

TRƯỜNG ĐH CÔNG NGHIỆP THỰC PHẨM TP. HCM

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐỒ ÁN CHUYÊN NGÀNH



NGUYỄN VĂN THÀNH

ĐỒ ÁN CHUYÊN NGÀNH

ĐIỀU KHIỂN THIẾT BỊ TỪ XA SỬ DỤNG

MODULE SIM800L

GVHD: Đoàn Xuân Nam

TP. HỒ CHÍ MINH, tháng 12 năm 2019

TRƯỜNG ĐH CÔNG NGHIỆP THỰC PHẨM
TP. HCM

KHOA ĐIỆN – ĐIỆN TỬ

BỘ MÔN: ĐỒ ÁN CHUYÊN NGÀNH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA
VIỆT NAM

Độc lập - Tự do - Hạnh phúc

TP. HCM, ngày 19 tháng 12 năm 2019

**NHẬN XÉT ĐỒ ÁN CHUYÊN NGÀNH
CỦA GIẢNG VIÊN HƯỚNG DẪN**

Tên mô hình:

ĐIỀU KHIỂN THIẾT BỊ TỪ XA SỬ DỤNG

MODULE SIM800L

Sinh viên thực hiện:

Nguyễn Văn Thành

2202180040

Giảng viên hướng dẫn:

Đoàn Xuân Nam

Đánh giá báo cáo

1. Về cuốn báo cáo:

Số trang _____

Số chương _____

Số bảng số liệu _____

Số hình vẽ _____

Số tài liệu tham khảo _____

Sản phẩm _____

Một số nhận xét về hình thức cuốn báo cáo:

2. Về đồ án thực tế:

3. Về tính ứng dụng thực tế:

4. Về thái độ làm việc của sinh viên:

Đánh giá chung:

Điểm từng sinh viên:

Nguyễn Văn Thành:...../10

Người nhận xét

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

TP. Hồ Chí Minh, ngày 19 tháng 12 năm 2019

Người thực hiện

(Họ tên sinh viên)

TRƯỜNG ĐH CÔNG NGHIỆP THỰC PHẨM
TP. HCM
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN: Đồ Án Chuyên Ngành

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

TP. HCM, ngày 19 tháng 12 năm 2019

ĐỀ CƯƠNG CHI TIẾT

TÊN ĐỒ ÁN: ĐIỀU KHIỂN THIẾT BỊ TỪ XA SỬ DỤNG MODULE SIM800L	
Giảng viên hướng dẫn: Đoàn Xuân Nam	
Thời gian thực hiện: Từ ngày 14/10/2019 đến ngày 19/12/2019	
Sinh viên thực hiện: Nguyễn Văn Thành	
Nội dung đề tài: <ul style="list-style-type: none">- Nguyên cứu hoạt động của hệ thống điều khiển thiết bị từ xa bằng điện thoại di động sử dụng Arduino và module Sim 800L.- Cách gửi và nhận thông tin qua giao tiếp UART- Cách điều khiển module relay và hiển thị LCD 16x2 qua giao tiếp I2C	
Xác nhận của giảng viên hướng dẫn	TP. HCM, ngày 19 tháng 12 năm 2019 Sinh viên

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI.....	1
1.1 Đặt vấn đề.....	1
1.2 Mục tiêu đề tài.....	1
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT.....	2
2.1: Sơ đồ khối và nhiệm vụ các khối.....	2
2.1.1 Sơ đồ khối.....	2
2.2.2 Nhiệm vụ các khối.....	3
2.2 Các thành phần trong đề tài.....	3
2.2.1 Giới thiệu chung.....	3
2.2.2 Cấu tạo và chức năng.....	6
CHƯƠNG 3: CƠ SỞ THỰC HIỆN.....	15
3.1 Sơ đồ khối và thiết kế mô hình.....	15
3.2 Nguyên lý hoạt động.....	25
CHƯƠNG 4 : KẾT QUẢ THỰC NGHIỆM.....	26
4.1 Mô hình mô phỏng.....	26
CHƯƠNG 5: KẾT LUẬN VÀ ĐỊNH HƯỚNG ĐỀ TÀI.....	28
5.1 Kết quả đạt được.....	28
5.2 Hạn chế.....	28
5.3 Hướng phát triển đề tài.....	29
PHỤ LỤC.....	30
Code chương trình.....	
Giới thiệu phần mềm.....	
Hướng dẫn cài đặt.....	

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1 Đặt vấn đề

Trong sự phát triển của các chuẩn điều khiển không dây như RF, hồng ngoại, Wifi, bluetooth... thì sử dụng GSM vẫn là lựa chọn tuyệt vời cho việc điều khiển thiết bị từ xa hàng nghìn kilomet thông qua dụng dụng sim từ các nhà mạng trong nước hay nước ngoài như mobiphone, viettle, vinaphone...

Việc sử dụng thiết bị điều khiển là chiếc điện thoại sử dụng hàng ngày sẽ trở nên vô cùng tiện lợi và hiệu quả. Có thể điều khiển ở bất cứ nơi đâu có phủ sóng của các nhà mạng.

1.1.1 Tầm quan trọng của hệ thống điều khiển qua SMS

Ưu điểm của thiết bị là điều khiển ở bất cứ nơi đâu có hệ thống mạng. Vì vậy để giải quyết vấn đề điều khiển thiết bị với khoảng cách rất xa với tính ổn định cao thì sử dụng module sim 800L là một sự lựa chọn hoàn hảo.

1.1.2 Mục đích nghiên cứu

Do thực tiễn hiện nay trong đời sống sinh hoạt của con người có một số thiết bị đòi hỏi việc điều khiển ở rất xa và cần tính ổn định cao nên việc nghiên cứu một loại giao tính đơn giản, hiệu quả và đặc biệt tiện lợi là một việc hết sức quan trọng. Giúp xóa bỏ những rào cản về khoảng cách cũng như phù hợp với các thiết bị nông nghiệp đang phát triển như hiện nay.

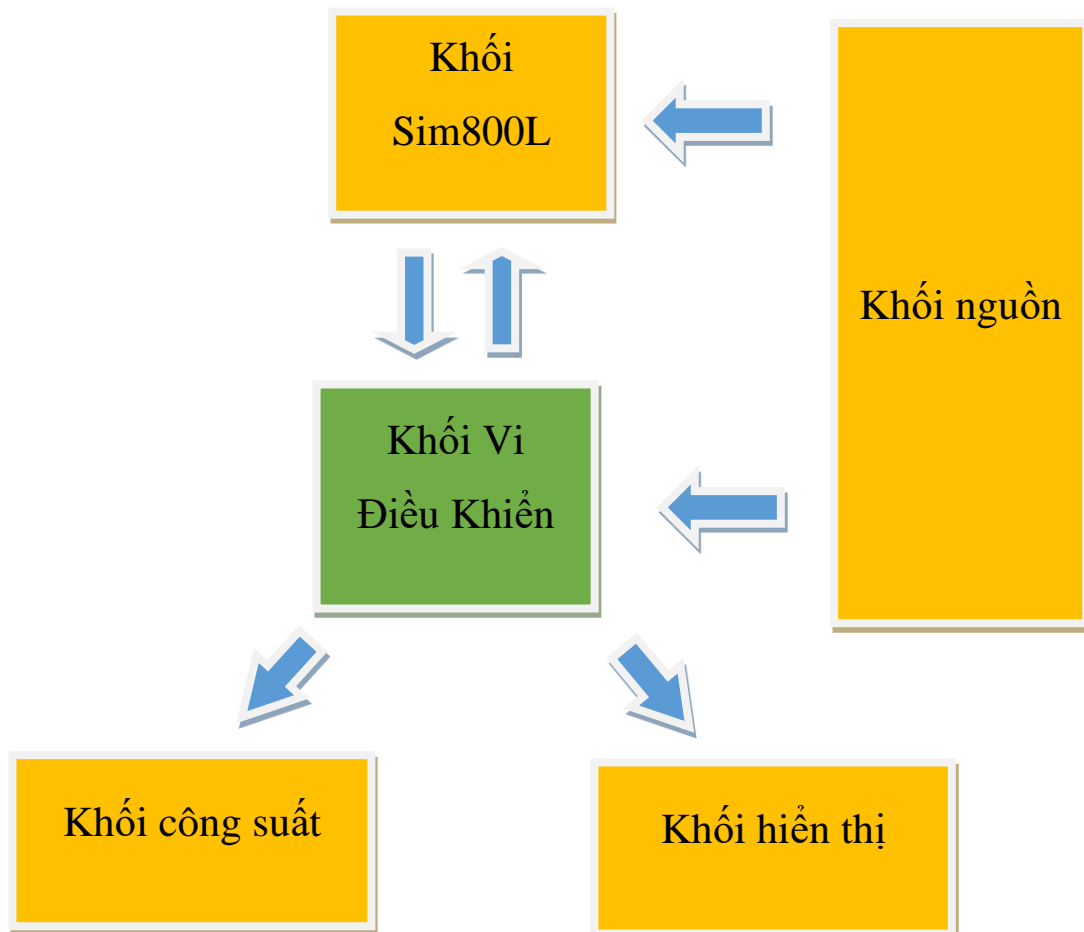
1.2 Mục tiêu đề tài

- Nguyên lý hoạt động và cách lập trình arduino.
- Tính ứng dụng dự án vào thực tế.
- Tìm hiểu về cách sử dụng module Sim800L và module Relay.
- Cách điều khiển và hiển thị màn hình LCD 16x2

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Sơ đồ khối và nhiệm vụ các khối

2.1.1 Sơ đồ khối





Hình 1 : Sơ đồ khối


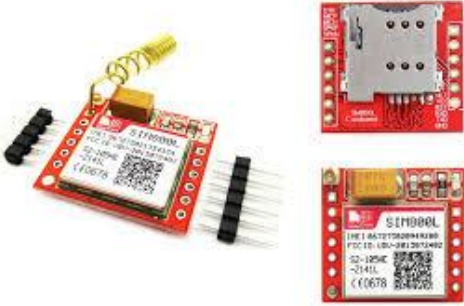

2.1.2 Nhiệm vụ các khối



- Khối nguồn: Cung cấp năng lượng cho toàn hệ thống bao gồm điện áp. 12V, 5V sử dụng module nguồn DC LM2596 cho dòng tải lên đến 3A.
- Khối Sim800L: giao tiếp UART để gửi và nhận tín hiệu tới bộ xử lý trung tâm Arduino.
- Khối vi điều khiển: sử dụng board arduino Nano tiện dụng với nhiều IO.
- Khối công suất: sử dụng module relay có opto cách li an toàn và hoạt động ở dải điện áp lớn.
- Khối hiển thị: sử dụng LCD 16x2 hiển thị các thông tin cơ bản và tương tác với hệ thống qua chuẩn giao tiếp I2C.

