BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC NHA TRANG

KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO MÔN HỌC

LẬP TRÌNH THIẾT BỊ NHÚNG

GV: Mai Cường Thọ

SVTH: Trần Khải Hoàn

MSSV: 59130790

Khánh Hòa – Tháng 10/2021

Mục lục

[Bài 1: Led nhấp nháy 3](#_Toc90820604)

[Bài 2: Nháy Led bằng nút 4](#_Toc90820605)

[Bài 3: Led sáng dần 5](#_Toc90820606)

[Bài 4: Led Chiết áp 6](#_Toc90820607)

[Bài 5: Cảm biến Âm thanh 7](#_Toc90820608)

[Bài 6: Led RGB 9](#_Toc90820609)

[Bài 7: Cảm biến nhiệt độ 11](#_Toc90820610)

[Bài 8: LED 7 Seg 1 digit BCD 13](#_Toc90820611)

[Bài 9: LED 7 SEG 2 DIGIT Cathode 15](#_Toc90820612)

[Bài 10: Điều khiển động cơ motor DC 17](#_Toc90820613)

[Bài 11: Cảm biến DHT11 độ ẩm và nhiệt độ và relay 19](#_Toc90820614)

[Bài 12: Sử dụng keypad để hiện thị mật khẩu bằng màng hình LCD 21](#_Toc90820615)

[Bài 13. Nháy 4 đèn Led thông qua STM32 25](#_Toc90820616)

[Bài 14. Ngắt ngoài trên STM32 28](#_Toc90820617)

[Bài 15. Chức năng ADC trên vi điều khiển STM32F4 32](#_Toc90820618)

[Bài 16. Sáng Led hình trái tim bằng STM32 35](#_Toc90820619)

[Bài 17. Điều khiển hiển thị lên màn hình LCD trên mạch STM32F401VE 42](#_Toc90820620)

# Bài 1: Led nhấp nháy

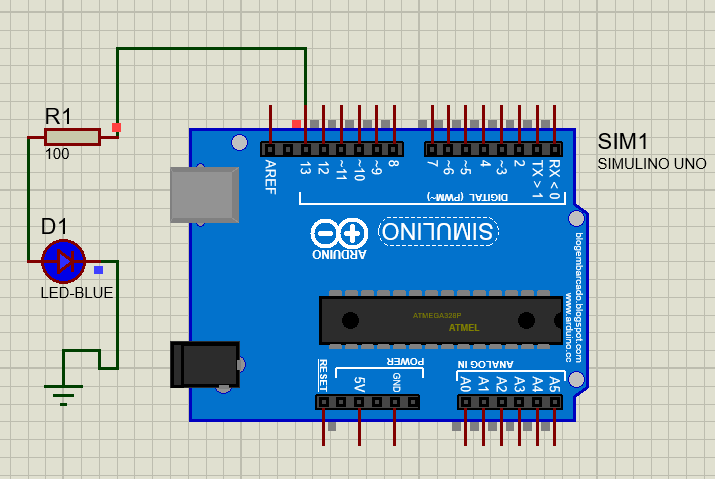
## Mô tả:

Bài này, thực hiện việc tự động nháy đèn led sau khoản thời gian 1 giây. Led được đấu vào cổng 13 của Board mạch.

## Đặc điểm linh kiện:

* Led xanh.
* Điện trở R1 100 Ω.

## Sơ đồ thiết kê:



Hình -Sơ đồ mạch bài 1

## Mã lệnh chính:

|  |
| --- |
| int LED=13;  void setup() {  pinMode(LED, OUTPUT);// Led đấu vào cổng 13  }  void loop() {  digitalWrite(LED, HIGH); // Mở led (HIGH điện thế ở mức cao)  delay(1000); // đợi 1 giây  digitalWrite(LED, LOW); // Tắt Led (Low điện thế ở mức thấp)  delay(1000;  } |

# Bài 2: Nháy Led bằng nút

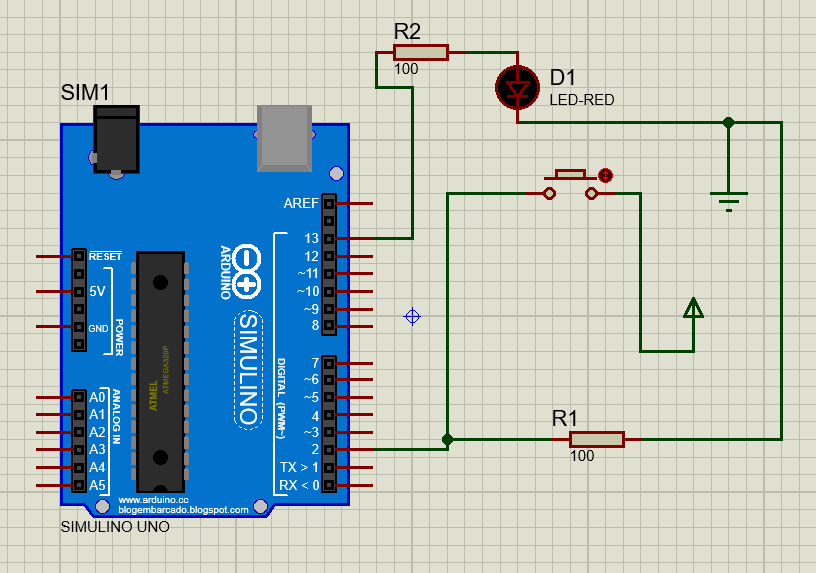
## Mô tả:

Bài này, thực hiện nháy led bằng cách ấn nút.Led được đấu vào cổng 13 của broad mạch và nút ấn được đấu vào cổng số 2.Nút được đấu vào cổng 2 của board mạch.

