Bìa

Lời mở đầu

Lời cảm ơn

**MỤC LỤC**

[PHẦN 1: INTERNET OF THINGS (GIỚI THIỆU, ĐỊNH NGHĨA, ỨNG DỤNG CỦA IOT,… TỰ THÊM VÀO) 5](#_Toc485132949)

[PHẦN 2: ARDUINO 6](#_Toc485132950)

[2.1. Giới thiệu các mạch, cảm biến và module sử dụng 6](#_Toc485132951)

[2.1.1. Arduino UNO R3 6](#_Toc485132952)

[2.1.2. Arduino Motor Shield L293D 10](#_Toc485132953)

[2.1.3. Cảm biến siêu âm SRF05 12](#_Toc485132954)

[2.1.4. Module thu phát hồng ngoại HX1838 13](#_Toc485132955)

[2.2. Sử dụng những IDE nào để lập trình, công dụng, cách sử dụng 15](#_Toc485132956)

[2.2.1. Arduino IDE 15](#_Toc485132957)

[2.1.1. Processing Arduino IDE 15](#_Toc485132958)

[2.3. Thư viện sử dụng 15](#_Toc485132959)

[2.1.1. IRremote 15](#_Toc485132960)

[2.1.2. Adafruit Motor Shield V2 15](#_Toc485132961)

[2.1.3. NewPing 16](#_Toc485132962)

[2.4. Cách thực hiện: 17](#_Toc485132963)

[2.4.1. Hello world 17](#_Toc485132964)

[2.4.2. Điều khiển led bằng remote 18](#_Toc485132965)

[2.4.3. Điều khiển động cơ xe bằng remote 20](#_Toc485132966)

[2.4.4. Làm radar dò vật cản 21](#_Toc485132967)

[2.4.5. Xe tự chạy 24](#_Toc485132968)

[PHẦN 3: TÀI LIỆU THAM KHẢO 26](#_Toc485132969)

# **PHẦN 1: INTERNET OF THINGS (GIỚI THIỆU, ĐỊNH NGHĨA, ỨNG DỤNG CỦA IOT,… TỰ THÊM VÀO)**

# **PHẦN 2: ARDUINO**

1. **Giới thiệu các mạch, cảm biến và module sử dụng**
   1. **Arduino UNO R3**

**Giới thiệu**

Arduino board có rất nhiều phiên bản với hiệu năng và mục đích sử dụng khác nhau như: Arduino Mega, Aruino LilyPad... Trong số đó, Arduino Uno R3 là một trong những phiên bản được sử dụng rộng rãi nhất bởi chi phí và tính linh động của nó.

**Hình ảnh**



**Thông số kỹ thuật**

|  |  |
| --- | --- |
| Vi điều khiển | ATmega328 họ 8bit |
| Điện áp hoạt động | 5V DC (chỉ được cấp qua cổng USB) |
| Tần số hoạt động | 16 MHz |
| Dòng tiêu thụ khoảng | 30mA |
| Điện áp vào khuyên dùng | 7-12V DC |
| Điện áp vào giới hạn | 6-20V DC |
| Số chân Digital I/O | 14 (6 chân hardware PWM) |
| Số chân Analog | 6 (độ phân giải 10bit) |
| Dòng tối đa trên mỗi chân I/O | 30 mA |
| Dòng ra tối đa (5V) | 500 mA |
| Dòng ra tối đa (3.3V) | 50 mA |
| Bộ nhớ flash | 32 KB (ATmega328) với 0.5KB dùng bởi bootloader |
| SRAM | 2 KB (ATmega328) |
| EEPROM | 1 KB (ATmega328) |

Arduino UNO có thể sử dụng 3 vi điều khiển họ 8bit AVR là ATmega8, ATmega168, ATmega328. Bộ não này có thể xử lí những tác vụ đơn giản như điều khiển đèn LED nhấp nháy, xử lí tín hiệu cho xe điều khiển từ xa, làm một trạm đo nhiệt độ - độ ẩm và hiển thị lên màn hình LCD,…

**Năng lượng**

Arduino UNO có thể được cấp nguồn 5V thông qua cổng USB hoặc cấp nguồn ngoài với điện áp khuyên dùng là 7-12V DC và giới hạn là 6-20V. Thường thì cấp nguồn bằng pin vuông 9V là hợp lí nhất nếu bạn không có sẵn nguồn từ cổng USB. Nếu cấp nguồn vượt quá ngưỡng giới hạn trên, bạn sẽ làm hỏng Arduino UNO.

