**LỜI CẢM ƠN**

Trong quá trình thực hiện và hoàn thành đồ án tốt nghiệp này, lời đầu tiên em xin cảm ơn chân thành đến toàn thể quý thầy cô trong Học viên Công nghệ Bưu chính Viễn thông và các thầy cô trong khoa Kỹ thuật điện tử nói chung, bộ môn Điện tử máy tính nói riêng, những người đã tận tình dạy dỗ, hướng dẫn và trang bị cho em những kiến thức bổ ích trong suốt những năm học vừa qua.

Đặc biệt, em xin chân thành gửi lời cảm ơn sâu sắc đến thầy giáo Ts.Nguyễn Quốc Uy, người đã trực tiếp hướng dẫn, chỉ bảo, dìu dắt em thực hiện đồ án, giúp em có những kiến thức về mặt chuyên môn cũng như thực tế và tạo mọi điều kiện giúp em trong suốt quá trình làm đồ án tốt nghiệp.

Cuối cùng em xin gửi lời cảm ơn đến gia đình, bạn bè đã luôn động viên, cổ vũ và đóng góp ý kiến trong quá trình học tập, nghiên cứu cũng như quá trình làm đồ án tốt nghiệp.

Mặc dù đã cố gắng hết sức để hoàn thiện đồ án tốt nghiệp nhưng do kiến thức ,kinh nghiệm và cơ sở vật chất vẫn có những hạn chế nhất định, do đó không thể tránh khỏi những thiếu sót. Vì vậy, em rất mong được sự cảm thông, ý kiến đóng góp từ quý thầy cô và các bạn.

Em xin chân thành cảm ơn!

|  |
| --- |
| *Hà Nội, ngày 22 tháng 12 năm 2019*  *Phạm Đức Khánh* |

MỤC LỤC

[LỜI CẢM ƠN 1](#_Toc28123143)

[DANH MỤC CÁC KÍ HIỆU, CÁC CHỮ VIẾT TẮT 5](#_Toc28123144)

[DANH MỤC HÌNH ẢNH 6](#_Toc28123145)

[DANH MỤC CÁC BẢNG 9](#_Toc28123146)

[LỜI NÓI ĐẦU 10](#_Toc28123147)

[CHƯƠNG I: TỔNG QUAN VỀ CÔNG NGHỆ LORA VÀ ỨNG DỤNG CỦA NÓ TRONG NÔNG NGHIỆP THÔNG MINH 12](#_Toc28123148)

[1.1. Tổng quan về công nghệ Lora 12](#_Toc28123149)

[1.1.1. Lora là gì ? 12](#_Toc28123150)

[1.1.2. Nguyên lý hoạt động và các thông số hoạt động của Lora 13](#_Toc28123151)

[1.2. Ứng dụng trong nông nghiệp thông minh 15](#_Toc28123152)

[CHƯƠNG II: THIẾT KẾ PHẦN CỨNG HỘP CẢM BIẾN THU THẬP DỮ LIỆU NHIỆT ĐỘ ĐỘ ẨM KHÔNG KHÍ VÀ ĐỘ ẨM ĐẤT 17](#_Toc28123153)

[2.1. Mô hình hệ thống thu thập dữ liệu từ các hộp cảm biến 17](#_Toc28123154)

[2.2. Xây dựng sơ đồ nguyên lý của một hộp cảm biến 18](#_Toc28123155)

[2.2.1. Sơ đồ khối 18](#_Toc28123156)

[2.2.2. Thiết kế nguyên lý khối nguồn cho mạch 19](#_Toc28123157)

[2.2.3. Thiết kế nguyên lý khối điều khiển trung tâm 22](#_Toc28123158)

[2.2.4. Thiết kế nguyên lý khối module Lora Rhf76-052 25](#_Toc28123159)

[2.2.5. Thiết kế nguyên lý khối cảm biến nhiệt độ độ ẩm 29](#_Toc28123160)

[2.2.6. Thiết kế sơ đồ nguyên lý kết nối STM32 và cảm biến độ ẩm đất 33](#_Toc28123161)

[2.3. Thiết kế layout cho mạch trong hộp cảm biến 38](#_Toc28123162)

[2.3.1. Giới thiệu 38](#_Toc28123163)

[2.3.2. Thiết kế layout khối nguồn 39](#_Toc28123164)

[2.3.3. Thiết kế layout khối điều khiển trung tâm 40](#_Toc28123165)

[2.3.4. Thiết kế layout khối module Lora 40](#_Toc28123166)

[2.3.5. Thiết kế layout các khối cảm biến 41](#_Toc28123167)

[2.3.6. Hình ảnh 3D trên phần mềm Altilum Designer của mạch 41](#_Toc28123168)

[CHƯƠNG III: XÂY DỰNG MODULE PHẦN MỀM ĐỌC DỮ LIỆU TỪ CÁC CẢM BIẾN VÀ TRUYỀN TẢI DỮ LIỆU THÔNG QUA LORA 43](#_Toc28123169)

[3.1. Xây dựng module phần mềm đọc dữ liệu từ cảm biến nhiệt độ độ ẩm không khí 43](#_Toc28123170)

[3.1.1. Quá trình giao tiếp của MCU với SHT75 43](#_Toc28123171)

[3.1.2. Xây dựng các hàm đọc dữ liệu 47](#_Toc28123172)

[3.2. Xây dựng module phần mềm đọc dữ liệu từ cảm biến độ ẩm đất 49](#_Toc28123173)

[3.2.1. Modbus-RTU 49](#_Toc28123174)

[3.2.2. Xây dựng các hàm đọc dữ liệu 50](#_Toc28123175)

[3.3. Xây dựng module phần mềm truyền dữ liệu cho bộ thu thông qua Lora 51](#_Toc28123176)

[3.3.1. Giao tiếp UART 51](#_Toc28123177)

[3.3.2. Lập trình gửi dữ liệu qua Lora 54](#_Toc28123178)

[3.4. Xây dựng module phần mềm đưa mạch vào chế độ nghỉ 57](#_Toc28123179)

[CHƯƠNG IV: THIẾT KẾ BỘ THU NHẬN DỮ LIỆU QUA LORA VÀ XÂY DỰNG GIAO DIỆN THEO DÕI THÔNG SỐ NHẬN ĐƯỢC TỪ BỘ THU 60](#_Toc28123180)

[4.1. Thiết kế bộ thu nhận dữ liệu qua Lora 60](#_Toc28123181)

[4.1.1. Sơ đồ khối 60](#_Toc28123182)

[4.1.2. Sơ đồ nguyên lý và layout 61](#_Toc28123183)

[4.2. Xây dựng giao diện theo dõi 65](#_Toc28123184)

[4.2.1. Giới thiệu về PyQt5 65](#_Toc28123185)

[4.2.2. Xây dựng giao diện theo dõi các thông số và kết nối với Thingsboard server 67](#_Toc28123186)

[CHƯƠNG V: HOÀN THIỆN VÀ KẾT LUẬN 73](#_Toc28123187)

[5.1. Hoàn thiện sản phẩm 73](#_Toc28123188)

[5.2. Kết luận 74](#_Toc28123189)

[5.3. Hướng phát triển của đề tài 74](#_Toc28123190)

[TÀI LIỆU THAM KHẢO 76](#_Toc28123191)

[PHỤ LỤC 77](#_Toc28123192)

**DANH MỤC CÁC KÍ HIỆU, CÁC CHỮ VIẾT TẮT**

**DANH MỤC HÌNH ẢNH**

Chương I:

[Hình 1. 1: Hình ảnh về Lora 7](#_Toc28039501)

[Hình 1. 2: Tỷ lệ C/R 10](#_Toc28039502)

[Hình 1. 3: Ứng dụng công nghệ trong nông nghiệp 11](#_Toc28039503)

Chương II:

[Hình 2. 1: Mô hình hệ thống thu thập dữ liệu từ các hộp cảm biến 13](#_Toc28039507)

[Hình 2. 2: Sơ đồ khối của một hộp cảm biến 14](#_Toc28039508)

[Hình 2. 3: Sơ đồ nguyên lý khối nguồn 15](#_Toc28039509)

[Hình 2. 4: Block diagram của IC HT7133 16](#_Toc28039510)

[Hình 2. 5: Hình ảnh pin cấp nguồn cho mạch 17](#_Toc28039511)

[Hình 2. 6: Sơ đồ nguyên lý khối vi điều khiển smt32 18](#_Toc28039512)

[Hình 2. 7: Các tụ lọc cho vi điều khiển STM32 20](#_Toc28039513)

[Hình 2. 8: Header để nạp code và gỡ rối 21](#_Toc28039514)

[Hình 2. 9: Block diagram của RHF-76052 21](#_Toc28039515)

[Hình 2. 10: Sơ đồ chân của module RHF76-052 22](#_Toc28039516)

[Hình 2. 11: Thông số kỹ thuật của module RHF76-052 23](#_Toc28039517)

[Hình 2. 12: Sơ đồ nguyên lý khối module Lora Rhf76-052 24](#_Toc28039518)

[Hình 2. 13: Hình ảnh cảm biến SHT7x,SHT1x 25](#_Toc28039519)

[Hình 2. 14: Block Diagram của SHT75 26](#_Toc28039520)

[Hình 2. 15: Thông số kỹ thuật hiệu suất của cảm biến 27](#_Toc28039521)

[Hình 2. 16: Sơ đồ kết nối MCU và SHT7x 28](#_Toc28039522)

[Hình 2. 17: Sơ đồ nguyên lý khối đo nhiệt độ độ ẩm không khí 28](#_Toc28039523)

[Hình 2. 18: Cảm biến đo độ ẩm đất 29](#_Toc28039524)

[Hình 2. 19: Các cách sử dụng cảm biến đất trong môi trường 30](#_Toc28039525)

[Hình 2. 20: Module chuyển đổi RS485 to TTL 31](#_Toc28039526)

[Hình 2. 21: Sơ đồ nguyên lý khối đo độ ẩm đất 32](#_Toc28039527)

[Hình 2. 22: PCB toàn bộ bo mạch(đã ẩn toàn bộ lớp phủ đồng) 33](#_Toc28039528)

[Hình 2. 23: Hình ảnh layout khối nguồn 34](#_Toc28039529)

[Hình 2. 24: Hình ảnh layout khối điều khiển trung tâm 35](#_Toc28039530)

[Hình 2. 25: Hình ảnh layout khối module Lora 35](#_Toc28039531)

[Hình 2. 26:Hình ảnh layout các khối cảm biến 36](#_Toc28039532)

[Hình 2. 27: Hình ảnh 3D của bo mạch trên phần mềm 36](#_Toc28039533)

Chương III:

[Hình 3. 1: Tín hiệu bắt đầu truyền nhận của MCU và SHT75 37](#_Toc28039540)

[Hình 3. 2: Ví dụ về đo độ ẩm bằng SHT75 38](#_Toc28039541)

[Hình 3. 3: Thông số d,d2 để tính nhiệt độ khi đọc từ cảm biến SHT75 38](#_Toc28039542)

[Hình 3. 4: Giá trị t1, t2 39](#_Toc28039543)

[Hình 3. 5: Công thức tính RHliner 39](#_Toc28039544)

[Hình 3. 6: Cấu trúc một bản tin Modbus RTU 42](#_Toc28039545)

[Hình 3. 7: Hai thiết bị giao tiếp UART 45](#_Toc28039546)

[Hình 3. 8: - Khung truyền dữ liệu trong giao tiếp UART 46](#_Toc28039547)

[Hình 3. 9: Mã lỗi và mô tả các lỗi của RHf76-052 48](#_Toc28039548)

[Hình 3. 10: Mã lệnh AT cho mo-đun Rhf76-052 48](#_Toc28039549)

[Hình 3. 11: Khi STM32 hoạt động ở chế độ STOP mode 50](#_Toc28039550)

Chương IV:

[Hình 4. 1: Sơ đồ khối bộ thu tín hiệu Lora kết hợp với máy tính 52](#_Toc28039551)

[Hình 4. 2: Nguyên lý khối USB=>SERIAL 53](#_Toc28039552)

[Hình 4. 3: Hình ảnh thực tế IC CH340 55](#_Toc28039553)

[Hình 4. 4: Sơ đồ nguyên lý khối module Lora 56](#_Toc28039554)

[Hình 4. 5: Hình ảnh 3D của bo mạch thu tín hiệu Lora 56](#_Toc28039555)

[Hình 4. 6: Hình ảnh biểu tượng của PyQt 57](#_Toc28039556)

Chương V:

[Hình 5. 1: Phía bên trong hộp cảm biến 70](#_Toc28120890)

[Hình 5. 2: Giao diện theo dõi 3 cảm biến trên máy tính chạy window 71](#_Toc28120891)

**DANH MỤC CÁC BẢNG**

Chương I:

[Bảng 1. 1: Bảng đặc điểm công nghệ Lora 9](#_Toc28039604)

Chương II:

[Bảng 2. 1: Bảng thông số các giá trị linh kiện trong khối nguồn 16](#_Toc28039611)

[Bảng 2. 2: Phạm vi hoạt động của module RHF76-052 22](#_Toc28039612)

[Bảng 2. 3: Bảng sơ đồ chân kết nối của SHT75 27](#_Toc28039613)

Chương III:

[Bảng 3. 1: Lệnh và mã lệnh cho SHT75 37](#_Toc28039618)

# LỜI NÓI ĐẦU

1. **Lý do chọn đề tài**

Với mong muốn tạo ra một hệ thống thu thập dữ liệu với khoảng cách xa hàng km, hay những khu vực không có Wifi, những nơi hẻo lánh không có sóng di động. Giúp mọi người có thể tiếp cận được, đơn giản như người nông dân có thể thu thập dữ liệu cảm biến nhiệt độ, độ ẩm đất từ cánh đồng xa xôi.

Trong thời đại công nghệ 4.0, với khả năng truyền không dây xa mang kèm tính năng tiết kiệm năng lượng công nghệ Lora rất phù hợp vào các hệ thống nông nghiệp thông minh, qua đó có thể giúp đỡ bà con nông dân thuận tiện hơn trong quá trình lao động và sản xuất của họ . Ứng dụng công nghệ Lora có thể giúp đỡ họ tăng suất và giảm thiệt hại từ tác động của môi trường . Từ ý tưởng đó em đã chọn đề tài “XÂY DỰNG HỘP CẢM BIẾN TRONG NÔNG NGHIỆP THÔNG MINH SỬ DỤNG CÔNG NGHỆ LORA” làm đề tài đồ án tốt nghiệp của mình.

1. **Mục tiêu**:

* Hiểu rõ đặc điểm, tính năng của công nghệ Lora.
* Thiết kế thành công hộp cảm biến đo nhiệt độ ẩm không khí và độ ẩm đất ứng dụng được công nghệ Lora để truyền tải dữ liệu.
* Xây dựng bộ thu dữ liệu từ hộp cảm biến và kết nối bộ thu với máy tính.
* Xây dựng giao diện hiển thị dữ liệu trên máy tính.
* Thiết kế vỏ hộp để có hộp hoạt động lâu dài trong môi trường tự nhiên.

1. **Phạm vi nghiên cứu:**

Đồ án tập trung nghiên cứu và đưa ra giải pháp thiết kế cho hộp cảm biến ứng dụng được công nghệ Lora và hoạt động lâu dài ổn định trong môi trường tự nhiên và có thể ứng dụng được trong nông nghiệp thông minh.

1. **Phương pháp nghiên cứu:**

Nghiện cứu lý thuyết thiết kế mô hình thực nghiệm và đánh giá kết quả đạt được.

1. **Bố cục đề tài:**

* Chương 1: Tổng quan về công nghệ Lora và ứng dụng của nó trong nông nghiệp thông minh.
* Chương 2: Thiết kế phần cứng hộp cảm biến thu thập dữ liệu nhiệt độ độ ẩm không khí và độ ẩm đất.
* Chương 3: Xây dựng module phần mềm đọc dữ liệu từ các cảm biến và truyền tải dữ liệu qua Lora.
* Chương 4: Thiết kế bộ thu nhận dữ liệu qua Lora và xây dựng giao diện theo dõi thông số nhận được từ bộ thu.
* Chương 5: Hoàn thiện và kết luận.

**CHƯƠNG I: TỔNG QUAN VỀ CÔNG NGHỆ LORA VÀ ỨNG DỤNG CỦA NÓ TRONG NÔNG NGHIỆP THÔNG MINH**

## 1.1. Tổng quan về công nghệ Lora

**LoRa**là viết tắt của Long Range Radio được nghiên cứu và phát triển bởi Cycleo và sau này được mua lại bởi công ty Semtech năm 2012. Với công nghệ này, chúng ta có thể truyền dữ liệu với khoảng cách lên hàng km mà không cần các mạch khuếch đại công suất; từ đó giúp tiết kiệm năng lượng tiêu thụ khi truyền/nhận dữ liệu. Do đó, LoRa có thể được áp dụng rộng rãi trong các ứng dụng thu thập dữ liệu như mạng cảm biến trong đó các node cảm biến có thể gửi giá trị đo đạc về trung tâm cách xa hàng km và có thể hoạt động với pin trong thời gian dài trước khi cần thay pin.



Hình 1. 1: Hình ảnh về Lora

Dưới đây là bảng đặc điểm của công nghệ giao tiếp LoRa về khoảng cách , chuẩn giao tiếp, công suất, lớp vật lý.

|  |  |
| --- | --- |
| Đặc điểm kỹ thuật | Các giá trị và định nghĩa |
| Khoảng cách | 2-5 km trong thành thị và 15 km vùng ngoại ô |
| Băng tần | Băng tần ISM 433MHZ, 868MHZ và 915 MHZ |
| Chuẩn giao tiếp | IEEE 802.15.4g |
| Điều chế | Điều chế giải phổ được sử dụng giải băng tần FM. Tần số tăng hoặc giảm trên thời gian nhất định được sử dụng để mã hóa dữ liệu khi được gửi . |
| Công suất | Một mạng LoRa quản lý hàng ngàn node |
| Điện năng | Ít hao tổn điện năng |
| Lớp vật lý | Quản lý tần số ,công suất ,điều chế ,tín hiệu giữa các node và gateway |

Bảng 1. 1: Bảng đặc điểm công nghệ Lora

Nguyên lý hoạt động và các thông số hoạt động của Lora

a) Nguyên lý hoạt động

LoRa sử dụng kỹ thuật điều chế gọi là Chirp Spread Spectrum. Có thể hiểu nôm na nguyên lý này là dữ liệu sẽ được băm bằng các xung cao tần để tạo ra tín hiệu có dãy tần số cao hơn tần số của dữ liệu gốc (cái này gọi là chipped); sau đó tín hiệu cao tần này tiếp tục được mã hoá theo các chuỗi chirp signal (là các tín hiệu hình sin có tần số thay đổi theo thời gian; có 2 loại chirp signal là up-chirp có tần số tăng theo thời gian và down-chirp có tần số giảm theo thời gian; và việc mã hoá theo nguyên tắc bit 1 sẽ sử dụng up-chirp, và bit 0 sẽ sử dụng down-chirp) trước khi truyền ra anten để gửi đi.