## Đặc điểm linh kiện:

* Led đỏ.
* Nút (button).
* Điện trở R1,R2 100 Ω .

## Sơ đồ thiết kế:



Hình -Sơ đồ mạch bài 2

## Mã lệnh chính:

|  |
| --- |
| const int buttonPin = 2;  const int ledPin = 13;  int buttonState = 0;  void setup() {  pinMode(ledPin, OUTPUT);  pinMode(buttonPin, INPUT); //nhận giá trị vào của Nút  }  void loop() {  buttonState = digitalRead(buttonPin); //Đọc giá trị của cổng số 2 của nút  if (buttonState == HIGH) { // Kiểm tra nếu nút được bấm thì buttonState là HIGH  digitalWrite(ledPin, HIGH);  } else {  digitalWrite(ledPin, LOW);  }  } |

# Bài 3: Led sáng dần

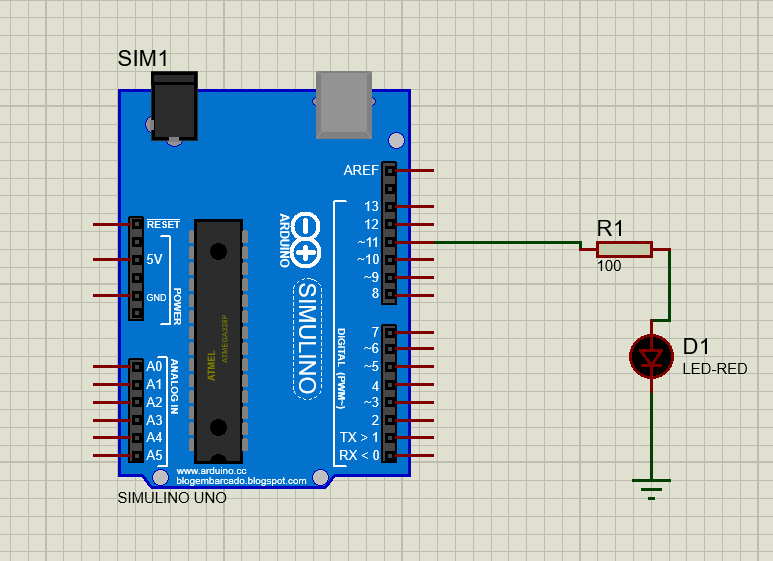
## Mô tả:

Bài này, thực hiện mở led sáng dần và tắt dần.Led được đầu vào cổng 11 của broad mạch có thể điều chế độ rộng xung.

## Đặc điểm linh kiện:

* Led Red đấu vào cổng 11 có thể điều chế độ rộng xung từ (0-255).
* Điện trở R1 100 Ω.

## Sơ đồ thiết kế:



Hình -Sơ đồ mạch led sáng dần

## Mã lệnh chính:

|  |
| --- |
| int brightness = 0;// Mức độ sáng  int LED=11;  void setup()  {  pinMode(LED, OUTPUT);  }  void loop()  {  for (brightness = 0; brightness <=255 ; brightness +=5){// tăng dần độ sáng của led  analogWrite(LED,brightness);  delay(30);// Độ trễ 30ms  }  for (brightness = 255; brightness <=0 ; brightness -=5) { //giảm dần độ sáng của led  analogWrite(LED,brightness);  delay(30);  }  } |