**Các chân năng lượng**

* GND (Ground): cực âm của nguồn điện cấp cho Arduino UNO. Khi bạn dùng các thiết bị sử dụng những nguồn điện riêng biệt thì những chân này phải được nối với nhau.
* 5V: cấp điện áp 5V đầu ra. Dòng tối đa cho phép ở chân này là 500mA.
* 3.3V: cấp điện áp 3.3V đầu ra. Dòng tối đa cho phép ở chân này là 50mA.
* Vin (Voltage Input): để cấp nguồn ngoài cho Arduino UNO, bạn nối cực dương của nguồn với chân này và cực âm của nguồn với chân GND.
* IOREF: điện áp hoạt động của vi điều khiển trên Arduino UNO có thể được đo ở chân này. Và dĩ nhiên nó luôn là 5V. Mặc dù vậy bạn không được lấy nguồn 5V từ chân này để sử dụng bởi chức năng của nó không phải là cấp nguồn.
* RESET: việc nhấn nút Reset trên board để reset vi điều khiển tương đương với việc chân RESET được nối với GND qua 1 điện trở 10KΩ.

**Bộ nhớ**

Vi điều khiển Atmega328 tiêu chuẩn cung cấp cho người dùng:

* **32KB bộ nhớ Flash**: những đoạn lệnh bạn lập trình sẽ được lưu trữ trong bộ nhớ Flash của vi điều khiển. Thường thì sẽ có khoảng vài KB trong số này sẽ được dùng cho bootloader nhưng đừng lo, bạn hiếm khi nào cần quá 20KB bộ nhớ này đâu.
* **2KB cho SRAM** (**S**tatic **R**andom **A**ccess **M**emory): giá trị các biến bạn khai báo khi lập trình sẽ lưu ở đây. Bạn khai báo càng nhiều biến thì càng cần nhiều bộ nhớ RAM. Tuy vậy, thực sự thì cũng hiếm khi nào bộ nhớ RAM lại trở thành thứ mà bạn phải bận tâm. Khi mất điện, dữ liệu trên SRAM sẽ bị mất.
* **1KB cho EEPROM** (**E**lectrically **E**raseble **P**rogrammable **R**ead **O**nly **M**emory): đây giống như một chiếc ổ cứng mini – nơi bạn có thể đọc và ghi dữ liệu của mình vào đây mà không phải lo bị mất khi cúp điện giống như dữ liệu trên SRAM.

**Các cổng vào ra**

Arduino UNO có 14 chân digital dùng để đọc hoặc xuất tín hiệu. Chúng chỉ có 2 mức điện áp là 0V và 5V với dòng vào/ra tối đa trên mỗi chân là 40mA. Ở mỗi chân đều có các điện trở pull-up từ được cài đặt ngay trong vi điều khiển ATmega328 (mặc định thì các điện trở này không được kết nối).

Một số chân digital có các chức năng đặc biệt như sau:

* **2 chân Serial**: 0 (RX) và 1 (TX): dùng để gửi (transmit – TX) và nhận (receive – RX) dữ liệu TTL Serial. Arduino Uno có thể giao tiếp với thiết bị khác thông qua 2 chân này. Kết nối bluetooth thường thấy nói nôm na chính là kết nối Serial không dây. Nếu không cần giao tiếp Serial, bạn không nên sử dụng 2 chân này nếu không cần thiết
* **Chân PWM (~): 3, 5, 6, 9, 10, và 11**: cho phép bạn xuất ra xung PWM với độ phân giải 8bit (giá trị từ 0 → 28-1 tương ứng với 0V → 5V) bằng hàm analogWrite(). Nói một cách đơn giản, bạn có thể điều chỉnh được điện áp ra ở chân này từ mức 0V đến 5V thay vì chỉ cố định ở mức 0V và 5V như những chân khác.
* **Chân giao tiếp SPI**: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Ngoài các chức năng thông thường, 4 chân này còn dùng để truyền phát dữ liệu bằng giao thức SPI với các thiết bị khác.
* **LED 13**: trên Arduino UNO có 1 đèn led màu cam (kí hiệu chữ L). Khi bấm nút Reset, bạn sẽ thấy đèn này nhấp nháy để báo hiệu. Nó được nối với chân số 13. Khi chân này được người dùng sử dụng, LED sẽ sáng.

Arduino UNO có 6 chân analog (A0 → A5) cung cấp độ phân giải tín hiệu 10bit (0 → 210-1) để đọc giá trị điện áp trong khoảng 0V → 5V. Với chân **AREF** trên board, bạn có thể để đưa vào điện áp tham chiếu khi sử dụng các chân analog. Tức là nếu bạn cấp điện áp 2.5V vào chân này thì bạn có thể dùng các chân analog để đo điện áp trong khoảng từ 0V → 2.5V với độ phân giải vẫn là 10bit.

Đặc biệt, Arduino UNO có 2 chân A4 (SDA) và A5 (SCL) hỗ trợ giao tiếp I2C/TWI với các thiết bị khác.

