**ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH**
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
**BỘ MÔN ĐIỆN TỬ**
----------o0o----------


**ĐỒ ÁN MÔN HỌC**

# HỆ THỐNG TƯỚI NƯỚC TỰ ĐỘNG

**GVHD: ThS. Nguyễn Trọng Luật**
**SVTH: Huỳnh Anh Tú**
**MSSV: 1713834**

**TP. HỒ CHÍ MINH, THÁNG 07 NĂM 2021**

# *LỜI CẢM ƠN*

Là sinh viên học tập tại Trường Đại Học Bách Khoa Hồ Chí Minh, bằng sự biết ơn và kính trọng, em xin chân thành cảm ơn các thầy cô đã nhiệt tình giảng dạy và giúp em có kiến thức để thực hiện đề tài này. Đặc biệt, em xin bày tỏ lòng biết ơn và lời cảm ơn sâu sắc tới thầy Nguyễn Trọng Luật, thầy đã trực tiếp hướng dẫn và giúp đỡ em trong suốt quá trình thực hiện đồ án môn học.

Do kiến thức cũng như kinh nghiệm còn hạn chế nên đề tài không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp từ thầy/cô để em có thêm kinh nghiệm và nền tảng cho công việc sau này.

Em xin chân thành cảm ơn!

*Tp. Hồ Chí Minh, ngày 20 tháng 7 năm 2021*

**Sinh viên**

**Huỳnh Anh Tú**

# MỤC LỤC

# DANH SÁCH HÌNH MINH HỌA

# TÓM TẮT ĐỒ ÁN

Trong thời đại công nghiệp 4.0, ngành nông nghiệp nước ta đứng trước yêu cầu đổi mới từ bỏ phương thức sản xuất thủ công, lạc hậu. Thay vào đó là việc ứng dụng kịp thời có chọn lọc các kỹ thuật tiên tiến của thời kì cách mạng 4.0 trong sản xuất nông nghiệp. Điều này sẽ góp phần tăng hiệu quả sản xuất và thương mại hóa sản phẩm, tăng tiện ích cho con người.

Đề tài gồm 6 chương :

Chương 1: Tổng quan đề tài "Hệ thống tưới cây tự động"

Chương 2: Cơ sở lí thuyết

Chương 3: Thiết kế hệ thống

Chương 4: Thi công hệ thống

Chương 5: Kết quả

Chương 6: Kết luận và hướng phát triển

Sau khi hoàn thành , kết quả phù hợp với tiêu chí đề ra giúp tiết kiệm thời gian và sức người trong việc chăm sóc cây. Sản phẩm phù hợp đặt trong các vườn cây, thảm cỏ …

# CHƯƠNG I : TỔNG QUAN ĐỀ TÀI

## 1.1 Đặt vấn đề

Việt Nam là một quốc gia đang phát triển, nông nghiệp vẫn giữ vai trò quan trọng trong nền kinh tế.

Quá trình hội nhập quốc tế đòi hỏi chất lượng nông sản càng cao; cùng với diện tích đất bị thu hẹp do đô thị hóa, do biến đổi khí hậu trong khi dân số tăng nên nhu cầu lương thực không ngừng tăng lên… là những thách thức rất lớn đối với sản xuất nông nghiệp.

Xuất phát từ những vấn đề thực tiễn trên em đã lựa chọn và tiến hành thiết kế mô hình hệ thống tưới tự động nhằm giảm bớt sức người cũng như tăng năng suất sản phẩm.

Hình 1-1 : Nông dân tưới nước thủ công

## 1.2 Mục đích đề tài

Để đáp ứng nhu cầu hội nhập quốc tế đòi hỏi nền nông nghiệp Việt Nam phải tăng cả năng suất cũng như chất lượng sản phẩm, song song với điều đó cũng cần phải làm giảm chi phí sản xuất để tăng tính cạnh tranh với quốc tế. Điều đó bắt buộc chúng ta phải áp dụng khoa học và công nghệ vào trong sản xuất.

Mục tiêu của đề tài là nghiên cứu, phân tích và thiết kế được mạch có khả năng điều khiển và giám sát hệ thống tưới nước tự động thông qua các thông tin thu thập về. Thông tin độ ẩm môi trường, độ ẩm đất được cảm biến đo chuyển tới khối xử lý dữ liệu. Dựa vào các thông tin trên để điều khiển máy bơm tưới nước tự động.

# CHƯƠNG II: CƠ SỞ LÍ THUYẾT

## 2.1 Khái niệm hệ thống tự động

Hệ thống điều khiển tự động là hệ thống, bao gồm các phần tử tự động nhằm điều khiển các quy trình xảy ra trong thiên nhiên, cuộc sống mà không có sự tham gia trực tiếp của con người.

Hiện nay, hệ thống tự động xuất hiện ngày càng phổ biến, gần như mọi lĩnh vực trong đời sống đều gắn với hệ thống tự động.

### 2.1.1 Trong y tế:

Tự động hóa trong các quy trình khám chữa bệnh, chăm sóc sức khỏe. Nhiều bệnh viện đã triển khai các robot trong các hoạt động khám bệnh, chẩn đoán bệnh, quản lý… Điều này giúp cho các bệnh viện, cơ sở y tế không bị quá tải và giúp bù đắp thiếu hụt nhân lực. Đặc biệt trong tình hình dịch bệnh phức tạp như hiện nay thì các hệ thống tự động là một giải pháp vô cùng thiết yếu.

Hình 2-1 : Robot tự động vận chuyển thuốc và thiết bị y tế

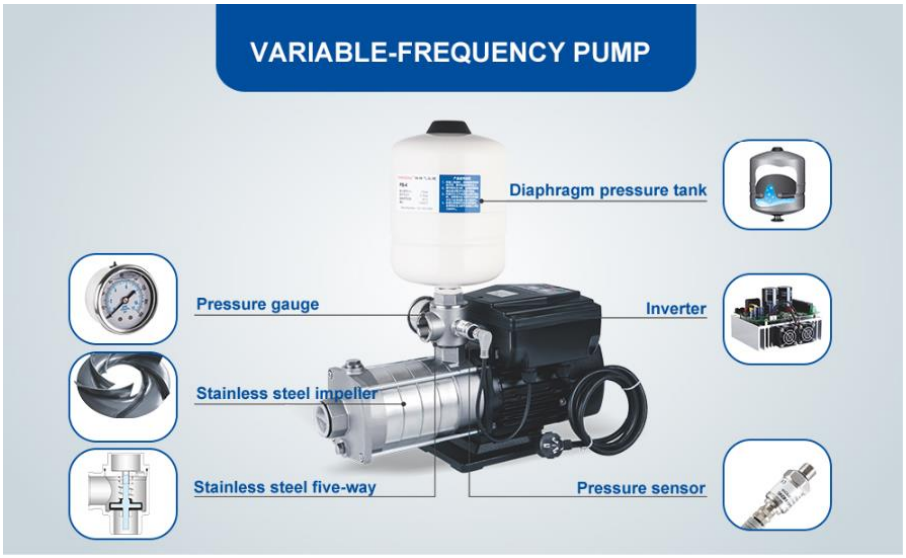
### 2.1.2 Trong giao thông

Nhiều hoạt động như tìm kiếm xe cứu hộ, phòng tránh ùn tắc giao thông đều được ứng dụng tự động hóa. Các hệ thống sẽ giám sát, cung cấp thông tin cho người dân để nắm bắt thông tin tốt nhất. Nó đã góp phần hỗ trợ kiểm soát giao thông, giúp đỡ hoạt động giao thông của người dân.

Hình 2-2 : Xe tự hành gửi đơn đặt hàng của Meituan Dianping ở Trung Quốc

### 2.1.3 Trong sinh hoạt hàng ngày

Các biến tần được sử dụng hợp lý trong các thiết bị quen thuộc. Điều này góp phần tiết kiệm năng lượng, chi phí cho người sử dụng. Đi cùng đó vẫn là công suất hoạt động mạnh mẽ thuận tiện trong quá trình sử dụng.




Hình 2-3 : Bơm biến tần giúp tiết kiệm điện năng tiêu thụ

### 2.1.4 Tự động hóa máy móc nông nghiệp

Việc phát triển nông nghiệp thông minh là vấn đề cấp thiết để cho ra nguồn thực phẩm sạch. Muốn có được nền nông nghiệp thông minh cần có sự kết hợp của nhiều lĩnh vực công nghệ. Đặc biệt các thiết bị máy móc sử dụng cần được ứng dụng tự động hóa – số hóa để tạo nên các chuỗi giá trị cho sản phẩm.

Các giải pháp tự động hóa được triển khai với quy mô lớn, tham gia vào hầu hết các khâu. Quá trình tự động hóa khi được áp dụng vào các thiết bị sẽ giúp tăng năng suất sản

xuất. Đặc biệt tạo ra số lượng lớn nhưng vẫn đảm bảo nguồn thực phẩm sạch theo đúng quy chuẩn. Doanh nghiệp, cá nhân cần kiểm soát tốt các khâu trong quá trình sản xuất để tránh sai sót.

Hình 2-4 : Máy cày tự hành của CNH Industrial

### 2.1.5 Tự động hóa máy móc công nghiệp

Tự động hóa trong công nghiệp đang ngày càng phát triển hơn. Các doanh nghiệp đều nhận thức được cần phải ứng dụng tự động hóa vào trong quy trình sản xuất. Các nhà máy, dây chuyền sản xuất đều được thiết lập các hệ thống tự động hoặc sử dụng robot. Các robot có công dụng rất lớn, đặc biệt có ích đối với những ngành sản xuất độc hại.

Hình 2-5 : Robot lắp ráp công nghiệp

Các giải pháp tự động hóa đã góp phần nào giảm tải sức lao động của con người. Cũng như phần nào giúp con người thực hiện cả những quá trình khó khăn, độc hại, tăng năng suất lên gấp đôi, thậm chí gấp ba. Các quy trình công nghiệp thường được thiết kế, lập trình các bộ điều khiển chung trong khi xử lý. Xu hướng hiện tại trong các doanh nghiệp đang gia tăng sử dụng các

robot để thực hiện các chức năng kiểm tra, hướng dẫn xử lý.

## 2.2 Ưu điểm và nhược điểm của hệ thống tự động

**Ưu điểm :**

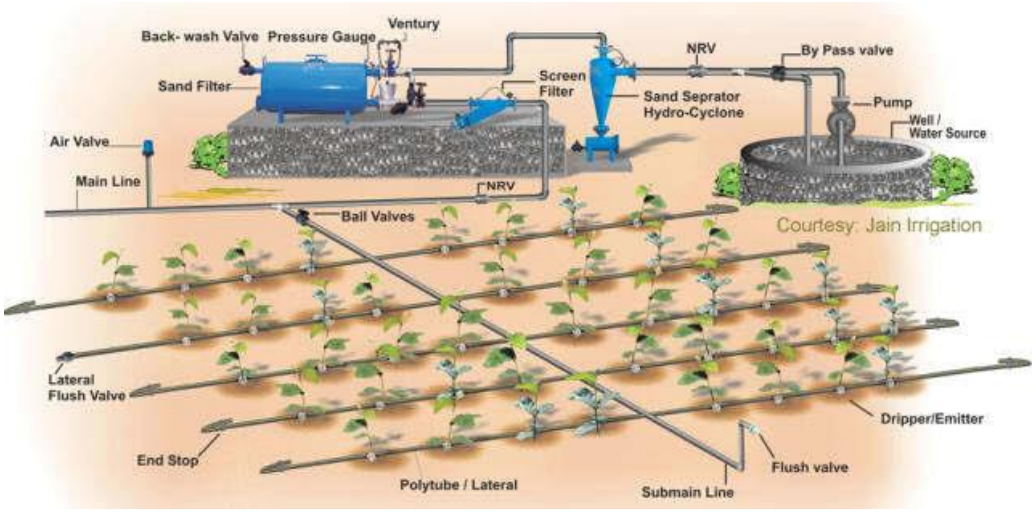
- Tăng thông lượng hoặc năng suất.
- Cải thiện chất lượng hoặc tăng khả năng dự báo về chất lượng.
- Cải thiện mạnh mẽ (thống nhất), quy trình hay sản phẩm.
- Tăng tính nhất quán của đầu ra.
- Giảm chi phí nhân công trực tiếp và chi phí nhân lực
- Cung cấp công việc ở cấp cao hơn trong việc phát triển, triển khai, bảo trì và hoạt động của các quá trình tự động.
- Thực hiện nhiệm vụ đó là vượt quá khả năng của con người về kích thước, trọng lượng, tốc độ, sức chịu đựng, …

**Khó khăn :**

- Dễ bị hư hỏng: Một hệ thống tự động có thể có một mức giới hạn của trí thông minh, nó cũng có thể bị trục trặc gây ra những việc ngoài ý muốn. Do đó phải kiểm tra bảo trì định kỳ.
- Không thể đoán trước, chi phí phát triển quá mức: Các nghiên cứu và phát triển chi phí của tự động hoá một quá trình có thể vượt quá chi phí tiết kiệm bằng cách tự động hóa bản thân.
- Chi phí ban đầu cao: Việc tự động hóa của một mới sản phẩm hoặc thực vật thường đòi hỏi một sự đầu tư ban đầu rất lớn so với chi phí đơn vị sản phẩm, mặc dù chi phí tự động hóa có thể được lan truyền trong nhiều sản phẩm và thời gian.
- Đối mặt với tình trạng thất nghiệp: Khi các máy móc tự động hóa thay thế lao động tay chân sẽ dẫn đến giờ làm việc thấp hoặc yêu cầu người lao động phải có những kiến thức cao để có thể thay đổi công việc.

