bit	value	7	tag (0), anchor (1)	6 - 5	UWB - off (0), passive (1), active (2)	4	firmware 1 (0), firmware 2 (1)	3	accelerometer enable (0, 1)	2	LED indication enabled (0, 1)	1	firmware update enable (0, 1)	0	reserved	2nd byte (bit 7 down to 0)		Bit	value	7	initiator enable, anchor specific (0, 1)	6	low power mode enable, tag specific (0, 1)	5	location engine enable, tag specific (0, 1)	4 - 0	reserved
1st byte (bit 7 down to 0)																																	
Bit	value																																
7	tag (0), anchor (1)																																
6 - 5	UWB - off (0), passive (1), active (2)																																
4	firmware 1 (0), firmware 2 (1)																																
3	accelerometer enable (0, 1)																																
2	LED indication enabled (0, 1)																																
1	firmware update enable (0, 1)																																
0	reserved																																
2nd byte (bit 7 down to 0)																																	
Bit	value																																
7	initiator enable, anchor specific (0, 1)																																
6	low power mode enable, tag specific (0, 1)																																
5	location engine enable, tag specific (0, 1)																																
4 - 0	reserved																																
2	Network ID (PAN ID)	2 Bytes	Mã định danh của mạng. Các thiết bị trong một mạng phải có cùng một PAN ID.																														
3	Location Data Mode	1 Bytes	<p>0 – Position Only: là chế độ của tag chỉ gửi vị trí gồm 12 bytes tọa độ x, y, z và 1 byte liên quan đến chất lượng vị trí.</p> <p>1 – Distances: thông tin nhận được sẽ là khoảng cách từ tag đến tối đa là 15 anchor.</p> <p>2 – Position and Distances: thông tin bao gồm vị trí và khoảng cách từ tag đến 4 anchor</p>																														
4	Location Data	106 Bytes Max	Thông tin vị trí của tag. Kích thước được định theo chế độ được chọn ở mục 3.																														
5	Device Info	29 Bytes	<p>Thông tin bao gồm:</p> <ul style="list-style-type: none"> <li>- Node ID (8 Bytes)</li> <li>- Hardware Version (4 Bytes)</li> <li>- Firmware 1 Version (4 Bytes): phiên bản Firmware 1 của DWM1001</li> <li>- Firmware 2 Version (4 Bytes): phiên bản Firmware 2 của DWM1001</li> <li>- Firmware 1 Checksum (4 Bytes)</li> <li>- Firmware 2 Checksum (4 Bytes)</li> <li>- Operation Flags (1 Bytes)</li> </ul>																														
6	Disconnect	1 Byte	Ngắt kết nối BLE với DWM1001																														
7	Anchor List	129 Bytes	Danh sách các ID của tất cả node Anchor đang hoạt động trong mạng, tối đa là 16 Anchor.																														

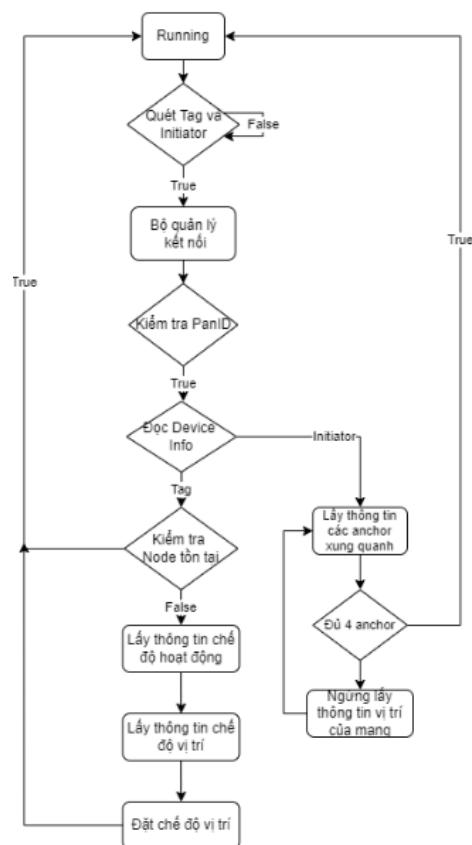
Dựa vào các characteristic được cung cấp, đồ án đưa ra giải pháp quản lý các node trong mạng UWB. Thiết bị sẽ quy định hai trạng thái cho mạng, gồm trạng thái *initialize* (khởi tạo) và trạng thái *running* (hoạt động). Lưu đồ thuật toán của hai trạng thái được mô tả ở hình 2.34 và 2.35.

Ở trạng thái Initialize, Gateway sẽ bắt đầu quét các anchor xung quanh mình. Khi quét được 1 anchor, thiết bị sẽ thiết lập kết nối GATT với node đó. Tiếp theo đó là bước kiểm tra PAN ID, nếu PAN ID của anchor trùng với của thiết bị, characteristic Device Info sẽ được đọc ngay sau đó. Với trường ID từ Device Info, Gateway có thể kiểm tra được liệu node anchor này đã nằm trong danh sách các node đã từng kết nối hay chưa. Nếu chưa, thông tin về chế độ hoạt động và về vị trí được đặt sẵn của Anchor sẽ lần lượt được đọc và quá trình quét thông tin sẽ kết thúc. Khi đã quét đủ 4 anchor, cùng với điều kiện trong 4 anchor đó bắt buộc phải có một anchor là initiator, thiết bị sẽ chuyển sang trạng thái Running.



**Hình 2.34 Lưu đồ thuật toán trạng thái Initialize**

Trạng thái Running ở hình 2.24, thiết bị sẽ quét liên tục các node xung quanh bao gồm tag và initiator. Với Tag, quá trình lấy thông tin tương tự như với anchor ở trạng thái Running, điểm khác là với Tag, cần đặt chế độ xuất vị trí là chế độ 0, bởi thiết bị Gateway chỉ cần thông tin về vị trí theo trục tọa độ, không cần thêm thông tin về khoảng cách từ tag đến các anchor xung quanh. Với initiator, thiết bị sẽ đọc characteristic Anchor List để kiểm tra trạng thái online của các anchor trong mạng. Nếu trong mạng không có đủ 4 anchor online, thiết bị sẽ ngừng lấy thông tin từ các Tag và chờ đến khi có đủ 4 anchor online.