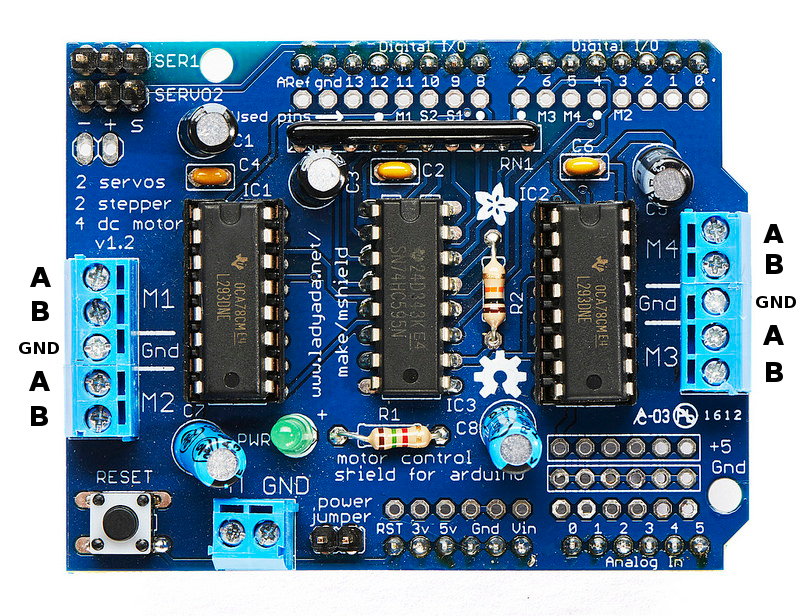
* 1. **Arduino Motor Shield L293D**

**Giới thiệu**

Motor Driver Shield L293D (Shield) là một trong các module mở rộng dành cho Arduino. Tương thích tốt với board Arduino Uno R3, Arduino Leonardo R3, Arduino Mega 2560.

Shield sử dụng 2 IC cầu H L293D và 1 IC logic 74HC595 điều khiển. Shield có thể điều khiển nhiều loại motor (servo, stepper, dc motor) khác nhau với mức áp đến 36V, dòng tối đa 600mA mỗi kênh điều khiển.

**Hình ảnh**



**Thông số kỹ thuật và khả năng điều khiển của Shield:**

* Hoạt động DC 6V-35V cấp cho động cơ.
* Dùng IC chuyên dụng L293D.
* Dòng điện 0.6A cho mỗi ngõ ra.
* 2 động cơ servo.
* 4 động cơ một chiều (M1, M2, M3, M4) độc lập thông qua 4 cầu H của L293D dòng 600mA (cực đại 1.2A).
* 2 động cơ bước loại đơn cực (unipolar) hoặc lưỡng cực (bipolar).
* Chống tự quay khi cấp nguồn (có sẵn các điện trở nối GND).
* Có sẵn nút RESET để khởi động lại board Arduino.

**Các chân trên Shield được kết nối với board Arduino như sau:**

* Servo Motor: 2 dây điều khiển 2 servo kết nối với chân số 9 và 10. Nguồn nuôi lấy trực tiếp từ board Arduino.
* Động cơ một chiều (DC):
  + Motor 1 nối với chân 11
  + Motor 2 nối với chân 3
  + Motor 3 nối với chân 5
  + Motor 4 nối với chân 6
* Động cơ bước: Chân 4, 7, 8, 12 dùng điều khiển motor thông qua IC 74HC595.

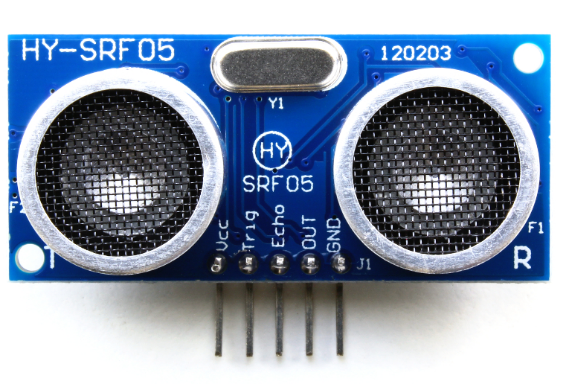
Trên shield có 1 jumper màu vàng PWR được sử dụng nếu kết nối nguồn ngoài cho board Arduino thông qua jack DC (dùng pin 9V chẳng hạn) thì nguồn nuôi motor được lấy luôn từ jack này, không cần nối nguồn với EXT\_PWR. Nếu ngắt jumper, bạn cần nối 1 nguồn riêng vào terminal EXT\_PWR để nuôi motor.

* 1. **Cảm biến siêu âm SRF05**

**Giới thiệu**

Cảm biến SRF05 là phiên bản nâng cấp của SRF04, cảm biến dùng để nhận biết khoảng cách đến vật cản nhờ sóng siêu âm phát ra từ cảm biến và nhận về.

