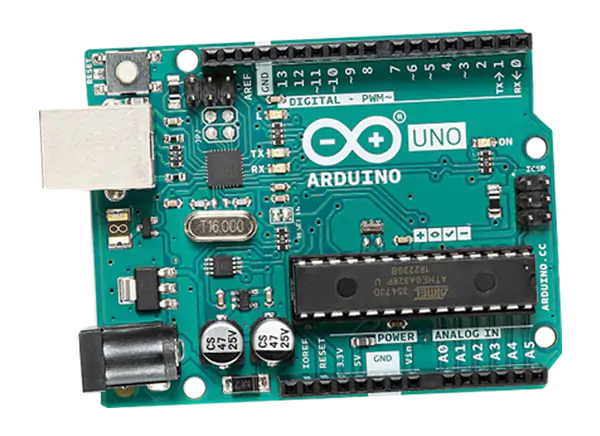
Khánh Hòa - 2021

****

BÁO CÁO HỆ THỐNG LẬP TRÌNH NHÚNG

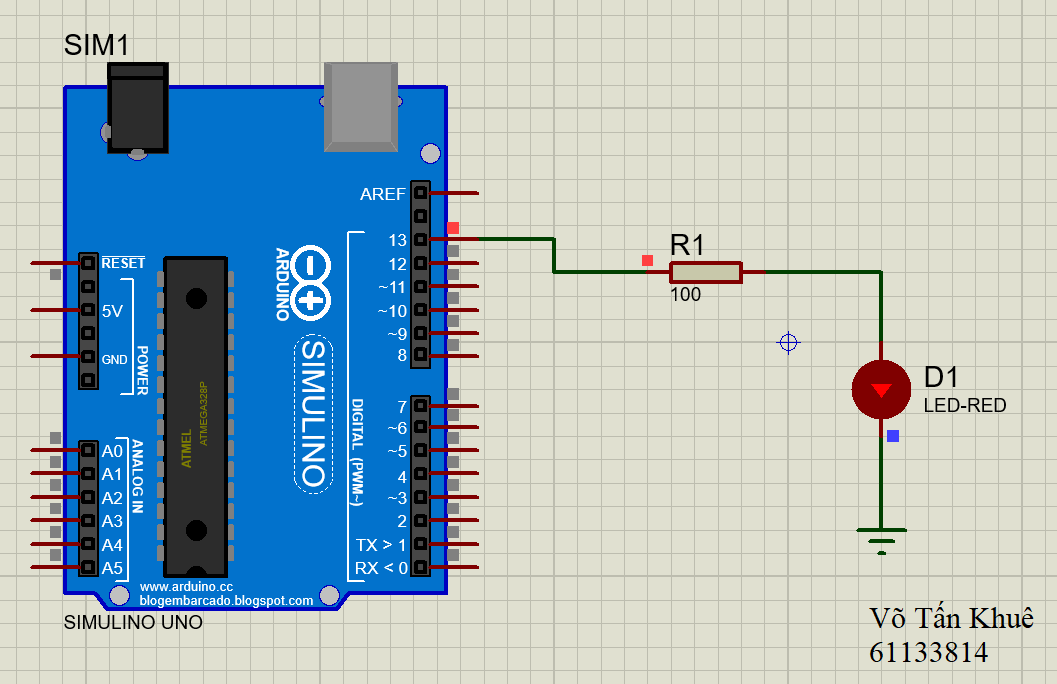
Sinh viên thực hiện: Võ Tấn Khuê Mã số sinh viên: 61133814

# Bài 1. Nháy đèn Led

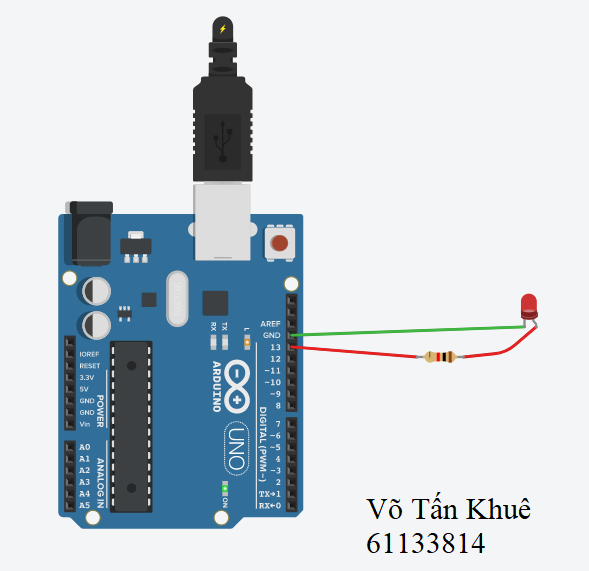
## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt đèn Led trong thời gian 1 giây, đèn Led được kết nối vào cổng số 13 của mạch Arduino.

## Sơ đồ thiết kế

****

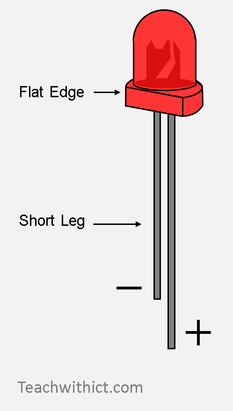
Hình 1. Sơ đồ mạch Proteus



Hình 2. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 mạch Arduino Uno
* 1 đèn Led-Red: gồm hai chân, một chân nối vào cổng Digital của Arduino, chân còn lại nối vào cổng GND (cực âm của nguồn điện cấp cho Arduino UNO thường gọi là “Tiếp đất” nghĩa là bằng 0V)**.** Đảm bảo rằng dòng điện qua đèn Led không bao giờ lớn hơn 20mA.



* 1 điện trở: 100 Ω

## Code chương trình

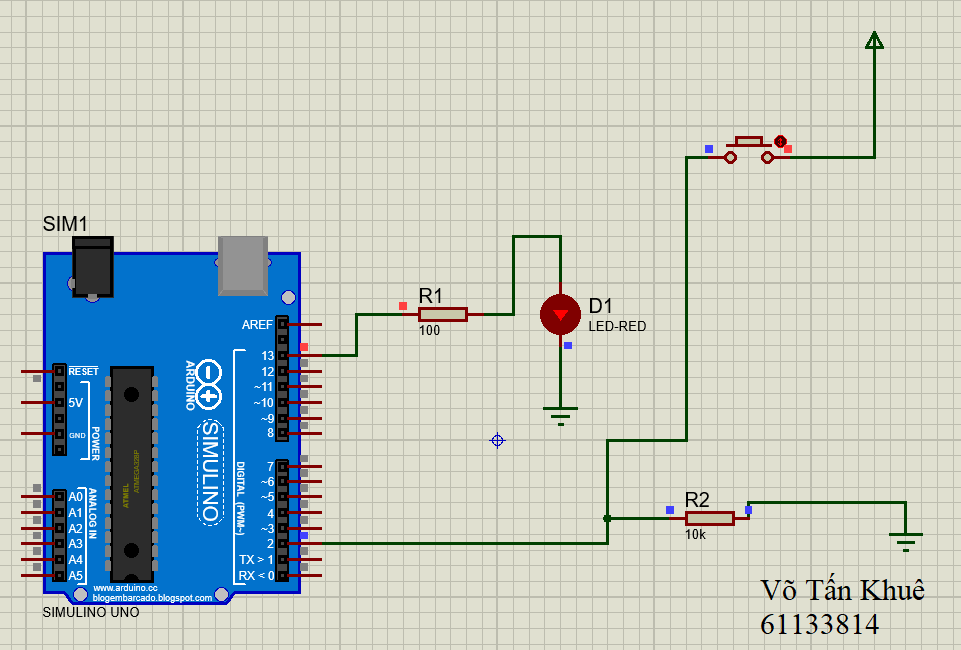
|  |
| --- |
| void **setup**() {  pinMode(13, OUTPUT); // Kết nổi với Arduino ở cổng số 13  } // Bạn có thể kết nối với Arduino ở cổng khác  void **loop**() {  digitalWrite(13, HIGH); // Bật Led  delay(1000); // Để Led sáng 1 giây  digitalWrite(13, LOW); // Tắt Led  delay(1000); // Trong 1 giây  } |

# Bài 2. Nút bấm đề bật đèn Led

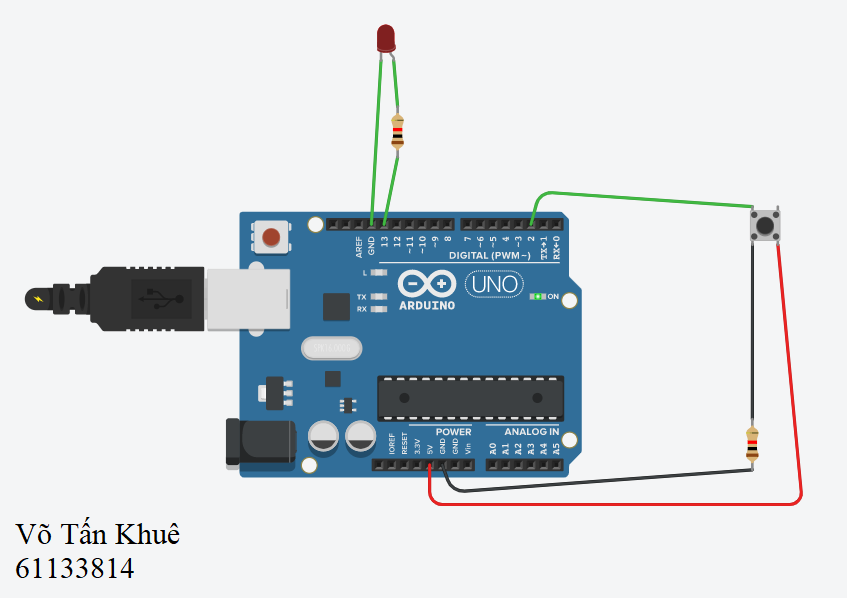
## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt đèn Led thông qua nút bấm đề, đèn Led được kết nối vào cổng số 13 của board mạch Arduino.

## Sơ đồ thiết kế

****

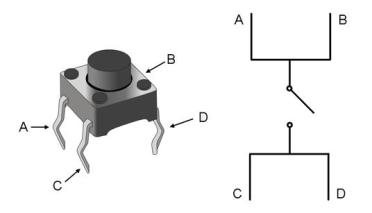
Hình 3. Sơ đồ mạch Proteus

****

Hình 4. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 đèn Led
* 1 mạch Arduino Uno
* 2 điện trở: 100 Ωnhằm hạn chế cường độ dòng điện qua đèn, giúp đèn sáng an toàn hơn.
* 1 Push Button (Nút nhấn)**:** là thiết bị đầu vào đơn giản nhất có thể được kết nối với bất kỳ bộ vi điều khiển hoặc bộ xử lý như Arduino. Một nút nhấn đơn giản nhất bao gồm bốn chân. Trong đó, các chân A và B, chân C và D được kết nối với nhau. Vì vậy, mặc dù nút nhấn có bốn chân nhưng về mặt kỹ thuật, nhưng chúng ta chỉ sử dụng hai chân.
* Điện áp hoạt động: 3.3 / 5V
* Tuổi thọ : 200.000 lần nhấn
* Nhiệt độ hoạt động: -25 ℃ đến + 70 ℃



Hình 5. Cấu tạo nút bấm

## Code chương trình

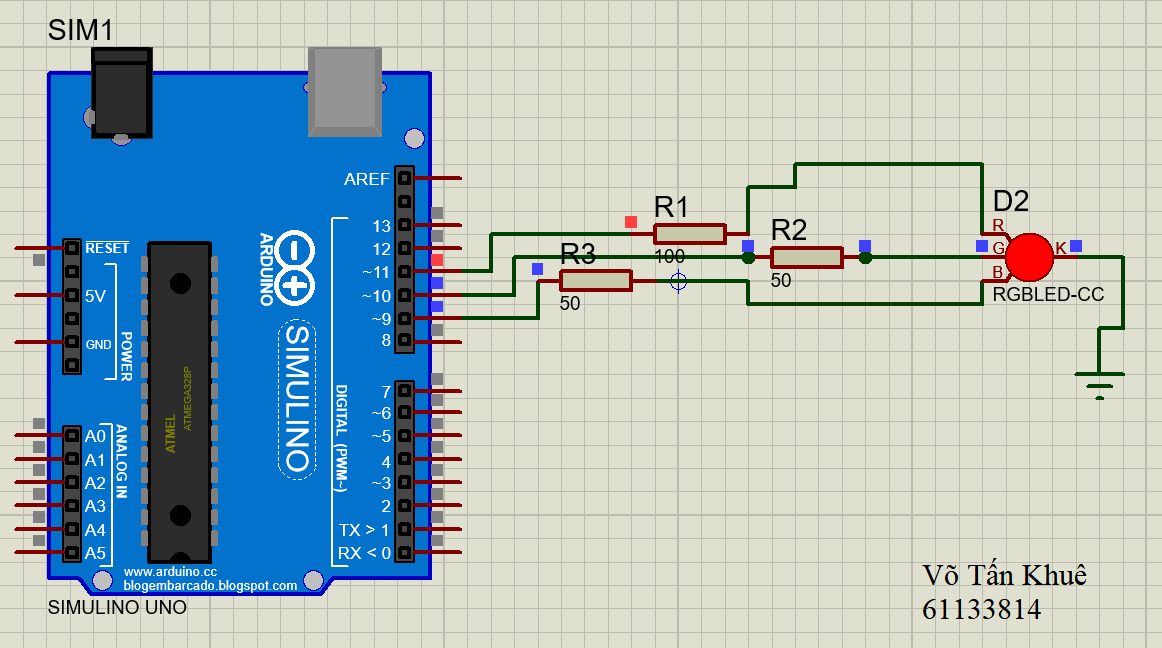
|  |
| --- |
| int x =0;  void **setup**() {  pinMode(2,INPUT);  pinMode(13,OUTPUT);  }  void **loop**() {  x = digitalRead(2); // Đọc tín hiêu ở đầu 2 và gán vào x  if(x == HIGH) {  digitalWrite(13, HIGH); // Bật đèn  } else {  digitalWrite(13, LOW); // Tắt đèn  }  delay(1000);  } |

# Bài 3. Nháy đèn Led RGB

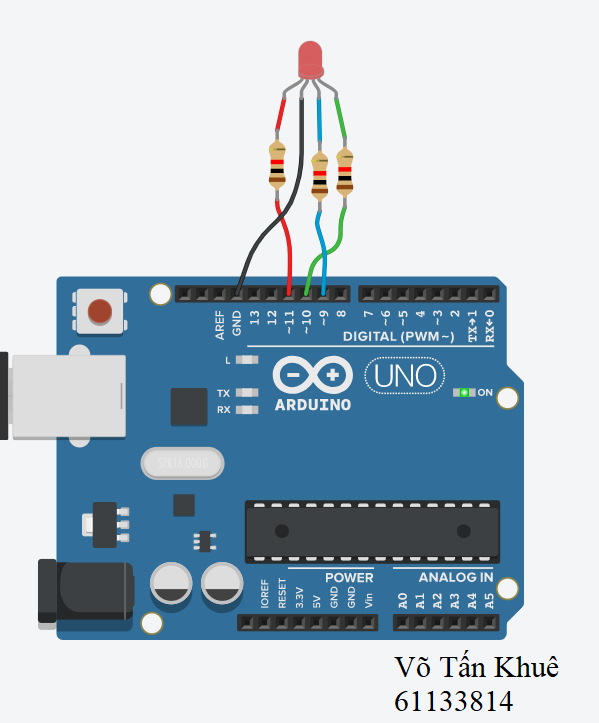
## Mô tả

Hệ thống được thiết kế điều khiển làm đèn RGB sáng ở màu đỏ, xanh lá cây, xanh dương, đèn được kết nối vào Arduino ở các cổng số 9, 10 ,11 (các chân PWM vì các cổng PWM cho phép bạn xuất ra xung PWM với độ phân giải 8bit, tức là tạo ra các giá trị từ 0 --> 28- l tương ứng với 0V --> 5V)

## Sơ đồ thiết kế



Hình 6. Sơ đồ mạch Proteus



Hình 7. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 3 điện trở: 100 Ω
* 1 Đèn RGB (Red, Green, Blue) trông giống như đèn LED thông thường, tuy nhiên, bên trong đèn thực sự có ba đèn LED, một màu đỏ, một màu xanh lá cây, một màu xanh lam. Bằng cách kiểm soát cường độ của từng đèn LED riêng lẻ, bạn có thể kết hợp khá nhiều màu sắc mà bạn muốn. Tổng số màu đèn RGB có thể sáng là 2563 màu.
* 1 mạch Arduino Uno