# Bài 4: Led Chiết áp

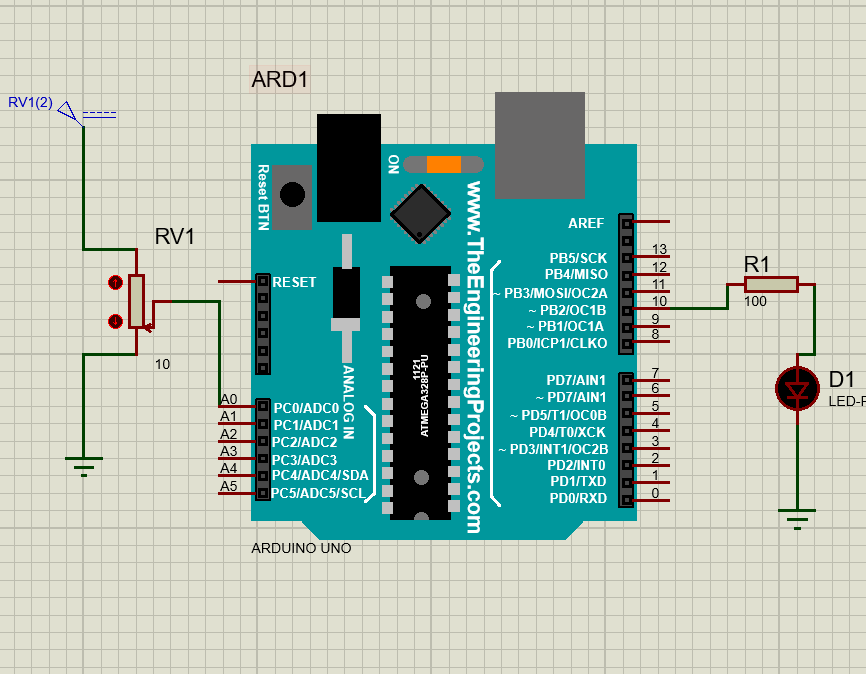
## Mô tả:

Bài này, thực hiện mở led sáng bằng cách sư dụng chiết áp.Led được đấu vào cổng 11 broad mạch có thể điều chế độ rộng xung.Chiết áp được đấu vào cổng A0 của broad mạch.

## Đặc điểm linh kiện:

* Led red đấu vào cổng 11 có thể điều chế độ rộng xung từ (0-255).
* Chiết áp POT gồm 10 mức có thể điều chỉnh từ (0-100Ω) , mỗi mức tăng 10Ω.Giá trị Input từ (0-1023) tương ứng với 10 mức .
* Điện trở R1 100Ω.

## Sơ đồ thiết kế:



Hình -Sơ đồ mạch Led chiết áp

## Mã lệnh chính:

|  |
| --- |
| int X= 0;  int Led = 10;  void setup()  {  pinMode(Led, OUTPUT);  }  void loop()  {  X=analogRead(A0);  int brightness= map(X,0,1023,0,255);// map biến đối giá trị input thành mức sáng  analogWrite(Led,brightness); delay(200);} |

# Bài 5: Cảm biến Âm thanh

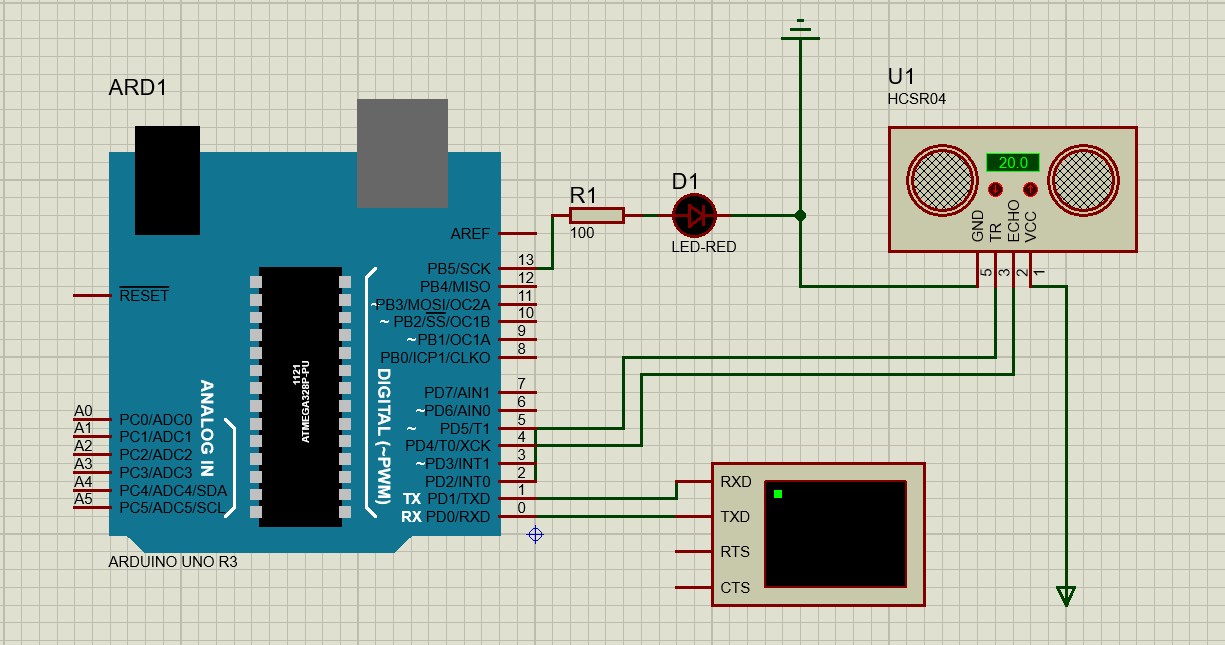
## Mô tả:

Bài này, sử dụng cảm biến âm thanh Ultrasonic HR04 để tính khoảng cách nhất định để làm Led sáng và sử dụng màng hình để hiện thị khoảng cách.Led được đấu vào cổng 13 của mạch.