2.2 Các thành phần trong đề tài

2.2.1 Giới thiệu chung

STT	Tên Linh Kiện	Chức Năng	Hình Ảnh
1	Arduino Nano	Board xử lý trung tâm	
2	LCD 16x2	Hiển thị dữ liệu 16 cột, 2 hàng	

3	Module I2C LCD	Giúp LCD giao tiếp I2C với Arduino	
4	Module Sim800L	Gửi và nhận dữ liệu tới Arduino	
5	Module relay 2 kênh	Điều khiển công suất cho các thiết bị	

6	Module LM2596	Ổn áp điện áp 4.2V cho Sim800L	
7	Adapter 12V	Cung cấp nguồn cho hệ thống	

2.2.2 Cấu tạo và chức năng

❖ *Arduino*

- Lịch sử phát triển Arduino



Hình 2 : Arduino thực tế

Arduino là một board mạch vi xử lý, nhằm xây dựng các ứng dụng tương tác với nhau hoặc với môi trường được thuận lợi hơn. Phần cứng bao gồm một board mạch nguồn mở được thiết kế trên nền tảng vi xử lý AVR Atmel 8bit, hoặc ARM Atmel 32-bit. Những Model hiện tại được trang bị gồm 1 cổng giao tiếp USB, 6 chân đầu vào analog, 14 chân I/O kỹ thuật số tương thích với nhiều board mở rộng khác nhau.

Được giới thiệu vào năm 2005, Những nhà thiết kế của Arduino cố gắng mang đến một phương thức dễ dàng, không tốn kém cho những người yêu thích, sinh viên và giới chuyên nghiệp để tạo ra những thiết bị có khả năng tương tác với môi trường thông qua các cảm biến và các cơ cấu chấp hành. Những ví dụ phổ biến cho những người yêu thích mới bắt đầu bao gồm các robot đơn giản, điều khiển nhiệt độ và phát hiện chuyển động. Đi cùng với nó là một môi trường phát triển tích hợp (IDE) chạy trên các máy tính cá nhân thông thường và cho phép người dùng viết các chương trình cho Arduino bằng ngôn ngữ C hoặc C++.

Arduino được khởi động vào năm 2005 như là một dự án dành cho sinh viên trại Interaction Design Institute Ivrea (Viện thiết kế tương tác Ivrea) tại Ivrea, Italy. Vào thời điểm đó các sinh viên sử dụng một "BASIC Stamp" (con tem Cơ Bản) có giá khoảng \$100, xem như giá dành cho sinh viên. Massimo Banzi, một trong những người sáng lập, giảng dạy tại Ivrea. Cái tên "Arduino" đến từ một quán bar tại Ivrea, nơi một vài nhà sáng lập của dự án này thường xuyên gặp mặt. Bản thân quán bar này có được lấy tên là Arduino, Bá tước của Ivrea, và là vua của Italy từ năm 1002 đến 1014. Lý thuyết phần cứng được đóng góp bởi một sinh viên người Colombia tên là Hernando Barragan.

Sau khi nền tảng Wiring hoàn thành, các nhà nghiên cứu đã làm việc với nhau để giúp nó nhẹ hơn, rẻ hơn, và khả dụng đối với cộng đồng mã nguồn mở. Trường này cuối cùng bị đóng cửa, vì vậy các nhà nghiên cứu, một trong số đó là David Cuarlierles, đã phổ biến ý tưởng này.

- Arduino Nano



Arduino Uno sử dụng vi điều khiển họ 8bit AVR là ATmega328. Bộ não này có thể xử lý những tác vụ đơn giản như điều khiển đèn LED nhấp nháy, xử lý tín hiệu cho xe điều khiển từ xa, làm một trạm đo nhiệt độ - độ ẩm và hiển thị lên màn hình LCD,... Arduino Nano SMD được trang bị chip ATmega328P và chip nạp CH340.

Bảng Thông số kỹ thuật

Vi điều khiển	ATmega328P
IC nạp và giao tiếp UART	CH340
Điện áp hoạt động	5V – DC
Điện áp đầu vào khuyến dùng	7-12V – DC
Điện áp đầu vào giới hạn	6-20V – DC
Số chân Digital I/O	14 (trong đó có 6 chân PWM)
Số chân Analog	8 (độ phân giải 10bit, nhiều hơn Arduino Uno 2 chân)
Dòng tối đa trên mỗi chân I/O	40mA

Bộ nhớ flash	32KB với 2KB dùng bởi bootloader
SRAM	2KB
EEPROM	1KB
Xung nhịp	16MHz
Kích thước	0.73" x 1.70"
Chiều dài	45mm
Chiều rộng	18mm

❖ Màn hình LCD 16x2



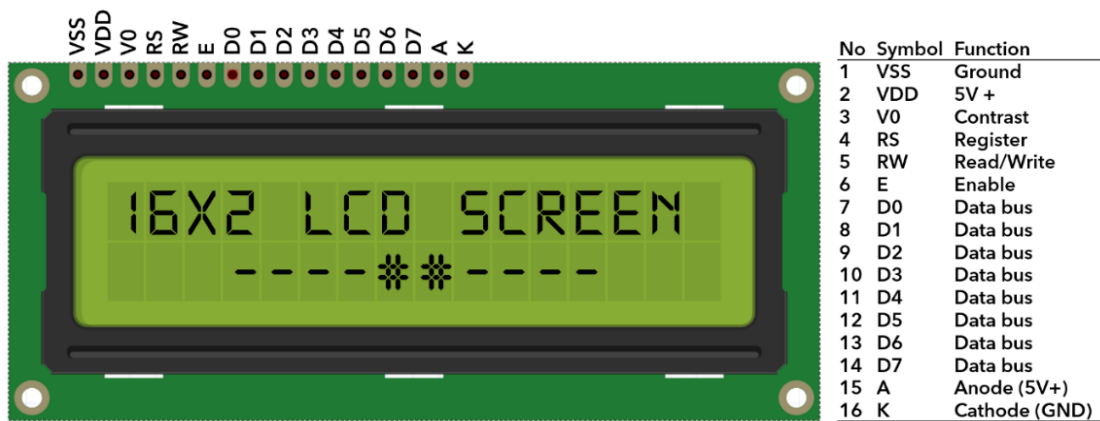
• Khái quát

Màn hình LCD 1602-Green/Blue 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. ...

• Thông số kỹ thuật

- Điện áp hoạt động là 5 V
- Kích thước: 80 x 36 x 12.5 mm
- Chữ đen, nền xanh lá hoặc chữ trắng, nền xanh dương.

- 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.
- Có bộ ký tự được xây dựng hỗ trợ tiếng Anh và tiếng Nhật, xem thêm HD44780 datasheet để biết thêm chi tiết.



Hình 3: Thông số kỹ thuật của LCD 16x2

❖ Module I2C LCD



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

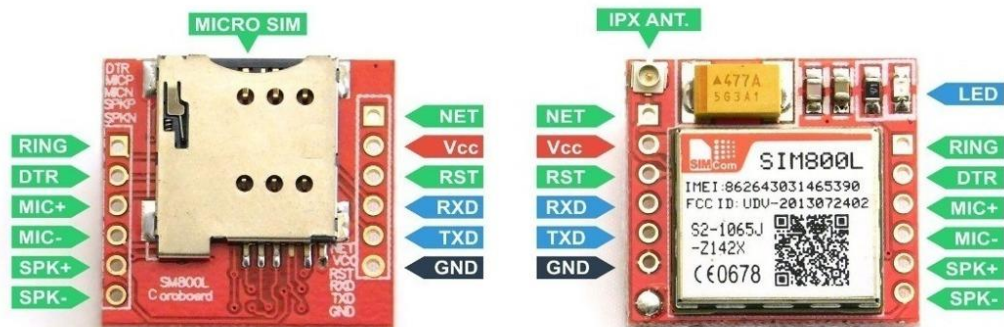
- Tiết kiệm chân cho vi điều khiển
- Dễ dàng kết nối với LCD

Thông số kĩ thuật

- Điện áp hoạt động: 2.5-6V DC
- Hỗ trợ màn hình: LCD1602,1604,2004 (driver HD44780)
- Tích hợp biến trở xoay điều chỉnh độ tương phản cho LCD 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

❖ Module Sim800L

Module Sim800L có khả năng nhắn tin SMS, nghe, gọi, GPRS, ... như một điện thoại nhưng có kích thước nhỏ nhất trong các loại module SIM (25 mm x 22 mm). Điều khiển module sử dụng bộ tập lệnh AT dễ dàng, chân kết nối dùng rào đực thông dụng (male header) chuẩn 100mil.