**Hình ảnh**



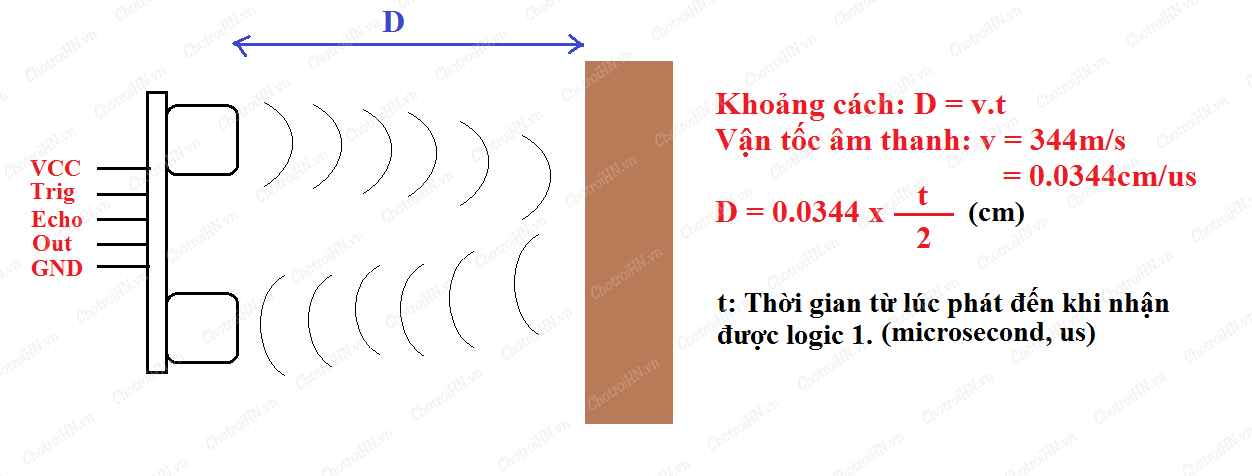
**Thông số kỹ thuật**

* SRF05 có 5 chân, bao gồm: VCC, Trigger, Echo, Out, và GND.
* Điện áp hoạt động: 5VDC
* Độ chính xác: 0.5cm

**Chức năng của các chân:**

* VCC: Chân cấp nguồn 5V.
* GND: Chân nối mass(0V).
* Trigger: Chân kích phát sóng siêu âm (ở chế độ 1). Ở chế độ 2 thì chân Trigger vừa là chân kích vừa là chân báotín hiệu phản xạ của sóng siêu âm.
* Echo: Chân báo có tín hiệu phản xạ của sóng siêu âm.
* Out: Không dùng (Not Connect).

**Nguyên lý hoạt động**



Để đo khoảng cách, ta sẽ phát 1 xung (1000 microSeconds - us) từ chân **Trig**. Sau đó, cảm biến sẽ tạo ra 1 xung HIGH ở chân **Echo** cho đến khi nhận lại được sóng phản xạ ở pin này. Chiều rộng của xung sẽ bằng với thời gian sóng siêu âm được phát từ cảm biển và quay trở lại.

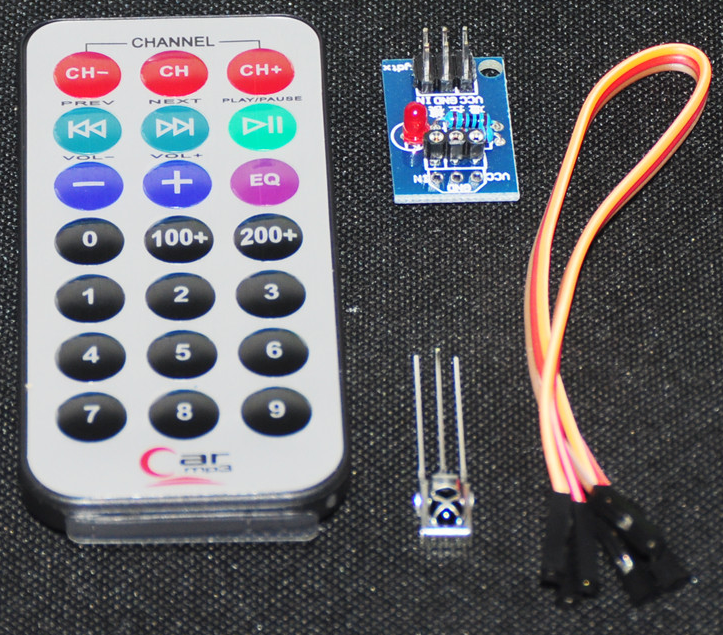
Tốc độ của âm thanh trong không khí khoảng 344 m/s = 0.0344 cm/us. Khi đã tính được thời gian, ta sẽ tính khoảng cách D theo công thức bên trên.

* 1. **Module thu phát hồng ngoại HX1838**

**Giới thiệu**

Remote điều khiển hồng ngoại được thiết kế nhỏ gọn và dễ dang kết nối với vi điều khiển. Khoảng cách hoạt động khoảng 5-8 mét, nhưng các ứng dụng thực tế phụ thuộc vào các yếu tố môi trường xung quanh (chẳng hạn như một màng ngăn, kính, hoặc chướng ngại vật khác bị chặn), bộ thu sử dụng HX1838, có thể nhận được bất kỳ mã kiểm soát 38Khz tần số dữ liệu từ xa.