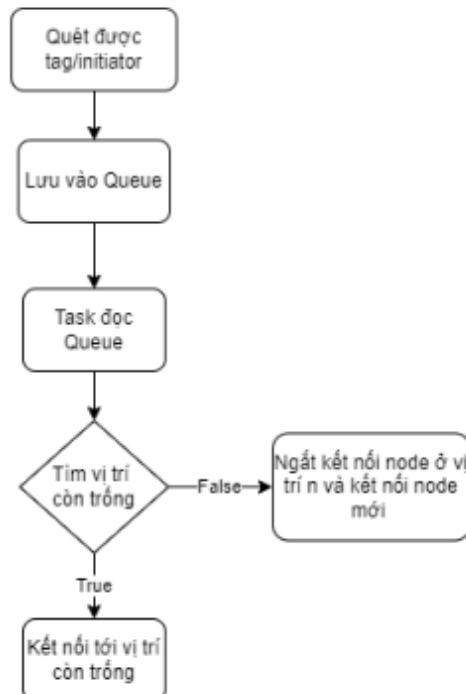


**Hình 2.35 Lưu đồ thuật toán trạng thái Running**

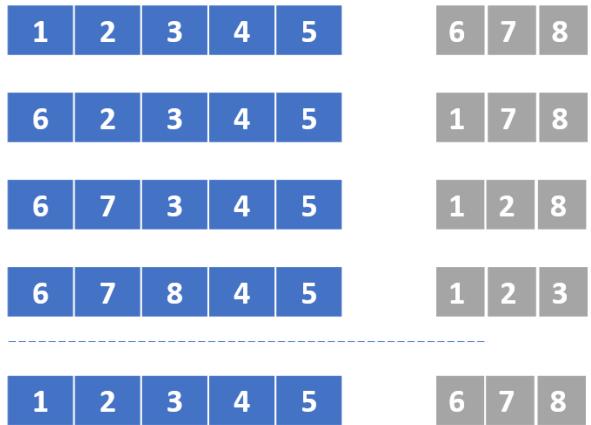
Sau khi một node được kết nối, vị trí của node đó sẽ được phân cố định đến khi thiết bị bị xóa tất cả các thông tin khi người dùng muốn cài đặt lại toàn bộ. Thông tin của node đó sẽ được lưu vào phân vùng NVS theo dạng key, value. Các thông tin được lưu lại bao gồm ID của node, địa chỉ bluetooth, các thông tin về chế độ hoạt động và thông tin chung của mạng. Các thông tin được lưu lại để phục vụ mục đích sau khi thiết bị khởi động lại, vẫn có đầy đủ thông tin về các node trong mạng để tiếp tục hoạt động của mạng. Thiết bị có thể khởi động lại do nhiều lý do như cập nhật firmware mới, chuyển đổi giữa Ethernet và WiFi,...

Trong trạng thái Running, sau khi quét được các node, thiết bị sẽ thiết lập kết nối GATT tới node đó để có thể giao tiếp, đọc các trạng thái như ở hình 2.35 và phải giữ các trạng thái đó liên tục. Tuy nhiên hầu hết các thiết bị nhúng do giới hạn về kiến trúc, tốc độ xử lý và bộ nhớ RAM nên không thể giữ quá nhiều kết nối GATT cùng một lúc, với ESP32, số kết nối tối đa là 5. Để có thể giao tiếp được nhiều tag cùng một lúc, khi ấy cần có kịch bản cụ thể để chia ra các khung thời gian sử dụng của mỗi thiết bị để quản lý được các thiết bị với tốc độ cập nhật tối đa.

Hình 2.36 mô tả lại quá trình quản lý kết nối, thiết bị tính toán để phân chia các vị trí kết nối (tạm gọi là slot) cho các thiết bị. Khi quét được một tag hoặc initiator, thông tin quét được sẽ được đưa vào một Queue và chờ được đọc bởi một Task. Ở Task đó, sau khi đã đọc được thông tin của node đang chưa được kết nối, sẽ tiến hành tìm vị trí còn trống trong 5 slot của hệ thống và kết nối tới thiết bị nếu vẫn còn vị trí còn trống. Nếu đã đủ 5 thiết bị kết nối, Task sẽ ngắt kết nối lần lượt các vị trí từ 1 đến 5 và kết nối các node mới vào. Quá trình được mô tả ở hình 2.37.



**Hình 2.36 Bộ lập lịch kết nối**



**Hình 2.37 Ví dụ quá trình quản lý kết nối**

Như đã mô tả ở trên, thiết bị sẽ lần lượt đọc vị trí của tất cả các tag hiện đang trong danh sách kết nối và gửi lên cloud. Chu kỳ đọc vị trí sau khi được thử nghiệm sẽ được cố định ở 50 ms để đảm bảo độ ổn định của hệ thống, để mỗi node khi kết nối vào mạng sẽ có thể đảm bảo được đọc vị trí ít nhất một lần, trước khi có thể bị ngắt kết nối và nhường vị trí cho các node khác.

### 2.6.2 MQTT

Kiến trúc về tốc độ xử lý cũng như bộ nhớ RAM của ESP32 đảm bảo cho việc triển khai MQTT kèm theo lớp bảo mật TLS được tron tru, ngoài ra trong SDK của hãng đã được hỗ trợ sẵn lớp bảo mật TLS, người dùng chỉ cần đưa đầy đủ các thông tin của TLS như Root CA, còn lại quá trình trao đổi public key, session key sẽ được SDK thực hiện.

Cloud của hệ thống sử dụng được thiết kế các topic MQTT theo chuẩn như đã trình bày ở phần 1.3.2.3, thiết bị cần tuân theo chuẩn đó và sẽ tạo ra các topic của riêng mình để định danh cho thiết bị đó.

**Bảng 2.9 MQTT Topic của thiết bị**

Topic	Mô tả
/devices/{Gateway_ID}/events	Bản tin sự kiện của Gateway
/devices/{Gateway_ID}/state	Bản tin trạng thái của Gateway
/devices/{Node_ID}/events	Bản tin sự kiện của node
/devices/{Node_ID}/state	Bản tin trạng thái của node

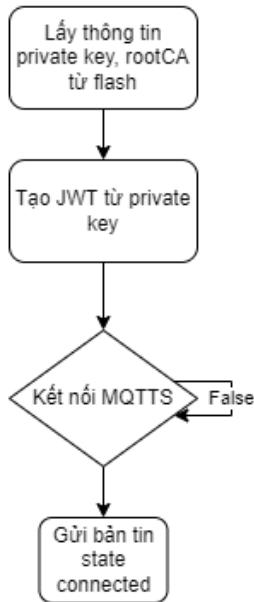
Với các topic được mô tả ở bảng 2.9, đồ án thiết kế thêm cấu trúc payload chung cho các topic đó. Payload sử dụng cấu trúc JSON gồm các trường:

- ts: timestamp. Mục đích của trường timestamp là để lưu trữ lại thời điểm xảy ra sự kiện hay thời điểm xảy ra trạng thái đó ở phía thiết bị, để khi đường truyền xảy ra lỗi và topic nhận được bị trễ thì có thể dựa vào timestamp để tiếp tục xử lý.
- id: trường ID định danh của thiết bị nhằm mục đích sắp xếp topic đúng với vị trí của thiết bị được tạo trên cloud
- method: là trường phân biệt giữa các trạng thái khác nhau, sự kiện khác nhau
- value: tùy vào các method, sẽ có các value khác nhau được mô tả ở bảng 2.8

**Bảng 2.10 Các payload thiết bị sử dụng**

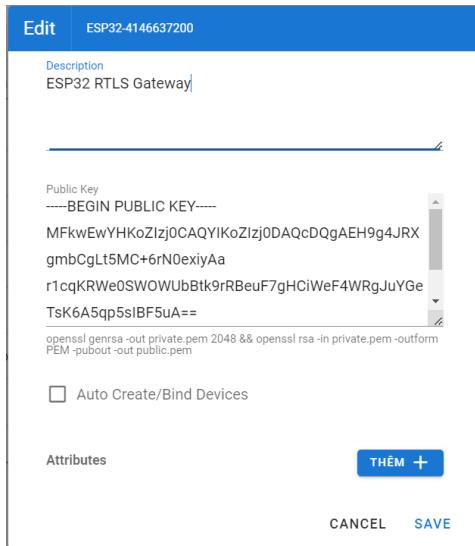
Thiết bị	Topic	Method	Value	Mô tả
Gateway	State	state	State: boolean	Topic được publish khi Gateway online
Gateway	Event	new_node	node_id: id của node mới kết nối node_type: tag/anchor slot_name: số thứ tự của node	Topic được publish khi có node mới vào mạng
Gateway	Event	dis_node	node_id: id của node	Topic được publish khi thiết bị xóa node khỏi hệ thống quản lý
Node	Event	loc_event	pos_x: vị trí trực x pos_y: vị trí trực y pos_z: vị trí trực z quality: chất lượng vị trí	Topic được publish khi có vị trí từ Tag
Node	Event	status	status: offline/online	Topic được gửi khi node online hoặc offline với timeout là 1 phút.