Theo Semtech công bố thì nguyên lý này giúp giảm độ phức tạp và độ chính xác cần thiết của mạch nhận để có thể giải mã và điều chế lại dữ liệu; hơn nữa LoRa không cần công suất phát lớn mà vẫn có thể truyền xa vì tín hiệu Lora có thể được nhận ở khoảng cách xa ngay cả độ mạnh tín hiệu thấp hơn cả nhiễu môi trường xung quanh.

Băng tần làm việc của LoRa từ 430MHz đến 915MHz cho từng khu vực khác nhau trên thế giới:

* 430MHz cho châu Á.
* 780MHz cho Trung Quốc.
* 433MHz hoặc 866MHz cho châu Âu.
* 915MHz cho USA.

b) Các thông số hoạt động của Lora

Speading Factor (SF):

Speading Factor là yếu tố lan truyền của Lora.SF xác định số lượng chrip signal khi mã hóa tín hiệu đã được điều chế tần số (chipped signal) của dữ liệu. Ví dụ nếu SF=12 có nghĩa là 1 mức logic của chipped signal sẽ được mã hóa bởi 12 xung chirp signal.

Ví dụ: SX1278 có giá trị SF từ 6 đến 12.

Giá trị SF càng lớn thì thời gian truyền càng lâu nhưng khoảng cách xa hơn.

Bandwidth (BW):

Bandwidth là đặc trưng cho lưu lượng tín hiệu được truyền qua thiết bị LoRa trong 1 khoảng thời gian. BW càng nhỏ thời gian truyền càng lâu nhưng khoảng cách đạt được xa hơn.

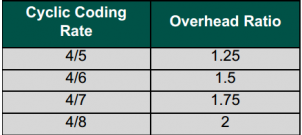
Các BW được sử dụng: 125 kHz, 250 kHz, 500 kHz

Code Rate(CR):

Code Rate là tỉ lệ mã hóa bit của LoRa. CR là số lượng bit được tự thêm vào mỗi trong Payload trong LoRa radio packet bởi LoRa chipset để mạch nhận có thể sử dụng để phục hồi lại 1 số bit dữ liệu đã nhận sai và từ đó phục hồi được nguyên vẹn dữ liệu trong Payload. Do đó, sử dụng CR càng cao thì khả năng nhận dữ liệu đúng càng tăng nhưng bù lại chip LoRa sẽ phải gửi nhiều dữ liệu hơn (có thể làm tăng thời gian truyền).

Có 4 giá trị cho CR là 4/5, 4/6, 4/7 và 4/8. VD: CR = 4/8 thì cứ mỗi 4 bits data nó sẽ được mã hóa bởi 8 bits.

Ví dụ : Với chipset SX1276 thì chúng ta có 4 giá trị cho CR là 4/5, 4/6, 4/7 và 4/8. Tương ứng mỗi giá trị của CR thì số lượng dữ liệu tăng thêm như sau:



Hình 1. 2: Tỷ lệ C/R

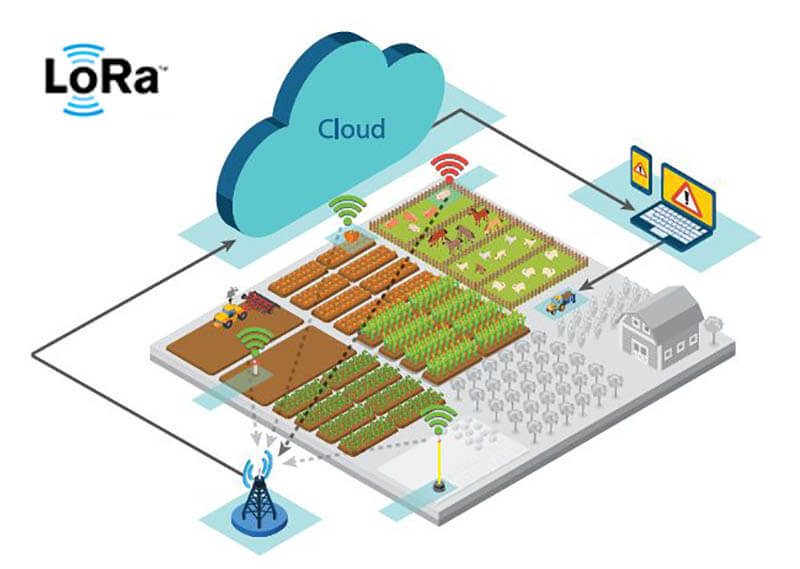
## 1.2. Ứng dụng trong nông nghiệp thông minh

Theo Tổ chức Lương thực và Nông nghiệp Liên Hợp Quốc, thế giới sẽ cần sản xuất thêm 70% lương thực vào năm 2050 so với năm 2006 để nuôi sống dân số toàn cầu ngày càng tăng. Các doanh nghiệp nông nghiệp đã và đang chuyển sang Internet of Things (IoT) để phân tích và khả năng sản xuất lớn hơn khi nhu cầu sản xuất và vận hành tiếp tục tăng.

Nông dân và chủ trang trại không còn có thể dựa vào tín hiệu thị giác hoặc các chỉ số thụ động và lịch sử khác để quản lý doanh nghiệp của họ. Thay vào đó, họ phải tạo ra các hoạt động hiệu quả hơn để thu thập thông tin thường xuyên trong lĩnh vực này, nhanh chóng tổng hợp dữ liệu và đưa ra quyết định kinh doanh thông minh để gặt hái những lợi ích của việc chăn nuôi chính xác và giám sát chăn nuôi theo thời gian thực.

Ứng dụng IoT được thiết lập để đẩy tương lai của kinh doanh nông nghiệp lên một tầm cao mới. Ở một số quốc gia, nông nghiệp thông minh đang trở nên phổ biến hơn, nhưng để đẩy nhanh việc áp dụng công nghệ IoT trên toàn thế giới, nông dân và người chăn nuôi cần các cảm biến năng lượng thấp, chi phí thấp.

Bằng cách tận dụng Công nghệ LoRa các doanh nghiệp nông nghiệp có thể theo dõi kỹ thuật số, quản lý và phân tích mọi khía cạnh của doanh nghiệp của họ, cải thiện hoạt động chung và lợi tức đầu tư (ROI). Từ theo dõi sức khỏe gia súc, theo dõi chăn nuôi bò đến tăng năng suất cây trồng, công nghệ LoRa cung cấp một nền tảng vững chắc cho tương lai của nông nghiệp thông minh vì nó dễ triển khai và giúp nông dân và người chăn nuôi phát triển kinh doanh.



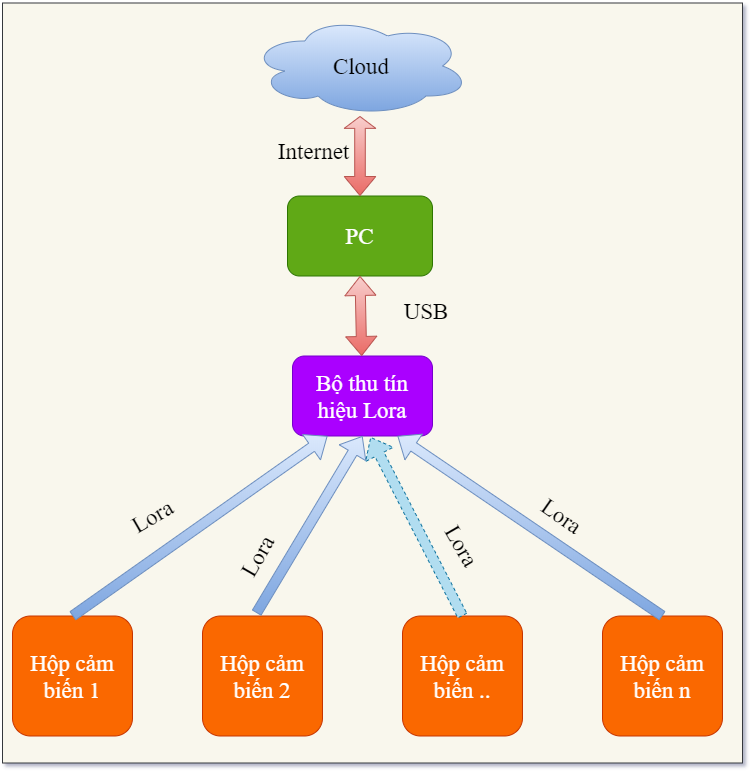
Hình 1. 3: Ứng dụng công nghệ trong nông nghiệp

# CHƯƠNG II: THIẾT KẾ PHẦN CỨNG HỘP CẢM BIẾN THU THẬP DỮ LIỆU NHIỆT ĐỘ ĐỘ ẨM KHÔNG KHÍ VÀ ĐỘ ẨM ĐẤT

Trong chuong nay em lam gi;..……………………………………

## 2.1. Mô hình hệ thống thu thập dữ liệu từ các hộp cảm biến

Từ ý tưởng sử dụng nhiều loại cảm biến từ các node cảm biến để có thể thu thập dữ liệu nhiệt độ độ ẩm không khí và độ ẩm đất từ cánh đồng rộng mênh mông có diện tích lên đến hàng km em đã xây dựng một mô hình hệ thống thu thập dữ liệu. Dưới đây là hình ảnh mô hình hệ thống thu thập dữ liệu từ các hộp cảm biến.



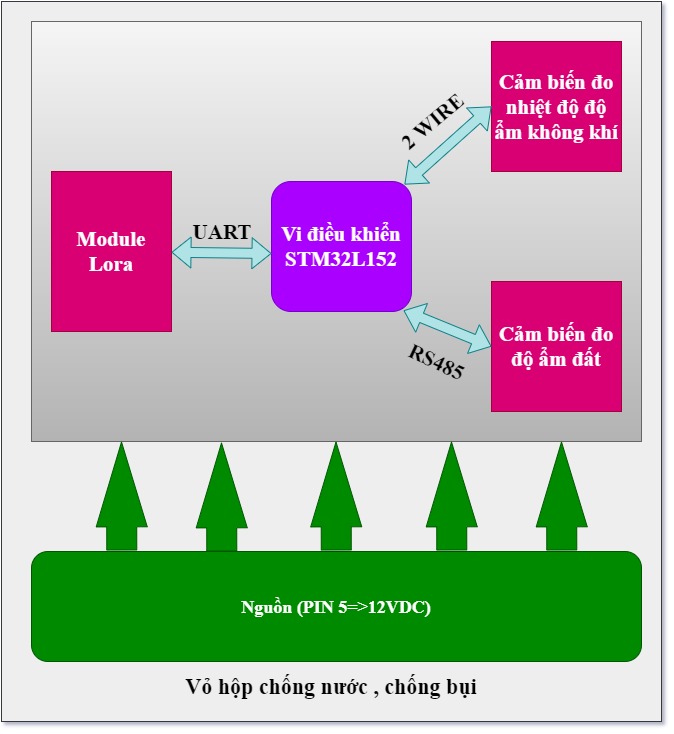
Hình 2. 1: Mô hình hệ thống thu thập dữ liệu từ các hộp cảm biến

Hệ thống sẽ có rất nhiều hộp cảm biến mỗi hộp cảm biến (node cảm biến) này thu thập dữ liệu tại một vị trí nhất định trong khu vực. Sau khi đo đạc chuẩn hóa dữ liệu , dữ liệu đo được sẽ được các hộp cảm biến này gửi lên cho bộ thu tín hiệu thông qua Lora. Bộ thu chỉ khi nào nhận được lệnh yêu cầu lấy dữ liệu từ máy tính sẽ chuyển vào hoạt động ở chế độ chờ nhận dữ liệu và khi nó nhận được dữ liệu nó sẽ gửi trả cho máy tính. Máy tính thu nhận dữ liệu này lưu trữ vào cơ sở dữ liệu đồng thời hiển thị dữ liệu lên giao diện để người sử dụng thuận tiện quan sát. Sau đó máy tính sẽ gửi dữ liệu đó lên một server thông qua internet. Vì vậy người sử dụng sẽ có thể quan sát nó mọi lúc mọi nơi. Các thiết kế chi tiết của từng khối sẽ được trình bày ở phần sau.

## 2.2. Xây dựng sơ đồ nguyên lý của một hộp cảm biến

### 2.2.1. Sơ đồ khối của hộp cảm biến

Dưới đây là hình ảnh sơ đồ khối của một hộp cảm biến , một hộp cảm biến sẽ có các khối như dưới đây:

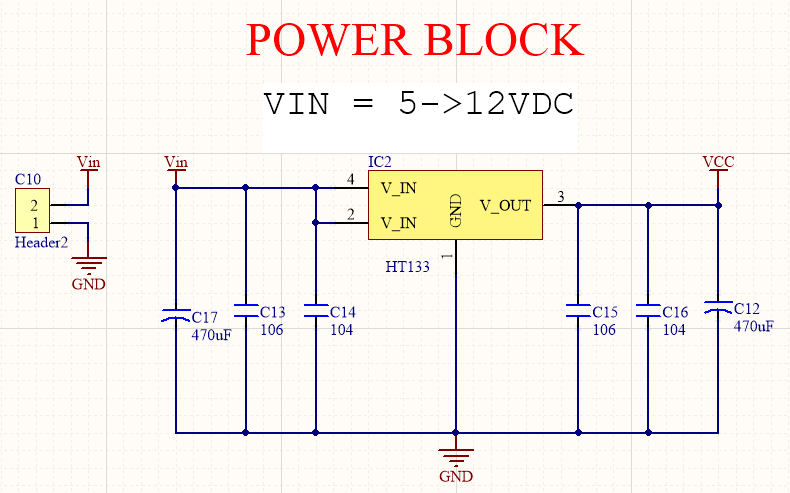


Hình 2. 2: Sơ đồ khối của một hộp cảm biến

Theo như sơ đồ khối trên , khối nguồn sử dụng 4 quả pin cung cấp nguồn cho toàn bộ các khối khác trong mạch, khối điều khiển trung tâm có một vi điều khiển STM32L152 để làm nhiệm vụ đo dữ liệu từ 2 khối cảm biến nhiệt độ độ ẩm không khí và cảm biến độ ẩm đất rồi giao tiếp với khối module Lora để gửi dữ liệu qua Lora. Tất cả được đóng gói trong một vỏ hộp chống bụi chống nước. Những phần sau sẽ đi chi tiết tới từng khối của bo mạch bên trong hộp.

### 2.2.2. Thiết kế nguyên lý khối nguồn cho mạch

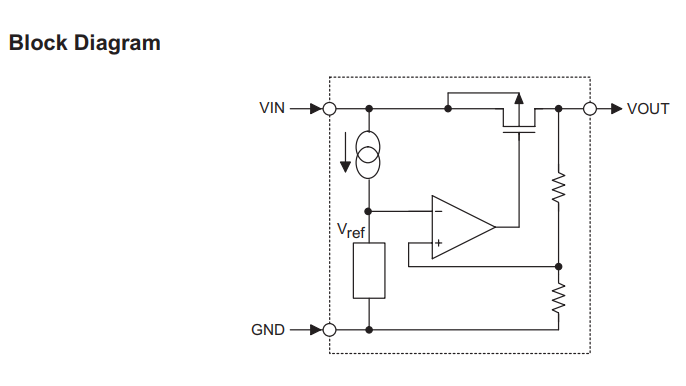
Dưới đây là hình ảnh sơ đồ nguyên lý khối nguồn của mạch:



Hình 2. 3: Sơ đồ nguyên lý khối nguồn

Khối này có nhiệm vụ cung cấp điện áp 3.3V để cho mạch hoạt động ổn định. Điện áp khoảng 5 -> 12 VDC được lấy từ nguồn pin sẽ được đưa qua IC ổn áp 3.3V để cấp nguồn cho vi điều khiển STM32 và Lora hoạt động.

IC HT7133 là IC ổn áp thuộc họ HT71xx cho điện áp đầu ra ổn định ở 3.3V. Dòng điện đầu ra tối đa 30 mA. Điện áp đầu vào tối đa 12V. Tuy nhiên không nên cung cấp điện áp quá gần với điện áp tối đa.