Thông số kĩ thuật

- Điện áp hoạt động : 3.7 - 4.2V
- Dòng khi ở chế độ chờ: 10 mA
- Dòng khi hoạt động: 100 mA đến 1A.
- Khe cắm SIM : MICROSIM
- Hỗ trợ 4 băng tần : GSM850MHz, EGSM900MHz, DSC1800Mhz, PCS1900MHz
- Có thể giao tiếp với vi điều khiển qua TTL không cần MAX232
- 1 led báo tín hiệu

Chức năng các chân:

- VCC: **Nguồn vào 4.2V.**
- TXD: Chân truyền Uart TX.
- RXD: Chân nhận Uart RX.
- DTR : Chân UART DTR, thường không xài.
- SPKP, SPKN: ngõ ra âm thanh, nối với loa để phát âm thanh (**8 Ohm-0.87W**).
- MICP, MICN: ngõ vào âm thanh, phải gắn thêm Micro để thu âm thanh.
- Reset: Chân khởi động lại Sim800L (thường không xài).
- RING : báo có cuộc gọi đến
- GND: Chân Mass, cấp 0V.

❖ Module relay



- Mạch 1 Relay Opto chọn mức kích High/Low (5/12/24VDC) được sử dụng để bật, tắt thiết bị AC/DC qua Relay, mạch có thể tùy chọn kích bằng mức cao hoặc thấp (High/Low) qua Jumper, ngoài ra mạch còn bổ sung thêm Opto cách ly cho độ an toàn và chống nhiễu vượt trội, thích hợp với các ứng dụng bật tắt, điều khiển thiết bị qua Relay.

- Tải tối đa: 250V / 10A hoặc DC 30V / 10A
- Dòng kích hoạt: 5mA
- Relay sử dụng điện áp: 5V
- Sử dụng SMD optocoupler cách ly, hoạt động ổn định, các mô-đun có thể được thiết lập để mức độ cao hay thấp bằng cách thay đổi jumper. Thiết kế sửa lỗi, ngay cả khi các dòng điều khiển bị hỏng, role sẽ không hoạt động.

❖ **Module LM2596**



Mạch giảm áp DC nhỏ gọn có khả năng giảm áp từ 30V xuống 1.5V mà vẫn đạt hiệu suất cao (92%). Thích hợp cho các ứng dụng chia nguồn, hạ áp, cấp cho các thiết bị như camera, motor, robot,...

Thông số kỹ thuật:

- Điện áp đầu vào: Từ 3V đến 30V.
- Điện áp đầu ra: Điều chỉnh được trong khoảng 1.5V đến 30V.
- Dòng đáp ứng tối đa là 3A.
- Hiệu suất : 92%
- Công suất : 15W
- Kích thước: 45 (dài) * 20 (rộng) * 14 (cao) mm

❖ Adapter 12V

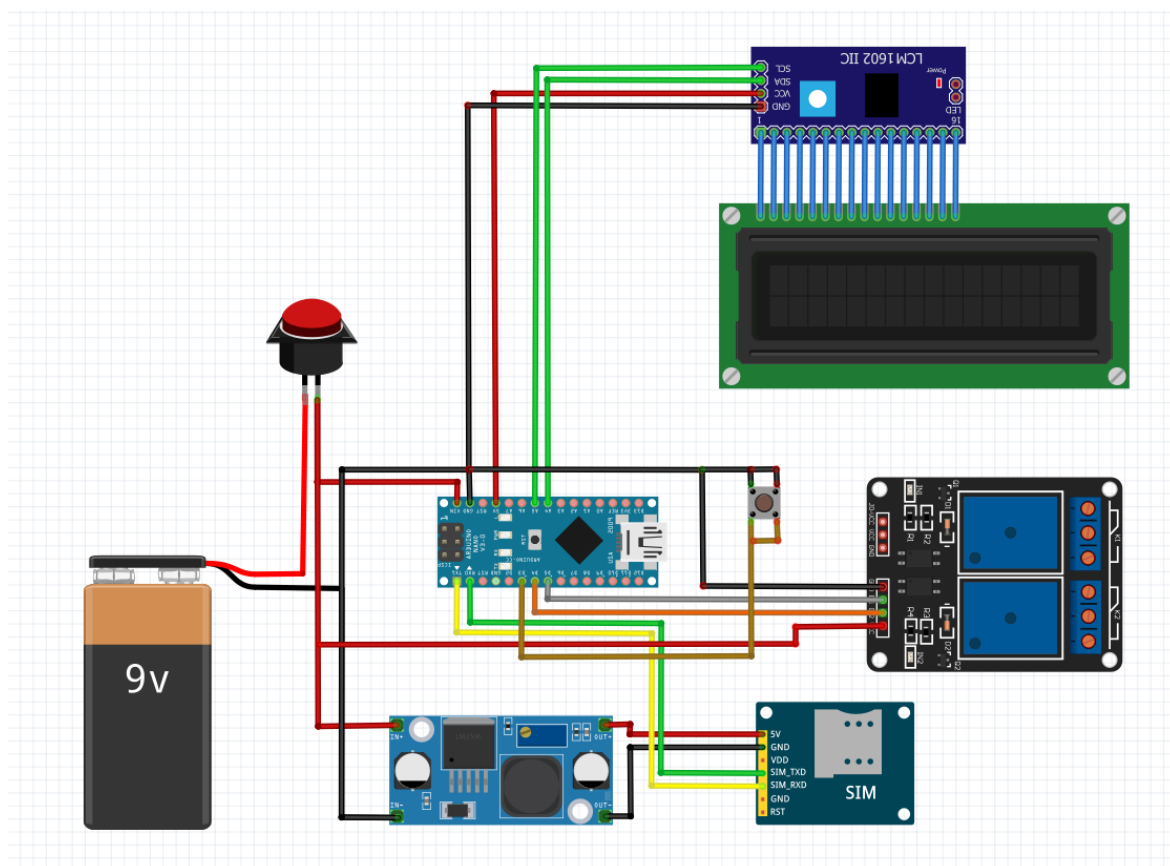
Nguồn Adapter 12V 2A cung cấp nguồn cho các thiết bị sử dụng điện áp 12V, nguồn có thiết kế nhỏ gọn, độ bền cao và dễ sử dụng.

Thông số kỹ thuật:

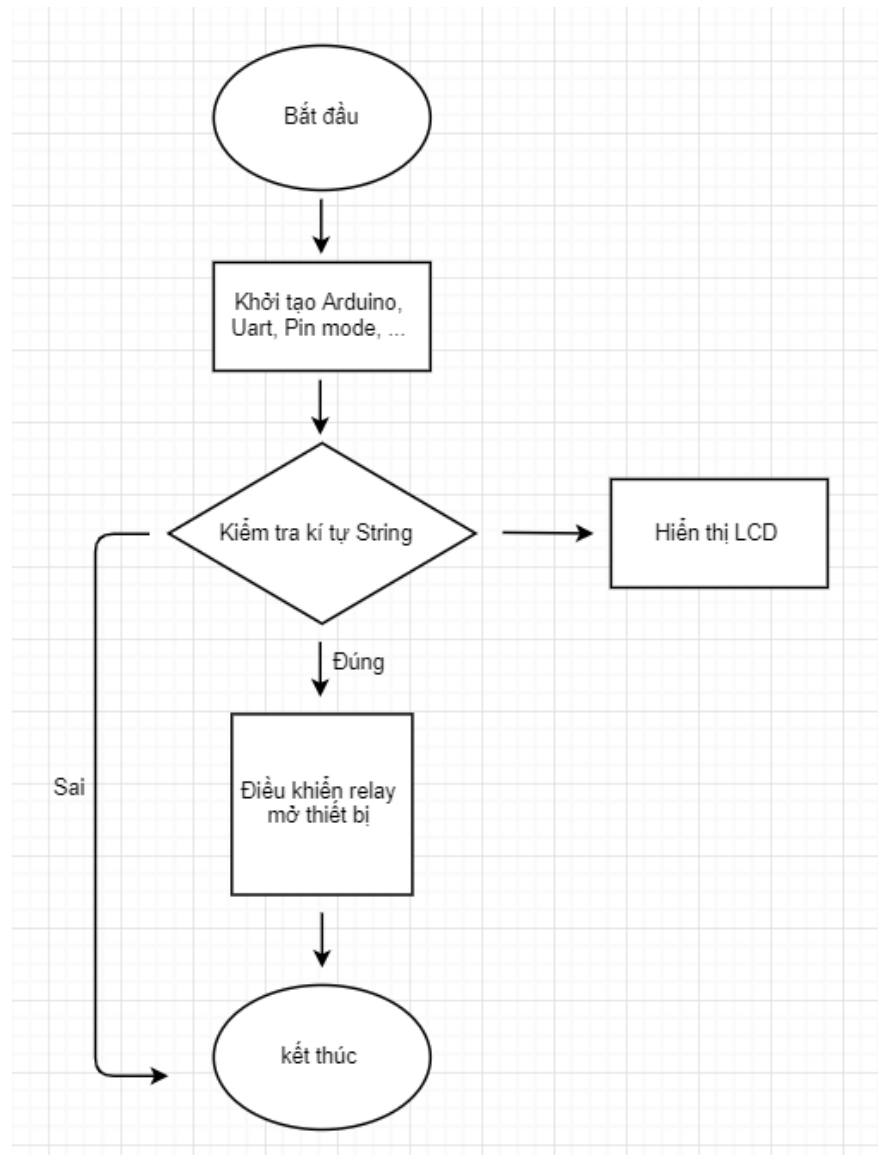
- Điện áp ngõ vào: 100V - 240V
- Output: DC 12V 2A / 1000mA
- Jack ngõ ra: 5.5mm x 2.1mm
- Kích thước: 7,6 x 7,5 x 2,6 cm
- Chiều dài cáp: 100cm
- Adapter kết nối bên trong cực dương (+) bên ngoài âm (-)