Sau khi đã có đầy đủ các định nghĩa về topic, payload sử dụng trong quá trình giao tiếp với cloud, đồ án sẽ đi vào phân tích quá trình kết nối của thiết bị tới cloud sử dụng MQTT với TLS hay còn gọi là MQTTS.



**Hình 2.38 Lưu đồ thuật toán quá trình kết nối MQTT**

Quá trình kết nối MQTT được bắt đầu khi thiết bị kết nối được WiFi hoặc Ethernet. Thông tin private key và rootCA sẽ được lấy từ bộ nhớ flash. Private key này khác với private key của lớp bảo mật TLS tuy nhiên về mục đích sử dụng là hoàn toàn giống nhau. Ở phía cloud, thiết bị sẽ được thêm sẵn một slot với ID của thiết bị và được cấp một public key như ở hình 2.39. Public key này được tạo ra cùng lúc và có mối quan hệ chặt chẽ với private key được lưu trong flash.



**Hình 2.39 Thêm public key khi khởi tạo thiết bị**

Private key đó sẽ được sử dụng để tạo ra một JWT, được đặt ở trường password của bản tin MQTT Connect.Thêm vào đó, root CA cũng sẽ được đọc từ flash, sau đó đưa vào khi kết nối tới MQTT để phục vụ quá trình thiết lập kết nối với TLS.

### 2.6.3 Smartconfig

Khi thiết bị đến tay người dùng, việc đầu tiên cần làm là kết nối WiFi cho thiết bị, thêm vào đó là cấu hình cho thiết bị thuộc registry nào, và cung cấp các private key cũng như root CA của Cloud, phục vụ cho quá trình kết nối MQTTS.

Với các thiết bị không hỗ trợ các phương tiện giao tiếp với người dùng như bàn phím, màn hình cảm ứng,... việc kết nối WiFi trực tiếp qua việc nhập tên và mật khẩu gần như là không thể. Do nhu cầu của thiết bị sử dụng WiFi ngày càng cao, Texas Instrument đã thiết kế ra một phương thức để có thể trao đổi dữ liệu về tên và mật khẩu WiFi với thiết bị không nằm trong mạng.

Trong phần trình bày của đồ án sẽ không đi sâu vào chi tiết của phương thức này mà sẽ giới thiệu về nguyên lý hoạt động của smartconfig. Phần cơ bản và quan trọng nhất của smartconfig đó là cách để một thiết bị đang trong mạng WiFi (thường là smartphone), có thể đưa dữ liệu tới một thiết bị ở ngoài mạng. SC sử dụng các gói tin UDP để đưa dữ liệu đi. Tuy nhiên, dữ liệu trong các gói tin UDP là dữ liệu đã được mã hóa và các thiết bị ở ngoài mạng không có key thì không thể giải mã được các dữ liệu đó. Khi đó, thiết bị sẽ sử dụng trường độ dài của bản tin. Trường độ dài của bản tin là trường không có mã hóa, khi đó các ký tự của tên WiFi, mật khẩu sẽ được đưa lần lượt

vào từng bản tin và gửi liên tục. Bản tin UDP của Smartphone sẽ được gửi đến địa chỉ IP Gateway, khi đó thiết bị ngoài mạng sẽ lắng nghe các bản tin xung quanh ở tất cả các kênh khác nhau. Khi bắt được gói tin ở kênh phù hợp, thiết bị sẽ lắng nghe ở kênh đó và kết thúc quá trình khi có bản tin kết thúc. Chi tiết về phương thức Smartconfig có thể tham khảo [tại đây](#).

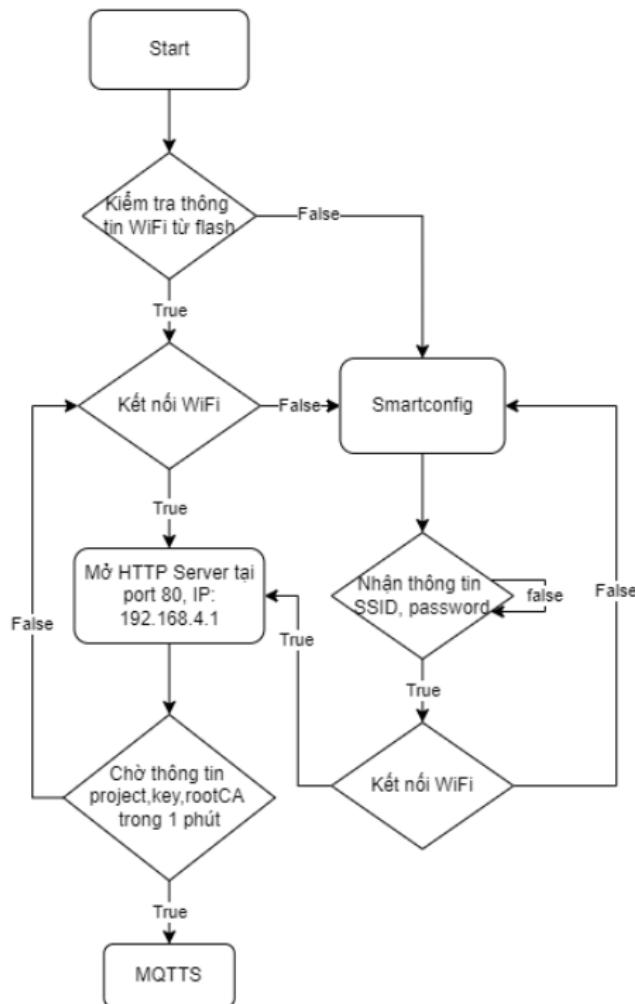
Sau khi kết nối WiFi xong, thiết bị sẽ bật chế độ Access Point, trở thành một điểm phát WiFi và sẽ được thiết lập một HTTP Server tại Access Point này. HTTP Server sẽ được gán địa chỉ IP là 192.168.4.1, kèm theo 2 api và các bản tin HTTP POST tới các uri đó được mô tả ở bảng 2.11. Toàn bộ quá trình smartconfig và cấu hình được mô tả trong lưu đồ thuật toán tại hình 2.40.

**Bảng 2.11 Các URI của HTTP Server**

URI	Body	Response
/api/setup/project	POST { "host": "{host}", "port": {port}, "project_id": "{project_id}", "registry_id": "{registry_id}" }	HTTP OK
/api/setup/cert	POST { "root_ca": { "length": {ca_length}, "data": "{root_ca_data}" } "private_key": { "length": {private_key_length}, "data": "{private_key_data}" } }	HTTP OK

Body của các gói tin HTTP POST là các gói tin theo dạng JSON. Với uri */api/setup/project*, nội dung sẽ gồm host – URL của cloud, port – port 8883 theo chuẩn MQTTS, project\_id là id của project được tạo sẵn trên cloud và tương tự với registry\_id là id của registry được tạo sẵn. Với uri */api/setup/cert*, thông tin sẽ gồm private\_key cho quá trình tạo JWT và root CA cho quá trình thiết lập kết nối MQTTS.