## 2.3 Ứng dụng của tự động hóa trong tưới tiêu cây trồng

Công trường thực vật là căn cứ địa sản xuất nông nghiệp của hiện đại hóa. Toàn bộ quá trình đều có thể điều khiển tự động để giảm bớt sức người, nâng cao sản lượng… Mặc dù tự động hóa ứng dụng từ rất lâu cho việc tưới tiêu, song nó chỉ phát triển ở một số nước phát triển, còn đối với các nước chậm phát triển tuy nền nông nghiệp chiếm tỉ lệ lớn nhưng việc ứng dụng tự động hóa cho việc tưới cây vẫn còn rất chậm. Hiện nay, được sự trợ giúp của nước ngoài các nước đang phát triển đã đưa dần tự động hóa vào đời sống và sản xuất, đặc biệt là các nước Đông Nam Á nói chung và trong đó có Việt Nam.




Hình 2-6 : Hệ thống tưới nước nhỏ giọt

Ngày nay với sự phát triển mạnh mẽ của công nghệ chế tạo thiết bị tự động hóa, kết hợp với thành tựu trong công nghệ vi điện tử và công nghệ thông tin, đã cho phép tạo nên một giải pháp tự động hóa trong mọi lĩnh vực không chỉ trên lĩnh vực nông nghiệp. Có thể nói tự động hóa trở thành xu hướng tất yếu cho bất kì quốc gia, vùng lãnh thổ nào muốn phát triển kinh tế trên Thế giới.

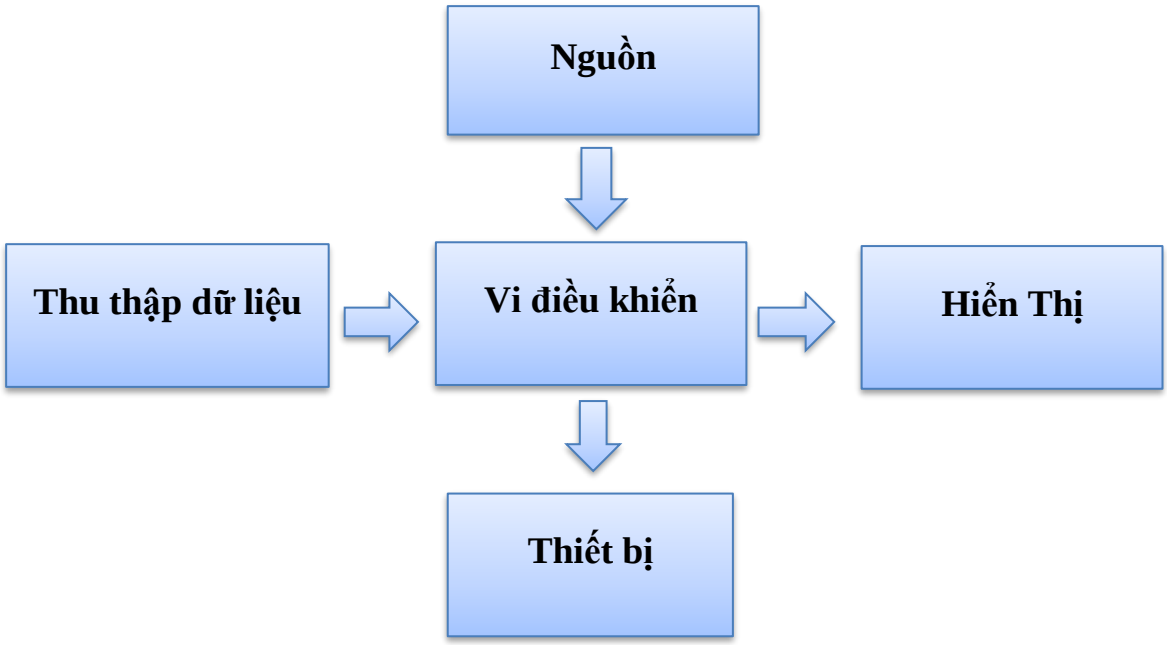
# CHƯƠNG III: THIẾT KẾ HỆ THỐNG

## 3.1 Giới thiệu về mô hình hệ thống

Hệ thống gồm 4 phần chính :

- Phần thứ nhất là khối thu thập dữ liệu gồm các cảm biến nhiệt độ-độ ẩm DHT-11, cảm biến mưa. Các cảm biến này sẽ thu thập thông tin trong môi trường hiện tại sau đó sẽ gửi thông tin đến trung tâm xử lí bằng công nghệ truyền dẫn không dây.
- Phần thứ hai là bộ điều khiển trung tâm có nhiệm vụ nhận tín hiệu gửi về từ khối thu thập dữ liệu xử lí . Thông tin sẽ được gửi đến người dùng qua wifi.
- Phần thứ ba là khối thiết bị gồm có máy bơm nước. Khi đạt đủ điều kiện hệ thống điều khiển sẽ bật ( tắt ) máy bơm cung cấp nước cho cây trồng.
- Phần thứ tư là khối hiển thị gồm có một màn hình LCD 16x2 để hiện thị thông số hiện tại tại nơi đặt thiết bị. Ngoài ra, thông tin còn được hiển thị qua phần mềm android.

## 3.2 Thiết kế sơ đồ khối cho hệ thống




Hình 3-1 Sơ đồ khối của hệ thống

## 3.3 Chức năng của từng khối

### 3.3.1 Vi điều khiển

Vi điều khiển là một máy tính được tích hợp trên một chip, nó thường được sử dụng để điều khiển các thiết bị điện tử. Vi điều khiển, thực chất, là một hệ thống bao gồm một vi xử lý có hiệu suất đủ dùng và giá thành thấp (khác với các bộ vi xử lý đa năng dùng trong máy tính) kết hợp với các ngoại vi như bộ nhớ, các module vào/ra, các module biến đổi số sang tương tự và tương tự sang số,... Ở máy tính thì các module thường được xây dựng bởi các chip và mạch ngoài.

Vi điều khiển thường được dùng để xây dựng các hệ thống nhúng. Nó xuất hiện khá nhiều trong các thiết bị điện, điện tử, máy giặt, lò vi sóng, điện thoại, đầu đọc DVD, thiết bị đa phương tiện, dây chuyền tự động,...

**KIT điều khiển ESP8266 NodeMCU**

Kit RF thu phát Wifi ESP8266 NodeMCU Lua CP2102 là kit phát triển dựa trên nền chip Wifi SoC ESP8266 với thiết kế dễ sử dụng và đặc biệt là có thể sử dụng trực tiếp trình biên dịch của Arduino để lập trình và nạp code, điều này khiến việc sử dụng và lập trình các ứng dụng trên ESP8266 trở nên rất đơn giản.

Khả năng lưu trữ và xử lý mạnh mẽ cho phép nó được tích hợp với các bộ cảm biến, vi điều khiển và các thiết bị ứng dụng cụ thể khác thông qua GPIOs với chi phí tối thiểu và một PCB tối thiểu. Với mức độ tích hợp cao trên chip, trong đó bao gồm các anten chuyển đổi balun, bộ chuyển đổi quản lý điện năng…Kit RF thu phát Wifi ESP8266 NodeMCU Lua CP2102 được dùng cho các ứng dụng cần kết nối, thu thập dữ liệu và điều khiển qua sóng Wifi, đặc biệt là các ứng dụng liên quan đến IoT.

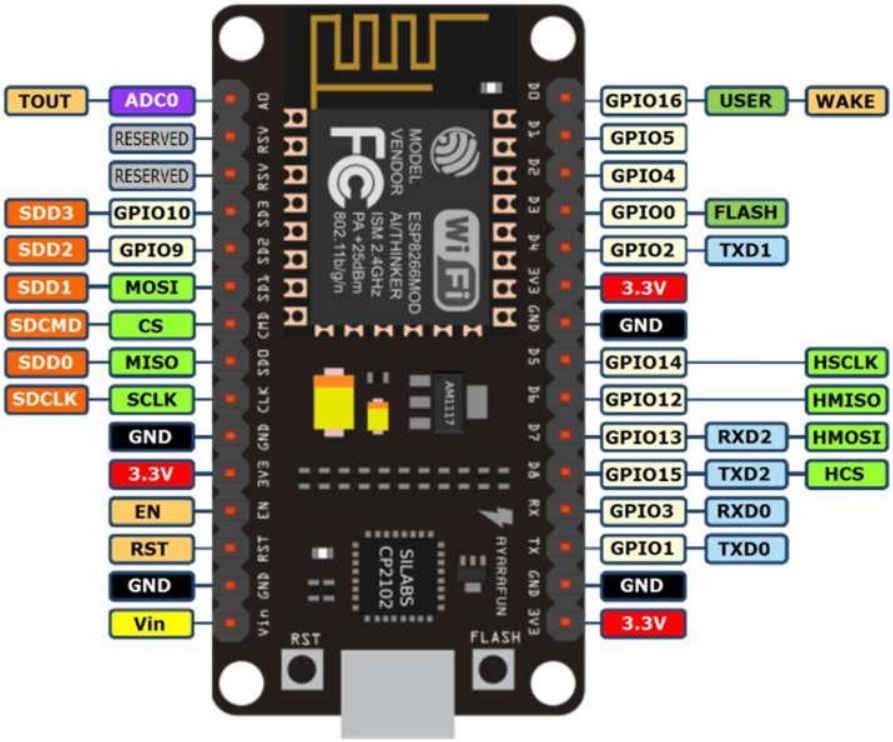
- ESP8266 cung cấp một giải pháp kết nối mạng Wi-Fi hoàn chỉnh và khép kín, cho phép nó có thể lưu trữ các ứng dụng hoặc để giảm tải tất cả các chức năng kết nối mạng Wi-Fi từ một bộ xử lý ứng dụng.
- Luôn phiên, phục vụ như một bộ chuyển đổi Wi-Fi, truy cập internet không dây có thể được thêm vào bất kỳ thiết kế vi điều khiển nào dựa trên kết nối đơn giản qua giao diện UART hoặc giao diện cầu CPU AHB.
- Khả năng lưu trữ và xử lý mạnh mẽ cho phép nó được tích hợp với các bộ cảm biến, vi điều khiển và các thiết bị ứng dụng cụ thể khác thông qua GPIOs với chi phí tối thiểu và một PCB tối thiểu. Với mức độ tích hợp cao trên chip, trong đó bao gồm các anten chuyển đổi balun, bộ chuyển đổi quản lý điện năng.

Hình 3-2 : Hình ảnh thực tế ESP8266 NodeMCU

Thông số kỹ thuật

| | |
|---|---|
| Chip điều khiển | ESP8266EX |
| WiFi | 2.4 GHz hỗ trợ chuẩn 802.11 b/g/n |
| Điện áp hoạt động | 3.3 V |
| Điện áp đầu vào | 5V (thông qua cổng USB) |
| Số chân I/O | 11 (tất cả các chân I/O đều có Interrupt/PWM/I2C/One-wire, trừ chân D0) |
| Số chân Analog Input | 1 (điện áp vào tối đa 3.3V) |
| Bộ nhớ Flash | 4MB |
| Hỗ trợ bảo mật | WPA/WPA2 |
| Tích hợp giao thức | TCP/IP |
| Ngôn ngữ lập trình | C/C++, Micropython, NodeMCU - Lua |

Sơ đồ chân ESP8266 Node MCU




Hình 3-3 : Sơ đồ chân của ESP8266 NodeMCU

Chức năng của các chân:

- VCC: Điện áp 3.3V .
- GND: Chân nối đất.
- Tx: Chân Tx của giao thức UART, kết nối đến chân Rx của vi điều khiển.
- Rx: Chân Rx của giao thức UART, kết nối đến chân Tx của vi điều khiển.
- RST: chân reset, kéo xuống mass để reset.
- 10 chân GPIO từ D0 – D8, có chức năng PWM, IIC, giao tiếp SPI, 1- Wire và ADC trên chân A0.

Tính năng của NODEMCU ESP8266:

- WiFi: 2.4 GHz hỗ trợ chuẩn 802.11 b/g/n, hỗ trợ WPA/WPA2.
- Điện áp cung cấp : DC 5 ~ 9V.
- Bộ nhớ Flash: 32MB.
- Chuẩn giao tiếp nối tiếp UART với tốc độ Baud lên đến115200
- Tích hợp ngăn xếp giao thứcTCP / IP.