CHƯƠNG 3: CƠ SỞ THỰC HIỆN

3.1 Sơ đồ kết nối, lưu đồ giải thuật



Hình 4: Sơ đồ kết nối

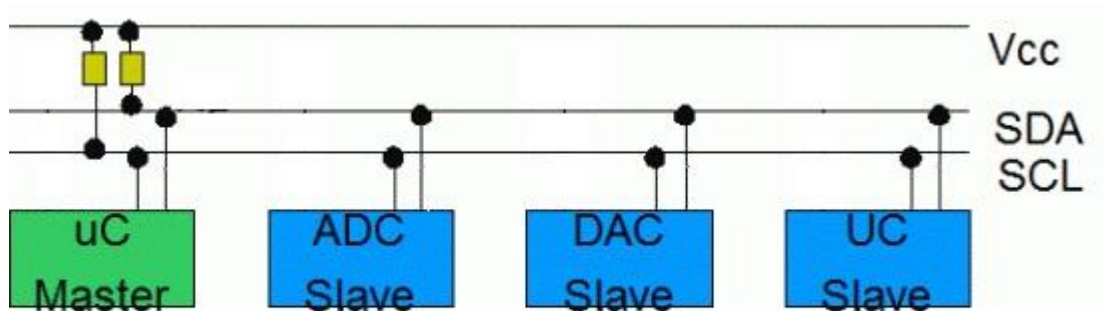


Hình 6: Lưu đồ giải thuật

- **Giao tiếp I2C:**

- Đầu năm 1980 Phillips đã phát triển một chuẩn giao tiếp nối tiếp 2 dây được gọi là I2C. I2C là tên viết tắt của cụm từ Inter-Integrated Circuit. Đây là đường Bus giao tiếp giữa các IC với nhau. I2C mặc dù được phát triển bởi Philips nhưng nó đã được rất nhiều nhà sản xuất IC trên thế giới sử dụng. I2C trở thành một chuẩn công nghiệp cho các giao tiếp điều khiển, có thể kể ra đây một vài tên tuổi ngoài Philips như: Texas Instrument(TI), MaximDallas, analog Device, National Semiconductor ... Bus I2C được sử dụng làm bus giao tiếp ngoại vi cho rất nhiều loại IC khác nhau như các loại Vi điều khiển 8051, PIC, AVR, ARM... chip nhớ như: RAM tĩnh (Static

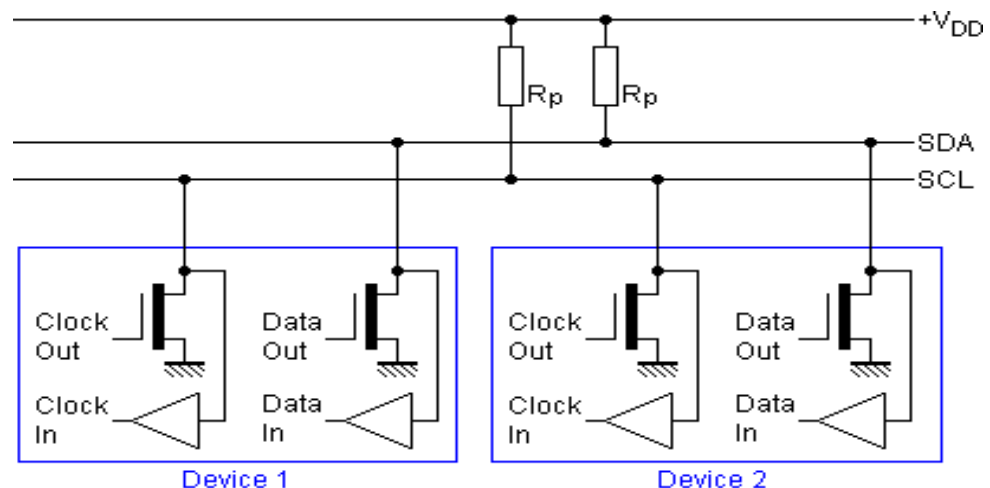
Ram), EEPROM, bộ chuyển đổi tương tự số (ADC), số tương tự(DAC), IC điều khiển LCD, LED...



Hình 5: Bus I2C và các thiết bị ngoại vi

a. Đặc điểm giao tiếp I2C

Một giao tiếp I2C gồm có 2 dây: Serial Data (SDA) và Serial Clock (SCL). SDA là đường truyền dữ liệu 2 hướng, còn SCL là đường truyền xung đồng hồ để đồng bộ và chỉ theo một hướng. Như ta thấy trên hình vẽ trên, khi một thiết bị ngoại vi kết nối vào đường bus I2C thì chân SDA của nó sẽ nối với dây SDA của bus, chân SCL sẽ nối với dây SCL.



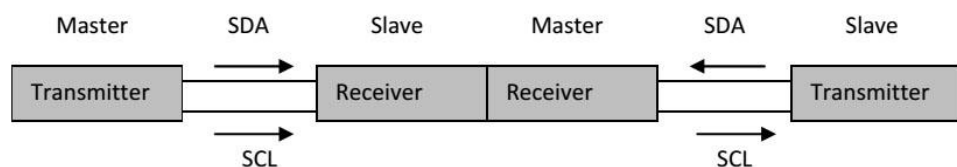
Hình 7: Kết nối thiết bị vào bus I2C ở chế độ chuẩn (Standard mode) và chế độ nhanh (Fast mode)

Mỗi dây SDA hay SCL đều được nối với điện áp dương của nguồn cấp thông qua một điện trở kéo lên (pullup resistor). Sự cần thiết của các điện trở kéo này là vì chân giao tiếp I2C của các thiết bị ngoại vi thường là dạng cực mở (opendrain hay opencollector). Giá trị của các điện trở này khác nhau

tùy vào từng thiết bị và chuẩn giao tiếp, thường dao động trong khoảng 1K đến 4.7k

Trở lại với hình 3.1, ta thấy có rất nhiều thiết bị (ICs) cùng được kết nối vào một bus I2C, tuy nhiên sẽ không xảy ra chuyện nhầm lẫn giữa các thiết bị, bởi mỗi thiết bị sẽ được nhận ra bởi một địa chỉ duy nhất với một quan hệ chủ/tớ tồn tại trong suốt thời gian kết nối. Mỗi thiết bị có thể hoạt động như là thiết bị nhận hoặc truyền dữ liệu hay có thể vừa truyền vừa nhận. Hoạt động truyền hay nhận còn tùy thuộc vào việc thiết bị đó là chủ (master) hay tớ (slave).

Một thiết bị hay một IC khi kết nối với bus I2C, ngoài một địa chỉ (duy nhất) để phân biệt, nó còn được cấu hình là thiết bị chủ hay tớ. Tại sao lại có sự phân biệt này? Đó là vì trên một bus I2C thì quyền điều khiển thuộc về thiết bị chủ. Thiết bị chủ nắm vai trò tạo xung đồng hồ cho toàn hệ thống, khi giữa hai thiết bị chủ-tớ giao tiếp thì thiết bị chủ có nhiệm vụ tạo xung đồng hồ và quản lý địa chỉ của thiết bị tớ trong suốt quá trình giao tiếp. Thiết bị chủ giữ vai trò chủ động, còn thiết bị tớ giữ vai trò bị động trong việc giao tiếp.



Hình 8 : Cơ chế giao tiếp I2C

Nhìn hình trên ta thấy xung đồng hồ chỉ có một hướng từ chủ đến tớ, còn luồng dữ liệu có thể đi theo hai hướng, từ chủ đến tớ hay ngược lại tớ đến chủ.

b. Chế độ hoạt động (tốc độ truyền):

Các bus I2C có thể hoạt động ở ba chế độ, hay nói cách khác các dữ liệu trên bus I2C có thể được truyền trong ba chế độ khác nhau

- Chế độ tiêu chuẩn (Standard mode).
- Chế độ nhanh (Fast mode) .
- Chế độ cao tốc High-Speed (Hs) mode.

Chế độ tiêu chuẩn:

- Đây là chế độ tiêu chuẩn ban đầu được phát hành vào đầu những năm 80
- Nó có tốc độ dữ liệu tối đa 100kbps
- Nó sử dụng 7-bit địa chỉ, và 112 địa chỉ tớ

Tăng cường hoặc chế độ nhanh:

- Tốc độ dữ liệu tối đa được tăng lên đến 400 kbps.
- Để ngăn chặn gai tiếng ồn, ngõ vào của thiết bị Fast-mode là Schmitt-triggered.
- Chân SCL và SDA của một thiết bị tới I2C ở trạng thái trở kháng cao khi không cấp nguồn.

Chế độ cao tốc (High-Speed):

Chế độ này đã được tạo ra chủ yếu để tăng tốc độ dữ liệu lên đến 36 lần nhanh hơn so với chế độ tiêu chuẩn. Nó cung cấp 1,7 Mbps (với $C_b = 400 \text{ pF}$), và 3.4Mbps (với $C > b = 100 \text{ pF}$).