Việc thực hiện smartconfig cũng như cấu hình các thông tin sẽ được thực hiện qua app, được hỗ trợ bởi team phần mềm. Sau khi app gửi thông tin tới thiết bị, thiết bị sẽ lần lượt được vào mạng WiFi, nhận được gói tin cấu hình project và gói tin cấu hình key, rootCA. Các thông tin cấu hình sẽ được lưu lại vào bộ nhớ flash, cụ thể là vùng nhớ user\_param được mô tả ở bảng 2.7.



**Hình 2.40 Lưu đồ thuật toán quá trình cấu hình thiết bị**

#### 2.6.4 Ethernet

Framework ESP-IDF của ESP32 hỗ trợ tầng MAC cho giao tiếp Ethernet với IC W5500, cùng với đó là TCP/IP stack, có thể được sử dụng cho cả Ethernet cũng như WiFi. Công việc chính cho phần Ethernet đó là phát hiện được tín hiệu Ethernet khi người dùng muốn sử dụng và chuyển đổi sang Ethernet khi nhận được IP.

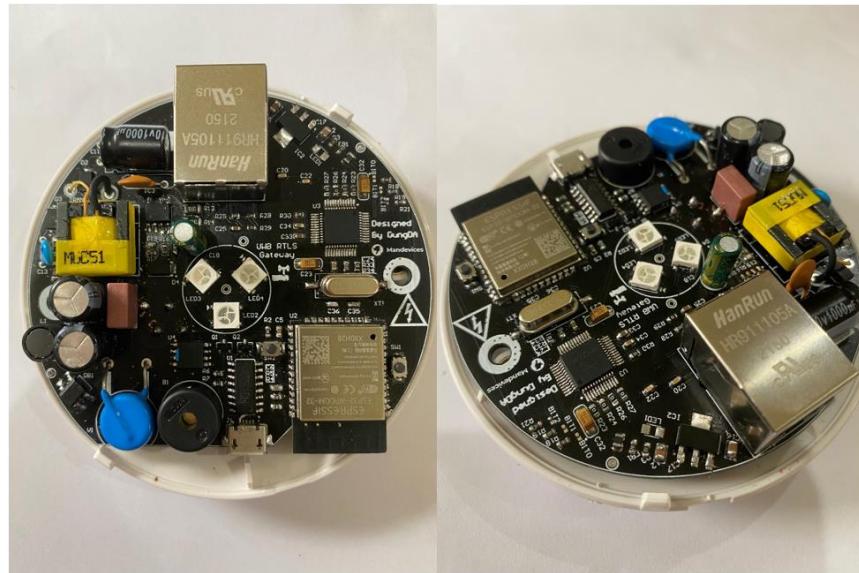
Khi có tín hiệu từ W5500, thiết bị sẽ reset để bắt đầu vào đầu chương trình, nơi mà sẽ lựa chọn một trong hai phương thức Ethernet và WiFi, mà trong đó Ethernet sẽ

được ưu tiên hơn. Khi reset, thiết bị sẽ không bị mất các thông tin về các thiết bị đã kết nối từ trước do đã được lưu vào flash. Trong vòng 1 phút, thiết bị sẽ chờ được cấp IP và sau đó sẽ tiếp tục reset để vào chế độ WiFi.

## CHƯƠNG 3. KẾT QUẢ THỬ NGHIỆM VÀ ĐÁNH GIÁ

### 3.1 Kết quả thiết kế, chế tạo

#### 3.1.1.1 Kết quả chế tạo mạch in



Hình 3.1 Mặt trên mạch PCB



Hình 3.2 Mặt dưới mạch PCB

### *3.1.1.2 Kết quả thiết bị đóng hộp*



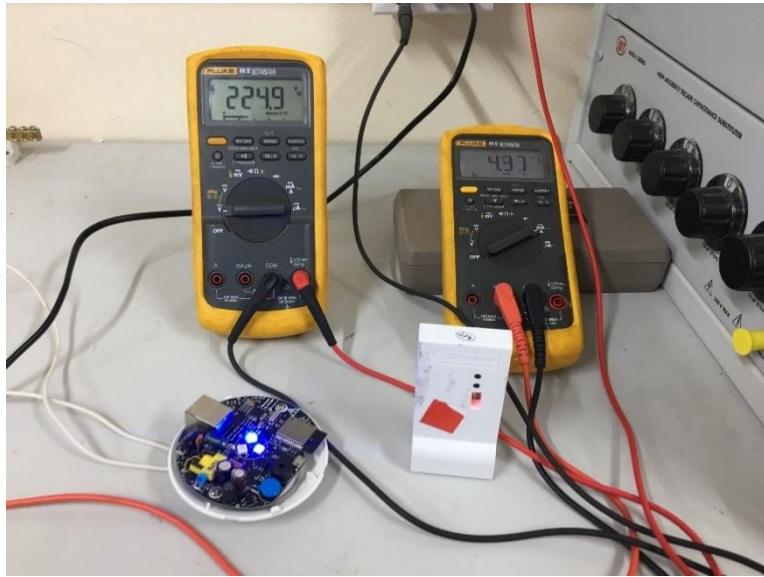
**Hình 3.3 Kết quả đóng hộp của thiết bị**

## **3.2 Kết quả thử nghiệm**

### *3.2.1 Kiểm tra hoạt động của thiết bị*

Mục đích của việc thử nghiệm hoạt động thiết bị là để kiểm tra từng khối trong thiết bị, đưa ra các bài test phù hợp độ ổn định của thiết bị trước khi đưa ra thị trường và sản xuất.

### 3.2.1.1 Kiểm tra hoạt động khởi nguồn



Hình 3.4 Kiểm tra hoạt động khởi nguồn

Khởi nguồn thiết bị hoạt động ổn định trong dải điện áp 220V 50Hz. Để kiểm tra hoạt động khởi nguồn, em sử dụng đồng hồ Fluke 88 với thang đo ACA 0.0001mA ~ 10A/1.2%, ACV 0.01mV ~ 1000V/0.5%. Kết quả thu được ở hình 3.4 với  $V_{RMS} = 225V$ ,  $I_{RMS} = 4.37mA$ . Như vậy công suất của thiết bị sẽ vào khoảng 1W khi bật tất cả các ngoại vi gồm WiFi, BLE, đèn LED, ....

### 3.2.1.2 Kiểm tra hoạt động khởi MCU

Sử dụng dây micro USB kết nối máy tính tới cổng micro trên mạch, máy tính nhận cổng COM để giao tiếp với thiết bị. Sau đó, sử dụng ESP-IDF Powershell để monitor các thông số log qua UART, nạp, xóa firmware mới cho thiết bị qua UART.