Hình 2. 4: Block diagram của IC HT7133

|  |  |  |
| --- | --- | --- |
| **Tham số** | **Ký hiệu** | **Giá trị** |
| Điện áp ngõ vào | Vin | ~5 VDC |
| Điện áp ngõ ra | VCC | 3.3 VDC |
| Dòng điện rõ ra |  | 250mA ( max ) |
| Tụ ngõ vào | C16,C17 | ,104,106 |
| Tụ ngõ ra | C18,C19,C20 | 470uF/25V , 104,106 |

Bảng 2. 1: Bảng thông số các giá trị linh kiện trong khối nguồn

Dưới đây là hình ảnh loại được sử dụng và thông số kỹ thuật của nó:



Hình 2. 5: Hình ảnh pin cấp nguồn cho mạch

Thông số kỹ thuật:

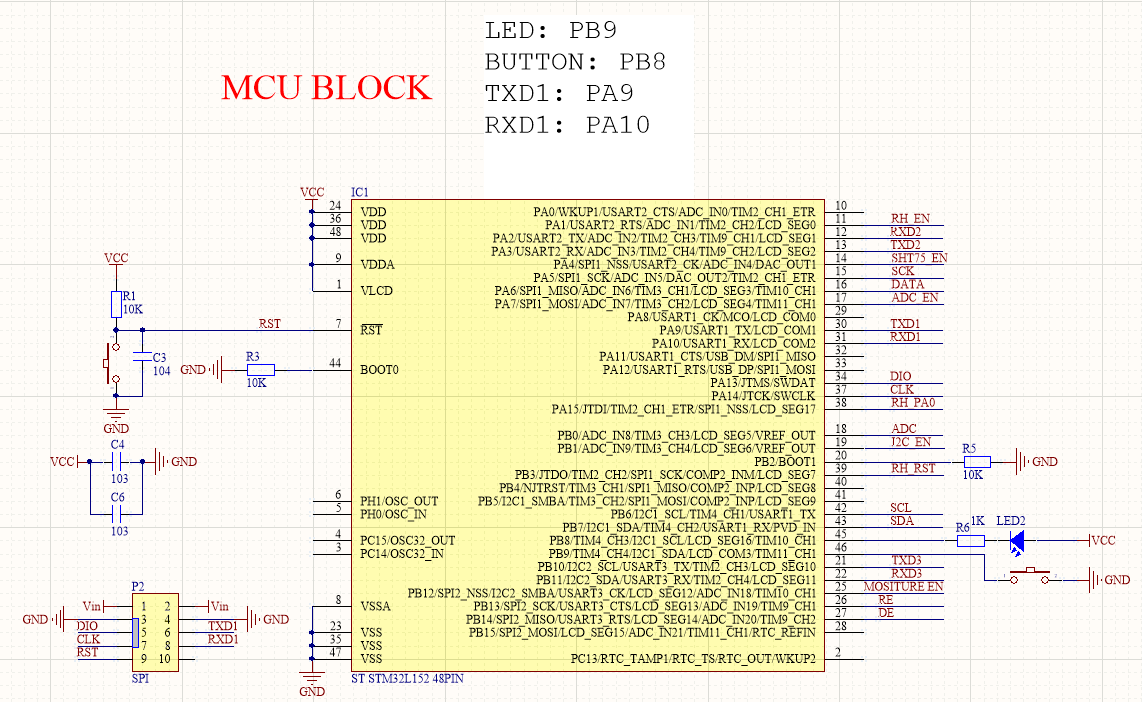
* Kích thước : 33 x 60 mm.
* Thay thế cho : Pin D, Pin R20, LR20, E95.
* Chất liệu : Alkaline.
* Điện áp : 1.5V.
* Model : E95 BP-2.
* Loại pin : Pin dùng 1 lần.

Tuy nhiên khi sử dụng loại pin này cần lưu ý một số điều sau:

* Pin cần được để tránh xa trẻ em , bảo quản ở những nơi thoáng , khô dáo và tránh tiếp xúc trực tiếp với ánh nắng mặt trời.
* Khi thiết bị không sử dụng trong thời gian dài nên tháo pin ra khỏi thiết bị để tránh bị chảy nước làm hỏng thiết bị .
* Nếu thiết bị bị han, dỉ do để nâu ngày hoặc do pin cũ chảy nước cần vệ sinh sạch sẽ trước khi lắp pin mới.
* Pin có các chất liệu kiềm, chì , cacbon rất độc hại khi sử dụng hết cần bỏ vào đúng nơi quy định tránh ảnh hưởng đến sức khỏe con người.

### 2.2.3. Thiết kế nguyên lý khối điều khiển trung tâm

Dưới đây là hình ảnh sơ đồ nguyên lý khối điều khiển trung tâm. Khối điều khiển trung tâm sử dụng vi điều khiển STM32L152. Đây là MCU ARM 32-bit Cortex-M3. Chíp tiết kiệm năng lượng có 128KB Flash, 16KB SRAM, 4KB EEPROM, hỗ trợ LCD, USB, ADC, DAC.



Hình 2. 6: Sơ đồ nguyên lý khối vi điều khiển smt32

Khối này giống như bộ não của toàn bộ bo mạch nó chịu trác nhiệm đọc dữ liệu từ các cảm biến từ các khối cảm biến đo độ ẩm đất, khối đo nhiệt độ độ ẩm khí và gửi tín hiệu đến khối module Lora để yêu cầu gửi dữ liệu qua Lora cho bộ thu.

Vì hộp cảm biến phải hoạt động ngoài đồng ruộng nên phải chạy bằng pin nên để tiết kiệm năng lượng và chi phí thiết kế sử dụng vi điều khiển STM32L152(loại đóng gói 48 PIN). Vi điều khiển này hỗ trợ 7 chế độ hoạt động để sử dụng năng lượng một cách phù hợp nhất.

* Sleep mode:

Ở chế độ này chỉ CPU ngừng hoạt động.Tất cả các ngoại vi khác vẫn hoạt động bình thường và có thể đánh thức CPU khi có một ngắt hoặc sự kiện.

* Low power run mode:

Chế độ này đạt được với bộ dao động RC đa năng bên trong (MSI) được đặt ở mức xung nhịp tối thiểu (65 kHz), thực hiện từ bộ nhớ SRAM hoặc Flash và bộ điều chỉnh bên trong ở chế độ này để giảm thiểu dòng hoạt động của bộ điều chỉnh. Trong chế độ này, tần số xung nhịp và số lượng thiết bị ngoại vi được bật đều bị giới hạn.

* Low power sleep mode:

Chế độ này đạt được bằng cách vào Sleep mode với bộ điều chỉnh điện áp bên trong ở chế độ Nguồn điện thấp để giảm thiểu dòng điện hoạt động của bộ điều chỉnh. Trong chế độ này cả tần số lẫn số lượng ngoại vi cho phép đều bị giới hạn. Khi đánh thức được kích hoạt bởi một sự kiện hoặc ngắt, hệ thống sẽ trở lại run mode với bộ điều chỉnh bật.

* Stop mode với RTC:

Chế độ này đạt mức tiêu thụ năng lượng thấp nhất trong khi vẫn giữ RAM và thanh ghi nội dung và đồng hồ thời gian thực. Tất cả các đồng hồ trong miền VCORE bị dừng, bộ dao động tinh thể PLL, MSI RC, HSI RC và HSE bị vô hiệu hóa. LSE hoặc LSI vẫn đang chạy. Bộ điều chỉnh điện áp ở chế độ năng lượng thấp.

* Stop mode mà không có RTC:

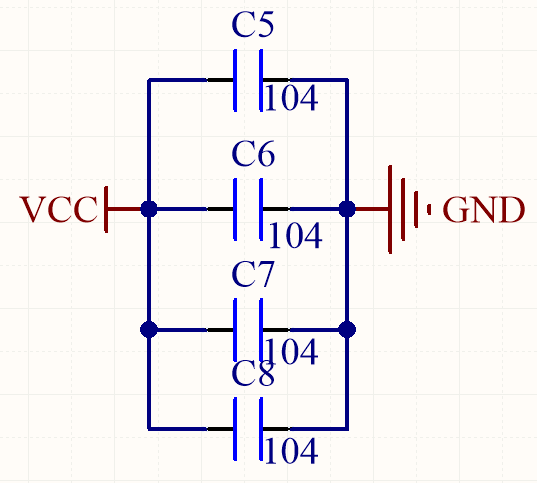
Chế độ này đạt mức tiêu thụ năng lượng thấp nhất trong khi vẫn giữ RAM và thanh ghi nội dung. Tất cả các đồng hồ trong miền VCORE bị dừng, bộ dao động tinh thể PLL, MSI RC, HSI RC và HSE bị vô hiệu hóa. LSE hoặc LSI vẫn đang chạy. Bộ điều chỉnh điện áp ở chế độ năng lượng thấp.

* Standby mode với RTC:

Chế độ này được sử dụng để đạt được mức tiêu thụ điện năng thấp nhất và đồng hồ thời gian thực. Bộ điều chỉnh điện áp bên trong được tắt để toàn bộ miền VCORE bị tắt. Các bộ dao động tinh thể PLL, MSI RC, HSI RC và HSE cũng bị tắt. LSE hoặc LSI vẫn đang chạy. Sau khi vào chế độ Chờ, RAM và nội dung thanh ghi bị mất ngoại trừ các thanh ghi trong mạch chờ (wakeup logic, IWDG, RTC, LSI, LSE Crystal 32K osc, RCC\_CSR).

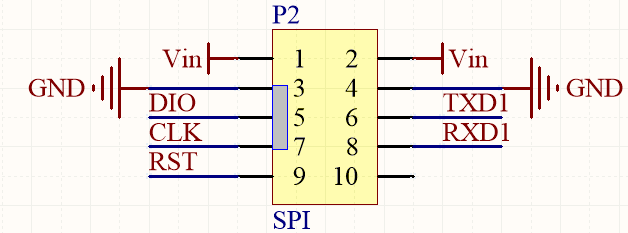
* Standby mode mà không có RTC:

Chế độ này được sử dụng để đạt được mức tiêu thụ điện năng thấp nhất. Bộ điều chỉnh điện áp bên trong được tắt để toàn bộ miền VCORE bị tắt. Các bộ dao động tinh thể PLL, MSI, RC, HSI và LSI RC, HSE và LSE cũng bị tắt. Sau khi vào chế độ Chờ, RAM và nội dung thanh ghi bị mất ngoại trừ các thanh ghi trong mạch chờ (logic đánh thức, IWDG, RTC, LSI, LSE Crystal 32K osc, RCC\_CSR).



Hình 2. 7: Các tụ lọc cho vi điều khiển STM32

Một số tụ lọc để gần nguồn của chíp để nhằm mục đích chống nhiễu cho vi điều khiển. Khi thiết kế cần đặt càng gần chân nguồn của vi điều khiển càng tốt.



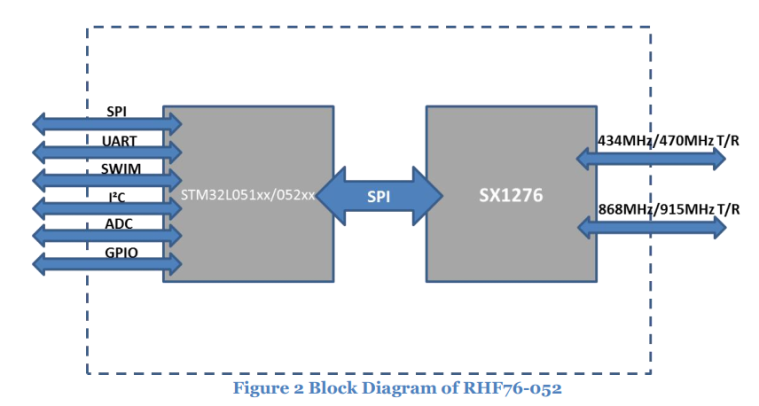
Hình 2. 8: Header để nạp code và gỡ rối

Kết nối với 1 header 5x2 để thuận tiện cho quá trình nạp code cho vi điều khiển và gỡ rối cho UART1.

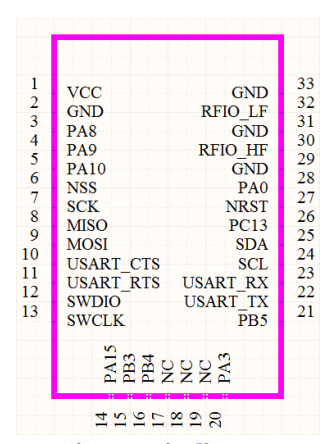
### 2.2.4. Thiết kế nguyên lý khối module Lora Rhf76-052

Chức năng chính của khối này là nhận tín hiệu điều khiển từ khối điều khiển trung tâm và tiến hành hoạt động gửi dữ liệu được gửi từ khối điều khiển trung tâm từ các hộp cảm biến cho bộ thu. Khối này sử dụng một module RHF76-052 và một số linh kiện hỗ trợ khác.

**a) Module RHF76-052**



Hình 2. 9: Block diagram của RHF-76052



Hình 2. 10: Sơ đồ chân của module RHF76-052

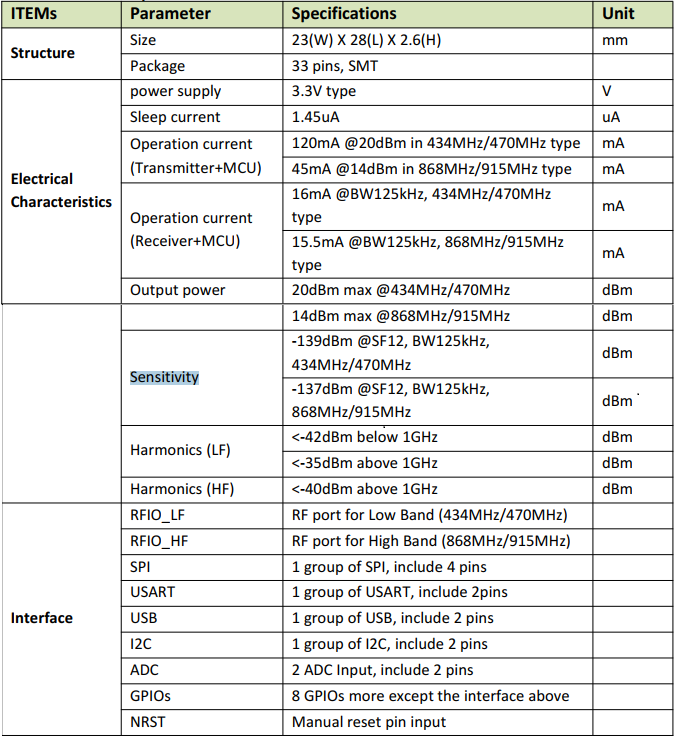
Module RisingHF ™ LoRaWAN ™ RHF76-052 được nhúng với LoRaWAN stack. Người sử dụng có thể sử dụng một host MCU để điều khiển module với tập lệnh AT đơn giản qua giao tiếp UART. Host MCU có thể là bất cứ vi điều khiển nào có uart. Trong đồ án này host MCU là STM32L152.

Phạm vi hoạt động:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ký hiệu | Mô tả | Min | Max | Unit |
| VCC | Nguồn cung cấp | 1.8 | 3.6 | V |
| TOP | Nhiệt độ | -40 | 85 | °**C** |
| POP | RF input level |  | +10 | dBm |

Bảng 2. 2: Phạm vi hoạt động của module RHF76-052

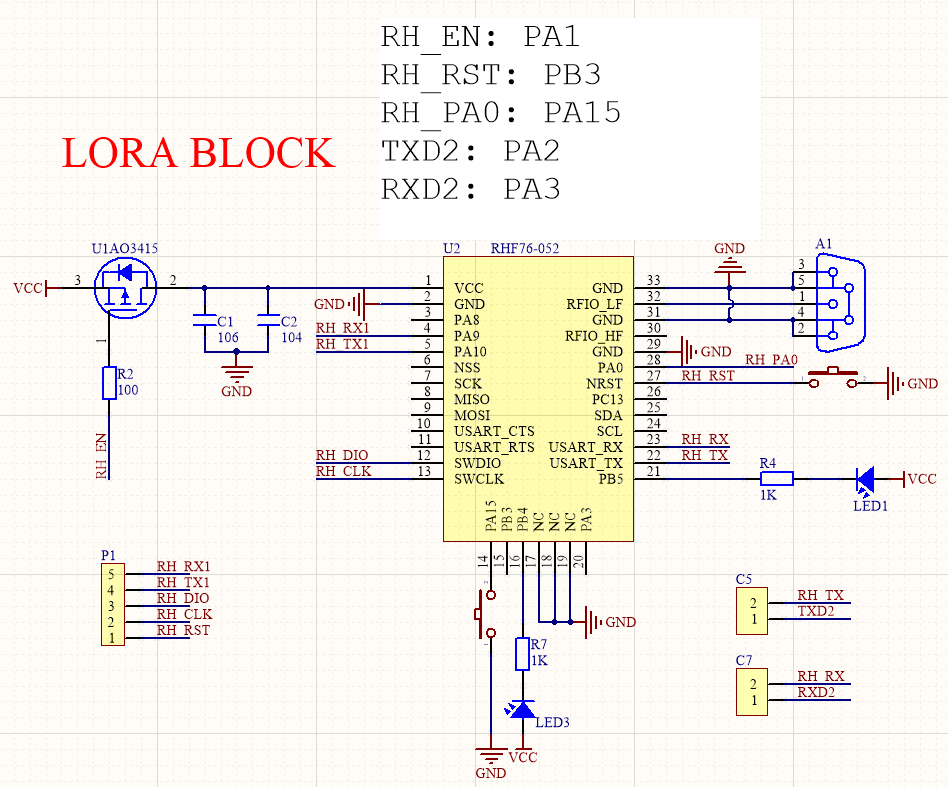
Thông số kỹ thuật của module RHF76-052:

****

Hình 2. 11: Thông số kỹ thuật của module RHF76-052

**b)Sơ đồ nguyên lý khối module Lora:**

Dưới đây là sơ đồ nguyên lý của khối module Lora:



Hình 2. 12: Sơ đồ nguyên lý khối module Lora Rhf76-052

Khối này được thiết kế như sau:

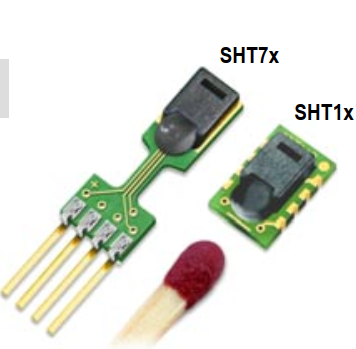
* Sử dụng anten thu ở dải tần số thấp 434MHz/470MHz nối vào chân số 32
* Chân 22 và chân 23 của modem được kết nối với các chân TX,RX của MCU STM32(host) thông qua 2 jump đực.
* Chân 4 và chân 5 của modem được sử dụng để upgare FW dựa trên kết nối UART.
* Chân 14 được sử dụng để kích hoạt chế độ DFU cho upgrage FW.
* Chân 16 được sử dụng để hiển thị trạng thái của xử lý LoraWAN.
* Chân 21 được dành riêng để kết nối LED
* Ngoài ra sử dụng 1 MOSFET để làm chân ENABLE cho module.

### 2.2.5. Thiết kế nguyên lý khối cảm biến nhiệt độ độ ẩm

Khối này có chức năng thu thập nhiệt độ và độ ẩm môi trường chuyển đổi thành tín hiệu để khối điều khiển trung tâm dễ dàng đọc được dữ liệu đó khi nó yêu cầu. Khối này sử dụng linh kiện chính là cảm biến đo nhiệt độ độ ẩm chuyên dụng SHT75.

**a) Cảm biến SHT75**

Để đo được độ ẩm và nhiệt độ của môi trường trong khu vực sử dụng cảm biến nhiệt độ độ ẩm chuyên dụng SHT75 thuộc họ SHT7x.