## Đặc điểm linh kiện:

* Led Red
* Điện trở R1 100Ω.
* Cảm biến âm thanh HCSR04 thang đo từ 2-300 cm gồm 4 chân Vcc, Gnd , Trig, Echo. Để đo khoảng cách, chân Trig sẽ phát ra xung ngắn (10 ms) sau đó chân Echo sẽ nhận được sóng âm này.Quảng đường âm đi được sẽ bằng thời gian âm đi được nhân với vận tốc âm thanh (340 m/s), tương đương 0.034 cm/microsecond ((340\*100)/10^6)
* Màng hình hiện thị được nối tiếp với mạch.

## Sơ Đồ thiết kế:



Hình -Sơ đồ mạch cảm biến âm thanh

## Mã lệnh chính:

|  |
| --- |
| const int Echo = 2;  const int Trig = 3;  int led = 13 ;  void setup()  {  pinMode(led, OUTPUT);  Serial.begin(9600); // Mở serial  pinMode(Trig, OUTPUT); // chân trig là phát xung  pinMode(Echo, INPUT); // chân echo là chân nhận xung  }  void loop()  {  float duration, inches, cm;  digitalWrite(Trig, LOW);//Tắt chân trig  delayMicroseconds(2);// Độ trễ 2 microsecond  digitalWrite(Trig, HIGH);// chân Trig phát xung  delayMicroseconds(10);  digitalWrite(Trig, LOW);  duration = pulseIn(Echo, HIGH); // hàm pulseIn đọc thời gian xung đã đi  cm = duration \* 0.034 / 2; ; //tốc độ âm thanh 0.034 cm/microsecond, quảng đường âm đi chia 2 ta được khoảng cách đến vật.  inches = cm\*0.39;  Serial.print(inches);// In ra khoảng cách theo thang đo inches.  Serial.print("in, ");  Serial.print(cm);  Serial.print("cm");// In ra khoảng cách theo thang đo cm.  Serial.println();  if (inches < 10) digitalWrite(led, HIGH); // khoảng cách bé 10 inches thì led sáng  else digitalWrite(led, LOW);  delay(1000);  } |

# Bài 6: Led RGB

## Mô tả:

Bài này sử dụng 3 biến trở để thay đổi 3 giá trị Red ,Blue ,Green của led RGB làm led sáng theo bảng màu tùy chọn dựa trên giá trị hiện thị trên màng hình hiển thị.Led RGB được đấu vào các chân có thể điểu chỉnh độ rộng xung trên bo mạch theo thứ tự Red cổng 11, Blue cổng 10, Green cổng 9.

## Đặc điểm linh kiện:

* Led RGB.
* Điện Trở R1,R2,R3 100Ω.
* Biến Trở RED, GREEN ,BLUE 1kΩ tăng giảm theo %.Biến trở sẽ nối lần lượt vào chân analog theo thứ tự A0,A1,A2.
* Màng hình hiện thị.

## Sơ Đồ thiết kế:

Hình -Sơ đồ mạch led rgb

## Mã lệnh chính:

|  |
| --- |
| int x,y,z; //Mức độ sáng  const int red=11; // Chân Red của led RGB  const int blue=10; // Chân Blue của led RGB  const int green=9; // Chân Green của led RGB  void setup()  {  Serial.begin(9600); //Tốc độ truyền nối tiếp 9600  pinMode(red, OUTPUT);  pinMode(blue, OUTPUT);  pinMode(green, OUTPUT);  }  void loop()  {  x=map(analogRead(A0),0,1023,0,255); // Đổi giá trị input của biến trở (0-1023) sang mức sáng (0-255)  y=map(analogRead(A1),0,1023,0,255);  z=map(analogRead(A2),0,1023,0,255);  analogWrite(red, x);  analogWrite(blue, y);  analogWrite(green, z);  Serial.println(x);  Serial.println(y);  Serial.println(z);  delay(100);  } |

# Bài 7: Cảm biến nhiệt độ

## Mô tả:

Bài này sử dụng cảm biến nhiệt đô TMP-36 để làm led sáng mức nhiệt độ nhất định.Chân Vout của cảm biến đấu vào chân analog A0 của mạch,LED được đấu vào chân 13 của broad mạch và sử dụng màng hình để hiện thị nhiệt độ.

## Đặc điểm linh kiện:

* LED Red
* Cảm biến nhiệt độ TMP-36 , có thể đo nhiệt độ từ -40ºC - 150ºC .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), 0.1V (-40°C) đến 2.0V (150°C) nhưng độ chính xác sẽ giảm khi trên 125°C.
* Chân số 1 là chân cấp nguồn 5V (chân này bạn có thể cắm vào nguồn 5V của Arduino khi sử dụng nó với Arduino).
* Chân thứ 2 là chân xuất tín hiệu tương tự (tín hiệu dạng xung) đấu vào chân analog A0 của broad mạch
* 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.
* Màng hình Serial.
* Biến trở R1 100Ω.

