

TP.HCM, ngày 20 tháng 6 năm 2019

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Đào Duy Tùng MSSV: 15151242

Nguyễn Đình Thiện MSSV: 15151221

Chuyên ngành: CNKT Điều khiển và Tự động hóa Mã ngành: 51

Hệ đào tạo: Đại học chính quy Mã hệ: 1

Khóa: 2015

1. Tên đề tài: Mô hình nhà vườn thông minh sử dụng Công nghệ IOT
2. Nhiệm vụ
 - 2.1 Các số liệu ban đầu
 - Chip STM32F103C8T6 và SIM808
 - Arduino Mega2560
 - 2.2 Nội dung thực hiện
 - Thiết kế và thi công mạch Driver dùng để thu thập dữ liệu cảm biến và điều khiển mạch công suất.
 - Viết chương trình giao tiếp giữa các module để truyền dữ liệu cảm biến và điều khiển các thiết bị điện
 - Thiết kế cơ khí của mô hình, tìm hiểu Webserver.
 - Thi công mô hình và viết báo cáo
3. Ngày giao nhiệm vụ: 20/03/2019
4. Ngày hoàn thành nhiệm vụ: 30/6/2019
5. Họ và tên cán bộ hướng dẫn: TS Nguyễn Văn Thái

CÁN BỘ HƯỚNG DẪN

BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN

TRƯỜNG ĐH SPKT TP. HỒ CHÍ MINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

KHOA ĐIỆN – ĐIỆN TỬ

ĐỘC LẬP - TỰ DO - HẠNH PHÚC

BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN

----o0o----

LỊCH TRÌNH THỰC HIỆN ĐỒ ÁN TỐT NGHIỆP

Họ và tên sinh viên 1: Đào Duy Tùng

Lớp: 151511B **MSSV:** 15151242

Họ và tên sinh viên 2: Nguyễn Đình Thiện

Lớp: 151511B **MSSV:** 15151221

Tên đề tài: Mô hình nhà vườn thông minh sử dụng Công nghệ IOT

Tuần/ngày	Nội dung	Xác nhận của GVHD
Tuần 1 18/3 – 24/3	Gặp giảng viên hướng dẫn và trao đổi về đề tài đồ án tốt nghiệp.	
Tuần 2 25/3 – 31/3	Viết đề cương và lịch trình thực hiện đồ án tốt nghiệp.	
Tuần 3 1/4– 7/4	Tìm hiểu đề tài và lựa chọn thiết bị.	
Tuần 4 8/5– 14/4	Tìm hiểu nguyên lý hoạt động của đề tài.	
Tuần 5 15/4 – 21/4	Thiết kế sơ đồ khối, sơ đồ nguyên lý.	
Tuần 6 22/4 – 28/4	Thiết kế phần cơ khí Solidworks của mô hình	
Tuần 7 29/4 – 5/5	Thi công mạch DRIVER (Thu thập giá trị cảm biến và mạch công suất)	
Tuần 8 6/5 – 12/5	Viết chương trình, kiểm tra kết nối giữa mạch GATEWAY và mạch DRIVER	
Tuần 9 13/5 – 19/5	Thi công phần cứng, lắp ráp mô hình nhà.	
Tuần 10 20/5 – 26/5	Thi công phần cứng, lắp ráp mô hình vườn.	

Tuần 11 27/5 – 2/6	Kết nối 2 mạch GATEWAY và mạch DRIVER, thử nghiệm qua web server	
Tuần 12 3/6 – 9/6	Hoàn thiện mô hình, đóng gói hệ thống và chạy thử nghiệm	
Tuần 13-16	Chạy thử nghiệm và cân chỉnh toàn hệ thống.	

10/6 – 30/6	Đánh giá kết quả đạt được, viết báo cáo.	
-------------	--	--

GV HƯỚNG DẪN
(Ký và ghi rõ họ và tên)

TRƯỜNG ĐH SPKT TP. HỒ CHÍ MINH **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**
KHOA ĐIỆN – ĐIỆN TỬ **ĐỘC LẬP - TỰ DO - HẠNH PHÚC**
BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN **----o0o----**

TP.HCM, ngày.... tháng.... năm

2019

LỜI CAM ĐOAN

Tôi xin cam kết đề tài này là do tôi tự thực hiện dựa vào một số tài liệu trước đó và không sao chép từ tài liệu hay công trình đã có trước đó.

Người thực hiện đề tài

Đào Duy Tùng Nguyễn Đình Thiện

LỜI CẢM ƠN

Để hoàn thành đề tài nghiên cứu này, lời đầu tiên cho phép chúng em được gửi lời cảm ơn chân thành đến toàn thể quý thầy cô **Trường Đại Học Sư Phạm Kỹ Thuật TP.HCM** nói chung và các thầy cô trong **Khoa Điện – Điện Tử** nói riêng, những người đã tận tình dạy dỗ, trang bị cho chúng em những kiến thức nền tảng và kiến thức chuyên ngành quan trọng, giúp nhóm chúng em có được cơ sở lý thuyết vững vàng và đã luôn tạo điều kiện giúp đỡ tốt nhất cho chúng em trong quá trình học tập và nghiên cứu.

Đặc biệt, chúng em xin chân thành cảm ơn Tiến sĩ Nguyễn Văn Thái đã tận tình giúp đỡ, đưa ra những định hướng nghiên cứu cũng như hướng giải quyết một số vấn đề để chúng em có thể thực hiện tốt đề tài. Trong thời gian làm việc với thầy, chúng em đã không ngừng tiếp thu thêm nhiều kiến thức được chỉ dạy từ thầy, luôn thể hiện một thái độ nghiên cứu nghiêm túc, hiệu quả và đây cũng là điều rất cần thiết trong quá trình học tập và làm việc sau này đối với chúng em. Em cũng xin cảm ơn Thầy Nguyễn Hữu Trung và anh Trần Đăng Khôi cùng các bạn thành viên của 3D VisionLab đã giúp đỡ chúng em rất nhiều trong quá trình thực hiện đề tài.

Cuối cùng em xin cảm ơn bố mẹ – những người đã sinh thành ra chúng em, luôn sát cánh và động viên chúng em từ những năm đầu đời đến bây giờ, khi chúng em đã hoàn thành 4 năm đại học. Bố mẹ luôn luôn tạo điều kiện tốt nhất để chúng em có thể theo học tại ngôi trường top đầu khu vực.

Mặc dù đã cố gắng hết sức, song do điều kiện thời gian và kinh nghiệm thực tế của nhóm nghiên cứu còn ít, cho nên đề tài không thể tránh khỏi thiếu sót. Vì vậy, chúng em rất mong nhận được sự đóng góp ý kiến của quý thầy, cô giáo.

Xin chân thành cảm ơn!

Người thực hiện đề tài

Đào Duy Tùng – Nguyễn Đình Thiện

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

TÓM TẮT

Hiện nay, việc phát triển của thế giới điện tử số phát triển một cách nhanh chóng và mạnh mẽ, điện tử số cụ thể là vi xử lí ngày càng trở nên đa dạng và các ứng dụng cũng gần gũi với chúng ta hơn. Cùng với sự phát triển đa dạng của ngành công nghiệp vi xử lí nên tài nguyên của vi xử lý cũng được nâng cao để đáp ứng các ứng dụng khác nhau trong thực tế. Mạng Internet ngày càng ứng dụng rộng rãi trong mọi lĩnh vực của đời sống xã hội. Công nghệ ngày càng phát triển đòi hỏi nhu cầu ứng dụng vào ngành công nghiệp nhằm giảm lao động, đảm bảo sức khỏe nhân công để không ảnh hưởng xấu đến chất lượng sản phẩm... đem lại hiệu quả cao cho nền công nghiệp ngày càng tiên tiến. Ở đồ án tốt nghiệp này nhóm em thiết kế, thi công mô hình với 1 trạm xử lý và 1 trạm trung tâm xử lý cho phép điều khiển và giám sát hệ thống. Trạm xử lý sẽ gửi tín hiệu, các thông số môi trường về cho trung tâm xử lý, các thông số này sẽ được giám sát tại trung tâm. Người sử dụng có thể điều khiển thiết bị điện ở khoảng cách xa, ở bất cứ nơi nào có Internet, Wifi, 3G, 4G trên Website được thiết kế, lệnh điều khiển sẽ được gửi về trạm xử lý để đóng cắt mạch công suất. Các giá trị nhiệt độ, độ ẩm không khí, độ ẩm đất, khí CO₂, cường độ ánh sáng của mô hình cũng được cập nhật lên giao diện này.

Chương 1. TỔNG QUAN

1.1. ĐẶT VẤN ĐỀ

Ngày nay với sự phát triển không ngừng của khoa học và công nghệ với những ứng dụng của khoa học kĩ thuật tiên tiến, thế giới chúng ta đã và đang ngày một thay đổi, văn minh và hiện đại hơn. Sự phát triển của kĩ thuật điện tử đã tạo ra hàng loạt những thiết bị với đặc điểm nổi bật như sự chính xác cao, tốc độ nhanh, gọn nhẹ là những yếu tố rất cần thiết cho hoạt động của con người đạt hiệu quả cao. Một trong những ứng dụng quan trọng trong công nghệ điện tử là kĩ thuật điều khiển từ xa. Nó đã góp phần rất lớn trong việc điều khiển các thiết bị từ xa nhằm đáp ứng nhu cầu của con người, giúp tiết kiệm được thời gian và quản lý dễ dàng.

Bắt nguồn từ những nhu cầu cần thiết đó và lấy cảm hứng từ dự án như:

“Tủ trồng rau sạch thông minh dùng công nghệ IOT”. Vì thế nhóm em đã quyết định chọn đề tài “**Mô hình nhà vườn thông minh sử dụng Công nghệ IOT**” để cải thiện thêm chức năng điều khiển được nhiều thiết bị và giám sát các cảm biến thông qua chuẩn truyền không dây của SIM808.

Nội dung chính của đề tài:

- Thiết kế và thi công module Gateway làm board trung tâm cho mạch trung tâm xử lý.
- Thiết kế và thi công module Driver làm board trung tâm cho mạch trạm xử lý.
- Thiết kế và thi công mô hình cơ khí.
- Điều khiển các thiết bị bằng Webserver.
- Giám sát trạng thái hoạt động của các thiết bị và giá trị cảm biến qua Webserver.

1.2. MỤC TIÊU

Đề tài: “**Mô hình nhà vườn thông minh sử dụng Công nghệ IOT**” bao gồm các vấn đề chính sau:

- Thiết kế và thi công mạch điều khiển bao gồm: mạch trung tâm xử lý GATEWAY và mạch trạm xử lý DRIVER.
- Tìm hiểu về hệ thống giám sát và điều khiển trên website <http://iotstar.vn>.
- Thiết kế và thi công mô hình ngôi nhà, mô hình tủ trồng rau, chạy thử nghiệm.

Chương 1. TỔNG QUAN

1.3. NỘI DUNG NGHIÊN CỨU

- NỘI DUNG 1: Tìm hiểu và tham khảo các tài liệu, giáo trình, nghiên cứu các chủ đề, các nội dung liên quan đến đề tài.
- NỘI DUNG 2: Thiết kế sơ đồ khối và sơ đồ nguyên lý cho hệ thống.
- NỘI DUNG 3: Thiết kế và thi công mạch trạm xử lý DRIVER với cảm biến nhiệt độ - độ ẩm, cảm biến độ ẩm đất, cảm biến CO2, cảm biến ánh sáng và điều khiển thiết bị công suất.
- NỘI DUNG 4: Thiết kế và thi công mạch GATEWAY với chip xử lý STM32 và SIM808.
- NỘI DUNG 5: Tìm hiểu Webserver <http://iotstar.vn> dùng để giám sát và điều khiển các thiết bị và cảm biến.
- NỘI DUNG 6: Viết được các chương trình để giao tiếp giữa Webserver, Module GATEWAY và Module DRIVER.
- NỘI DUNG 7: Thiết kế và thi công mô hình hoàn thiện.
- NỘI DUNG 8: Chạy thử nghiệm và cân chỉnh hệ thống.
- NỘI DUNG 9: Viết báo cáo khóa luận tốt nghiệp.
- NỘI DUNG 10: Báo cáo đồ án tốt nghiệp.

1.4. GIỚI HẠN

- Sử dụng ARM MCU (STM32), SIM808, Arduino Mega2560
- Điều khiển và giám sát các thiết bị, cảm biến qua Webserver.
- Sử dụng các cảm biến như nhiệt độ - độ ẩm, cảm biến độ ẩm đất, cảm biến ánh sáng, cảm biến CO2 với độ chính xác khá cao.
- Mô hình chỉ ứng dụng bật tắt cho các thiết bị công suất nhỏ như đèn, quạt, bơm...

Chương 2. CƠ SỞ LÝ THUYẾT

2.1. CƠ SỞ LÝ THUYẾT VỀ IOT

2.1.1. Giới thiệu về cách mạng công nghiệp 4.0

Bối cảnh xuất hiện

Còn được gọi là cuộc cách mạng số với sự xuất hiện của những công nghệ như Internet vạn vật (IOT-Internet Of Things), trí tuệ nhân tạo (AI-Artificial Intelligence), thực tế ảo (VR-Virtual Reality), tương tác thực tại ảo (AR-Augmented Reality), mạng xã hội, điện toán đám mây,... Cuộc cách mạng lần thứ 4 này bắt đầu từ những năm 2000. Nó nhằm chuyển đổi toàn bộ thế giới thực sang thế giới số.

Cụm từ “Công nghiệp 4.0” (Industrie 4.0) nổi lên ở Đức năm 2013. Khi đó người ta nói về chiến lược công nghệ cao, điện toán hóa ngành sản xuất mà không cần sự tham gia của con người. Hiện nay, Công nghiệp 4.0 đã vượt ra khỏi khuôn khổ dự án của Đức. Nó có sự tham gia của nhiều nước. Giờ đây, trở thành một phần quan trọng của cuộc cách mạng công nghiệp lần thứ tư.

Cuộc cách mạng này được cho rằng sẽ ảnh hưởng mạnh tới mọi mặt đời sống kinh tế, xã hội. Điển hình là các ngành dệt may, da giày, dịch vụ, các ngành y tế, giao thông... thậm chí là cả nông nghiệp.



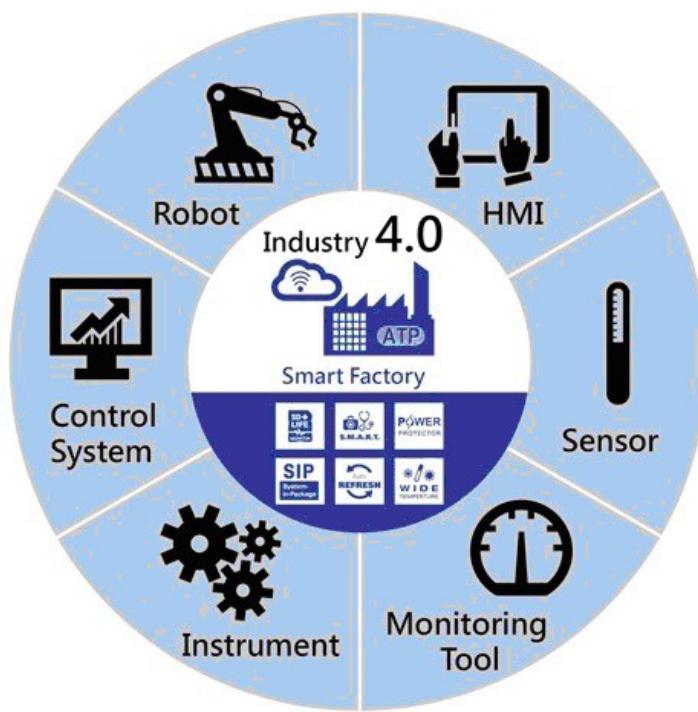
Hình 2.1 : Cách mạng công nghiệp 4.0

Theo Gartner, Cách mạng Công nghiệp 4.0 xuất phát từ khái niệm “Industrie 4.0” trong một báo cáo của chính phủ Đức năm 2013. “Industrie 4.0” kết nối các hệ

Chương 2. CƠ SỞ LÝ THUYẾT

thống nhúng và cơ sở sản xuất thông minh. Để tạo ra sự hội tụ kỹ thuật số giữa Công nghiệp, Kinh doanh, chức năng và quy trình bên trong.

“Cách mạng công nghiệp đầu tiên sử dụng năng lượng nước và hơi nước để cơ giới hóa sản xuất. Cuộc cách mạng lần 2 diễn ra nhờ ứng dụng điện năng để sản xuất hàng loạt. Cuộc cách mạng lần 3 sử dụng điện tử và công nghệ thông tin để tự động hóa sản xuất. Nay giờ, cuộc Cách mạng Công nghiệp Thứ tư đang nảy nở từ cuộc cách mạng lần ba. Nó kết hợp các công nghệ lại với nhau, làm mờ ranh giới giữa vật lý, kỹ thuật số và sinh học”. Đây là khái niệm của Klaus Schwab.



Hình 2.2: IOTS và Tự động hóa robot trong cách mạng công nghiệp 4.0

Cuộc cách mạng này sẽ diễn ra trên ba lĩnh vực chính bao gồm: công nghệ sinh học, kỹ thuật số và vật lý. Đặc trưng nổi bật của cuộc cách mạng này đó là dùng công nghệ thay thế dần sự có mặt của con người trong mọi hoạt động.

Những yếu tố cốt của cách mạng 4.0 gồm 4 yếu tố gồm chuỗi khối (blockchain), trí tuệ nhân tạo (AI), Vạn vật kết nối – Internet of Things (IoT) và dữ liệu lớn (Big Data).

Chương 2. CƠ SỞ LÝ THUYẾT

Trên đây chỉ là một số kiến thức cơ bản cho bạn đọc cái nhìn khái quát về cuộc cách mạng công nghiệp này. Chúng ta càng cần trang bị những kiến thức tốt hơn nữa để đón đầu xu thế của thế giới.

2.1.2. Tổng quan về Internet Of Things (IOT)

Mạng lưới vạn vật kết nối Internet hoặc là Mạng lưới thiết bị kết nối Internet viết tắt là IoT (tiếng Anh: Internet of Things) là một kịch bản của thế giới, khi mà mỗi đồ vật, con người được cung cấp một định danh của riêng mình, và tất cả có khả năng truyền tải, trao đổi thông tin, dữ liệu qua một mạng duy nhất mà không cần đến sự tương tác trực tiếp giữa người với người, hay người với máy tính. IoT đã phát triển từ sự hội tụ của công nghệ không dây, công nghệ vi cơ điện tử và Internet. Nói đơn giản là một tập hợp các thiết bị có khả năng kết nối với nhau, với Internet và với thế giới bên ngoài để thực hiện một công việc nào đó.

Hay hiểu một cách đơn giản IoT là tất cả các thiết bị có thể kết nối với nhau . Việc kết nối thì có thể thực hiện qua Wi-Fi, mạng viễn thông băng rộng (3G, 4G), Bluetooth, ZigBee, hồng ngoại... Các thiết bị có thể là điện thoại thông minh, máy pha cafe, máy giặt, tai nghe, bóng đèn, và nhiều thiết bị khác. Cisco, nhà cung cấp giải pháp và thiết bị mạng hàng đầu hiện nay dự báo: Đến năm 2020, sẽ có khoảng 50 tỷ đồ vật kết nối vào Internet, thậm chí con số này còn gia tăng nhiều hơn nữa. IoT sẽ là mạng khổng lồ kết nối tất cả mọi thứ, bao gồm cả con người và sẽ tồn tại các mối quan hệ giữa người và người, người và thiết bị, thiết bị và thiết bị. Một mạng lưới IoT có thể chứa đến 50 đến 100 nghìn tỉ đối tượng được kết nối và mạng lưới này có thể theo dõi sự di chuyển của từng đối tượng. Một con người sống trong thành thị có thể bị bao bọc xung quanh bởi 1000 đến 5000 đối tượng có khả năng theo dõi.

Những dịch vụ liên quan đến “Things”: hệ thống IoT có khả năng cung cấp các dịch vụ liên quan đến “Things”, chẳng hạn như bảo vệ sự riêng tư và quản lý giữa Physical Thing và Virtual Thing. Để cung cấp được dịch vụ này, cả công nghệ phần cứng và công nghệ thông tin(phần mềm) sẽ phải thay đổi.

Chương 2. CƠ SỞ LÝ THUYẾT

Tính không đồng nhất: Các thiết bị trong IoT là không đồng nhất vì nó có phần cứng khác nhau, và network khác nhau. Các thiết bị giữa các network có thể tương tác với nhau nhờ vào sự liên kết của các network.

Thay đổi linh hoạt: Status của các thiết bị tự động thay đổi, ví dụ, ngủ và thức dậy, kết nối hoặc bị ngắt, vị trí thiết bị đã thay đổi, và tốc độ đã thay đổi... Hơn nữa, số lượng thiết bị có thể tự động thay đổi.

Quy mô lớn: Sẽ có một số lượng rất lớn các thiết bị được quản lý và giao tiếp với nhau. Số lượng này lớn hơn nhiều so với số lượng máy tính kết nối Internet hiện nay. Số lượng các thông tin được truyền bởi thiết bị sẽ lớn hơn nhiều so với được truyền bởi con người.

Một hệ thống IoT phải thoả mãn các yêu cầu sau:

Kết nối dựa trên sự nhận diện: Nghĩa là các “Things” phải có ID riêng biệt. Hệ thống IoT cần hỗ trợ các kết nối giữa các “Things”, và kết nối được thiết lập dựa trên định danh (ID) của Things.

Khả năng cộng tác: hệ thống IoT khả năng tương tác qua lại giữa các network và Things.

Khả năng tự quản của network: Bao gồm tự quản lý, tự cấu hình, tự chữa bệnh, tự tối ưu hóa và tự có cơ chế bảo vệ. Điều này cần thiết để network có thể thích ứng với các domains ứng dụng khác nhau, môi trường truyền thông khác nhau, và nhiều loại thiết bị khác nhau.

Dịch vụ thoả thuận: dịch vụ này để có thể được cung cấp bằng cách thu thập, giao tiếp và xử lý tự động các dữ liệu giữa các “Things” dựa trên các quy tắc(rules) được thiết lập bởi người vận hành hoặc tùy chỉnh bởi các người dùng.

Các Khả năng dựa vào vị trí(location-based capabilities): Thông tin liên lạc và các dịch vụ liên quan đến một cái gì đó sẽ phụ thuộc vào thông tin vị trí của Things và người sử dụng. Hệ thống IoT có thể biết và theo dõi vị trí một cách tự động. Các dịch vụ dựa trên vị trí có thể bị hạn chế bởi luật pháp hay quy định, và phải tuân thủ các yêu cầu an ninh.

Chương 2. CƠ SỞ LÝ THUYẾT

Bảo mật: Trong IoT, nhiều “Things” được kết nối với nhau. Chình điều này làm tăng mối nguy trong bảo mật, chẳng hạn như bí mật thông tin bị tiết lộ, xác thực sai, hay dữ liệu bị thay đổi hay làm giả.

Bảo vệ tính riêng tư: tất cả các “Things” đều có chủ sở hữu và người sử dụng của nó. Dữ liệu thu thập được từ các “Things” có thể chứa thông tin cá nhân liên quan chủ sở hữu hoặc người sử dụng nó. Các hệ thống IoT cần bảo vệ sự riêng tư trong quá trình truyền dữ liệu, tập hợp, lưu trữ, khai thác và xử lý. Bảo vệ sự riêng tư không nên thiết lập một rào cản đối với xác thực nguồn dữ liệu.

Khả năng quản lý: hệ thống IoT cần phải hỗ trợ tính năng quản lý các “Things” để đảm bảo network hoạt động bình thường. Ứng dụng IoT thường làm việc tự động mà không cần sự tham gia người, nhưng toàn bộ quá trình hoạt động của họ nên được quản lý bởi các bên liên quan.

Tác động của IoT rất đa dạng, trên các lĩnh vực: quản lý hạ tầng, y tế, xây dựng và tự động hóa, giao thông.... Cụ thể trong lĩnh vực y tế, Thiết bị IoT có thể được sử dụng để cho phép theo dõi sức khỏe từ xa và hệ thống thông báo khẩn cấp. Các thiết bị theo dõi sức khỏe có thể dao động từ huyết áp và nhịp tim màn với các thiết bị tiên tiến có khả năng giám sát cấy ghép đặc biệt, chẳng hạn như máy điều hòa nhịp hoặc trợ thính tiên tiến. cảm biến đặc biệt cũng có thể được trang bị trong không gian sống để theo dõi sức khỏe và thịnh vượng chung là người già, trong khi cũng bảo đảm xử lý thích hợp đang được quản trị và hỗ trợ người dân lấy lại mất tinh di động thông qua điều trị là tốt. thiết bị tiêu dùng khác để khuyến khích lối sống lành mạnh, chẳng hạn như, quy mô kết nối hoặc máy theo dõi tim mặc.