**Hình ảnh**



**Thông số kỹ thuật**

* Cảm biến thu hồng ngoại: HX1838
* Điện áp hoạt động: DC 3.3 ~ 5V
* Tín hiệu ngõ ra: Digital
* Khoảng cách thu tín hiệu: 5 ~ 8m
* Tích hợp 1 Remote theo module
* Kích thước module: 0.94 in x 0.67 in x 0.47 in (2.4 cm x 1.7 cm x 1.2 cm)
* Khối lượng: 0.18 oz (5g)

**Chức năng các chân**

* Vcc -5V
* GND - GND
* D0 - Ngõ ra digital

1. **Sử dụng những IDE nào để lập trình, công dụng, cách sử dụng**
   1. **Arduino IDE**

a

* 1. **Processing Arduino IDE**

a

1. **Thư viện sử dụng**
   1. **IRremote**

**Mục đích sử dụng:**

Sử dụng tín hiệu thu được từ module thu hồng ngoại và phân tích thành mã HEX. Dựa vào mã HEX để điều khiển theo ý muốn của lập trình viên.

**Keywords & phương thức:**

* IRrecv irrecv(receiverPin);: tạo đối tượng IRrecv mới có tên là irrecv sử dụng tham số là receiverPin.
* decode\_results results; : lưu kết quả giải mã được.
* irrecv.enableIRIn(); : bắt đầu giải mã tín hiệu IR.
* irrecv.decode(&results);: trả về true nếu có tín hiệu đến.
* irrecv.resume; : đợi tín hiệu IR tiếp theo.

**Download tại:**

<https://github.com/z3t0/Arduino-IRremote>

* 1. **Adafruit Motor Shield V2**

**Mục đích sử dụng:**

Điều khiển các loại motor: motor DC, servo, step motor.

**Download tại:**

<https://github.com/adafruit/Adafruit_Motor_Shield_V2_Library>

* 1. **NewPing**

**Mục đích sử dụng:**

Tính khoảng cách từ vật thể đến cảm biến. Thư viện NewPing này cho phép ping cảm biến luôn và đáng tin cậy ở tốc độ 30 lần / giây.

**Phương thức:**

[max\_cm\_distance] là khoảng cách tối đa mới.

* sonar.ping([max\_cm\_distance]) - Gửi một ping và nhận được echo thời gian (trong micro giây) dưới dạng một kết quả.
* sonar.ping\_in([max\_cm\_distance]) – Tương tự phương thức trên nhưng kết quả trả về là in.
* sonar.ping\_cm([max\_cm\_distance]) - Tương tự phương thức trên nhưng kết quả trả về là cm.
* sonar.ping\_median(iterations [, max\_cm\_distance]) - Thực hiện nhiều lệnh ping (mặc định = 5), loại bỏ ping phạm vi và trung bình trở lại trong micro giây.
* sonar.convert\_in(echoTime) - Convert echoTime từ micro giây sang in.
* sonar.convert\_cm(echoTime) - Convert echoTime từ micro giây sang cm.
* sonar.ping\_timer(function [, max\_cm\_distance]) - Gửi hàm ping và call để kiểm tra nếu ping đã hoàn thành.
* sonar.check\_timer() - Kiểm tra xem ping đã trở lại trong giới hạn khoảng cách thiết lập.
* NewPing::timer\_ms(frequency, function) - Gọi hàm liên tục với tần số micro giây.
* NewPing::timer\_stop() – Dừng timer.

**Download tại:**

<https://github.com/PaulStoffregen/NewPing>

1. **Cách thực hiện:**
   1. **Hello world**

**Mục đích:**

Tìm hiểu và lập trình trên arduino bài đầu tiên. Làm đèn led trên Arduino UNO R3 nhấp nháy theo chu kỳ 1 giây sáng – 1 giây tắt.

**Phần cứng dụng:**

* Mạch Arduino UNO R3.
* Cáp USB type C.

**Cách thực hiện:**

Gắn cáp kết nối Arduino UNO R3 với máy tính qua cổng USB type C.

**Code:**

int led = 13; // chân digital 13 cần được kết nối với đèn LED

void setup()// Hàm setup chạy một lần duy nhất khi khởi động chương trình

{

pinMode(led, OUTPUT);

}

void loop() // Hàm loop chạy mãi mãi sau khi kết thúc hàm setup()

{

digitalWrite(led, HIGH); // bật đèn led sáng

delay(1000); // dừng trong 1 giây => thây đèn sáng được 1 giây

digitalWrite(led, LOW); // tắt đèn led

delay(1000); // dừng trong 1 giây => thấy đèn tối được 1 giây

}

* 1. **Điều khiển led bằng remote**

**Mục đích:**

Phát triển bài code phía trên, nhưng lúc này bật tắt đèn led theo ý muốn. Khi bấm vào điều khiển phím 1 thì đèn sẽ bật, khi bấm vào phím 2 thì đèn sẽ tắt.