****

Hình 2. 13: Hình ảnh cảm biến SHT7x,SHT1x

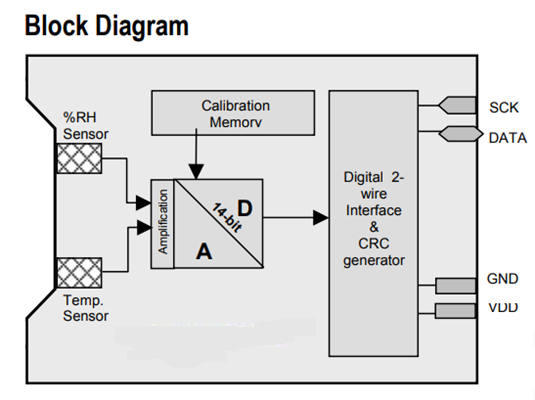
Đây là loại cảm biến chuyên dụng được sản xuất với công nghệ (CMOSens®) để đảm bảo độ tin cậy cao và thời gian hoạt động lâu dài, ngoài ra nó còn có các tính năng nổi bật như sau:

* Có đầu ra số (digital output).
* Ổn định lâu dài.
* Không yêu cầu thêm các thành phần chuyển đổi bên ngoài.
* Tiêu thụ năng lượng cực thấp.
* Thiết kế nhỏ gọn(4 pin).
* Tự động ngắt nguồn.
* Dải đo nhiệt độ -40 đến 120 độ C.
* Độ ẩm: 0 đến 100 %.

Cảm biến này được ứng dụng cho:

* Trong ô tô, máy tự động.
* HVAC.
* Trạm thời tiết.
* Công nghệ tự động.
* Y khoa.
* Máy tạo độ ẩm.

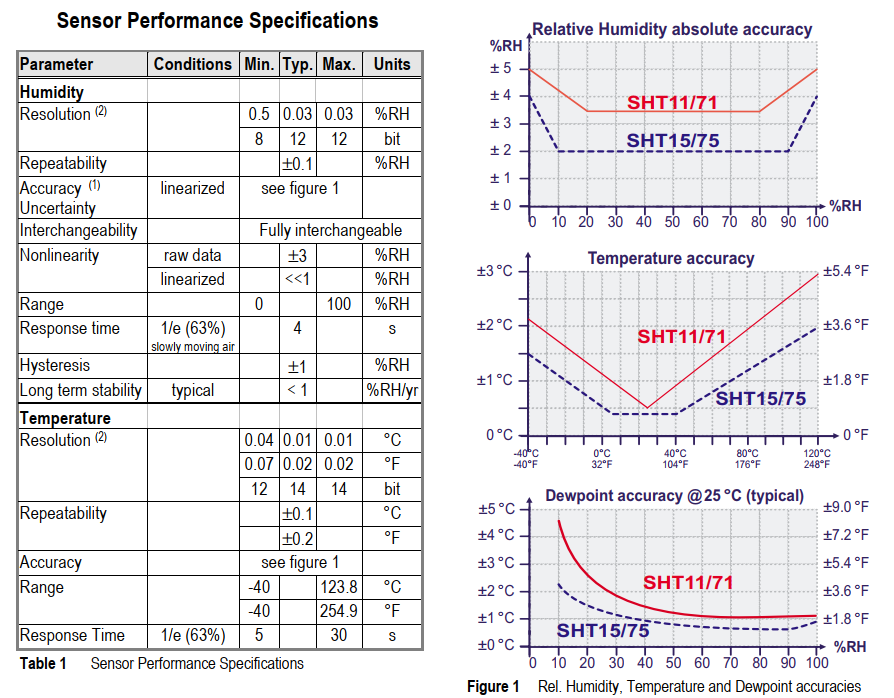
Dưới đây là Block Diagram của nó:



Hình 2. 14: Block Diagram của SHT75

Ở bên trong SHT75 có 2 cảm biến nhiệt độ độ ẩm riêng biệt sau khi đọc được dữ liệu từ môi trường chúng được đưa qua bộ chuyền đổi A/D 14 bít để biến thành tín hiệu số sau đó đưa qua giao diện 2 dây số để giao tiếp với vi điều khiển.

Thông số kỹ thuật hiệu suất của cảm biến:



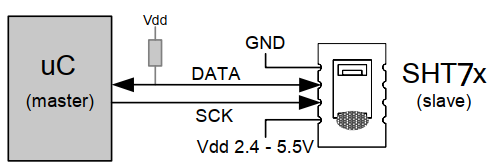
Hình 2. 15: Thông số kỹ thuật hiệu suất của cảm biến

**b) Sơ đồ nguyên lý khối đo nhiệt độ độ ẩm không khí**

Sơ đồ chân SHT75:

|  |  |  |
| --- | --- | --- |
| Chân số | Tên | Mô tả |
| 1 | GND | Đất |
| 2 | DATA | Dữ liệu nối tiếp 2 chiều |
| 3 | SCK | Clock nối tiếp, input |
| 4 | VDD | Nguồn cung cấp 2.4-5.5 V |

Bảng 2. 3: Bảng sơ đồ chân kết nối của SHT75

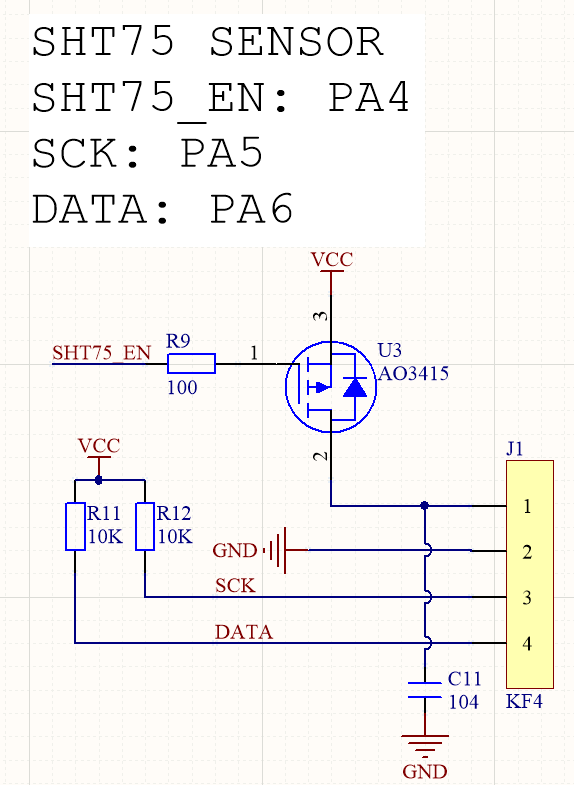


Hình 2. 16: Sơ đồ kết nối MCU và SHT7x

SHT75 giao tiếp với MCU thông qua giao tiếp 2 dây nối tiếp (2 dây 2 chiều) để tối ưu hóa:

* *Serial clock input(SCK):* Chân SCK được sử dụng để đồng bộ giao tiếp giữa MCU và SHT75.
* *Serial data(DATA):* Chân DATA được sử dụng để truyền dữ liệu vào ra cho cảm biến (2 chiều). Dữ liệu thay đổi sau sườn xuống và có giá trị ở cạnh lên của chân SCK. Cần có 1 điện trở pull-up cho chân này.

Dưới đây là hình ảnh sơ đồ nguyên lý của khối này:



Hình 2. 17: Sơ đồ nguyên lý khối đo nhiệt độ độ ẩm không khí

Các chân DATA và chân SCK của cảm biến được kết nối đến 2 chân PA5 và PA6 của vi điều khiển STM32. Ngoài ra thông qua một MOSFET nối với chân PA4 của vi điều khiển để kiểm soát sự bật tắt nguồn của cảm biến.

### 2.2.6. Thiết kế sơ đồ nguyên lý kết nối STM32 và cảm biến độ ẩm đất

Khối này có chức năng chính là thu thập dữ liệu độ ẩm đất thông qua cảm biến độ ẩm đất vào kết hợp với các khối khác để gửi dữ liệu độ ẩm đất đã đo được qua Lora. Cảm biến độ ẩm đất trong khối này có tên là RS-WS-N01-TR. Tín hiệu nó trả về là tín hiệu RS485.

**a) Cảm biến đo độ ẩm đất**

Để đo được độ ẩm đất trong môi trường , trong đề tài sử dụng cảm biến đo độ ẩm đất RS-WS-N01-TRđể đo thông số độ ẩm đất trong khu vực. Dưới đây là hình ảnh cảm biến này.



Hình 2. 18: Cảm biến đo độ ẩm đất

Thông số kỹ thuật của cảm biến:

* Điện áp cung cấp: 5 ~ 30V DC
* Công suất tiêu thụ: 0.4W
* Khoảng độ ẩm đo được: 0 ~ 100% RH
* Độ chính xác: ± 3% RH (mặc định)
* Môi trường làm việc: -20 ℃ ~ 60 ℃,0 ~ 80% RH
* Tín hiệu đầu ra: RS485

Cách sử dụng của cảm biến:

Với cảm biến này có hai cách để đo lường cách thứ nhất có thể cắm cảm phần đầu đo của cảm biến xuống đất, cách thứ hai là chôn toàn bộ phần đầu đo của cảm biến xuống đất với đường kính hố chôn lớn hơn 20 cm.



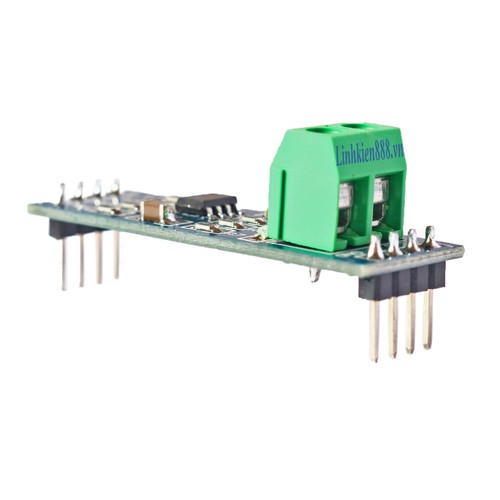
Hình 2. 19: Các cách sử dụng cảm biến đất trong môi trường

Một số chú ý khi sử dụng loại cảm biến này:

* Để dữ liệu đo được chính xác cần cắm sâu tất cả các đầu đo xuống đất.
* Để tránh ánh nắng trực tiếp với ánh sáng mặt trời mạnh gây ra bởi nhiệt độ cảm biến là quá cao.
* Không bẻ cong, tác dụng lực mạnh vào đầu đo cảm biến.
* Lớp bảo vệ cảm biến IP68, cảm biến này hoàn toàn có thể ngâm trong nước.

**b) Sơ đồ nguyên lý khối đo độ ẩm đất**

Như phần trên đã đề cập đầu ra của cảm biến này là tín hiệu RS485. Để vi điều khiển STM32 có thể giao tiếp được với cảm biến này cần sử dụng thêm một module chuyển đổi RS485 to TTL. Dưới đây là hình ảnh của nó.



Hình 2. 20: Module chuyển đổi RS485 to TTL

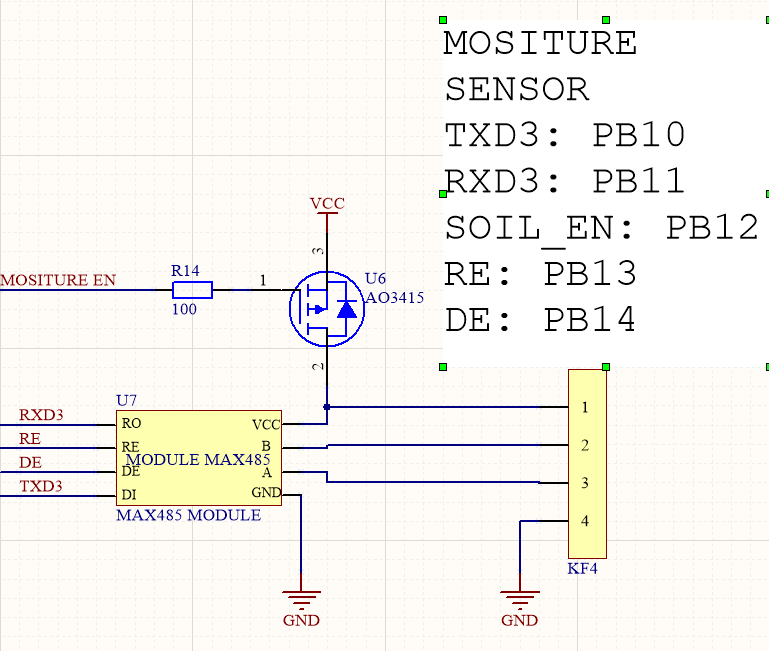
**Thông số kỹ thuật:**

* IC sử dụng: MAX485.
* Điện áp làm việc: 5V (mức TTL).
* Hoạt động hiện tại: 120uA ~ 500uA.
* Công suất hoạt động: 830mW.
* Tốc độ truyền tải tối đa: 2.5Mbps.
* Cho phép lên đến 32 bộ thu phát trên bus đường truyền.
* Khoảng cách truyền tối đa lên tới 1200m.
* Kích thước: 44 x 14mm.

**Các chân của module gồm có:**

* DI- DATA IN (Dữ liệu vào).
* RO - RECEIVE OUT (Tín hiệu nhận đầu ra).
* DE- DATA ENABLE; RE RECEIVE ENABLE (Cho phép dữ liệu vào và cho phép được nhận) 2 chân này thường để nối liền với nhau và kết nối với Pin 2.
* VCC nối với +5v.
* GND nối với GND, GND của 2 arduino có thể dùng cáp để nối hoặc cần được nối đất tại chỗ.
* A nối với A; B nối với B thông qua cặp cáp xoắn.

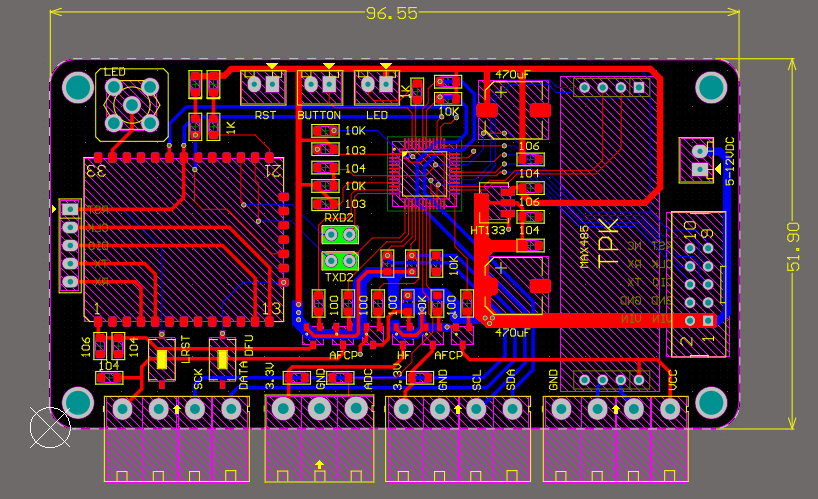
Trong thiết kế các chân RO, DI của module được nối lần lượt tới các chân RX, TX của bộ UART3 của vi điều khiển STM32. Các chân RE, DE được nối lần lượt tới các chân PB13, PB14 của vi điều khiển . Các chân A, B của module được nối tới jump để kết nối tới các chân A, B của cảm biến. Dưới đây là hình ảnh cụ thể.



Hình 2. 21: Sơ đồ nguyên lý khối đo độ ẩm đất

## 2.3. Thiết kế layout cho mạch trong hộp cảm biến

Thông thường PCB được cấu tạo từ các tấm đồng rất mỏng được ép với nhau, xen giữa chúng là các chất nền đóng vai trò là chất cách điện, thường là các sợi thủy tinh được bao bởi nhựa epoxy. Tùy theo mức độ gia nhiệt, độ cứng mà các chất nền này được gọi là PrePreg hoặc Core. Core là chất nền được xử lý dưới nhiệt độ và áp suất cao hơn, nên cứng và cách điện tốt hơn so với PrePreg. Trong thiết kế này để phù hợp với yêu cầu thực hiện thiết kế mạch 2 lớp với kích thước chiều ngang và dọc như hình dưới đây(đơn vị mm, đã ẩn lớp phủ đồng).



Hình 2. 22: PCB toàn bộ bo mạch(đã ẩn toàn bộ lớp phủ đồng)

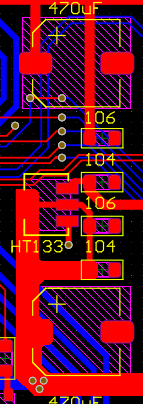
Các thông số của vật liệu tạo nên PCB gồm:

* Hằng số điện môi (Relative Dielectric Constant), được ký hiệu là εr hoặc Dk: Hằng số điện môi càng cao sẽ có một số ưu điểm sau:
  + Khả năng cách ly tốt hơn
  + Các tín hiệu có thể truyền với tốc độ nhanh hơn
  + Trở kháng đường truyền cao hơn với cùng một kích thước, từ đó giúp giảm độ rộng của dây dẫn, tiết kiệm diện tích trong khi layout các mạch có mật độ cao.
  + Các thành phần cảm kháng ký sinh giảm.
* Hệ số suy hao (Loss Tangent), ký hiệu tan(δ) hoặc Df: Đo lường sự suy hao của tín hiệu khi được dẫn truyền trên đường truyền. Hệ số suy hao càng thấp thì càng tốt.
* Kiểu dệt của sợi thủy tinh (Fiberglass Weave Composition): Mật độ các sợi thủy tinh càng dày thì trở kháng của các đường tín hiệu càng đồng đều, càng tốt.
* Hiệu ứng bề mặt (Skin Effect): Tần số tín hiệu càng cao, trở kháng của đường dây dẫn càng lớn, gây cản trở tín hiệu. Vì vậy phải tính toán kích thước dây dẫn phù hợp với tần số làm việc để bảo đảm trở kháng chính xác.

Như vậy, có thể đưa ra kết luận, với tín hiệu có tần số càng cao thì chất lượng PCB càng phải tốt: hằng số điện môi lớn, hệ số suy hao nhỏ, vải sợi thủy tinh trong các chất nền nên có mật độ cao, đồng đều.