## Code chương trình

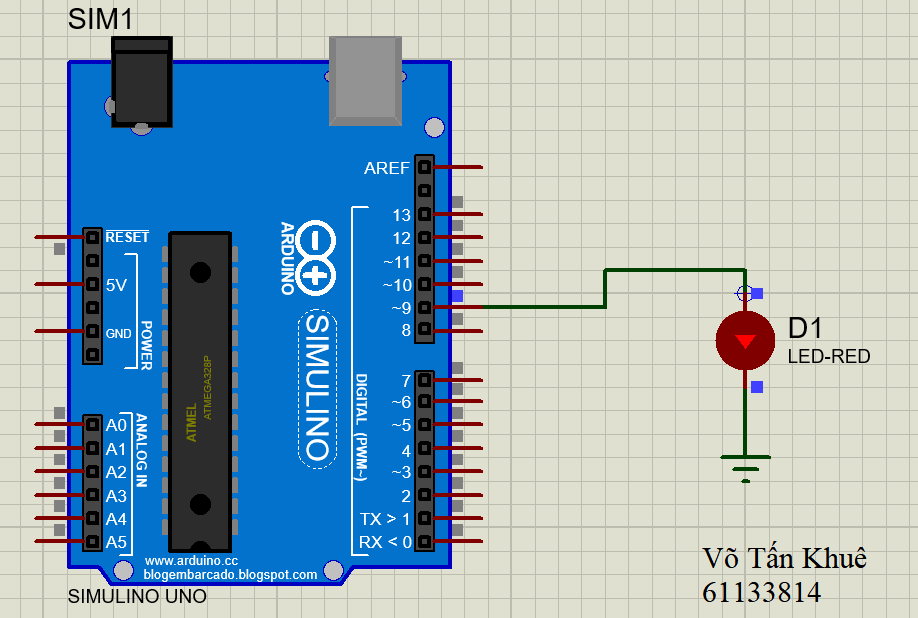
|  |
| --- |
| int denDo = 9; // Bạn có thể thay đổi các cổng khác sao cho phải là  int denXanhDuong = 10; // các cổng PWM (3, 5, 6, 9, 10, và 11)  int denXanhLa = 11;  void **setup**() {  pinMode(denDo, OUTPUT);  pinMode(denXanhDuong, OUTPUT);  pinMode(denXanhLa, OUTPUT);  }  void **loop**() {  // Bạn có thể thay đổi màu bằng cách đổi lại các giá trị cường độ dòng điện //qua các cổng  analogWrite(denDo, 0);  analogWrite(denXanhDuong, 255);  analogWrite(denXanhLa, 0);  delay(500);  analogWrite(denDo, 255);  analogWrite(denXanhDuong, 0);  analogWrite(denXanhLa, 0);  delay(500);  analogWrite(denDo, 0);  analogWrite(denXanhDuong, 0);  analogWrite(denXanhLa, 255) ;  delay(500);  } |

# Bài 4. Đèn Led sáng dần

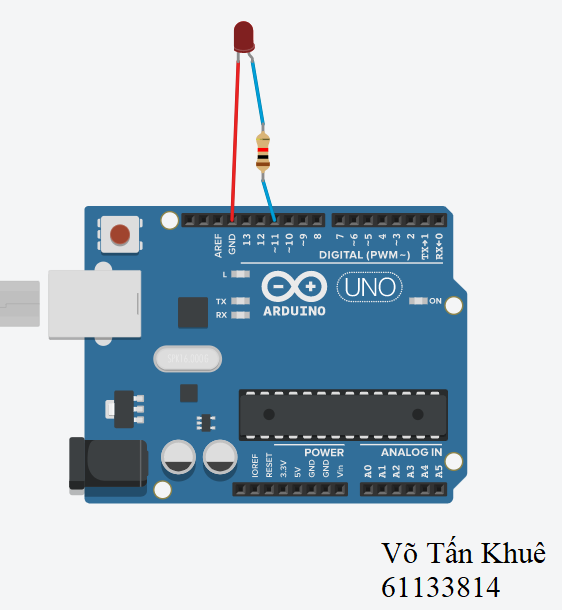
## Mô tả

Hệ thống được thiết kế đèn Led sáng dần sau đó giảm dần, đèn được kết nối vào Arduino ở cổng số 9.

## Sơ đồ thiết kế



Hình 8. Sơ đồ mạch Proteus



Hình 9. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 đèn Led
* 1 Điện trở: 100 Ω
* 1 mạch Arduino Uno

## Code chương trình

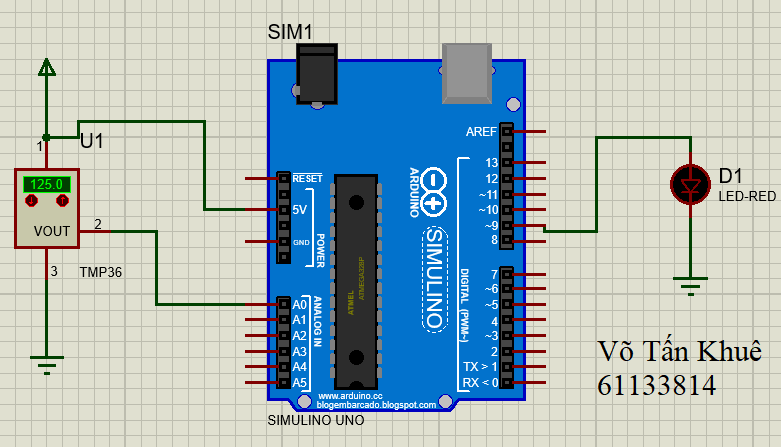
|  |
| --- |
| int i = 0;  void **setup**() {  pinMode(9,OUTPUT); // Bạn có thể sử dụng cổng PWM khác như 3, 5, // 6, 9, 10, và 11  }  void **loop**() {  for (i = 0; i<=255; i +=5) {  analogWrite(9,i);  delay(30);  }  For (i = 0; i<=255; i -=5) {  analogWrite(9,i);  delay(30);  }  } |

# Bài 5. Cảm biến nhiệt độ

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt đèn Led thông qua cảm biến nhiệt độ Tmp36, đèn Led được kết nối vào cổng 9 của mạch Arduino.

## Sơ đồ thiết kế



Hình 10. Sơ đồ mạch Proteus



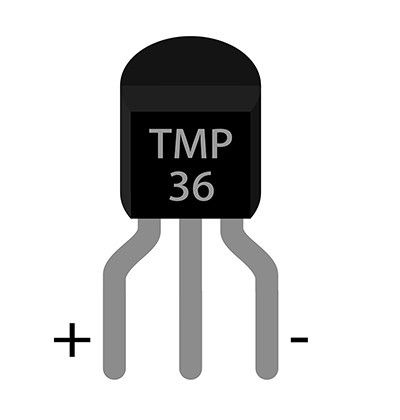
Hình 11. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 đèn Led
* 1 mạch Arduino Uno
* 1 Tmp36 là một cảm biến nhiệt độ độ chính xác, điện áp thấp do Analog Devices sản xuất. Nó là một con chip cung cấp đầu ra điện áp tỷ lệ tuyến tính với nhiệt độ tính bằng °C (từ -40oC đến 125oC). Do đó rất dễ sử dụng với Arduino.

Sơ đồ cấu tạo gồm 3 chân:

* Chân số 1 là chân cấp nguồn 5V (chân này có thể cắm vào nguồn 5V của Arduino).
* Chân thứ 2 là chân xuất tín hiệu tương tự (tín hiệu dạng xung).
* Chân thứ 3 là chân nối mát hay chân GND (khi sử dụng với Arduino các bạn có thể lấy từ chân GND từ Arduino.



Bởi vì nó xuất một tín hiệu tương tự nên khi làm việc với Arduino chúng ta sử dụng hàm **analogRead**(). Khi Arduino làm việc nó sẽ trả tín hiệu tương tự này về giá trị từ 0-1023 tùy thuộc vào điện áp từ 0-5V(5000mV). Điện áp thực tế sẽ là điện áp mà Arduino đọc được sau đó nhân với (5000/1024). Sau đó chúng ta phải trừ đi điện áp bù 400 và chia cho 10 sẽ ra kết quả là độ C.

## Code chương trình

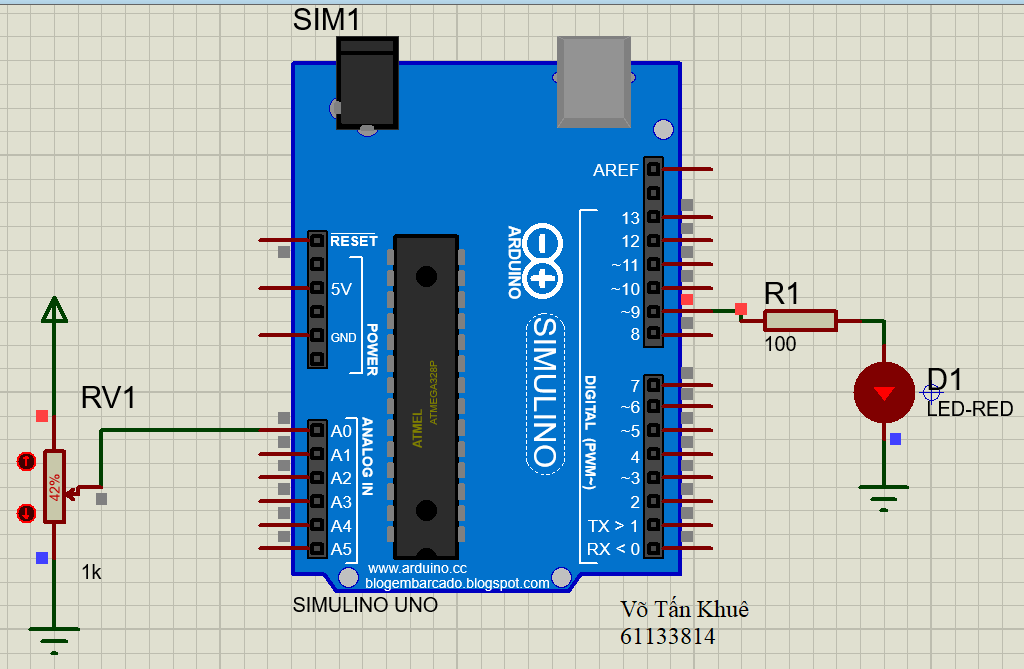
|  |
| --- |
| void **setup**()  { pinMode(9,OUTPUT); // bạn có thể thay đổi cổng khác }  void **loop**()  {  int giaTri = analogRead(A0);  int nhietDo = map (giaTri, 20,358,-40,125);  if(nhietDo > 25){  digitalWrite(9,HIGH); // Bật đèn  }  else {  digitalWrite(9,LOW); // Tắt đèn  }  delay(1000);  } |

# Bài 6. Điều khiển độ sáng của đèn bằng chiết áp

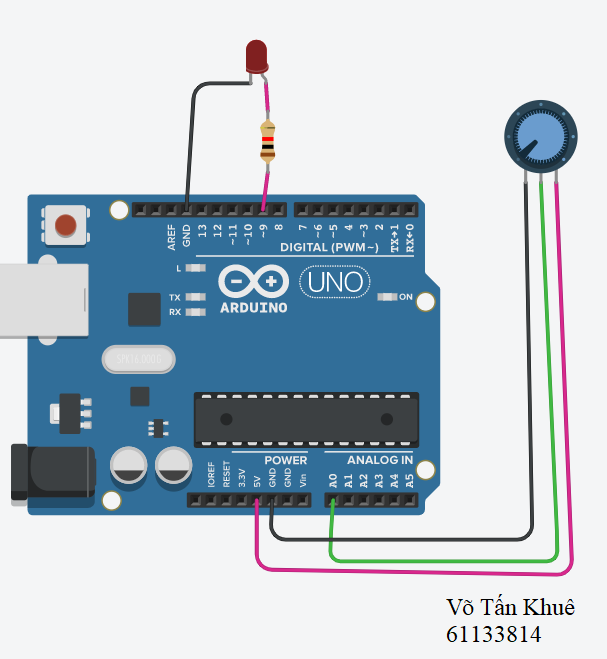
## Mô tả

Hệ thống được thiết kế điều chỉnh độ sáng của đèn bằng chiết áp, đèn được kết nối Arduno ở cổng số 9 còn chiếc áp ở cổng A0

## Sơ đồ thiết kế



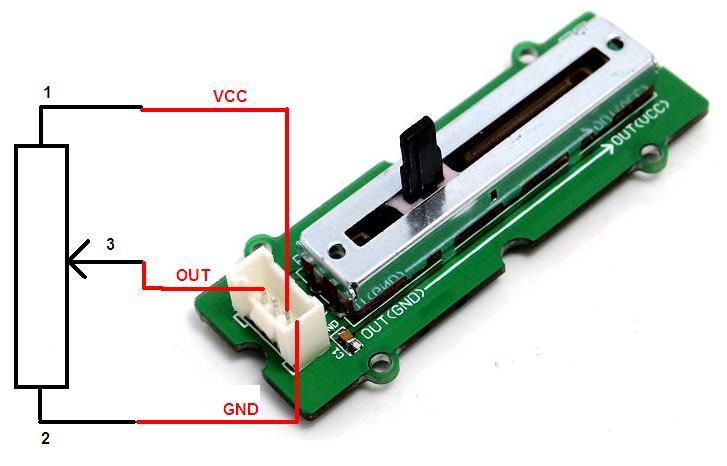
Hình 12. Sơ đồ mạch Proteus



Hình 13. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 đèn Led
* 1 điện trở: 100 Ω
* 1 chiết áp (Potentiometer) là sự kết hợp với một biến trở tuyến tính có điện trở tối đa là 10KΩ. Khi bạn di chuyển thanh trượt từ bên này sang bên kia, điện áp đầu ra của nó sẽ nằm trong khoảng từ 0 V đến Vcc mà bạn áp dụng. Cấu tạo gồm ba chân OUT, GND, VCC như trong hình bên dưới.



Hình 14. Cấu tạo Potentiometer

* 1 mạch Arduino Uno

## Code chương trình

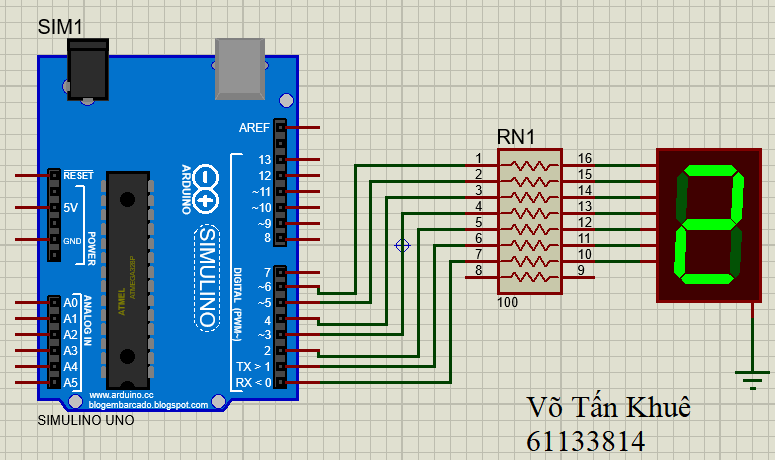
|  |
| --- |
| int x =0;  void **setup**()  {  pinMode(9, OUTPUT);  pinMode(A0, INPUT);  }  void **loop**()  {  x = analogRead(A0);  int doSang = map(x,0,1023,0,255); // Chuyển sang độ sáng  analogWrite(9,doSang);  } |

# Bài 7. Led 7 đoạn

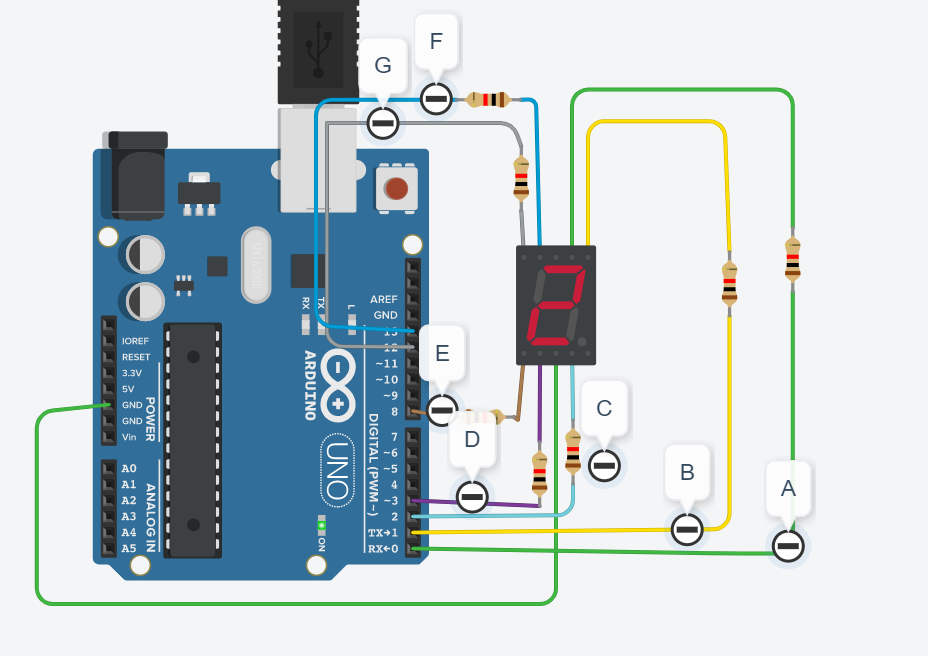
## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt đèn Led 7 đoạn trong thời gian 1 giây, đèn Led được kết nối board mạch Arduino thông qua điện trở 16 cổng RES16DIPIS.