## Sơ đồ thiết kế:

Hình -Sơ đồ mạch cảm biến nhiệt

## Mã lệnh chính:

|  |
| --- |
| #define SENSOR\_PIN A0 // Input của Cảm biến  #define LED 12  float voltage = 0;  float sensor = 0;  float celsius = 0;  float fahrenheit = 0;  void setup() {  Serial.begin(9600); // Bật serial monitor  pinMode(LED,OUTPUT);  }  void loop()  {  sensor = analogRead(SENSOR\_PIN);  voltage = (sensor\*5000)/1024; // Chuyển đổi tín hiệu cảm biến sang mili Volt  voltage = voltage-500; // Trừ đi điện áp bù  celsius = voltage1/10; // Chuyển đổi mV sang độ C  fahrenheit = ((celsius \* 1.8)+32); // Đổi độ C sang độ F  Serial.print(celsius,2);  Serial.println(" do C");  Serial.print(fahrenheit,2);  Serial.println(" do F");  if(celsius >30) // Nhiệt độ > 30°C thì LED sáng  {  digitalWrite(LED,HIGH);  }  else  digitalWrite(LED,LOW);  delay (1000);  } |

# Bài 8: LED 7 Seg 1 digit BCD

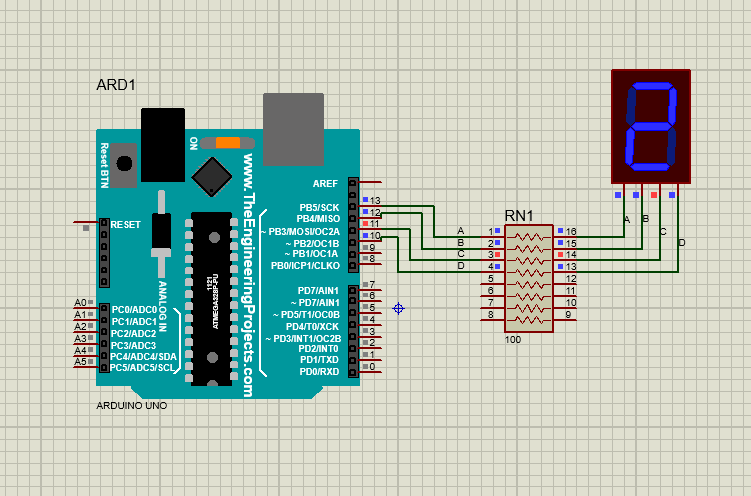
## Mô tả:

Bài này sử dụng Led 7 seg 1 digit tích hợp BCD để hiện số từ 0 đến 3.Chân 13,12,11,10 được nối với mạch tương ứng với chân A,B,C,D của led.

## Đặc điểm linh kiện:

* Biến trở RN1 gồm 8 vào và 8 chân ra mức điện trở là 100Ω.
* Led 7 seg 1 digit BCD blue , các số sẽ được chuyển đổi sang dạng nhịn phân, với các chân ra 13,12,11,10 tương ứng với chân A,B,C,D của Led .
* Khi có dữ liệu đi vào các chân A,B,C,D thì led sẽ tự động giải mã các dữ liệu đó sang các chân a,b,c,d,e,f,g của Led 7 seg để có thể sáng theo số mà dữ liệu thập phân đưa vào.

## Sơ đồ thiết kế:



## Mã lệnh chính:

|  |
| --- |
| int A=13, B=12, C=11, D=10;  void setup(){  pinMode(A,OUTPUT);  pinMode(B,OUTPUT);  pinMode(C,OUTPUT);  pinMode(D,OUTPUT);  }  void KHONG(){  digitalWrite(A,LOW);  digitalWrite(B,LOW);  digitalWrite(C,LOW);  digitalWrite(D,LOW);  }  void MOT(){  digitalWrite(A,LOW);  digitalWrite(B,LOW);  digitalWrite(C,LOW);  digitalWrite(D,HIGH);  }  void HAI(){  digitalWrite(A,LOW);  digitalWrite(B,LOW);  digitalWrite(C,HIGH);  digitalWrite(D,LOW);  }  void BA(){  digitalWrite(A,LOW);  digitalWrite(B,LOW);  digitalWrite(C,HIGH);  digitalWrite(D,HIGH);  }  void loop(){  KHONG();  delay(1000);  MOT();  delay(1000);  HAI();  delay(1000);  BA();  delay(1000);  } |