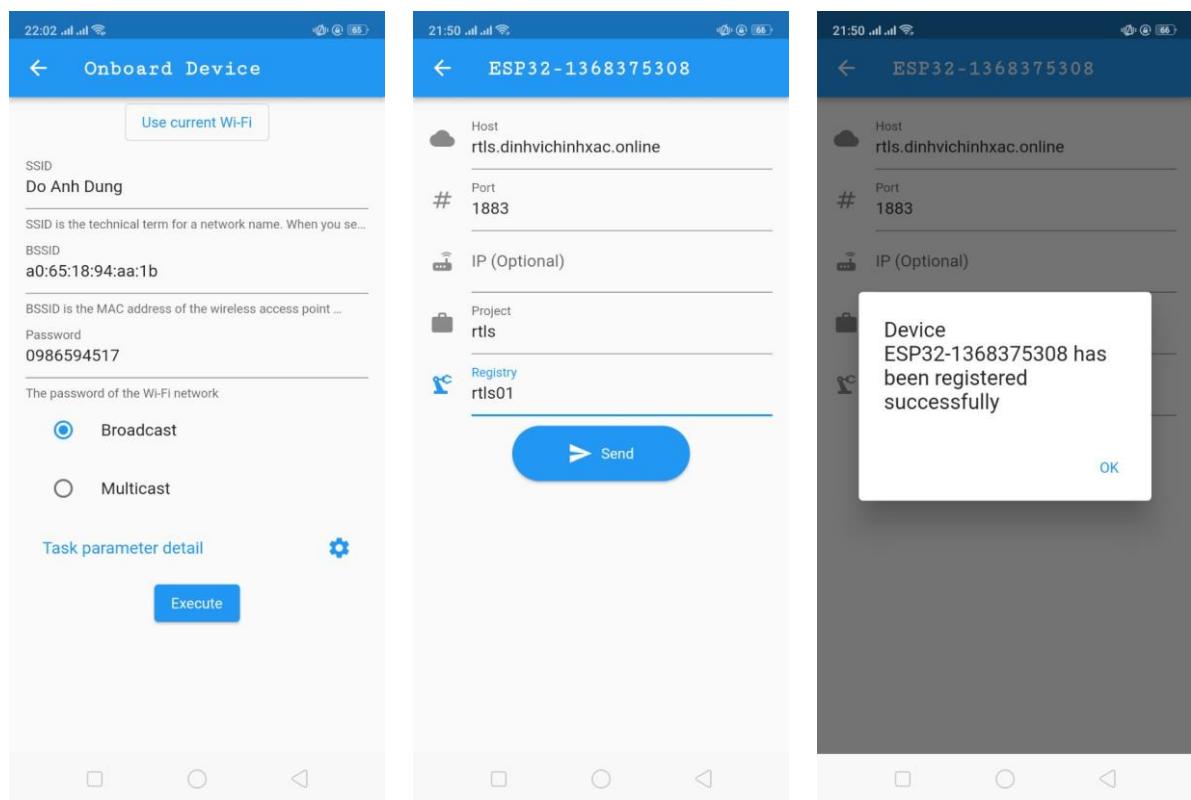
```
I (347) cpu_start: Starting scheduler on PRO CPU.
I (0) cpu_start: Starting scheduler on APP CPU.
I (362) log: Main APP running.
=====
Fw: 125 Hw: 101 Type: Device Author: DungDA Build: Aug 6 2022, 16:04:50
=====
I (451) syslog: SysLog partition found: log_data ,0x2ce000 ,0xfe000
I (461) syslog: Empty location to Save data: Offset 552, Block index 4
I (461) system_api: Base MAC address is not set
I (463) system_api: read default base MAC address from EFUSE
I (470) log: Chip ID 1368375308
I (473) nvs_store: NVS partition found: nvs ,0x9000 ,0x4000
I (480) nvs_store: Update ["FIRMWARE_VER"] in NVS: Done
I (485) nvs_store: Update ["HARDWARE_VER"] in NVS: Done
I (491) nvs_store: ["FIRMWARE_VER"] ["125"]
I (496) nvs_store: ["HARDWARE_VER"] ["101"]
```

Hình 3.5 Kiểm tra hoạt động monitor, nạp, xóa firmware qua cổng microUSB

### 3.2.1.3 Kiểm tra hoạt động khỏi Ethernet

### 3.2.1.4 Kiểm tra tính năng Smartconfig

Cài đặt app cho smartphone, sau đó đưa thiết bị vào chế độ config bằng cách nhấn giữ nút reset trong 5 giây. Sau khi thiết bị vào trạng thái cấu hình, đèn LED báo hiệu sẽ có màu đỏ và thực hiện các thao tác như hình 3.4. Theo dõi trên màn hình log dữ liệu của thiết bị, quan sát được quá trình smartconfig, kết nối WiFi và đọc các dữ liệu cấu hình, key, rootCA, và cuối cùng là kết nối với MQTT Broker qua TLS. Khi kết thúc chế độ cấu hình, đèn LED báo hiệu sẽ có màu xanh, tương ứng với trạng thái kết nối thành công MQTT Broker.



Hình 3.6 Giao diện APP Smartconfig

```
(1061) smartconfig: Running Smart config.  
I (1069) phy_init: phy_version 4670,719f96,Feb 18 2021,17:07:07  
I (1170) wifi_mode : sta (40:91:51:8f:c0:0c)  
I (1171) wifi:enable tsf  
I (1223) smartconfig: SC version: V3.0.1  
I (5324) wifi:ic_enable_sniffer  
I (5326) smartconfig: Start to find channel...  
I (5326) smartconfig: Scan done  
I (21512) smartconfig: TYPE: ESPTOUCH  
I (21513) smartconfig: T|PHONE MAC:9c:0c:df:1d:c2:33  
I (21513) smartconfig: T|AP MAC:a0:65:18:94:aa:1b  
I (21514) smartconfig: Found channel on 11-0. Start to get ssid and password...  
I (21522) smartconfig: Found channel  
I (24388) smartconfig: T|pswd: 0986594517  
I (24388) smartconfig: T|ssid: Do Anh Dung  
I (24389) smartconfig: T|bssid: a0:65:18:94:aa:1b  
I (24389) wifi:ic_disable_sniffer  
(24393) smartconfig: Got SSID and password  
(24397) smartconfig: SSID:Do Anh Dung  
(24402) smartconfig: PASSWORD:0986594517  
I (25701) wifi:new:<11,0>, old:<11,0>, ap:<255,255>, sta:<11,0>, prof:1  
I (26937) wifi:state: init -> auth (b0)  
I (26942) wifi:state: auth -> assoc (0)  
I (26948) wifi:state: assoc -> run (10)  
W (26955) wifi:cba-addtid:0 (ifx:0, a0:65:18:94:aa:1b), tid:0, ssn:0, winSize:64  
I (26965) wifi:connected with Do Anh Dung, aid = 7, channel 11, BW20, bssid = a0:65:18:94:aa:1b  
I (26966) wifi:security: WPA2-PSK, phy: bgn, rssi: -44  
I (26978) wifi:pm start type: 1
```

**Hình 3.7 Thiết bị smartconfig và kết nối WiFi**

```
(34336) esp_server: Starting server on port: '80'  
(34344) esp_server: Registering URI handlers  
(34347) log: Time synchronization start.  
(34751) log: The current date/time in VietNam is: Fri Aug 5 21:50:17 2022  
(56494) esp_server: content-len[121]  
(56495) esp_server: uri = api/setup/project  
(56495) esp_server: Host: rtls.dinhvichinhxac.online  
(56499) esp_server: Port: 1883  
(56503) esp_server: ProjectID: rtls  
(56507) esp_server: Location: default  
(56511) esp_server: RegistryID: rtls01  
(56518) nvs_store: Update ["HOST"] in NVS: Done  
(56522) nvs_store: Update ["PORT"] in NVS: Done  
(56528) nvs_store: Update ["PROJECT"] in NVS: Done  
(56534) nvs_store: Update ["LOCATION"] in NVS: Done  
(56539) nvs_store: Update ["REGISTRY"] in NVS: Done  
(56573) esp_server: content-len[3196]
```