## Sơ đồ thiết kế



Hình 15. Sơ đồ mạch Proteus



Hình 16. Sơ đồ mạch Tinkercad

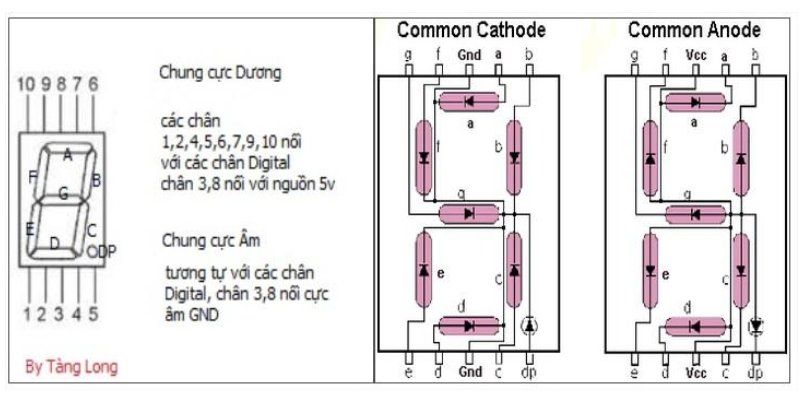
## Đặc điểm của linh kiện

* 1 đèn Led 7 đoạn

Led 7 đoạn thật chất chỉ là bảy đèn Led được kết nối song song. Mỗi đèn trong số bảy đèn Led được gọi là một đoạn vì khi được chiếu sáng, đoạn đó tạo thành một phần của chữ số (cả Thập phân và Hệ lục phân) sẽ được hiển thị. Đèn Led thứ 8 bổ sung bên cạnh vì đôi khi được sử dụng để chỉ báo dấu chấm thập phân. Mỗi đoạn led được đánh dấu từ a tới g. Đèn Led có bộ sung đèn thứ tám gọi là “chấm thập phân” (Decimal Point) ký hiệu DP được sử dụng khi hiển thị số không phải là số nguyên.

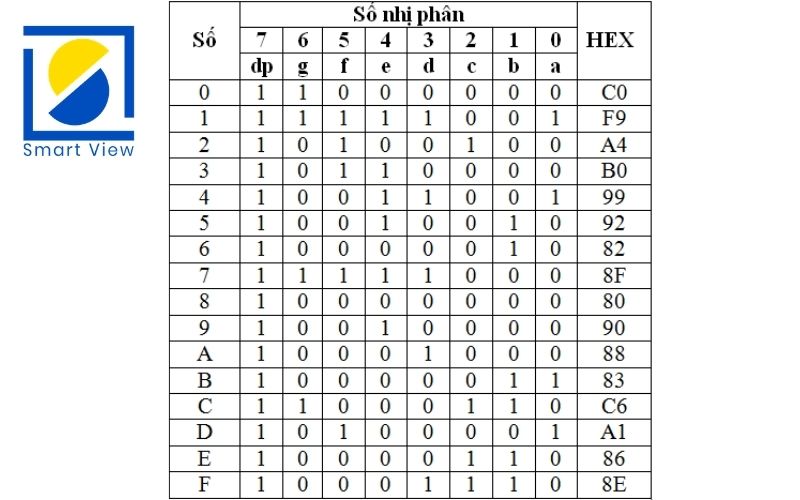
Dựa vào các cực được nối, có thể phân loại LED 7 đoạn như sau:

* Loại dương (Common Anode): nếu cực dương (anode) của tất cả 8 LED được nối với nhau và các cực âm (cathode) đứng riêng lẻ.
* Loại âm (Common Cathode): nếu cực âm (cathode) của tất cả 8 LED được nối với nhau và các cực dương (anode) đứng riêng lẻ.

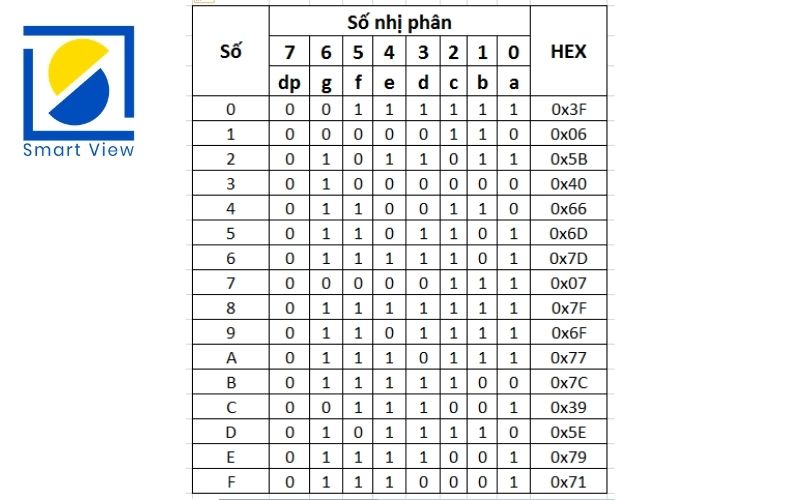


Hình 17. Cấu tạo Led 7 đoạn

Nguyên lý hoạt động: muốn Led nào sáng thì Led đó phải được phân cực thuận. Do đó muốn tạo ra chữ số nào ta chỉ cần cho Led ở các vị trí tương ứng sáng lên. Bảng mô tả cách tạo ra các chữ số để hiển thị lên LED 7 đoạn:



Hình 18. Common Anode



Hình 19. Common Cathode

* 1 mạch Arduino Uno
* 1 điện trở RES16DIPIS: 100 Ω, điện trở này có 16 cổng với các cặp tương ứng với nhau.

## Code chương trình

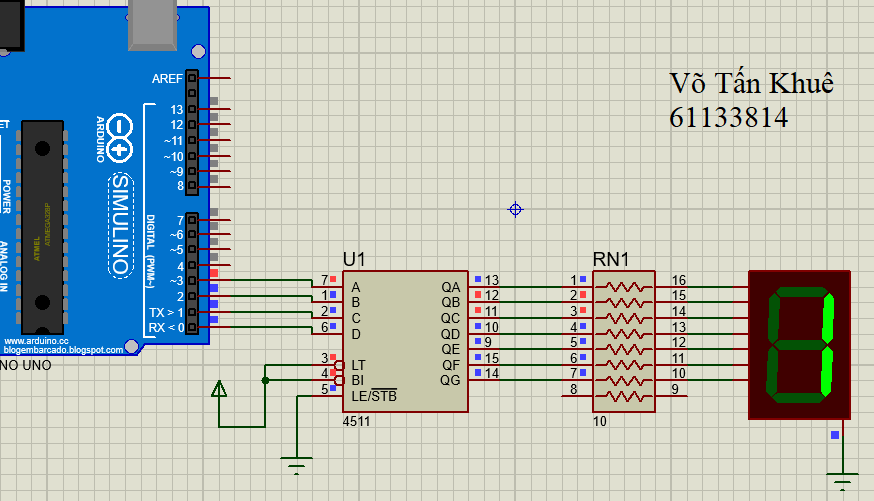
|  |
| --- |
| int a=6, b=5, c=4, d=3,e=2, f=1, g=0;  void **setup**()  {  pinMode(a, OUTPUT); pinMode(b, OUTPUT);  pinMode(c, OUTPUT); pinMode(d, OUTPUT);  pinMode(e, OUTPUT); pinMode(f, OUTPUT);  pinMode(g, OUTPUT);  }  void **loop**()  { // Hiển thị số 0  digitalWrite(a, HIGH); digitalWrite(b, HIGH);  digitalWrite(c, HIGH); digitalWrite(d, HIGH);  digitalWrite(e, HIGH); digitalWrite(f, HIGH);  digitalWrite(g, LOW);  delay(1000); // Tạm ngưng 1s  // Hiển thị số 1  digitalWrite(a, LOW); digitalWrite(b, HIGH);  digitalWrite(c, HIGH); digitalWrite(d, LOW);  digitalWrite(e, LOW); digitalWrite(f, LOW);  digitalWrite(g, LOW);  delay(1000);  // Hiển thị số 2  digitalWrite(a, HIGH); digitalWrite(b, HIGH);  digitalWrite(c, LOW); digitalWrite(d, HIGH);  digitalWrite(e, HIGH); digitalWrite(f, LOW);  digitalWrite(g, HIGH);  delay(1000);  } |

# Bài 8. Led 7 đoạn BCD

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt đèn Led 7 đoạn trong thời gian 1 giây, đèn Led được kết nối vào board mạch Arduino thông qua RES16DIPIS và IC4511

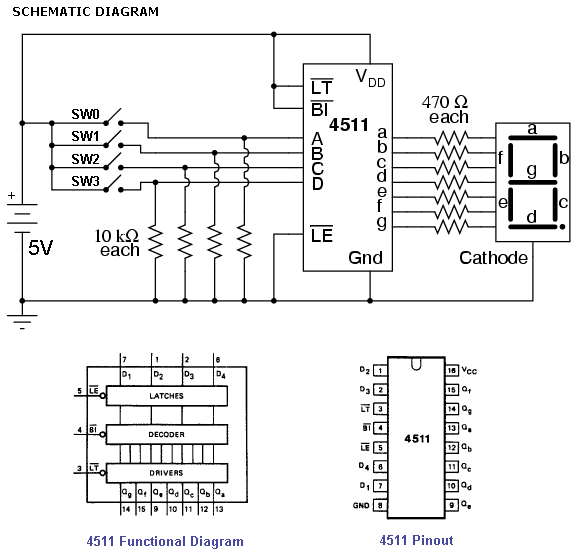
## Sơ đồ thiết kế



Hình 20. Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

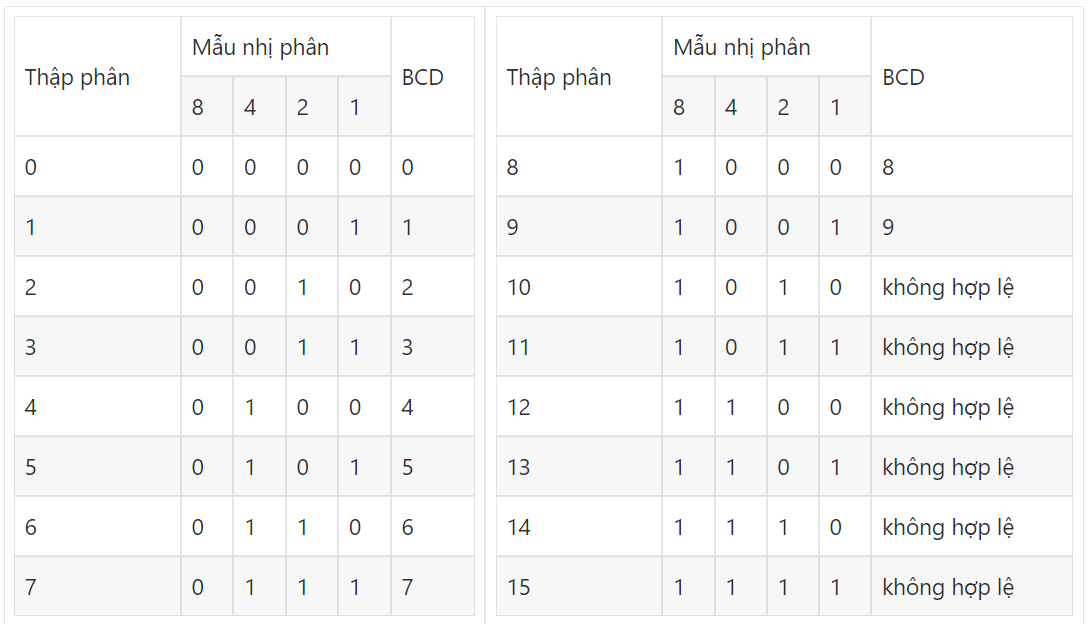
* 1 đèn led 7 đoạn
* 1 bộ điện trở 16 cổng 100 Ω
* 1 IC 4511



Hình 21. Cấu trúc và sơ đồ test - thử IC 4511

Cụ thể:

Chân 16 cấp nguồn VCC (Không được quá 5V), chân 8 là chân nối GND (mass). Các chân 1,2,6,7 là các chân tín hiệu vào ứng với B,C,D,A. Các chân 15,14,13,12,11,10,9 là các chân ra, các chân này sẽ được nối với Led 7 đoạn. Chân thứ 3 LT (Lamp test) như tên gọi của nó chân 3 này là chân kiểm tra led 7 đoạn, nếu ta cắm chân này xuống mass thì bộ giải mã sẽ sáng cùng lúc với 7 đoạn. Chân này chỉ phục vụ để kiểm tra xem có Led nào bị hỏng hay không và trong thực tế không sử dụng nó. Chân 4 BI/RB0 luôn luôn được kết nối với mức cao, nếu kết nối với mức thấp thì toàn bộ led sẽ không sáng bất chấp trạng thái ngõ vào là gì. Chân 5 RBI kết nối với mức cao.



Hình 22. Bảng chuyển đổi BCD ra số thập phân

## Code chương trình

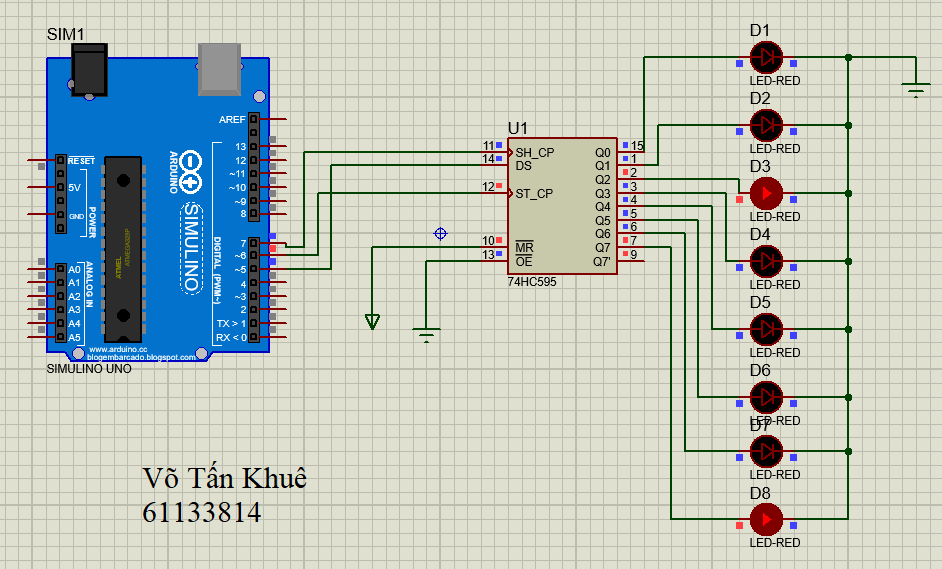
|  |
| --- |
| int a=0, b=1, c=2, d=3;  void **setup**() {  pinMode(a,OUTPUT); pinMode(b,OUTPUT);  pinMode(c,OUTPUT); pinMode(d,OUTPUT);  }  void **KHONG**(){ //Hiển thị số 0  digitalWrite(a,LOW); digitalWrite(b,LOW);  digitalWrite(c,LOW); digitalWrite(d,LOW);  }  void **MOT**(){ //Hiển thị số 1  digitalWrite(a,LOW); digitalWrite(b,LOW);  digitalWrite(c,LOW); digitalWrite(d,HIGH);  }  void **HAI**(){ //Hiển thị số 2  digitalWrite(a,LOW); digitalWrite(b,LOW);  digitalWrite(c,HIGH); digitalWrite(d,LOW);  }  void **loop**() {  KHONG(); delay(500);  MOT(); delay(500);  HAI(); delay(500);  } |

# Bài 9. Sáng 8 Led bằng IC 74HC595

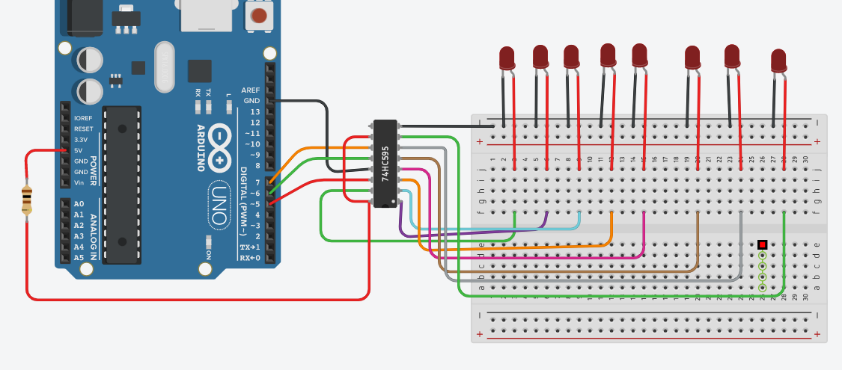
## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt đèn 8 Led bằng IC 74HC595, được kết nối với mạch Arduino qua các cổng 5,6,7.