Một bus I2C có thể hoạt động ở nhiều chế độ khác nhau:

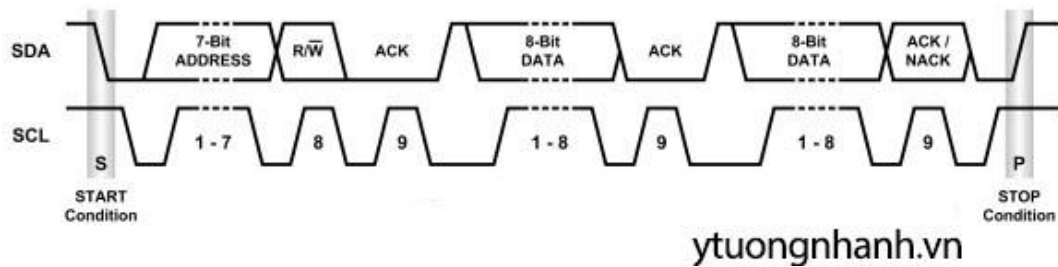
- Một chủ một tớ (one master - one slave)
- Một chủ nhiều tớ (one master - multi slave)
- Nhiều chủ nhiều tớ (Multi master - Multi slave)

Dù ở chế độ nào, một giao tiếp I2C đều dựa vào quan hệ chủ/tớ. Giả thiết một thiết bị A muốn gửi dữ liệu đến thiết bị B, quá trình được thực hiện như sau:

- Thiết bị A (Chủ) xác định đúng địa chỉ của thiết bị B (tớ), cùng với việc xác định địa chỉ, thiết bị A sẽ quyết định việc đọc hay ghi vào thiết bị tới. Thiết bị A gửi dữ liệu tới thiết bị B

- Thiết bị A kết thúc quá trình truyền dữ liệu.

Khi A muốn nhận dữ liệu từ B, quá trình diễn ra như trên, chỉ khác là A sẽ nhận dữ liệu từ B. Trong giao tiếp này, A là chủ còn B vẫn là tớ. Chi tiết việc thiết lập một giao tiếp giữa hai thiết bị sẽ được mô tả chi tiết trong các mục dưới đây.

Trình tự truyền bit trên đường truyền:

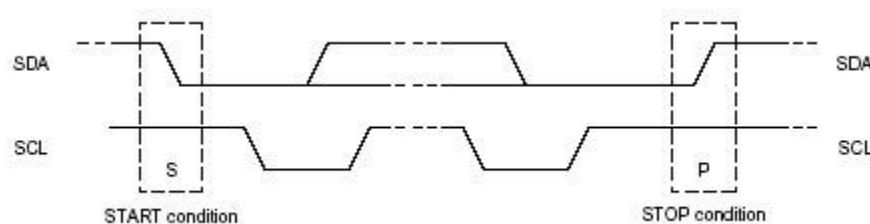
Hình 9: Trình tự truyền bit

- Thiết bị chủ tạo một điều kiện start. Điều kiện này thông báo cho tất cả các thiết bị tớ lắng nghe dữ liệu trên đường truyền
- Thiết bị chủ gửi địa chỉ của thiết bị tớ mà thiết bị chủ muốn giao tiếp và cờ đọc/ghi dữ liệu (nếu cờ thiết lập lên 1 byte tiếp theo được truyền từ thiết bị tớ đến thiết bị chủ, nếu cờ thiết lập xuống 0 thì byte tiếp theo truyền từ thiết bị chủ đến thiết bị tớ).
- Khi thiết bị tớ trên bus I2C có địa chỉ đúng với địa chỉ mà thiết bị chủ gửi sẽ phản hồi lại bằng một xung ACK.
- Giao tiếp giữa thiết bị chủ và tớ trên bus dữ liệu bắt đầu. Cả chủ và tớ đều có thể nhận hoặc truyền dữ liệu tùy thuộc vào việc truyền thông là đọc hay viết. Bộ truyền gửi 8 bit dữ liệu tới bộ nhận, Bộ nhận trả lời với một bit ACK.
- Để kết thúc quá trình giao tiếp, thiết bị chủ tạo ra một điều kiện stop.

c. Điều kiện START và STOP (START and STOP conditions)

START và STOP là những điều kiện bắt buộc phải có khi một thiết bị chủ muốn thiết lập giao tiếp với một thiết bị nào đó trên bus I2C. START là điều kiện khởi đầu, báo hiệu bắt đầu của giao tiếp, còn STOP báo hiệu kết thúc một giao tiếp. Hình dưới đây mô tả điều kiện START và STOP.

Ban đầu khi chưa thực hiện quá trình giao tiếp, cả hai đường SDA và SCL đều ở mức cao (**SDA = SCL = HIGH**). Lúc này bus I2C được coi là rỗi (“bus free”), sẵn sàng cho một giao tiếp. Hai điều kiện START và STOP là không thể thiếu trong việc giao tiếp giữa các thiết bị I2C với nhau.



Hình : Giao thức start và stop

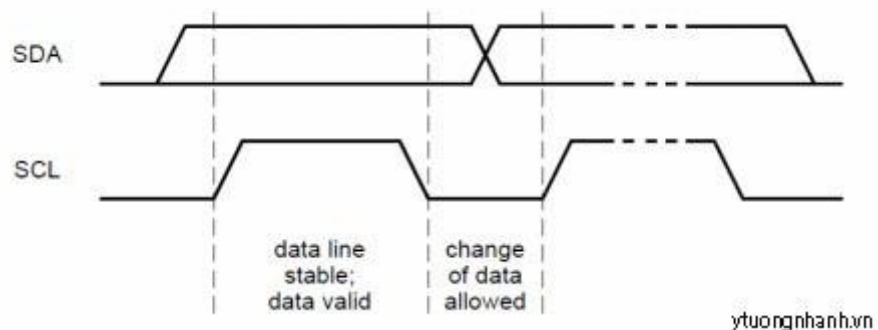
Điều kiện START: một sự chuyển đổi trạng thái từ cao xuống thấp trên đường SDA trong khi đường SCL đang ở mức cao (cao = 1; thấp = 0) báo hiệu một điều kiện START

Điều kiện STOP: Một sự chuyển đổi trạng thái từ mức thấp lên cao trên đường SDA trong khi đường SCL đang ở mức cao. Cả hai điều kiện START và STOP đều được tạo ra bởi thiết bị chủ. Sau tín hiệu START, bus I2C coi như đang trong trạng thái làm việc (busy). Bus I2C sẽ rỗi, sẵn sàng cho một giao tiếp mới sau tín hiệu STOP từ phía thiết bị chủ.

Sau khi có một điều kiện START, trong quá trình giao tiếp, khi có một tín hiệu START được lặp lại thay vì một tín hiệu STOP thì bus I2C vẫn tiếp tục trong trạng thái bận. Tín hiệu START và lặp lại START (Repeated START) đều có chức năng giống nhau là khởi tạo một giao tiếp.

Truyền dữ liệu:

Mỗi xung clock có một bit dữ liệu được truyền. Mức tín hiệu SDA chỉ được thay đổi khi xung clock đang ở mức thấp và ổn định khi xung clock ở mức cao. Thiết bị tớ có thể lấy mẫu dữ liệu khi xung clock ở mức cao.

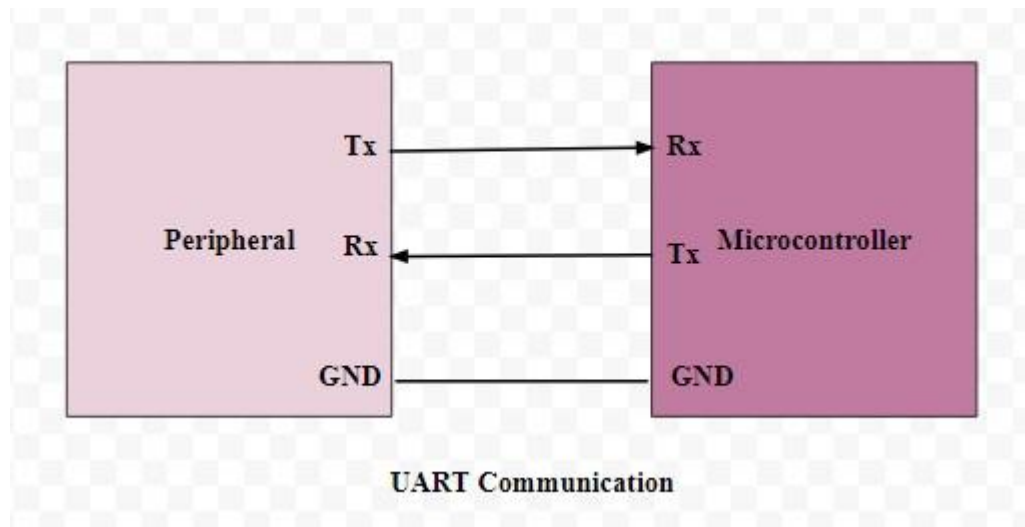


Hình 10: Truyền dữ liệu

Giao tiếp UART:

Giao tiếp UART có tên đầy đủ là “Universal Asynchronous Receiver / Transmitter”, và nó là một vi mạch sẵn có trong một vi điều khiển nhưng không giống như một giao thức truyền thông (I2C & SPI). Chức năng chính của UART là truyền dữ liệu nối tiếp. Trong UART, giao tiếp giữa hai thiết bị

có thể được thực hiện theo hai cách là giao tiếp dữ liệu nối tiếp và giao tiếp dữ liệu song song.