**Hình 3.8 Thiết bị nhận gói tin cấu hình project**

```

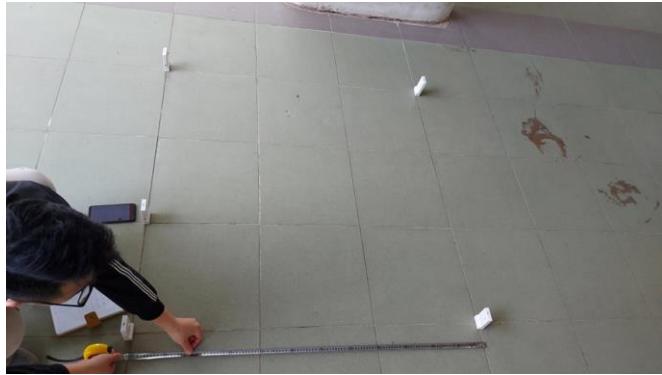
t (56672) uri: api/setup/cert
t (56681) root_ca : [2110]-----BEGIN CERTIFICATE-----
MIIF6TCkABwIBAqXUAsRAsIkYUemrmtTP815xzKjiiY0wQDQYJkoZhvcNAQEN
BQAvEqYIxCaZ1AbgIBMVABTA1ZQmSwpQYDVQYQIDA1QSDeQAUgAA1UEB#F5Gb2kx
F4jAuBNgBAwQDm1LzTzQxtGhcdGzVmexJ#tAaMwQDMSw1lTzXQzGxhdGzVm0x
J5jkAgrNBWVHNBX3J0bPmZG1uaHzpWj2hpbmh4WVub25as51MCAXDT1TyDUw
TE2ZDM10M10yDzXmJh1wNDA1UETWjzDyBjCgELA1GKA1UEBMVKAxzJ3BgNV
BAGMAB1M04wADWVQDHFADW1V5wTswTEMBPQIA1UEGwNa11wD1vBfGz0y9b1yEW
BMOGAUeCwNa11wD1b2f0n9yBemTEmGCGQ1AUewAfD0frcsy5k5wldmlj
aLu1hahY9v5bxpmwUggE11MAGGScG51b3D0QEAUgA4ICDwAuggA1kaoICAQDE
nhRH5/70EFDnead0EPYRBl/c19j3axPTag1b2v164y+Eh12/s7ks1smbcwlCB
r-pXvPiUQAF0gA67NkGdnqNGSKYqjpiDn/t0spCEafLuAmQgKwdu/Mzmn9
4EPgFx19pt9rx0yNxJbfKlxnsNxpH0j3axPTag1b2v164y+Eh12/s7ks1smbcwlCB
LMTQ5PlF1dyC0/TkFxJ+do953mZUkC05s7TfCaXQ2K9Ph+MuA12z2xegEqY
LMTQ5LkZQbJ1g82bw3luwlRorK8B0YeoN69M5L2epEcF/BfA007ygtIDTM20/Duxa
ULs4xPL6HZtE079m9avFbgxotCx1u7xa7vOfArT81v0mGqvzLhuGa1zd51xw0
Jt4yT3P5L6010emDuo/ERTX0Lp+ir+j1B15xzsu1tPIyX0wku155ndF89+yodKJ2d
KK9T1KicqB0abzfvbve950KtMhNaOeB1uEeMmN08N6g/rgsfwDvczrW1z1277
zyVlhVAlX4NpkSwTE7o1+QMCz1KpcfqVghJawHgjcgNWyNs6/f9/Xpma01/G1R5
cna8yQdwhsXJ7/2DmpfJ1ksaIYOLkIdEz31DuF1Ujh1d75tgAkgLkRfTzg
+0+im11/ldqyds3JKQTqYf/fe6dX0XFdv/+1vDv2T1QDQABo1M0Tu7dglnVHQ4E
FgQUsandRuyipveJMRQ1La4Chrm4/04wHvdR0jB8gfwFoAUsandRuyipveJMR
01a4Ch2FnD0wvDVR0tAQH/awBa/wEb/ +AnBkgK1g9w0B0AQAFoAACAgAT1Gv-
B2tBzMsH1zC135b7Q3ZEF01Bh1A01fD1pnyMrK1gkGngC6N8ZBhPaR1C
kxNWMydzb9898InKuHoEs2B1wvhdyhNerf5TzJ1K11/1K1tFdKnhJk19
Ku/cygBwL1H9wy4QEutB0rlw1Fa1AVFHlW81pu1m6iq7Gz4cHaZcpDnBeGy7
0y1leze0xSyvBmY5jhsAKdJvg9hdB0Ict1fGe8z0Stkj2s63Zk1n1NH0d1jSYm1
j09P3D1NSZKbWMP9bIa1J24LOC+rnfMLIKHOZCSYkr1d6u6h6x4d4UHEKT
f8hK8Pug99dkvHdI41941ZLXKEgYB7/FUG7LkFpxprY110dplkd1m788XAt68
f1vT/m7WjN9wpLMD+Rv+jwv14G2LcC1uA1PkzWbYez1Xj1p7yZx10L2t3eFP
n11PmK8Fw2L2L1f6G29yBQxEpFwz74C4iJmdMpq+Rw71PmRzPeRbxMyuH
lw15jwgNaVL5PQo+ueFpsYUrB95X0tHw15g/bf8fJd0r3k+b09nuq/h/qACAA
ufBmfSM8e0Nmx19y54f105a0wQtbC9yMhjoMgv3A1LvvVQxqASvU1gTpN+PWeY9c
uKyof+910zVtVlnQPK/80dYnusbsq31QnEz56c-
-----END CERTIFICATE-----
t (56859) log: write 2110 byte to partition
t (56871) nvs_store: Update ["CA_LEN"] in NVS: Done
t (56872) private_key : [226]-----BEGIN EC PRIVATE KEY-----
MhQgAACQEEICQF1jk0fSqzGckV1U2MUsaw2wpG3jw3dz5z3Kyp9ku6Akbggkhj0
PQBMB6FEAO1ABgsX0fLNLsEJh+Lzm5wkoOpJ3A32BlespoShouxymMkWt+NC4v4a
5j05+tkF4+Aefi0MdMrM7y07zaAd/EUf=
-----END EC PRIVATE KEY-----
t (56893) log: write 226 byte to partition
t (56899) nvs_store: Update ["PRVKEY_LEN"] in NVS: Done

```

**Hình 3.9 Thiết bị nhận gói tin key và rootCA**

### **3.2.2 Thử nghiệm tính năng của hệ thống**

#### **3.2.2.1 Thử nghiệm độ chính xác của hệ thống DWM1001**



**Hình 3.10 Thử nghiệm độ chính xác của hệ thống DWM1001**

Là một trong những phần quan trọng nhất của hệ thống, việc kiểm thử độ chính xác của hệ thống UWB là rất cần thiết để có thể đánh giá được tính chính xác cũng như tốc độ đáp ứng của hệ thống.

Mục tiêu: kiểm tra độ chính xác theo lý thuyết của module là  $< 10$  cm và khoảng cách tối đa để giữ được độ chính xác đó là 60m trong điều kiện LoS.

Kích bản: hệ thống được cấu hình sử dụng 4 anchor và 1 tag. Khoảng cách của các anchor và tag sẽ lần lượt được thay đổi. Kết quả sẽ được theo dõi sử dụng app Android được nhà sản xuất cung cấp. App Android sử dụng giao tiếp BLE để thu thập các thông tin từ các node. Các mẫu đo được lấy sau mỗi 5 giây, diện tích đo là diện tích mặt phẳng được tạo bởi 4 anchor.