## Sơ đồ thiết kế



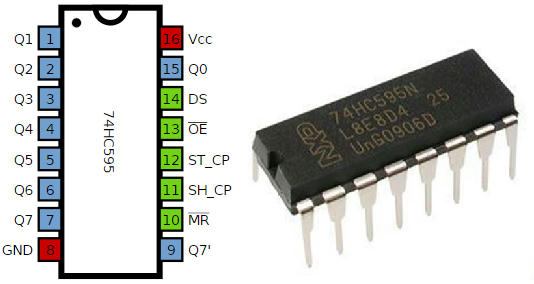
Hình 23. Sơ đồ mạch Proteus



Hình 24. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 8 đèn Led
* 1 mạch IC 74HC595 là IC ghi dịch (shift register) 8 bit kết hợp chốt dữ liệu, đầu vào nối tiếp, đầu ra song song. IC này thường dùng trong các mạch quét Led 7, Led ma trận… Để tiết kiệm số chân cho các vi điều khiển. Có thể mở rộng số chân vi điều khiển bao nhiêu tùy thích mà không IC nào có thể làm được bằng cách mắc nối tiếp ngõ vào dữ liệu các IC với nhau.



Hình 25. Sơ đồ chân IC 74HC595

* 1 mạch Arduino Uno

## Code chương trình

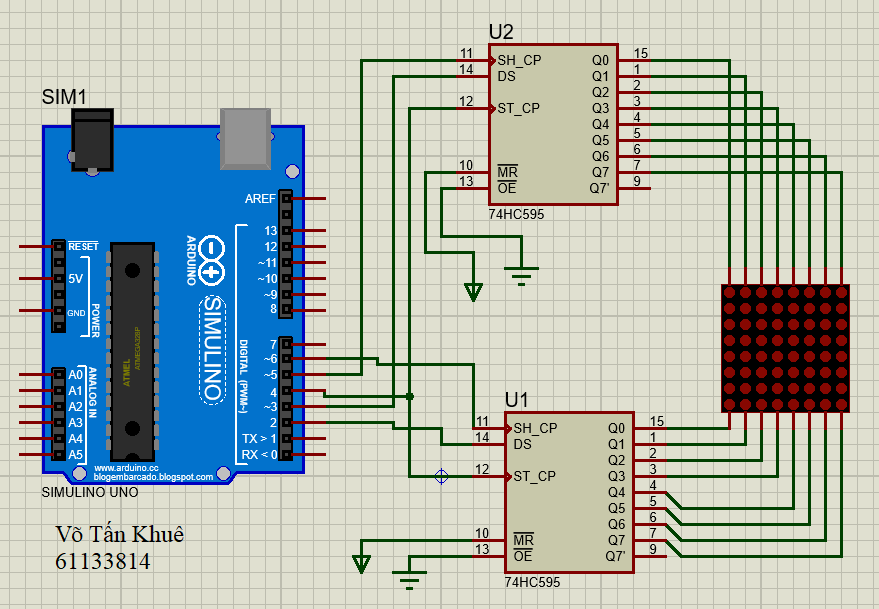
|  |
| --- |
| #define \_latch 6 // Khai báo chân latch được kết nối với chân số 6 của Arduino  #define \_clock 7 //Khai báo chân clock được kết nối với chân số 7 của Arduino  #define \_data 5 //Khai báo chân data được kết nối với chân số 5 của Arduino  void **setup**()  {  pinMode(\_latch, OUTPUT); pinMode(\_clock, OUTPUT);  pinMode(\_data, OUTPUT);  }  void **loop**()  {  for (int i=0; i< 256;i++){  digitalWrite(\_latch,LOW);  //Ta sử dụng hàm vào ra nâng cao:  //shiftOut(chân\_xuất\_data, chân\_clock, LSBFIRST, dữ\_liệu);  shiftOut(\_data, \_clock,LSBFIRST, i);  //LSBFIRST (bít thứ tự nhỏ trước) còn MSBFIRST là ngược lại  digitalWrite( \_latch,HIGH);  delay(500);  }  } |

# Bài 10. Led Matrix 8x8

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều đèn Led Matrix 8x8.

## Sơ đồ thiết kế



Hình 26. Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

* 1 mạch Arduino Uno
* 2 mạch IC 74HC595
* 1 bảng mạch Led Matrix 8x8 Red

## Code chương trình

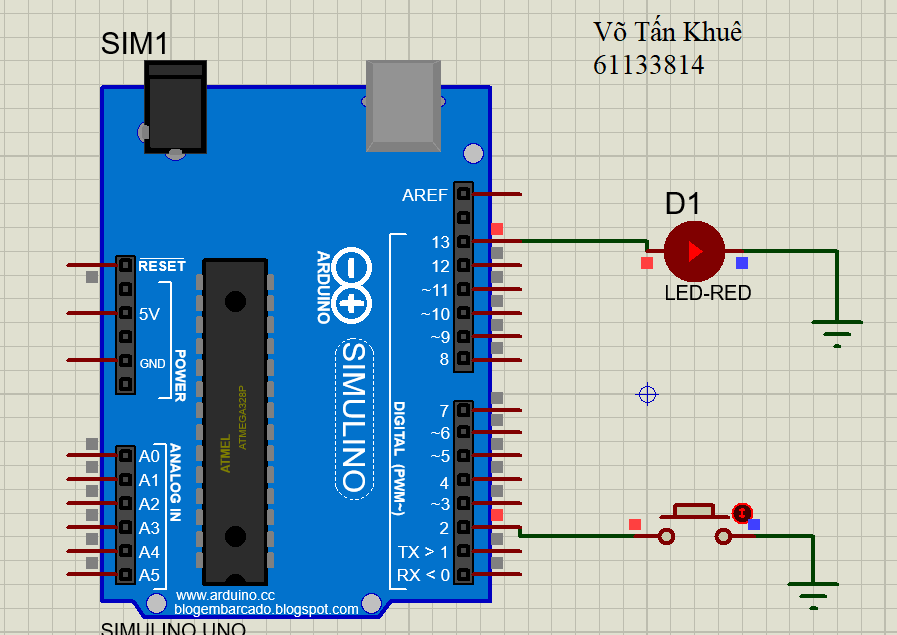
|  |
| --- |
|  |

# Bài 11. Sử dụng ngắt ngoài tắt đèn LED khi nhấn nút bấm đề

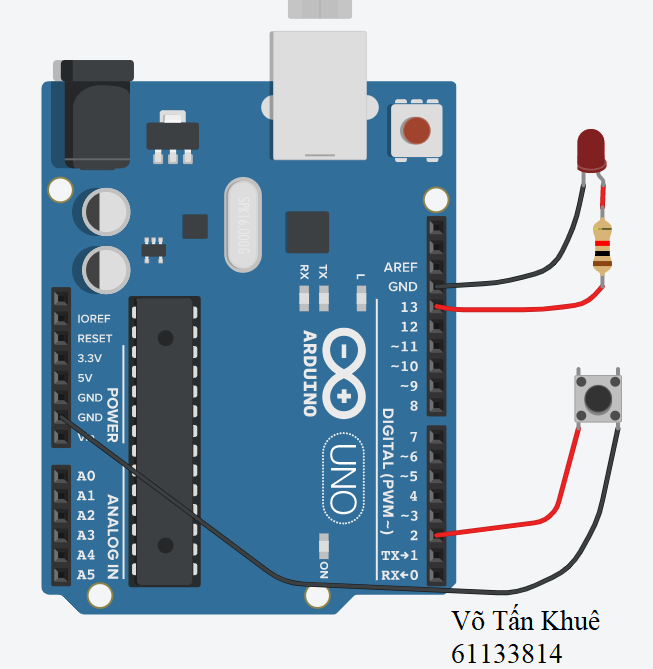
## Mô tả

Hệ thống được thiết kế thực hiện việc ngắt đèn Led đang sáng qua nút bấm đề.

## Sơ đồ thiết kế



Hình 27. Sơ đồ mạch Proteus



Hình 28. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 đèn Led
* 1 nút bấm

## Code chương trình

|  |
| --- |
| // Khai báo biến trangthai kiểu volatile để tránh sai số trong quá trình thực hiện ngắt.  volatile int trangthai = HIGH;  void **setup**()  {  pinMode(13, OUTPUT);  digitalWrite(2, HIGH); // Treo chân ngắt 0, chân số 2 lên mức cao.  attachInterrupt(0, daotrangthai, CHANGE);  }  void **loop**() // đoạn chương trình chính, vòng lặp vô tận.  {  digitalWrite(13, trangthai); // Gán trạng thái cho đen  }  void **daotrangthai**() // chương trình ngắt  {  trangthai = !trangthai; // đảo trang thái.  } |

# Bài 12. Điều khiển động cơ DC với IC L293D và Drduino

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển động cơ DC thông qua IC điều khiển động cơ L293D

Để có thể điều khiển hoàn toàn động cơ DC tức là chúng ta phải điều khiển tốc độ và chiều quay của nó. Để thực hiện diều này ta có thể kết hợp hai kỹ thuật sau. **Chế độ rộng xung (PWM)** để kiểm soát tốc độ và **Mạnh cầu H (H-Bridge)** để điều khiển hướng quay.

**PWM - Để kiểm soát tốc độ**

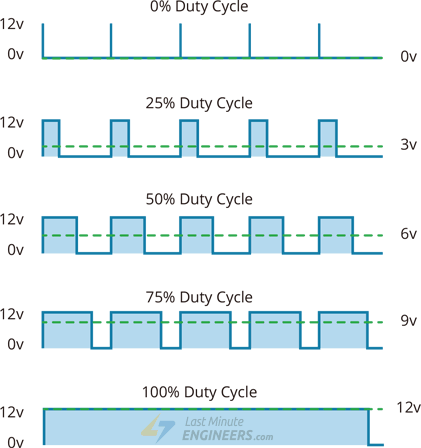
Tốc độ của động cơ DC có thể được điều khiển bằng cách thay đổi điện áp đầu vào của nó. Một kỹ thuật phổ biến để thực hiện việc này là sử dụng PWM (Pulse Width Modulation – Điều chế độ rộng xung).

PWM là một kỹ thuật trong đó giá trị trung bình của điện áp đầu vào được điều chỉnh bằng cách gửi một loạt các xung ON-OFF.

Điện áp trung bình tỷ lệ thuận với chiều rộng của các xung được gọi là chu kỳ làm việc (Duty Cycle).

Chu kỳ làm việc càng cao, điện áp trung bình cung cấp cho động cơ DC càng lớn (Động cơ quay nhanh) và chu kỳ làm việc càng thấp, điện áp trung bình cung cấp cho động cơ DC càng ít (Động cơ quay chậm).

Dưới đây hình ảnh minh họa kỹ thuật PWM với các chu kỳ làm việc khác nhau và điện áp trung bình.



Hình 29. Kỹ thuật điều chế độ rộng xung (PWM)

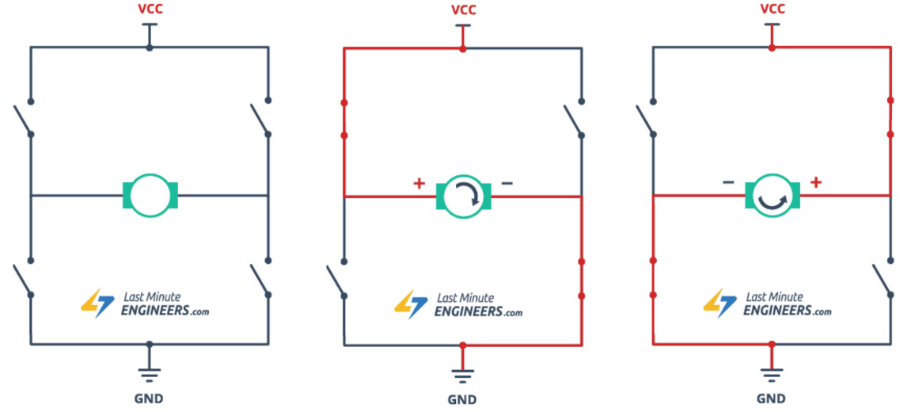
**H-Bridge - Để điều khiển hướng quay**

Chiều quay của động cơ DC có thể được điều khiển bằng cách thay đổi cực tính của điện áp đầu vào. Một kỹ thuật phổ biến để làm điều này là sử dụng mạch cầu H.

Một mạch cầu H gồm có bốn công tắc với động cơ ở trung tâm tạo thành một sắp xếp giống như hình chữ H.

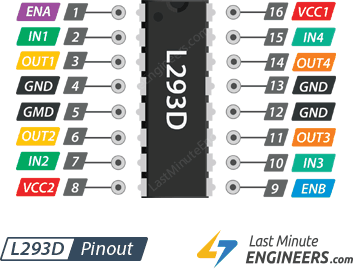
Đóng hai công tắc cụ thể cùng một lúc sẽ đảo ngược cực tính của điện áp đặt vào động cơ. Điều này gây ra sự thay đổi hướng quay của động cơ.

Hình ảnh động bên dưới minh họa mạch H-Bridge hoạt động.



Hình 30. H-Bridge hoạt động

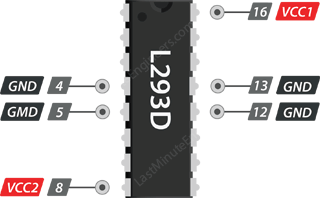
**IC điều khiển động cơ L293D**



Hình 31. Sơ đồ các chân của L293D

L293D là trình điều khiển động cơ H-Bridge có khả năng điều khiển một cặp động cơ DC cùng một lúc hoặc điều khiển hướng của hai động cơ này một cách độc lập. Điều đó có nghĩa là nó có thể truyền động riêng lẻ tối đa hai động cơ, lý tưởng cho việc xây dựng nền tảng robot hai bánh.

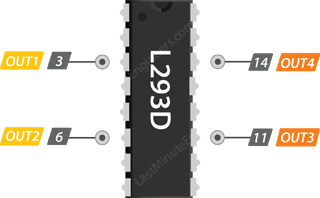
**Nguồn IC L293D**



Hình 32. Các chân nguồn và GND của L293D

IC L293D thực tế có chân nguồn đầu vào là **VCC1** và **VCC2**, trong đó **VCC1** dùng để kết nối với nguồn +5V để cho phép IC hoạt động còn **VCC2** dùng Kết nối với chân nguồn cho động cơ đang chạy (4.5V đến 36V). Cả 2 đầu này đều nối đến một điểm chung là **GND**.

**Các cổng kết nối đầu ra**



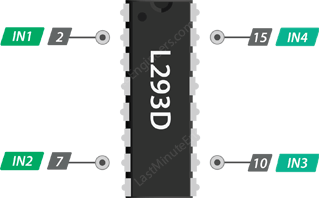
Hình 33. Các chân kết nối đầu ra của L293D

Các chân kết nối đầu ra của trình điều khiển động cơ L293D cho hai động cơ A và B được đưa ra qua các chân **OUT1**, **OUT2** và **OUT3**, **OUT4** nằm ở hai phía tương ứng. Bạn có thể kết nối hai động cơ DC có điện áp từ 4,5V đến 36V với các chân kết nối đầu ra này. Mỗi chân trên IC có thể cung cấp tới 600mA cho động cơ DC. Tuy nhiên, lượng dòng điện cug cấp cho động cơ còn phụ thuộc vào nguồn điện của hệ thống.

**Chân điều khiển**

Đối với các chân L293D, có hai loại chân điều khiển cho phép chúng ta điều khiển tốc độ và hướng quay của động cơ DC cùng một lúc. Direction control pins và Speed control pins.