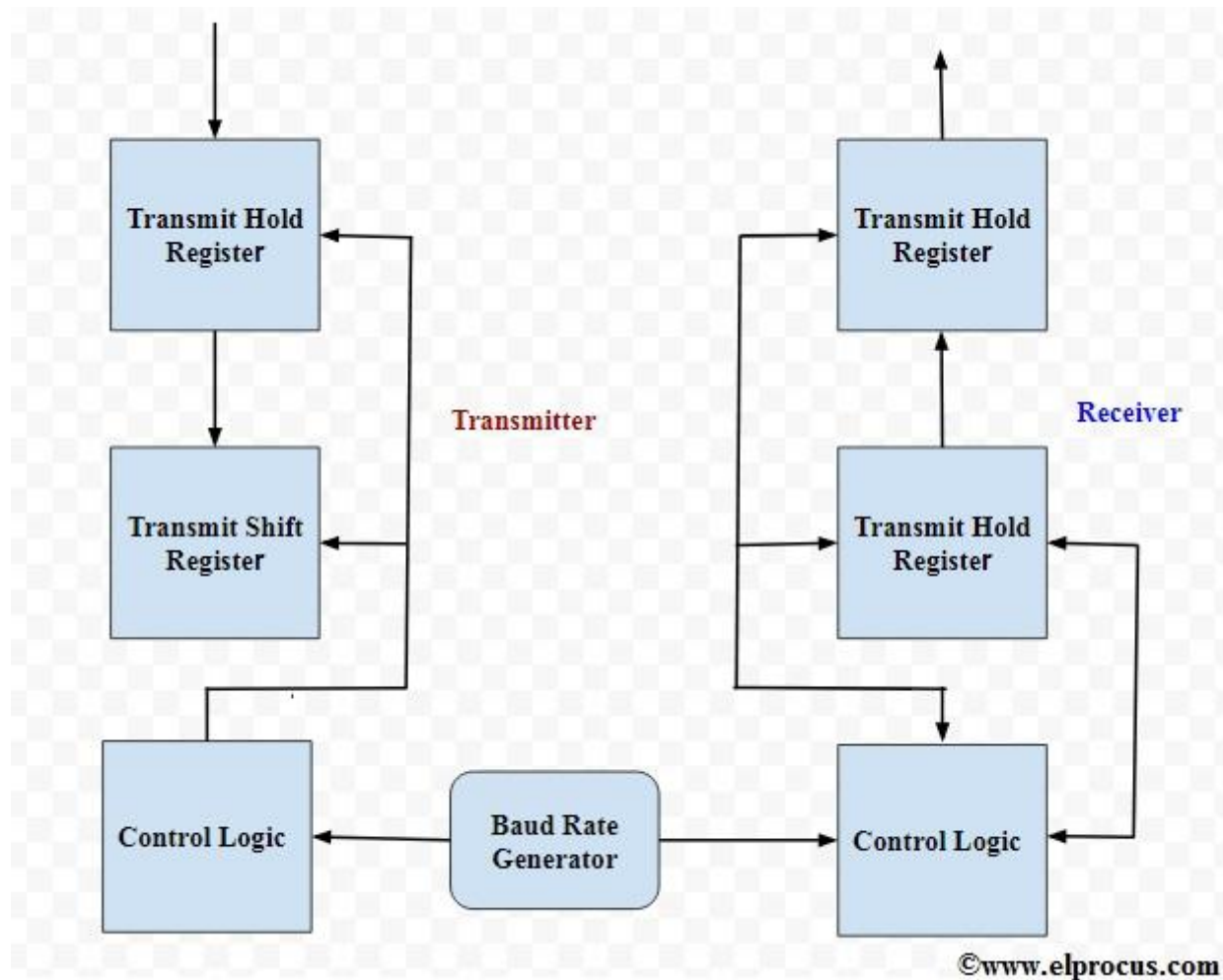


Sơ đồ khối UART

Sơ đồ khối UART bao gồm hai thành phần là máy phát và máy thu được hiển thị bên dưới. Phần máy phát bao gồm ba khối là thanh ghi giữ truyền, thanh ghi dịch chuyển và logic điều khiển. Tương tự, phần máy thu bao gồm một thanh ghi giữ, thanh ghi thay đổi và logic điều khiển. Hai phần này thường được cung cấp bởi một bộ tạo tốc độ baud. Trình tạo này được sử dụng để tạo tốc độ khi phần máy phát và phần máy thu phải truyền hoặc nhận dữ liệu.

Thanh ghi giữ trong máy phát bao gồm byte dữ liệu được truyền. Các thanh ghi thay đổi trong máy phát và máy thu di chuyển các bit sang phải hoặc trái cho đến khi một byte dữ liệu được truyền hoặc nhận. Một logic điều khiển đọc (hoặc) ghi được sử dụng để biết khi nào nên đọc hoặc viết.

Máy phát tốc độ baud giữa máy phát và máy thu tạo ra tốc độ dao động từ 110 bps đến 230400 bps. Thông thường, tốc độ truyền của vi điều khiển là 9600 đến 115200.

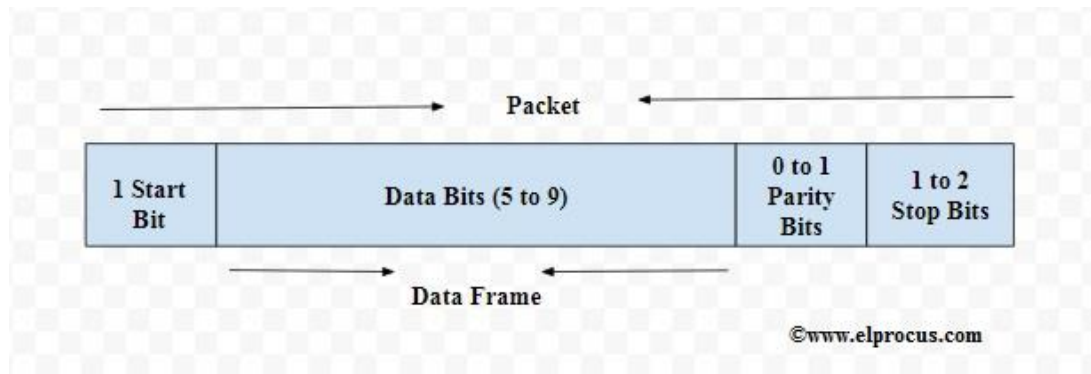


Hình 11: Sơ đồ khối UART

Truyền thông UART

Trong giao tiếp này, có hai loại UART có sẵn là truyền UART và nhận UART và giao tiếp giữa hai loại này có thể được thực hiện trực tiếp với nhau. Đối với điều này, chỉ cần hai cáp để giao tiếp giữa hai UART. Luồng dữ liệu sẽ từ cả hai chân truyền (Tx) và nhận (Rx) của UARTs. Trong UART, việc truyền dữ liệu từ Tx UART sang Rx UART có thể được thực hiện không đồng bộ (không có tín hiệu CLK để đồng bộ hóa các bit 0 / 1).

Việc truyền dữ liệu của UART có thể được thực hiện bằng cách sử dụng bus dữ liệu ở dạng song song bởi các thiết bị khác như vi điều khiển, bộ nhớ, CPU, v.v. Sau khi nhận được dữ liệu song song từ bus, nó tạo thành gói dữ liệu bằng cách thêm ba bit như bắt đầu, dừng lại và trung bình. Nó đọc từng bit gói dữ liệu và chuyển đổi dữ liệu nhận được thành dạng song song để loại bỏ ba bit của gói dữ liệu. Tóm lại, gói dữ liệu nhận được bởi UART chuyển song song về phía bus dữ liệu ở đầu nhận.



Hình 12: Truyền thông UART

Start-bit

Start-bit còn được gọi là bit đồng bộ hóa được đặt trước dữ liệu thực tế. Nói chung, một đường truyền dữ liệu không hoạt động được điều khiển ở mức điện áp cao. Để bắt đầu truyền dữ liệu, truyền UART kéo đường dữ liệu từ mức điện áp cao (1) xuống mức điện áp thấp (0). UART thu được thông báo sự chuyển đổi này từ mức cao sang mức thấp qua đường dữ liệu cũng như bắt đầu hiểu dữ liệu thực. Nói chung, chỉ có một start-bit.

Bit dừng

Bit dừng được đặt ở phần cuối của gói dữ liệu. Thông thường, bit này dài 2 bit nhưng thường chỉ sử dụng 1 bit. Để dừng sóng, UART giữ đường dữ liệu ở mức điện áp cao.

Bit chẵn lẻ

Bit chẵn lẻ cho phép người nhận đảm bảo liệu dữ liệu được thu thập có đúng hay không. Đây là một hệ thống kiểm tra lỗi cấp thấp & bit chẵn lẻ có sẵn trong hai phạm vi như Chẵn lẻ – chẵn lẻ cũng như Chẵn lẻ – lẻ. Trên thực tế, bit này không được sử dụng rộng rãi nên không bắt buộc.

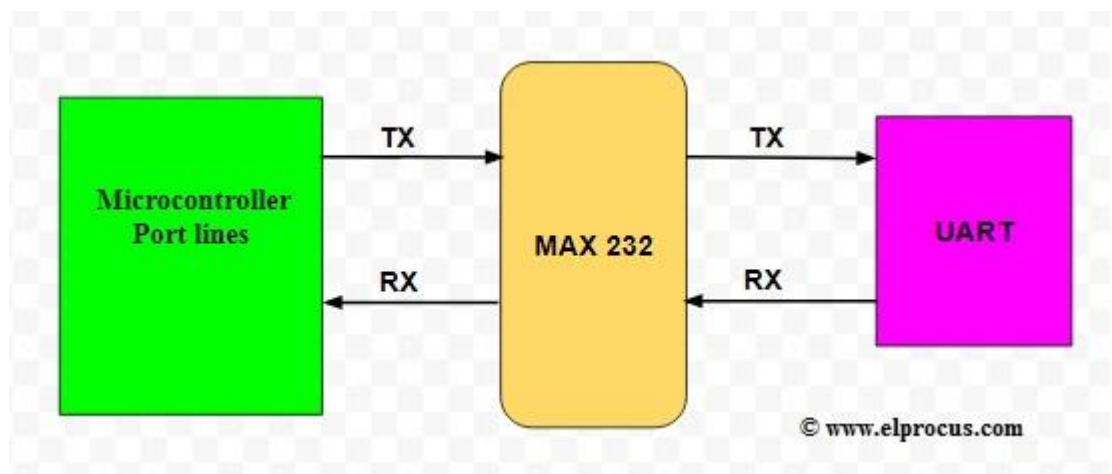
Dữ liệu bit hoặc khung dữ liệu

Các bit dữ liệu bao gồm dữ liệu thực được truyền từ người gửi đến người nhận. Độ dài khung dữ liệu có thể nằm trong khoảng 5 & 8. Nếu bit chẵn lẻ không được sử dụng thì chiều dài khung dữ liệu có thể dài 9 bit. Nói chung, LSB của dữ liệu được truyền trước tiên sau đó nó rất hữu ích cho việc truyền.