Internet of Things đến năm 2020(Theo báo cáo của BI Intelligence):

- + 4 tỷ người kết nối với nhau
- + 4 ngàn tỷ USD doanh thu
- + Hơn 25 triệu ứng dụng
- + Hơn 25 tỷ hệ thống nhúng và hệ thống thông minh
- + 50 ngàn tỷ Gigabytes dữ liệu

Chương 2. CƠ SỞ LÝ THUYẾT

2.1.3. Tổng quan về Nhà thông minh (Smart Home)

Nhà thông minh là ngôi nhà được trang bị các hệ thống tự động thông minh cùng với cách bố trí hợp lý, các hệ thống này có khả năng tự điều phối các hoạt động trong ngôi nhà theo thói quen sinh hoạt và nhu cầu cá nhân của gia chủ. Chúng ta cũng có thể hiểu ngôi nhà thông minh là một hệ thống chỉnh thể mà trong đó, tất cả các thiết bị điện tử gia dụng đều được liên kết với thiết bị điều khiển trung tâm và có thể phối hợp với nhau để cùng thực hiện một chức năng. Các thiết bị này có thể tự đưa ra cách xử lý tình huống được lập trình trước, hoặc là được điều khiển và giám sát từ xa.

Giải pháp nhà thông minh sẽ biến những món đồ điện tử bình thường trong ngôi nhà trở nên thông minh và gần gũi với người dùng hơn, chúng được kiểm soát thông qua các thiết bị truyền thông như điều khiển từ xa, điện thoại di động... ngôi nhà thông minh đơn giản nhất có thể được hình dung bao gồm một mạng điều khiển liên kết một số lượng cố định các thiết bị điện, điện tử gia dụng trong ngôi nhà và chúng được điều khiển thông qua một chiếc điều khiển từ xa. Chỉ với kết nối đơn giản như trên cũng đủ để hài lòng một số lượng lớn các cá nhân có nhu cầu nhà thông minh ở mức trung bình.

Vậy liệu nhà thông minh có làm thay đổi các thói quen vốn đã rất gắn bó từ trước đến nay với hầu hết mọi người?.

Chúng ta đều biết phần lớn căn hộ từ trung bình đến cao cấp đều sử dụng các loại điều khiển từ xa để điều khiển máy lạnh, ti vi...còn lại phần lớn các thiết bị khác như hệ thống đèn, bình nước nóng lạnh...phải điều khiển bằng tay. Những việc như vậy đôi lúc sẽ đem lại sự bất tiện, khi mà chúng ta mong muốn có một sự tiện nghi và thoải mái hơn, vừa có thể tận hưởng nằm trên giường coi ti vi vừa có thể kiểm soát được hệ thống các thiết bị trong nhà chỉ với một chiếc smartphone hay máy tính bảng.

Các thành phần cơ bản trong hệ thống nhà thông minh:

- Hệ thống quản lý chiếu sáng
- Hệ thống kiểm soát vào ra
- Hệ thống giải trí đa phương tiện

Chương 2. CƠ SỞ LÝ THUYẾT

- Hệ thống quản lý cấp điện, nước, gas
- Hệ thống điều hòa không khí, kiểm soát môi trường
- Hệ thống các công tắc điều khiển trạng thái
- Hệ thống mạng, xử lý trung tâm và sự kết hợp hoạt động

2.1.4. Lý thuyết về canh tác rau sạch

2.1.4.1. Ánh sáng ảnh hưởng đến sự phát triển của cây rau

Là yếu tố quan trọng nhất cho sự sinh trưởng và sự phát triển của rau. Ánh sáng mặt trời là nguồn năng lượng duy nhất, vô tận để cây xanh quang hợp, biến các chất vô cơ, nước và khí cacbonic thành hợp chất hữu cơ tích lũy trong lá, hoa, quả, củ... phục vụ cho nhu cầu sống của con người và các động vật.

Các loài rau khác nhau có nhu cầu về ánh sáng không giống nhau: các loại rau trồng vào mùa hè yêu cầu độ chiếu sáng mạnh, thời gian chiếu sáng dài 12 – 14 giờ/ngày. Rau trồng vào mùa đông yêu cầu cường độ ánh sáng yếu và thời gian chiếu sáng từ 8 đến 12 giờ/ngày.

Ngày nay, ngoài ánh sáng Mặt trời, người ta còn dùng hệ thống đèn huỳnh quang hay đèn led để bổ sung ánh sáng cho rau trồng trong nhà có mái che.

Như chúng ta biết ánh sáng mặt trời là tổng hợp của ánh sáng trắng trong khi cây trồng chỉ hấp thụ hai dải ánh sáng chính làm dải ánh sáng màu xanh dương (425nm-475nm) và dải ánh sáng đỏ (620nm-730nm) cho việc quang hợp. Vì vậy nếu lựa chọn đèn led để thay thế cho ánh sáng mặt trời chúng ta nên lựa chọn loại đèn có bước sóng từ 350 – 800nm.

Theo nghiên cứu, thêm 10% ánh sáng màu xanh có bước sóng từ 400 đến 500 nm cùng ánh sáng màu đỏ có bước sóng đỏ 660nm (theo tỷ lệ 70% đỏ và 30% xanh) vào đèn LED thì sẽ giúp rau diếp phát triển mạnh mẽ trong khi củ cải và rau cải bó xôi lại khó phát triển, có sản lượng thấp hơn.

Việc bổ sung thêm 24% ánh sáng màu lục có bước sóng từ 500 đến 600 nm vào đèn LED có ánh sáng màu đỏ và màu xanh sẽ giúp tăng cường sự phát triển cho rau diếp.

Chương 2. CƠ SỞ LÝ THUYẾT

Còn đối với loại cây cải ngọt, có thể phát triển mạnh mẽ nhất ở ánh sáng màu đỏ có bước sóng 660nm và ánh sáng màu xanh có bước sóng 430nm (tỷ lệ 60% đỏ và 40% xanh).

2.4.4.2. Nhiệt độ

Là yếu tố quan trọng nhất trong sinh trưởng và sự phát triển của cây rau. Nhiệt độ chính là yếu tố tạo nên các vùng khí hậu khác nhau trên trái đất và từ đó có các tập đoàn rau riêng biệt cho từng vùng. Mỗi loài rau đòi hỏi có nhiệt độ thích hợp để sống.

Nhiệt độ còn ảnh hưởng đến sự phát triển, sự nở hoa, chất lượng sản phẩm, khả năng bảo quản, thời gian ngủ của hạt và ảnh hưởng đến sự phát triển của sâu bệnh trên các loại rau.

2.4.4.3. Độ ẩm

Độ ẩm trong không khí, trong đất có tác động đến các giai đoạn sinh trưởng của cây như sự nảy mầm của hạt, sự ra hoa, kết hạt, thời gian chín của quả, chất lượng rau, sản lượng, sinh trưởng sinh dưỡng, phát sinh sâu bệnh và bảo quản hạt giống.

Nhiệt độ và độ ẩm có quan hệ mật thiết với nhau và có tác động lớn đến sinh trưởng, tái sinh của nhiều loài rau, đặc biệt là trong sản xuất hạt giống.

2.4.4.4. Đất

Là nơi bộ rễ rau phát triển, giữ chặt cây. Rau cần đất tốt, có chế độ dinh dưỡng cao. Bộ rễ của các loài rau nói chung ăn nồng trong khoảng 25 – 30cm nên tính chịu hạn, chịu nóng kém, do đó đất trồng rau phải là chân đất cao, dễ tiêu nước. Có độ pH phù hợp với từng loại rau: các loại cải bao, su lơ, xà lách, đậu bắp, hành tỏi, cần tây chịu được độ pH = 5,5 – 6,7; các loại đậu, cà rốt, cà, dưa chuột, ớt, cải củ, bí, su hào có độ pH = 5,5 – 6,8; khoai tây, dưa hấu pH = 5,0 – 6,8.

PH là kí hiệu chỉ độ chua, độ kiềm của đất. Biểu hiện bằng nồng độ ion H^+ trong môi trường. Độ chua của đất được chia ra: pH – 4 rất chua; 5 chua; 6 hơi chua; 7 trung bình; 7,5 kiềm yếu; 8 và trên 8 là kiềm. Ở Việt Nam phần lớn đất đồi

Chương 2. CƠ SỞ LÝ THUYẾT

trọc, đất bạc màu là đất chua, có độ pH = 4. Đất trũng, đất lầy thụt cũng là đất chua pH -4-5; đất trồng trọt tốt bón phân, tro nhiều pH = 4-7. Phần lân cây ưa đất trung tính, pH trên dưới 7.

2.4.4.5. Chất khoáng

Rau là cây trồng ngắn ngày nhưng sản lượng lại rất lớn, có loại đạt 20 – 60 tấn/ha, nên rau cần lượng chất dinh dưỡng rất lớn. Các chất dinh dưỡng này rau lấy từ đất không đủ, nên người trồng rau phải bổ sung bằng các loại phân bón. Dù là rau ăn lá, rau ăn củ, rau ăn quả cũng cần đầy đủ các chất dinh dưỡng cơ bản là đạm (N), lân (P), kali (K) và một số nguyên tố vi lượng khác.

2.4.4.6. Nước tưới

Thành phần hóa học trong rau chủ yếu là nước, chiếm đến 90%, do đó lượng nước cây cần lấy vào trong tự nhiên là rất lớn. Nước còn là môi trường sống của một số loại rau (rau muống...). Nước nơi chất khoáng hòa tan được rẽ hút vào nuôi cây. Nước cũng là môi trường để pha các loại thuốc trừ sâu bệnh. Nên muốn có năng suất rau cao, cần đảm bảo lượng nước đủ theo nhu cầu của từng loại rau.

Nơi trồng rau phải gần nguồn nước sạch, nước được lấy từ giếng khoan, ao hồ có nước lưu thông. Không được dùng nước thải sinh hoạt, nước từ các bệnh viện, các khu công nghiệp thải ra chưa được qua hệ thống xử lý. Nước tưới bẩn không làm rau bị chết mà chính các yếu tố độc hại ấy tích lại trong rau, gây ngộ độc cho người tiêu dùng.

2.2. CƠ SỞ LÝ THUYẾT VỀ MẠCH ĐIỆN

Trong phạm vi đề tài có sử dụng dụng chip STM32 và SIM808 dùng để làm mạch trung tâm giao tiếp với server và giao tiếp với mạch công suất. Board Arduino Mega2560 dùng để thu thập dữ liệu và điều khiển thiết bị.

Arduino Mega2560 đã và đang được sử dụng rất rộng rãi trên thế giới, và ngày càng chứng tỏ được sức mạnh của chúng thông qua vô số ứng dụng độc đáo của người dùng trong cộng đồng nguồn mở(open-source). Trong khi đó Module SIM 808 là Module được thiết kế tối ưu với giá thành thấp và phục vụ chủ yếu cho việc giám sát, điều khiển các thiết bị từ xa thông qua GMS/GPRS. Sau đây chúng ta

Chương 2. CƠ SỞ LÝ THUYẾT

sẽ tìm hiểu từng khối module cũng như hoạt động của chúng để phục vụ cho việc thực hiện đề tài.

2.2.1. Vi điều khiển STM32F103C8T6

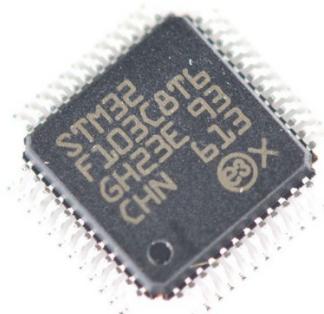
STM32 là một trong những dòng chip phổ biến của ST với nhiều họ thông dụng như F0,F1,F2,F3,F4..... STM32F103 thuộc họ F1 với lõi là ARM COTEX M3. STM32F103 là vi điều khiển 32 bit, tốc độ tối đa là 72Mhz. Giá thành cũng khá rẻ so với các loại vi điều khiển có chức năng tương tự. Mạch nạp cũng như công cụ lập trình khá đa dạng và dễ sử dụng.

Một số ứng dụng chính: dùng cho driver để điều khiển ứng dụng, điều khiển ứng dụng thông thường, thiết bị cầm tay và thuốc, máy tính và thiết bị ngoại vi chơi game, GPS cơ bản, các ứng dụng trong công nghiệp, thiết bị lập trình PLC, biến tần, máy in, máy quét, hệ thống cảnh báo, thiết bị liên lạc nội bộ...

Phần mềm lập trình: có khá nhiều trình biên dịch cho STM32 như IAR Embedded Workbench, Keil C... Ở đây mình sử dụng Keil C nên các bài viết sau mình chỉ đề cập đến Keil C.

Thư viện lập trình: có nhiều loại thư viện lập trình cho STM32 như: STM32snippets, STM32Cube LL, STM32Cube HAL, Standard Peripheral Libraries, Mbed core. Mỗi thư viện đều có ưu và khuyết điểm riêng, ở đây mình xin phép sử dụng Standard Peripheral Libraries vì nó ra đời khá lâu và khá thông dụng, hỗ trợ nhiều ngoại vi và cũng dễ hiểu rõ bản chất của lập trình.

Mạch nạp: có khá nhiều loại mạch nạp như : ULINK, J-LINK , CMSIS-DAP, STLINK... ở đây mình sử dụng Stlink vì giá thành khá rẻ và debug lỗi cũng tốt.



Chương 2. CƠ SỞ LÝ THUYẾT

Hình 2.3: Chip STM32F103C8T6

Cấu hình chi tiết của STM32F103C8T6:

- ARM 32-bit Cortex M3 với clock max là 72Mhz.
- Bộ nhớ:
 - 64 kbytes bộ nhớ Flash(bộ nhớ lập trình).
 - 20kbytes SRAM.
- Clock, reset và quản lý nguồn.
 - Điện áp hoạt động 2.0V -> 3.6V.
 - Power on reset(POR), Power down reset(PDR) và programmable voltage detector (PWD).
 - Sử dụng thạch anh ngoài từ 4Mhz -> 20Mhz.
 - Thạch anh nội dùng dao động RC ở mode 8Mhz hoặc 40khz.
 - Sử dụng thạch anh ngoài 32.768khz được sử dụng cho RTC.
- Trong trường hợp điện áp thấp:
 - Có các mode :ngủ, ngừng hoạt động hoặc hoạt động ở chế độ chờ.
 - Cấp nguồn ở chân Vbat bằng pin để hoạt động bộ RTC và sử dụng lưu trữ data khi mất nguồn cấp chính.
- 2 bộ ADC 12 bit với 9 kênh cho mỗi bộ.
 - Khoảng giá trị chuyển đổi từ 0 – 3.6V.
 - Lấy mẫu nhiều kênh hoặc 1 kênh.
 - Có cảm biến nhiệt độ nội.
- DMA: bộ chuyển đổi này giúp tăng tốc độ xử lý do không có sự can thiệp quá sâu của CPU.
 - 7 kênh DMA.
 - Hỗ trợ DMA cho ADC, I2C, SPI, UART.
- 7 timer.
 - 3 timer 16 bit hỗ trợ các mode IC/OC/PWM.

Chương 2. CƠ SỞ LÝ THUYẾT

- o 1 timer 16 bit hỗ trợ để điều khiển động cơ với các mode bảo vệ như ngắn input, dead-time..
 - o 2 watchdog timer dùng để bảo vệ và kiểm tra lỗi.
 - o 1 sysTick timer 24 bit đếm xuống dùng cho các ứng dụng như hàm Delay....
- Hỗ trợ 9 kênh giao tiếp bao gồm:
 - o 2 bộ I2C(SMBus/PMBus).
 - o 3 bộ USART(ISO 7816 interface, LIN, IrDA capability, modem control).
 - o 2 SPIs (18 Mbit/s).
 - o 1 bộ CAN interface (2.0B Active)
 - o USB 2.0 full-speed interface
 - Kiểm tra lỗi CRC và 96-bit ID.

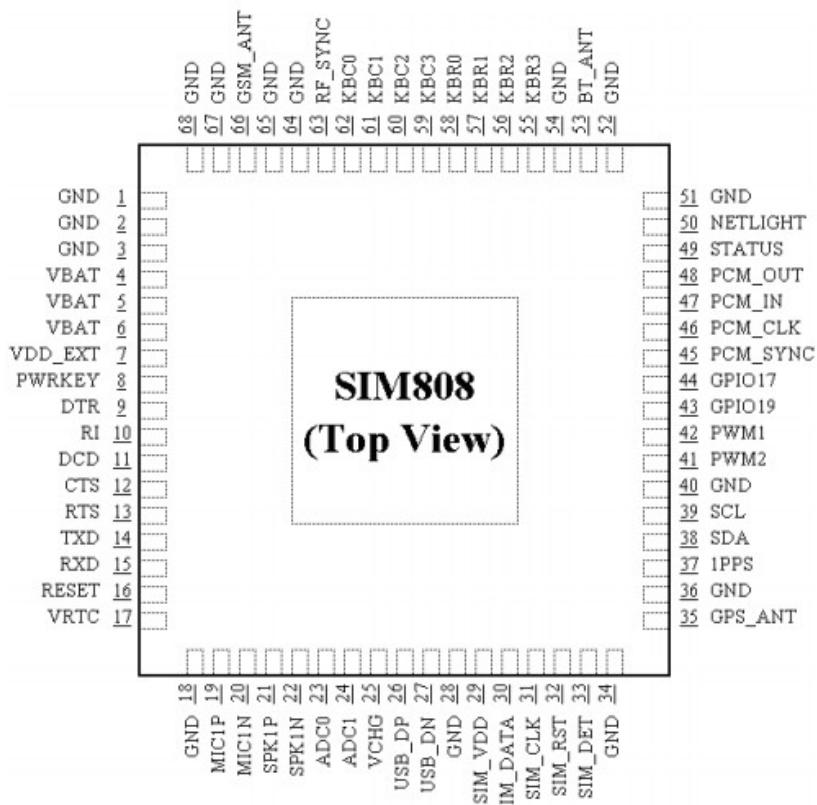
2.2.2. Module SIM808

Module SIM808 là bản nâng cấp của module SIM908 cũ với đầy đủ các tính năng GSM/GPRS/GPS nhưng có độ chính xác và độ ổn định cao hơn.



Hình 2.4: Module SIM808

Chương 2. CƠ SỞ LÝ THUYẾT



Hình 2.5: Sơ đồ chân của SIM808

Thông số kỹ thuật

- Dải băng tần 850/900/1800/1900 MHz
- Hỗ trợ GPRS 12/10
- Trạm GPRS loại B
- Tuân thủ GSM phase 2/2 + - class 4 (2 W @ 850/900 MHz) -Class 1 (1 W @ 1800/1900 MHz)
- Kích thước: 24x24x2.6 mm
- Trọng lượng: 3,3g
- Điều khiển thông qua các lệnh AT (3GPP TS 27.007, 27.005 và SIMCOM(Các lệnh AT nâng cao)
- Dải điện áp cung cấp 3,4 ~ 4,4V
- Tiêu thụ điện năng thấp
- Nhiệt độ hoạt động: -40~85°C
- Thông số kỹ thuật cho dữ liệu GPRS:

Chương 2. CƠ SỞ LÝ THUYẾT

- GPRS class 12: tối đa 85,6 kbps (đường xuống / đường lên)
- Hỗ trợ PBCCH
- Sơ đồ mã hóa CS 1, 2, 3, 4
- Ngăn xếp PPP
- USSD
- Thông số kỹ thuật cho SMS qua GSM / GPRS:
- Trò tới điểm MO và MT
- SMS
- Chế độ văn bản và PDU

Tính năng phần mềm:

- Giao thức 0710 MUX
- Giao thức TCP / UDP nhúng
- FTP / HTTP
- MMS
- POP3 / SMTP
- DTMF
- Phát hiện gây nhiễu
- Bản ghi âm
- SSL
- BT 3.0 (tùy chọn)
- TTS CN (tùy chọn)
- Nhúng AT (tùy chọn)
- Khả năng tương thích:
 - Giao diện lệnh di động AT
 - Đặc điểm kỹ thuật cho GPS:
 - Loại máy thu
 - 22 theo dõi / 66 mua lại
 - Mã kênh GPS L1 C / A

BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN

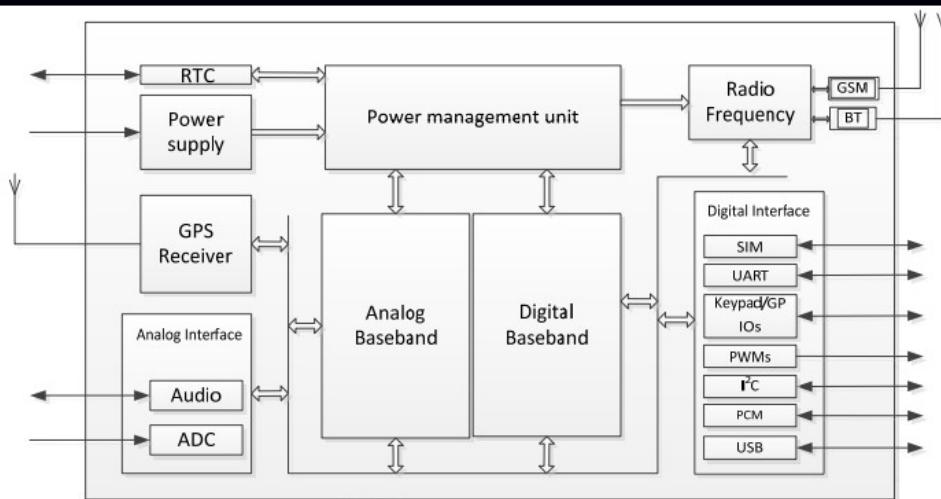
Chương 2. CƠ SỞ LÝ THUYẾT

- Độ nhạy
 - Theo dõi: -165 dBm
 - Khởi động nguội: -148 dBm
- Thời gian sửa lỗi đầu tiên
 - Bắt đầu lạnh: 32s (typ.)
 - Bắt đầu nóng: <1s
 - Khởi động: 3s
- Độ chính xác Vị trí nằm ngang: <2,5m CEP

Giao diện

- 68 miếng đệm bao gồm
- Giao diện âm thanh analog
- Giao diện PCM (tùy chọn)
- Giao diện SPI (tùy chọn)
- Sao lưu RTC
- Giao diện nối tiếp
- Giao diện USB
- Giao diện với SIM ngoài 3V / 1.8V
- Giao diện bàn phím
- GPIO
- ADC
- Tấm ăng ten GSM
- Tấm ăng ten BT
- Bảng ăng ten GPS

Chương 2. CƠ SỞ LÝ THUYẾT



Hình 2.6: Sơ đồ chức năng SIM808

2.2.3. Board Arduino Mega2560

Arduino Mega 2560 là sản phẩm tiêu biểu cho dòng mạch Mega là dòng bo mạch có nhiều cải tiến so với Arduino Uno (54 chân digital IO và 16 chân analog IO). Đặc biệt bộ nhớ flash của MEGA được tăng lên một cách đáng kể, gấp 4 lần so với những phiên bản cũ của UNO R3. Điều này cùng với việc trang bị 3 timer và 6 cổng interrupt khiến bo mạch Mega hoàn toàn có thể giải quyết được nhiều bài toán hóc búa, cần điều khiển nhiều loại động cơ và xử lý song song nhiều luồng dữ liệu số cũng như tương tự

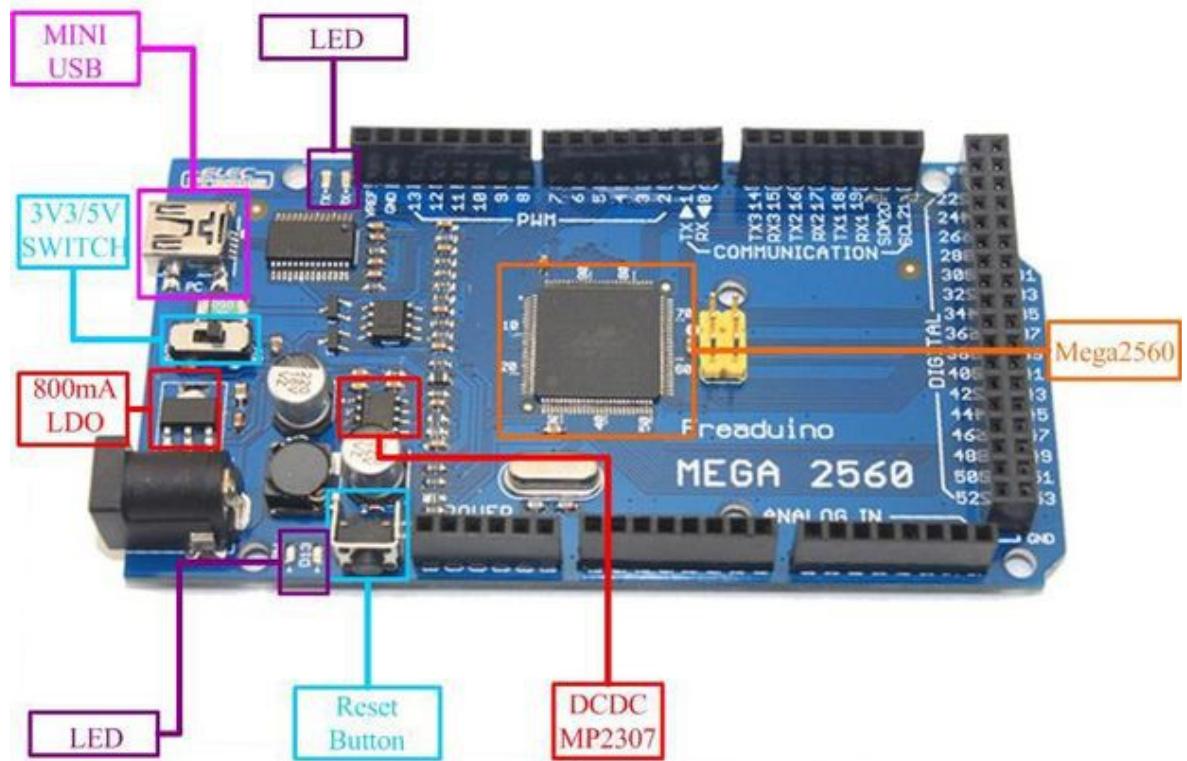
Ngoài việc phát triển được ưu tiên, việc kế thừa cũng được đặc biệt lưu ý. Trên mạch MEGA các chân digital vẫn từ 0-13, analog từ 0-5 và các chân nguồn tương tự thiết kế của UNO. Do vậy chúng ta dễ dàng phát triển nghiên cứu theo kiểu gấp ghép module từ Arduino UNO bê sang Arduino mega. Ngoài ra, ở phiên bản này, các nhà thiết kế đã mạnh dạn thay đổi thiết kế. Để có thêm được nhiều vùng nhớ và nhiều chân IO hơn, một con chip khác đã thay thế cho Atmega1280. Theo dòng phát triển của vi điều khiển nhúng, những dự án lớn cần nhiều dung lượng flash hơn. Do vậy, Arduino Mega 2560 ra đời với sứ mệnh giải những bài toán như thế.