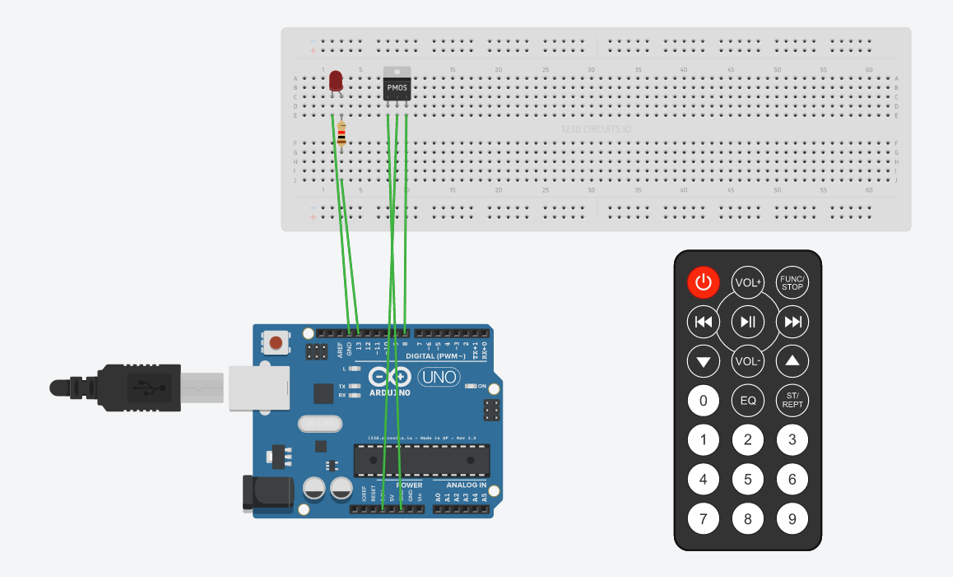
**Phần cứng dụng:**

* Mạch Arduino UNO R3.
* Cáp USB type C.
* Đèn led, điện trở.
* Remote, module thu hồng ngoại HX1838.
* Breadboard (nếu có).

**Cách thực hiện:**

* Cắm chân GRD của cảm biến hồng ngoại với chân GND của Arduino.
* Chân VCC nối với output 3,3V của Arduino.
* Chân OUT gắn chân số 8 của Arduino.
* Led gắn thêm điện trở 220ohm và đi qua chân 13 của Arduino. Chân còn lại cắm vào GND.

Chi tiết mạch như hình dưới:



**Code:**

#include <IRremote.h> // thư viện hỗ trợ IR remote

const int receiverPin = 8; // chân digital 8 dùng để đọc tín hiệu

IRrecv irrecv(receiverPin); // tạo đối tượng IRrecv mới

decode\_results results;// lưu giữ kết quả giải mã tín hiệu

const int LED1 = 13;

void setup()

{

Serial.begin(9600);// serial

irrecv.enableIRIn(); // start the IR receiver

pinMode(LED1, OUTPUT); //Tất cả các chân LED là OUTPUT

digitalWrite(LED1, LOW); //Mặc định các đèn LED tắt

}

// translate IR signals

void translateIR()

{

switch (results.value)

{

case 0xFF6897://phim so 1

digitalWrite(LED1, HIGH);

break;

case 0xFF9867://phim so 2

digitalWrite(LED1, LOW);

break;

}

}

void loop()

{

if (irrecv.decode(&results)) // nếu nhận được tín hiệu

{

translateIR();

Serial.println(results.value, HEX);

delay(200);

irrecv.resume(); // nhận giá trị tiếp theo

}

}

* 1. **Điều khiển động cơ xe bằng remote**

Mục đích:

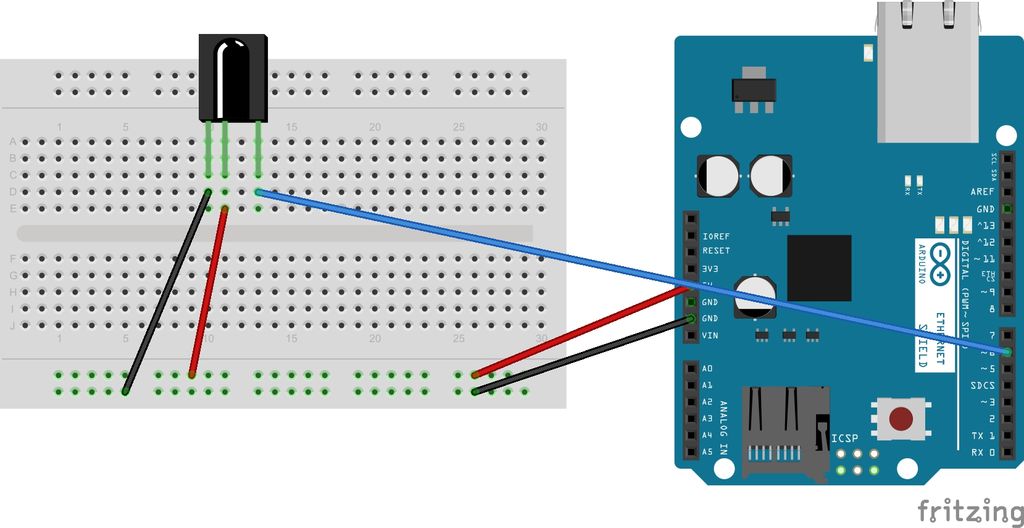
Tạo mô hình xe và điều khiển bằng remote.