- Tích hợp chuyển đổi TR, balun, LNA, bộ khuếch đại công suất và phù hợp với mạng.
- Hỗ trợ nhiều loại anten.
- Wake up và truyền các gói dữ liệu trong <2ms.
- Hỗ trợ cả 2 giao tiếp TCP và UDP.
- Hỗ trợ các chuẩn bảo mật như: OPEN, WEP, WPA_PSK, WPA2_PSK, WPA_WPA2_PSK
- Có 3 chế độ hoạt động: Client, Access Point, Both Client and Access Point.
- Quản lý năng lượng NODEMCU ESP8266
- ESP8266 được thiết kế cho điện thoại di động, điện tử lắp ráp và ứng dụng InternetofThings với mục đích đạt được mức tiêu thụ điện năng thấp nhất với sự kết hợp của nhiềukỹ thuật độc quyền. Kiến trúc tiết kiệm năng lượng hoạt động trong 3 chế độ: chế độ hoạt động, chế độ ngủ và chế độ ngủ sâu.
- Bằng cách sử dụng các kỹ thuật quản lý nguồn điện và kiểm soát chuyển đổi giữa chế độ ngủ ESP8266 tiêu thụ chưa đầy 12uA ở chế độ ngủ nhỏ hơn 1.0mW so với (DTIM = 3) hoặc ít hơn 0.5mW (DTIM = 10) để giữ kết nối với các điểm truy cập.
- Khi ở chế độ ngủ, chỉ có bộ phận hiệu chỉnh đồng hồ thời gian thực và cơ quan giám sát vẫn hoạt động. Đồng hồ thời gian thực có thể được lập trình để đánh thức ESP8266 ở bất kỳ khoảng thời gian cần thiết nào.
- ESP8266 có thể được lập trình để thức dậy khi một điều kiện chỉ định được phát hiện. Tính năng tối thiểu thời gian báo thức này của ESP8266 có thể được sử dụng bởi Tính năng tối thiểu thời gian báo thức của ESP8266 có thể được sử dụng bởi thiết bị di động SOC. Cho phép chúng vẫn ở chế độ chờ, điện năng thấp cho đến khi Wifi là cần thiết.
- Để đáp ứng nhu cầu điện năng của thiết bị di động và điện tử lắp giáp, ESP8266 có thể được lập trình để giảm công suất đầu ra của PA phù hợp với các ứng dụng khác nhau. Bằng việc tắt khoảng tiêu thụ năng lượng.

Các chip có thể được thiết lập ở các trạng thái sau:

- **OFF**: chân CHIP_PD ở mức thấp. Các RTC (đồng hồ thời gian) bị vô hiệu hóa và mọi thanh ghi sẽ bị xóa.
- **SLEEP DEEP:** Các RTC được kích hoạt, khi đó các phần còn lại của chip sẽ ở trạng thái off. RTC phục hồi bộ nhớ nội bộ để lưu trữ các thông tin kết nối WiFi cơ bản.

- **SLEEP**: Chỉ RTC hoạt động. Các dao động tinh thể được vô hiệu hóa. Bất kỳ sự kiện wakeup (MAC, host, RTC hẹn giờ, ngắt ngoài) sẽ đưa chip vào trạng thái wakeup.
- **Wakeup:** Trong trạng thái này, hệ thống đitừ trạng thái ngủ sang trạng thái PWR. Các dao động tinh thể và PLLs được kích hoạt.
- **Trạng thái ON:** Xung clock tốc độ cao hoạt động và gửi đến mỗi khối được kích hoạtbằng cách đăng ký kiểm soát xung clock. Mức độ thấp hơnclock gating được thực hiện ở cấp khối, bao gồm cả CPU, có thể đạt được bằng cách sử dụng lệnh WAIT, trong khi hệ thống trên off.

### 3.3.2 Thu thập dữ liệu

Cảm biến là thiết bị điện tử cảm nhận những trạng thái hay quá trình vật lý, hóa học hay sinh học của môi trường cần khảo sát, và biến đổi thành tín hiệu điện để thu thập thông tin về trạng thái hay quá trình đó.

Một số loại cảm biến nhiệt độ độ ẩm thông dụng :


Hình 3-4 : Cảm biến LM35

Cảm biến nhiệt độ LM35 có điện áp Analog đầu ra tuyến tính theo nhiệt độ thường được sử dụng để đo nhiệt độ của môi trường hoặc theo dõi nhiệt độ của thiết bị,..., cảm biến có kiểu chân TO-92 với chỉ 3 chân rất dễ giao tiếp và sử dụng.

*Thông số kỹ thuật:*

- Điện áp hoạt động: 4~20VDC
- Công suất tiêu thụ: khoảng 60uA
- Khoảng đo: -55°C đến 150°C
- Điện áp tuyến tính theo nhiệt độ: 10mV/°C

- Sai số: 0.25°C
- Kiểu chân: TO92
- Kích thước: 4.3 × 4.3mm

Hình 3-5 : Cảm biến độ ẩm HS1101

Cảm biến độ ẩm HS1101 Humidity Sensor được sử dụng để đo độ ẩm môi trường, cảm biến hoạt động như 1 tụ điện khi độ ẩm tăng điện dung của cảm biến sẽ tăng, ngược lại khi độ ẩm giảm giá trị điện dung sẽ giảm tương ứng, cảm biến có chất lượng tốt, độ bền cao.

Lưu ý không sử dụng cảm biến trong các môi trường yếm khí, lên men sẽ làm hư hỏng bề mặt cảm biến, chỉ sử dụng trong môi trường bình thường hoặc thuần hơi nước.

*Thông số kỹ thuật:*

- Điện áp hoạt động: 5~10V.
- Giao tiếp: ngõ ra analog.
- Dải đo của độ ẩm: 1 đến 99%RH.
- Nhiệt độ hoạt động: -40 đến 100$^{o}$C.

Hình 3-6 Cảm biến nhiệt độ độ ẩm DHT 11

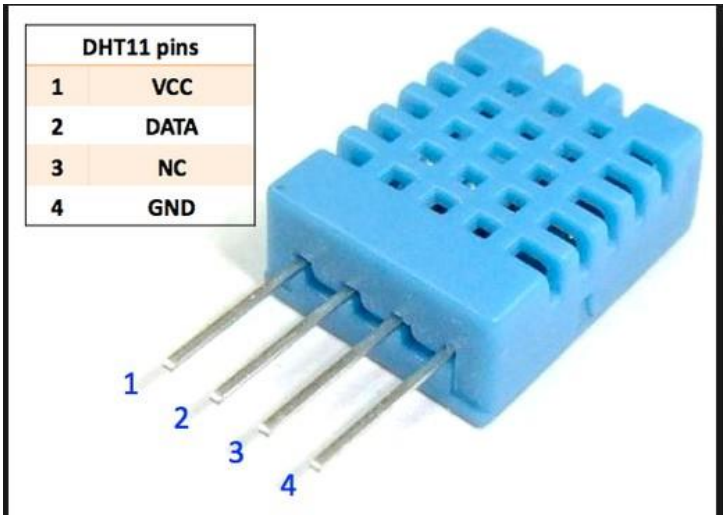
DHT11 Là cảm biến nhiệt độ, độ ẩm rất thông dụng hiện nay vì chi phí rẻ và rất dễ lấy dữ liệu thông qua giao tiếp 1-wire ( giao tiếp digital 1-wire truyền dữ liệu duy nhất). Cảm biến được tích hợp bộ tiền xử lý tín hiệu giúp dữ liệu nhận về được chính xác mà không cần phải qua bất kỳ tính toán nào.

*Đặc điểm:*

- Điện áp hoạt động : 3V - 5V (DC)
- Dải độ ẩm hoạt động : 20% - 90% RH, sai số ±5%RH
- Dải nhiệt độ hoạt động : 0°C ~ 50°C, sai số ±2°C
- Tần số lấy mẫu tối đa: 1 Hz ( 1 giây/lần )
- Khoảng cách truyền tối đa: 20m

Sơ đồ chân Cảm biến DHT11 gồm 2 chân cấp nguồn, và 1 chân tín hiệu. Hiện nay, thông dụng ngoài thị trường có hai loại đóng gói cho DHT11: 3 chân và 4 chân.


Hình 3-7 : Cảm biến DHT 11 loại 3 chân



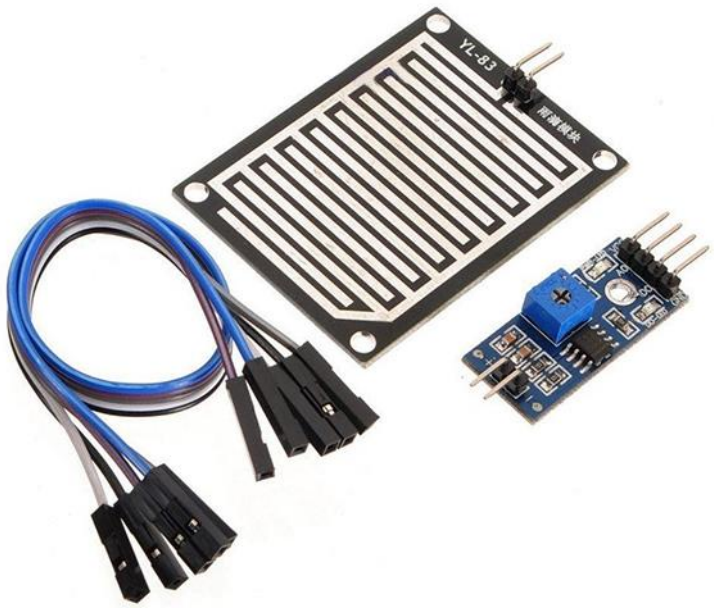

Hình 3-8 : Cảm biến DHT 11 loại 4 chân

Nhận xét: Cảm biến nhiệt độ, độ ẩm DHT11 với giá thành rẻ, dễ sử dụng, thích hợp sử dụng trong các ứng dụng yêu cầu độ chính xác không cao, môi trường không khắc nghiệt.

*Nguyên lý hoạt động:*

Để có thể giao tiếp với DHT11 theo chuẩn 2 chân vi xử lý thực hiện theo 2 bước:

- Gửi tín hiệu muốn đo (Start) tới DHT11 xác nhận lại.
- Khi giao tiếp được với DHT11, Cảm biến sẽ gửi lại 5 byte dữ liệu và nhiệt độ đo được.



Hình 3-9 : Module Cảm biến mưa

Mạch cảm biến mưa là một module cảm biến được sử dụng rộng rãi trong việc phát hiện mưa vì ưu điểm dễ dàng lắp đặt và chi phí thấp. Cảm biến hoạt động bằng cách so sánh điện áp của mạch ngoài trời với giá trị đã được đặt trước thông qua biến trở trên cảm biến. Từ đó module dễ dàng điều khiển đóng cắt relay.

*Thông số kỹ thuật**:***

- Kích thước: 5.4*4.0 mm
- Dày 1.6 mm
- Điện áp: 5V
- Led báo nguồn ( Màu xanh)
- Led cảnh báo mưa ( Màu đỏ)
- Hoạt động dựa trên nguyên lý: Nước rơi  vào phần cảm biến sẽ tạo điện ra áp trên cấc chân D0 A0
- Có 2 dạng tín hiệu Analog (A0) và Digital (D0).
- Dạng tín hiệu : TTL, đầu ra 100mA ( Có thể sử dụng trực tiếp Relay, Còi công suất nhỏ...)
- Điều chỉnh độ nhạy bằng biến trở.Sử dụng LM358 để chuyển AO --> DO.

*Nguyên lí hoạt động:*

- Khi có nước rơi trên cảm biến, sẽ có điện áp trong khoảng từ 0V đến 5V trên chân A0 và được đưa vào bộ so sánh sử dụng ic LM393, để đưa ra chân D0 điện áp mức 0 hoặc mức 1. Biến trở có tác dụng điều chỉnh đó nhạy, bạn có thể tùy ý quyết định với lượng mưa nào thì cảm biến sẽ đưa ra mức 1.

- Ngoài ra, cảm biến còn đưa trực tiếp chân A0 ra cho các bạn có thể tiến hành đo lường, xác định lưu lượng mưa bằng cách giao tiếp với vi điều khiển và các bộ chuyển đổi ADC.

Trong đề tài này cảm biến mưa có tác dụng để ngắt máy bơm nếu trời mưa nhằm tiết kiệm nước và đảm bảo cho cây trồng không bị úng nước.

### 3.3.3 Hiển thị

**Màn hình hiển thị LCD 1602A**

Ngày nay, thiết bị hiên thị LCD (Liquid Crystal Display) được sử dụng rất nhiều các ứng dụng của vi điều khiển. LCD có rất nhiều ưu điểm so với các dạng hiển thị khác. Nó có khả năng hiển thị kí tự đa dạng, trực quan (chữ, số và kí tự đồ họa) dễ dàng đưa vào mạch ứng dụng theo nhiều giao thức giao tiếp khác nhau, tốn rất ít tài nguyên và giá thành rẻ...

**Ưu điểm:**

- Hiển thị kí tự đa dạng , trực quan ( chữ , số và kí tự đồ họa )
- Dễ dàng đưa vào mạch ứng dụng theo nhiều giao thức giao tiếp khác nhau
- Tốn ít tài nguyên hệ thống và giá thành rẻ

**Nhược điểm:** Vì mật độ điểm ảnh trên màn LCD rất thấp nên khi ra ngoài ánh sáng mặt trời thì màu sắc hiển thị rất kém cũng như dễ nhìn thấy các hoạt điểm ảnh trên màn hình.