Arduino Mega được thiết kế cho nhiều dự án khó. Với 54 chân I/O kỹ thuật số, 16 chân analog, cùng không gian khá rộng để bạn có thể tích hợp các mạch điện tử của dự án của bạn lên đó.

Tính năng nổi bật của Arduino Mega 2560 R3

Chương 2. CƠ SỞ LÝ THUYẾT

Arduino Mega 2560 là board mạch vi điều khiển, xây dựng dựa trên Atmega 2560. Nó có 54 chân I/O (trong đó có 15 chân có thể sử dụng làm chân output với chức năng PWM), 16 chân đầu vào Analog, 4 UART, 1 thạch anh 16Mhz, 1 cổng USB, 1 jack nguồn, 1 header, 1 nút nhấn reset. Nó chứa mọi thứ cần thiết hỗ trợ cho người lập trình vi điều khiển, đơn giản chỉ việc kết nối nó với máy tính bằng cable USB là có thể bắt đầu học tập. Mạch Arduino 2560 sử dụng tương thích với phần lớn các Shield của Arduino UNO



Hình 2.7: Mặt trước của Mega2560 và các khối cơ bản

Thông số kỹ thuật:

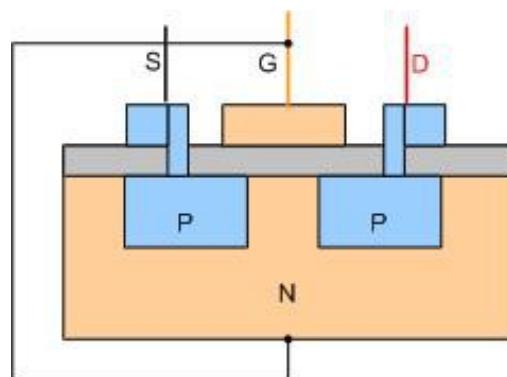
- Vi điều khiển: ATmega2560
- Điện áp hoạt động: 5V
- Điện áp đầu vào (được đề nghị): 7-12V
- Điện áp đầu vào (giới hạn): 6-20V
- Số lượng chân I/O: 54 (trong đó có 15 cung cấp sản lượng PWM)
- Số lượng chân Input Analog: 16

Chương 2. CƠ SỞ LÝ THUYẾT

- Dòng điện DC mỗi I/O: 20 mA
- Dòng điện DC với chân 3.3V: 50 mA
- Bộ nhớ flash: 256 KB trong đó có 8 KB sử dụng bởi bộ nạp khởi động
- SRAM: 8 KB
- EEPROM: 4 KB
- Tốc độ đồng hồ: 16 MHz
- Chiều dài: 101,52 mm
- Bề rộng: 53,3 mm
- Cân nặng: 37 g

2.2.4. Cơ sở lý thuyết về MOSFET

Mosfet là Transistor hiệu ứng trường (Metal Oxide Semiconductor Field Effect Transistor) là một Transistor đặc biệt có cấu tạo và hoạt động khác với Transistor thông thường mà ta đã biết. Mosfet thường có công suất lớn hơn rất nhiều so với BJT. Đối với tín hiệu 1 chiều thì nó coi như là 1 khóa đóng mở. Mosfet có nguyên tắc hoạt động dựa trên hiệu ứng từ trường để tạo ra dòng điện, là linh kiện có trở kháng đầu vào lớn thích hợp cho khuếch đại các nguồn tín hiệu yếu.



Hình 2.8: Cấu tạo của Mosfet ngược Kênh N

G : Gate gọi là cực cổng

S : Source gọi là cực nguồn

D : Drain gọi là cực máng

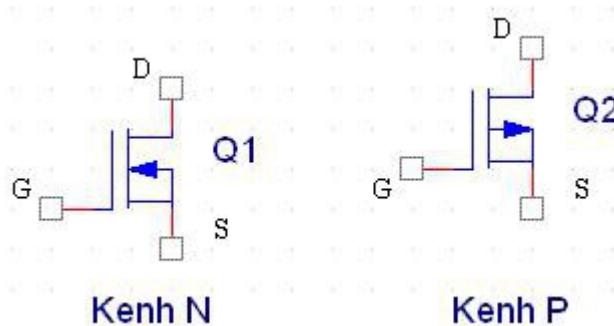
Chương 2. CƠ SỞ LÝ THUYẾT

Trong đó : G là cực điều khiển được cách ly hoàn toàn với cấu trúc bán dẫn còn lại bởi lớp điện môi cực mỏng nhưng có độ cách điện cực lớn dioxit-silic (SiO_2). Hai cực còn lại là cực gốc (S) và cực máng (D). Cực máng là cực đón các hạt mang điện.

Mosfet có điện trở giữa cực G với cực S và giữa cực G với cực D là vô cùng lớn , còn điện trở giữa cực D và cực S phụ thuộc vào điện áp chênh lệch giữa cực G và cực S (UGS)

Khi điện áp UGS = 0 thì điện trở RDS rất lớn, khi điện áp UGS > 0 => do hiệu ứng từ trường làm cho điện trở RDS giảm, điện áp UGS càng lớn thì điện trở RDS càng nhỏ.

Ký hiệu



Hình 2.9: Ký hiệu 2 loại mosfet

Hiện nay các loại mosfet thông dụng bao gồm 2 loại:

N-MOSFET: chỉ hoạt động khi nguồn điện Gate là zero, các electron bên trong vẫn tiếp hành hoạt động cho đến khi bị ảnh hưởng bởi nguồn điện Input.

P-MOSFET: các electron sẽ bị cut-off cho đến khi gia tăng nguồn điện thế vào ngõ Gate

Qua đó ta thấy Mosfet này có chân tương đương với Transistor

- + Chân G tương đương với B
- + Chân D tương đương với chân C
- + Chân S tương đương với E

Chương 2. CƠ SỞ LÝ THUYẾT

Nguyên lý hoạt động

Mosfet hoạt động ở 2 chế độ đóng và mở. Do là một phần tử với các hạt mang điện cơ bản nên Mosfet có thể đóng cắt với tần số rất cao. Nhưng mà để đảm bảo thời gian đóng cắt ngắn thì vấn đề điều khiển lại là vấn đề quan trọng .

Mạch điện tương đương của Mosfet . Nhìn vào đó ta thấy cơ chế đóng cắt phụ thuộc vào các tụ điện ký sinh trên nó.

+ Đối với kênh P : Điện áp điều khiển mở Mosfet là Ugs0. Dòng điện sẽ đi từ S đến D

+ Đối với kênh N : Điện áp điều khiển mở Mosfet là Ugs >0. Điện áp điều khiển đóng là Ugs<=0. Dòng điện sẽ đi từ D xuống S.

Do đảm bảo thời gian đóng cắt là ngắn nhất người ta thường : Đối với Mosfet Kênh N điện áp khóa là Ugs = 0 V còn Kênh P thì Ugs=~0

2.2.5. Chuẩn truyền thông UART

UART : Universal Asynchronous Receiver/Transmitter, là kiểu truyền thông tin nối tiếp không đồng bộ, UART thường được dùng trong máy tính công nghiệp, truyền thông, vi điều khiển, hay một số các thiết bị truyền tin khác.

Một số thông số:

Baud rate (tốc độ Baud) : Khi truyền nhận không đồng bộ để hai module hiểu được nhau thì cần quy định một khoảng thời gian cho 1 bit truyền nhận. Theo định nghĩa thì tốc độ baud là số bit truyền trong một giây.

Frame (khung truyền): Do kiểu truyền thông nối tiếp này rất dễ mất dữ liệu nên ngoài tốc độ, khung truyền cũng được cài đặt từ ban đầu để tránh bớt sự mất mát dữ liệu này. Khung truyền quy định số bit trong mỗi lần truyền, các bít báo như start, stop, các bit kiểm tra như parity, và số bit trong một data.

Bit Start: Là bit bắt đầu trong khung truyền Bit này nhằm mục đích báo cho thiết bị nhận biết quá trình truyền bắt đầu, trên AVR bit Start có trạng thái là 0.

Chương 2. CƠ SỞ LÝ THUYẾT

Data: Dữ liệu cần truyền Data không nhất thiết phải 8 bit, có thể là 5, 6, 7, 8, 9 . Trong UART bit LSB được truyền đi trước, Bit MSB được truyền đi sau.

Parity bit: Là bit kiểm tra dữ liệu đúng không. Có 2 loại parity: chẵn (even parity), lẻ (odd parity). Parity chẵn là bit parity thêm vào để số bit 1 trong data + parity = chẵn, parity lẻ là bit parity thêm vào để số bit 1 trong data + parity = lẻ. Bit Parity là không bắt buộc nên có thể dùng hoặc không.

Stop: là bit báo cáo kết thúc khung truyền, thường là mức 5V và có thể có 1 hoặc 2 bit stop .

2.3. CƠ SỞ LÝ THUYẾT VỀ CẢM BIẾN

2.3.1. Cảm biến SHT10

Cảm biến độ ẩm, nhiệt độ đất SHT10 có vỏ bảo vệ thường được sử dụng trong nông nghiệp với các ứng dụng cần độ bền, độ chính xác và độ ổn định cao, cảm biến có cấu tạo gồm cảm biến SHT10 phía trong, bên ngoài là lớp vỏ bảo vệ cảm biến khỏi các tác động vật lý từ môi trường như bụi, nước,... tuy nhiên vẫn đo được chính xác độ ẩm và nhiệt độ.

Cảm biến độ ẩm, nhiệt độ đất SHT10 có vỏ bảo vệ V2 sử dụng trong môi trường đất, tức là bạn có thể chôn cảm biến dưới đất để đo các thông số độ ẩm nhiệt độ của đất, tuy nhiên các bạn cần lưu ý để chôn cảm biến ở khoảng cách hợp lý để không dẫn đến tình trạng nước ngập úng tràn vào làm hư cảm biến.



Hình 2.10: Cảm biến đo nhiệt độ, độ ẩm SHT10

Thông số kỹ thuật:

Chương 2. CƠ SỞ LÝ THUYẾT

- Cảm biến độ ẩm, nhiệt độ đất SHT10 có vỏ bảo vệ V2
- Cảm biến phía trong: SHT10
- Điện áp sử dụng: 3 - 5VDC
- Khoảng nhiệt độ đo được: -40 ~ 120 độ C, sai số 0.5 độ C
- Khoảng độ ẩm đo được: 0 ~100% RH, sai số 4.5% RH
- Có sẵn trở treo 10K.
- Độ dài: 77mm
- Đường kính: 16mm
- Dây dẫn: 1m

Sơ đồ dây:

VCC - Màu nâu - cấp nguồn 3 ~ 5VDC.

GND - Màu đen - cấp nguồn 0VDC ~ Mass.

DATA - Màu vàng.

CLK - Màu xanh dương.

Cách lấy giá trị cảm biến

Cảm biến SHT10 đo độ ẩm và nhiệt độ ở 2 chế độ:

Chế độ 1: đo nhiệt độ 12 bít, độ ẩm 8 bít.

Chế độ 2: đo nhiệt độ 14 bít, đo độ ẩm 12 bít. Đây là chế độ thường được sử dụng

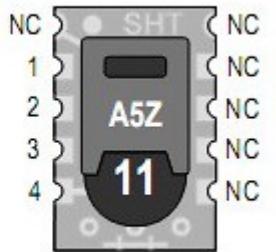
Sai số độ ẩm: ± 3

Sai số nhiệt độ: ± 0.4

- Điện áp hoạt động: 2.4 - 5.5

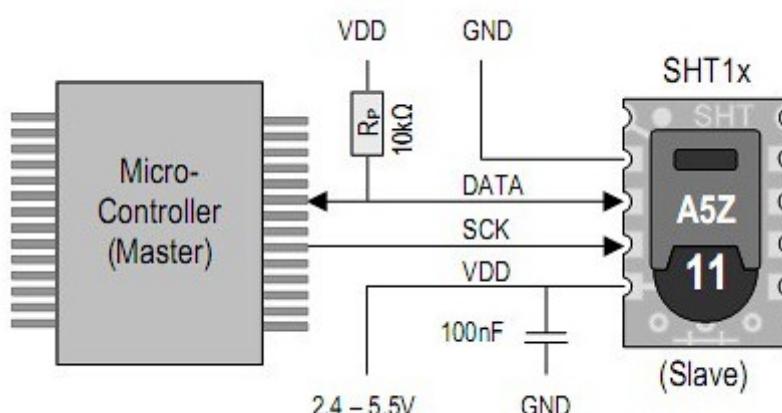
Chương 2. CƠ SỞ LÝ THUYẾT

Pin	Name	Comment	
1	GND	Ground	
2	DATA	Serial Data, bidirectional	
3	SCK	Serial Clock, input only	
4	VDD	Source Voltage	
NC	NC	Must be left unconnected	



Hình 2.11: Cấu hình chân của SHT10

Sơ đồ đấu dây cảm biến SHT10



Hình 2.12: Sơ đồ kết nối với vi điều khiển

Quá trình gửi dữ liệu xuống SHT10 gồm 4 bước:

Bước 1: Truyền xung Start:

- Chân DATA =1
- SCK có xung(từ thấp lên cao) sau đó VDK kéo chân DATA xuống thấp
- SCK có xung tiếp theo, VDK giữ DATA ở thấp. Khi đó chân SHT10 biết là VDK muốn giao tiếp với nó.
- Ta đưa chân DATA lên 1. Chuẩn bị quá trình gửi lệnh xuống SHT

Bước 2: Gửi lệnh xuống SHT

- Gửi 0: cho chân DATA xuống 0, sau đó kích xung SCK từ Thấp lên Cao. Khi đó SHT10 sẽ đọc tín hiệu tại chân DATA và nhận giá trị
- Gửi 1: cho chân DATA lên 1, sau đó kích xung SCK từ thấp lên cao.
- VDK gửi lệnh 8 bit xuống SHT10
3 bit đầu là 0
5 bit sau xác định yêu cầu của VDK với SHT10

Chương 2. CƠ SỞ LÝ THUYẾT

Bảng 2.1: Lệnh gửi xuống VDK của SHT10

Yêu cầu	Mã gửi
Đo giá trị nhiệt độ	00011
Đo giá trị độ ẩm	00101
Đọc trạng thái thanh ghi	00111
Ghi trạng thái thanh ghi	00110

Bước 3: Kiểm tra lỗi:

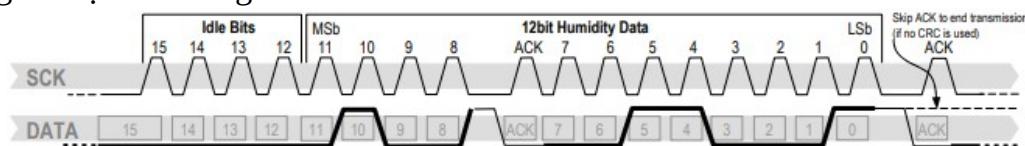
- Sau khi gửi đủ 8 bit lệnh xuống SHT10, thì chúng ta kiểm tra lỗi
- Cho chân DATA lên 1, sau đó chuyển chế độ là chân INPUT
- Kích 1 xung từ thấp lên cao tại SCK
- Đọc lại chân DATA
 - Nếu =0: Gửi lệnh OK
 - Nếu =1 : Gửi lệnh có lỗi.
- Nếu gửi lệnh ok, ta chờ 1 thời gian để SHT10 đo nhiệt độ và gửi lại data cho VDK.

Lúc này chân DATA vẫn là chân INPUT.

SHT10 sẽ kéo chân DATA lên 1 trong quá trình đo chờ kết quả.

Quá trình đọc kết quả:

- Quá trình đọc dữ liệu bắt đầu khi chân DATA được SHT10 kéo xuống thấp. Khi đó SHT thông báo với VDK xử lý xong (đo xong nhiệt độ, hoặc độ ẩm)
- Dữ liệu kết quả độ ẩm hay nhiệt độ SHT10 gửi tới VDK đều dạng 16 bit. Cứ sau 8 bit được gửi từ SHT10, VDK lại truyền xuống 1 bit ACK = 0 tới SHT10. Khi nhận được bit này, SHT10 sẽ truyền byte tiếp theo lên.
- Ngoài 16 bit dữ liệu ra, SHT10 còn gửi lại 1 byte CheckSum. Ta có thể sử dụng hoặc không
- Nếu không sử dụng thì sau khi nhận được 8 bit byte thấp ta cho tín hiệu ACK =1.
- Khi đó SHT10 sẽ chuyển sang chế độ Sleep. Và chuẩn bị cho lệnh tiếp theo.
- Khung dữ liệu SHT10 gửi lên VDK:



Hình 2.13: Khung dữ liệu của SHT10 gửi lên VDK

Chương 2. CƠ SỞ LÝ THUYẾT

Đọc byte cao dữ liệu(MSB):

- Chân DATA VĐK là đầu vào
- Khi kích xung SCK từ thấp lên cao, sau đó đọc dữ liệu từ chân DATA
- Bit 0: DATA = 0;
- Bit 1 : DATA =1;
- Khi đọc được 8 bit. Chuyển chân DATA của VĐK thành chân Output
- VĐK kéo chân DATA = 0;
- Có xung kích SCK từ thấp lên cao. (truyền bit ACK).

Lặp lại quá trình trên để đọc byte thấp.

Byte CheckSum có thể đọc hoặc không. Nếu đọc thì chú ý bit ACK =1. Để SHT10 chuyển vào trạng thái Sleep. Chuẩn bị quá trình làm việc tiếp theo.

Bước 4: Xử lý kết quả:

Giá trị nhiệt độ:

$$T = d_1 + d_2 \cdot SOT$$

Trong đó: SOT= 0.01 khi đo với giá trị độ C, 0.018 với giá trị độ F. Giá trị d1,d2 tra trong bảng dưới.

Bảng 2.2. Bảng quy đổi giá trị đo

VDD	D1(độ C)	D1(độ F)
5V	-40.1	-40.2
4V	-39.8	-39.6
3.5V	-39.7	-39.5
3V	-39.6	-39.3
2.5V	-39.4	-39.9

Giá trị độ ẩm đọc được gọi là : SORH

Giá trị độ ẩm tuyến tính:

$$RH_{\text{linear}} = c_1 + c_2 \cdot SORH + c_3 \cdot SORH^2 (\%RH)$$

Các giá trị c1, c2, c3 được dựa trên bảng sau.

BỘ MÔN TỰ ĐỘNG ĐIỀU KHIỂN

Chương 2. CƠ SỞ LÝ THUYẾT

Bảng 2.3: Các giá trị để tính độ ẩm

SO _{rh}	C1	C2	C3
12 bit	-2.0468	0.0367	-0.5955
8 bit	-2.0468	0.5872	-0.084

Do giá trị độ ẩm còn có liên hệ với nhiệt độ tại thời điểm đo. Ta có công thức tính giá trị độ ẩm chính xác.

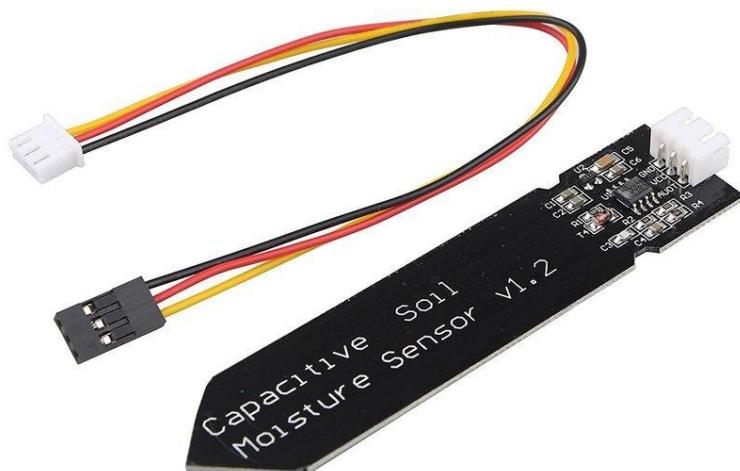
$$RH_{true} = (T_{\circ C} - 25) \cdot (t_1 + t_2 \cdot SO_{RH}) + RH_{linear}$$

Bảng 2.4: Quy đổi độ ẩm từ nhiệt độ

SO _{RH}	t ₁	t ₂
12 bit	0.01	0.00008
8 bit	0.01	0.00128

2.3.2. Cảm biến độ ẩm đất điện dung SEN0193

Cảm biến độ ẩm đất điện dung SEN0193 đầu ra analog Cảm biến độ ẩm đất arduino là loại cảm biến độ ẩm đất mới - cảm biến điện dung với độ bền và tuổi thọ cao hơn nhiều các loại cảm biến điện trở thông thường. Cảm biến đo độ ẩm đất này hoạt động với điện áp 3.3V đến 5.5V, đặc biệt nó hoạt động ngay cả trên bảng điều khiển Arduino 3.3V.



Chương 2. CƠ SỞ LÝ THUYẾT

Hình 2.14: Cảm biến độ ẩm đất điện dung

Thông số kỹ thuật của cảm biến độ ẩm của đất:

- Điện áp hoạt động: 3.3 ~ 5.5 VDC
- Điện áp đầu ra: 0 ~ 3.0 VDC
- Giao diện: PH2.54-3P
- Kích thước: 98 x 23mm (LxW)
- Khối lượng: 20 gram

Nguyên lý hoạt động của cảm biến độ ẩm đất kiểu điện dung:

Mạch cảm biến sử dụng mạch tạo xung NE555 với tụ điện trong mạch để tính tần số là hai bản cực được thiết kế về hai mặt PCB. Khi ở môi trường ẩm, hơi nước bao quanh 2 bản cực như môi trường điện môi. Mật độ hơi nước càng lớn thì hằng số điện môi càng cao dẫn tới ụ dung của mạch tạo tần số tăng nên tần số sẽ giảm xuống ($f = 1/(R \cdot C)$ - tham khảo datasheet của NE555) và ngược lại. Tần số này đưa qua mạch lọc thông thấp sẽ xuất mức điện áp tương ứng với giá trị độ ẩm đất ở chân AOUT.

Cách hiệu chỉnh:

Bước 1: Lau thật sạch cảm biến để khô, cấp nguồn 5V và đo được giá trị ADC (bằng các cổng ADC của Arduino, stm32, stm8,...) được giá trị khô (tương ứng với 0%) (Giả sử là 520)

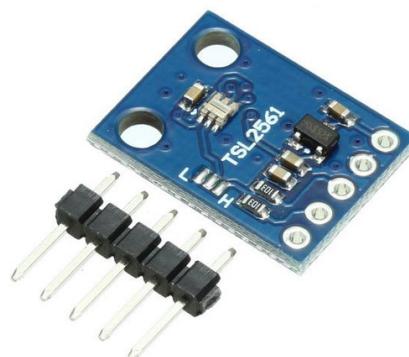
Bước 2: Cho cảm biến vào trong cốc nước sạch, đo được giá trị ADC ở chân Aout là giá trị bão hòa (100%) (Giả sử là 200) - Độ ẩm càng tăng thì áp ra càng giảm

Bước 3: Giá trị độ ẩm từ 0 đến 100% tương ứng là từ 520 đến 200 là 320 đơn vị. Lưu ý: Giá trị % này không tương ứng với giá trị độ ẩm đất tiêu chuẩn. Trong thực tế cần dùng máy móc đo kiểm chính xác hoặc dựa vào kinh nghiệm để lấy giá trị các mức khô- vừa- ướt để đưa vào các thuật toán điều khiển.