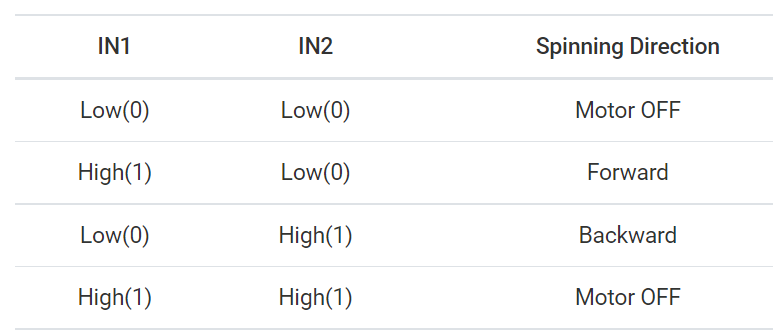
**Direction Control Pins**



Hình 34. Các chân điều khiển hướng của L293D

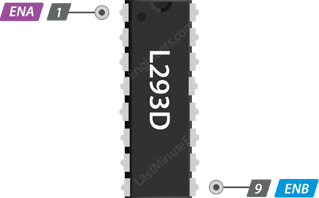
Sử dụng các chân điều khiển hướng, chúng ta có thể điều khiển động cơ quay tiến hoặc lùi. Các chân này điều khiển các công tắc của mạch H-Bridge nằm bên trong IC L293D. IC có hai chân điều khiển hướng cho mỗi phía. Các chân **IN1**, **IN2** điều khiển hướng quay của động cơ A và **IN3**, **IN4** điều khiển động cơ B.

Chiều quay của động cơ có thể được điều khiển bằng cách áp dụng logic HIGH (5 Volts) hoặc logic LOW (Ground) cho các chân này.



Hình 35. Biểu đồ minh họa cách thực hiện

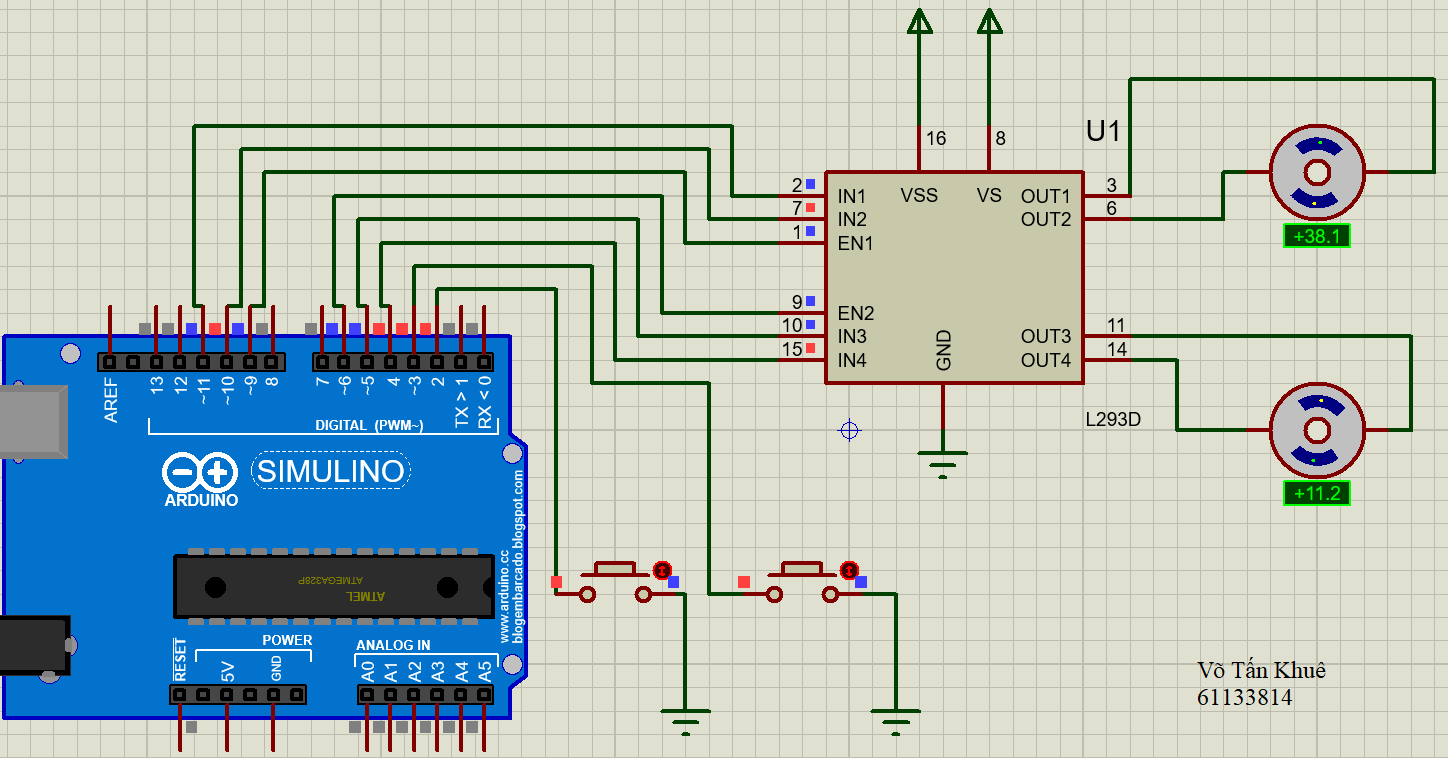
**Speed Control Pins**



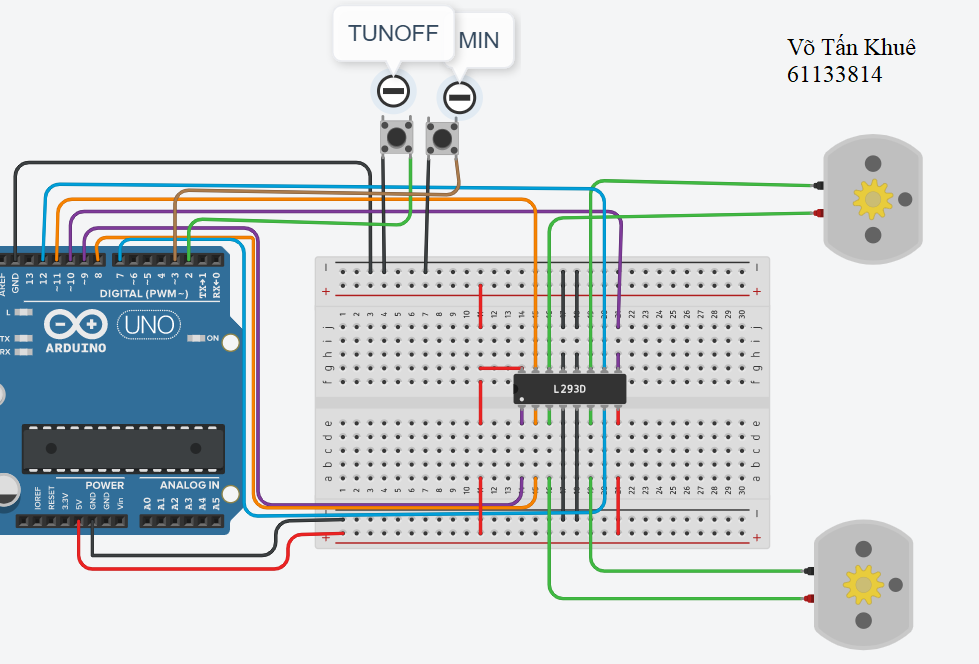
Hình 36. Các chân điều khiển tốc độ của L293D

Các chân điều khiển tốc độ **ENA** và **ENB** được sử dụng để bật, tắt và điều khiển tốc độ của động cơ A và động cơ B.

## Sơ đồ thiết kế



Hình 37. Sơ đồ mạch Proteus



Hình 38. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

## Code chương trình

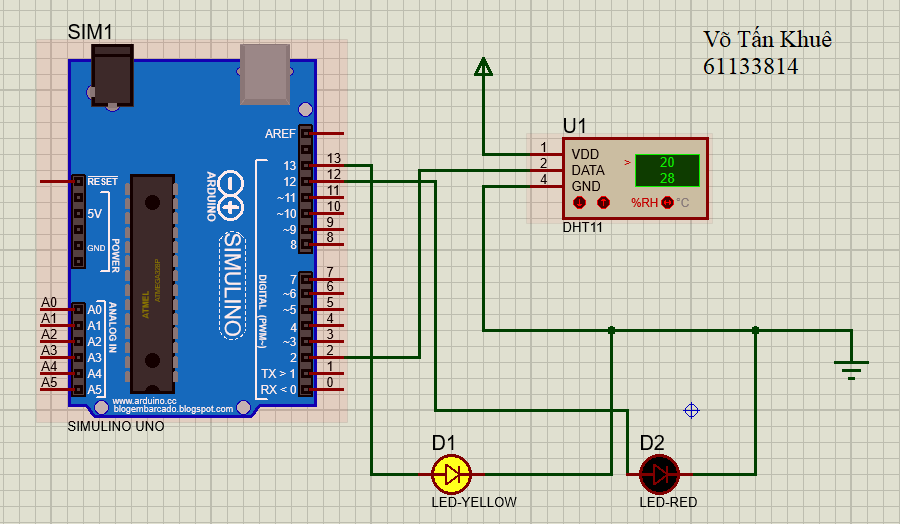
|  |
| --- |
| // Động cơ 1  int tocdo = 9; int inp1 = 11; int inp2 = 10;  // Động cơ 2  int tocdo2 = 6; int inp3 = 5; int inp4 = 4;  void setup()  {  pinMode(tocdo, OUTPUT); pinMode(inp1, OUTPUT);  pinMode(inp2, OUTPUT); pinMode(tocdo2, OUTPUT);  pinMode(inp3, OUTPUT); pinMode(inp4, OUTPUT);  digitalWrite(2, HIGH);  attachInterrupt(0, tatdongco, CHANGE);  digitalWrite(3, HIGH);  attachInterrupt(1,dieukientocdogiam, CHANGE);    //Tắt động cơ  digitalWrite(inp1,LOW); digitalWrite(inp2,LOW);  digitalWrite(inp3,LOW); digitalWrite(inp4,LOW);  }  void tatdongco()  {  digitalWrite(inp1,LOW); digitalWrite(inp2,LOW);  digitalWrite(inp3,LOW); digitalWrite(inp4,LOW);  }  void directionControl()  {  // Cho động cơ quay với tốc độ tối đa  // Giá trị của PWM thay đổi từ 0 đến 255  analogWrite(tocdo, 255);  analogWrite(tocdo2, 255);  // Tắt động cơ A và B  digitalWrite(inp1, HIGH); digitalWrite(inp2, LOW);  digitalWrite(inp3, HIGH); digitalWrite(inp4, LOW);  delay(2000);  // Thay đổi chiều quay của động cơ  digitalWrite(inp1, LOW); digitalWrite(inp2, HIGH);  digitalWrite(inp3, LOW); digitalWrite(inp4, HIGH);  delay(2000);    // Tắt tất cả các động cơ  digitalWrite(inp1, LOW); digitalWrite(inp2, LOW);  digitalWrite(inp3, LOW); digitalWrite(inp4, LOW);  }  void speedControl()  {  digitalWrite(inp1, LOW); digitalWrite(inp2, HIGH);  digitalWrite(inp3, LOW); digitalWrite(inp4, HIGH);    for(int i=0; i<256 ;i++){  analogWrite(tocdo,i);  analogWrite(tocdo2,i);  delay(10);  }    for(int i=255;i >= 0; --i){  analogWrite(tocdo,i);  analogWrite(tocdo2,i);  delay(10);  }  //tat dong co  digitalWrite(inp1,LOW); digitalWrite(inp2,LOW);  digitalWrite(inp3,LOW); digitalWrite(inp4,LOW);  }  void dieukientocdotang()  {  //bat cac dong co  digitalWrite(inp1,LOW); digitalWrite(inp2,HIGH);  digitalWrite(inp3,LOW); digitalWrite(inp4,HIGH);    //tang toc do den max  for(int i=0; i<256 ;i++){  analogWrite(tocdo,i);  analogWrite(tocdo2,i);  delay(10);  }  }  void dieukientocdogiam()  {  //bat cac dong co  digitalWrite(inp1,LOW); digitalWrite(inp2,HIGH);  digitalWrite(inp3,LOW); digitalWrite(inp4,HIGH);    //giam toc do ve 0  for(int i=255;i >= 0; --i){  analogWrite(tocdo,i);  analogWrite(tocdo2,i);  delay(10);  }  }  void loop()  {  directionControl();  delay(1000);  speedControl();  delay(1000);  } |

# Bài 13. Hệ thống theo dõi nhiệt độ và độ ẩm dựa trên cảm biến DHT11

## Mô tả

Hệ thống được thiết kế thực hiện đọc thông số nhiệt độ và độ ẩm từ môi trường, cảnh báo thông qua Led đó nếu nhiệt độ quá 30oC và Led vàng nếu độ ẩm dưới 40%.

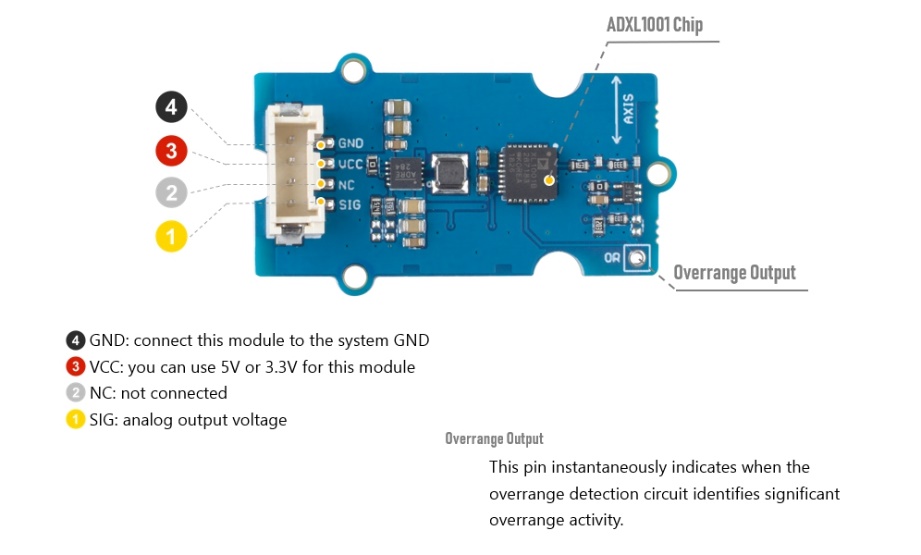
## Sơ đồ thiết kế



Hình 39. Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

* DHT11 là cảm biến số nhiệt độ và 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. Có ba chân kết nối Vcc, GND, OUT



Hình 40. Sơ đồ DHT11

Thông số kỹ thuật

* Điện áp hoạt động: 3V - 5V (DC)
* Dãi độ ẩm hoạt động: 20% - 90% RH, sai số ±5%RH
* Dãi nhiệt độ hoạt động: 0°C ~ 50°C, sai số ±2°C
* Khoảng cách truyển tối đa: 20m

Để đọc nhiệt độ và độ ẩm từ môi trường. Ta sử dụng các phương thức ReadHumidity() và readTemperature() với lớp DHT (đóng gói trong thư viện DHT.h). Hàm tạo của lớp này có 2 tham số (cổng gắn với chân Signal để đọc dữ liệu, loại cảm biến). Loại cảm biến có thể nhận 1 trong 2 giá trị là DTH11 hoặc DTH12.

* 1 mạch Arduino
* 1 đèn led vàng, 1 led đỏ

## Code chương trình

|  |
| --- |
| #include "DHT.h" // Khai báo thư viện DHT.h  #define dht\_1 2  #define DHTTYPE DHT11 //Khai báo DHT11  DHT dht(dht\_1, DHTTYPE); // Hàm DHT  void **setup**() {  dht.begin();  }  void **loop**() {  float doam = dht.readHumidity(); //Lấy chỉ số độ ẩm  float nhietdo = dht.readTemperature(); //Lấy chỉ số nhiệt độ  if(nhietdo > 30) { // Nếu nhiệt độ lớn hơn 30 thì sáng đèn đỏ  digitalWrite(12,HIGH);  }  Else digitalWrite(12,LOW);  if(doam <40 ) { // Nếu độ ẩm nhỏ hơn 30 thì sáng đèn vàng  digitalWrite(13,HIGH);  }  Else digitalWrite(13,LOW);  } |

# Bài 14. Đèn Giao Thông

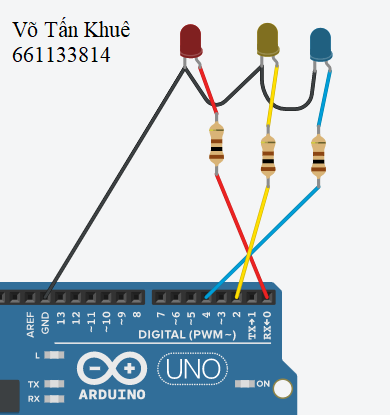
## Mô tả

Hệ thống được thiết kế tạo thành đèn giao thông qua việc bật tắt ba đèn đỏ, xanh, vàng trong một thời gian nhất định. Ở ví dụ này đèn đỏ sáng trong 45s gắng vào mạch Arduino ở cổng 0, đèn vàng sáng 10s ở cổng 2, đèn xanh sáng 30s ở cổng 4.