Giao diện UART

Hình dưới đây cho thấy UART giao tiếp với vi điều khiển. Giao tiếp UART có thể được thực hiện bằng ba tín hiệu như TXD, RXD và GND.

Bằng cách sử dụng điều này, chúng ta có thể hiển thị một văn bản trong máy tính cá nhân từ board vi điều khiển 8051 cũng như mô-đun UART. Trong board 8051, có hai giao diện nối tiếp như UART0 và UART1. Ở đây, giao diện UART0 được sử dụng. Chân Tx truyền thông tin đến chân PC & Rx nhận thông tin từ PC. Tốc độ Baud có thể được sử dụng để biểu thị tốc độ của cả vi điều khiển và PC. Việc truyền và nhận dữ liệu có thể được thực hiện đúng khi tốc độ truyền của cả vi điều khiển và PC là tương tự nhau.



Hình 13: Giao diện UART

3.2 Nguyên lý hoạt động

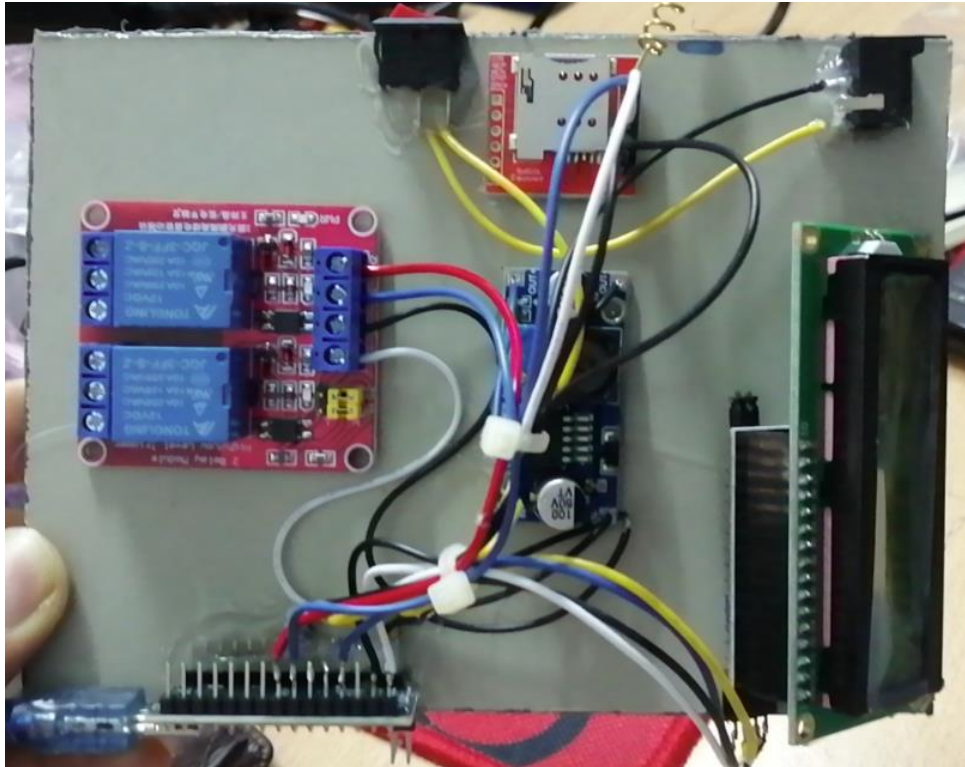
- Sau khi khởi động xong kết nối Uart và hệ thống, Module Sim800L sẽ được kích hoạt và hiển thị thông tin lên màn hình LCD 16x2. Nếu ký tự đọc được từ tin nhắn gửi về Module sim800L đúng cú pháp lệnh set up trước trong code thì Arduino sẽ điều khiển đóng / mở relay để điều khiển thiết bị.

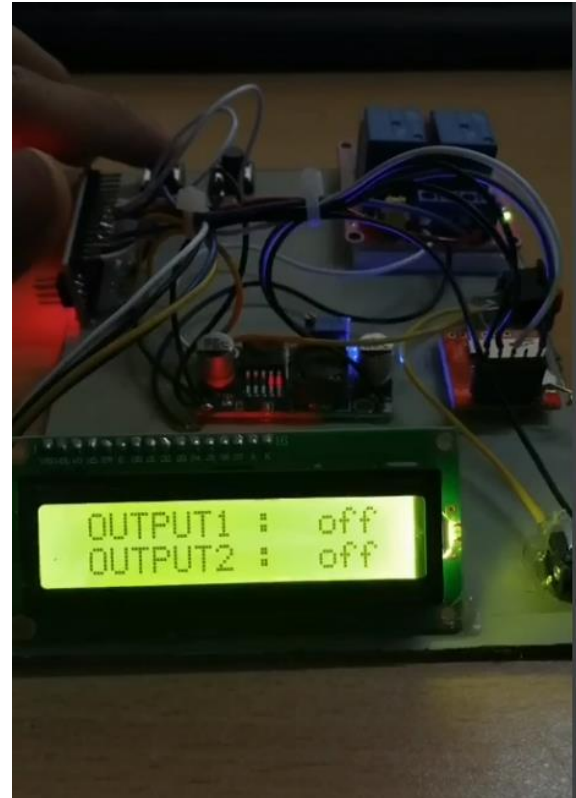
CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

4.1 Mô hình thực tế

Đồ án tập trung vào nghiên cứu cơ chế hoạt động của ARDUINO về cách truy xuất dữ liệu và cách lập trình. Về cơ bản đồ án đã đạt được những mục tiêu đã đề ra.

Một số hình ảnh về mô hình:





CHƯƠNG 5: KẾT LUẬN VÀ ĐỊNH HƯỚNG ĐỀ TÀI

5.1 Kết quả đạt được

a. Cơ sở lý thuyết

- Hiểu được cách điều khiển và hoạt động của Arduino, module Sim800L, LCD 16x2, module relay và module nguồn LM2596...
- Tinh chỉnh các hệ số lập trình và phần cứng để hệ thống hoạt động hoàn chỉnh.
- Áp dụng các kiến thức đã học vào thực tiễn.

b. Thi công đồ án

- Thi công được tất cả các phần cứng.
- Lập trình và đã thử nghiệm các phần cứng đã hoạt động ổn định.
- Sử dụng vật liệu có sẵn.
- Cơ cấu đơn giản nhưng hiệu năng cao.

5.2 Hạn chế

Trong thời gian làm đồ án vừa qua, em đã nghiên cứu và tích hợp nhiều phương pháp điều khiển khác nhau.

- Tìm hiểu rất kỹ tất cả phần cứng và phần mềm của hệ thống.
- Tìm hiểu cách sử dụng và cách kết nối thiết bị với nhau.
- Hiểu được cách thức giao tiếp I2C, UART giữa các trang thiết bị với nhau.

Tuy nhiên, do giới hạn về thời gian và trình độ nên trong khi làm đồ án tốt nghiệp lần này em cũng không tránh khỏi những hạn chế. Em mong được sự góp ý của các thầy cô và các bạn để có thể xây dựng và thiết kế hoàn chỉnh mô hình hơn trong tương lai.

5.3 Hướng phát triển của đề tài

- Áp dụng vào đời sống thực tiễn
- Tự động hóa trang thiết bị.
- Có tính ứng dụng cao và chi phí thấp.
- Tăng năng suất .