Chương 2. CƠ SỞ LÝ THUYẾT

2.3.3. Cảm biến cường độ ánh sáng TSL2561

Cảm biến cường độ ánh sáng (lux) TSL2561 được sử dụng để đo cường độ ánh sáng thường, hồng ngoại theo đơn vị lux với độ ổn định và độ chính xác cao, cảm biến có ADC nội và bộ tiền xử lý nên giá trị được trả ra là giá trị trực tiếp cường độ ánh sáng lux mà không phải qua bất kỳ xử lý hay tính toán nào thông qua giao tiếp I2C .



Hình 2.15: Cảm biến đo cường độ ánh sáng TSL2561

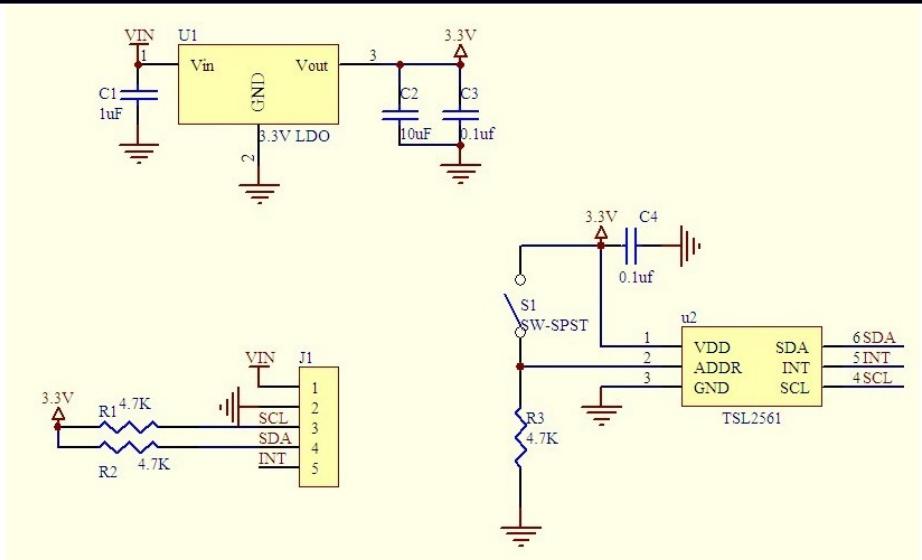
Thông số kỹ thuật:

- IC chính: TSL2561
- Nguồn: 3.3~5VDC
- Dòng điện tiêu thụ: 0.6mA
- Đo được cường độ ánh sáng thường và hồng ngoại (IR).
- Giao tiếp: I2C mức TTL 3.3~5VDC
- Khoảng đo: 0.1 ~ 40.000 Lux
- Kích cỡ: 20 x 14mm

Một số ví dụ về độ rọi của ánh sáng:

- Vào buổi tối : 0.001 - 0.02 Lux
- Ánh trăng : 0.02 - 0.3 lux
- Trời nhiều mây trong nhà : 5 - 50 lux
- Trời nhiều mây ngoài trời : 50 - 500 lux
- Trời nắng trong nhà : 100 - 1000 lux
- Ánh sáng cần thiết để đọc sách: 50 - 60 lux

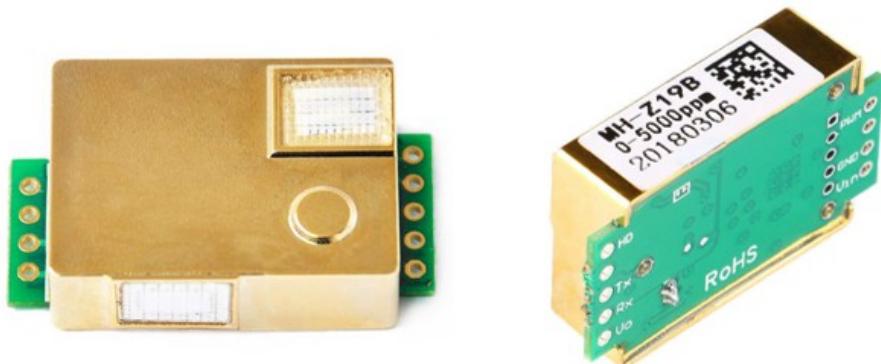
Chương 2. CƠ SỞ LÝ THUYẾT



Hình 2.16: Sơ đồ chân của cảm biến

2.3.4. Cảm biến nồng độ khí CO2 MH-Z19

Cảm biến khí CO2 MH-Z19 được sử dụng để đo nồng độ khí CO2 trong môi trường, cảm biến sử dụng phương pháp đo non-dispersive infrared (NDIR) nên có độ nhạy, độ chính xác cao. Ngoài ra cảm biến còn tích hợp thêm cảm biến nhiệt độ, tín hiệu cảm biến được xuất ra theo cả hai dạng là Digital (UART) và Analog rất dễ đo đạc, giao tiếp và sử dụng.



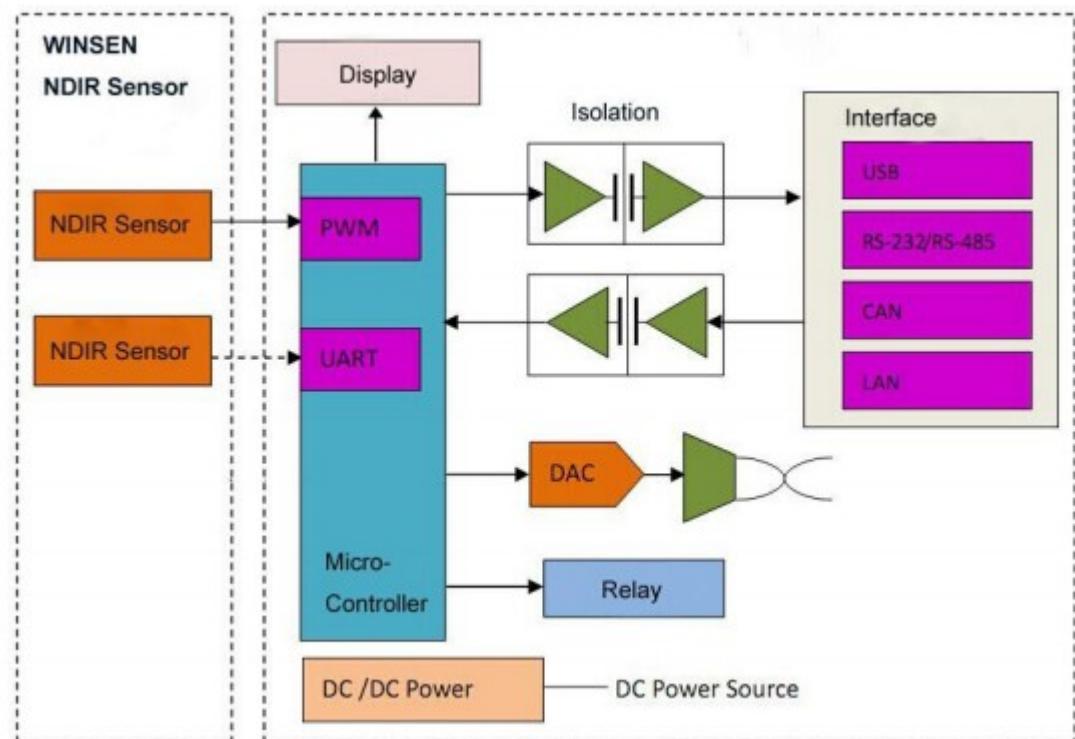
Hình 2.17: Cảm biến đo nồng độ khí CO2

THÔNG SỐ KỸ THUẬT

- Điện áp làm việc: 4.5~5.5VDC
- Dòng định mức: < 85 mA
- Interface level: 3.3 V

Chương 2. CƠ SỞ LÝ THUYẾT

- Measuring range: 0~5%VOL optional
- Tín hiệu đầu ra: PWM / UART (TTL-3V3)/ Analog 0.4-2VDC
- Preheat time: 3min
- Response Time: T90 < 90s
- Nhiệt độ hoạt động: 0°C ~ 50°C
- Độ ẩm hoạt động: 0~95%RH
- Khối lượng: 15g
- Lifespan: >5 year
- Kích thước: 57.5×34.7×16mm [L×W×H]



Hình 2.18: Sơ đồ khối cảm biến CO₂

Cảm biến đo nồng độ CO₂ có thể lấy được giá trị bằng nhiều phương pháp như ADC, qua truyền thông UART, qua kênh PWM.

2.4. CƠ SỞ LÝ THUYẾT VỀ TRUYỀN THÔNG KHÔNG DÂY GSM/GPRS/GPS

Hệ thống thông tin di động toàn cầu (tiếng Anh: Global System for Mobile Communications; tiếng Pháp: Groupe Spécial Mobile; viết tắt: GSM) là một công nghệ dùng cho mạng thông tin di động. Dịch vụ GSM được sử dụng bởi hơn 2 tỷ người trên 212 quốc gia và vùng lãnh thổ. Các mạng thông tin di động GSM cho phép có thể roaming với nhau do đó những máy điện thoại di động GSM của các mạng GSM khác nhau ở có thể sử dụng được nhiều nơi trên thế giới.

Chương 2. CƠ SỞ LÝ THUYẾT

GSM là chuẩn phổ biến nhất cho điện thoại di động (ĐTDĐ) trên thế giới. Khả năng phủ sóng rộng khắp nơi của chuẩn GSM làm cho nó trở nên phổ biến trên thế giới, cho phép người sử dụng có thể sử dụng ĐTDĐ của họ ở nhiều vùng trên thế giới. GSM khác với các chuẩn tiên thân của nó về cả tín hiệu và tốc độ, chất lượng cuộc gọi. Nó được xem như là một hệ thống ĐTDĐ thế hệ thứ hai (second generation, 2G). GSM là một chuẩn mở, hiện tại nó được phát triển bởi 3rd Generation Partnership Project (3GPP) Đứng về phía quan điểm khách hàng, lợi thế chính của GSM là chất lượng cuộc gọi tốt hơn, giá thành thấp và dịch vụ tin nhắn. Thuận lợi đối với nhà điều hành mạng là khả năng triển khai thiết bị từ nhiều người cung ứng. GSM cho phép nhà điều hành mạng có thể sẵn sàng dịch vụ ở khắp nơi, vì thế người sử dụng có thể sử dụng điện thoại của họ ở khắp nơi trên thế giới.

Dịch vụ vô tuyến gói đa năng (GPRS - General Packet Radio Service) là một công nghệ kỹ thuật gói, dựa trên GSM. Lợi ích chính của GPRS là nguồn tài nguyên vô tuyến được truy xuất chỉ khi dữ liệu thật sự được gửi đi giữa trạm di động và mạng, được phát triển dựa trên các thành phần của mạng GSM hiện có, vì vậy tiết kiệm được chi phí đồng thời sử dụng được tài nguyên tiết kiệm, giảm nghẽn mạch (chi phí để nâng cấp mạng GSM lên GPRS chỉ bằng 1/10 chi phí nâng cấp từ mạng GSM lên GPRS). Hơn nữa, GPRS còn nâng cao được chất lượng dịch vụ dữ liệu, tăng độ tin cậy. GPRS áp dụng nguyên tắc gói vô tuyến để truyền gói dữ liệu hiệu quả hơn giữa trạm di động GSM và mạng dữ liệu gói bên ngoài. Chuyển mạch gói chia dữ liệu ra thành các gói nhỏ rồi truyền riêng rẽ sau đó tập hợp lại ở phía thu.

Một người sử dụng GPRS có thể sử dụng đến 8 khe thời gian để đạt tốc độ tối đa hơn 100kbit/s. Tuy nhiên đây là tốc độ đỉnh, nếu nhiều người cùng sử dụng thì tốc độ bit sẽ thấp hơn.

Trong khi hệ thống GSM sử dụng chuyển mạch kênh để truyền thoại, thì hệ thống GPRS sử dụng chuyển mạch gói, nhưng đều theo chuẩn GSM. Khi một bản tin được truyền đi, nó được chia thành nhiều gói. Khi những gói này đến chỗ thu nó được tập hợp lại cho ra bản tin ban đầu. Tất cả các gói này đều được lưu trong bộ đệm dữ liệu.

Chương 2. CƠ SỞ LÝ THUYẾT

Những gói dữ liệu từ MS có thể dùng nhiều kênh vô tuyến khác nhau trong suốt quá trình truyền.

MS trong hệ thống GPRS có thể chỉ được dùng cho chuyển mạch kênh, hoặc cho cả chuyển mạch kênh và chuyển mạch gói.

2.5. LÝ THUYẾT VỀ WEB SERVER VÀ TRUYỀN NHẬN DỮ LIỆU QUA CHUỖI JSON

2.5.1. Web Server

Web server dịch ra tiếng Việt nghĩa là máy chủ. Web server là máy tính lớn được kết nối với tập hợp mạng máy tính mở rộng. Đây là một dạng máy chủ trên internet mỗi máy chủ là một IP khác nhau và có thể đọc các ngôn ngữ như file *.htm và *.html... Tóm lại máy chủ là kho để chứa toàn bộ dữ liệu hoạt động trên internet mà nó được giao quyền quản lý.

Web server phải là một máy tính có dung lượng lớn, tốc độ rất cao để có thể lưu trữ vận hành tốt một kho dữ liệu trên internet. Nó sẽ điều hành trơn chu cho một hệ thống máy tính hoạt động trên internet, thông qua các cổng giao tiếp riêng biệt của mỗi máy chủ. Các web server này phải đảm bảo hoạt động liên tục không ngừng nghỉ để duy trì cung cấp dữ liệu cho mạng lưới máy tính của mình. Để hiểu hơn web server chính là máy chủ, được thiết kế với các siêu tính năng dùng để chứa các dữ liệu cho một phần mạng lưới máy tính trên internet. Tất cả những hoạt động dịch vụ trên internet nào đều phải có máy chủ này mới hoạt động được.

Đôi nét về web server

- Web server có thể xử lý dữ liệu và cung cấp thông tin đến máy khách thông qua các máy tính cá nhân trên môi trường Internet qua giao thức HTTP, giao thức được thiết kế để gửi các file đến trình duyệt Web, và các giao thức khác. (Ví dụ: khi các bạn truy cập vào trang web vinahost.vn máy chủ sẽ cung cấp đến các bạn tất cả dữ liệu về trang web đó thông qua lệnh giao tiếp)
- Máy tính nào cũng có thể là một máy chủ nếu cài đặt lên nó một chương trình phần mềm Server Software và sau đó kết nối vào Internet.

Chương 2. CƠ SỞ LÝ THUYẾT

- Phần mềm Web Server Software cũng giống như các phần mềm khác, nó dùng để cài đặt và chạy trên bất kì máy tính nào đáp ứng đủ yêu cầu về bộ nhớ. Nhờ có chương trình này mà người sử dụng có thể truy cập đến các thông tin của trang Web từ một máy tính khác ở trên mạng
- Khi là SEO chúng ta thường gặp các máy chủ nhỏ, máy chủ ảo và thông thường chúng ta hay thuê máy chủ dạng VPS hay Hosting để lưu giữ liệu trang web của mình.

2.5.2. Chuỗi JSON

JSON là chữ viết tắt của **Javascript Object Notation**, đây là một dạng dữ liệu tuân theo một quy luật nhất định mà hầu hết các ngôn ngữ lập trình hiện nay đều có thể đọc được, bạn có thể sử dụng lưu nó vào một file, một record trong CSDL rất dễ dàng. JSON có định dạng đơn giản, dễ dàng sử dụng và truy vấn hơn XML rất nhiều nên tính ứng dụng của nó hiện nay rất là phổ biến, theo tôi thì trong tương lai tới trong các ứng dụng sẽ sử dụng nó là đa số.

Cú pháp của JSON rất đơn giản là mỗi thông tin dữ liệu sẽ có 2 phần đó là key và value, điều này tương ứng trong CSDL là tên field và giá trị của nó ở một record nào đó. Tuy nhiên nhìn qua thì đơn giản nhưng nếu ta mở xem nó ra thì có một vài điều như sau:

- Chuỗi JSON được bao lại bởi dấu ngoặc nhọn {}
- Các key, value của JSON bắt buộc phải đặt trong dấu nháy kép " ".

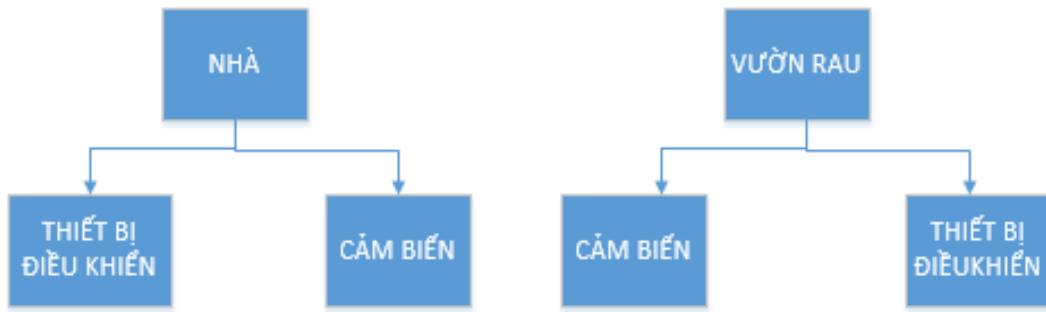
Nếu có nhiều dữ liệu (nhiều cặp key => value) thì ta dùng dấu phẩy (,) để ngăn cách

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

3.1. YÊU CẦU ĐIỀU KHIỂN

3.1.1. Sơ đồ khối phần cơ khí của mô hình

Sơ đồ khối phần cơ khí của mô hình như sau:

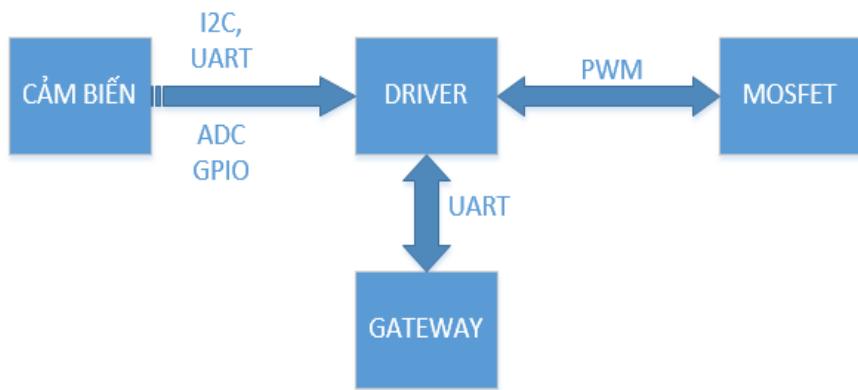


Hình 3.1: Sơ đồ khối phần cơ khí của hệ thống

Phần cơ khí của mô hình gồm 2 phần chính là mô hình nhà thông minh và mô hình vườn rau thông minh. Cả 2 mô hình được thiết kế và thi công tách biệt nhau. Tại mỗi mô hình đều được gắn các cảm biến và các thiết bị để điều khiển.

3.1.2. Sơ đồ khối phần mạch điện và lập trình của mô hình

Với yêu cầu của hệ thống đã đặt ra, sơ đồ khối phần mạch điện của hệ thống như sau:



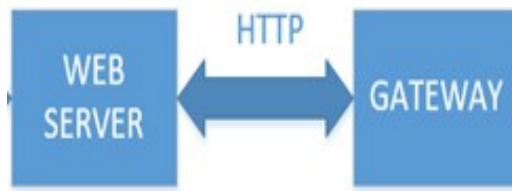
Hình 3.2: Sơ đồ khối phần mạch điện của hệ thống

Hệ thống gồm 2 khối chính:

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

- ✓ Gateway: có chức năng nhận và xử lý dữ liệu được gửi từ mạch Driver, phân tích dữ liệu sau đó thực hiện đưa lên cơ sở dữ liệu của web và điều khiển, truyền dữ liệu nhận được từ người dùng đến mạch Driver để điều khiển các mosfet ngõ ra.
- ✓ Driver: nhận và xử lý dữ liệu từ các cảm biến sau đó gửi đến Gateway qua truyền thông UART đồng thời nhận tín hiệu điều khiển từ Gateway để điều khiển các thiết bị công suất tại Driver.

3.1.3. Sơ đồ khái niệm Web Server và giao diện người dùng



Hình 3.3: Sơ đồ khái niệm Webserver

Các dữ liệu cảm biến sau khi được đưa qua GATEWAY sẽ được xử lý và đưa lên web server thông qua phương thức POST của giao thức HTTP. Khi có yêu cầu điều khiển thiết bị, Gateway sẽ quét trên server và GET giá trị của các công tắc, đưa vào chuỗi, sau đó Gateway sẽ giải mã chuỗi và gửi xuống Driver để điều khiển thiết bị.

3.2. TÍNH TOÁN THIẾT KẾ MẠCH DRIVER

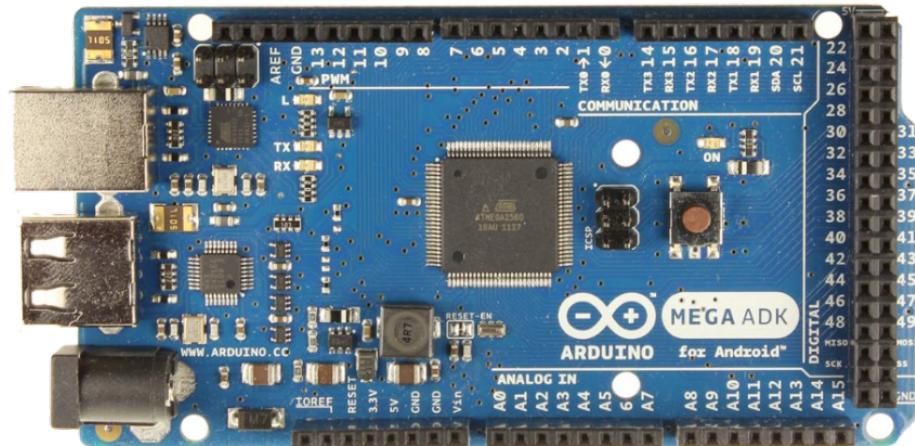
3.2.1. Tính toán thiết kế từng khái niệm cho mạch DRIVER

3.2.1.1. Khái niệm xử lý trung tâm.

Hiện nay có nhiều loại Module và vi điều khiển thông dụng trên thị trường như là: AT89S52, PIC16F887, Module STM32F4, Board Arduino UNO... Nhưng mỗi loại đều có một ưu điểm và khuyết điểm khác nhau như vi điều khiển AT89S52 có ưu điểm là giá thành rẻ nhưng khuyết điểm là không tích hợp các module như ADC, PWM và dễ bị nhiễu. Còn dòng PIC16F887 là dòng PIC thông dụng hiện nay, nó có ưu điểm là tích hợp đầy đủ các module thông dụng, khó bị nhiễu nhưng đổi lại thì có khuyết điểm là giá thành khá cao đồng thời tính tiện lợi trong sử dụng của nó không linh hoạt. Dòng Arduino là một trong những hệ thống nhúng mạnh

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

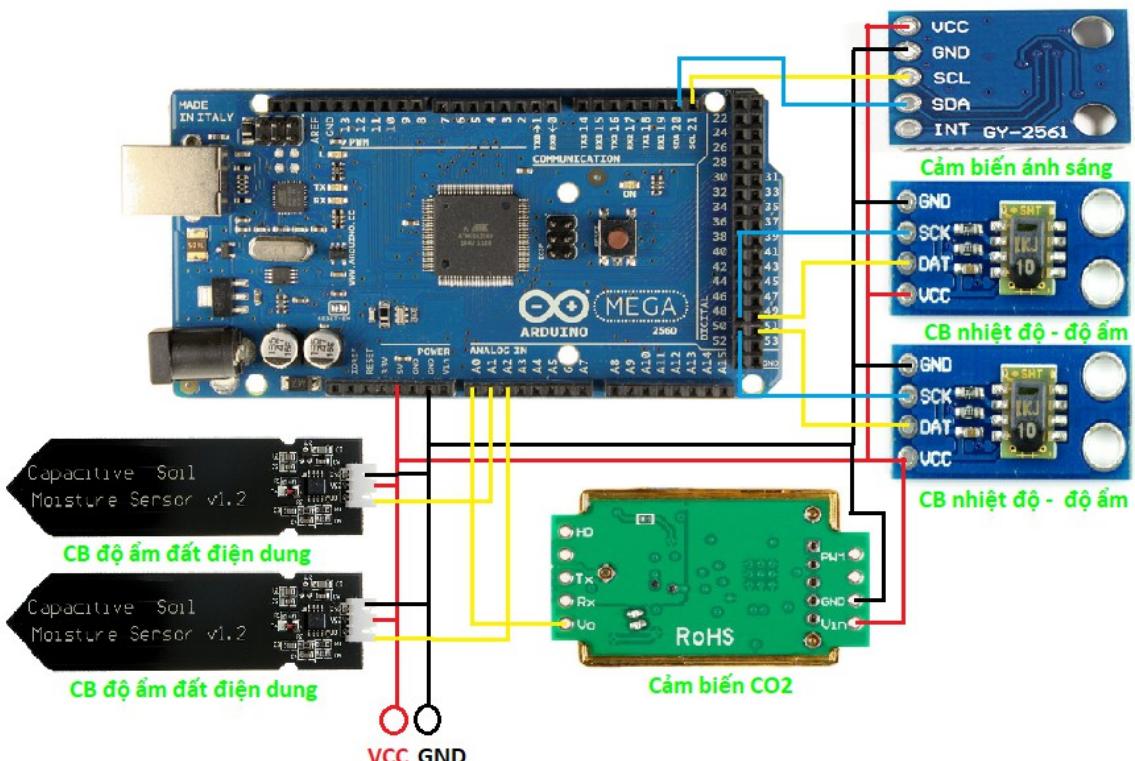
nhất hiện nay, với những ưu điểm như có những Board tích hợp linh hoạt, lập trình đơn giản, dễ sử dụng, có nhiều tính năng thông dụng. Do nhu cầu phát triển thêm các tính năng cho đề tài sau này nên nhóm quyết định chọn Board điều khiển Arduino, cụ thể ở đây là Board Arduino Mega2560.