# Bài 9: LED 7 SEG 2 DIGIT Cathode

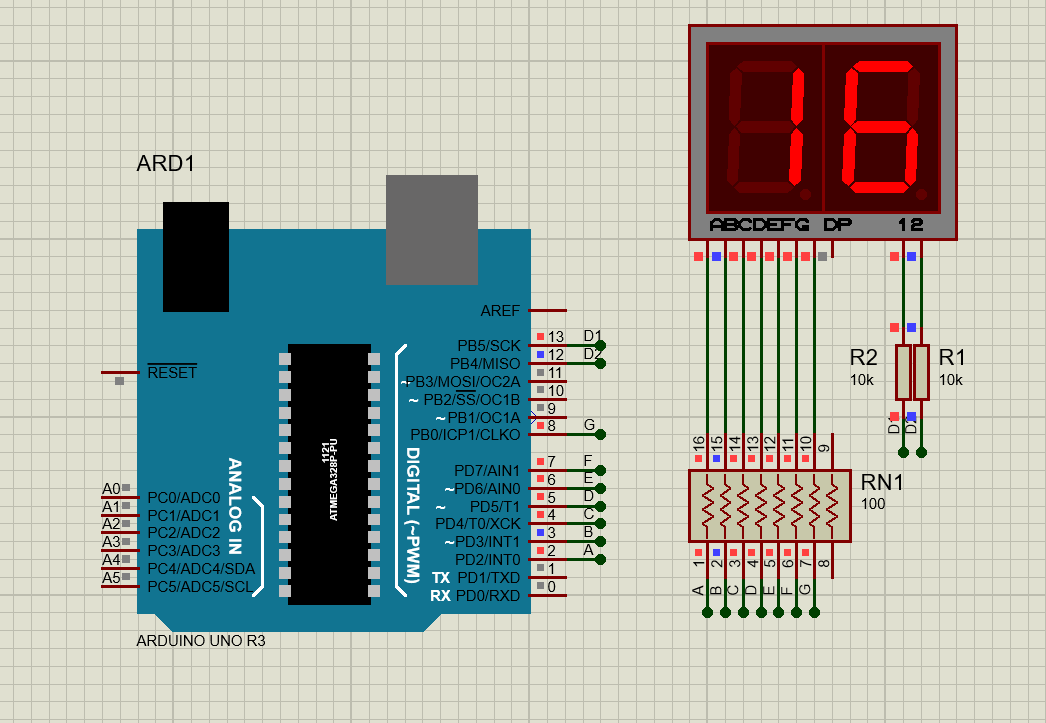
## Mô tả:

Bài này sử dụng Led 7 Seg 2 digit để hiên thi số từ 0 đến 99.

## Đặc điểm linh kiện:

* RN1 điện trở gồm 8 chân và 8 chân ra, mức điện trở là 100Ω.
* Led 7 seg 2 digit cathode gồm các chân a,b,c,d,e,f,g,DP,1,2
* Gồm 2 digit 1 số ở sau và digit 2 số ở trước mỗi digit sẽ được hiện thị nếu chân 1 ở mức high đối với digit 1 và chân 2 ở mức high đối với chân digit 2.
* Các chân a,b,c,d,e,f,g là các chân ứng với các bóng led nhỏ ở trong đèn

## Sơ đồ thiết kế:



## Mã lệnh chính:

|  |
| --- |
| int digit[10] = {0b0111111, 0b0000110, 0b1011011, 0b1001111, 0b1100110, 0b1101101, 0b1111101, 0b0000111, 0b1111111, 0b1101111};  int digit1, digit2;//các chân enable digit1 và digit2  void setup()  {  for (int i = 2; i < 9; i++) // setup các chân a,b,c,d,e,f,g  {  pinMode(i, OUTPUT);  }  pinMode(12, OUTPUT);//chân enable digit 1  pinMode(13, OUTPUT);//chân enable digit 2  }  void loop() {  for (int j = 0; j <= 99; j++) // hiện thị số từ 0 đến 99  {  digit2 = j / 10;  digit1 = j % 10;  for ( int k = 0; k < 20; k++)  {  digitalWrite(12, HIGH);//hiện thị số ở trước  digitalWrite(13, LOW);  dis(digit2);  delay(10);  digitalWrite(13, HIGH);//hiện thị số ở sau  digitalWrite(12, LOW);  dis(digit1);  delay(10);  }  }  }  void dis(int num)  {  for (int i = 2; i < 9; i++)  {  digitalWrite(i, bitRead(digit[num], i - 2)); //bitRead trả về giá trị tại một bit i của một số nguyên tương ứng với các chân a,b,c,d,e,f,g tương ứng vơi i chạy từ 2-9 đổi với arduino  }  } |