## Sơ đồ thiết kế



Hình 41. Sơ đồ mạch Proteus



Hình 42. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 Mạch Arduino
* 3 điện trở: 100 Ω
* 1 đèn Led màu đỏ, 1 Led màu vàng, 1 Led xanh

## Code chương trình

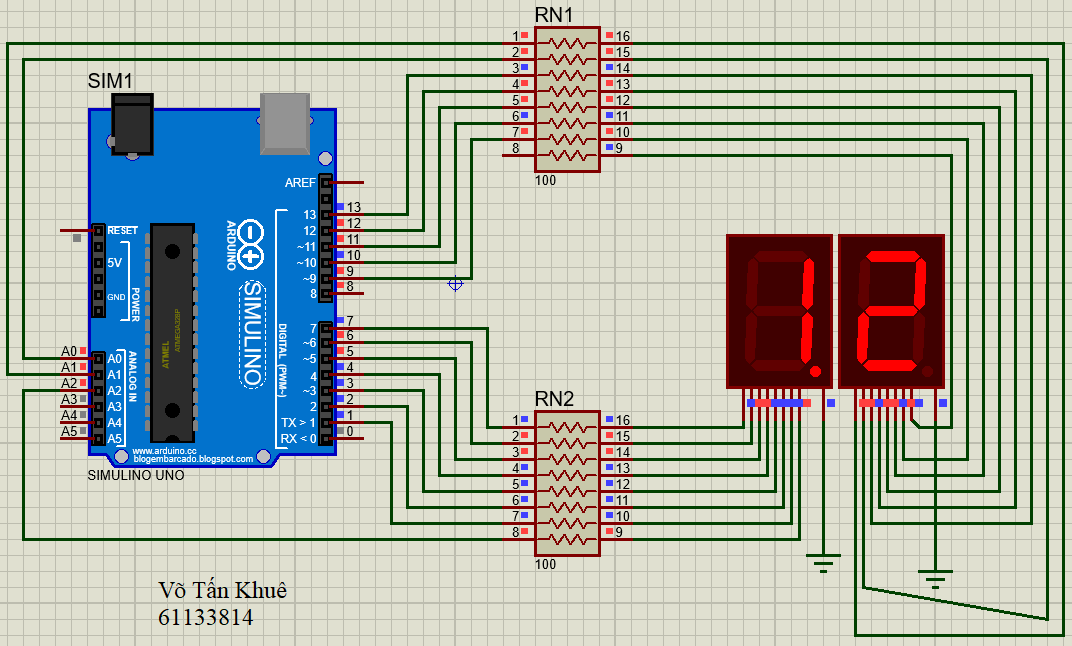
|  |
| --- |
| Int dem = 0;  void **setup**()  {  pinMode(0, OUTPUT);  pinMode(2, OUTPUT);  pinMode(4, OUTPUT);  }  void **loop**()  {  dem = dem + 1; delay(1000); //Biến dem để tính thời gian trong 1s  if(dem == 85){ //Tổng thời gian 3 đèn sáng là 85s  dem=0;  }  if(dem<=45){  digitalWrite(0, HIGH);  digitalWrite(2, LOW); //Bật đèn đỏ  digitalWrite(4, LOW);  }  if(dem>45&&dem<=55){  digitalWrite(0, LOW);  digitalWrite(2, HIGH); // Bật đèn vàng  digitalWrite(4, LOW);  }  if(dem>=56){  digitalWrite(0, LOW);  digitalWrite(2, LOW); //Bật đèn xanh  digitalWrite(4, HIGH);  }  } |

# Bài 15. Hiển thị số đếm trên 2 đèn Led 7 đoạn

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển hiện các số đếm từ 0 đến 9, sau đó hiện số đếm từ 0.0 đến 9.9 với Led 2 đèn Led 7 đoạn.

## Sơ đồ thiết kế



Hình 43. Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

* 2 đèn Led 7 đoạn
* 2 điện trở RES16: 100 Ω, điện trở này có 16 cổng

## Code chương trình

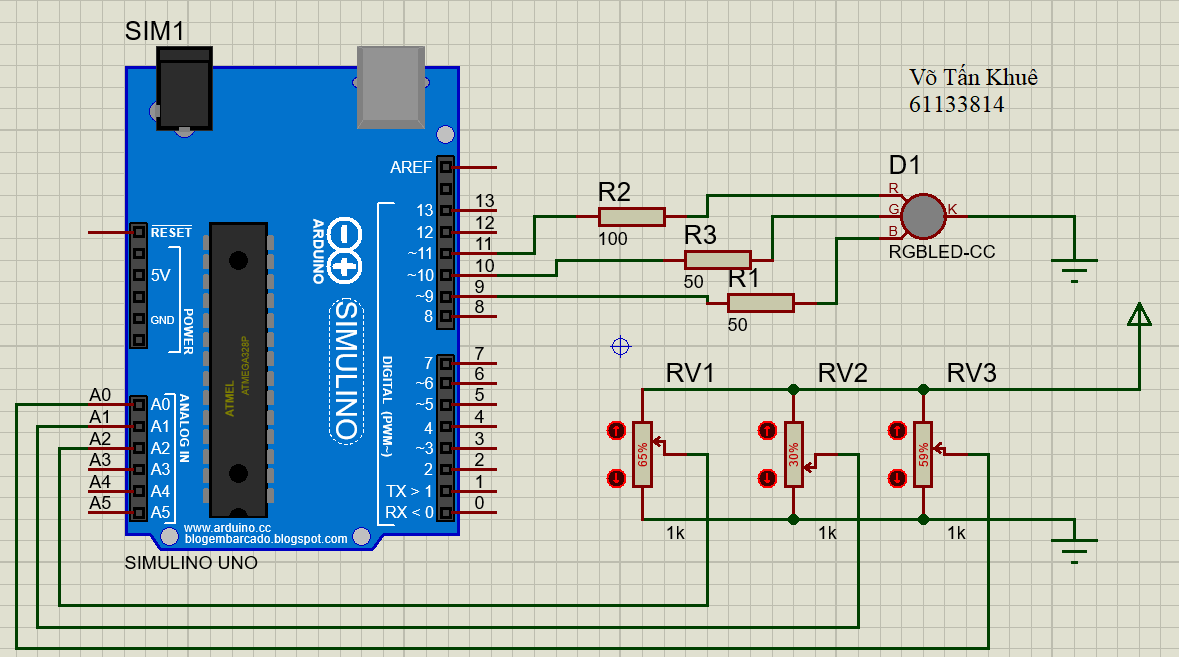
|  |
| --- |
| int display0[] = {7, 6, 5, 4, 3, 2, 1, A2};  int display1[] = {7, 6, 5, 4, 3, 2, 1, A2};  int display2[] = {A1, A0, 13, 12, 11, 10, 9, 8};  int nums0[10][8] = {  {1, 1, 1, 1, 1, 1, 0, 0}, //Hiển thị số 0  {0, 1, 1, 0, 0, 0, 0, 0}, // 1  {1, 1, 0, 1, 1, 0, 1, 0}, // 2  {1, 1, 1, 1, 0, 0, 1, 0}, // 3  {0, 1, 1, 0, 0, 1, 1, 0}, // 4  {1, 0, 1, 1, 0, 1, 1, 0}, // 5  {1, 0, 1, 1, 1, 1, 1, 0}, // 6  {1, 1, 1, 0, 0, 0, 0, 0}, // 7  {1, 1, 1, 1, 1, 1, 1, 0}, // 8  {1, 1, 1, 1, 0, 1, 1, 0} // 9  };  int nums[10][8] = {  {1, 1, 1, 1, 1, 1, 0, 1}, // Hiển thị số 0  {0, 1, 1, 0, 0, 0, 0, 1}, // 1  {1, 1, 0, 1, 1, 0, 1, 1}, // 2  {1, 1, 1, 1, 0, 0, 1, 1}, // 3  {0, 1, 1, 0, 0, 1, 1, 1}, // 4  {1, 0, 1, 1, 0, 1, 1, 1}, // 5  {1, 0, 1, 1, 1, 1, 1, 1}, // 6  {1, 1, 1, 0, 0, 0, 0, 1}, // 7  {1, 1, 1, 1, 1, 1, 1, 1}, // 8  {1, 1, 1, 1, 0, 1, 1, 1} // 9  };  const byte dot = A2; // Hiển thị dấu “.”  const int showTime = 200; // Thiết lập thời gian giữa 2 lần hiển thị  void **setup**() {  pinMode(dot, OUTPUT);  for(byte i = 0; i < 9; i++){  pinMode(display0[i], OUTPUT);  pinMode(display1[i], OUTPUT);  pinMode(display2[i], OUTPUT);  }  }  void **displayNum**(int display[], int num[]) {  for(byte i = 0; i < 8; i++)  digitalWrite(display[i], num[i]);  }  void **loo**p() {  for(byte i = 0; i < 10; i++) {  displayNum(display0, nums0[i]);  delay(showTime);  }    for(byte i = 0; i < 10; i++) {  displayNum(display1, nums[i]);  for(byte j = 0; j < 10; j++) {  displayNum(display2, nums[j]);  delay(showTime);  }  }  } |

# Bài 16. Đổi màu đèn RGB bằng Chiết Áp

## Mô tả

Hệ thống được thiết kế điều khiển làm đèn RGB sáng ở các màu bất kỳ thông qua ba chiết áp được kết nối vào Arduino ở các cổng số A0, A1, A2.

## Sơ đồ thiết kế



Hình 44. Sơ đồ mạch Proteus



Hình 45. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 Đèn RBB
* 3 Điện trở
* 3 Chiết áp

## Code chương trình

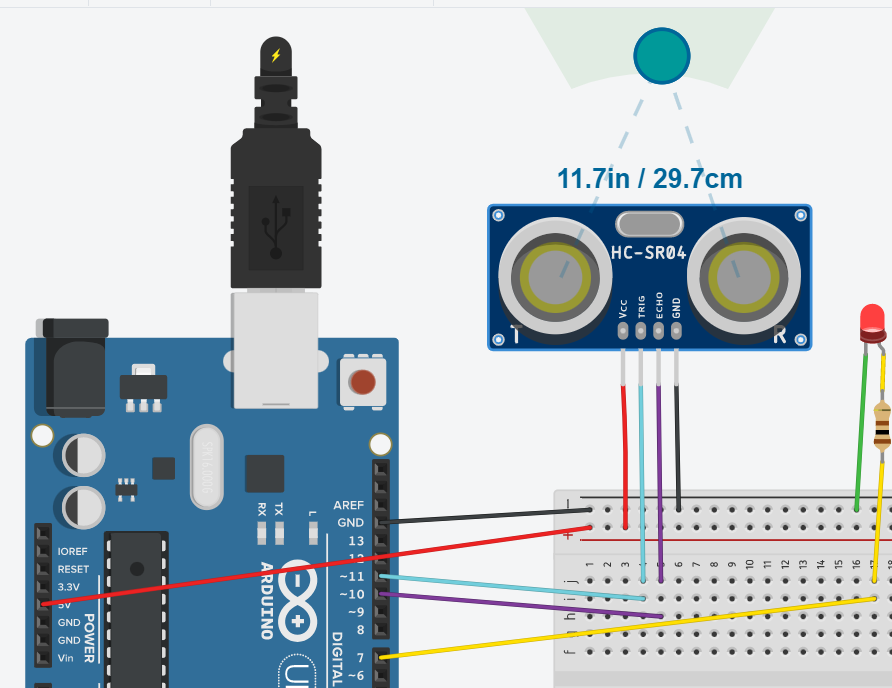
|  |
| --- |
| int led1 = 9; int led2 = 10; int led3 = 11; // Khai báo số chân Arduino  void **setup**() // nối với 3 đèn  {  pinMode(led1, OUTPUT);  pinMode(led2, OUTPUT);  pinMode(led3, OUTPUT);  }  void **loop**()  {  int val1 = map(analogRead(A0),0,1023,0,255); //Đọc giá trị chiết áp 1  int val2 = map(analogRead(A1),0,1023,0,255); //Đọc giá trị chiết áp 2  int val3 = map(analogRead(A2),0,1023,0,255); //Đọc giá trị chiết áp 3  analogWrite(led1,val1);  analogWrite(led2,val2);  analogWrite(led3,val3);  } |

# Bài 17. Bật tắt đèn Led với cảm biến HC-SR04

## Mô tả

Hệ thống được thiết kế bật/tắt đèn Led với HC-SR04 được gắn vào chân 10,11 của Arduino. Đèn chỉ sáng nếu cảm biến nhận thấy vật dưới 50m.

## Sơ đồ thiết kế

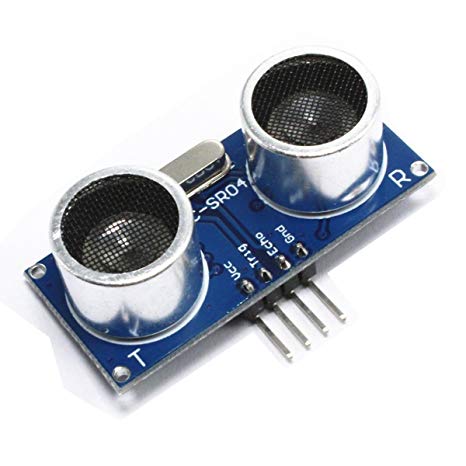


Hình 46. Sơ đồ mạch Proteus

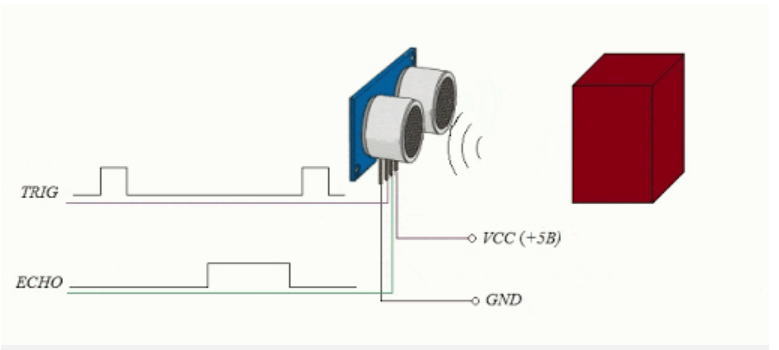
## Đặc điểm của linh kiện

* 1 đèn Led
* 1 điện trở: 100 Ω
* 1 mạch Arduino Uno
* 1 cảm biến HC-SR04

Cảm biến siêu âm Ultrasonic HC-SR04 được sử dụng để nhận biết khoảng cách từ vật thể đến cảm biến nhờ sóng siêu âm, cảm biến có thời gian phản hồi nhanh, độ chính xác cao, phù hợp cho các ứng dụng phát hiện vật cản, đo khoảng cách bằng sóng siêu âm.



Hình 47. Module cảm biến siêu âm HC-SR04



Hình 48. Phát sóng trên cảm biến siêu âm HC-SR04

Nguyên lý hoạt động cảm biến siêu âm HC-SR04

Để đo khoảng cách bằng cảm biến siêu âm HC-SR04, ta sẽ phát 1 xung rất ngắn (5 microSeconds) từ chân Trig. Tiếp theo, 1 xung HIGH ở chân Echo sẽ được cảm biến tạo ra và phát đi cho đến khi nhận lại được sóng phản xạ ở chân này. Lúc này, độ rộng của xung sẽ bằng với thời gian sóng siêu âm được phát từ cảm biến và phản xạ lại. Trong không khí, tốc độ âm thanh đạt mức 340 m/s (hằng số), tương đương với 29,412 microSeconds/cm (106 / (340\*100)). Khi đã tính được thời gian, ta sẽ chia cho 29,412 để ra giá trị khoảng cách.