Hình 3.4: Board Arduino Mega2560

3.2.1.2. Khối thu thập dữ liệu cảm biến

Khối cảm biến gồm các module cảm biến nhiệt độ, độ ẩm SHT10, cảm biến độ ẩm đất điện dung SEN0193, cảm biến đo cường độ ánh sáng TSL2561 và cảm biến đo nồng độ CO₂ MH-Z19.

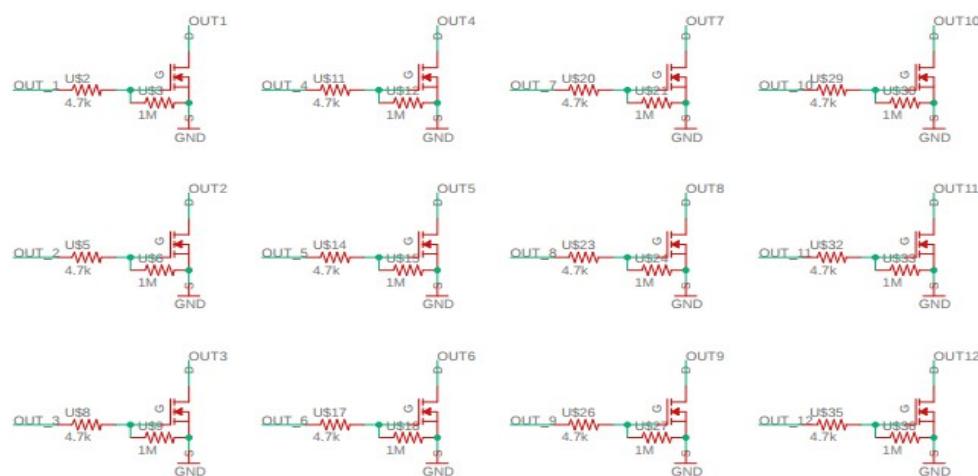


Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

Hình 3.5: Sơ đồ nối dây cảm biến

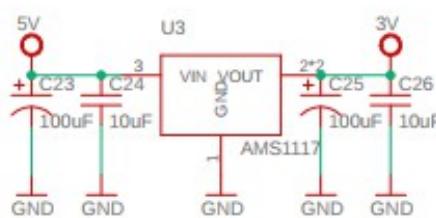
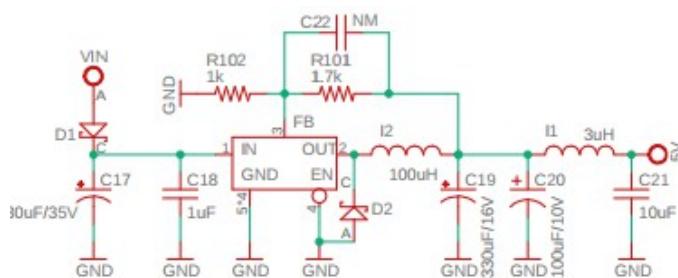
3.2.1.3. Khối công suất

Trong thực tế có rất nhiều cách để điều khiển đóng ngắt thiết bị. Để phù hợp cho hệ thống nhóm em chọn dùng điều khiển từ các chân PWM của vi điều khiển. Chân điều khiển sẽ xuất 1 xung có giá trị từ 0 tới 255, tương ứng với độ rộng xung từ 0 tới 100%. Các chân PWM của module Arduino sẽ điều khiển kích hoạt MOSFET. Các MOSFET sẽ đóng ngắt các thiết bị. Thiết bị nhóm sử dụng gồm có đèn 12V, quạt 12V, máy bơm 12V, 20 Led 12V, động cơ servo 5V.



Hình 3.6: Sơ đồ nguyên lý mạch công suất ngõ ra

3.2.1.4. Khối giảm áp



Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

Hình 3.7: Sơ đồ nguyên lý mạch giảm áp

Trong đồ án tốt nghiệp của nhóm sử dụng bộ nguồn 12V để cung cấp cho cả mạch, Mạch giảm áp cho ra điện áp 3.3VDC và 5VDC để cảm biến có thể hoạt động được.

3.2.1.5. Khối nguồn

Bảng 3.1: Dòng tiêu thụ của các linh kiện

Thiết bị	Dòng tiêu thụ
Mạch Driver	500mA
Mạch Gateway	1A
LED vườn 12V	320mA
Bơm 12V	1.25A
Quạt 12V	150mA
Đèn báo 12V	150mA
Cảm biến CO2 MHZ-19B	100mA
Cảm biến TSL2561	100mA
Cảm biến SHT10	100mA
Cảm biến SEN0193	100mA

Từ bảng cho biết dòng tiêu thụ của các linh kiện và thiết bị,

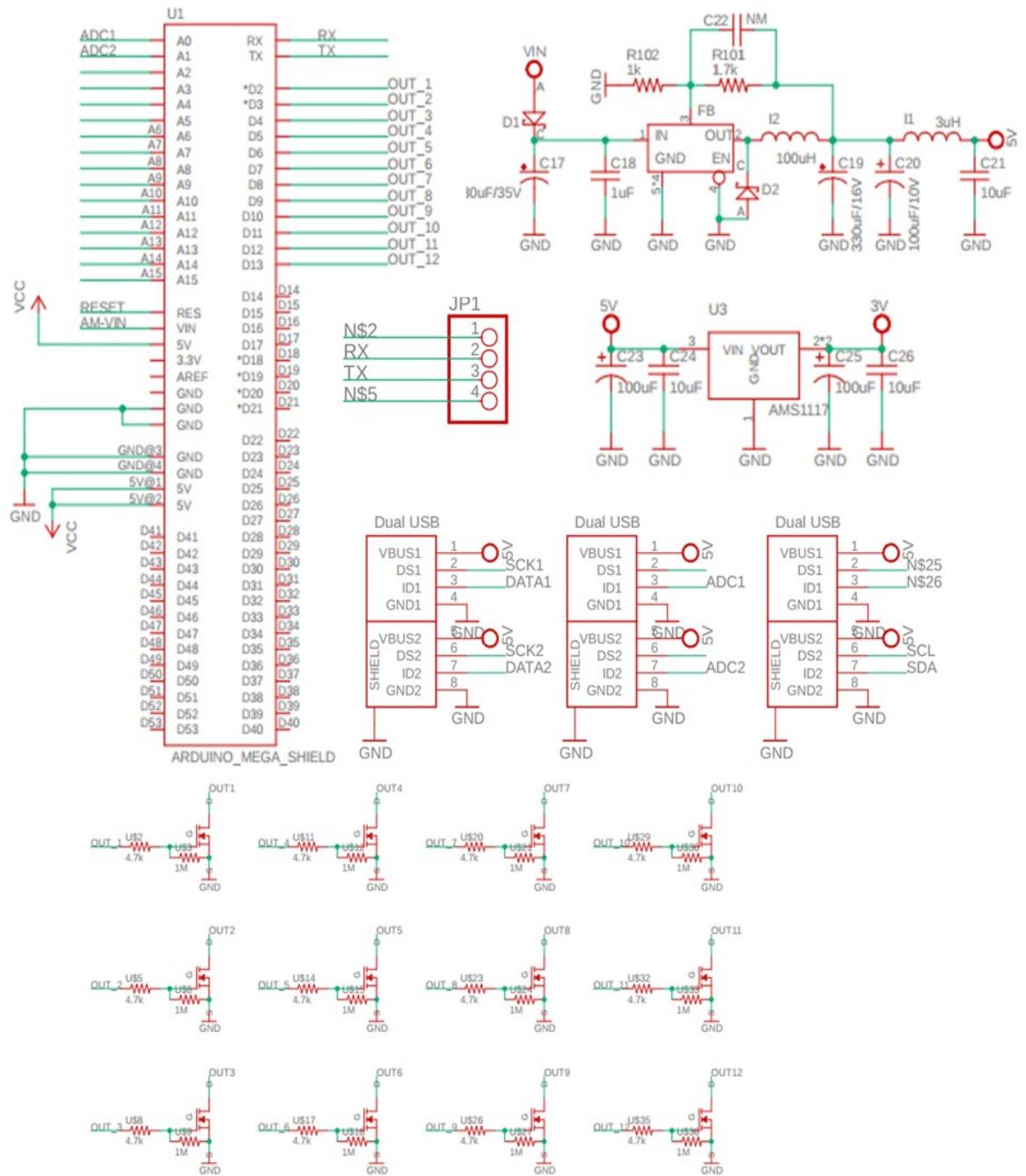
$$I_1 = I_{\text{Gateway}} + I_{\text{Driver}} + 2I_{\text{led}} + I_{\text{bơm}} + 3I_{\text{quạt}} + 3I_{\text{đèn}} + I_{\text{Tsl2561}} + I_{\text{ht10}} + I_{\text{sen0193}} + I_{\text{mhz-19b}} = 1000 + 500 + 320*2 + 1250 + 150*3 + 150*3 + 100 + 100 + 2*100 + 2*100 = 4850 \text{ mA} = 4.85 \text{ A}$$

=> Chọn nguồn 12V 5A cho cả hệ thống.

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

3.2.2 Sơ đồ nguyên lý và sơ đồ PCB mạch DRIVER

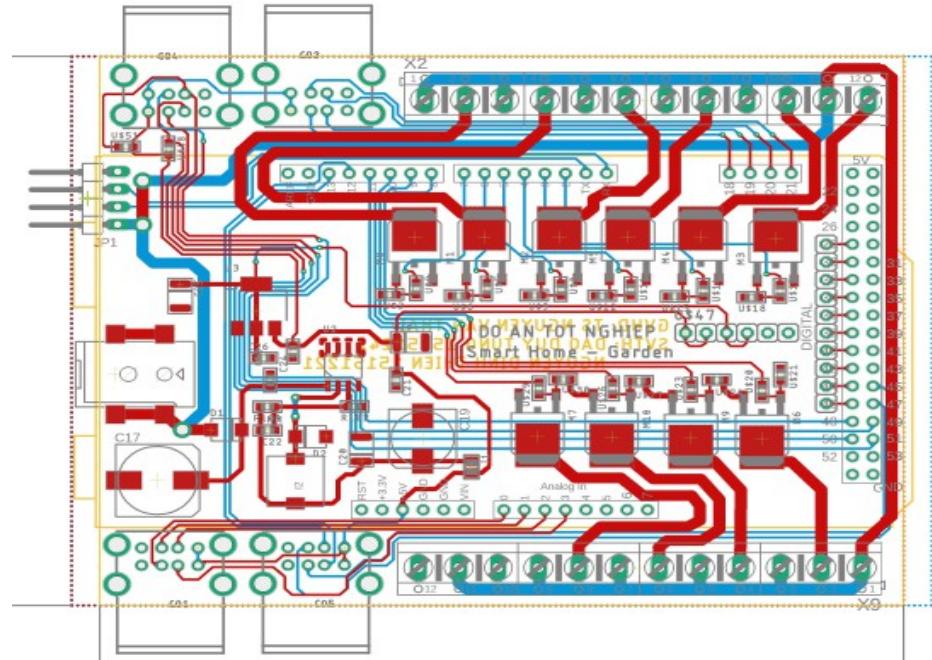
- Sơ đồ nguyên lý**



Hình 3.8: Sơ đồ nguyên lý mạch DRIVER

- Sơ đồ mạch PCB**

Mạch PCB được nhóm thiết kế và đi dây 2 lớp trên phần mềm Eagle.



Hình 3.9: Mạch PCB được đi dây trên phần mềm Eagle

3.3. TÍNH TOÁN THIẾT KẾ MẠCH GATEWAY

3.3.1. Khối vi xử lý trung tâm và giao tiếp với web server

STM32 là chip vi điều khiển 32 bit lõi Arm Cortex với cấu hình mạnh mẽ cho dù với phân khúc thấp nhất STM32F0x cũng có thể hoạt động lên tới 48Mhz, 64kB Flash, 16kB RAM, 8 bộ Timer 16 bit, 1 bộ Timer 32 bit, 10 bộ ADC 12 bit, 8 bộ USART, 2 bộ SPI, 2 bộ I2C. Giá của dòng vi điều khiển này ở trên thị trường là khá thấp so với những dòng vi điều khiển khác.. Đây là mức giá rẻ nhất so với các dòng vi điều khiển có cấu hình tương đương.



Hình 3.10: STM32

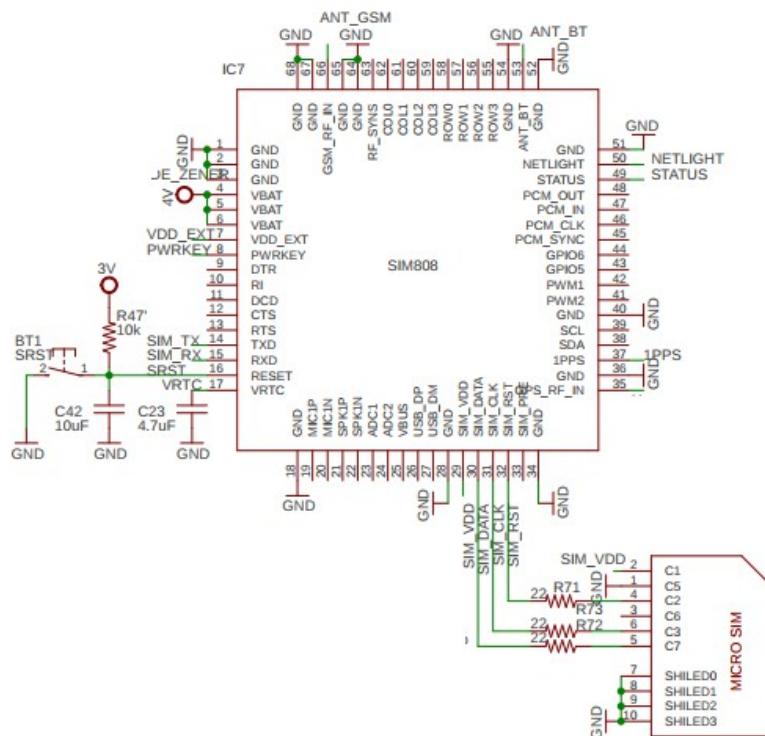


Hình 3.11: SIM808

SIM808 là bản nâng cấp của mạch SIM908 cũ với đầy đủ các tính năng GSM/GPRS/GPS nhưng có độ chính xác và độ ổn định cao hơn.

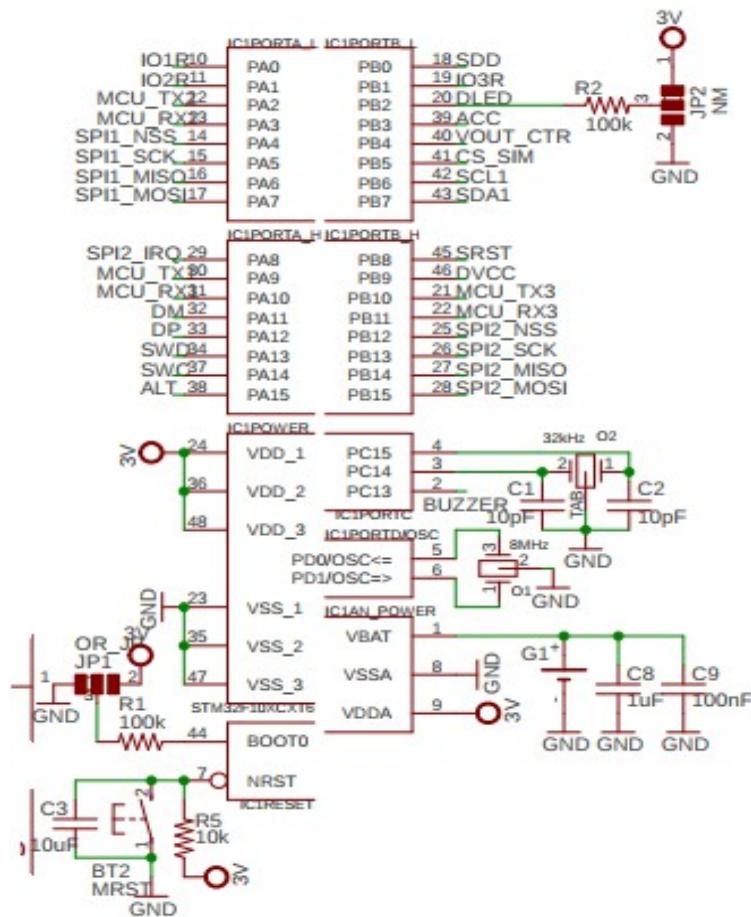
3.3.2 Sơ đồ nguyên lý và sơ đồ PCB của mạch GATEWAY

Sơ đồ nguyên lý khối vi xử lý trung tâm

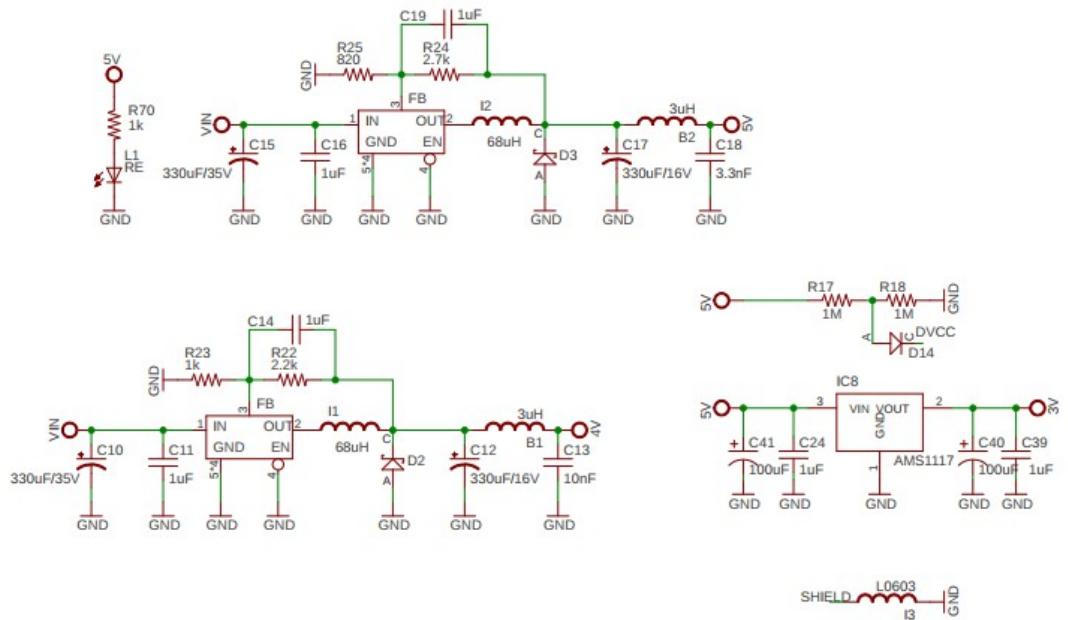


Hình 3.12: Sơ đồ nguyên lý khối SIM808 và thẻ MicroSIM

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

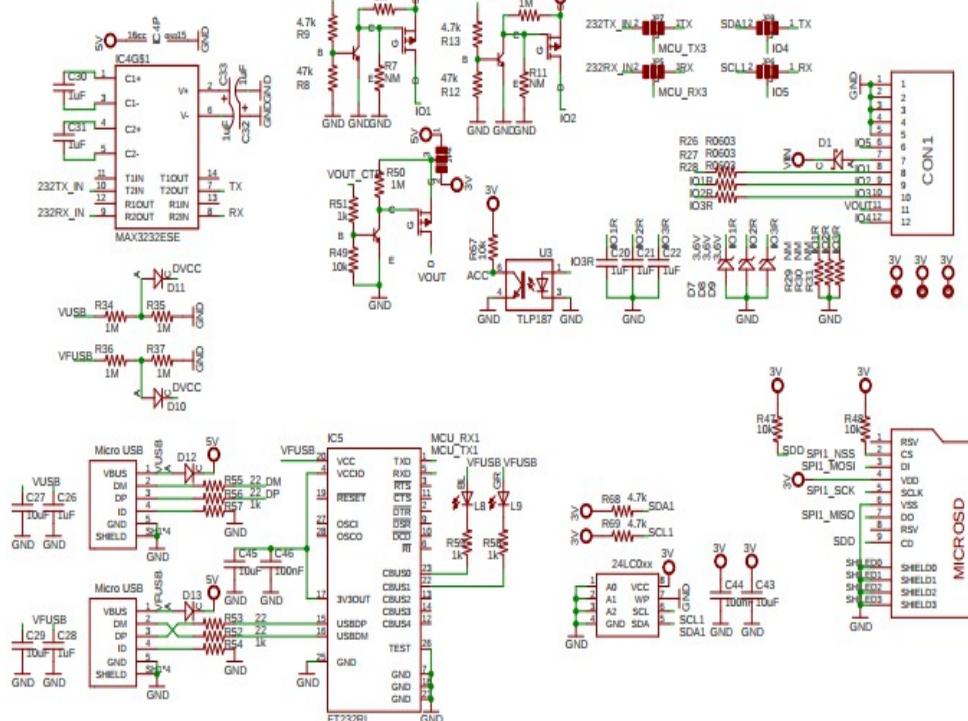


Hình 3.13: Sơ đồ nguyên lý vi điều khiển STM32



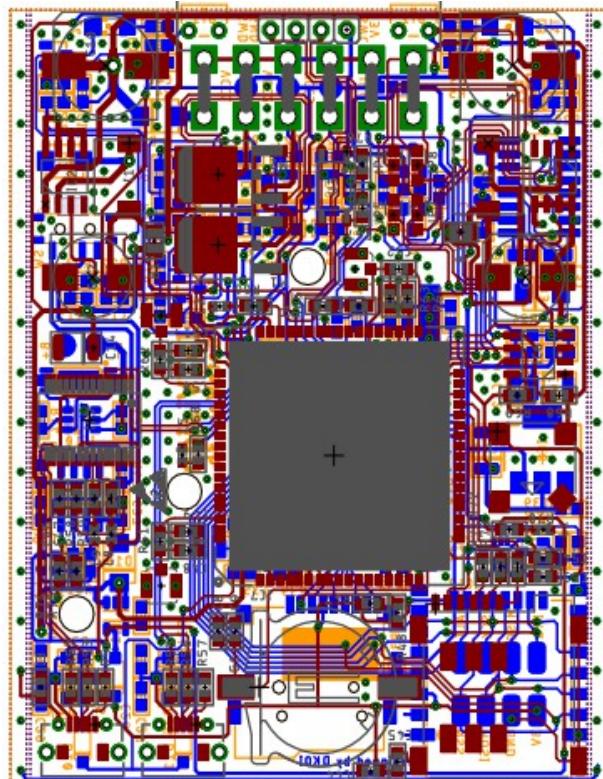
Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

Hình 3.14: Sơ đồ nguyên lý khối nguồn



Hình 3.15: Sơ đồ ra chân RS232, MicroSD, GPS

- Sơ đồ mạch PCB**



Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

Hình 3.16: Sơ đồ mạch in PCB GATEWAY

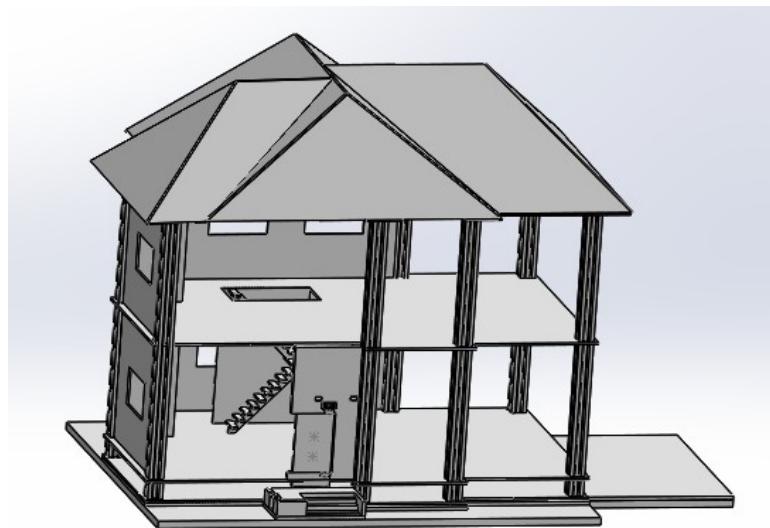
3.4. TÍNH TOÁN THIẾT KẾ CƠ KHÍ CỦA MÔ HÌNH

Các thiết kế cơ khí của nhóm được vẽ bằng phần mềm Solidworks 2017, sau khi dựng 3D sẽ được xuất 2D trên bản vẽ và thi công.