# Bài 10: Điều khiển động cơ motor DC

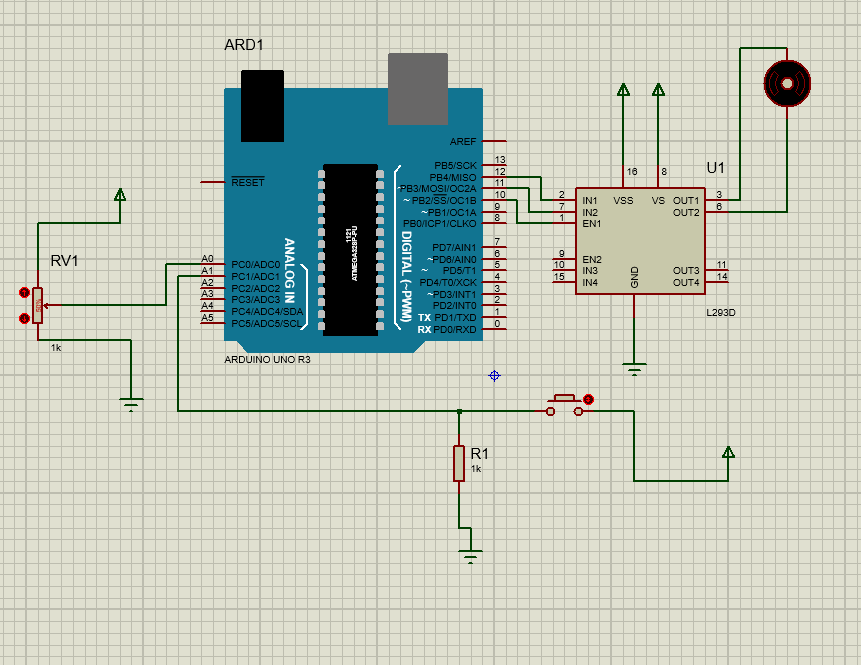
## Mô tả:

Bài nảy sử dụng để điều khiển động cơ, biến trở RV1 để điều khiển tốc độ của động cơ, nút bấm để điều khiển động cơ đi tới và đi lùi và L293D dùng để nối với mạch để điều khiển các thông số truyền vào từ Andurno

## Đặc điểm linh kiện:

* L293D
* Có thể được sử dụng để chạy hai động cơ DC với cùng một IC.
* Có thể kiểm soát tốc độ và hướng IN1,IN2 chỉnh hướng, EN1 chỉnh tốc độ.
* Điện áp động cơ Vcc2 (Vs): 4,5V đến 36V.
* Dòng động cơ cao nhất tối đa: 1.2A.
* Dòng động cơ liên tục tối đa: 600mA.
* Điện áp cung cấp cho Vcc1 (vss): 4,5V đến 7V.
* Motor DC động cơ 2 chiều có quay theo chiều kim động hồ và ngược lại
* Biến trở 1000Ω.
* Nút bấm để điều khiển hướng của động cơ.
* Điện trở R1 1000Ω.

## Sơ đồ thiết kế:



## Mã lệnh chính:

|  |
| --- |
| #define motorPin1 12 // L293D Input 12  #define motorPin2 11 // L293D Input 11  #define speedPin 10 // L293D enable chân 10  // chân biến trở nối với A0  int Mspeed = 0;  void setup() {  pinMode(motorPin1, OUTPUT);  pinMode(motorPin2, OUTPUT);  pinMode(speedPin, OUTPUT);  Serial.begin(9600);  }  void loop() {  int switchPin= analogRead(A1);// nút bấm khổi động động cơ  Mspeed = analogRead(A0)/4;// tốc độ của động cơ  int x=digitalRead(2);  analogWrite (speedPin, Mspeed);//Mspeed từ 0-255  Serial.println(switchPin);  Serial.println("Toc Do: ");  Serial.println(Mspeed);  delay(1000);  if (digitalRead(switchPin)) // chân switchPin =1 thì động cơ đi lùi và ngược lại  {  digitalWrite(motorPin1, LOW);  digitalWrite(motorPin2, HIGH);  }  else {  digitalWrite(motorPin1, HIGH);  digitalWrite(motorPin2, LOW);  }  } |

# Bài 11: Cảm biến DHT11 độ ẩm và nhiệt độ và relay

## Mô tả:

Bài này sử dụng cảm biến độ ẩm và nhiệt độ DHT11 để tắt mở đèn dây tóc sử dụng relay để cấp nguồn điện ngoài cho đèn dây tóc.

## Đặt điểm linh kiện:

* Relay RL1 5V.
* DHT11
* Mức đo 20-90%RH ,0-50 ℃ .
* Sai số ±5％RH và ±2℃
* Gồm 3 chân VDD(nguồn),GND đất , Data(chân dữ liệu)
* L1 12V bóng đèn dây tóc.
* LED red.
* Điện trở R1 200Ω.
* V1 nguồn ngoài 12V.