**Phần cứng dụng:**

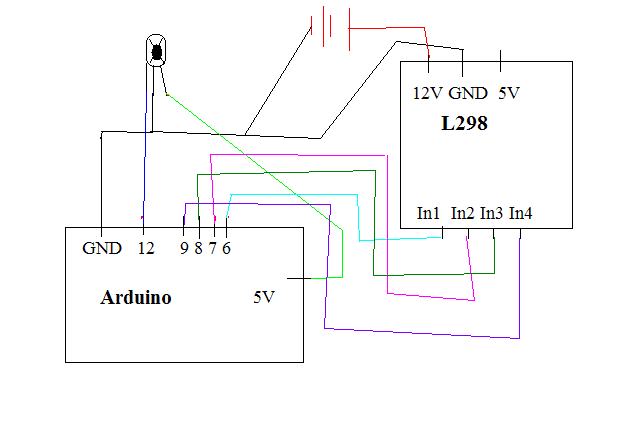
* Mạch Arduino UNO R3.
* Cáp USB type C.
* Remote, module thu hồng ngoại HX1838.
* Motor Shield L293D.
* 4 hộp số motor DC (loại vàng) cùng với 4 gá motor DC + 2 tấm khung.
* Nguồn cấp cho Arduino và Motor Shield: Pin Samsung 2230mAh (3 pin).

**Cách thực hiện:**

Sơ đồ kết nối remote:



Tiếp tục lắp ráp theo sơ đồ sau:



**Code:**

<https://1drv.ms/t/s!Alz9b1jklLHRthBOhbggGeiquBbA>

* 1. **Làm radar dò vật cản**

**Mục đích:**

Tạo Robot của mình có thể quét địa hình, phát hiện chướng ngại vật, hỗ trợ Robot di chuyển một cách đơn giản hơn khi gặp vật cản.

**Phần cứng dụng:**

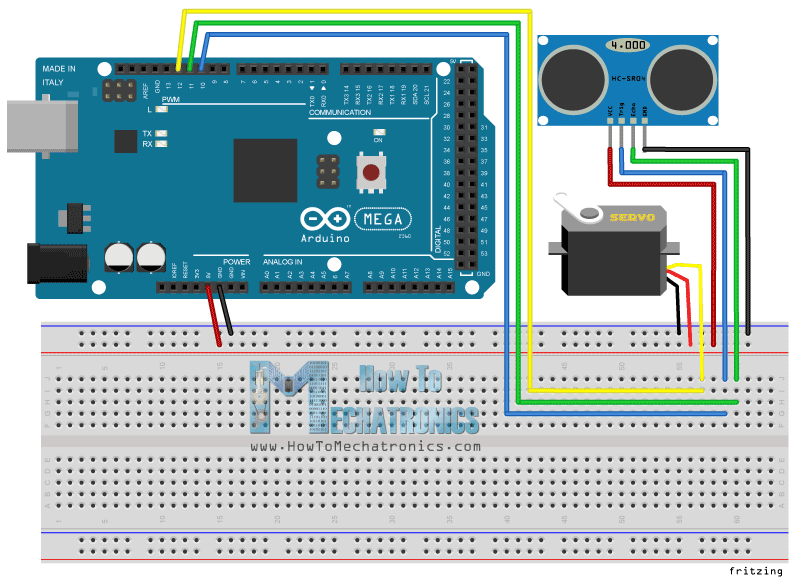
* Mạch Arduino UNO R3.
* Cáp USB type C.
* Motor Servo Tower Pro SG90.
* Cảm biến siêu âm SRF05.

**Cách thực hiện:**

Phần cứng kết nối như sau:

|  |  |  |
| --- | --- | --- |
| Chân Siêu âm | Chân Servo | Chân Arduino UNO |
| Vcc | Vcc | 5V |
| GND | GND | GND |
|  | Data | 12 |
| Trig |  | 10 |
| Echo |  | 11 |