### 2.3.1. Thiết kế layout khối nguồn

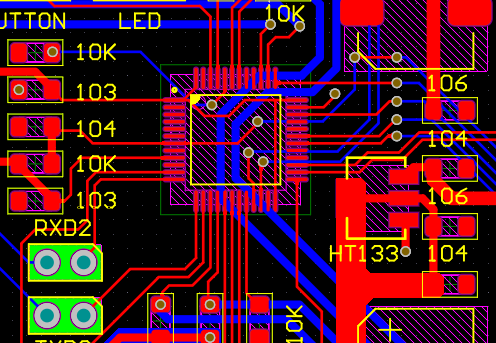
Các linh kiện ở khối nguồn dùng để hạ áp nhằm mục đích đưa nguồn Vin từ pin vào giảm xuống 3.3VDC để cung cấp cho các chíp hoạt động. Các linh kiện trong khối này được sắp xếp gần nhau và được căn chỉnh hợp lý để bo mạch trong thẩm mỹ hơn. Dây VIN được phủ một lớp đồng để giúp chịu dòng cao. Tương tự dây VCC(3.3V) cũng được vẽ to hơn các dây khác.



Hình 2. 23: Hình ảnh layout khối nguồn

### 2.3.3. Thiết kế layout khối điều khiển trung tâm

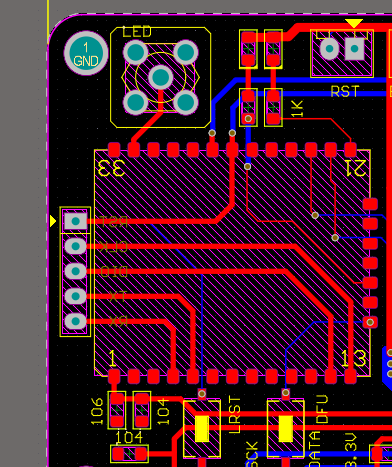
Vi điều khiển STM32 được đặt ở phần trung tâm phía trên của bo mạch cách xa các khối khác. Từ đây kéo ra các bus để kết nối với các linh kiện của các khối khác. Các tụ bypass được bố trí gần với chân nguồn của vi điều khiển nhằm chống nhiễu. sử dụng các via để kết nối giữa hai layer.



Hình 2. 24: Hình ảnh layout khối điều khiển trung tâm

### 2.3.4. Thiết kế layout khối module Lora

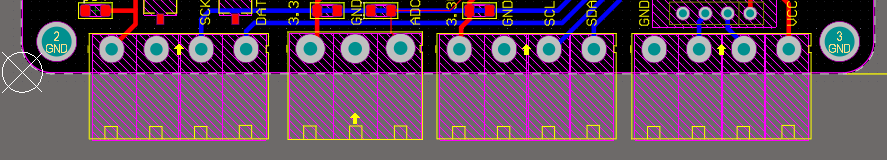
Mô-đun lora được bố trí ở bên trái mạch đặt đế anten của mo-đun cách ở góc mạch tránh xa các linh kiện khác. Các nút bấm reset và các đèn led được sắp xếp ở góc mạch nơi dễ dàng nhìn thấy. Ngoài ra còn có một jump đực để kết nối một số chân của module. Hai tụ lọc bypass được đặt gần chân VCC của module. Dưới đây là hình ảnh layout:



Hình 2. 25: Hình ảnh layout khối module Lora

### 2.3.5. Thiết kế layout các khối cảm biến

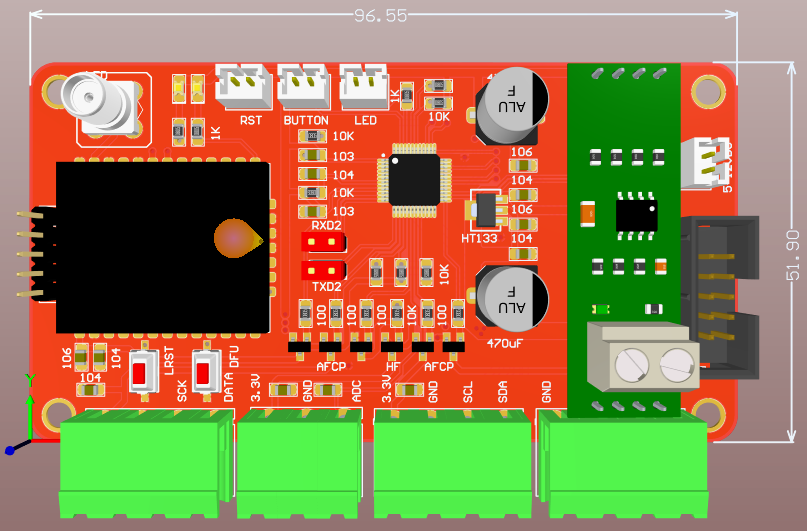
Để thuận tiện cho việc kết nối các cảm biến ra ngoài, bo mạch bố trí các cầu đấu ở bên dưới . Để có thể dễ dàng cắm các cảm biến vào khu vực đó. Về cảm biến đo độ ẩm đất do cần thêm một module chuyển đổi nên module đó được thiết kế cắm vào jump cái trên bo mạch để có thể thuận tiện cho việc sửa chữa bảo trì sau này. Dưới đây là hình ảnh:



Hình 2. 26:Hình ảnh layout các khối cảm biến

### 2.3.6. Hình ảnh 3D trên phần mềm Altilum Designer của mạch

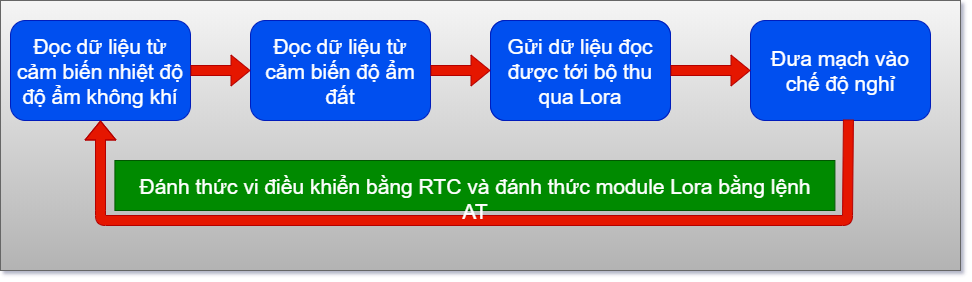
Dưới đây là một số hình ảnh 3D của bo mạch sau khi thiết kế trên phần mềm Altilum Designer:



Hình 2. 27: Hình ảnh 3D của bo mạch trên phần mềm

# CHƯƠNG III: XÂY DỰNG MODULE PHẦN MỀM ĐỌC DỮ LIỆU TỪ CÁC CẢM BIẾN VÀ TRUYỀN TẢI DỮ LIỆU THÔNG QUA LORA

Sau khi đã hoàn thành viết thiết kế mạch nguyên lý và mạch in trong chương này tập trung lập trình cho vi điều khiển STM32 trên mạch để mạch hoạt động đúng theo yêu cầu đã đặt ra là thu thập dữ liệu từ 2 con cảm biến gửi dữ liệu tới bộ thu và tiêu thụ mức năng lượng tối thiểu. Dưới đây là mô hình chu trình hoạt động của mạch.

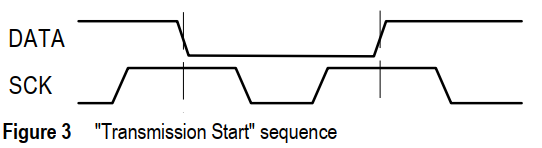


Dưới đây sẽ trình bày rõ cách xây dựng module phần mềm cho mỗi bước trong chu trình trên.

## 3.1. Xây dựng module phần mềm đọc dữ liệu từ cảm biến nhiệt độ độ ẩm không khí

### 3.1.1. Quá trình giao tiếp của MCU với SHT75

Để bắt đầu quá trình truyền nhận, MCU cần phải gửi một tín hiệu bắt đầu truyền(Transmission Start) cho SHT75. Tín hiệu này được tạo ra bằng cách kéo chân DATA xuống mức thấp trong khi chân SCK vẫn ở mức cao giữ nó ở mức thấp cho đến khi mức cao của chu kỳ tiếp theo của chân SCK.



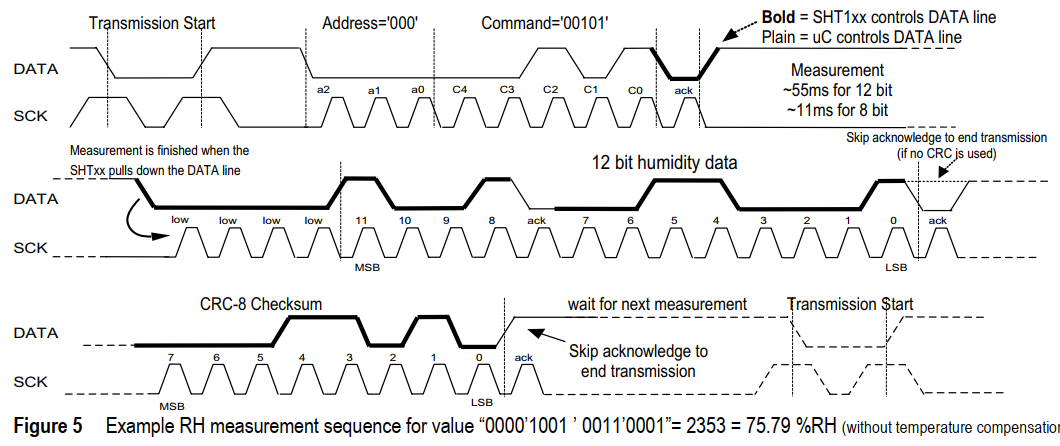
Hình 3. 1: Tín hiệu bắt đầu truyền nhận của MCU và SHT75

Tiếp theo MCU sẽ gửi 3 bít địa chỉ(000) và 5 bít lệnh. SHT75 cho biết việc nhận lệnh thích hợp bằng cách kéo pin DATA xuống thấp (bit ACK) sau sườn xuống SCK thứ 8. Chân DATA sẽ lại được kéo lên mức cao sau sườn âm thứ 9 của SCK. Dưới đây là bảng mã lệnh của SHT75.

|  |  |
| --- | --- |
| Lệnh | Mã lệnh |
| Reserved | 0000x |
| Đo nhiệt độ | 00011 |
| Đo độ ẩm | 00101 |
| Đọc thanh ghi trạng thái | 00111 |
| Ghi vào thanh ghi trạng thái | 00110 |

Bảng 3. 1: Lệnh và mã lệnh cho SHT75

Tiếp theo là quá trình đọc dữ liệu từ cảm biến. Sau khi gửi lệnh đo (‘00000101, đối với RH, 00 00000011 triệt cho Nhiệt độ), MCU phải chờ cho phép đo hoàn thành. Quá trình này mất khoảng 11/55/210 ms tùy theo phép đo là 8/12/14 bit. Để báo hiệu việc hoàn thành phép đo, SHT75 kéo DATA xuống mức thấp(ACK). MCU phải chờ tín hiệu sẵn sàng dữ liệu này trước khi bắt đầu bật lại SCK.



Hình 3. 2: Ví dụ về đo độ ẩm bằng SHT75

Quá trình truyền được đảm bảo bởi 8 bít checksum nó dùng để đảm bảo rằng bất kỳ lỗi nào trên đường truyền cũng được phát hiện và loại bỏ.

Ngoài ra SHT75 còn có 1 thanh ghi trạng thái 8 bit để cấu hình 1 vài chức năng nâng cao như cấu hình độ phân giải cho bộ A/D.

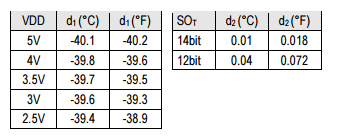
Chuyển đổi dữ liệu 12 bit ra giá trị nhiệt độ và độ ẩm:

Với giá trị nhiệt độ được chuyển đổi theo công thức:

T = d1 + d2.SOT

T là giá trị nhiệt độ

Giá trị d1 ,d2 tham khảo bảng sau:



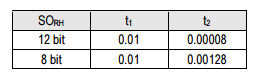
Hình 3. 3: Thông số d1,d2 để tính nhiệt độ khi đọc từ cảm biến SHT75

SOT là giá trị 12 bit dữ liệu nhiệt độ đọc về . Với độ ẩm độ ẩm được chuyển đổi theo công thức:

HM = (T - 25) \* (t1+t2\*SORH) + RHliner

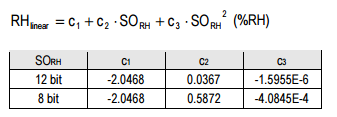
T là nhiệt độ đo được tính theo độ C.

Giá trị t1,t2 các bạn xem trong bảng



Hình 3. 4: Giá trị t1, t2

SORH là giá trị 12 bit độ ẩm đọc được. RHliner là giá trị được tính theo công thức :



Hình 3. 5: Công thức tính RHliner

Giả sử đọc được 12 bit độ ẩm là 0100 0011 0001 quy đổi nó ra là bằng 1073 vậy SORH = 1073 vậy ta có

RHlinear = -2.0468 + 0.0367\*1073 + 9.633.10^(-7)\*1073^2 = 38.44.

Giả sử nhiệt độ đo được bằng SHT75 lúc đó là 30 độ C thì :

HM = (30-25)\*(0.01+0x00008\*1073)+38.44=38.9%.

### 3.1.2. Xây dựng các hàm đọc dữ liệu

Dưới đây là prototype của các hàm xây dựng để đọc dữ liệu từ SHT75:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Definitions function \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

uint8\_t meas(uint8\_t cmd, uint16\_t \*result, uint8\_t block);

uint8\_t measure(float \*temp, float \*humi, float \*dew);

uint8\_t measRdy(void);

uint8\_t getResult(uint16\_t \*result);

uint8\_t writeSR(uint8\_t value);

uint8\_t readSR(uint8\_t \*result);

uint8\_t reset(void);

uint8\_t putByte(uint8\_t value);

uint8\_t getByte(uint8\_t ack);

void startTransmission(void);

void resetConnection(void);

float SHT75\_calcTemp(uint16\_t rawData);

float SHT75\_calcHumi(uint16\_t rawData, float temp);

float SHT75\_calcDewpoint(float humi, float temp);

void calcCRC(uint8\_t value, uint8\_t \*crc);

uint8\_t bitrev(uint8\_t value);

void GPIO\_Config\_Output(uint16\_t);

void GPIO\_Config\_Input(uint16\_t);

void SHT75\_Setup(void);

Trong đó các hàm dùng để cài đặt khởi tạo các chân giao tiếp với cảm biến:

* void GPIO\_Config\_Output(uint16\_t);
* void GPIO\_Config\_Input(uint16\_t);
* void SHT75\_Setup(void);

Các hàm làm nhiệm vụ giao tiếp trực tiếp với cảm biến như đọc một byte nhận 1 một byte từ cảm biến, tạo tín hiệu start, tạo tín hiệu reset:

* uint8\_t writeSR(uint8\_t value);
* uint8\_t readSR(uint8\_t \*result);
* uint8\_t reset(void);
* uint8\_t putByte(uint8\_t value);
* uint8\_t getByte(uint8\_t ack);
* void startTransmission(void);
* void resetConnection(void);

Các hàm tính toán hiệu chỉnh kết quả:

* float SHT75\_calcTemp(uint16\_t rawData);
* float SHT75\_calcHumi(uint16\_t rawData, float temp);
* float SHT75\_calcDewpoint(float humi, float temp);
* void calcCRC(uint8\_t value, uint8\_t \*crc);
* uint8\_t bitrev(uint8\_t value);

Các hàm để đo lường kết quả :

* uint8\_t meas(uint8\_t cmd, uint16\_t \*result, uint8\_t block);
* uint8\_t measure(float \*temp, float \*humi, float \*dew);
* uint8\_t measRdy(void);
* uint8\_t getResult(uint16\_t \*result);

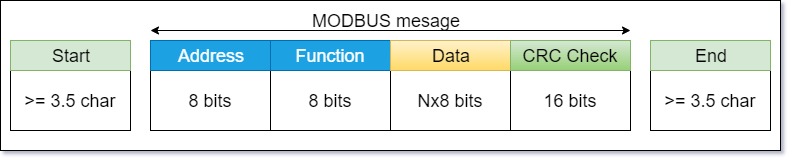
Chi tiết về các hàm sẽ trình bày trong phần phụ lục ????

## 3.2. Xây dựng module phần mềm đọc dữ liệu từ cảm biến độ ẩm đất

### 3.2.1. Modbus-RTU

Đầu ra của cảm biến độ ẩm đất là tín hiệu RS485, và nó tuân theo quy luật truyền thông của Modbus RTU. Giao thức Modbus RTU là một giao thức mở, sử dụng đường truyền vật lý RS-232 hoặc RS485 và mô hình dạng Master-Slave. Đây là một giao thức được sử dụng rộng rãi trong nhiều lĩnh vực như BMS (Building Management Systems), tự động hóa, công nghiệp, điện lực...

Một bản tin Modbus RTU bao gồm: 1 byte địa chỉ  -  1 byte mã hàm - n byte dữ liệu - 2 byte CRC như hình ở dưới:



Hình 3. 6: Cấu trúc một bản tin Modbus RTU

Chức năng và vai trò cụ thể như sau:

* **Byte địa chỉ**: xác định thiết bị mang địa chỉ được nhận dữ liệu (đối với Slave) hoặc dữ liệu nhận được từ địa chỉ nào (đối với Master). Địa chỉ này được quy định từ 0 – 254.
* **Byte mã hàm**: được quy định từ Master, xác định yêu cầu dữ liệu từ thiết bị Slave.
* **Byte dữ liệu**: xác định dữ liệu trao đổi giữa Master và Slave.

Đọc dữ liệu:

* + Master:  2 byte địa chỉ dữ liệu - 2 byte độ dài dữ liệu.
* + Slave: 2 byte địa chỉ dữ liệu - 2 byte độ dài dữ liệu - n byte dữ liệu đọc được.
* Ghi dữ liệu:
* +Master: 2 byte địa chỉ dữ liệu - 2 byte độ dài dữ liệu - n byte dữ liệu cần ghi.
* + Slave: 2 byte địa chỉ dữ liệu - 2 byte độ dài dữ liệu.
* **Byte CRC**: 2 byte kiểm tra lỗi của hàm truyền. cách tính giá trị của Byte CRC 16Bit.