**Thông số kĩ thuật màn hình 1602A**

Màn hình text LCD1602 sử dụng driver HD44780, có khả năng hiển thị 2 dòng với mỗi dòng 16 ký tự, màn hình có độ bền cao, rất phổ biến, nhiều code mẫu và dễ sử dụng thích hợp cho những người mới học và làm dự án.

- Điện áp hoạt động là 5 V.
- Kích thước: 80 x 36 x 12.5 mm
- Chữ đen, nền xanh
- Khoảng cách giữa hai chân kết nối là 0.1 inch tiện dụng khi kết nối với Breadboard.
- Tên các chân được ghi ở mặt sau của màn hình LCD hỗ trợ việc kết nối, đi dây điện.
- Có đèn led nền, có thể dùng biến trở hoặc PWM điều chỉnh độ sáng để sử dụng ít điện năng hơn.
- Có thể được điều khiển với 6 dây tín hiệu


Hình 3-10 : Màn hình LCD 1602A

Sơ đồ chân :

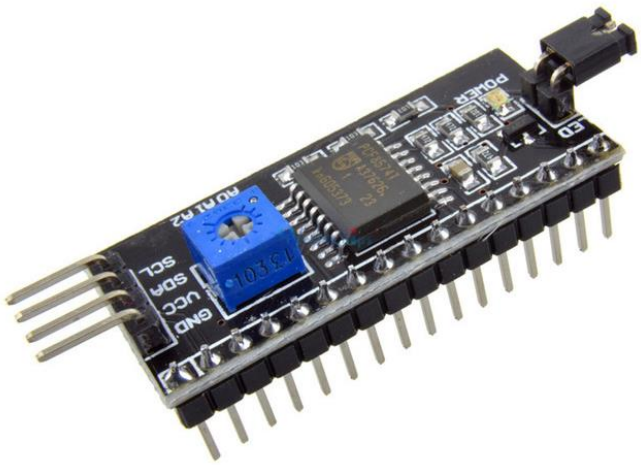
| Chân | Ký hiệu | Mô tả | Giá trị |
|---|---|---|---|
| 1 | VSS | GND | 0V |
| 2 | VCC | | 5V |
| 3 | V0 | Độ tương phản | |
| 4 | RS | Lựa chọn thanh ghi | RS=0 (mức thấp) chọn thanh ghi lệnh<br>RS=1 (mức cao) chọn thanh ghi dữ liệu |
| 5 | R/W | Chọn thanh ghi đọc/viết dữ liệu | R/W=0 thanh ghi viết<br>R/W=1 thanh ghi đọc |
| 6 | E | Enable | |
| 7 | DB0 | Chân truyền dữ liệu | 8 bit: DB0-DB7 |
| 8 | DB1 | | |
| 9 | DB2 | | |
| 10 | DB3 | | |
| 11 | DB4 | | |
| 12 | DB5 | | |
| 13 | DB6 | | |
| 14 | DB7 | | |
| 15 | A | Cực dương led nền | 0V đến 5V |
| 16 | K | Cực âm led nền | 0V |

*Chức năng các chân:*

- Vss: Chân nối đất cho LCD
- VDD: Chân cấp nguồn cho LCD, ta nối chân này với VCC= 5V
- VEE: Điều chỉnh độ tương phản của LCD.

- RS: Chân chọn thanh ghi (Registor select). Nối chân RS với mức logic "0" (GND) hoặc mức logic "1" (VCC) để chọn thanh ghi. Logic "0": Bus DB0 -> DB7 sẽ nối với thanh ghi lệnh I của LCD (ở chế độ “ghi" – write) hoặc nối với bộ đếm địa chỉ của LCD (ở chế độ “đọc" - read). Logic "1": Bus DB0 -> DB7 sẽ nối với thanh ghi DR bên trong LCD. Chân chọn chế độ đọc/ghi (Read/Write). Nối chân R/W với mức logic "0" để LCD hoạt động ở chế độ ghi, hoặc R/W nối với logic "1" để LCD ở chế độ đọc.
- R/W: Chân cho phép (Enable). Sau khi các tín hiệu được đặt lên 9 bus DB0-> DB7, các lệnh chỉ được chấp nhận khi có 1 xung cho phép của chấn E.
    - Chế độ ghi: Dữ liệu ở bus sẽ được LCD chuyển vào (chấp nhận) thanh ghi bên trong nó khi phát hiện một xung (high- to- low transition) của tín hiệu chân E.
    - Chế độ đọc: Dữ liệu được LCD xuất ra DB0> DB7 khi phát hiện sườn lên (low - to- high transition) ở chân E và được LCD giữ ở bus đến khi nào chân E xuống mức thấp
- DB0 ->DB7: Tám đường của bus dữ liệu dùng để trao đổi thông tin với MPU.Có 2 chế độ sử dụng đường 8 bus này :
    - Chế độ 8 bit: Dữ liệu được truyền trên cả 8 đường với bit MSB là bit DB7.
    - Chế độ 4 bit: Dữ liệu được truyền trên 4 đường từ DB4 -> DB7 bit MSB là bit DB7.

**Module I2C Arduino**




Hình 3-11 Module I2C LCD 16x2

LCD có quá nhiều chân gây khó khăn trong quá trình kết nối và chiếm dụng nhiều chân của vi điều khiển. Module chuyển đổi I2C cho LCD sẽ giải quyết vấn đề này cho bạn, thay vì sử dụng tối thiểu 6 chân của vi điều khiển để kết nối với LCD (RS, EN, D7, D6, D5 và D4) thì với module chuyển đổi bạn chỉ cần sử dụng 2 chân (SCL, SDA) để kết nối. Module chuyển đổi I2C hỗ trợ các loại LCD sử dụng driver HD44780(LCD 1602, LCD 2004, … ), kết nối với vi điều khiển thông qua giao tiếp I2C, tương thích với hầu hết các vi điều khiển hiện nay.

**Ưu điểm :**

- Tiết kiệm chân cho vi điều khiển
- Dễ dàng kết nối với LCD

Sơ đồ chân :

| | |
|---|---|
| Chân 1 | Vcc |
| Chân 2 | SDA/TX |
| Chân 3 | SCL/RX |
| Chân 4 | GND (0V) |
| Chân 5 | Output-đầu ra quét xung đến cột 2 |
| Chân 6 | Input- nhận các trạng thái hàng một |
| Chân 7 | Output- đầu ra quét xung đến cột một |

| | |
|---|---|
| Chân 8 | Input- nhận các trạng thái hàng bốn |
| Chân 9 | Output- đầu ra quét xung đến cột ba |
| Chân 10 | Input- nhận các trạng thái của hàng ba |
| Chân 11 | Input- nhận các trạng thái hàng hai |
| Chân 12 | Chế độ nhảy: Khi mở điều khiển I2C,<br>khi bị đóng điều khiển nối tiếp |

*Thông số kĩ thuật:*

- Điện áp hoạt động: 2.5-6V DC
- Hỗ trợ màn hình: LCD1602,1604,2004 (driver HD44780)
- Giao tiếp: I2C
- Địa chỉ mặc định: 0X27 (có thể điều chỉnh bằng ngắn mạch chân A0/A1/A2)
- Kích thước: 41.5mm(L)x19mm(W)x15.3mm(H)
- Trọng lượng: 5g
- Tích hợp Jump chốt để cung cấp đèn cho LCD hoặc ngắt
- Tích hợp biến trở xoay điều chỉnh độ tương phản cho LCD

### 3.3.4 Thiết bị

Hình 3-12 Máy bơm nước 1 chiều R385 5-12V

Máy bơm áp lực mini 12V/12W/2L Smartpumps DP-12W2L là dòng máy bơm mini nhỏ lưu lượng 2 lít/phút có khả năng tự hút. Được dùng bơm thực phẩm, bơm làm mát máy khoan, bơm định lượng, bơm nước hồ cá, bơm tưới và bơm chất lỏng, bơm phun xịt khử trùng, bơm hóa chất diệt khuẩn, máy bơm nước rửa tay, máy bơm nước sạch và bơm phun sương.

*Thông số kỹ thuật:*

- Điện áp Vào Jack DC 5.5x2.1mm, DC6- 12V (đề nghị 9V-1A hoặc 12V-1A)
- Dòng Tiêu Thụ: 0.5-0.7A
- Đường kính động cơ: 27mm đường kính đầu: 44mm Tổng chiều dài: 60mm
- Lưu lượng: 1,5-2 L / Min
- Chiều dài ống hút tối đa: 2 m
- Đẩy Cao: lên đến 3 mét
- Tuổi Thọ: lên đến 2500H
- Nhiệt độ nước cho phép: lên đến 80 độ C

# CHƯƠNG IV: THI CÔNG HỆ THỐNG

## 4.1 Mô Hình Thực Nghiệm

### 4.1.1 Phần mềm mô phỏng Altium


Hình 4-1 : Phần mềm Altium Designer

Altium Designer trước kia có tên gọi quen thuộc là Protel DXP, là một trong những công cụ vẽ mạch điện tử mạnh nhất hiện nay. Được phát triển bởi hãng Altium Limited. Altium designer là một phần mềm chuyên nghành được sử dụng trong thiết kế mạch điện tử.

Altium ngày nay đang là một trong những phần mềm vẽ mạch điện tử mạnh và được ưa chuộng ở Việt Nam. Ngoài việc hỗ trợ tốt cho hoạt động vẽ mạch, Altium còn hỗ trợ tốt trong việc quản lý mạch, trích xuất file thống kê linh kiện.

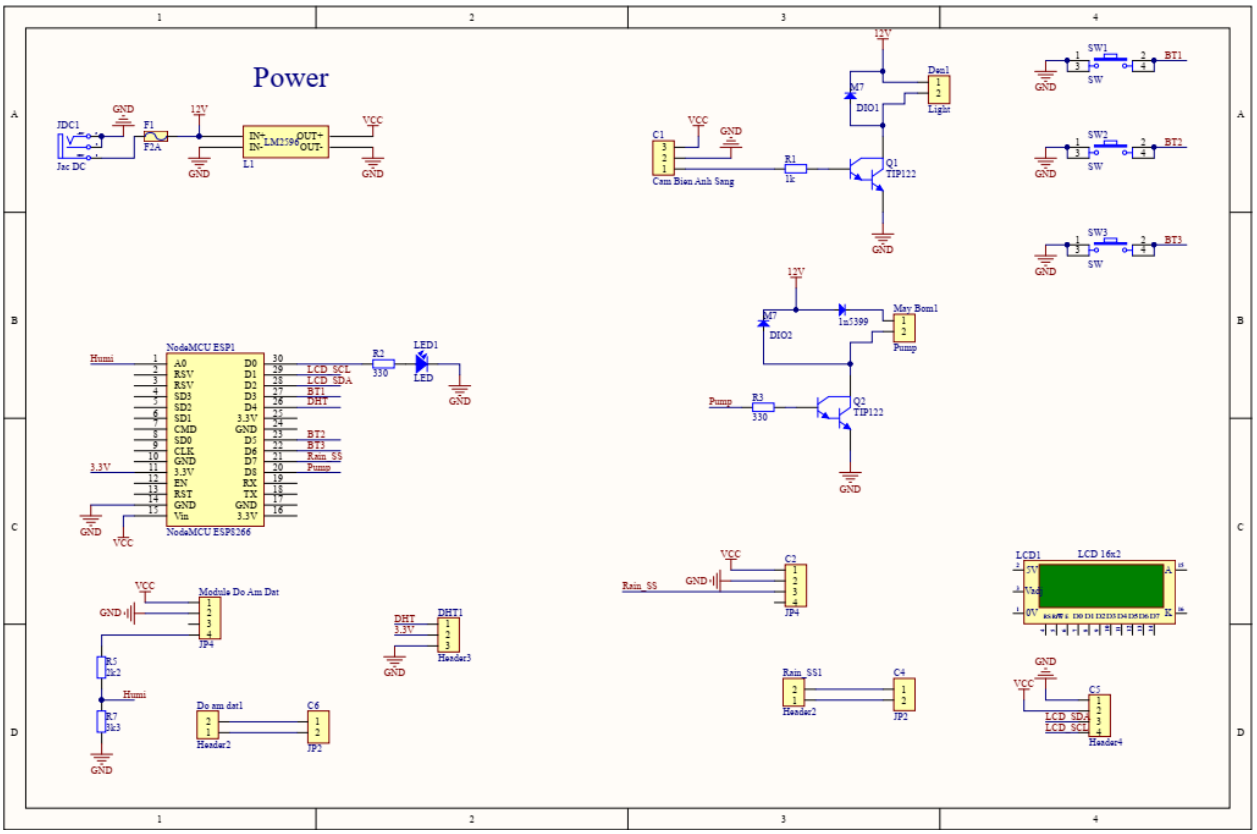
Một số điểm vượt trội của Altium:

- Giao diện thiết kế, quản lý và chỉnh sửa thân thiện, dễ dàng biên dịch, quản lý file, quản lý phiên bản cho các tài liệu thiết kế.
- Hỗ trợ mạnh mẽ cho việc thiết kế tự động, đi dây tự động theo thuật toán tối ưu, phân tích lắp ráp linh kiện. Hỗ trợ việc tìm các giải pháp thiết kế hoặc chỉnh sửa mạch, linh kiện, netlist có sẵn từ trước theo các tham số mới.
- Mở, xem và in các file thiết kế mạch dễ dàng với đầy đủ các thông tin linh kiện, netlist, dữ liệu bản vẽ, kích thước, số lượng…
- Hệ thống các thư viện linh kiện phong phú, chi tiết và hoàn chỉnh bao gồm tất cả các linh kiện nhúng, số, tương tự…

- Đặt và sửa đối tượng trên các lớp cơ khí, định nghĩa các luật thiết kế, tùy chỉnh các lớp mạch in, chuyển từ schematic sang PCB, đặt vị trí linh kiện trên PCB.
- Mô phỏng mạch PCB 3D, đem lại hình ảnh mạch điện trung thực trong không gian 3 chiều, hỗ trợ MCAD-ECAD, liên kết trực tiếp với mô hình STEP, kiểm tra khoảng cách cách điện, cấu hình cho cả 2D và 3D.