Ví dụ đo mức chất lỏng có trong bồn chứa bằng cảm biến siêu âm HC-SR04. Ta sẽ có các bước như sau:

1. Đặt chân TRIG lên mức Cao (5V) trong 10 μs (microseconds)
2. Sau đó module siêu âm ghi lại thời gian và gửi ra sóng âm tần số 40Khz
3. Sóng siêu âm truyền xuống bề mặt chất lỏng trong bồn chứa và phản xạ lại
4. Sóng phản xạ truyền ngược về đầu dò
5. Module siêu âm nhận được sóng phản xạ và đánh dấu thời gian nhận được tín hiệu này

Cuối cùng, module siêu âm đưa chân ECHO lên mức cao trong khoảng thời gian (microseconds) phản hồi sóng âm (Gửi đi – nhận về) và tính toán ra khoảng cách.Kết quả trên chân ECHO: 58 μs/cm. Vì vậy, nếu chân ECHO lên mức cao trong thời gian 5800 μs (5.8 ms) , thì chúng ta tính được khoảng cách giữa cảm biến và mức chất lỏng trong bể là: **5800μs / 58μs/cm = 100cm = 1m**

Thông số kỹ thuật cảm biến siêu âm HC-SR04:

* Model: HC-SR04
* Điện áp làm việc: 5VDC
* Dòng điện: 15mA
* Tần số: 40 KHZ
* Khoảng cách phát hiện: 2cm – 4m
* Tín hiệu đầu ra: Xung mức cao 5V, mức thấp 0V
* Góc cảm biến: Không quá 15 độ.
* Độ chính xác cao: Lên đến 3mm
* Chế độ kết nối: VCC / Trig (T-Trigger) / Echo (R-Receive) / GND

Module cảm biến có 4 chân:

* Chân VCC: Dùng để cấp nguồn 5v
* Chân Trig: Chân digital output
* Chân Echo: Chân digital input
* Chân GND: Chân 0v

## Code chương trình

|  |
| --- |
| int trig = 11, echo = 10;  int led = 7;  float distance;  void setup()  {  pinMode(trig,OUTPUT);  pinMode(echo,INPUT);  pinMode(led, OUTPUT);  Serial.begin(9600);  }  float getDistance(){  digitalWrite(trig, HIGH);  delayMicroseconds(5);  digitalWrite(trig,LOW);  int timer = pulseIn(echo, HIGH);  return timer/58.3f;  }  void loop()  {  distance = getDistance();  if(distance < 50){  digitalWrite(led, HIGH);  }  else digitalWrite(led, LOW);  Serial.println(distance);  } |

# Bài 18. Xe ba bánh

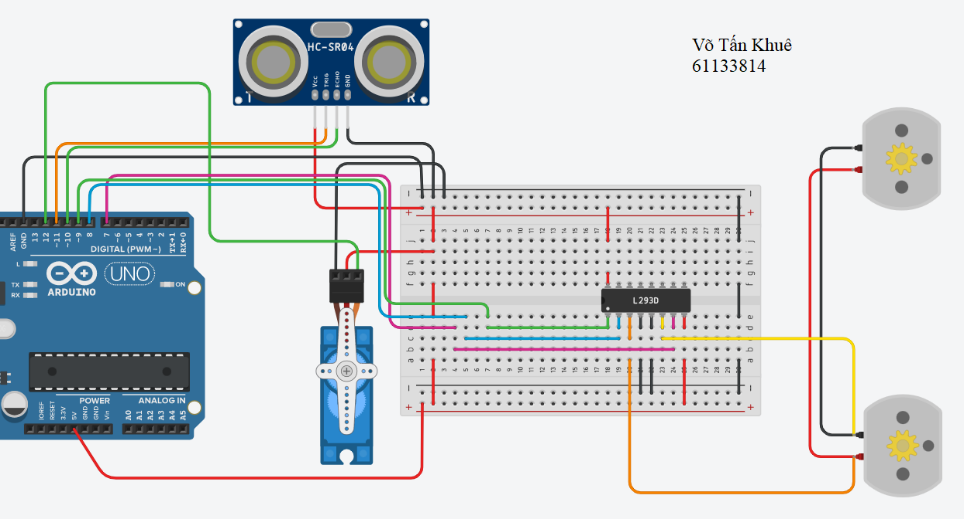
## Mô tả

Hệ thống được thiết kế một xe ba bánh, trong đó 1 động cơ điều kiển hướng ở bánh trước, 2 động cơ điều khiển 2 bánh sau, phía trước được gắn một cảm biến vật cản sử dụng sóng âm HC-SR04. Xe tự hành với những yêu cầu như:

Liên tục thực hiện phát hiển vật cản phía trước sau mỗi giây. Khi vật cản ở khoảng cách Y = 1 (m), thực hiện các xử lý sau:

Nếu tốc độ xe chậm, dừng xe và lùi xe ngược lại trong khoảng thời gian 10 giây. Nếu tốc độ xe nhanh, đánh tay lái sang trái 20 độ và giảm tốc độ Z+10 đơn vị, rồi dừng hẳn sau khoảng thời gian 5 giây. Với giá trị Y và Z nhanh, chậm là do tự định nghĩa

## Sơ đồ thiết kế



Hình 49. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 1 cảm biến vật cản song âm âm HC-SR04
* 1 L2930
* 2 động cơ DC
* 1 mạch Arduino Uno

## Code chương trình

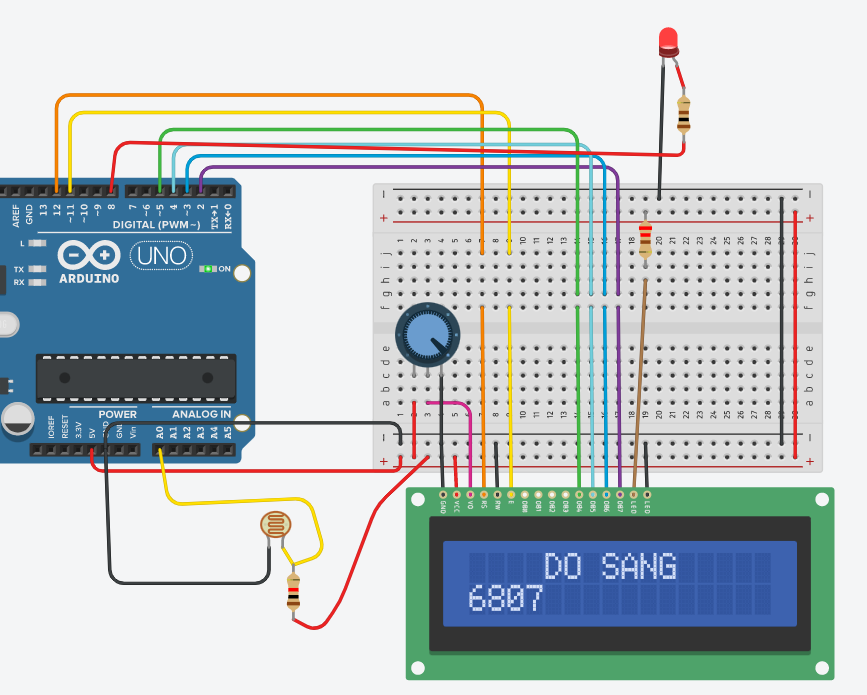
|  |
| --- |
| #include <Servo.h> //Khai báo thư viện  int trig = 11, echo = 10;  float distance;  int tocdo = 9; int inp1 = 8; int inp2 = 7; // Động cơ sau  Servo myservo; // Động cơ trước  void **setup**()  {  pinMode(trig,OUTPUT);  pinMode(echo,INPUT);  myservo.attach(12, 500, 2500);  pinMode(tocdo, OUTPUT);  pinMode(inp1, OUTPUT);  pinMode(inp2, OUTPUT);  digitalWrite(inp1,LOW); // Tắt động cơ  digitalWrite(inp2,LOW);  Serial.begin(9600);  }  float **Dokhoancach**(){  digitalWrite(trig, HIGH);  delayMicroseconds(5);  digitalWrite(trig,LOW);  int timer = pulseIn(echo, HIGH);  return timer/58.3f;  }  void **tatdongco**()  {  digitalWrite(inp1,LOW); digitalWrite(inp2,LOW);  delay(2000);  }  void **loop**()  {  int x = 200;  digitalWrite(inp1, LOW);  digitalWrite(inp2, HIGH);  analogWrite(tocdo,140);  //Cho Y là 1 m và Z trung bình là 150  distance = Dokhoancach();  Serial.println(distance);    if(distance < 100){  if(x < 150){  tatdongco();  for(int i=0;i<=10;i++){  digitalWrite(inp1, HIGH);  digitalWrite(inp2, LOW);  Serial.println(i);  delay(1000);  }  digitalWrite(inp1,LOW);  digitalWrite(inp2,LOW);  }else {  myservo.write(20);  for(int i=5;i>=0;i--){  x=x-10;  Serial.println(x);  Serial.println(i);  delay(1000);  }  tatdongco();  }  }else  {  digitalWrite(inp1, LOW);  digitalWrite(inp2, HIGH);  }  } |

# Bài 19. Hiển thị giá trị cảm biến ánh sáng Lên LCD

## Mô tả

Hệ thống nhúng sử dụng board mạch Arduino được thiết kế liên tục đọc giá trị độ sáng cảm biến được và hiện ra LCD sau mỗi giây, nếu giá trị cảm nhận được nhỏ hơn 50% thang đo thì bật đèn Led sáng và ngược lại.

## Sơ đồ thiết kế



## Đặc điểm của linh kiện

## Code chương trình

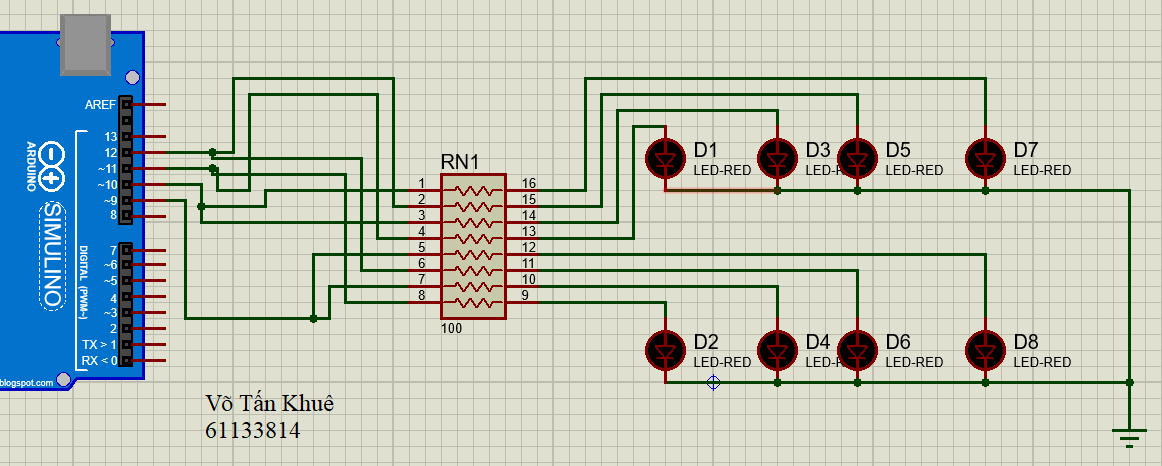
|  |
| --- |
|  |

# Bài 20. Nháy 8 đèn Led

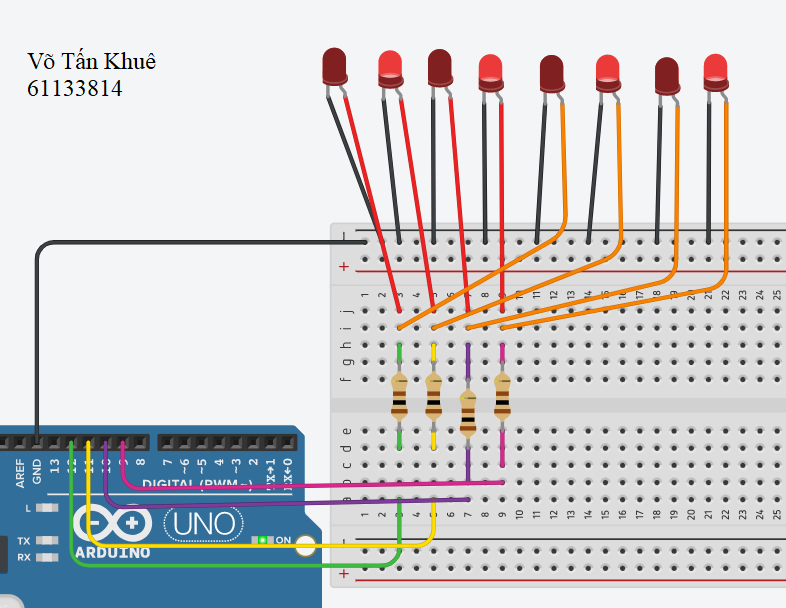
## Mô tả

Hệ thống nhúng sử dụng board mạch Arduino được thiết kế thực hiện sáng tuần tự từng cặp Led (có 8 led tất cả), sau đó tắt đồng thời các Led được gắn ở các port lẻ và sáng đồng thời các Led ở port chẵn, sau đó chớp 3 lần đồng thời 8 Led

## Sơ đồ thiết kế



Hình 50. Sơ đồ mạch Proteus



Hình 51. Sơ đồ mạch Tinkercad

## Đặc điểm của linh kiện

* 8 đèn Led
* 4 điện trở: 100 Ω
* 1 mạch Arduino Uno

## Code chương trình

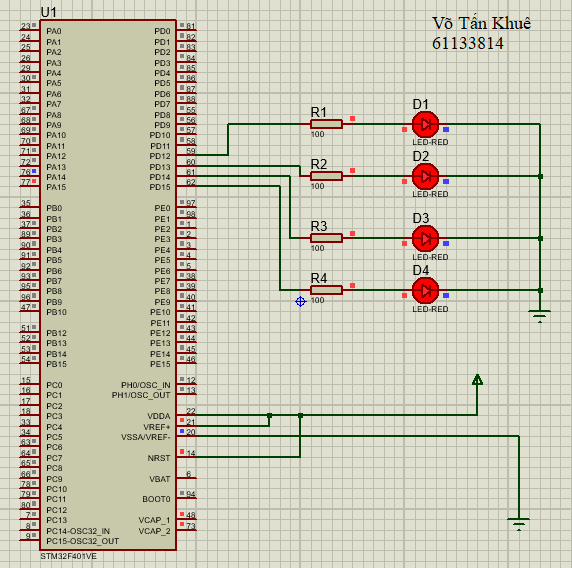
|  |
| --- |
| void **setup**(){  pinMode(12, OUTPUT); pinMode(11, OUTPUT);  pinMode(10, OUTPUT); pinMode(9, OUTPUT);  }  void **sangtungcap**(){  digitalWrite(12, HIGH); digitalWrite(11, LOW);  digitalWrite(10, LOW); digitalWrite(9, LOW);  delay(500);  digitalWrite(12, LOW); digitalWrite(11, HIGH);  digitalWrite(10, LOW); digitalWrite(9, LOW);  delay(500);  digitalWrite(12, LOW); digitalWrite(11, LOW);  digitalWrite(10, HIGH); digitalWrite(9, LOW);  delay(500);  digitalWrite(12, LOW); digitalWrite(11, LOW);  digitalWrite(10, LOW); digitalWrite(9, HIGH);  delay(500);  }  void **sangdenvtchan**(){  digitalWrite(12, LOW); digitalWrite(11, HIGH);  digitalWrite(10, LOW); digitalWrite(9, HIGH);  }  void **chapnhaybalan**(){  digitalWrite(12, HIGH); digitalWrite(11, HIGH);  digitalWrite(10, HIGH); digitalWrite(9, HIGH);  delay(500);  digitalWrite(12, LOW); digitalWrite(11, LOW);  digitalWrite(10, LOW); digitalWrite(9, LOW);  delay(500);  digitalWrite(12, HIGH); digitalWrite(11, HIGH);  digitalWrite(10, HIGH); digitalWrite(9, HIGH);  delay(500);  digitalWrite(12, LOW); digitalWrite(11, LOW);  digitalWrite(10, LOW); digitalWrite(9, LOW);  delay(500);  digitalWrite(12, HIGH); digitalWrite(11, HIGH);  digitalWrite(10, HIGH); digitalWrite(9, HIGH);  delay(500);  digitalWrite(12, LOW); digitalWrite(11, LOW);  digitalWrite(10, LOW); digitalWrite(9, LOW);  delay(500);  }  void **loop**(){  sangtungcap(); delay(1000);  sangdenvtchan(); delay(1000);  chapnhaybalan(); delay(1000);  } |

# Bài 21. Nháy 4 đèn Led thông qua STM32F401VE

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển 4 đèn Led tự động bật/tắt sau 1 giây thông qua 4 chân PD12, PD13, PD14 và PD15 của STM32F401VE.