Sơ đồ kết nối



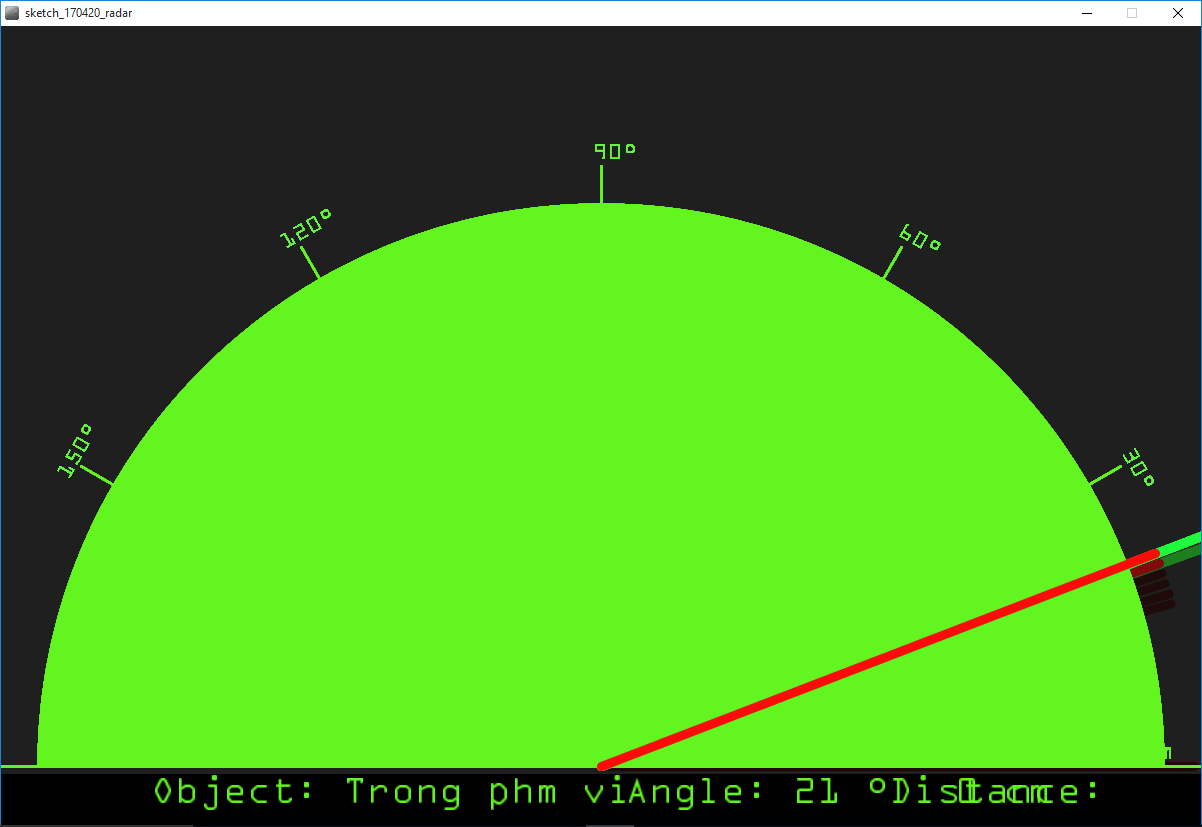
**Code Arduino:**

<https://goo.gl/j1zvRW>

**Code Processing:**

<https://goo.gl/248V8d>

**Kết quả:** (ở giao diện debug của Processing)



* 1. **Xe tự chạy**

**Mục đích:**

Tạo xe sẽ chạy tự động theo đường thẳng, khi gặp vật cản sẽ lùi lại và quét đường đi, sau đó chọn hướng nào có đường đi xa nhất.

**Phần cứng dụng:**

* Mạch Arduino UNO R3.
* Cáp USB type C.
* Motor Servo Tower Pro SG90.
* Cảm biến siêu âm SRF05.
* Motor Shield L293D.
* 4 hộp số motor DC (loại vàng) cùng với 4 gá motor DC + 2 tấm khung.
* Nguồn cấp cho Arduino và Motor Shield: Pin Samsung 2230mAh (3 pin).

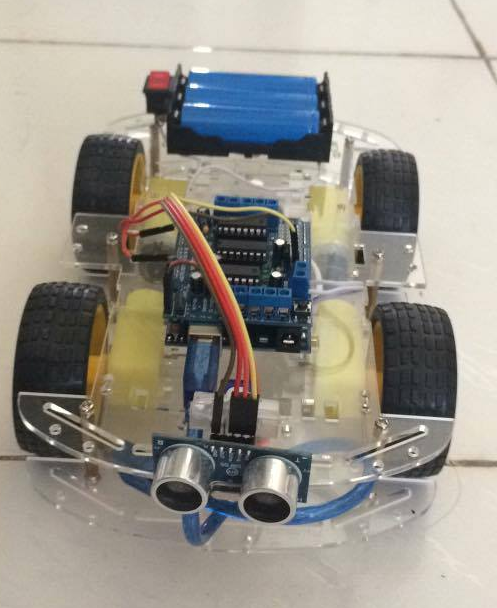
**Cách thực hiện:**

* Thư viện Adafruit Motor Shield: dùng để điều khiển motor và servo.
* Thư viện NewPing: dùng để nhận tín hiệu và đo khoảng cách từ vật đến cảm biến.
* Ghép Motor Shield L293D vào Arduino UNO R3.
* Gắn 2 motor vào vị trí M3, M4 của Motor Shield L293D.
* Gắn Servo vào vị trí Serv1 của Motor Shield L293D.
* SRF05: chân TRIG hàn vào chân 11, chân ECHO hàn vào chân 3 của Motor Shield L293D.
* Nguồn cắm vào cổng nguồn của Motor Shield L293D.

**Code:**

<https://drive.google.com/file/d/0Bw20y9iI_8beelRLMVZqeXY1N3c/>

**Hình ảnh sản phẩm:**



# **PHẦN 3: TÀI LIỆU THAM KHẢO**