### 4.1.2 Sơ đồ nguyên lí

Dùng phần mềm mô phỏng Altium thiết kế sơ đồ nguyên lí

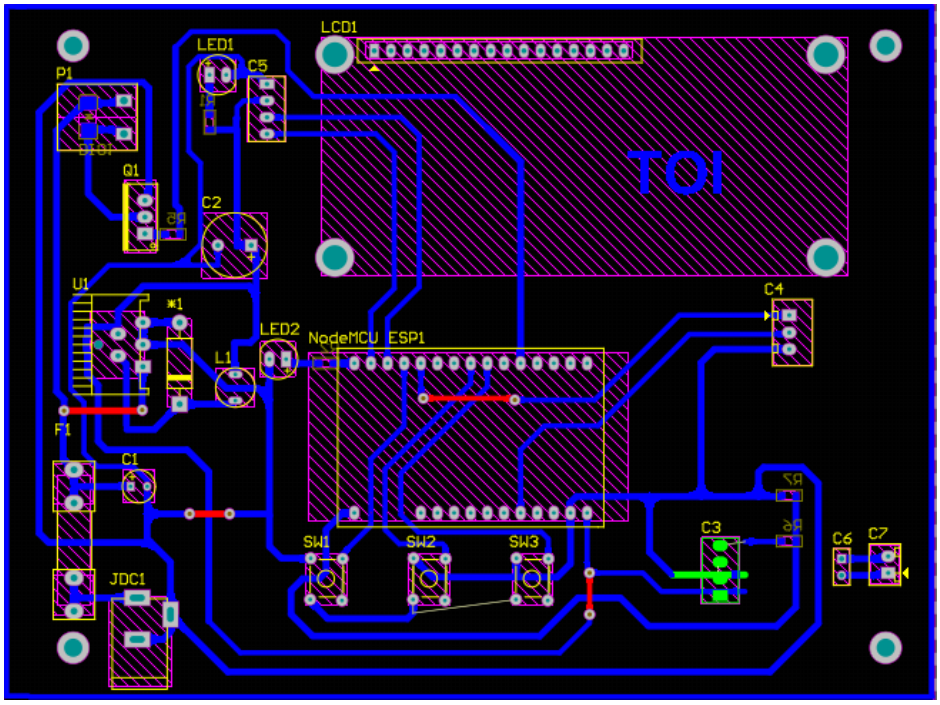



Hình 4-2 : Sơ đồ nguyên lí mạch

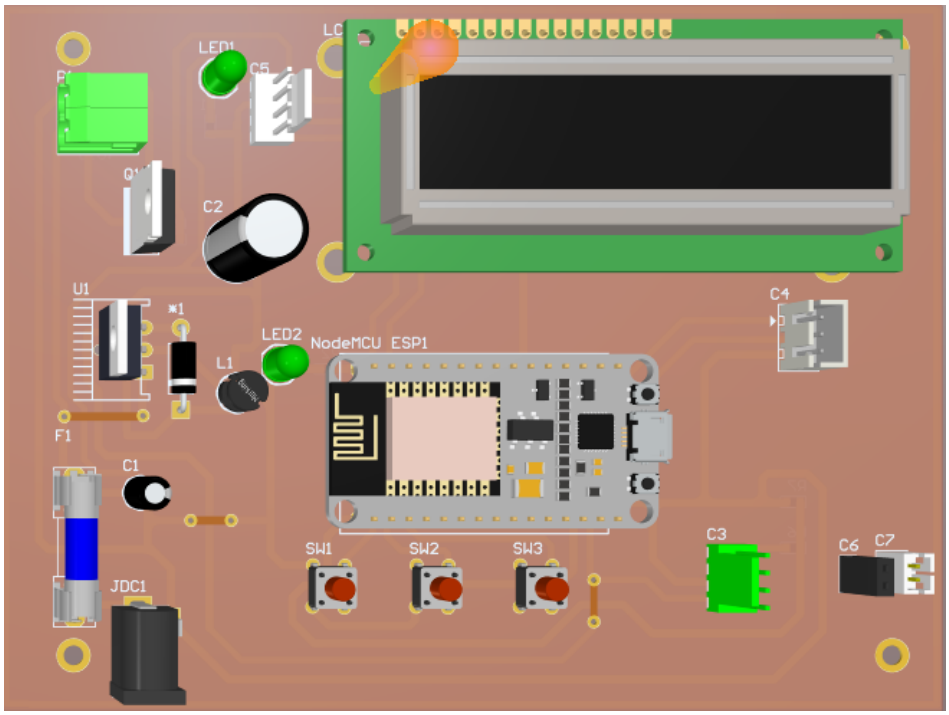
Sơ đồ nguyên lí gồm 4 khối: vi điều khiển, cảm biến, hiển thị, thiết bị (máy bơm). Khi mạch được cấp điện áp 5 VDC, cảm biến và sẽ lấy các giá trị nhiệt độ, độ ẩm đưa về vi điều khiển để xử lí. Sau khi nhận dữ liệu từ các hỗ trợ, vi điều khiển tiến hành so sánh các giá trị nhiệt độ, độ ẩm với các giá trị tương ứng. Sau khi kiểm tra và tính toán các giá trị trên, nếu thỏa mã điều kiện cài đặt, vi điều khiển sẽ phát tín hiệu cho thiết bị (ở đây là máy bơm), khi máy bơm nhận được tín hiệu của vi điều khiển sẽ thực hiện bơm nước tự động trong khoảng thời gian cài đặt. Tất cả các giá trị nhiệt độ, độ ẩm và các bước cài đặt cho hệ thống sẽ được hiển thị trên LCD 16x2.

### 4.1.3 Nối dây cho mạch

Sau khi vẽ sơ đồ nguyên lí cho mạch điện ta tiến hành vẽ sơ đồ nối dây cho mạch




Hình 4-3 : Mặt trước của mạch điện

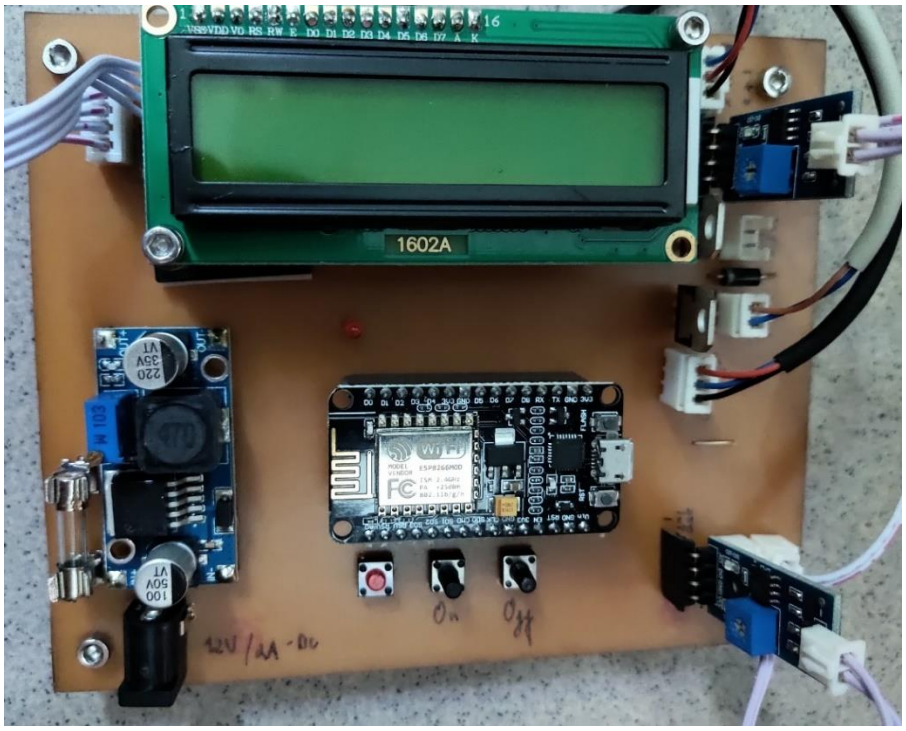



Hình 4-4 : Hình ảnh 3D của mạch điện

## 4.2 Linh kiện cần dùng

| STT | Tên linh kiện | Số lượng |
|---|---|---|
| 1 | NodeMCU ESP8266 | 1 |
| 2 | LCD 16x2 | 1 |
| 3 | Moldule I2C Arduino | 1 |
| 4 | DHT-11 | 1 |
| 5 | Module cảm biến mưa | 1 |
| 6 | Jack DC | 1 |
| 7 | LED | 2 |
| 8 | Nút nhấn | 3 |
| 9 | LM2576 | 1 |
| 10 | Cầu chì | 1 |
| 11 | Transistor Tip122 | 1 |
| 12 | Tụ 1000µF-35V | 1 |

Mạch in sau khi gắn linh kiện :