## Sơ đồ thiết kế

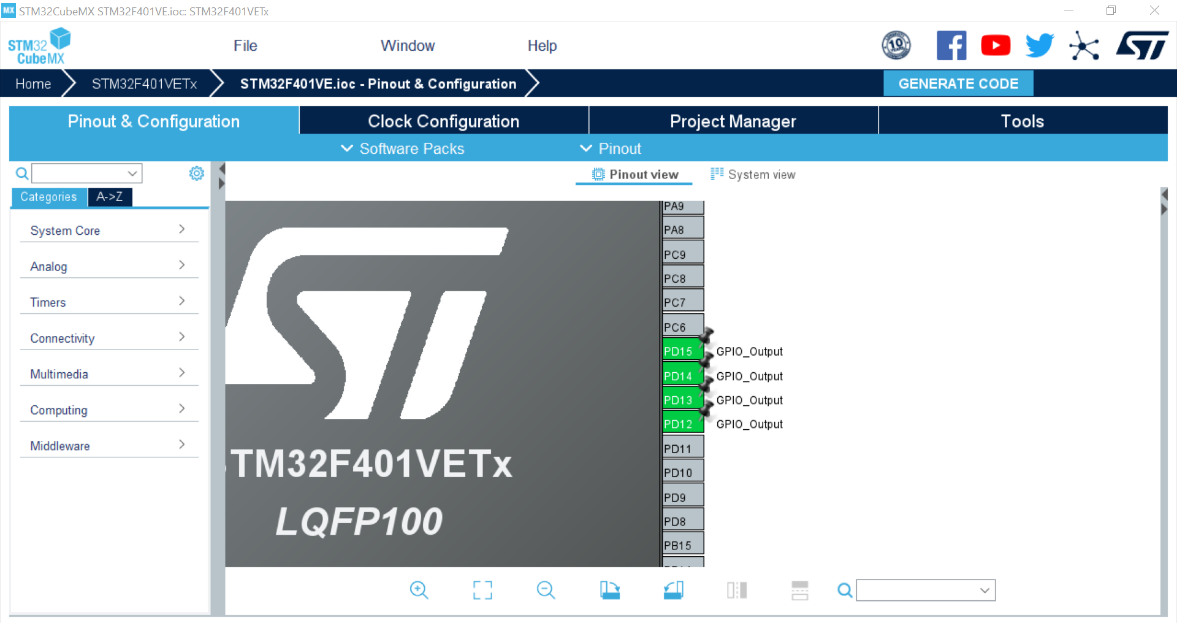


Hình 52. Sơ đồ mạch Proteus

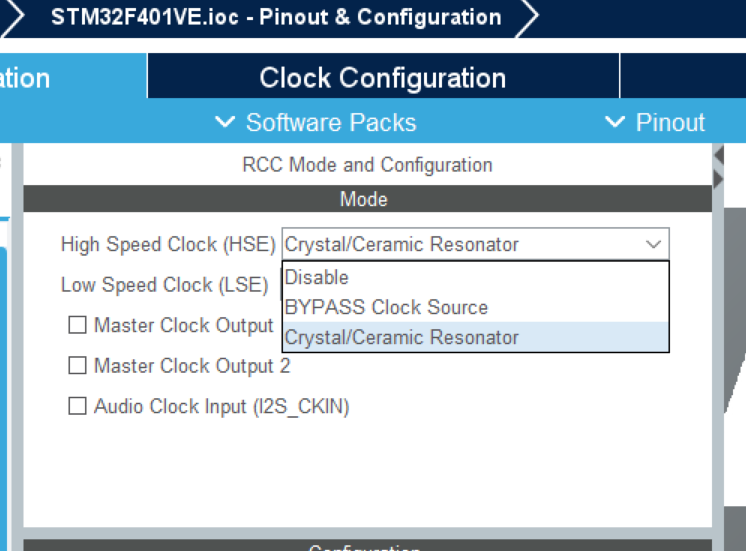
## Đặc điểm của linh kiện

* 4 điện trở: 100 Ω
* 4 đèn Led
* 1 STM32F401VE

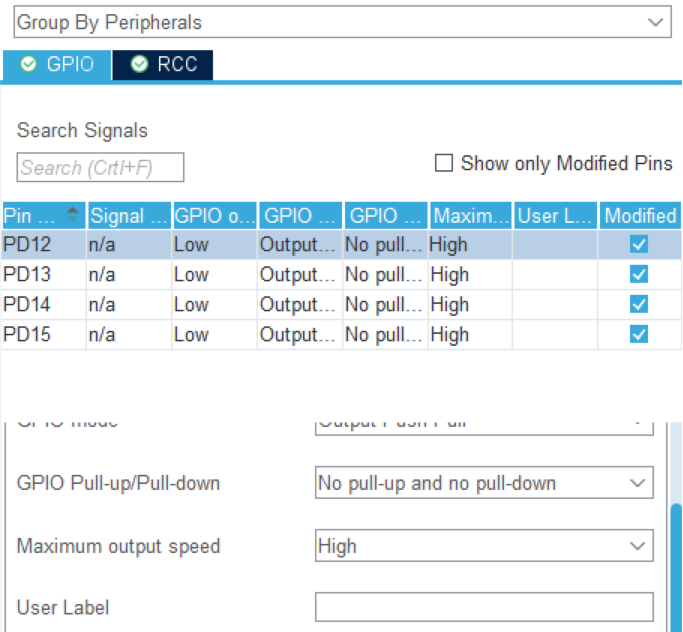
Các cấu hình với CubeMX



Hình 53. Chọn các cổng PD12, PD13, PD14, PD15 xuất dữ liệu



Hình 54. Chọn nguồn xung cho Chip



Hình 55. Cấu hình cho các chân GPIO

## Code chương trình

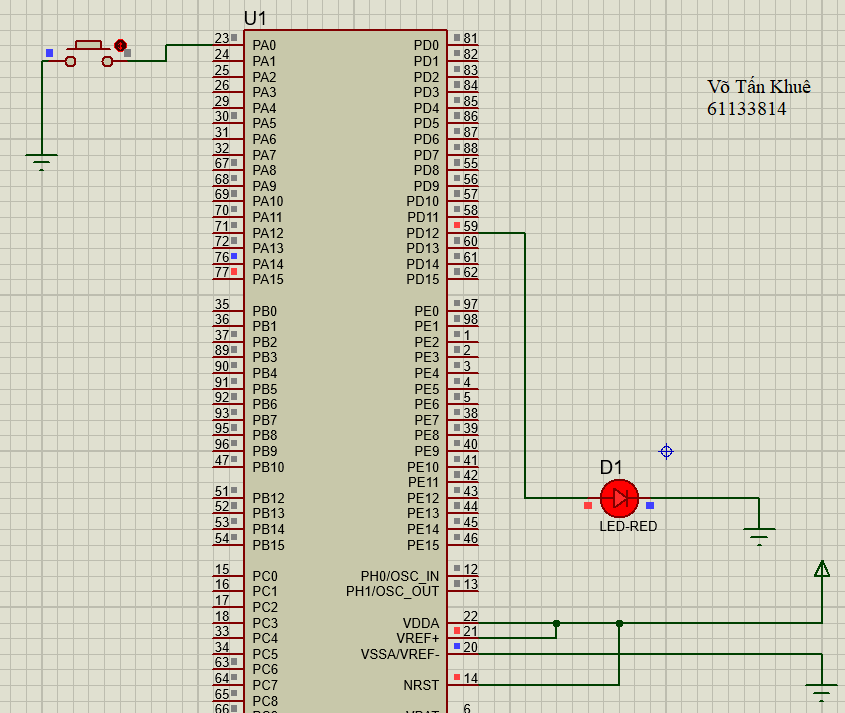
|  |
| --- |
| int main(void)  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  while (1)  {  HAL\_GPIO\_TogglePin(GPIOD,GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15);  HAL\_Delay(1000); //delay 1s  }  } |

# Bài 22. Ngắt ngoài trên STM32F401VE

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển ngắt ngoài bật/tắt đèn Led trên STM32F401VE.Trên MCU này, nhà sản xuất đã kết nối sẵn cho chúng ta chân PA0 với User button nên ta sử dụng chân này để kết nối với nút bấm.

## Sơ đồ thiết kế



Hình 56. Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

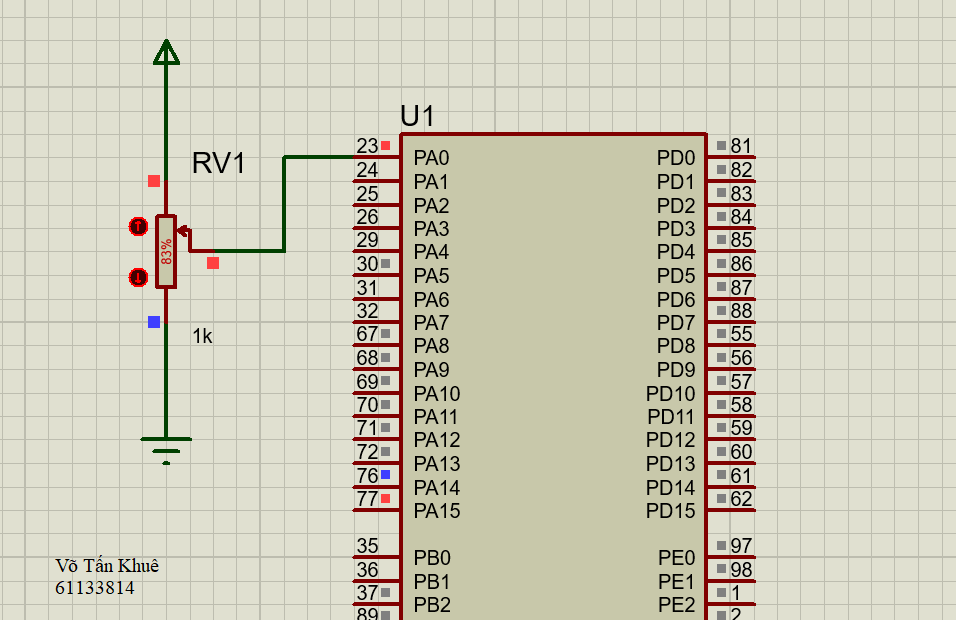
## Code chương trình

|  |
| --- |
| void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin)  {  if(GPIO\_Pin == GPIO\_PIN\_0)  { HAL\_GPIO\_TogglePin(GPIOD, GPIO\_PIN\_12); }  } |

# Bài 23.

## Mô tả

## Sơ đồ thiết kế



Hình 57. Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

## Code chương trình

|  |
| --- |
| void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin)  {  if(GPIO\_Pin == GPIO\_PIN\_0)  { HAL\_GPIO\_TogglePin(GPIOD, GPIO\_PIN\_12); }  } |

# Bài 24.

## Mô tả

## Sơ đồ thiết kế

## Đặc điểm của linh kiện

## Code chương trình

|  |
| --- |
|  |

# Bài 25.

## Mô tả

## Sơ đồ thiết kế

## Đặc điểm của linh kiện

## Code chương trình

# Bài 26.

## Mô tả

## Sơ đồ thiết kế

## Đặc điểm của linh kiện

## Code chương trình

MỤC LỤC

[Bài 1. Nháy đèn Led 1](#_Toc85750689)

[1. Mô tả 1](#_Toc85750690)

[2. Sơ đồ thiết kế 1](#_Toc85750691)

[3. Đặc điểm của linh kiện 1](#_Toc85750692)

[4. Code chương trình 2](#_Toc85750693)

[Bài 2. Nút bấm đề bật đèn Led 2](#_Toc85750694)

[1. Mô tả 2](#_Toc85750695)

[2. Sơ đồ thiết kế 3](#_Toc85750696)

[3. Đặc điểm của linh kiện 3](#_Toc85750697)

[4. Code chương trình 4](#_Toc85750698)

[Bài 3. Nháy đèn Led RGB 4](#_Toc85750699)

[1. Mô tả 4](#_Toc85750700)

[2. Sơ đồ thiết kế 5](#_Toc85750701)

[3. Đặc điểm của linh kiện 5](#_Toc85750702)

[4. Code chương trình 6](#_Toc85750703)

[Bài 4. Đèn Led sáng dần 6](#_Toc85750704)

[1. Mô tả 6](#_Toc85750705)

[2. Sơ đồ thiết kế 7](#_Toc85750706)

[3. Đặc điểm của linh kiện 7](#_Toc85750707)

[4. Code chương trình 7](#_Toc85750708)

[Bài 5. Cảm biến nhiệt độ 8](#_Toc85750709)

[1. Mô tả 8](#_Toc85750710)

[2. Sơ đồ thiết kế 8](#_Toc85750711)

[3. Đặc điểm của linh kiện 9](#_Toc85750712)

[4. Code chương trình 10](#_Toc85750713)

[Bài 6. Điều khiển độ sáng của đèn bằng chiết áp 10](#_Toc85750714)

[1. Mô tả 10](#_Toc85750715)

[2. Sơ đồ thiết kế 10](#_Toc85750716)

[3. Đặc điểm của linh kiện 11](#_Toc85750717)

[4. Code chương trình 11](#_Toc85750718)

[Bài 7. Led 7 đoạn 12](#_Toc85750719)

[1. Mô tả 12](#_Toc85750720)

[2. Sơ đồ thiết kế 12](#_Toc85750721)

[3. Đặc điểm của linh kiện 13](#_Toc85750722)

[4. Code chương trình 14](#_Toc85750723)

[Bài 8. Led 7 đoạn BCD 15](#_Toc85750724)

[1. Mô tả 15](#_Toc85750725)

[2. Sơ đồ thiết kế 16](#_Toc85750726)

[3. Đặc điểm của linh kiện 16](#_Toc85750727)

[4. Code chương trình 16](#_Toc85750728)

[Bài 9. Sáng 8 Led bằng IC 74HC595 17](#_Toc85750729)

[1. Mô tả 17](#_Toc85750730)

[2. Sơ đồ thiết kế 17](#_Toc85750731)

[3. Đặc điểm của linh kiện 18](#_Toc85750732)

[4. Code chương trình 18](#_Toc85750733)

[Bài 10. Led Matrix 8x8 19](#_Toc85750734)

[1. Mô tả 19](#_Toc85750735)

[2. Sơ đồ thiết kế 19](#_Toc85750736)

[3. Đặc điểm của linh kiện 19](#_Toc85750737)

[4. Code chương trình 20](#_Toc85750738)

[Bài 11. Sử dụng ngắt ngoài tắt đèn LED khi nhấn nút bấm đề 20](#_Toc85750739)

[1. Mô tả 20](#_Toc85750740)

[2. Sơ đồ thiết kế 20](#_Toc85750741)

[3. Đặc điểm của linh kiện 20](#_Toc85750742)

[4. Code chương trình 21](#_Toc85750743)

[Bài 12. Điều khiển động cơ DC với IC L293D và Drduino 21](#_Toc85750744)

[1. Mô tả 21](#_Toc85750745)

[2. Sơ đồ thiết kế 25](#_Toc85750746)

[3. Đặc điểm của linh kiện 26](#_Toc85750747)

[4. Code chương trình 26](#_Toc85750748)

[Bài 13. Hệ thống theo dõi nhiệt độ và độ ẩm dựa trên cảm biến DHT11 29](#_Toc85750749)

[1. Mô tả 29](#_Toc85750750)

[2. Sơ đồ thiết kế 29](#_Toc85750751)

[3. Đặc điểm của linh kiện 30](#_Toc85750752)

[4. Code chương trình 30](#_Toc85750753)

[Bài 14. Đèn Giao Thông 31](#_Toc85750754)

[1. Mô tả 31](#_Toc85750755)

[2. Sơ đồ thiết kế 31](#_Toc85750756)

[3. Đặc điểm của linh kiện 32](#_Toc85750757)

[4. Code chương trình 32](#_Toc85750758)

[Bài 15. Hiển thị số đếm trên 2 đèn Led 7 đoạn 33](#_Toc85750759)

[1. Mô tả 33](#_Toc85750760)

[2. Sơ đồ thiết kế 33](#_Toc85750761)

[3. Đặc điểm của linh kiện 33](#_Toc85750762)

[4. Code chương trình 34](#_Toc85750763)

[Bài 16. Đổi màu đèn RGB bằng Chiết Áp 35](#_Toc85750764)

[1. Mô tả 35](#_Toc85750765)

[2. Sơ đồ thiết kế 36](#_Toc85750766)

[3. Đặc điểm của linh kiện 36](#_Toc85750767)

[4. Code chương trình 36](#_Toc85750768)

[Bài 17. Xe ba bánh 37](#_Toc85750769)

[1. Mô tả 37](#_Toc85750770)

[2. Sơ đồ thiết kế 38](#_Toc85750771)

[3. Đặc điểm của linh kiện 38](#_Toc85750772)

[4. Code chương trình 38](#_Toc85750773)

[Bài 18. Hiển thị giá trị cảm biến ánh sáng Lên LCD 40](#_Toc85750774)

[1. Mô tả 40](#_Toc85750775)

[2. Sơ đồ thiết kế 41](#_Toc85750776)

[3. Đặc điểm của linh kiện 41](#_Toc85750777)

[4. Code chương trình 41](#_Toc85750778)

[Bài 19. Nháy 8 đèn Led 41](#_Toc85750779)

[1. Mô tả 41](#_Toc85750780)

[2. Sơ đồ thiết kế 41](#_Toc85750781)

[3. Đặc điểm của linh kiện 42](#_Toc85750782)

[4. Code chương trình 42](#_Toc85750783)

[Bài 20. 44](#_Toc85750784)

[1. Mô tả 44](#_Toc85750785)

[2. Sơ đồ thiết kế 44](#_Toc85750786)

[3. Đặc điểm của linh kiện 44](#_Toc85750787)

[4. Code chương trình 44](#_Toc85750788)