3.4.1. Thiết kế mô hình nhà thông minh

Thiết kế mô hình nhà thông minh trên phần mềm Solidworks 2017. Mô hình nhà thông minh được thiết kế giống như 1 ngôi nhà ngoài thực tế, nhà có cấu trúc 2 tầng.

- Chiều dài : 660mm
- Chiều rộng: 400mm
- Chiều cao đế nhà: 60mm
- Chiều cao tầng 1: 180mm
- Chiều cao tầng 2: 170mm
- Chiều cao mái: 150mm
- Diện tích mô hình nhà: $264000\text{mm}^2 = 0.64\text{m}^2$



Hình 3.17: Mô hình nhà thông minh

3.4.2. Thiết kế mô hình vườn rau thông minh

Thiết kế mô hình vườn rau thông minh trên phần mềm Solidworks 2017

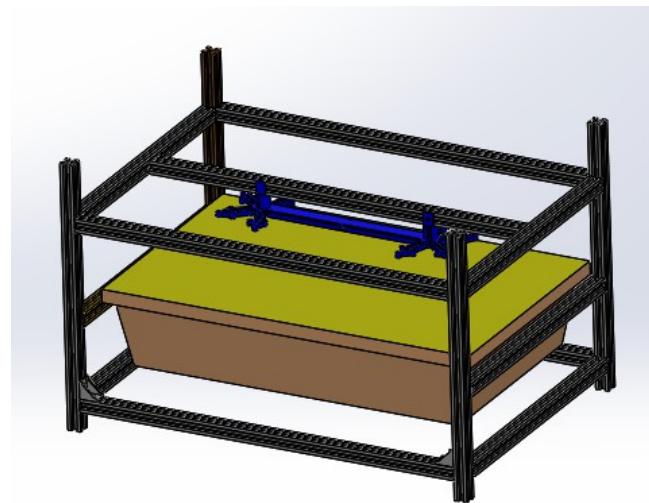
Mô hình vườn rau thông minh được thiết kế giống như 1 vườn rau thu nhỏ nhưng với quy mô nhỏ hơn. Khung vườn được làm bằng những thanh nhôm định

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

hình, rau được đựng trong chậu và trên khung vườn có gắn những giá đỡ để chứa thiết bị như đèn, quạt, béc tưới.

Kích thước khung mô hình vườn rau thông minh: 580x400x400 (mm)

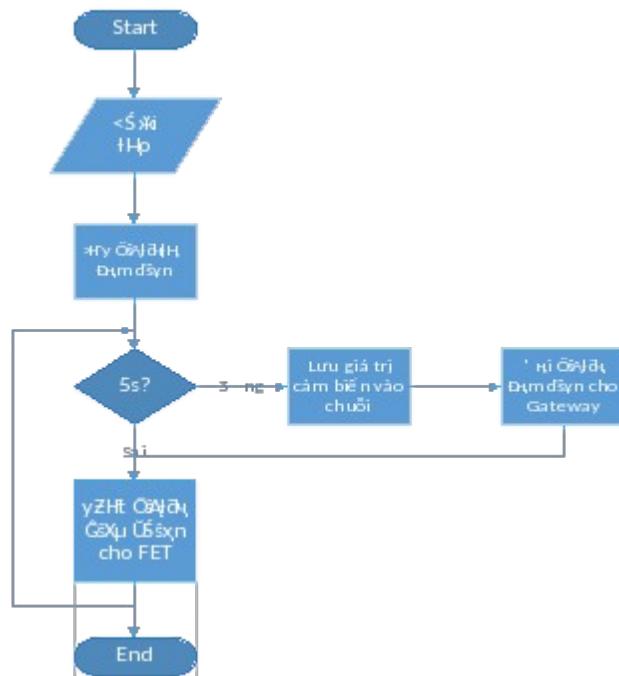
Kích thước chậu rau: 570x360x110 (mm)



Hình 3.18. Thiết kế mô hình vườn rau

3.5. NGUYÊN LÝ HOẠT ĐỘNG CỦA TOÀN BỘ HỆ THỐNG

3.5.1. Lưu đồ giải thuật đọc giá trị cảm biến và điều khiển



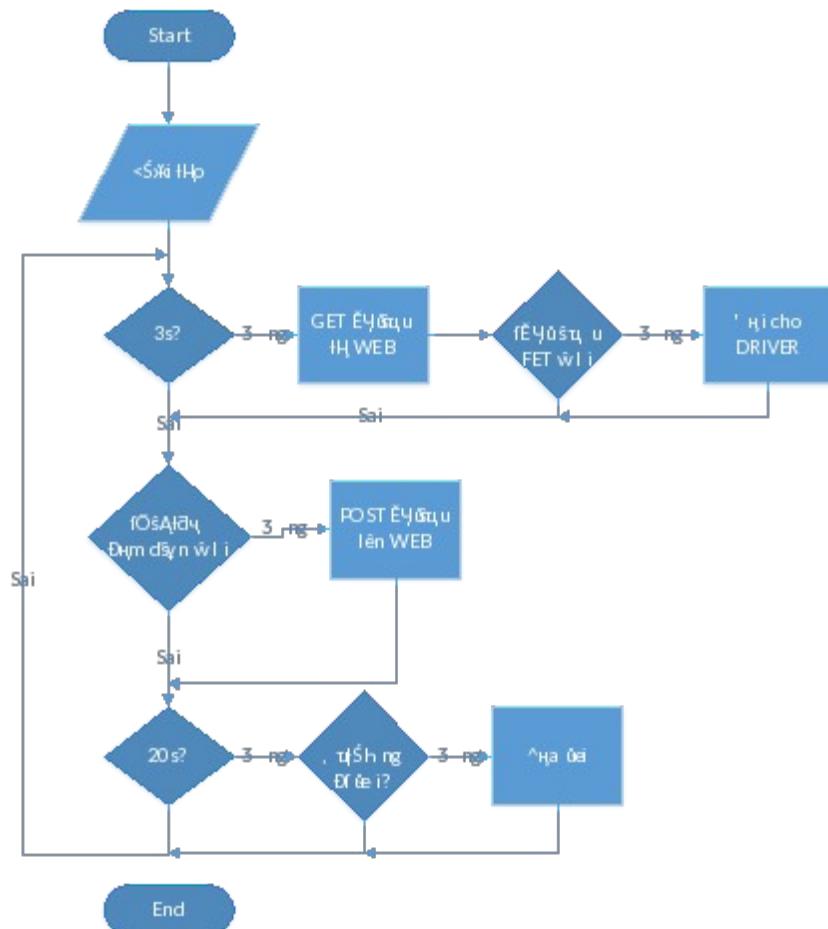
Hình 3.19: Lưu đồ thuật toán mạch DRIVER

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

Bắt đầu, khi cấp nguồn cho mạch Driver, hệ thống sẽ khởi tạo các giá trị ban đầu, sau đó gọi các hàm lấy giá trị cảm biến lưu vào các biến. Trong thời gian 5 giây tiếp theo, các giá trị sẽ được lưu vào chuỗi và gửi qua mạch Gateway qua chân TX3 của mạch Driver. Nếu trong 5 giây, không có giá trị của cảm biến thì hệ thống sẽ kiểm tra chuỗi được nhận được chân RX3, sau đó xử lý chuỗi, chuyển chuỗi thành những giá trị để điều khiển mosfet qua các chân PWM

Sau 5 giây, mạch Driver sẽ nhận được chuỗi điều khiển qua chân RX3, hệ thống sẽ xử lý chuỗi, chuyển chuỗi thành các giá trị để điều khiển mosfet qua các chân PWM.

3.5.2. Lưu đồ giải thuật của thuật toán giao tiếp với mạch GATEWAY



Hình 3.20: Lưu đồ giải thuật mạch GATEWAY

Bắt đầu, khi cấp nguồn cho mạch GATEWAY, hệ thống sẽ khởi tạo các giá trị ban đầu, khởi tạo các biến ban đầu . Sau đó gọi khởi tạo các hàm định nghĩa cho SIM808 MCU, khởi tạo TIMER, UART, ngắt UART.

Chương 3. TÍNH TOÁN VÀ THIẾT KẾ

Vì mạch DRIVER sẽ gửi chuỗi qua TX3 khi có các giá trị của cảm biến, nên bên mạch GATEWAY, hàm kiểm tra giá trị chuỗi nhận được qua chân RX3 sẽ được đưa vào hàm ngắt. Khi có tín hiệu, hệ thống sẽ ưu tiên lấy dữ liệu trước.

Trong thời gian 3 giây tiếp theo, hàm GET giá trị từ web sẽ thực hiện để lấy những giá trị của các công tắc. Nếu có giá trị mới, GATEWAY sẽ nhận được và gửi cho mạch DRIVER qua chân TX3. Nếu không có tín hiệu GET được từ web. Hệ thống sẽ ngay lập tức POST dữ liệu cảm biến lên web.

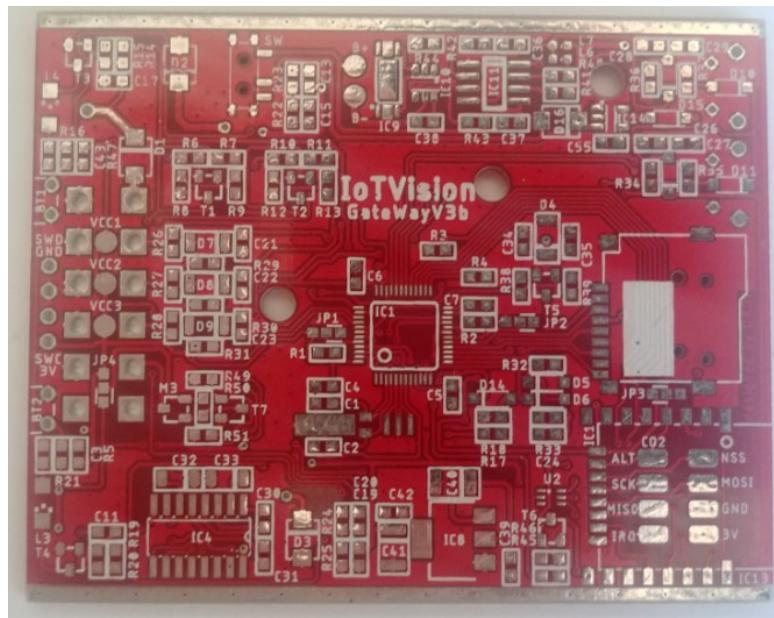
GATEWAY sẽ kiểm tra lại hệ thống sau mỗi chu kỳ 20 giây, thời gian được đếm bằng TIMER nên sẽ không ảnh hưởng tới các chương trình khác. Chương trình kiểm tra sẽ gồm : Kiểm tra nguồn, kiểm tra kết nối GPRS. Nếu có lỗi thì chương trình sẽ sửa lỗi. Nếu không có lỗi thì sẽ tiếp tục 20 giây tiếp theo.

CHƯƠNG 4 : THI CÔNG HỆ THỐNG

4.1. THI CÔNG PHẦN MẠCH ĐIỆN

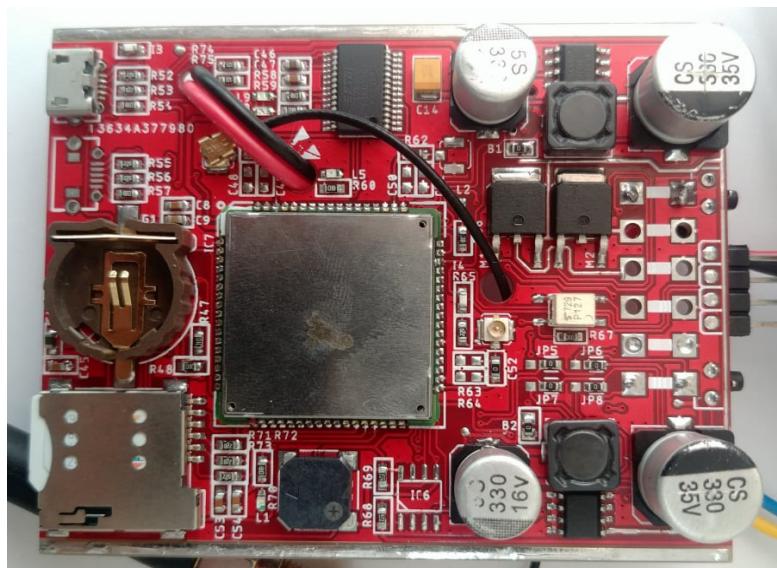
4.1.1. Thi công mạch GATEWAY

Sau khi thiết kế mạch nguyên lý, sơ đồ đi dây, sơ đồ mạch PCB, nhóm sẽ tiến hành thi công mạch.



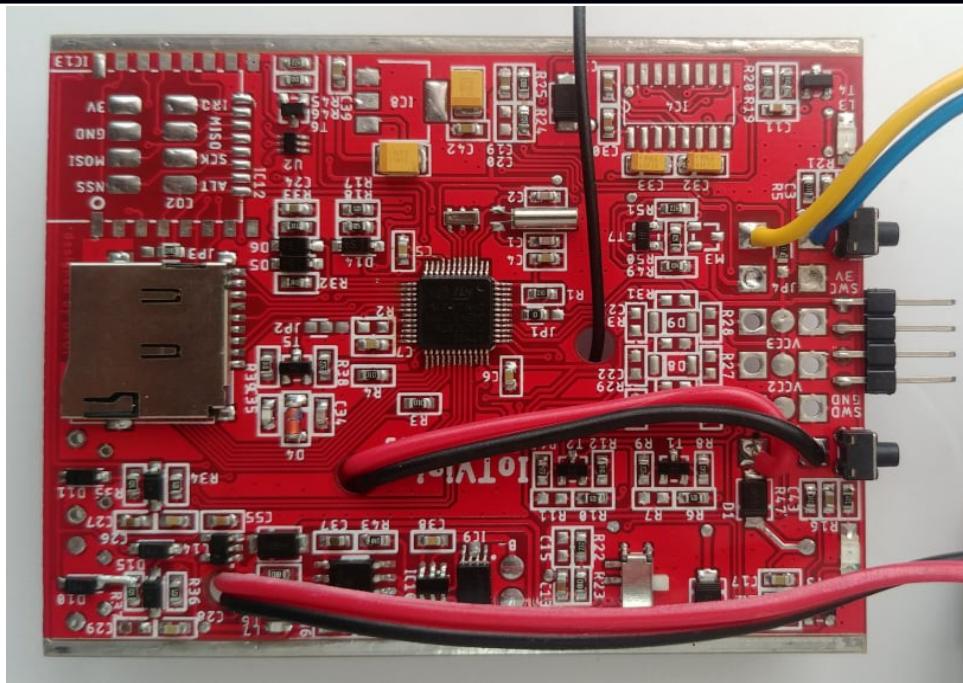
Hình 4.1: Mạch GATEWAY sau khi in PCB

Sau khi thi công mạch PCB, nhóm tiến hành hàn chân linh kiện lên mạch.



Hình 4.2: Mạch GATEWAY sau khi hoàn thành

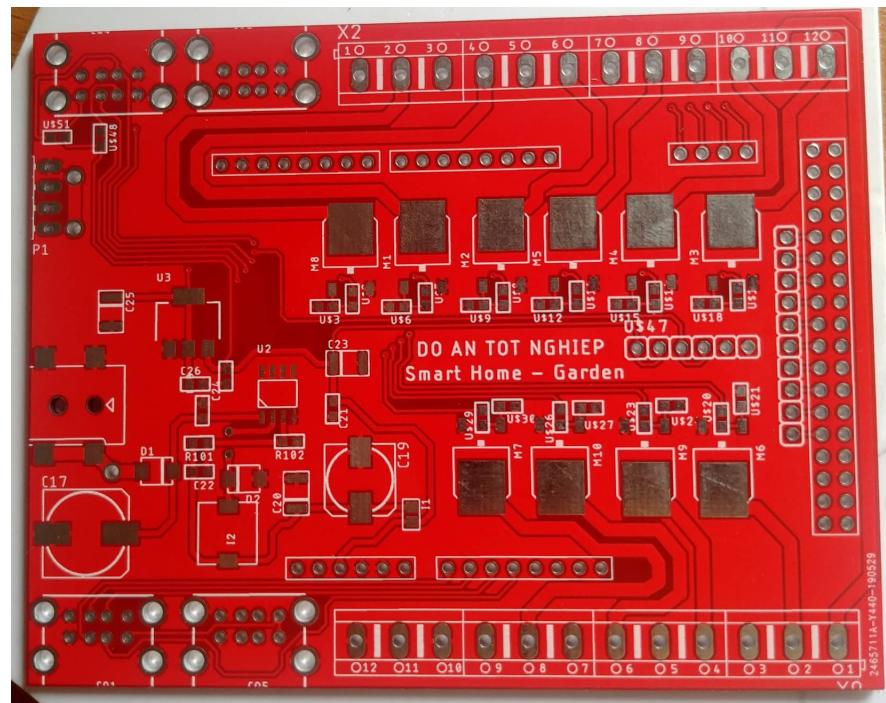
Chương 4. THI CÔNG HỆ THỐNG



Hình 4.3: Mặt sau của mạch GATEWAY sau khi hoàn thành

4.1.2. Thi công mạch DRIVER

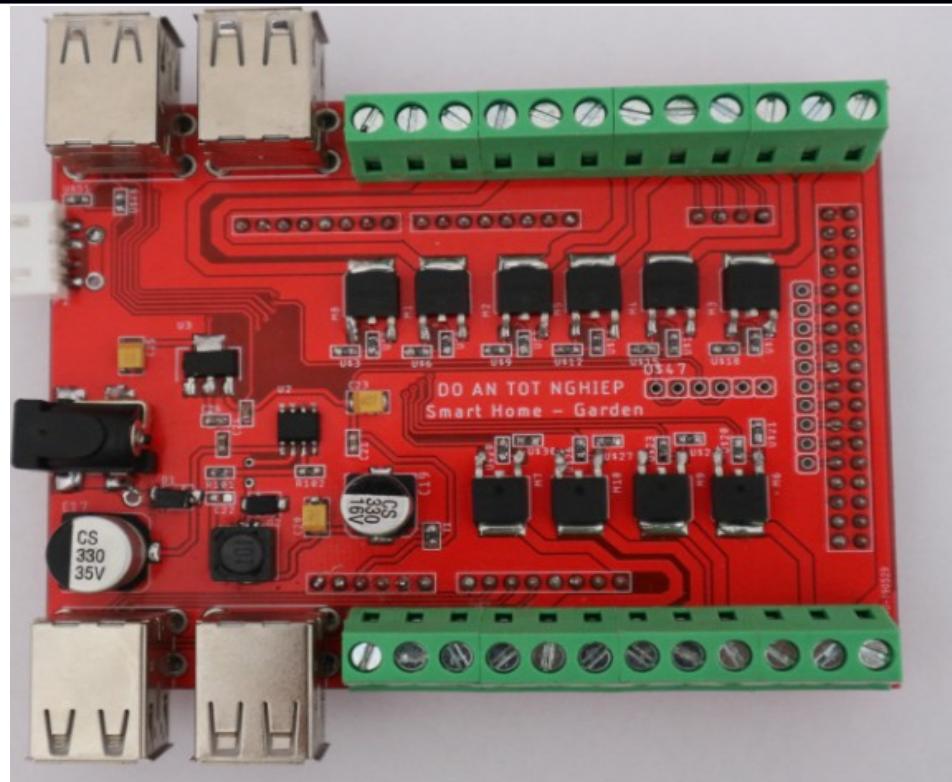
Sau khi thiết kế mạch nguyên lý, sơ đồ đi dây, sơ đồ mạch PCB, nhóm sẽ tiến hành thi công mạch.



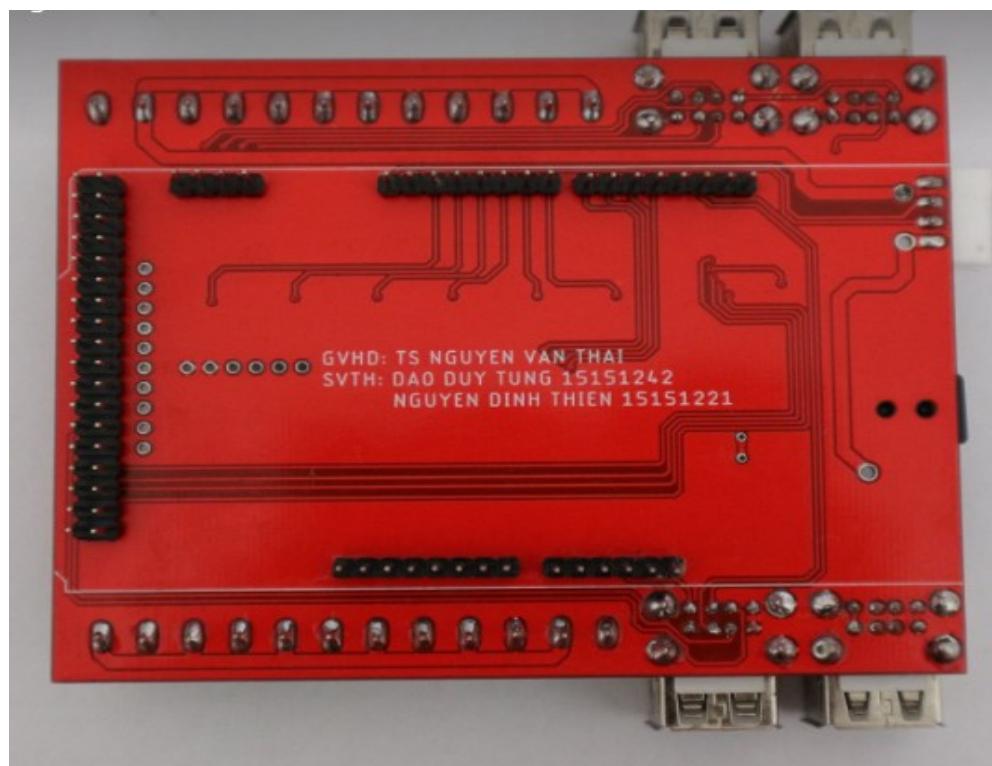
Hình 4.4: Mạch sau khi được in PCB

Sau khi mạch được thi công PCB, nhóm sẽ tiến hành hàn linh kiện lên mạch.

Chương 4. THI CÔNG HỆ THỐNG



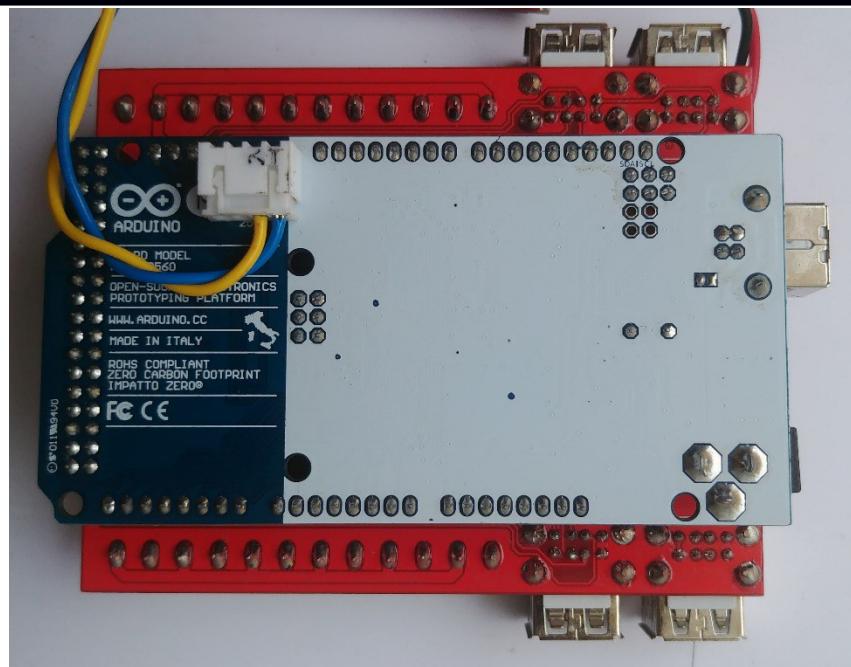
Hình 4.5: Mặt trước của mạch Driver



Hình 4.6: Mặt sau của mạch Driver

Mạch Driver sau khi thi công sẽ cắm lên Board Arduino Mega2560.