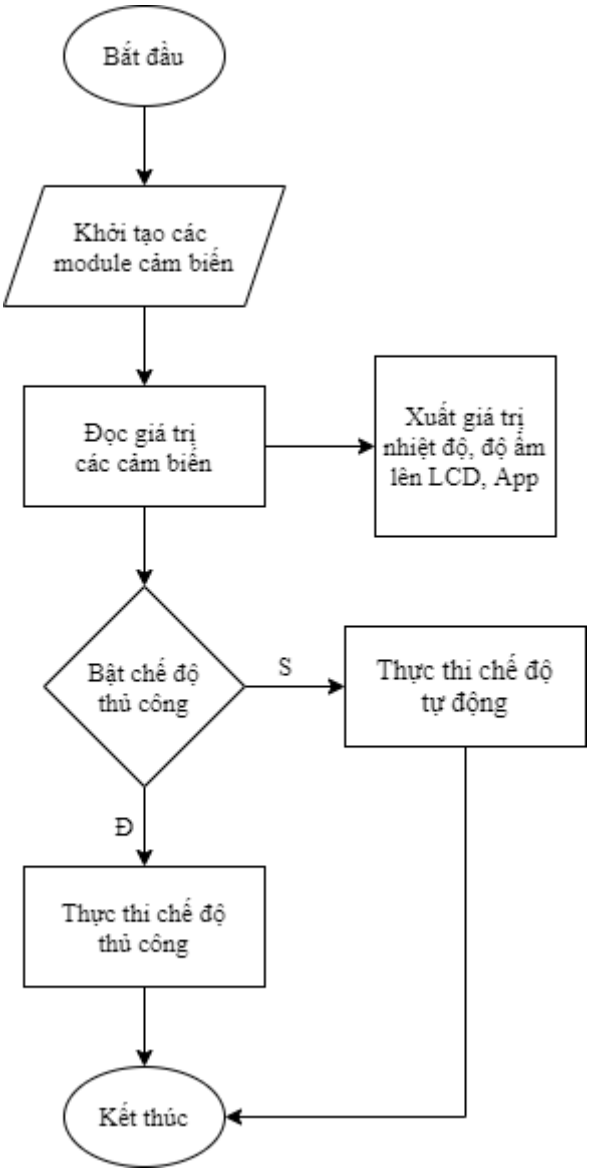



Hình 4-6 : Mạch điện hoàn chỉnh

## 4.3 Lập trình hệ thống

### 4.3.1 Lưu đồ giải thuật

#### 4.3.1.1 Lưu đồ thuật toán chương trình chính

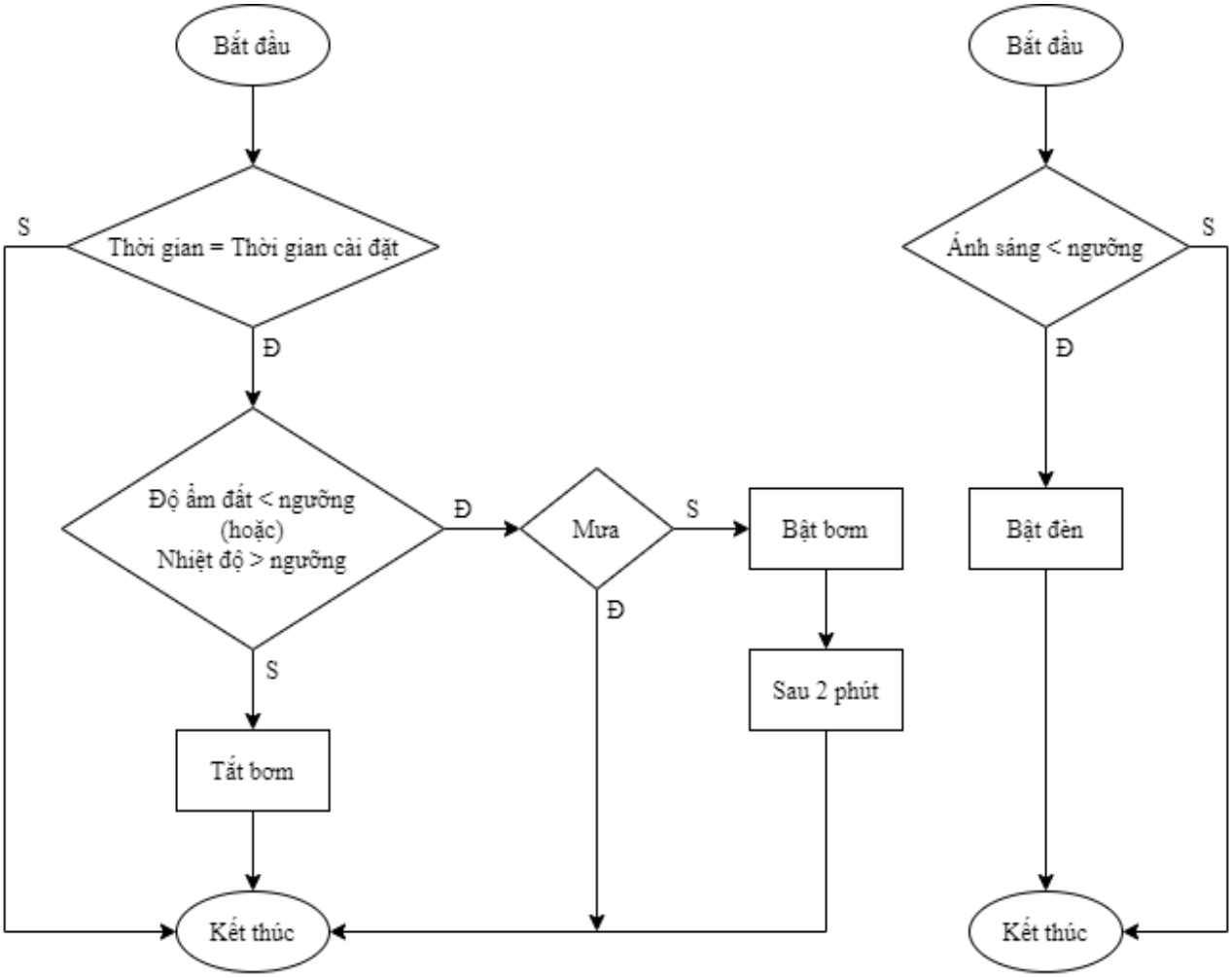



Hình 4-7 : Lưu đồ chương trình chính

**Nguyên lý chương trình :**

Chương trình bắt đầu sẽ khởi tạo các module cảm biến, đọc giá trị cảm biến và xuất giá trị nhiệt độ, độ ẩm lên LCD và App, sau đó thực thi hai chế độ thủ công hoặc chế độ tự động để bật tắt máy bơm.

### 4.3.1.2 Lưu đồ thuật toán chế độ tự động



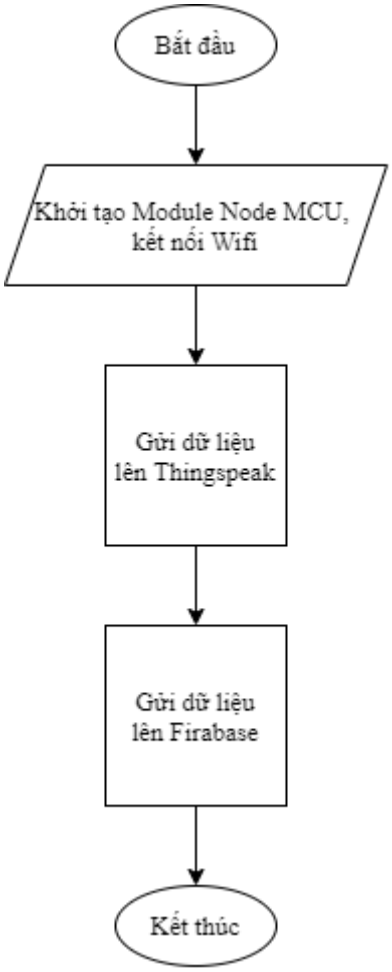

Hình 4-8 : Lưu đồ chương trình ở chế độ tự động

**Nguyên lý hoạt động :**

Bắt đầu kiểm tra thời gian đã cài đặt trên app, đến thời gian hẹn trước, kiểm tra nhiệt độ, độ ẩm đất thông qua cảm biến nhiệt độ và độ ẩm so với ngưỡng đã cài đặt, nếu độ ẩm đất bé hơn mức cài đặt và nhiệt độ lớn hơn mức cài đặt thì tiếp tục kiểm tra xem trời có mưa không thông qua cảm biến mưa, nếu không bật mắt bơm trong 2 phút. Sau 2 phút tắt bơm và kết thúc chương trình.

Song song với đó, kiểm tra ánh sáng có nhỏ hơn mức ngưỡng, nếu nhỏ hơn bật đèn.

#### 4.3.1.3 Lưu đồ gửi dữ liệu




Hình 4-9 : Lưu đồ gửi dữ liệu

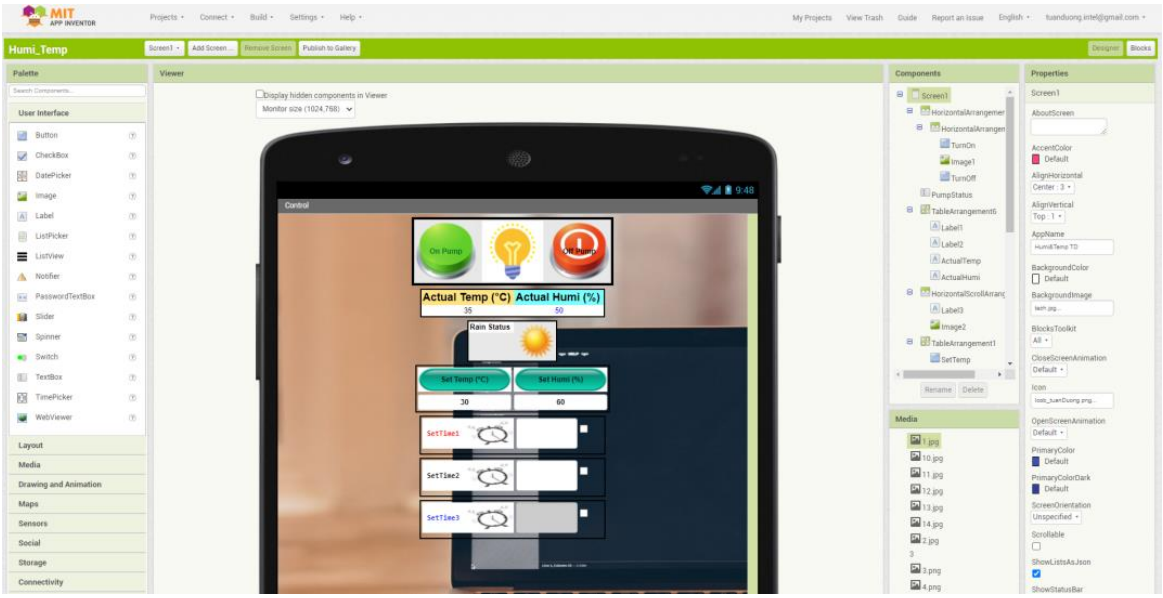
**Nguyên lý hoạt động :**

Sau khi nhận được thông tin dữ liệu từ cảm biến nhiệt độ độ ẩm DHT 11, khởi tạo module Node MCU sẽ kết nối với wifi người dùng và gửi dữ liệu lên Firebase.

### 4.4.2 MIT App Inventor

MIT App Inventor dành cho Android là một ứng dụng web nguồn mở ban đầu được cung cấp bởi Google và hiện tại được duy trì bởi Viện Công nghệ Massachusetts (MIT).

Nền tảng cho phép nhà lập trình tạo ra các ứng dụng phần mềm cho hệ điều hành Android (OS). Bằng cách sử dụng giao diện đồ họa, nền tảng cho phép người dùng kéo và thả các khối mã (blocks) để tạo ra các ứng dụng có thể chạy trên thiết bị Android.




Hình 4-9 : Giao diện MIT App Inventor

Những tính năng có trên MIT App Inventor là:

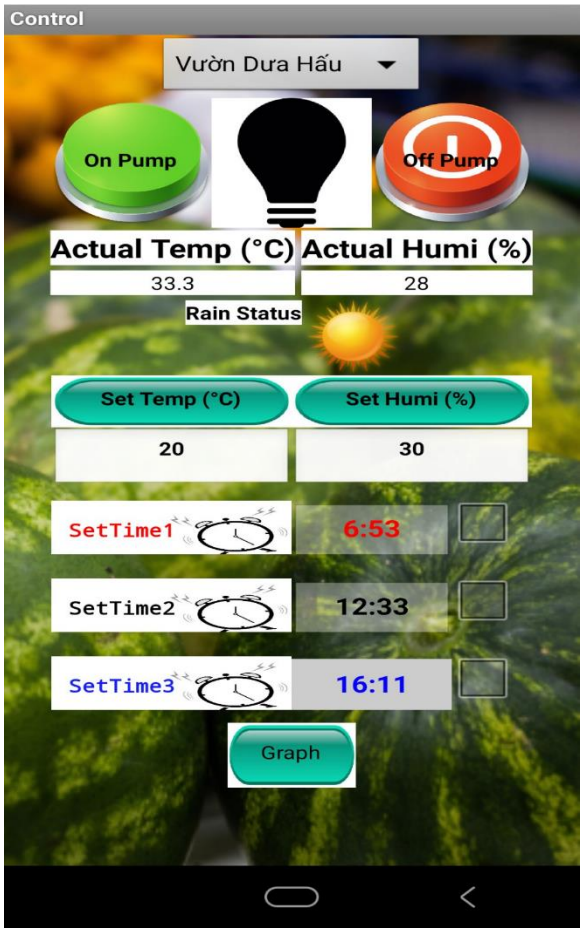
- Cho phép xây dựng nhanh chóng những thành phần cơ bản (components) của một ứng dụng Android: Nút bấm, nút lựa chọn, chọn ngày giờ, ảnh, văn bản, thông báo, kéo trượt, trình duyệt web.
- Sử dụng nhiều tính năng trên điện thoại: Chụp ảnh, quay phim, chọn ảnh, bật video hoặc audio, thu âm, nhận diện giọng nói, chuyển lời thoại thành văn bản, dịch.
- Hỗ trợ xây dựng game bằng các components: Ball, Canvas, ImageSprite.
- Cảm biến: đo gia tốc (AccelerometerSensor), đọc mã vạch, tính giờ, con quay hồi chuyển (gyroscopeSensor), xác định địa điểm (locationSensor), NFC, đo tốc độ (pedometer), đo khoảng cách xa gần với vật thể (proximitySensor).
- Kết nối: Danh bạ, email, gọi điện, chia sẻ thông qua các ứng dụng mạng xã hội khác trên thiết bị, nhắn tin, sử dụng twitter qua API, bật ứng dụng khác, bluetooth, bật trình duyệt.
- Lưu trữ: đọc hoặc lưu tệp txt, csv, sử dụng FusiontablesControl, tạo cơ sở dữ liệu đơn giản trên điện thoại hoặc trên đám mây thông qua server tự tạo hoặc Firebase.
- Điều khiển robot thông qua LegoMindstorms.
- Mua bán trong ứng dụng, Floating button, Báo thức, cảm biến ánh sáng, kết nối dữ liệu SQLite…

Nhược điểm của app Inventor là:

- Lập trình viên chưa thể sử dụng mọi tính năng của Android và việc này phụ thuộc vào khi nào mở rộng mới có tính năng bạn cần có được tạo ra. Khuyết điểm này chỉ có thể khắc phục bằng cách tự xây dựng mở rộng cho App Inventor.
- Do ứng dụng được phát triển trên server của MIT, giới hạn dung lượng của mỗi project chỉ là 5mb.

Mặc dù có những nhược điểm như vậy, nhưng MIT App Iventor vẫn là một nền tảng mạnh mẽ giúp những ai mới bắt đầu lập trình trên Android có thể tạo ra được những ứng dụng hoàn thiện. Những ưu điểm trên của MIT app hoàn toàn phù hợp với đề tài.