Về RS485:

Khi một mạng cần phải chuyển các khối nhỏ thông tin trên một khoảng cách dài, RS-485 thường là chuẩn giao tiếp được lựa chọn. Các nút mạng có thể là máy tính cá nhân, vi điều khiển, hoặc bất kỳ thiết bị có khả năng truyền thông nối tiếp không đồng bộ. So với Ethernet và giao diện mạng khác, phần cứng và giao thức yêu cầu của RS-485 đơn giản hơn và rẻ hơn **.**

  Những đặc điểm nổi trội của RS485 là nó có thể hỗ trợ một mạng lên tới 32 trạm thu phát trên cùng một đường truyền, tốc độ baud có thể  lên tới 115.200 cho một  khoảng cách là 4000feet (1200m).

Với kiểu truyền cân bằng và các dây được xoắn lại với nhau nên khi nhiễu xảy ra  ở dây này thì cũng xảy ra ở dây kia, tức là hai dây cùng nhiễu giống nhau. Điều này làm cho điện áp sai biệt giữa hai dây thay đổi không đáng kể nên tại nơi thu vẫn nhận được tín hiệu đúng nhờ tính năng đặc biệt của bộ thu đã loại bỏ nhiễu. Liên kết RS485 được sử dụng rất rộng rãi trong công nghiệp, nơi mà môi trường nhiễu khá cao và sự  tin tưởng vào tính  ổn định của hệ  thống là điều quan trọng. Bên cạnh đó khả năng truyền thông qua khoảng cách xa ở tốc độ cao cũng rất được quan tâm, đặc biệt là tại những nơi mà có nhiều trạm giao tiếp được trải ra trên diện rộng.

**3.2.2. Xây dựng các hàm đọc dữ liệu**

Dưới đây là prototype của các hàm được xây dựng để đo độ ẩm đất:

#ifndef \_\_SOIL\_MOISTURE\_H\_\_

#define \_\_SOIL\_MOISTURE\_H\_\_

#include "stm32l1xx.h"

#include "stm32l1xx\_rcc.h"

#include "stm32l1xx\_gpio.h"

#include "stm32l1xx\_usart.h"

#include "misc.h"

#include "stm32l1xx\_exti.h"

#include "stm32l1xx\_syscfg.h"

#include "delay.h"

#include "usart.h"

#define RE GPIO\_Pin\_13 //PB13

#define DE GPIO\_Pin\_14 //PB14

void Soil\_init(void);

void Soil\_cmd(void);

#endif

Hàm Soil\_init(); dùng để khởi tạo về các chân các thành phần kết nối tới cảm biến.

Hàm Soil\_cmd(); chứa các lệnh yêu cần lấy dữ liệu cho một lần đo với cảm biến.

Chi tiết về các hàm sẽ được trình bày trong phần Phụ lục.

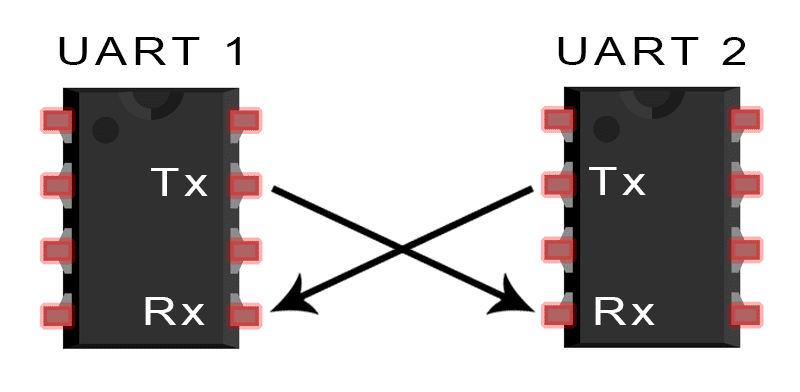
## 3.3. Xây dựng module phần mềm truyền dữ liệu cho bộ thu thông qua Lora

### 3.3.1. Giao tiếp UART

Thuật ngữ UART trong tiếng anh là viết tắt của cụm từ: Universal Asynchronous serial Reveiver and Transmitter, nghĩa là bộ truyền nhận nối tiếp không đồng bộ.

Trong giao tiếp UART cả 2 thiết bị kết nối trực tiếp với nhau. UART truyền chuyển đổi dữ liệu song song từ thiết bị điều khiển như CPU ​​thành dạng nối tiếp, truyền nó nối tiếp đến UART nhận, sau đó chuyển đổi dữ liệu nối tiếp thành dữ liệu song song cho thiết bị nhận.Do đó chỉ cần đường dây kết nối để truyền và nhận dữ liệu giữa 2 thiết bị có hỗ trợ UART. Dữ liệu được truyền từ chân TX của thiết bị truyền sang chân RX của thiết bị nhận. UART truyền dữ liệu không đồng bộ,điều đó có nghĩa là không cần một chân clock để đồng bộ hóa giữa 2 thiết bị. Thay vì tín hiệu clock, UART truyền thêm các bit start và stop cho gói dữ liệu được truyền. Các bit này xác định điểm bắt đầu và kết thúc của gói dữ liệu để UART nhận biết khi nào bắt đầu đọc các bit.

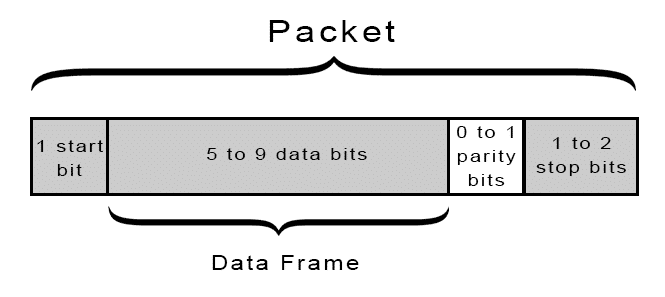
Khi thiết bị nhận phát hiện bit start, nó bắt đầu đọc các bit đến ở một tần số cụ thể được gọi là tốc độ baund. Tốc độ truyền là thước đo tốc độ truyền dữ liệu, được biểu thị bằng bit trên giây (bps). Cả hai thiết bị UART phải hoạt động ở cùng tốc độ baund. Cả hai UART cũng phải được cấu hình để truyền và nhận cùng một cấu trúc gói dữ liệu.



Hình 3. 7: Hai thiết bị giao tiếp UART

Start bit: Đường truyền dữ liệu UART thường được giữ ở mức điện áp cao khi nó không truyền dữ liệu. Để bắt đầu truyền dữ liệu, UART truyền sẽ kéo đường truyền từ cao xuống thấp trong một chu kỳ xung nhịp. Khi UART nhận phát hiện sự chuyển đổi điện áp cao sang thấp, nó bắt đầu đọc các bit trong khung dữ liệu ở tần số của tốc độ truyền.

Data Frame (khung truyền): do truyền thông nối tiếp mà nhất là nối tiếp không đồng bộ rất dễ mất hoặc sai lệch dữ liệu, quá trình truyền thông theo kiểu này phải tuân theo một số quy cách nhất định. Bên cạnh tốc độ baud, khung truyền là một yếu tốc quan trọng tạo nên sự thành công khi truyền và nhận. Khung truyền bao gồm các quy định về số bit trong mỗi lần truyền, các bit “báo” như bit Start và bit Stop, các bit kiểm tra như Parity, ngoài ra số lượng các bit trong một data  cũng được quy định bởi khung truyền. Nó có thể dài 5 bit đến 8 bit nếu sử dụng một bit chẵn lẻ. Nếu không có bit chẵn lẻ nào được sử dụng, khung dữ liệu có thể dài 9 bit. Trong hầu hết các trường hợp, dữ liệu được gửi với bit ít thấp nhất trước tiên.



Hình 3. 8: - Khung truyền dữ liệu trong giao tiếp UART

  Parity bit:Parity là bit dùng kiểm tra dữ liệu truyền đúng không (một cách tương đối). Có 2 loại parity là parity chẵn (even parity) và parity lẻ (odd parity). Bởi vì Các bit có thể được thay đổi bằng bức xạ điện từ, tốc độ truyền không khớp hoặc truyền dữ liệu đường dài. Sau khi UART nhận đọc khung dữ liệu, nó đếm số bit có giá trị là 1 và kiểm tra xem tổng số là số chẵn hay số lẻ. Nếu bit chẵn lẻ là 0 (chẵn lẻ), thì số bít 1 trong khung dữ liệu sẽ có tổng số chẵn. Nếu bit chẵn lẻ là 1 (chẵn lẻ), thì số bít 1 trong khung dữ liệu sẽ tổng thành một số lẻ. Khi bit chẵn lẻ khớp với dữ liệu, UART biết rằng việc truyền không có lỗi. Nhưng nếu bit chẵn lẻ là 0 và tổng là số lẻ; hoặc bit chẵn lẻ là 1 và tổng số chẵn, UART biết rằng các bit trong khung dữ liệu đã thay đổi.

Stop bits: Để báo hiệu sự kết thúc của gói dữ liệu, UART gửi sẽ điều khiển đường truyền dữ liệu từ mức thấp đến mức cao trong ít nhất hai bit.

### 3.3.2. Lập trình gửi dữ liệu qua Lora

Dưới đây là một số quy ước , ký hiệu của tập lệnh AT của module:

Quy ước:

* Tất cả các lệnh đều có phản hồi
* Độ dài lệnh không bao giờ vượt quá 528 ký tư
* Một lệnh AT có giá trị phải két thúc với ‘\n’,’r\n’
* Nếu tính năng timeout được bật, kết thúc ‘\n’ sẽ không bắt buộc
* <LF> có nghĩa là ký tự dòng mới. <CR> có nghĩa là vận chuyển trở lại
* UART được được cấu hình “9600,8,n,1” (8 bits data, no parity,1 bít stop)

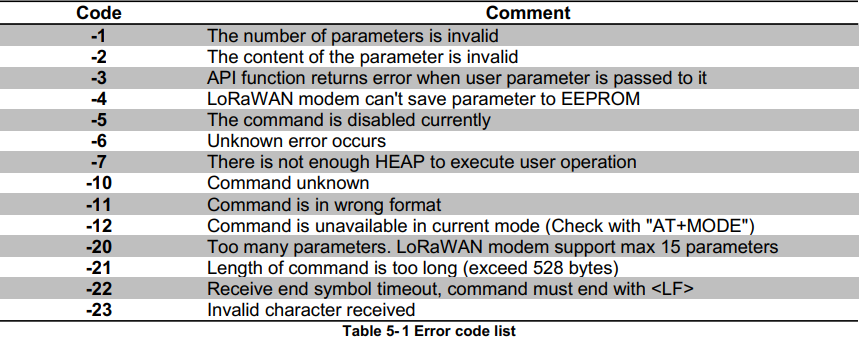
Ký hiệu

* **= ==>** cài đặt giá trị cho lệnh
* **?**  ==> Truy vấn
* **:** ==> bắt đầu một danh sách tham số đầu vào
* **+ ==>**  Tiền tố của lệnh
* **, ==>** Dấu phân cách các tham số
* Space **==>** Ký tự trống,có thể sự dụng để format lệnh

Định dạng(Format):

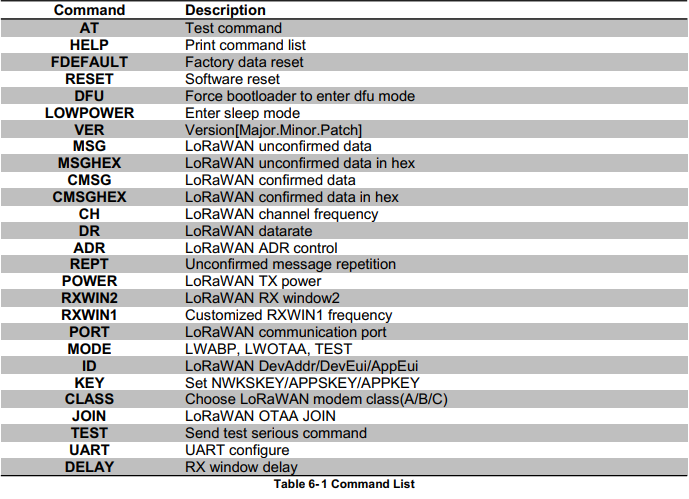
* Truy vấn: sử dụng lệnh truy vấn để kiểm tra cấu hình modem LoRaWAN,cũng như cấu hình kênh,trạng thái ADR, nguồn TX,v..v..
  + AT+COMMAND
  + AT+COMMAND?
  + AT+COMMAND=?
* Cấu hình/Điều khiển: sử dụng lệnh cấu hình/ điều khiển để cài đặt cấu hình mới hoặc điều khiển
  + AT+COMMAND=DATA
* Return: Trả về dữ liệu với định dạng như sau "CMD: RETURN DATA"
  + COMMAND: “RETURN DATA”

Mã lỗi(Error): Đây là bảng mã lỗi của module khi có lỗi có thể kiểm tra lỗi tương ứng ở bảng này.

****

Hình 3. 9: Mã lỗi và mô tả các lỗi của RHf76-052

Đây là bảng mô tả các lệnh của tập lệnh AT:

****

Hình 3. 10: Mã lệnh AT cho mo-đun Rhf76-052

Để gửi dữ liệu, trước hết dữ liệu được định dạng dưới dạng chuỗi rồi chỉ cần dùng lệnh gửi trong tập lệnh để gửi đi là hoàn toàn có thể gửi lệnh đi một cách dễ dàng. Bên dưới là đoạn code đo dữ liệu rồi gửi dữ liệu đi, dữ liệu được định dạng theo format như sau G04\_24\_17\_22\_12 :

Tương ứng như sau:

* G04: Tên của hộp cảm biến
* 24: Giá trị độ ẩm đất
* 17 : Giá trị nhiệt độ không khí
* 22: Giá trị độ ẩm không khí
* 12: Giá trị điểm sương

while(1){

SHT75\_measTemp(&rawData);

temperature = SHT75\_calcTemp(rawData);

SHT75\_measHumi(&rawData);

humidity = SHT75\_calcHumi(rawData, temperature);

dewpoint = SHT75\_calcDewpoint(humidity, temperature);

Soil\_cmd();

delay\_ms(1000);

sprintf(value,"%X%X",array[3],array[4]);

int soil = 0;

soil = (int)strtol(value, NULL, 16);

sprintf(TX\_buf,"%s\_%d\_%d\_%d\_%d\n", UID\_TX,(int)(soil/100),(int)temperature,(int)humidity,(int)dewpoint);

if((int)(soil/100)>=0 && (int)(soil/100) <=100){

lora\_enterTestMode();

printf("AT+TEST=TXLRSTR,%s\r\n",TX\_buf);

}

delay\_ms(2000);

GPIO\_ToggleBits(GPIOB,LED);

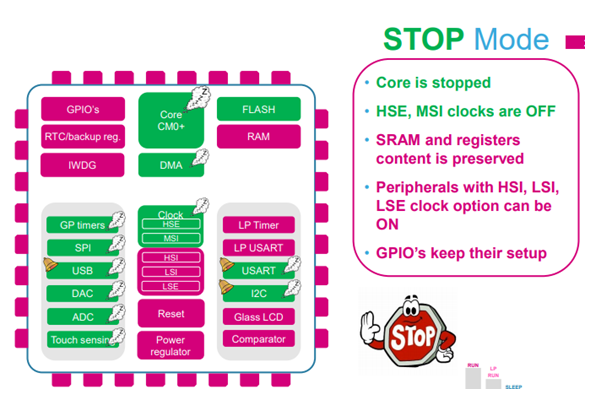
printf("AT\r\n");

}

}

## 3.4. Xây dựng module phần mềm đưa mạch vào chế độ nghỉ

Để có thể hoạt động ngoài lâu dài và có thể tiết kiệm năng lượng một cách hiệu quả nhất có thể, ngoài việc thiết kế cần có được tối ưu dùng nguồn pin chất lượng tốt, cần có phải lập trình để đưa vi điều khiển vào chế độ ngủ có tiêu thụ điện năng hợp lý nhất có thể và đồng thời cũng phải đưa module Lora vào chế độ ngủ của nó. Sau khi đọc cảm biến, gửi dữ liệu cảm biến qua Lora ngay sau đó cảm vi điều khiển trước tiên sẽ gửi lệnh để đưa module Lora vào chế độ Low-power, sau đó tự nó cũng sẽ được chuyển vào chế độ tiết kiệm năng lượng STOP mode và sẽ được đánh thức bằng RTC sau khoảng 5-10 phút sau khoảng thời gian này nó sẽ thức dậy và tiếp tục đo dữ liệu rồi lại gửi đi cứ như vậy. Module cũng sẽ được đánh thức khi vi điều khiển gửi cho nó 1 lệnh AT.