**Bảng 3.1 Kết quả thử nghiệm tính năng hệ thống DWM1001**

Diện tích (cm x cm)	Lý thuyết (cm)	Thực đo (cm)			
		Lần 1	Lần 2	Lần 3	Lần 4
120 x 120	(40,0)	(33,2)	(32, 0)	(33, 2)	(32, 6)
	(80, 40)	(75, -8)	(77, 5)	(79, 4)	(78, 7)
	(80, 80)	(77, 71)	(77, 77)	(77, 82)	(78, 84)
320 x 320	(160, 200)	(156, 198)	(149, 193)	(151, 192)	(157, 193)
1000x520	(640, 320)	(649, 325)	(639, 319)	(645, 323)	(643, 321)

3000x2000	(2988,1992)	(2992,1993)	(2995,1991)	(2994,1994)	(2993,1992)
5000x3000	(4986,2985)	(4983, 2986)	(4988, 2984)	(4979,2981)	(4982,2987)

Dựa trên kết quả thử nghiệm ở bảng 3.1 cùng với nhiều phép đo riêng lẻ khác, em có thể đi đến kết luận, hệ thống DWM1001 đạt được độ chính xác 10cm ổn định trong mặt phẳng với diện tích 40m x 40m. Với diện tích lớn hơn, độ chính xác sẽ giảm xuống còn khoảng 20 – 30 cm.

Kết quả thử nghiệm trên là một bước quan trọng trong quá trình thiết kế hệ thống, kiểm nghiệm lại tính năng của toàn hệ thống UWB để có thể tự tin hơn trong việc thiết kế Gateway sau này.

### 3.2.2.2 Thử nghiệm hoạt động hệ thống DWM1001 kết hợp Gateway

Sau khi đã có kết quả thử nghiệm hệ thống UWB và đã thiết kế xong Gateway, em tiến hành thử nghiệm tổng thể hệ thống gồm hệ thống UWB, Gateway và Cloud. Môi trường thử nghiệm gồm môi trường không gian thoáng trong quá trình nghiên cứu, thiết kế gateway và môi trường nhà máy với các điều kiện nhiễu, vật cản.

#### Môi trường không gian thoáng:



**Hình 3.11 Không gian thử nghiệm**

Không gian thử nghiệm có diện tích là 30m x 6m, với 4 anchor ở 4 điểm góc của không gian. Mỗi anchor được đặt ở độ cao 4m để mô phỏng lại vị trí của anchor trong các không gian thực tế.



**Hình 3.12 Thiết bị Gateway và 6 Tag**

Thiết bị gateway được cấp nguồn 5V từ ắc quy để phục vụ cho thử nghiệm, WiFi được lấy từ nguồn phát điện thoại. Kịch bản test tương tự như phần 3.2.2 với bài test độ chính xác, thêm vào đó là thử nghiệm kết nối 6 tag cùng một lúc để kiểm tra độ ổn định của module quản lý node trên gateway.

Hình 3.13 là kết quả của quá trình initialize, gateway kết nối đủ 4 anchor, đọc thông tin và sau đó ngắt kết nối tới 4 anchor đó.

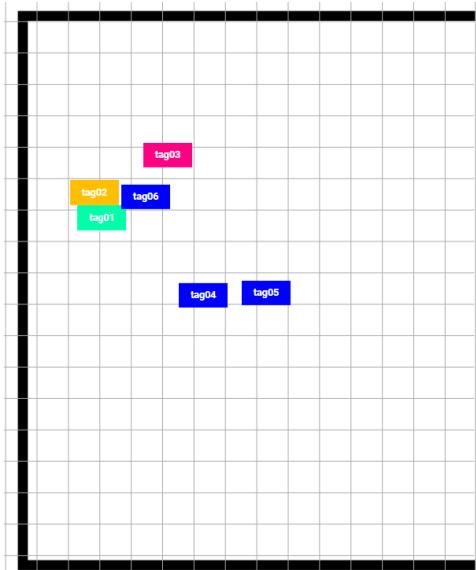
Events	{ "ts": 1659983140, "method": "status", "values": { "node_id": "DECAF84AAAD00375", "slot_name": "anchor04", "status": "offline" } }
Events	{ "ts": 1659983139, "method": "status", "values": { "node_id": "DECA13303DF002E1", "slot_name": "anchor03", "status": "offline" } }
Events	{ "ts": 1659983139, "method": "status", "values": { "node_id": "DECAC62C09902217", "slot_name": "anchor02", "status": "offline" } }
Events	{ "ts": 1659983139, "method": "status", "values": { "node_id": "DECAD42574C045F7", "slot_name": "anchor01", "status": "offline" } }
Events	{ "ts": 1659983137, "method": "new_node", "values": { "node_id": "DECAF84AAAD00375", "node_type": "anchor", "slot_name": "anchor04" } }
Events	{ "ts": 1659983132, "method": "new_node", "values": { "node_id": "DECA13303DF002E1", "node_type": "anchor", "slot_name": "anchor03" } }
Events	{ "ts": 1659983128, "method": "new_node", "values": { "node_id": "DECAC62C09902217", "node_type": "anchor", "slot_name": "anchor02" } }
Events	{ "ts": 1659983126, "method": "new_node", "values": { "node_id": "DECAD42574C045F7", "node_type": "anchor", "slot_name": "anchor01" } }
State	{ "ts": 1659983124, "id": 1368375308, "method": "state", "values": { "state": true } }

**Hình 3.13 Quá trình Initialize**

Sau đó thiết bị sẽ quét tất cả các tag và module quản lý node sẽ hoạt động, quản lý các tag xung quanh nó. Kết quả hiển thị vị trí của 6 tag được mô tả ở hình 3.15.

Events	{ "ts": 1659983169, "method": "new_node", "values": { "node_id": "DECAA6DED4E02247", "node_type": "tag", "slot_name": "tag06" } }
Events	{ "ts": 1659983154, "method": "new_node", "values": { "node_id": "DECA06121C702218", "node_type": "tag", "slot_name": "tag05" } }
Events	{ "ts": 1659983150, "method": "new_node", "values": { "node_id": "DECA60D0AE50231C", "node_type": "tag", "slot_name": "tag04" } }
Events	{ "ts": 1659983147, "method": "new_node", "values": { "node_id": "DECA9157B570222C", "node_type": "tag", "slot_name": "tag03" } }
Events	{ "ts": 1659983145, "method": "new_node", "values": { "node_id": "DECA5A264AB024D8", "node_type": "tag", "slot_name": "tag02" } }
Events	{ "ts": 1659983142, "method": "new_node", "values": { "node_id": "DECAC98CD5A045C8", "node_type": "tag", "slot_name": "tag01" } }