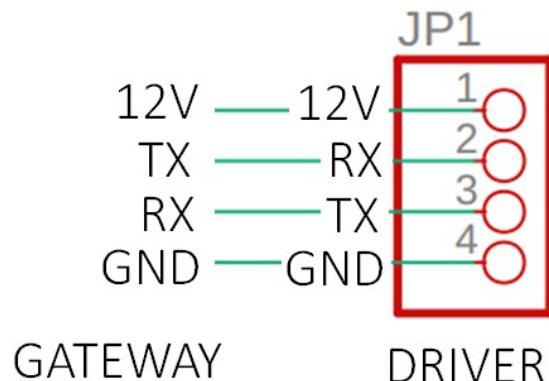
Chương 4. THI CÔNG HỆ THỐNG



Hình 4.7: Mạch Driver hoàn chỉnh

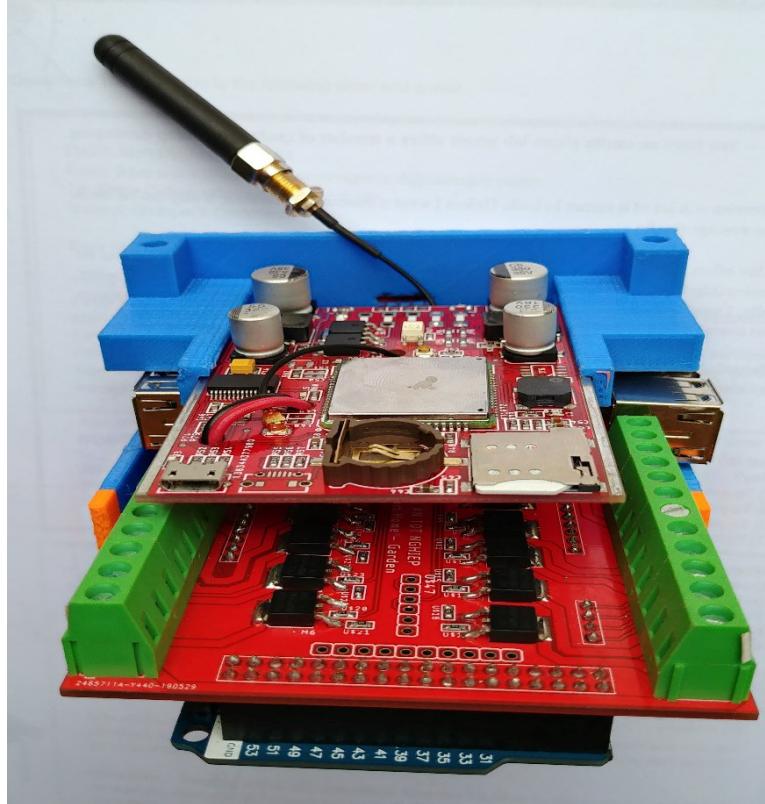
4.1.3. Kết nối Gateway và Driver với nhau.

Sau khi thi công xong 2 mạch GATEWAY và mạch DRIVER, nhóm tiến hành kết nối 2 mạch lại với nhau.



Hình 4.8: Sơ đồ kết nối chân 2 mạch

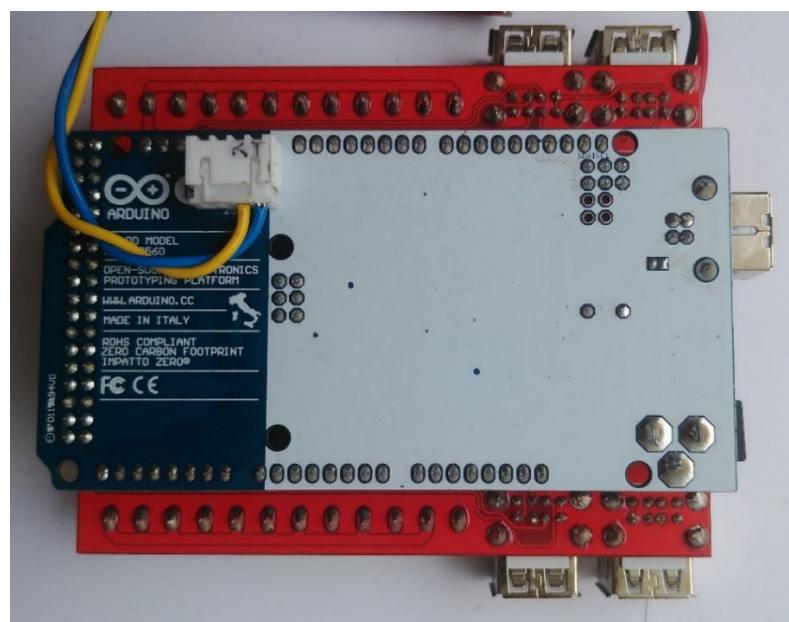
Chương 4. THI CÔNG HỆ THỐNG



Hình 4.9: Kết nối mạch Driver và mạch Gateway với nhau

4.1.4. Kết nối cảm biến với mạch

Các đầu dò cảm biến được ra chân thành các đầu USB để tiện cho việc vận hành.



Hình 4.10: Các cảm biến và dây của thiết bị được kết nối

Chương 4. THI CÔNG HỆ THỐNG

4.3. THI CÔNG PHẦN CƠ KHÍ

4.3.1. Thi công mô hình nhà thông minh

Dựa trên bản vẽ Solidworks, mô hình nhà thông minh được thi công đúng với kích thước như trên bản vẽ.

Một số vật liệu để thi công mô hình nhà:

Bảng 4.1. Một số vật liệu để thi công mô hình nhà thông minh

STT	Tên vật liệu
1	Nhôm đinh hình 20x20
2	Bìa Foam dày 3mm
3	Bìa Foam dày 10mm
4	Mica màu đen, trong suốt
5	Ốc vít, keo



Hình 4.11: Mô hình nhà thông minh sau khi được thi công

Sau khi thi công xong mô hình cơ khí, nhóm sẽ tiến hành gắn các thiết bị để điều khiển cho ngôi nhà.

Chương 4. THI CÔNG HỆ THỐNG

Thiết bị gồm có 1 quạt 12V và 3 bóng đèn 12V. Thiết bị được bố trí như sau: Phòng khách có 1 bóng đèn, 1 quạt, 1 bóng đèn tại phòng ngủ, và 1 bóng đèn ở ban công.



Hình 4.12: Đèn và quạt phòng khách được gắn lên mô hình

4.3.2. Thi công mô hình vườn rau thông minh

Dựa trên bản vẽ Solidworks, mô hình vườn thông minh được thi công đúng với kích thước như trên bản vẽ.

Một số vật liệu để thi công mô hình vườn rau thông minh:

Bảng 4.2. Một số vật liệu để thi công mô hình vườn rau thông minh

STT	Tên vật liệu
1	Nhôm định hình 20x20
2	Ke góc 90 độ
3	Thanh chữ L
4	Mica màu đen
5	Ốc vít, keo
6	Chậu nhựa

Chương 4. THI CÔNG HỆ THỐNG



Hình 4.13: Khung nhôm định hình của mô hình

Thiết bị điều khiển của mô hình vườn rau thông minh gồm có quạt, bơm, và đèn LED UV.

Đèn LED dùng để chiếu sáng gồm 20 đèn led, chia làm 4 chùm, mỗi chùm 5 led được mắc nối tiếp.

Điện áp hoạt động của 1 bóng LED là 2.7VDC, khi mắc 5 bóng nối tiếp thì điện áp sẽ xấp xỉ 12V. Dòng điện của 1 chùm LED khi đo bằng VOM là khoảng 0.1A, khi mắc 4 chùm song song thì dòng điện là 0.4A.



Chương 4. THI CÔNG HỆ THỐNG

Hình 4.14: Thi công đèn LED UV



Hình 4.15: Thi công quạt 12V



Hình 4.16: Thi công bơm 12V và đi dây béc tưới trong đế LED

4.4. THI CÔNG GIAO DIỆN WEB

4.4.1. Giới thiệu trang web iotstar.vn của 3D VisionLab

Trong đồ án của nhóm sử dụng nền tảng web của 3D Vision Lab, trực thuộc trường Đại học Sư phạm Kỹ thuật TP HCM đó là trang web iotstar.vn. Được viết

Chương 4. THI CÔNG HỆ THỐNG

bằng ngôn ngữ Javascript, ngôn ngữ web là ASP.NET MVC 5, có đầy đủ chức năng đăng nhập, đăng ký, đăng xuất. Cơ sở dữ liệu được xây dựng trên phần mềm Microsoft SQL Server.



Hình 4.17: Giao diện trang web Iotstar.vn

4.4.2. Thêm cảm biến và thiết bị trên web IOT

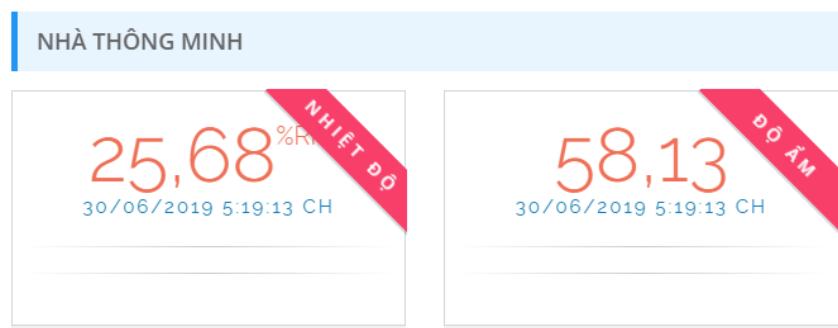
Sau khi đăng ký tài khoản, giao diện web sẽ có những nút bấm cho nhóm lựa chọn để thêm dự án, thêm nút bấm và cảm biến.



Hình 4.18: Dự án nhóm đã tạo trên web

Sau khi tạo dự án, nhóm tiến hành thêm cảm biến và công tắc điều khiển thiết bị trên dự án nhà thông minh, vườn rau thông minh.

Chương 4. THI CÔNG HỆ THỐNG



Hình 4.19: Thêm cảm biến nhiệt độ và độ ẩm



Hình 4.20: Thêm công tắc điều khiển thiết bị trong nhà



Hình 4.21: Thêm cảm biến trong vườn

4.5. KẾT QUẢ THỰC NGHIỆM

4.5.1. Thu thập, giám sát giá trị cảm biến

Khi truy cập vào trang web, ngay ở giao diện đầu tiên sẽ hiển thị những thông số giám sát nhóm đã thêm.

Tất cả các giá trị cảm biến đều được giám sát theo thời gian thực, có lưu lại tại cơ sở dữ liệu, thời gian cập nhật là khoảng 10 giây, các giá trị cảm biến sẽ thu thập 1 lần.

Chương 4. THI CÔNG HỆ THỐNG



Hình 4.22: Độ ẩm không khí và nhiệt độ tại vườn



Hình 4.23: Độ ẩm không khí, nhiệt độ, CO2, độ ẩm đất, ánh sáng tại vườn

4.5.2. Điều khiển thiết bị qua giao diện web

Khi truy cập vào dự án, ngay bên dưới phần giám sát cảm biến sẽ là phần điều khiển thiết bị.



Hình 4.24: Giao diện điều khiển thiết bị

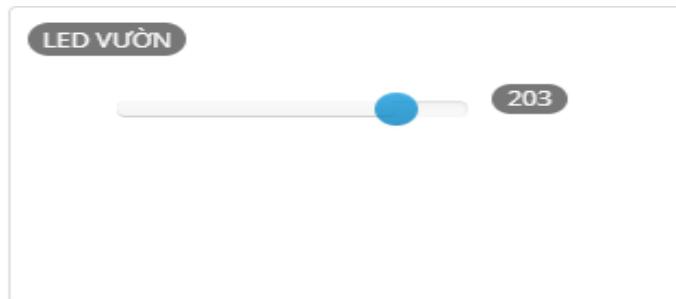
Khi thiết kế giao diện điều khiển, có 7 thiết bị là điều ON – OFF bao gồm: Đèn trang trí, đèn phòng khách, quạt, đèn phòng ngủ, đèn ban công, quạt vườn rau và béc tươi.

Chương 4. THI CÔNG HỆ THỐNG

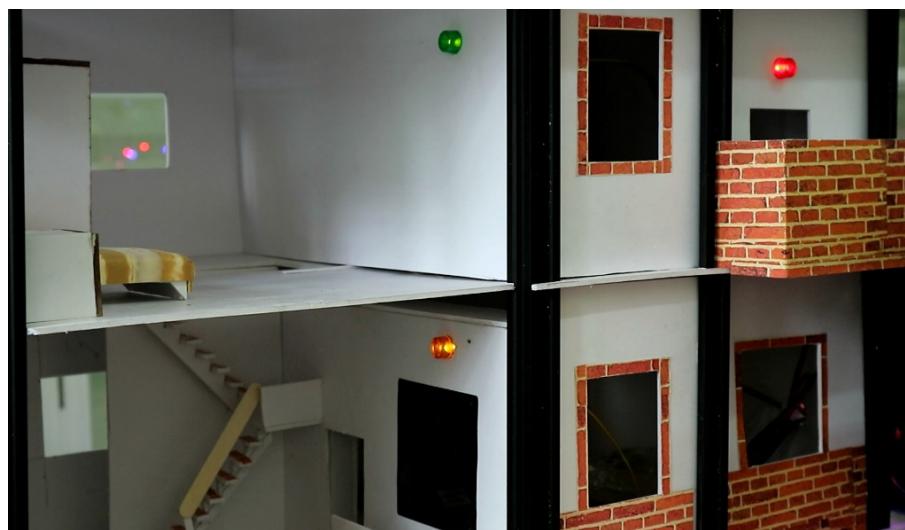
Phần đèn LED UV bên mô hình vườn rau, nhóm thiết kế 1 thanh trượt slider. Thanh trượt slider có giá trị từ 0-255, khi điều khiển thì người dùng có thể điều khiển được độ sáng.



Hình 4.25: Bật lần lượt từng thiết bị



Hình 4.26: Thay đổi độ sáng của đèn theo giá trị từ 0 – 255



Hình 4.27: Điều khiển các thiết bị trong nhà

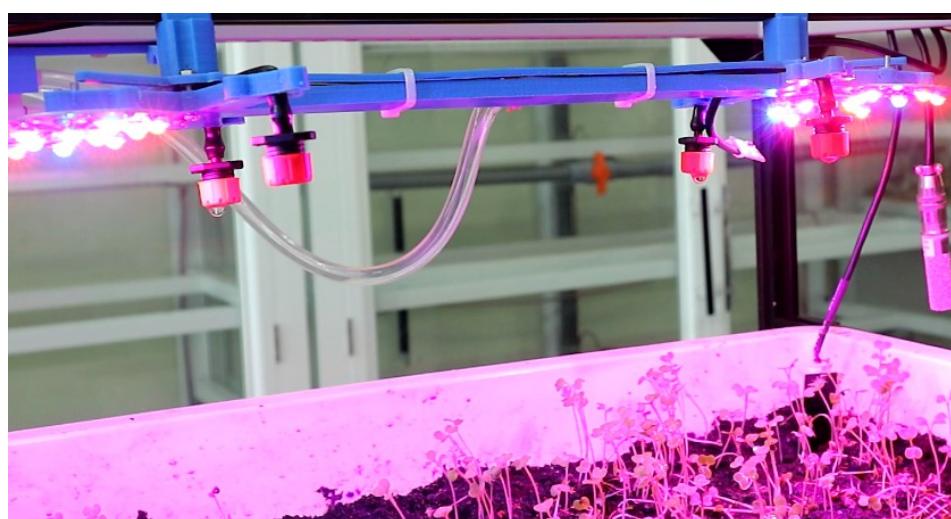
Chương 4. THI CÔNG HỆ THỐNG



Hình 4.28: Điều khiển bơm tưới và quạt



Hình 4.29: Điều khiển LED sáng 20%



Hình 4.30: Điều khiển LED sáng 70%

Chương 4. THI CÔNG HỆ THỐNG

4.5.3. Kết quả thực nghiệm

Bảng 4.3: Bảng số liệu thực nghiệm

Công việc	Webserver	Thời gian đáp ứng	Đánh giá
Điều khiển thiết bị (20 lần)	20 lần thành công	2-5s	Thời gian đáp ứng nhanh
Giám sát các cảm biến (24h)	Ôn định	5s	Sai số không đáng kể.

Chương 5. KẾT QUẢ

Nhóm đã đạt được những thành công nhất định thông qua việc tìm hiểu, nghiên cứu các đối tượng liên quan đến đồ án; thiết kế và thi công được mạch điều khiển gọn đẹp và xây dựng được một mô hình thực tế. Cụ thể như sau:

- ✓ Sử dụng thành thạo phần mềm Solidwork, thiết kế được mô hình cơ khí 1 cách chắc chắn, đẹp mắt.
- ✓ Nghiên cứu, thiết kế mạch Driver dựa trên board mạch Arduino, sử dụng phần mềm vẽ mạch điện tử Eagle 1 cách thành thạo.
- ✓ Nghiên cứu thiết kế mạch nguyên lý và sơ đồ đi dây của board mạch Gateway.
- ✓ Lập trình được 3 dòng vi điều khiển, giao tiếp 3 vi điều khiển khác nhau là ATMega2560 và STM32 và SIM808
- ✓ Tìm hiểu về lập trình web, xây dựng giao diện website.
- ✓ Tìm hiểu nghiên cứu, cách vận hành web iotstar.vn.
- ✓ Củng cố lại chuẩn giao tiếp UART, chuẩn giao tiếp 1 dây, giao tiếp 2 dây đã được học. Tiếp cận được phương thức GET – POST trong giao thức HTTP.
- ✓ Tính toán, chọn linh kiện, thi công để hoàn thiện được phần mạch điện.
- ✓ Thi công thành công mô hình cơ khí nhà và vườn.
- ✓ Giám sát được các giá trị nhiệt độ, độ ẩm, ánh sáng, CO₂ qua giao diện web.
- ✓ Điều khiển bật, tắt các thiết bị trong mô hình, tất cả thiết bị đều có thể điều khiển Analog, thay đổi giá trị từ 0 – 255.
- ✓ Lập trình cho hệ thống hoạt động ổn định qua nhiều lần kiểm tra, xử lý triệt để các tình huống mạch hoạt động sai so với yêu cầu. Hệ thống điều khiển từ xa trên giao diện trang web dễ dàng sử dụng. Cảm biến hoạt động chính xác, đáp ứng trong các trường hợp sự cố nhanh, hiệu quả.

Chương 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. KẾT LUẬN

Với những mục tiêu đã đề ra, đề tài đã giải quyết và hoàn thành được những yêu cầu ban đầu là thiết kế và thi công được một mạch trung tâm xử lý (mạch Gateway) và 1 trạm xử lý (mạch Driver) có thể điều khiển các thiết bị điện trong mô hình nhà và vườn, đưa được dữ liệu cảm biến lên webserver , giám sát giá trị cảm biến theo thời gian thực, điều khiển được thiết bị qua giao diện web. Ngoài ra nhóm cũng đã giải quyết được bài toán về mô hình cơ khí, mô hình chắc chắn, chuyên nghiệp và an toàn.

Trong quá trình làm đồ án, sinh viên đã rút ra được nhiều kinh nghiệm để tạo ra một sản phẩm hoàn thiện như: đầu tư thời gian, linh kiện trên thị trường, thi công mạch PCB, hiểu biết về các linh kiện, thiết kế board mạch, cơ khí, tính toán, đo đạc thi công mô hình,...

6.2. HƯỚNG PHÁT TRIỂN

Một số hướng phát triển để hoàn thiện đề tài như là:

- Kết nối internet, điều khiển kết hợp giữa Bluetooth, wifi và viết một app Android để theo dõi thuận tiện hơn cho người giám sát và điều khiển.
- Giao tiếp được với camera để có thể giám sát trực tiếp.
- Thêm mạch công suất lớn để có thể điều khiển nhiều tải công suất lớn.
- Thay thế Module SIM808 thành SIM7500 để tốc độ điều khiển và thực hiện lệnh nhanh hơn. SIM7500 là dòng chip có tích hợp mạng 4G. Có thể thêm ESP8266 và SocketIO để thời gian đáp ứng nhanh hơn.
- Thay thế phương thức truyền tín hiệu, từ UART thành LORA để tín hiệu có thể truyền xa hơn, kết nối được nhiều trạm xử lý hơn.

TÀI LIỆU THAM KHẢO

TÀI LIỆU THAM KHẢO

SÁCH THAM KHẢO:

[1] Trần Thu Hà, “Điện tử cơ bản”, NXB ĐH Quốc Gia Tp.HCM, 2013.

TRANG WEB THAM KHẢO:

[1] Nguyễn Thị Trọng Nghĩa, “Phương thức GET và POST”, 2017.
<https://viblo.asia/p/phuong-thuc-get-va-post-aWj53VBYl6m>

[2] MechaSolution , “UART và giao tiếp Serial ”, 2018
<https://mechasolution.vn/Blog/uart-va-giao-tiep-serial>

[3] TAPIT ARM , “STM32 CubeMX và Keil C lập trình STM32”, 2019
<https://tapit.vn/huong-dan-su-dung-stm32cubemx-va-keil-c-lap-trinh-stm32/>

[4] Murditsky Yaroslav , “GPRS Module SIM808 Intergration”, 2019
<https://community.st.com/s/question/0D50X00009fCxynSAC/gprs-module-sim808-integration>

PHỤ LỤC

PHỤ LỤC

❖ Phần mềm lập trình cho Arduino

✓ Giới thiệu

Trong đề tài này nhóm sử dụng phần mềm ARDUINO IDE vì nó tiện lợi và hỗ trợ tốt hơn cho người sử dụng, nó là một trình soạn thảo văn bản, giúp bạn viết code để nạp vào bo mạch Arduino, dùng được cho cả board NodeMCU và Arduino mà nhóm đang sử dụng.

✓ Cài đặt phần mềm

Bước 1: Vào đường link <https://www.arduino.cc/en/Main/Software/>

Bước 2: Bấm chuột vào Windows free installation ZIP package .

Bước 3: Download file cài đặt và giải nén file cài đặt.

Bước 4: Chạy file

❖ Phần mềm lập trình cho vi điều khiển STM32

✓ Giới thiệu về STM32CubeMX và Keil C

STM32CubeMX là một công cụ hỗ trợ cấu hình và sinh code cho MCU STM32. Tất cả các công việc cấu hình, nâng cấp đều được thực hiện qua giao diện đồ họa. Việc này giúp cho việc lập trình trên STM32 dễ dàng hơn, rút ngắn được thời gian nghiên cứu và phát triển.

Keil C là chương trình hỗ trợ khá đầy đủ trong việc lập trình cho vi điều khiển họ 8051 ngoài việc biên dịch bằng ngôn ngữ C bạn cũng có thể biên dịch dưới dạng ASM. Đây cũng là một cách để tìm hiểu thêm về ngôn ngữ ASM rất tốt cho bạn

✓ Cài đặt phần mềm

Bước 1: Download chương trình : <https://www.st.com/en/development-tools/stm32cubemx.html>

Bước 2: Download chương trình:

http://www.4file.net/z82tvi43ufm7/Keil_C4_Full.rar.html

Bước 2: Chạy chương trình

Bước 3: Nhấn Next, chờ khi chương trình cài xong và nhấn Finish.