Hình 3. 11: Khi STM32 hoạt động ở chế độ STOP mode

Chế độ này đạt mức tiêu thụ năng lượng thấp nhất trong khi vẫn giữ RAM và thanh ghi nội dung và đồng hồ thời gian thực (RTC). Tất cả các đồng hồ trong miền VCORE bị dừng, bộ dao động tinh thể PLL, MSI RC, HSI RC và HSE bị vô hiệu hóa. LSE hoặc LSI vẫn đang chạy. Bộ điều chỉnh điện áp ở chế độ năng lượng thấp. Dưới đây là hai hàm được viết để đưa vi điều khiển vào hoạt động ở chế độ stop mode:

void PWR\_Init(void){

/\* Enable Clocks \*/

RCC\_APB1PeriphClockCmd(RCC\_APB1Periph\_PWR, ENABLE);

while(PWR\_GetFlagStatus(PWR\_FLAG\_VOS) == SET);

/\*Vcore = 1.8V\*/

PWR\_VoltageScalingConfig(PWR\_VoltageScaling\_Range2);

/\* Disable prefetch buffer \*/

FLASH->ACR &= ~FLASH\_ACR\_PRFTEN;

/\* Enable flash instruction cache\*/

FLASH->ACR |= (0x01<<3);

FLASH->ACR |= (0x01<<4);

/\*Turn off ADC\*/

ADC1->CR2 |= ~(0x01<<0);

/\*Turn off Temperature\*/

ADC->CCR &= (uint32\_t)(~ADC\_CCR\_TSVREFE);

}

void PWR\_Lowpower(void){

/\* Prepare to enter stop mode \*/

PWR->CR |= PWR\_CR\_CWUF; // clear the WUF flag after 2 clock cycles

PWR->CR &= ~( PWR\_CR\_PDDS ); // Enter stop mode when the CPU enters deepsleep

PWR->CR |= (0x01<<9);// disable interrupt VRIF

PWR\_UltraLowPowerCmd(ENABLE);

PWR->CR &= 0xFFFFFFEF;// disable interrupt VRIF

PWR\_FastWakeUpCmd(ENABLE);

SCB->SCR |= SCB\_SCR\_SLEEPDEEP\_Msk; // low-power mode = stop mode

PWR\_EnterSTOPMode(PWR\_Regulator\_LowPower, PWR\_STOPEntry\_WFI);

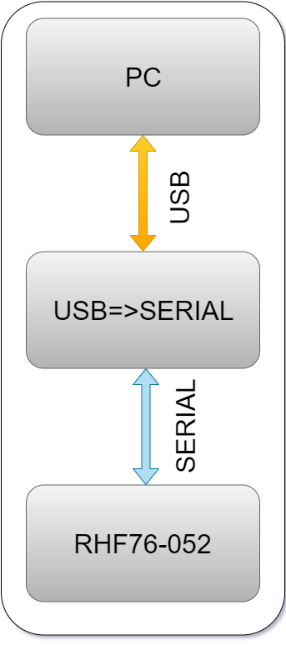
}

# CHƯƠNG IV: THIẾT KẾ BỘ THU NHẬN DỮ LIỆU QUA LORA VÀ XÂY DỰNG GIAO DIỆN THEO DÕI THÔNG SỐ NHẬN ĐƯỢC TỪ BỘ THU

## 4.1. Thiết kế bộ thu nhận dữ liệu qua Lora

### 4.1.1. Sơ đồ khối

Hình dưới đây là sơ đồ khối phần cứng để kết hợp với máy tính để thu dữ liệu từ các hộp cảm biến (Lora Node). Do trên một số máy tính Laptop hoặc máy hiện giờ không còn tích hợp cổng COM như trước nên sử dụng USB to COM để có thể sử dụng trên nhiều loại máy tính khác nhau.

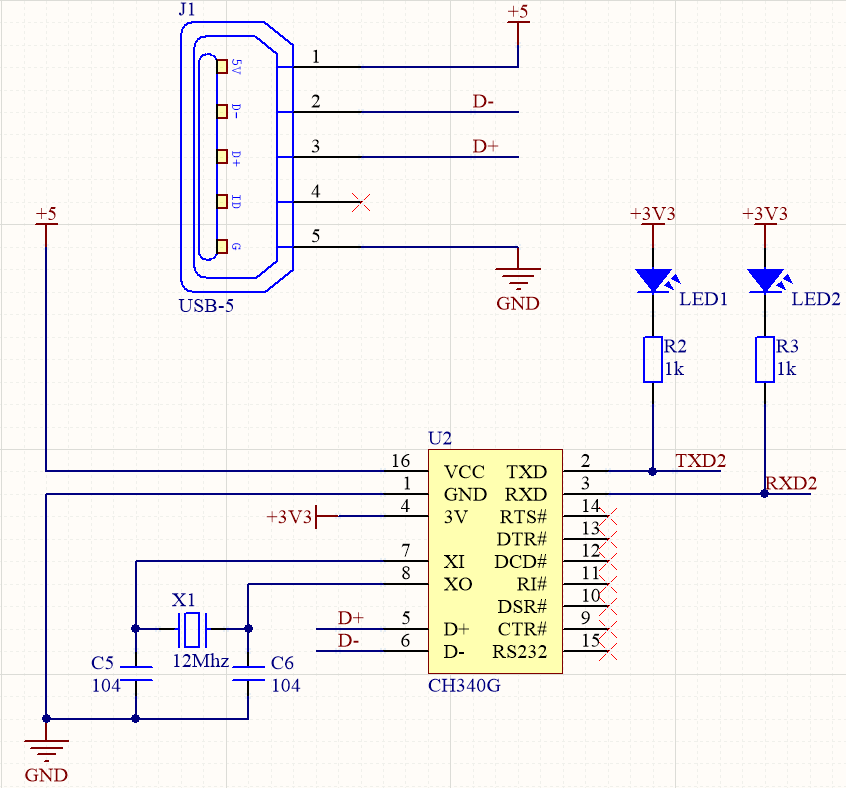


Hình 4. 1: Sơ đồ khối bộ thu tín hiệu Lora kết hợp với máy tính

Trên máy tính sẽ sử dụng code python để xây dựng module phần mềm nhằm giao tiếp nó với mạch RHF76-052 thông qua tập lệnh AT của module này.

### 4.1.2. Sơ đồ nguyên lý và layout

Dưới đây là sơ đồ mạch nguyên lý được thiết kế trên phần mềm Atilum Designer. Dưới đây là sơ đồ mạch nguyên lý khối USB => SERIAL:



Hình 4. 2: Nguyên lý khối USB=>SERIAL

Khối này sử dụng IC CH340 làm IC chính và kết hớp sử dụng USB Cable Type A - Micro B 5 Position, 1.0m để tiện cho việc kết nối với máy tính. Trong khối này tích hợp 2 led đơn dán LED1 và LED2 để thuận tiện cho việc biết sự giao tiếp giữa máy tính và khối Module Lora. Để IC CH340 này hoạt động nó cần được cấp nguồn 5V lấy từ máy tính và có 1 thạch anh 12 MHZ kết nối vào 2 chân XI và XO của nó.

Giới thiệu về IC CH340:

CH340 là bộ chuyển đổi tín hiệu USB theo hai hướng như sau.

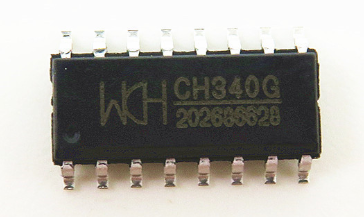
* Bộ chuyển đổi USB sang các cổng giao diện tiếp nối.
* Bộ chuyển đổi USB sang giao diện máy in.

Trong cổng giao diện nối tiếp, CH340 cung cấp tín hiệu MODEM thông thường và được sử dụng để làm lớn các giao diện nối tiếp không đồng bộ của máy tính hoặc nâng cấp các thiết bị nối tiếp  trong quá trình truyền tín hiệu USB một cách trực tiếp.Chi tiết  hơn về bộ chuyển đổi USB sang giao diện máy thì có thể tham khảo đến IC tương tự như CH340DS2.Trong cổng hồng ngoại (giáo thức kết nối các thiết bị điện tử), CH340 được thêm vào bộ thu phát (thu phát tín hiệu) có thể tạo ra một adapter USB  (bộ nguồn USB) or SIR (sự truy tìm thông tin giao tiếp). IC này thường được tích hợp trên các bo mạch Arduino hoặc các mạch Node MCU.

Đặc điểm của IC CH340:

* Tốc độ của cổng giao diện thiết bị USB, phù hợp với Đặc điểm kỹ thuật USB phiên bản 2.0.
* Mô phỏng giao diện nối tiếp theo tiêu chuẩn, được sử dụng để nâng cấp các thiết bị ngoại vi cũ qua cổng USB.
* Hoàn toàn phù hợp với các chương trình ứng dụng nối tiếp trong thiết bị máy tính chạy trên hệ điều hành Windows.
* Giao diện nối tiếp hai chiều, thiết lập bộ thu phát đệm các tín hiệu, hỗ trợ tốc độ truyền thông tin liên lạc khác nhau từ 50bps đến 2Mbps.
* Hỗ trợ tín hiệu MODEM thông thường như RTS DTR, DCD, RI, DSR và CTS.
* Bổ sung thêm thiết bị chuyển đổi như  RS232, RS485, RS422 và giao diện khác.
* Hỗ trợ hoàn toàn các cổng giao tiếp hồng ngoại IrDA SIR và tốc độ truyền tín hiệu khác nhau từ 2400 bps đến 115200 bps.
* Thông qua USB chuyển đổi giao diện nối tiếp, CH340 chỉ phù hợp ở mức tương đối của các chương trình ứng dụng.
* Hỗ trợ 5V và nguồn điện áp 3.3V.

Hình ảnh thực tế của IC:

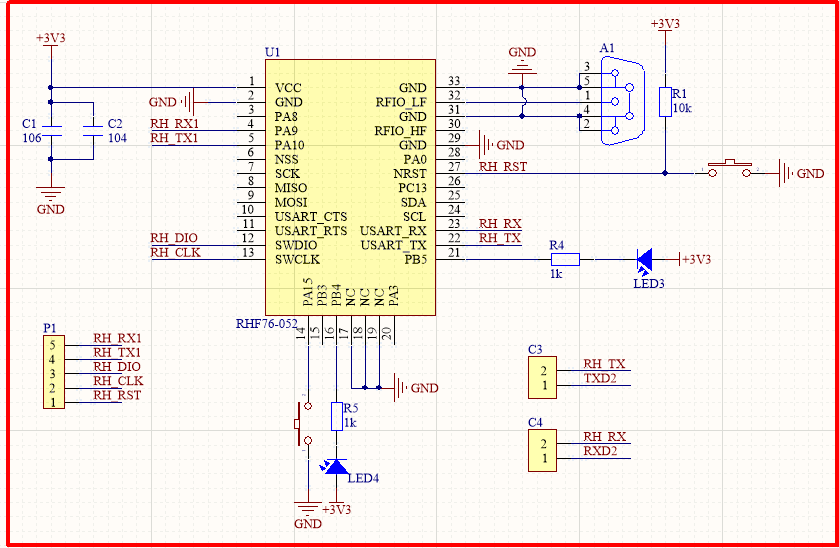


Hình 4. 3: Hình ảnh thực tế IC CH340

Dưới đây là hình ảnh sơ đồ nguyên lý của khối module Lora được thiết kế tương tự như ở các hộp cảm biến:

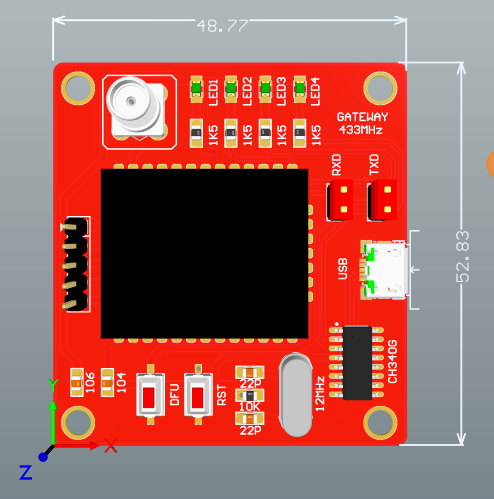
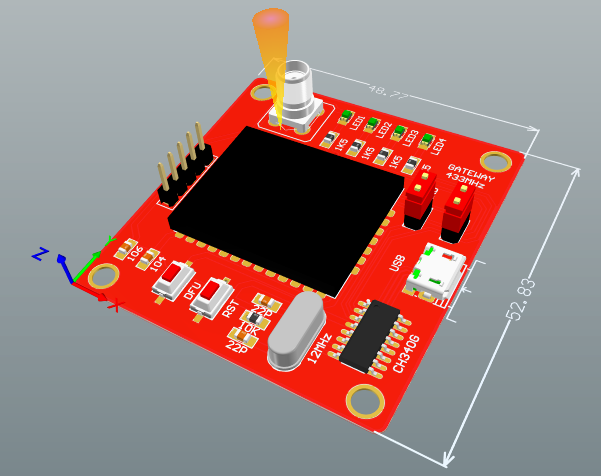
Dưới đây là thiết kế của module:

* Sử dụng anten thu ở dải tần số thấp 434MHz/470MHz nối vào chân số 32
* Chân 22 và chân 23 của modem được kết nối với các chân TX, RX của IC CH340(host) thông qua 2 jump.
* Chân 4 và chân 5 của modem được sử dụng để upgare FW dựa trên kết nối UART.
* Chân 14 được sử dụng để kích hoạt chế độ DFU cho upgrage FW.
* Chân 16 được sử dụng để hiển thị trạng thái của xử lý LoraWAN.
* Chân 21 được dành riêng để kết nối LED và kết nối trở để hạn dòng cho nó.
* Chân 27 kết nối với 1 nút nhấn dán 2 chân và trở kéo lên 10K.



Hình 4. 4: Sơ đồ nguyên lý khối module Lora

Trên bo mạch có tích hợp LED và jump chốt để tiện gỡ rối. Đế anten được bố trí gấn rìa của bo mạch để tiện việc lắp anten. Cũng như vậy Jack USB cũng được đặt ở rìa của bo mạch. Dưới đây là một số hình ảnh 3D của bo mạch.

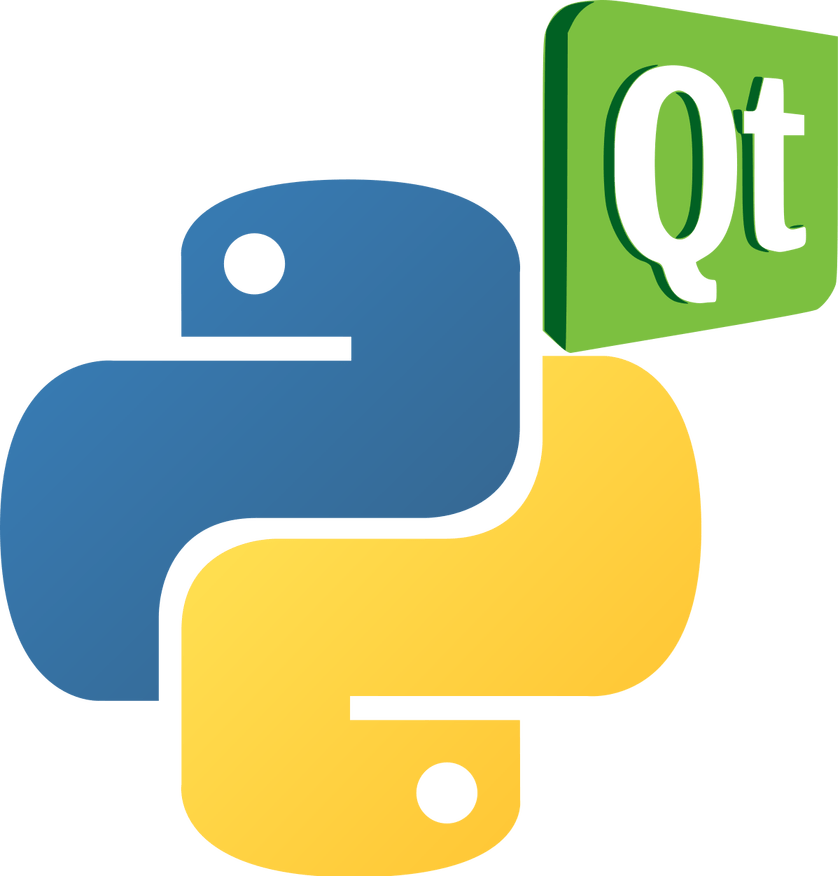
****

Hình 4. 5: Hình ảnh 3D của bo mạch thu tín hiệu Lora

## 4.2. Xây dựng giao diện theo dõi

### 4.2.1. Giới thiệu về PyQt5

Qt là một Application framework đa nền tảng viết trên ngôn ngữ C++ , được dùng để phát triển các ứng dụng trên desktop, hệ thống nhúng và mobile. Hỗ trợ cho các platform bao gồm : Linux, OS X, Windows, VxWorks, QNX, Android, iOS, BlackBerry, Sailfish OS và một số platform khác. PyQt là Python interface của Qt, kết hợp của ngôn ngữ lập trình Python và thư viện Qt, là một thư viện bao gồm các thành phần giao diện điều khiển **(widgets , graphical control elements**).



Hình 4. 6: Hình ảnh biểu tượng của PyQt

PyQt API bao gồm các module bao gồm số lượng lớn với các classes và functions hỗ trợ cho việc thiết kế ra các giao diện giao tiếp với người dùng của các phần mềm chức năng. Hỗ trợ với Python 2.x và 3.x.

Các class của PyQt5 được chia thành các module, bao gồm :

* QtCore : là module bao gồm phần lõi không thuộc chức năng GUI, ví dụ dùng để làm việc với thời gian, file và thư mục, các loại dữ liệu, streams, URLs, mime type, threads hoặc processes.
* QtGui : bao gồm các class dùng cho việc lập trình giao diện (windowing system integration), event handling, 2D graphics, basic imaging, fonts và text.
* QtWidgets : bao gồm các class cho widget, ví dụ : button, hộp thoại, … được sử dụng để tạo nên giao diện người dùng cơ bản nhất.
* QtMultimedia : thư viện cho việc sử dụng âm thanh, hình ảnh, camera,…
* QtBluetooth : bao gồm các class giúp tìm kiếm và kết nối với các thiết bị có giao tiếp với phần mềm.
* QtNetwork : bao gồm các class dùng cho việc lập trình mạng, hỗ trợ lập trình TCP/IP và UDP client , server hỗ trợ việc lập trình mạng.
* QtPositioning : bao gồm các class giúp việc hỗ trợ xác định vị.
* Enginio : module giúp các client truy cập các Cloud Services của Qt.
* QtWebSockets : cung cấp các công cụ cho WebSocket protocol.
* QtWebKit : cung cấp các class dùng cho làm việc với các trình duyệt Web , dựa trên thư viện WebKit2.
* QtWebKitWidgets : các widget cho WebKit.
* QtXml : các class dùng cho làm việc với XML file.
* QtSvg : dùng cho hiển thị các thành phần của SVG file.
* QtSql : cung cấp các class dùng cho việc làm việc với dữ liệu.
* QtTest : cung cấp các công cụ cho phép test các đơn vị của ứng dụng với PyQt5.

Trong đồ án này em sử dụng PyQt5 để xây dựng giao diện theo dõi các cảm biến trên máy tính.