**Hình 3.14 Thiết bị kết nối 6 tag**

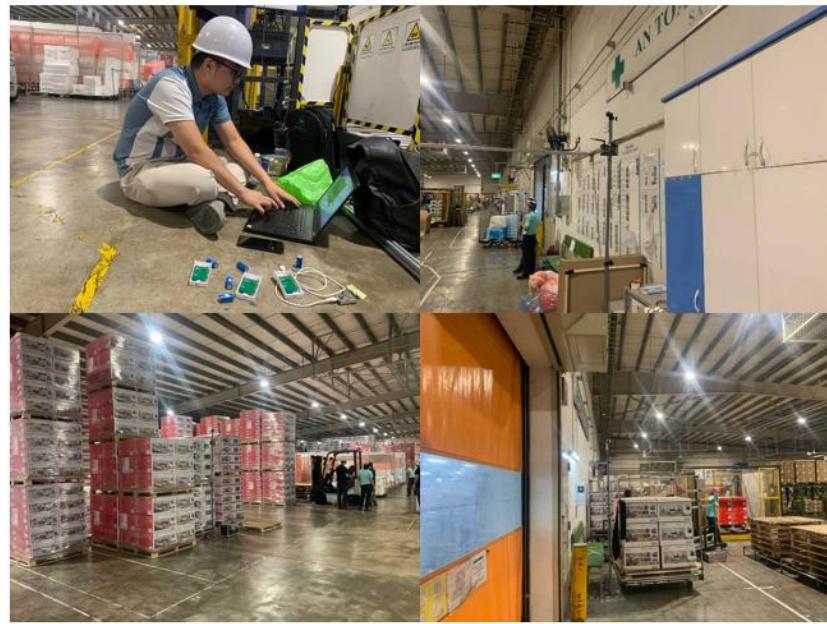


**Hình 3.15 Vị trí của 6 tag hiển thị trên bản đồ**

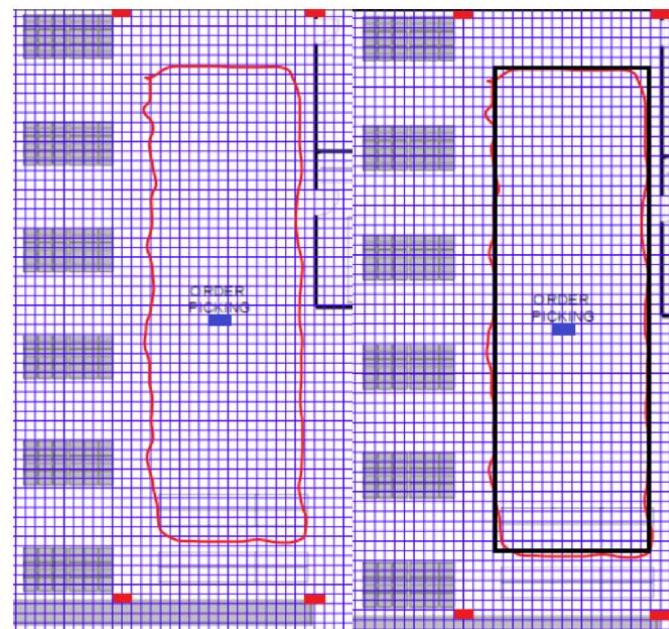
#### Môi trường nhà máy:

Bài toán đặt ra tại nhà máy là theo dõi các xe nâng hàng để từ đó xác định được vị trí của các pallet hàng. Hệ thống thử nghiệm bao gồm 4 cột cố định được gắn anchor, với độ cao 2.5m, tạo thành diện tích 30m x 10m. Vị trí của Gateway được đặt ở tâm của diện tích, đảm bảo khoảng cách khoảng 16m tới mỗi node anchor.

Tag được gắn trên xe nâng và đi theo đường đã được định sẵn. Kết quả của bài test được mô tả ở hình 3.14. Hình bên trái là kết quả của quá trình đo, hình bên phải là so sánh giữa kết quả đo và kết quả mong muốn. Mỗi ô trên hình tương ứng với kích thước 50 cm x 50 cm, đối sánh với hình vẽ ta thấy được các điểm kết quả đo không lệch quá 1 ô so với kết quả mong muốn, tương ứng với đo chính xác nhỏ hơn 50cm. Cùng với đó, kết nối và tín hiệu trong môi trường nhà máy ổn định là một dấu hiệu tốt cho bài test. Do thời gian và hạn chế về cơ sở vật chất được cung cấp tại nhà máy khi thử nghiệm như xe nâng, không gian,... nên chưa thể thực hiện được nhiều bài test nhất có thể.



Hình 3.16 Thủ nghiệm hệ thống tại hiện trường



Hình 3.17 Kết quả thử nghiệm hệ thống tại hiện trường

## CHƯƠNG 4. KẾT LUẬN

### 4.1 Kết luận chung

Sau thời gian nghiên cứu, phát triển hệ thống, em đã đạt được các kết quả sau:

- Phân tích, lựa chọn hệ thống UWB, mô hình mạng phù hợp với điều kiện kinh tế và ứng dụng.
- Phân tích, thiết kế phần cứng bao gồm mạch PCB và vỏ hộp, cùng với firmware của thiết bị Gateway phù hợp với các yêu cầu bài toán đặt ra

Thiết bị Gateway cùng với hệ thống đã đạt được các kết quả:

- Hệ thống yêu cầu phải có 4 anchor, với ít nhất một initiator trong 4 anchor đó. Gateway có thể quản lý 4 anchor và vô hạn các tag.
- Độ chính xác của vị trí từ các tag đạt độ ổn định trong khoảng 10 – 30 cm. Tốc độ cập nhật tối đa vào khoảng 10Hz và giảm dần dựa trên số lượng tag mà hệ thống quản lý.
- Các chức năng WiFi, Ethernet, BLE, giao tiếp Flash, LED RGB, còi chíp hoạt động ổn định.
- Các thông tin trao đổi giữa Gateway và Cloud được bảo mật tuyệt đối sử dụng lớp bảo mật TLS.
- Thiết bị được thiết kế chức năng smartconfig và chế độ cấu hình, cho phép người dùng, người lắp đặt tự cấu hình thiết bị theo ý muốn.

Tuy nhiên, do thời gian nghiên cứu và thiết kế còn chưa đủ, thiết bị và hệ thống vẫn còn những hạn chế sau:

- Chưa được thử nghiệm với số lượng lớn tag do hạn chế về kinh tế
- Chưa được tích hợp tính năng OTA – cập nhật firmware từ xa do chưa có server OTA để thử nghiệm
- Hệ thống chưa được kiểm nghiệm chạy liên tục ở môi trường thực tế như nhà máy để kiểm tra độ ổn định tại môi trường nhà máy.

- Luồng cấu hình dữ liệu từ phía Cloud xuống thiết bị chưa được tích hợp do phía Cloud chưa hỗ trợ.

## 4.2 Hướng phát triển

Sau khi thiết bị được cung cấp server OTA, thiết bị sẽ cập nhật thêm tính năng OTA để có thể bảo trì, cập nhật firmware cho thiết bị từ xa.

Với thiết bị Gateway, sẽ được phát triển để tích hợp thêm các thiết bị BLE xung quanh, trở thành một Gateway chung cho tất cả các thiết bị BLE. Cùng với đó, luồng dữ liệu cấu hình từ Cloud xuống thiết bị sẽ được triển khai ngay khi có hỗ trợ từ phía Cloud.

Với hệ thống UWB, em sẽ tiếp tục nghiên cứu thêm mô hình sử dụng Bridge Node (thay vì sử dụng BLE, sẽ sử dụng UWB để giao tiếp với các node trong mạng). Khi đó sẽ tận dụng tối đa được tài nguyên của hệ thống.

## **TÀI LIỆU THAM KHẢO**

- [1] [Online]. Available: <https://en.wikipedia.org/wiki/Ultra-wideband>
- [2] [Online]. Available: <https://cloud.google.com/iot-core>
- [3] [Online]. Available: <https://tiptopsecurity.com/how-does-https-work-rsa-encryption-explained/>
- [3] Espressif, esp-wroom-32\_datasheet\_en\_v3.3, 2022
- [4] Novel Bits, Intro to Bluetooth Low Energy, 2018
- [5] NXP, TEA1721 Product Datasheet, 2012
- [6]Wiznet, W5500 Datasheet V1.0.2, 2013
- [7]Winbond, W25Q128FV Datasheet, 2021
- [7]DWM1001 Firmware Application Programming Interface (API) Guide V2.2, 2019
- [8]DWM1001 System Overview And Performance V2.0, 2018