PHỤ LỤC

❖ Chương trình điều khiển cho board Driver:

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_TSL2561_U.h>
#include<String.h>

#include "SHT1x.h"
#define dataPin1 49 // gpio sht10
#define sckPin1 48
#define dataPin2 51 // gpio sht10
#define sckPin2 50
SHT1x th_sensor1(dataPin1, sckPin1); //SHT10 1
SHT1x th_sensor2(dataPin2, sckPin2); //SHT10 2
Adafruit_TSL2561_Unified tsl = Adafruit_TSL2561_Unified(TSL2561_ADDR_FLOAT, 12345);
int sensor_pin1 = A1;//analog độ ẩm đất

int sensor_pin2 = A2;//analog độ ẩm đất
int sensor_pin3 = A0;//analog CO2
int output_value1 ;
int output_value2 ;
int ppm ;
int ppmnew;

char str1[6];
char str2[6];
char str3[6];
char str4[6];
char str5[3];
char str6[3];
char str7[6];
```

PHỤ LỤC

```
char str8[6];
char buffer[64];
char str[35];
char s1[5];
char s2[5];
char s3[5];
char s4[5];
char s5[5];
char s6[5];
char s7[5];
char s8;
char s9;
char s10;
char s[5];
int out_1 = 7; //LED 1
int out_2 = 6; //LED 2
int out_3 = 5; //BƠM
int out_4 = 4; //QUẠT vườn
int out_5 = 3;
int out_6 = 2;
int out_7 = 8; //ĐÈN PHÒNG KHÁCH
int out_8 = 9; //QUẠT PHÒNG KHÁCH
int out_9 = 10; //ĐÈN PHÒNG NGỦ
int out_10 = 11;// ĐÈN BAN CÔNG
int out_11 = 12; //PWM ra chan USB trên (CỦA)
int out_12 = 13; //PWM ra chan USB dưới
int num1 ;
int num2 ;
int num3 ;
int num4 ;
int num5;
int num6;
```

PHỤ LỤC

```
int num7 ;  
int num8;  
unsigned long preTimer;  
  
void displaySensorDetails(void)//hien thi thong tin sensor  
{  
    sensor_t sensor;  
    tsl.getSensor(&sensor);  
  
    delay(500);  
}  
void configureSensor(void)  
{  
    tsl.enableAutoRange(true);  
    tsl.setIntegrationTime(TSL2561_INTEGRATIONTIME_13MS);  
    delay(1000);  
}  
  
void setup(void)  
{  
    Serial.begin(9600);  
    Serial3.begin(115200);  
    pinMode(out_1, OUTPUT);  
    pinMode(out_2, OUTPUT);  
    pinMode(out_3, OUTPUT);  
    pinMode(out_4, OUTPUT);  
    pinMode(out_5, OUTPUT);  
    pinMode(out_6, OUTPUT);  
    pinMode(out_7, OUTPUT);  
    pinMode(out_8, OUTPUT);  
  
    if(!tsl.begin())
```

PHỤ LỤC

```
displaySensorDetails();
configureSensor();
Serial.println("OK");

delay(2000);
}

void loop()
{
    read_sht101();
    read_sht102();
    read_doamdat1();
    read_doamdat2();
    tsl2561();
    readco2();
    //checkstr();
    TimerIsr();
    if(millis() - preTimer >= 4000)
    {
        Serial3.println(str);
        Serial.println(str);
        preTimer = millis();
    }
    sprintf(str,"%s%s%s%s%s%s%s",str1,str2,str3,str4,str5,str6,str7,str8);
    //sprintf(strx,"%s",buffer);
    TimerIsr();
    //Serial.println(str);
    //delay(10);
}
void TimerIsr(){
if (Serial3.available()>0){
    Serial3.readBytes(buffer, Serial3.available());
```

PHỤ LỤC

```
Serial.println("buffer");
Serial.println(buffer);
delay(200);
s1[0]=buffer[0];
s2[0]=buffer[2];
s3[0]=buffer[4];
s4[0]=buffer[6];
s5[0]=buffer[8];
s6[0]=buffer[10];
s7[0]=buffer[12];
s8=buffer[14];
s9=buffer[15];
s10=buffer[16];
delay(100);
num1 = atoi(s1); // CỦA
num2 = atoi(s2); // đèn phòng khách
num3 = atoi(s3); //Quạt phòng khách
num4 = atoi(s4); //Đèn phòng ngủ
num5 = atoi(s5); //Đèn ban công
num6 = atoi(s6); //Quạt vòi
num7 = atoi(s7); //Béc tưới
s[0] = s8;
s[1] = s9;
s[2] = s10;
num8 = atoi(s); //LED vòi
analogWrite(out_1,num8); //LED 1
analogWrite(out_2,num8); //LED 2

if (num7 == 1) analogWrite(out_3,240);//tưới bơm
else
analogWrite(out_3,0);
if (num6 == 1)
```

PHỤ LỤC

```
{  
    analogWrite(out_4,240); // QUẠT VƯỜN  
    analogWrite(out_5,240); // QUẠT VƯỜN  
}  
else  
{  
    analogWrite(out_4,0);  
    analogWrite(out_5,0);  
}  
if (num5 == 1) analogWrite(out_10,240); // đèn ban công  
else  
analogWrite(out_10,0);  
  
if (num4 == 1) analogWrite(out_9,240); // đèn phòng ngủ out9  
else  
analogWrite(out_9,0);  
  
if (num3 == 1) analogWrite(out_8,240); //quạt phòng khách  
else  
analogWrite(out_8,0);  
  
if (num2 == 1) analogWrite(out_7,240); //đèn phòng khách  
else  
analogWrite(out_7,0);  
  
if (num1 == 1) analogWrite(out_11,250); //đèn phòng khách  
else  
analogWrite(out_11,0);  
}  
}  
  
void read_sht101()
```

PHỤ LỤC

```
{  
    float temp_c1;  
    float humid1;  
  
    // Read values from the sensor  
    humid1 = th_sensor1.readHumidity();  
  
    temp_c1 = th_sensor1.retrieveTemperatureC();  
    temp_c1 = temp_c1-3;  
    dtostrf(temp_c1, 5, 2, str1);  
    //Serial.println(str1);  
  
    dtostrf(humid1, 5, 2, str2);  
    // Serial.println(str2);  
    delay(50);  
}  
  
void read_sht102()  
{  
    float temp_c2;  
    float humid2;  
    humid2 = th_sensor2.readHumidity();  
    temp_c2 = th_sensor2.retrieveTemperatureC();  
    temp_c2 = temp_c2-3;  
  
    dtostrf(temp_c2, 5, 2, str3);  
    //Serial.println(str3);  
    dtostrf(humid2, 5, 2, str4);  
    //Serial.println(str4);  
    delay(50);  
}  
void read_doadat1()
```

PHỤ LỤC

```
{  
    output_value1= analogRead(sensor_pin1);  
    output_value1 = map(output_value1,1023,0,0,100);  
    if(output_value1>99 || output_value1<10)  
    {  
        output_value1=10;  
    }  
    sprintf(str5, "%d", output_value1);  
    //Serial.println(str5);  
    delay(100);  
    delay(50);  
}  
  
void read_dreamdat2()  
{  
    output_value2= analogRead(sensor_pin2);  
    output_value2 = map(output_value2,1023,0,0,100);  
    if(output_value2>99 || output_value2<10)  
    {  
        output_value2=99;  
    }  
    sprintf(str6, "%d", output_value2);  
    // Serial.println(str6);  
    delay(50);  
}  
  
void tsl2561()  
{  
    sensors_event_t event;  
    tsl.getEvent(&event);  
    if (event.light)  
    {  
        dtostrf(event.light, 5, 0, str7);  
    }  
}
```

PHỤ LỤC

```
else
{
    Serial.println("Sensor overload");
}

delay(50);
}

void readco2()
{
    ppmnew=ppm;
    ppm= map(analogRead(sensor_pin3),0,2000,0,5000);
    ppm+= (ppm-ppmnew)*0.005;
    delay(100);

    sprintf(str8, "%d", ppm);
    // Serial.println(str8);
    // Serial.println("het chu ki");
    delay(50);
}
```

❖ Chương trình điều khiển cho board GATEWAY:

```
/* USER CODE BEGIN Header */

/**
 ****
 ****
 * @file      : main.c
 * @brief     : Main program body
```

PHỤ LỤC

```
/* Includes -----*/
#include "main.h"
#include "i2c.h"
#include "rtc.h"
#include "spi.h"
#include "tim.h"
#include "usart.h"
#include "gpio.h"

/* Private includes -----*/
/* USER CODE BEGIN Includes */
#include "string.h"
#include "SIM808.h"
#include "eeprom24xx.h"
#include "timerControl.h"
#include "debugControl.h"
/* USER CODE END Includes */

/* Private typedef -----*/
/* USER CODE BEGIN PTD */
#define HTTP_PARA3_SS_HOME
"AT+HTTPPARA=\"URL\",iotstar.vn/api/CamBienMaster?ma=M16\r\n"
#define HTTP_PARA3_SW_HOME
"AT+HTTPPARA=\"URL\",iotstar.vn/api/CongTacMaster?ma=M16\r\n"
#define HTTP_PARA3_SS_GARD
"AT+HTTPPARA=\"URL\",iotstar.vn/api/CamBienMaster?ma=M17\r\n"
#define HTTP_PARA3_SW_GARD
"AT+HTTPPARA=\"URL\",iotstar.vn/api/CongTacMaster?ma=M17\r\n"

/* USER CODE BEGIN PV */
typedef struct
```

PHỤ LỤC

```
{  
    uint8_t Run_Status;  
    uint8_t Fix_Status;  
    uint16_t UTC_Year;  
    uint8_t UTC_Month;  
    uint8_t UTC_Day;  
    uint8_t UTC_Hour;  
    uint8_t UTC_Minute;  
    float   UTC_Second;  
    float   Latitude;  
    float   Longitude;  
}GPS_Info_Type;  
  
typedef struct  
{  
    char User_Name[30];  
    char User_Phone[13];  
}User_Info_Type;  
  
RTC_TimeTypeDef RTCtime;  
RTC_DateTypeDef RTCdate;  
char rtc_fetch = 0x31;  
char rtc_catch;  
  
GPS_Info_Type GPS;  
User_Info_Type User;  
  
char status[50]      = "";  
char eepromData[256] = "";  
char buffer2[500]     = "";  
char buffer3[35]      = "";  
char data_uart3;
```

PHỤ LỤC

```
char driver[12];
char json_post_data[200];
char json_get_data[200];
char str[50];
char i = 0;
char *ptr;
char feed = 0;

char SHTtemp1[10] = "",SHTthumi1[10] = "";
char SHTtemp2[10] = "",SHTthumi2[10] = "";
char CAPhum1[10] = "",CAPhum2[10] = "";
char lux[10] = "",ppm[10] = "";

char home_periph[20] = "";
char gard_periph[20] = "";

uint16_t postYear;
uint8_t postMonth;
uint8_t postDay;
uint8_t postHour;
uint8_t postMinute;
uint8_t postSecond;

uint8_t debug_string = 1;
uint8_t getVDD = 0;
uint8_t SIM_active = 0;
uint8_t GPRS_active = 0;
uint8_t HTTP_active = 0;
uint8_t enable_GPS = 0;
uint8_t debug = 0;
uint8_t postData = 0;
uint8_t getData = 0;
```

PHỤ LỤC

```
uint8_t getSMS      = 0;
uint8_t getGPS          = 0;
uint8_t SMS_getRST   = 0;
uint8_t SMS_getRTC   = 0;

uint16_t timer1_ms        = 0;
uint16_t debugging_PreTimer = 0;
uint16_t debuggingTime      = 20000;
uint16_t GetGPS_PreTimer   = 0;
uint16_t PostData_PreTimer = 0;
uint16_t GetData_PreTimer  = 0;

char User_phoneNum[]     = "+84383678987";
char GetSMS_array[500]      = "";
char GPS_data[120]       = "";
char GPS_latitude[12]    = "10.7823952";
char GPS_Lastlatitude[12] = "10.7823952";
char GPS_longitude[12]   = "106.7014437";
char GPS_Lastlongitude[12]= "106.7014437";
/* USER CODE END PV */

/* Private function prototypes -----
void SystemClock_Config(void);
static void MX_NVIC_Init(void);
/* USER CODE BEGIN PFP */

/* USER CODE END PFP */

/* Private user code -----
/* USER CODE BEGIN 0 */
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
```

PHỤ LỤC

```
if(GPIO_Pin == DVCC_Pin)
    VDD_detect();
}

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if(htim->Instance == TIM1)
    {
        timer1_ms++;
        if(timer1_ms >= 60000)
            timer1_ms = 0;
    }
}

void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
    if(huart->Instance == USART2)
    {
        HAL_UART_Receive_IT(&huart2,
(uint8_t*)buffer2,sizeof(buffer2));
    }
    if(huart->Instance == USART3)
    {
        buffer3[i] = data_uart3;
        i++;
        if(strstr(buffer3,"\r\n")!=NULL)
        {
            ptr = strstr(buffer3,"\r\n");
            *ptr = 0; ptr++;
            *ptr = 0; i = 0;
            strncpy(SHTtemp1,&buffer3[0],5);
            strncpy(SHThumi1,&buffer3[5],5);
        }
    }
}
```

PHỤ LỤC

```
        strncpy(SHTtemp2,&buffer3[10],5);
        strncpy(SHTThumi2,&buffer3[15],5);
        strncpy(CAPhumi1,&buffer3[20],2);
        strncpy(CAPhumi2,&buffer3[22],2);
        strncpy(lux,&buffer3[24],5);
        strncpy(ppm,&buffer3[29],4);
    }

    HAL_UART_Receive_IT(&huart3,(uint8_t*)&data_uart3,1);
}

/*
 * USER CODE END 0
 */

int main(void)
{
/* USER CODE BEGIN 1

    /* Configure the system clock */
    SystemClock_Config();

    /* Initialize all configured peripherals */
    MX_GPIO_Init();
    MX_SPI1_Init();
    MX_SPI2_Init();
    MX_USART2_UART_Init();
    MX_USART3_UART_Init();
    MX_RTC_Init();
    MX_TIM1_Init();
    MX_I2C1_Init();

    /* Initialize interrupts */
}
```

PHỤ LỤC

```
MX_NVIC_Init();

/* USER CODE BEGIN 2 */

while(HAL_UART_Receive_IT(&huart2,
(uint8_t*)buffer2,sizeof(buffer2))!=HAL_OK){}; //--SIM808 communication
while(HAL_UART_Receive_IT(&huart3,(uint8_t*)&data_uart3,1) !=HAL_OK){}; //--Driver communication

HAL_Delay(500);

VDD_detect();
buzz(1);

SIM808_init();
updateLastGPS(); /*Set init Location*/
GetGPS_control();
SMS_Init();

HAL_TIM_Base_Start_IT(&htim1);

/* USER CODE END 2 */

/* USER CODE BEGIN WHILE */

while (1)

{
    if(SMS_getRST)
    {
        SMS_getRST = 0;
        SMS_Init();
    }
    //GetGPS_control();
    GetSMS_control();
    PostData_control();
    GetData_control();
}
```

PHỤ LỤC

```
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
    updateTimer();
    DebuggingSystem();
}

/* USER CODE END 3 */
}

/***
 * @brief System Clock Configuration
 * @retval None
 */
void SystemClock_Config(void)
{
    RCC_OscInitTypeDef RCC_OscInitStruct = {0};
    RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
    RCC_PeriphCLKInitTypeDef PeriphClkInit = {0};

    /** Initializes the CPU, AHB and APB busses clocks
    */
    RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE|RCC_OSCILLATORTYPE_LSE;
    RCC_OscInitStruct.HSEState = RCC_HSE_ON;
    RCC_OscInitStruct.HSEPredivValue = RCC_HSE_PREDIV_DIV1;
    RCC_OscInitStruct.LSEState = RCC_LSE_ON;
    RCC_OscInitStruct.HSIState = RCC_HSI_ON;
    RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
    RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
    RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL9;
    if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
```

PHỤ LỤC

```
{  
    Error_Handler();  
}  
/** Initializes the CPU, AHB and APB busses clocks  
*/  
RCC_ClkInitStruct.ClockType      = RCC_CLOCKTYPE_HCLK|  
RCC_CLOCKTYPE_SYSCLK  
                                |RCC_CLOCKTYPE_PCLK1|  
RCC_CLOCKTYPE_PCLK2;  
RCC_ClkInitStruct.SYSCLKSource      =  
RCC_SYSCLKSOURCE_PLLCLK;  
RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;  
RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;  
RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;  
  
if      (HAL_RCC_ClockConfig(&RCC_ClkInitStruct,  
FLASH_LATENCY_2) != HAL_OK)  
{  
    Error_Handler();  
}  
PeriphClkInit.PeriphClockSelection = RCC_PERIPHCLK_RTC;  
PeriphClkInit.RTCClockSelection = RCC_RTCCLKSOURCE_LSE;  
if (HAL_RCCEx_PeriphCLKConfig(&PeriphClkInit) != HAL_OK)  
{  
    Error_Handler();  
}  
}  
  
/**  
 * @brief NVIC Configuration.  
 * @retval None  
 */
```

PHỤ LỤC

```
static void MX_NVIC_Init(void)
{
    /* USART2_IRQHandler interrupt configuration */
    HAL_NVIC_SetPriority(USART2_IRQHandler, 0, 0);
    HAL_NVIC_EnableIRQ(USART2_IRQHandler);

    /* EXTI9_5_IRQHandler interrupt configuration */
    HAL_NVIC_SetPriority(EXTI9_5_IRQHandler, 0, 2);
    HAL_NVIC_EnableIRQ(EXTI9_5_IRQHandler);

    /* USART3_IRQHandler interrupt configuration */
    HAL_NVIC_SetPriority(USART3_IRQHandler, 0, 1);
    HAL_NVIC_EnableIRQ(USART3_IRQHandler);

    /* TIM1_UP_IRQHandler interrupt configuration */
    HAL_NVIC_SetPriority(TIM1_UP_IRQHandler, 0, 3);
    HAL_NVIC_EnableIRQ(TIM1_UP_IRQHandler);

}

/* USER CODE BEGIN 4 */
void delay(unsigned long t){while(t--);}

void buzz(char _times)
{
    if(1)
    {
        for(int j = 0; j < _times; j++)
        {
            for(int i = 0; i < 350; i++)
            {

HAL_GPIO_WritePin(BUZZER_GPIO_Port,BUZZER_Pin,GPIO_PIN_SET);
delay(5500);
```

PHỤ LỤC

```
HAL_GPIO_WritePin(BUZZER_GPIO_Port,BUZZER_Pin,GPIO_PIN_RESET)
;
    delay(5500);
}
if(_times != 1)
    delay(3500000);
}

}

void VDD_detect(void)
{
getVDD = HAL_GPIO_ReadPin(DVCC_GPIO_Port,DVCC_Pin);
if(getVDD)
{
    GPIO_InitTypeDef GPIO_InitStruct;
    /*Configure GPIO pin : DVCC_Pin */
    GPIO_InitStruct.Pin = DVCC_Pin;
    GPIO_InitStruct.Mode = GPIO_MODE_IT_FALLING;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    HAL_GPIO_Init(DVCC_GPIO_Port, &GPIO_InitStruct);
}
else
{
    GPIO_InitTypeDef GPIO_InitStruct;
    /*Configure GPIO pin : DVCC_Pin */
    GPIO_InitStruct.Pin = DVCC_Pin;
    GPIO_InitStruct.Mode = GPIO_MODE_IT_RISING;
    GPIO_InitStruct.Pull = GPIO_NOPULL;
    HAL_GPIO_Init(DVCC_GPIO_Port, &GPIO_InitStruct);
}
```

PHỤ LỤC

```
}

void UserInfo_update(void)
{
    char _tmp_str[] = "NHA VUON THONG MINH\n+84383678987\n";
    char _size = sizeof(_tmp_str);
    for(int i = 0; i < _size; i++)
        EEPROM24XX_Save(i+2,&_tmp_str[i],1);
}

void GetSMS_control(void)
{
    if(getSMS)
    {
        Send_Command_SIM(SMS_List,OK,5000,0);
        HAL_Delay(500);
        strcpy(GetSMS_array,buffer2);
        Clear_Buffer2();
        while(strstr(GetSMS_array,"REC UNREAD")!=NULL)
        {
            ptr = strstr(GetSMS_array,"+84");
            strncpy(User_phoneNum,ptr,12); ptr += 12;
            if(SMS_getRTC)
            {
                ptr += 6;
                RTCdate.Year  = (*ptr - 0x30)*10; ptr+=1;
                RTCdate.Year  += (*ptr - 0x30);   ptr+=2;
                RTCdate.Month = (*ptr - 0x30)*10; ptr+=1;
                RTCdate.Month += (*ptr - 0x30);   ptr+=2;
                RTCdate.Date  = (*ptr - 0x30)*10; ptr+=1;
                RTCdate.Date  += (*ptr - 0x30);   ptr+=2;
            }
        }
    }
}
```

PHỤ LỤC

```
    RTCtime.Hours  = (*ptr - 0x30)*10; ptr+=1;
    RTCtime.Hours += (*ptr - 0x30);   ptr+=2;
    RTCtime.Minutes = (*ptr - 0x30)*10; ptr+=1;
    RTCtime.Minutes+= (*ptr - 0x30);   ptr+=2;
    RTCtime.Seconds = (*ptr - 0x30)*10; ptr+=1;
    RTCtime.Seconds+= (*ptr - 0x30);

    HAL_RTC_SetDate(&hrtc,&RTCdate,RTC_FORMAT_BIN);

    HAL_RTC_SetTime(&hrtc,&RTCtime,RTC_FORMAT_BIN);

    SMS_getRTC = 0;
}

if(strstr(GetSMS_array,"TTget"))
{
    sprintf(GetSMS_array,"TTget: %2d/%2d/%4d %2d:
%2d:
%2d\nHello!",postDay,postMonth,postYear,postHour,postMinute,postSecond);
    sendSMS(User_phoneNum,GetSMS_array);

    HAL_Delay(2000);

    ptr = strstr(GetSMS_array,"TTget");
}

if(strstr(GetSMS_array,"TTrst"))
{
    SMS_getRST = 1;
    sprintf(GetSMS_array,"TTrst: %2d/%2d/%4d %2d:
%2d:%2d\nReset
Module!",postDay,postMonth,postYear,postHour,postMinute,postSecond);
    sendSMS(User_phoneNum,GetSMS_array);

    HAL_Delay(2000);

    ptr = strstr(GetSMS_array,"TTrst");
}

ptr+=5;
```

PHỤ LỤC

```
        strcpy(GetSMS_array,ptr);
    }
    if(SMS_getRST)
    {
        Power_Off();
        debug = 1;
    }
    memset(GetSMS_array,0,sizeof(GetSMS_array));
    Send_Command_SIM(SMS_Delete,OK,1000,1);
    getSMS = 0;
}
}

void GetData_control(void)
{
    if(getData)
    {
        //buzz(2);
        //--Get HOME SW
        if(Get_HTTP_Data(HTTP_PARA3_SW_HOME))
        {
            strcpy(json_get_data,buffer2);
            HAL_Delay(300);
            Clear_Buffer2();
            ptr = strstr(json_get_data,"M16,,"); ptr+=5;
            strcpy(home_periph,ptr);
            ptr = strstr(home_periph,"\""); *ptr = 0;
        }
        else
        {
            if(debug_string)
            {
```

PHỤ LỤC

```
        strcpy(status,"GET DATA fail");
        HAL_Delay(200);
    }

    if(Check_HTTP_Para(HTTP_PARA_CHECK))
        Init_HTTP();
    }

    HAL_UART_Transmit(&huart3,(uint8_t*)str,sizeof(str),1000);
    getData = 0;
    GetData_PreTimer = timer1_ms;
}

}

void PostData_control(void)
{
    if(postData)
    {
        //Post GARDEN SS

        sprintf(json_post_data,"\"0,12345678m1M1OEdMigR1KU0yYcxSwdtlWrbjf+O
d9MkGW4JZr8=,Pr13,M17,,%s-%s-%s-%s-
%s\"",SHTtemp2,SHTumi2,CAPhumi1,lux,ppm);

        if(!Post_HTTP_Data(json_post_data,HTTP_PARA3_SS_GARD))
        {
            if(debug_string)
            {
                strcpy(status,"Post Data Fail!");
                HAL_Delay(2000);
            }
        }
        else
            strcpy(status,"Post Data Successful!");
    }
    //--Post HOME SS
}
```

PHỤ LỤC

```
sprintf(json_post_data,"\"0,12345678m1M1OEdMigR1KU0yYcxSwdtlWrbjf+O
d9MkGW4JZr8=,Pr13,M16,,%s-%s\"",SHTtemp1,SHThumi1);

    if(!Post_HTTP_Data(json_post_data,HTTP_PARA3_SS_HOME))
    {
        if(debug_string)
        {
            strcpy(status,"Post Data Fail!");
            HAL_Delay(2000);
        }
    }
    else
        strcpy(status,"Post Data Successful!");

    postData = 0;
    PostData_PreTimer = timer1_ms;
}

}

void updateLastGPS(void)
{
    HAL_RTC_GetTime(&hrtc,&RTCtime,RTC_FORMAT_BIN);
    HAL_RTC_GetDate(&hrtc,&RTCdate,RTC_FORMAT_BIN);
    postYear  = RTCdate.Year + 2000;
    postMonth = RTCdate.Month;
    postDay   = RTCdate.Date;
    postHour  = RTCtime.Hours;
    postMinute = RTCtime.Minutes;
    postSecond = RTCtime.Seconds;
    strcpy(GPS_latitude,GPS_Lastlatitude);
    strcpy(GPS_longitude,GPS_Lastlongitude);
    SMS_getRTC = 1;
```

PHỤ LỤC

```
}

/* USER CODE END 4 */

#ifndef USE_FULL_ASSERT
void assert_failed(uint8_t *file, uint32_t line)
{
    /* USER CODE BEGIN 6 */
    /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```