Kết quả thu được:




Hình 4-10 : Giao diện chính của phần mềm

Chức năng của phần mềm:

- On pump: Bật máy bơm.
- Off pump: Tắt máy bơm.
- Actual Temp : Nhiệt độ hiện tại.

- Actual Humi: Độ ẩm hiện tại.
- Set temp/humi : Điều kiện nhiệt độ/độ ẩm để máy bơm tự động bật.
- Set time: hẹn giờ bật máy bơm nếu như đạt đủ điều kiện nhiệt độ và độ ẩm hiện tại nhỏ hơn nhiệt độ và độ ẩm cài đặt.
- Rain Status: thời tiết hiện tại

# CHƯƠNG V: KẾT QUẢ

## 5.1 Kết quả thu được

### 5.1.1 Sử dụng module ESP8266 Node MCU

ESP8266 là một mạch vi điều khiển có thể giúp chúng ta điều khiển các thiết bị điện tử. Điều đặc biệt của nó, đó là sự kết hợp của module Wifi tích hợp sẵn bên trong con vi điều khiển chính. Hiện nay, ESP8266 rất được giới nghiên cứu tự động hóa Việt Nam ưa chuộng vì giá thành cực kỳ rẻ (chỉ bằng một con Arduino Nano), nhưng lại được tích hợp sẵn Wifi, bộ nhớ flash 8Mb.

### 5.1.2 Sử dụng cảm biến

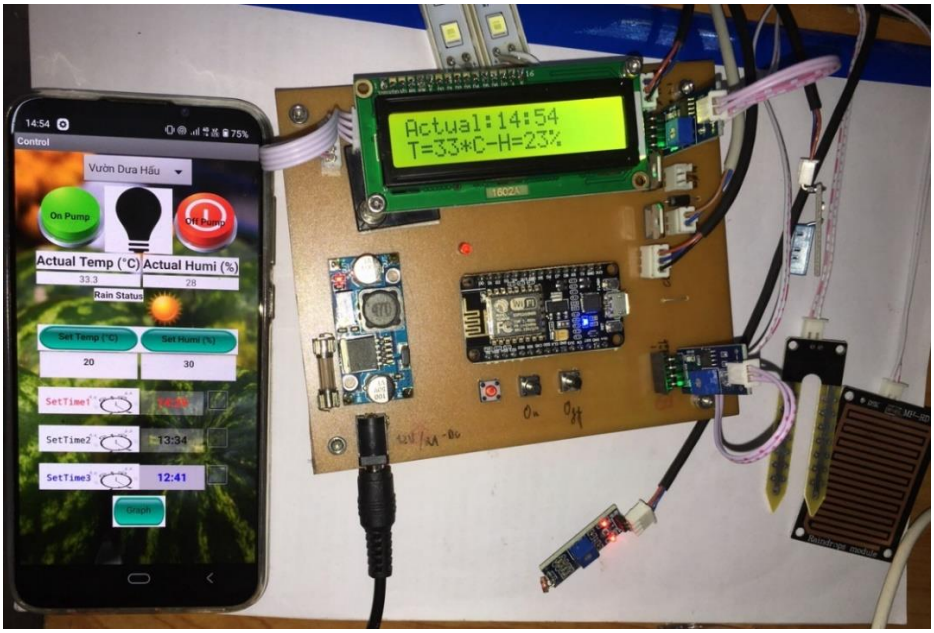
Trong quá quá trình thực hiện đề tài, bản thân đã co thêm được nhiều kiến thức về các loại cảm biến, nguyên lí hoạt động cũng như ứng dụng của nó trong thực tế.

### 5.1.3 Lập trình app MIT App Invetor

Ngoài phương pháp bật/tắt máy bơm thủ công, MIT App Invetor giúp người sử dụng bật/tắt máy bơm từ xa cũng như nắm bắt thông tin nhiệt độ độ ẩm một cách dễ dàng, thuận tiện ngay cả khi không có nhà. Qua đề tài này, bản thân đã có thêm nhiều kiến thức lập trình, cách lấy thời gian hiện tại cũng như cách xử lí dữ liệu.

## 5.2 Kết quả thực nghiệm

Trong quá trình nghiên cứu, thiết kế và thi công, kết quả của đề tài đáp ứng được các yêu cầu đặt ra.




Hình 5-1 : Sản phẩm sau khi hoàn thành

Thông tin dữ liệu sau khi được thu thập từ các cảm biến sẽ được gửi lên màn hình chính của mạch, đồng thời gửi lên app Thông tin gửi lên app gồm có nhiệt độ độ ẩm, trạng thái của

máy bơm. Ngoài ra, phần mềm còn có thế cho phép người dùng cài đặt thời gian tự động bất máy bơm thông qua phần set time. Nếu như nhiệt độ thực tế thấp hơn nhiệt độ cài đặt và độ ẩm thực tế cao hơn độ ẩm cài đặt thì máy bơm sẽ không hoạt động.

# CHƯƠNG VI: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 6.1 Kết luận

Đề tài "Hệ thống tưới cây tự động" với mục đích giảm thiểu công sức con người, tăng năng suất và chất lượng cây trồng. Kết quả thực nghiệm đã đáp ứng được nhưng yêu cầu đề ra. Dưới đây là một số ưu-nhược điểm của hệ thống:

**Ưu điểm:**

- Giải quyết được những yêu cầu đề ra của đề tài
- Hệ thống tiêu thụ ít điện năng
- Dễ dàng quan sát, sử dụng
- Chi phí thấp
- Tiết kiệm thời gian, chi phí nhân công

Tuy nhiên, đây là mô hình nên sự thiếu sót là điều không thể tránh khỏi

**Nhược điểm:**

- Không gian thu thập dữ liệu của cảm biến còn hạn chế
- Mô hình chưa gọn
- Thông tin dữ liệu chưa được ổn định

## 6.2 Hướng phát triển đề tài

Hệ thống cần được chỉnh sửa để hoàn chỉnh hơn, dưới đây là những vấn đề được đề ra nhằm hoàn thiện hệ thống hơn:

- Thiết kế giao diện app sao cho dễ nhìn và dễ tiếp cận người dùng
- Tích hợp nhiều cảm biến để thông tin dữ liệu được chính xác hơn
- Ngoài thiết bị bơm có thể tích hợp thêm hệ thống chiếu sáng và quạt đối với cây trồng trong nhà kính.