PHỤ LỤC

Code chương trình

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);

#include <SoftwareSerial.h>

SoftwareSerial mySerial(3, 2);

char incomingByte;

String incomingData;

bool atCommand = true;


#define OUT1 4

#define OUT2 5

#define button1 6

#define button2 7

int index = 0;

int bien=0;

int bien1=0;

int val;

String number = "";

String message = "";

void setup(){
```

```
Serial.begin(9600);

mySerial.begin(9600);

pinMode(OUT1, OUTPUT);

pinMode(OUT2, OUTPUT);

pinMode(button1, INPUT_PULLUP);

pinMode(button2, INPUT_PULLUP);

digitalWrite(OUT1,LOW);

digitalWrite(OUT2,LOW);

lcd.init();

lcd.backlight();

lcd.setCursor(0,0);

lcd.print("Dieu khien thiet");

lcd.setCursor(0,1);

lcd.print("bi bang Sim 800L");

delay(4000);

lcd.setCursor(0,0);

lcd.print("Nguyen Van Thanh");

lcd.setCursor(0,1);

lcd.print("MSSV: 2202180040");

while(!mySerial.available()){

    mySerial.println("AT");

    delay(1000);

    Serial.println("connecting....");

}
```

```
Serial.println("Connected..");

mySerial.println("AT+CMGF=1"); //Set SMS Text Mode

delay(1000);

mySerial.println("AT+CNMI=1,2,0,0,0"); //procedure, how to receive messages
from the network

delay(1000);

mySerial.println("AT+CMGL=\"REC UNREAD\""); // Read unread messages

Serial.println("Ready to receive Commands..");

lcd.clear();

lcd.setCursor(0,0);

lcd.print(" OUTPUT1 : off ");

lcd.setCursor(0,1);

lcd.print(" OUTPUT2 : off ");

}

void manual(){

if(digitalRead(button1)==0){

    bien++; while(digitalRead(button1)==0);

    if (bien==2){bien=0;}

    }

if(digitalRead(button2)==0){

    bien1++; while(digitalRead(button2)==0);

    if (bien1==2){bien1=0;}

    }
```

```
if (bien==1 || val==1){  
    digitalWrite(OUT1, HIGH);  
    lcd.setCursor(12,0);  
    lcd.print("on ");}
```

```
if (bien==0 && val==2){  
    digitalWrite(OUT1, LOW);  
    lcd.setCursor(12,0);  
    lcd.print("off");}
```

```
if (bien1==1 || val==3){  
    digitalWrite(OUT2, HIGH);  
    lcd.setCursor(12,1);  
    lcd.print("on ");}
```

```
if(bien1==0&& val==4){  
    digitalWrite(OUT2, LOW);  
    lcd.setCursor(12,1);  
    lcd.print("off");}  
}
```

```
void loop(){  
    if(mySerial.available()){  
        delay(100);  
        // Serial buffer  
        while(mySerial.available()){
```

```
incomingByte = mySerial.read();
incomingData += incomingByte;
}

delay(10);

if(atCommand == false){
    receivedMessage(incomingData);
}

else{
    atCommand = false;
}

//delete messages to save memory
if (incomingData.indexOf("OK") == -1){
    mySerial.println("AT+CMGDA=\"DEL ALL\"");
    delay(1000);
    atCommand = true;
}

incomingData = "";
}

manual();
}

void receivedMessage(String inputString){

    //Nhận số điện thoại người gửi
    index = inputString.indexOf("")+1;
```



```
inputString = inputString.substring(index);  
index = inputString.indexOf("");  
number = inputString.substring(0,index);  
Serial.println("Number: " + number);  
  
//Nhận tin nhắn người gửi  
index = inputString.indexOf("\n")+1;  
message = inputString.substring(index);  
message.trim();  
Serial.println("Message: " + message);  
message.toUpperCase(); // uppercase the message received  
//điều khiển thiết bị  
  
if (message.indexOf("ON1") > -1){  
    val=1;  
    Serial.println("Command: Lamp1 Turn On.");  
}  
if (message.indexOf("OFF1") > -1){  
    val=2;  
    Serial.println("Command: Lamp1 Turn Off.");  
}  
if (message.indexOf("ON2") > -1){  
    val=3;  
    Serial.println("Command: Lamp2 Turn On.");
```

```
}  
  
if (message.indexOf("OFF2") > -1){  
    val=4;  
    Serial.println("Command: Lamp2 Turn Off.");  
}  
  
delay(50);  
}
```

Giới thiệu phần mềm sử dụng

- Phần mềm Arduino IDE được lập trình để giao tiếp với các board Arduino, gọi là **Intergrated Development Environment (IDE)**. Công cụ này được đội ngũ kỹ sư của Arduino phát triển và có thể chạy trên Windows , MAC OS X và Linux.

Hướng dẫn cài đặt phần mềm

a. Cài đặt arduino

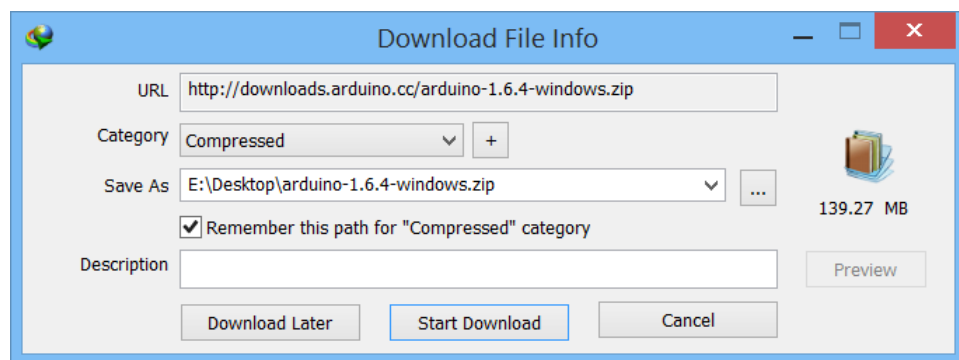
- **Bước 1:** Truy cập địa chỉ <http://arduino.cc/en/Main/Software/...> . Đây là nơi lưu trữ cũng như cập nhật các bản IDE của Arduino. Bấm vào mục [Windows ZIPfile for non admin install](#) như hình minh họa.



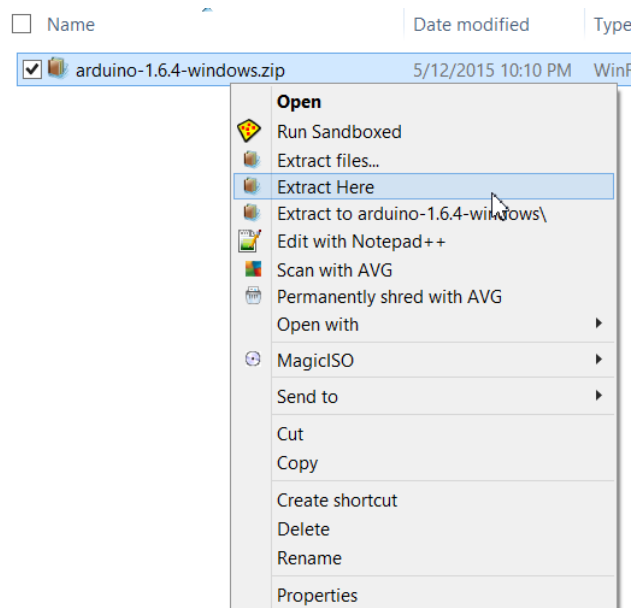
Bạn sẽ được chuyển đến một trang mời quyền góp tiền để phát triển phần mềm cho Arduino, tiếp tục bấm [JUST DOWNLOAD](#) để bắt đầu tải.


Contribute to the Arduino Software

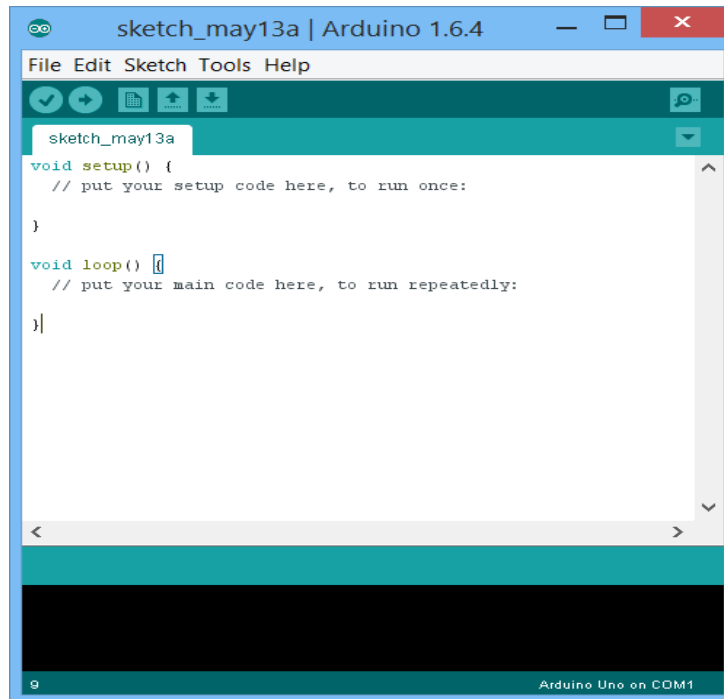
Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). [Learn more on how your contribution will be used.](#)



- **Bước 2:** Sau khi download xong, các bạn bấm chuột phải vào file vừa download **arduino-1.6.4-windows.zip** và chọn “**Extract here**” để giải nén.



- **Bước 3:** Copy thư mục **arduino-1.6.4** vừa giải nén đến nơi lưu trữ.
- **Bước 4:** Chạy file  **arduino.exe** trong thư mục **arduino-1.6.4** để khởi động **Arduino IDE**



b. Cài đặt Driver

- Để máy tính của bạn và board Arduino giao tiếp được với nhau, chúng ta cần phải cài đặt driver trước tiên.
- Nếu bạn dùng Windows 8, trong một số trường hợp Windows **không cho phép** bạn cài Arduino driver (do driver không được kí bằng chữ kí số hợp lệ). Do vậy bạn cần vào Windows ở chế độ **Disable driver signature enforcement** thì mới cài được driver
- Xem hướng dẫn thực hiện tại bài viết [Disabling Driver Signature on Windows 8](#) của **SparkFun**.
- Đầu tiên, các bạn chạy file **arduino-1.6.4\drivers\dpinst-x86.exe** (Windows x86) hoặc **arduino-1.6.4\drivers\dpinst-amd64.exe** (Windows x64). Cửa sổ **“Device Driver Installation Wizard”** hiện ra, các bạn chọn **Next** để tiếp tục.



- Khi có yêu cầu xác nhận cài đặt driver, chọn **“Install”**



- Đợi khoảng 10 giây trong lúc quá trình cài đặt diễn ra ...



- Quá trình cài đặt đã hoàn tất. Bấm “Finish” để thoát.



TÀI LIỆU THAM KHẢO

- <http://arduino.vn>
- <http://hshop.vn>
- www.alldatasheet.com