**4.2.2. Xây dựng giao diện theo dõi các thông số và kết nối với Thingsboard server**

**a) Đọc dữ liệu từ bộ thu qua ngôn ngữ python**

Để có được dữ liệu để đưa lên giao diện theo dõi trước tiên máy tính cần phải đọc được dữ liệu từ bộ thu. Việc đọc dữ liệu từ bộ thu bằng ngôn ngữ python được thực hiện rất đơn giản thông qua việc sử dụng module pyserial. Module này là một module truy cập cổng nối tiếp. Nó hỗ trợ python chạy trên Windows, Linux,.. Dưới đây là đoạn code python đọc dữ liệu tự bộ thu tín hiệu Lora, class Gateway được xây dựng để sử dụng tạo đối tượng GW giao tiếp với bộ thu qua UART. Hàm Lora\_init() được viết để gửi những lệnh yêu cầu lấy dữ liệu từ bộ thu.

class Gateway():

def \_\_init\_\_(self, port\_name, baudrate, timeout):

self.ser = serial.Serial()

self.ser.port = str(port\_name)

self.ser.baudrate = baudrate

self.ser.timeout = timeout

def open(self):

if (self.ser.is\_open == False):

self.ser.open()

def write\_data(self, data):

self.ser.write(data.encode('utf-8'))

def read\_data(self):

return self.ser.readlines()

def Lora\_init():

global GW

GW = Gateway("COM16", 9600, 0.2)

GW.open()

GW.write\_data("AT\r\n")

print(GW.read\_data())

GW.write\_data("AT+MODE=TEST\r\n")

print(GW.read\_data())

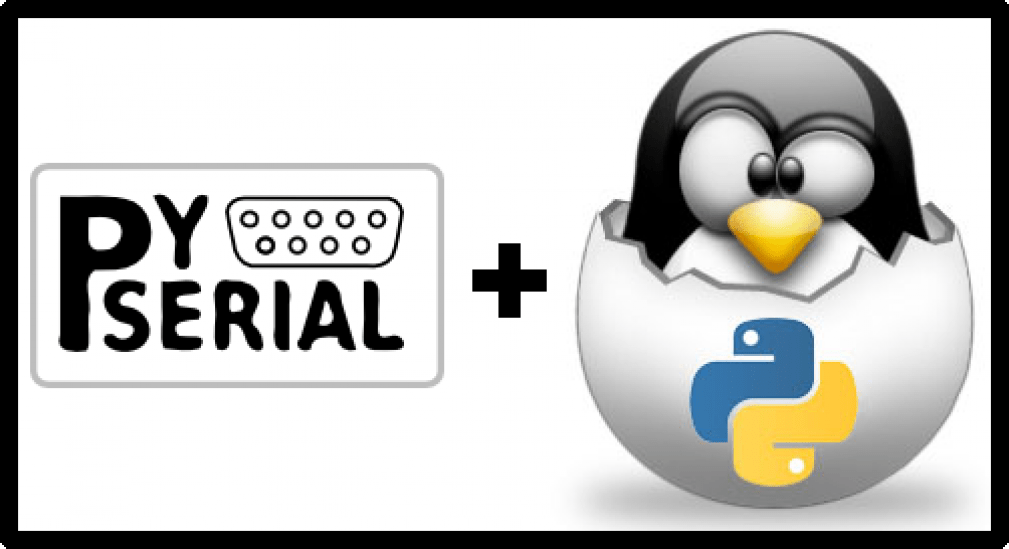
GW.write\_data("AT+TEST=RFCFG,433\r\n")

print(GW.read\_data())

GW.write\_data("AT+TEST=RXLRPKT\r\n")

print(GW.read\_data())

Dưới đây là hình ảnh biểu tưởng của thư viện pyserial:

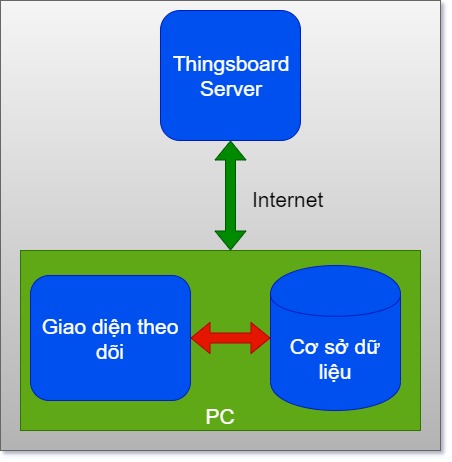


Hình 4. 7: Hình ảnh module Pyserial

Khi bộ thu nhận dữ liệu từ các hộp cảm biến các dữ liệu này đã được mã hóa thành dạng chuỗi hex, sau đó máy tính đọc được chuỗi hex đó thực hiện đoạn code python để chuyển đổi về đúng định dạng và lấy được những giá trị mà nó mong muốn sau đó nó sẽ lưu trữ dữ liệu dưới dạng chuỗi json để sau này có thể gửi lên cloud server một cách dễ dàng.

**b) Xây dựng giao diện theo dõi thông số**

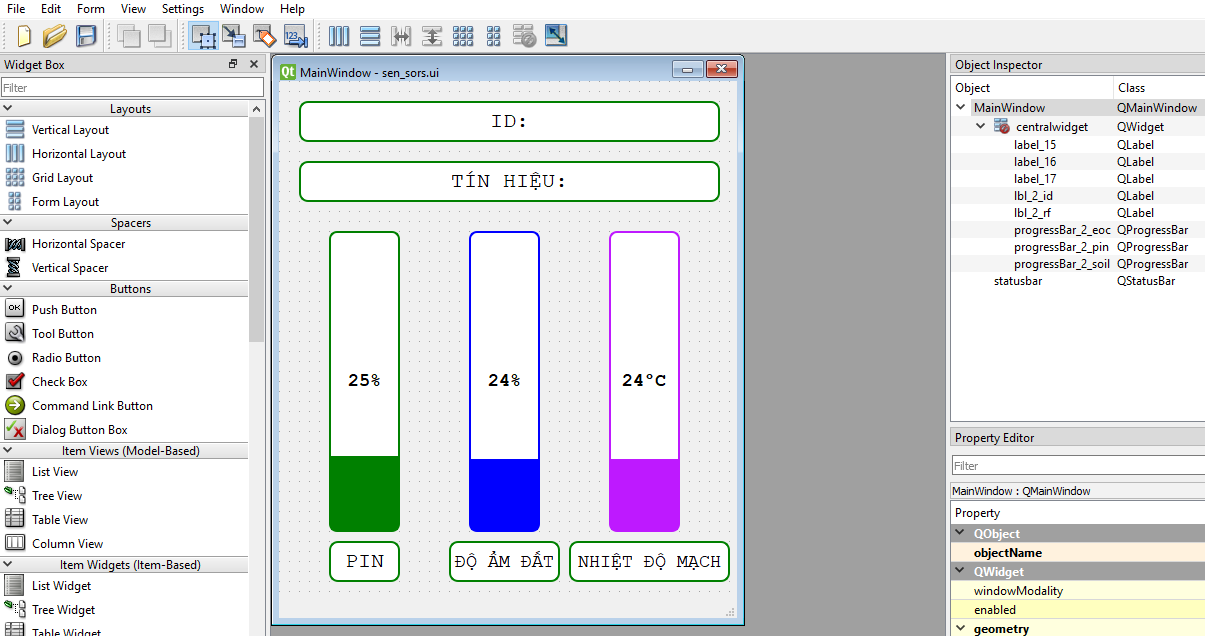
Sau khi đã có đầy đủ các dữ liệu lấy được từ bộ thu, dữ liệu sẽ được hiển thị lên giao diện và lưu trữ vào cơ sở dữ liệu. Nếu máy tính có kết nối internet dữ liệu đó có thể được đẩy lên Thingsboard Server. Mô hình được mô tả như hình dưới đây:



Hình 4. 8: Kết nối giữa máy tính và server

Để xây dựng giao diện theo dõi thông số được thuận tiện và nhanh chóng trong đồ án này em sử dụng một phần mềm hỗ trợ có tên là Qt Designer. Với phần mềm này người lập trình có thể kéo thả một cách đơn giản và sau này bảo trì sửa chữa cũng rất thuận tiện.

Đây là một định dạng dựa trên XML đặc biệt lưu trữ các widget của bạn dưới dạng cây. Khi sử dụng phần mềm này để thiết kế giao diện , nó sẽ giúp người lập trình tạo ra một file có định dạng \*ui. Người lập trình có thể tải các tệp này trong thời gian chạy hoặc dịch chúng sang ngôn ngữ lập trình như C ++ hoặc Python. Dưới đây là hình ảnh khi sử dụng phần mềm này để thiết kế giao diện:



Hình 4. 9: Phần mềm Qt Designer

Như đã giới thiệu ở trên, sau khi đã hoàn kéo thả hoàn thành một Form sẽ tạo ra một file giao diện có đuôi \*.ui. Sau đó cần import file giao diện này vào trong file main.py của project thông qua cú pháp như đoạn code dưới đây:

from PyQt5 import QtWidgets, uic, QtGui, QtCore

import sys

import time

from PyQt5.QtWidgets import QMessageBox, QFileDialog, QAction, QGroupBox, QTableWidget, QTableWidgetItem, QWidget

from PyQt5.QtCore import QTimer, QTime, QThread, pyqtSignal, QDate, Qt

from PyQt5.QtGui import QPixmap, QCloseEvent, QColor

import random

import os

import paho.mqtt.client as mqtt

import json

from datetime import datetime

import urllib.request

import socket

import constant as CONSTANT

App = QtWidgets.QApplication([])

app = uic.loadUi("guis\\main.ui")// đọc file giao diện main.ui

# app.setWindowFlag(QtCore.Qt.WindowCloseButtonHint, False)

if \_\_name\_\_ == "\_\_main\_\_":

app.show()

sys.exit(App.exec())

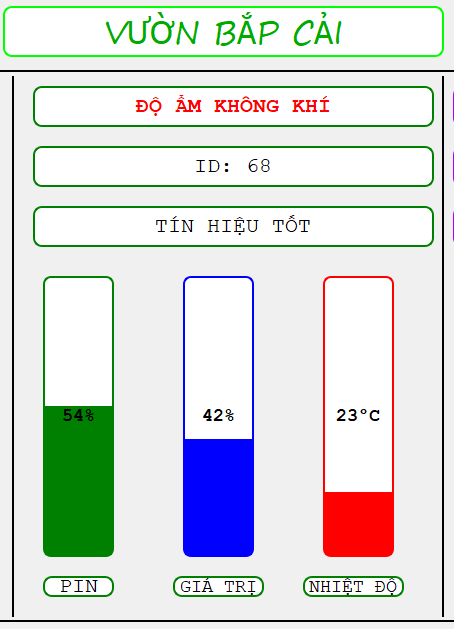
Dưới đây là giao diện được thiết kế để theo dõi các dữ liệu được gửi về từ hộp cảm biến. Các giá trị được sau khi lấy từ bộ thu được đưa lên giao diện này để cho người nông dân có thể biết rằng hộp cảm biến có đang hoạt động không, pin còn bao nhiêu % và giá trị cảm biến đo được ở khu vực đó là bao nhiêu sẽ đưa ra cảnh báo khi giá trị vượt ngưỡng cho phép. Ngoài ra nếu Internet nó sẽ gửi dữ này lên một server đã cài đặt nền tảng Thingsboard. Nếu không có mạng dữ liệu sẽ được lưu vào các bảng trong cơ sở dữ liệu mỗi ngày sẽ có bảng dành riêng cho ngày hôm đó với định dạng như sau data\_of\_ngay\_thanh\_nam.

Về SQLite:

SQLite là một hệ quản trị cơ sở dữ liệu quan hệ, khác với các hệ quản trị CSDL khác nó không cần client–server database engine (một dịch vụ, ứng dụng cài đặt đầy đủ thông qua nó kết nối, tương tác với các CSDL), nó được nhúng vào rất nhiều chương trình khác nhau, từ destop, mobile đến website. Hầu hết ngôn ngữ lập trình có các thư viện hỗ trợ kết nối, truy vấn đến SQLite như: PHP C, Java, C#, JavaScript... Nó được nhúng vào các trình duyệt, vào các hệ thống nhúng, hệ điều hành như Android, IOS ... đều đã nhúng mặc định .



Hình 4. 10: Biểu tượng SQLite



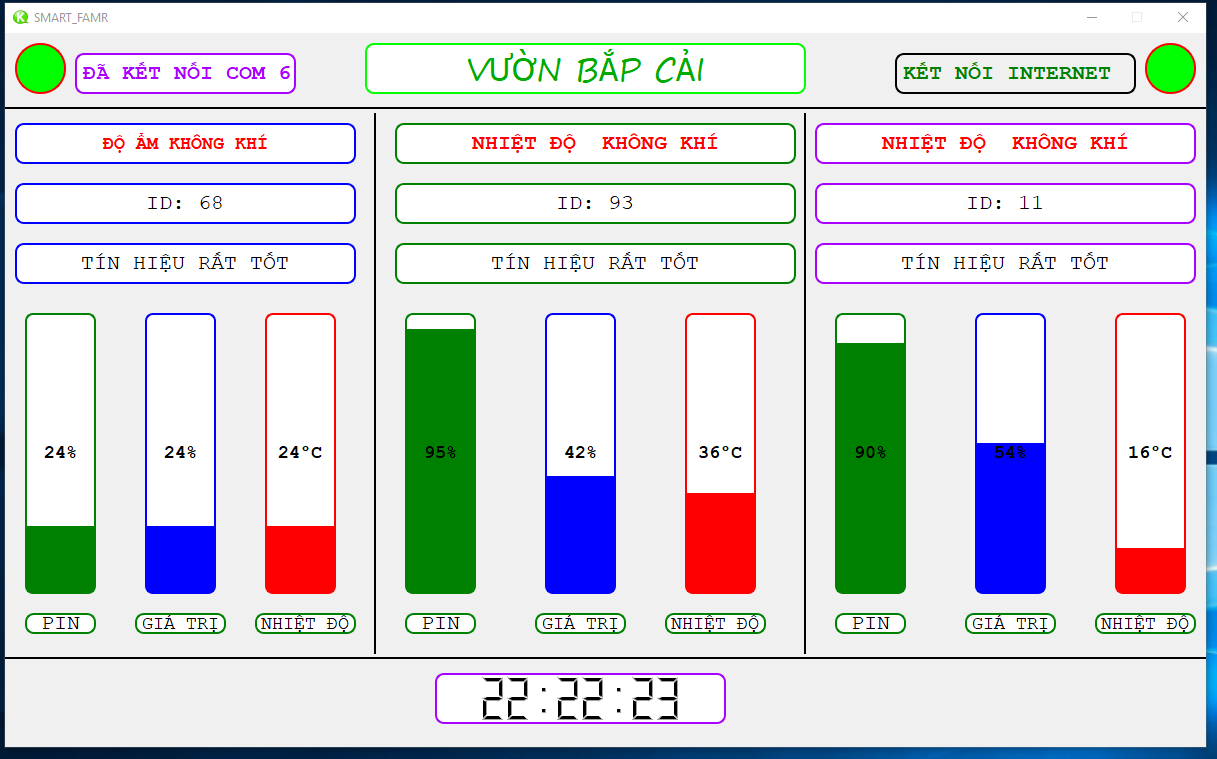
Hình 4. 11: Giao diện hiển thị cho một cảm biến

# CHƯƠNG V: KẾT LUẬN

Sau khi đã hoàn thành các bước xây dựng, mạch và pin sẽ được đưa vào đóng gói bên trong hộp kỹ thuật chống nước JL-00C của Lioa. Phần mềm giao diện theo dõi thông số sẽ được chuyển đổi thành bộ cài đặt đuôi \*.exe để có thể cài đặt trên các máy tính chạy hệ điều hành Window. Dưới đây là hình ảnh sản phẩm khi đóng hộp và giao diện phần mềm hoàn chỉnh.



Hình 5. 1: Phía bên trong hộp cảm biến



Hình 5. 2: Giao diện theo dõi 3 cảm biến trên máy tính chạy window

Đề tài “Xây dựng hộp cảm biến trong nông nghiệp thông minh sử dụng công nghệ lora” đã hoàn thành các nội dung trong đề cương được phê duyệt. Đề tài đã đạt được một số kết quả chính như sau:

* Hiểu rõ đặc điểm, tính năng của công nghệ Lora.
* Thiết kế thành công hộp cảm biến đo nhiệt độ ẩm không khí và độ ẩm đất ứng dụng được công nghệ Lora để truyền tải dữ liệu.
* Xây dựng bộ thu dữ liệu từ hộp cảm biến và kết nối bộ thu với máy tính.
* Xây dựng giao diện hiển thị dữ liệu trên máy tính.
* Thiết kế vỏ hộp để có hộp hoạt động lâu dài trong môi trường tự nhiên.
* **Đã triển khai thực tế tại trang trại Thụy Hương, Hà Nội**
* **Đạt giải 3 trong cuộc thi Advantech 2019.**

Hướng phát triển của đề tài

Trong tương lai em sẽ tích hợp với các thiết bị cảm biến chuyên dụng trong nông nghiệp. Em sẽ nghiên cứu và phát triển phần cứng tốt hơn để tăng độ bền, độ ổn định của hộp, tăng thời gian sử dụng của pin. Ngoài ra em sẽ tìm hiểu và sử dụng các nền tảng IoT hỗ trợ để có thể xây dựng một mạng LoraWan để đề tài có thể ứng dụng thực tiễn vào trong nền nông nghiệp của Việt Nam và cố gắng xây dựng một nền tảng Iot của riêng mình.

# TÀI LIỆU THAM KHẢO

[1] Tham khảo thông tin trên trang chủ : <https://lora-alliance.org/about-lorawan>

[2] Wireless Sensor Networks Authors: **Yang**, Shuang-Hua

[3] <http://image.sciencenet.cn/olddata/kexue.com.cn/bbs/upload/12615WSN-2007.pdf>  
[4] Anna Ha’c, Wireless Sensor Network Designs, University of Hawaii at Manoa, Honolulu, USA, John Wiley & Sons Ltd, 2003.

[5]. Xiong, S.-M.; Wang, L.-M.; Qu, X.-Q.; Zhan, Y.-Z. Application Research of WSN in Precise Agriculture Irrigation. In Proceedings of International Conference on Environmental Science and Information Application Technology, Wuhan, China, 4– 5 July 2009; Volume 2, pp. 297-300.

[6]. KSHITIJ SHINGHAL, ARTI NOOR, etc, Wireless Sensor Networks in agriculture: for potato farming.

[7]. B. S. Kang , S. Kim , F. Ren , B. P. Gila , C. R. Abernathy and S. J. Pearton, (2005), “Comparison of \MOS and Schottky W/Pt– GaN diodes for hydrogen detection,” Sens. Actuators,Chem., vol.104, pp.232, 2005.

[8] <https://pythonhosted.org/pyserial/>

# PHỤ LỤC