# TÀI LIỆU THAM KHẢO

[1] Internet of things với esp8266, www.cachdung.com

[2] https://www.alldatasheet.com

[3] https://www.arduino.cc

[4] https://vi.wikipedia.org

[5] https://firebase.google.com

# PHỤ LỤC

```
#include <FirebaseArduino.h> // Gui du lieu len Firebase
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <ThingSpeak.h> // Gui du lieu led web thingspeak
#include <WiFiUdp.h> // LIB use for realtime
#include <NTPClient.h> // LIB use for realtime
//#include <SHT1x.h> // Sensor SHT10
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "DHT.h"
#define DHTTYPE DHT11
LiquidCrystal_I2C lcd(0x27,16,2);
// SCL -D1, SDA-D2
#define Led_test D0
#define BT1 D3
#define BT2 D5
#define BT3 D6
#define Pump_Machine D8
#define dataPin  D6
#define clockPin D7
#define Rain_Sensor A0
#define rain_value 315
#define dht_dpin 2 // D4


DHT dht(dht_dpin, DHTTYPE);
#define WIFI_SSID "KhoaDien" // WiFi name
#define WIFI_PASSWORD "12345678"  // Password  WiFi

const char *server = "api.thingspeak.com";
unsigned long myChannelNumber = 1236409;
const char * myWriteAPIKey = "NZLCGEGOCP6TD5AB";



//SHT1x sht1x(dataPin, clockPin);
float temp_c,old_temp = 0 ;
float humidity, old_humi = 0 ;
unsigned long time_check = 0,time_send_data = 0 ;
```

```
WiFiClient client;


//Declare
float temp_point = 0 ,humi_point = 0 ;
signed int hrs_point1,minu_point1;
signed int hrs_point2,minu_point2;
signed int hrs_point3,minu_point3;
signed int time_start =0, time_finish = 0;
signed int time_turn_on = 2 ;
signed int check_box1=0,check_box2=0,check_box3=0;

String real_time;
signed int sec,minu,hrs;
long time_process = 0;
int display_time=0,process_control_sendata= 0,process_control = 0 ;
boolean flag_turn_on_pump = 0;
boolean flag_send_humi = 0, flag_send_temp = 0, flag_send_data =
0,old_flag_pump = 0;
boolean flag_follow_condition = 0;
int flag_check_rain_ss = 2,old_flag_rain = 5, flag_check_time = 0 ;
String convert_send;
String data;
unsigned long count_average = 0;
float AverageHumi_Day = 0 ,AverageHumi_Month = 0 ,AverageTemp_Day = 0
,AverageTemp_Month = 0 ;
unsigned long SumHumi_Day = 0, SumHumi_Month = 0, SumTemp_Day =0,
SumTemp_Month = 0;
boolean flag_send_average = 0 ;
int process_control_send_average = 0;

WiFiUDP u;
NTPClient n(u,"3.vn.pool.ntp.org",7*3600);

void blink_led(byte number);
void display_actual();
void display_setpoint();
void convert_time_h_m(String get_time,int *_hour,int *_minute);
void get_real_time(int _timeout,int *_hour,int *_minute,int *_sec);
void get_firebase();
void control_pump();
```

```
void check_conditions_on_pump();
void check_conditions_off_pump();
void check_rain_sensor();
void calculator_average_data();
void ini_process();
void auto_process();
void manual_process();

void setup() {
  Serial.begin(9600);
  pinMode(Led_test, OUTPUT);
  pinMode(Pump_Machine, OUTPUT);
  pinMode(BT1, INPUT_PULLUP);
  pinMode(BT2, INPUT_PULLUP);
  pinMode(BT3, INPUT_PULLUP);

  lcd.begin();
  lcd.backlight();
  lcd.print("Hello    ");
  dht.begin();
  if (digitalRead(BT3)== 0)
  {
        process_control = 5; // go to Manual Mode
        digitalWrite(Led_test,1);
  }
  else
  {
          // connect to wifi.
   WiFi.begin(WIFI_SSID, 12345678);
  Serial.print("connecting");
  while (WiFi.status() != Ket_noi_thanh_cong)
            {
                  lcd.setCursor(0,1);
                  lcd.print("Connecting Wifi       ");
                  Serial.print(".");
                  delay(500);
                  process_control++;
                  if (process_control == 10)
                  {
                        process_control = 5;
                        break;
```

```
                  }
            }
  if ( process_control <10)
  {
            process_control = 0;
            lcd.setCursor(0,1);
            lcd.print("Connected           ");
            Serial.println();
            Serial.print("connected: ");
            Serial.println(WiFi.localIP());
            Firebase.begin("https://humi-sensor-ad-default-
rtdb.firebaseio.com/
BU4FDhRsyYLISkjxvdhUGoj38uM56ekmgiXDOhZ8");//dia chi firebase
            Firebase.stream("/GetData");
            ThingSpeak.begin(client);
            n.begin();
            n.update();
            real_time= n.getFormattedTime();
            blink_led(4);
  }
  else
  {
      digitalWrite(Led_test,1);
      lcd.clear();
      process_control = 5;
  }
  }
}


void loop()
{

      switch (process_control)
      {
            case 0:
                  ini_process();
                  break;
            case 4:
                  auto_process();
                  break;
```

```
            case 5:
                  manual_process();
                  break;
      }
}


void blink_led(byte number)
{
  for (int i = 0 ; i<number;i++ )
  {
    digitalWrite(Led_test,1);
    delay(200);
    digitalWrite(Led_test,0);
    delay(200);
  }
}
void display_actual()
{
  lcd.setCursor(0,0);
  lcd.print("Actual:");
  lcd.print(hrs);
  lcd.print(":");
  lcd.print(minu);
  lcd.print("        ");

  lcd.setCursor(0,1);
  //lcd.print(round(temp_c));
  lcd.print("T=");
  lcd.print((int)temp_c);
  lcd.print("*C-");
  lcd.print("H=");
  //lcd.print(round(humidity));
  lcd.print((int)humidity);
  lcd.print("%      ");
}

void display_setpoint()
{
      if (check_box1 == 1)
      {
```

```
        lcd.setCursor(0,0);
        lcd.print("SetTime:");
        lcd.print(hrs_point1);
        lcd.print(":");
        lcd.print(minu_point1);
        lcd.print("     ");

}
else
{
        if (check_box2 == 1)
        {
                lcd.setCursor(0,0);
                lcd.print("SetTime:");
                lcd.print(hrs_point2);
                lcd.print(":");
                lcd.print(minu_point2);
                lcd.print("     ");

        }
        else
        {
                if (check_box3 == 1)
                {
                        lcd.setCursor(0,0);
                        lcd.print("SetTime:");
                        lcd.print(hrs_point3);
                        lcd.print(":");
                        lcd.print(minu_point3);
                        lcd.print("     ");

                }
                else
                {
                        lcd.setCursor(0,0);
                        lcd.print("!! Not set Time       ");

                }
        }
}
```

```
      lcd.setCursor(0,1);
      lcd.print("Set:T=");
      lcd.print((int)temp_point);
      lcd.print("*C-");
      lcd.print("H=");
      lcd.print((int)humi_point);
      lcd.print("%      ");
}

void convert_time_h_m(String get_time,int *_hour,int *_minute)
{
      byte moc[3];byte count_moc=0;
      String chuoi1, chuoi2;
      get_time.remove(0,1);
      get_time.remove((get_time.length()-1),1);
//Serial.print(get_time );
    //Data is string, type: 12:20= => Convert to int
    for (int i = 0 ; i<get_time.length();i++)
  {
    if (get_time.charAt(i) == ':')
    {
      moc[count_moc] = i;
      count_moc++;
    }
  }
  chuoi1 = get_time;
  chuoi2 = get_time;
  chuoi1.remove(moc[0]); // Ham remove :
http://arduino.vn/reference/library/string/1/huong-dan-ham/remove
  chuoi2.remove(0,moc[0]+1);
  *_hour = chuoi1.toInt();
  *_minute = chuoi2.toInt();
  //Serial.print(*_hour );Serial.print("-" );Serial.println(*_minute
);
}
void get_real_time(int _timeout,int *_hour,int *_minute,int *_sec)
{
   if (millis()- time_process > _timeout)
  {
      time_process = millis();
      n.update();
```

```
      real_time= n.getFormattedTime();  // Get time from Internet
      byte moc[3];byte count_moc=0;
      String chuoi1, chuoi2,chuoi3;

      //Data is string, type: 12:20:35 => Convert to int
      for (int i = 0 ; i< real_time.length();i++)
      {
            if (real_time.charAt(i) == ':')
            {
                  moc[count_moc] = i;
                  count_moc++;
            }
      }
      chuoi1 = real_time;
      chuoi2 = real_time;
      chuoi3 = real_time;
      chuoi1.remove(moc[0]);
      chuoi2.remove(0,moc[0]+1);
      chuoi3.remove(0,moc[1]+1);
      *_hour = chuoi1.toInt();
      *_minute = chuoi2.toInt();
      *_sec = chuoi3.toInt();
  }
}
void get_firebase() // Lấy dữ liệu từ Firebase
{
  if (Firebase.available()) // Nếu fire có sự thay đổi dữ liệu

  {
    FirebaseObject event = Firebase.readEvent();
    String eventType = event.getString("type");
    eventType.toLowerCase();
    if (eventType == "put")
      {
        String path = event.getString("path");
        data = event.getString("data");
        if (path == "/pump")
        {
          if (data == "0")
          {
                  flag_turn_on_pump = 0 ;
```

```
        // Serial.println("Da nhan du lieu tu FireBase - OFF May
Bom");
          }
          if (data == "1")
          {
                   flag_turn_on_pump = 1;
        //  Serial.println("Da nhan du lieu tu FireBase - ON May
Bom");
          }
        }
            if (path == "/SetHumi")
        {
          data.remove(0,1);
          data.remove((data.length()-1),1);
              humi_point =  data.toFloat();
        }
            if (path == "/SetTemp")
        {
          data.remove(0,1);
          data.remove((data.length()-1),1);
              temp_point =  data.toFloat();
        }
            if (path == "/SetTime1")
        {
                convert_time_h_m(data,&hrs_point1,&minu_point1);
        }
            if (path == "/SetTime2")
        {
                convert_time_h_m(data,&hrs_point2,&minu_point2);
        }
            if (path == "/SetTime3")
        {
                convert_time_h_m(data,&hrs_point3,&minu_point3);
        }

            if (path == "/CheckBox1")
        {
                data.remove(0,1);
                data.remove((data.length()-1),1);
                check_box1 = data.toInt();
        }
```

```
            if (path == "/CheckBox2")
        {
                  data.remove(0,1);
                  data.remove((data.length()-1),1);
                  check_box2 = data.toInt();
        }
            if (path == "/CheckBox3")
        {
                  data.remove(0,1);
                  data.remove((data.length()-1),1);
                  check_box3 = data.toInt();
        }


      }
  }
}
void control_pump()
{
      if (old_flag_pump == flag_turn_on_pump)
      {}
      else
      {
            if (flag_check_rain_ss ==1)
            {
                  flag_turn_on_pump =0; // Neu dang mua, thi luon
luon tat may bom
            }
            old_flag_pump = flag_turn_on_pump;
            convert_send = String(old_flag_pump);
      //    Serial.println("Bat May Bom");
            Firebase.setString("/PumpStatus", convert_send);
            delay(50);
            Firebase.setString("/PumpStatus", convert_send);
            delay(50);
            Firebase.setString("/PumpStatus", convert_send);
            if (old_flag_pump == 1 && flag_check_rain_ss ==0)
            {
                  digitalWrite(Pump_Machine,HIGH);
                  digitalWrite(Led_test,HIGH);
            }
```

```
            if (old_flag_pump == 0)
            {
                  digitalWrite(Pump_Machine,LOW);
                  digitalWrite(Led_test,LOW);
            }
      }

}
void check_conditions_on_pump()
{
      if (check_box1 == 1 )
      {
            if (hrs == hrs_point1 && minu == minu_point1 && sec <10)
            {
                  flag_check_time = 1;
            }
      }
      if (check_box2 == 1 )
      {
            if (hrs == hrs_point2 && minu == minu_point2 && sec <10)
            {
                  flag_check_time = 1;
            }
      }
      if (check_box3 == 1 )
      {
            if (hrs == hrs_point3 && minu == minu_point3 && sec <10)
            {
                  flag_check_time = 1;
            }
      }

      if (flag_check_time == 1)
      {
            if (temp_c >temp_point || humidity < humi_point)
            {
                  flag_turn_on_pump = 1;
                  time_start = minu;
                  time_finish = time_start + time_turn_on;
```

```
                  if (time_finish >=60)time_finish = time_finish -
60;
                  flag_follow_condition = 1;
            }
            flag_check_time = 0;
      }

}
void check_conditions_off_pump()
{
      if (flag_follow_condition == 1)
      {
            if(minu == time_finish)
            {
                  flag_turn_on_pump = 0;
                  flag_follow_condition = 0;
            }
      }
}

void check_rain_sensor()
{
  Serial.print("Rain
Sensor");Serial.println(analogRead(Rain_Sensor));
      if(analogRead(Rain_Sensor)< rain_value) // Rain On
      {
            flag_check_rain_ss = 1;
            flag_turn_on_pump =0;
  // Serial.print("Troi Mua");
      }
      else
      {
            flag_check_rain_ss =0;
      }
      if (flag_check_rain_ss != old_flag_rain)
      {

      //    Serial.print("Rain
Sensor");Serial.println(analogRead(Rain_Sensor));
            old_flag_rain = flag_check_rain_ss;
            convert_send = String(old_flag_rain);
```

```
            Firebase.setString("/RainSensor1", convert_send);

      }
}
void calculator_average_data()
{
      SumTemp_Day = SumTemp_Day + temp_c;
      SumHumi_Day = SumHumi_Day + humidity;
      count_average ++;
      //if (hrs == 23 && minu == 55 && flag_send_average == 0 && sec
< 25)
      if (hrs == hrs_point3 && minu == minu_point3 &&
flag_send_average == 0 && sec < 30)
      {
            flag_send_average = 1;
            AverageTemp_Day =(float) SumTemp_Day / count_average;
            AverageHumi_Day =(float) SumHumi_Day / count_average;
      }
      if (flag_send_average == 1)
      {
            process_control_send_average++;
            switch(process_control_send_average)
                  {
                   case 1:

      //ThingSpeak.writeField(myChannelNumber, 3, AverageTemp_Day,
myWriteAPIKey);
                              break;
                   case 2:
                              Firebase.setFloat("/AverageTemp_Day",
AverageTemp_Day);
                              break;
                   case 3:
                       //     ThingSpeak.writeField(myChannelNumber,
4, AverageHumi_Day, myWriteAPIKey);
                               break;
                   case 4:
                              Firebase.setFloat("/AverageHumi_Day",
AverageHumi_Day);
                              break;
                  }
```

```
            if(process_control_send_average == 4)
            {
                  process_control_send_average=0;
                  flag_send_average = 0;
                  SumTemp_Day = 0 ;
                  SumHumi_Day = 0 ;
                  count_average = 0 ;
            }
      }

}
void ini_process() //0
{
      FirebaseObject fbo = Firebase.get("/GetData");
      lcd.setCursor(0,0);
      lcd.print("Waitting             ");
      lcd.setCursor(0,1);
      lcd.print("Ini Process ....     ");

      data = fbo.getString("CheckBox1");
      data.remove(0,1);
      data.remove((data.length()-1),1);
      check_box1 = data.toInt();
//    Serial.println(check_box1);
      delay(200);
      data = fbo.getString("CheckBox2");
      data.remove(0,1);
      data.remove((data.length()-1),1);
      check_box2 = data.toInt();
//    Serial.println(check_box2);
      delay(200);
      data = fbo.getString("CheckBox3");
      data.remove(0,1);
      data.remove((data.length()-1),1);
      check_box3 = data.toInt();
//    Serial.println(check_box3);

      delay(200);
      data = fbo.getString("SetHumi");
      data.remove(0,1);
      data.remove((data.length()-1),1);
```

```
      humi_point =  data.toFloat();
//    Serial.println(humi_point);
      delay(200);
      data = fbo.getString("SetTemp");
      data.remove(0,1);
      data.remove((data.length()-1),1);
      temp_point =  data.toFloat();
//    Serial.println(temp_point);
      delay(200);

  data = fbo.getString("pump");
  if (data == "0")
    {
    flag_turn_on_pump = 0 ;
    }
    if (data == "1")
    {
    flag_turn_on_pump = 1;
    }
    delay(200);
      data = fbo.getString("SetTime1");
      convert_time_h_m(data,&hrs_point1,&minu_point1);
      delay(200);
      data = fbo.getString("SetTime2");
      convert_time_h_m(data,&hrs_point2,&minu_point2);
      delay(200);
      data = fbo.getString("SetTime3");
      convert_time_h_m(data,&hrs_point3,&minu_point3);

      process_control = 4;

}
void auto_process() // 4
{
    get_real_time(10000,&hrs, &minu,&sec);
      if (millis() - time_check >8000)
      {
        time_check = millis();
      //  temp_c = sht1x.readTemperatureC();
        //humidity = sht1x.readHumidity();
    temp_c = dht.readTemperature();//sht1x.readTemperatureC();
```

```
    humidity = dht.readHumidity();//sht1x.readHumidity();
        check_rain_sensor();
    if((temp_c<100) && (temp_c > -5) )
    {
    if (   (abs(temp_c-old_temp) > 0.5)    ||     (abs( humidity -
old_humi) > 1.5)    )
    {
      flag_send_data = 1;
      old_temp = temp_c;
      old_humi = humidity;
    }
        if (flag_send_data == 1 )
      {
       process_control_sendata++;
      switch(process_control_sendata)
         {
          case 1:
           // ThingSpeak.writeField(myChannelNumber, 2, humidity,
myWriteAPIKey);
            break;
          case 2:
            Firebase.setFloat("/ActualHumi1", humidity);
            break;
          case 3:
         //   ThingSpeak.writeField(myChannelNumber, 1, temp_c,
myWriteAPIKey);
             break;
          case 4:
            Firebase.setFloat("/ActualTemp1", temp_c);
            break;
         }
      if(process_control_sendata == 4)
      {
        process_control_sendata=0;
        flag_send_data = 0;
      }
      }
      calculator_average_data(); // Calculate and Send
    }

      }
```

```
    get_firebase();

      // Hiển thị LCD ////
    display_time++;
    if (display_time<60) display_actual();
    if (display_time>=60)
    {
      display_setpoint();
      if (display_time>=120)display_time=0;
    }
      ////////////////////


      if (digitalRead(BT2) == 0)
      {
            flag_turn_on_pump = 1 ;
            Firebase.setString("/GetData/pump", "1");

      }
      if (digitalRead(BT3) == 0 )
      {
            flag_turn_on_pump = 0 ;
            Firebase.setString("/GetData/pump", "0");
      }
      check_conditions_on_pump();
      check_conditions_off_pump();
      control_pump();
}

void manual_process()
{
  lcd.setCursor(0,0);
  lcd.print("Manual Mode");
      if (millis() - time_check >2000)
      {
            time_check = millis();
            temp_c =
dht.readTemperature();//sht1x.readTemperatureC();
            humidity = dht.readHumidity();//sht1x.readHumidity();
            lcd.setCursor(0,1);
            lcd.print((int)temp_c);
```

```
            lcd.print("*C-");
            lcd.print("H:");
            lcd.print((int)humidity);
            lcd.print("%      ");
            digitalWrite(Led_test,flag_turn_on_pump);
            flag_turn_on_pump = !flag_turn_on_pump;

        }
  if (digitalRead(BT2) == 0)
  {
        digitalWrite(Pump_Machine,HIGH);
  }
  if (digitalRead(BT3) == 0 )
  {
        digitalWrite(Pump_Machine,LOW);
  }
}
```