## Sơ đồ thiết kế:



## Mã lệnh chính:

|  |
| --- |
| #include "DHT.h"  #define DHTPIN 2 //chân data  #define DHTTYPE DHT11 //chọn loại cảm biến DHT 11 hoặc DHT22  DHT dht(DHTPIN, DHTTYPE);  void setup() {  Serial.begin(9600);  Serial.println(F("DHTxx test!"));  pinMode(12,OUTPUT);  pinMode(11,OUTPUT);  dht.begin();  }  void loop() {  delay(2000);  float h = dht.readHumidity();  // đọc giá trị của độ ẩm  float t = dht.readTemperature();  // đọc giá trị của nhiệt độ theo độ C  float f = dht.readTemperature(true); // đọc giá trị nhiệt độ theo F(isFahrenheit = true)  if (isnan(h) || isnan(t) || isnan(f)) {  Serial.println(F("Failed to read from DHT sensor!"));  return;  }  // tính giá trị tương đối của nhiệt độ theo độ F  float hif = dht.computeHeatIndex(f, h);  // tính giá trị tương đối của nhiệt độ theo độ C (isFahreheit = false)  float hic = dht.computeHeatIndex(t, h, false);  Serial.print("Humidity: ");  Serial.print(h);  Serial.print("% Temperature: ");  Serial.print(t);  Serial.print("\*C ");  Serial.print(f);  Serial.print("\*F Heat index: ");  Serial.print(hic);  Serial.print("\*C ");  Serial.print(hif);  Serial.println("\*F");  //nhiệt độ lơn hơn 30 độ C thì bóng đèn 12V sáng  if(t>30)  {  digitalWrite(11,HIGH);  delay(100);  }  else digitalWrite(11,LOW);  // độ ẩm bé hơn 30% thì đèn led sáng  if(h<30){  digitalWrite(12,HIGH);  }  else digitalWrite(12,LOW);  } |

# Bài 12: Sử dụng keypad để hiện thị mật khẩu bằng màng hình LCD

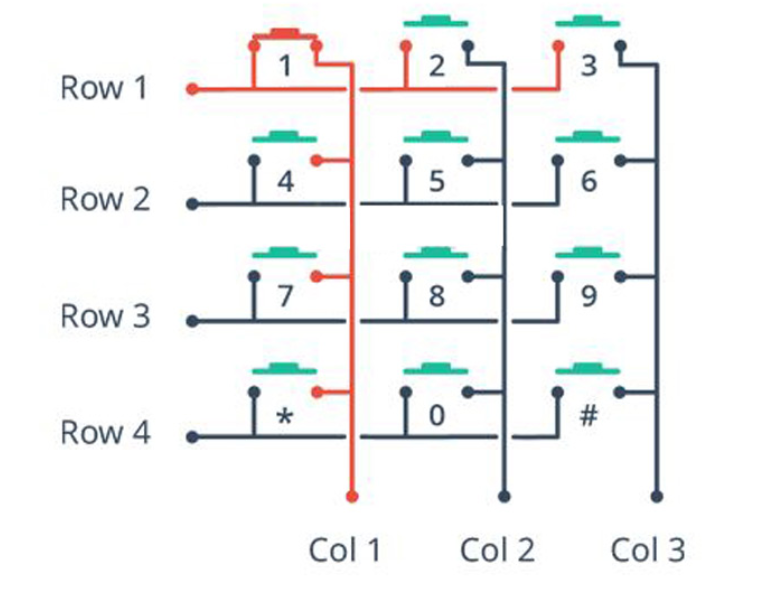
## Mô tả:

Bài này sử dụng keypad và màng hình LCD 16\*2 để hiện thị màng hình đăng nhập. Khi nhập đúng mật khẩu thì màng hình sẽ hiện thị đăng nhập thành công và Led sẽ sáng .

## Đặc điểm linh kiện:

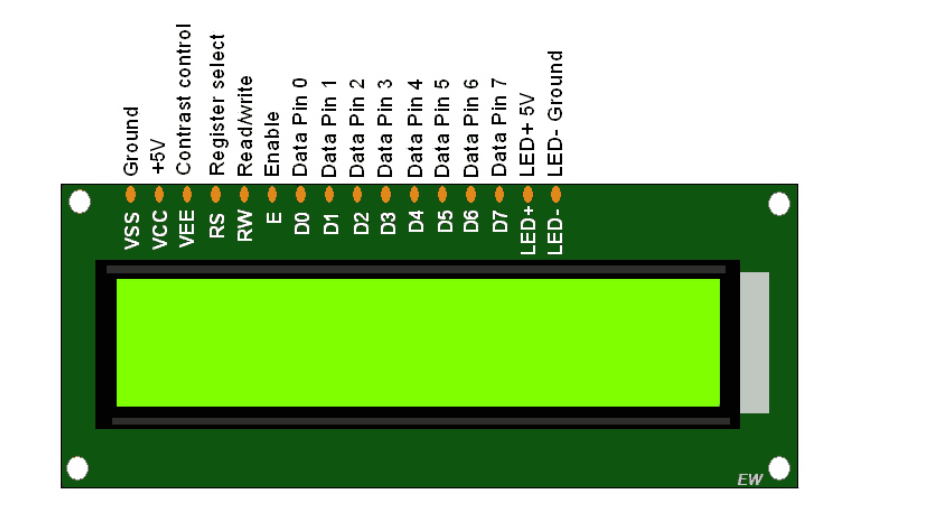
-Key Pad

Bàn phím 4 × 3 có tổng cộng 12 nút ở dạng Ma trận. 4 hàng là đầu vào và 3 cột là đầu ra. Mỗi nút được kết nối từ bên này sang hàng và từ bên kia sang cột. Ví dụ: nếu chúng ta nhấn công tắc số 1, đầu vào của hàng này được lưu ở đầu ra của cột của nó. Hình ảnh bên dưới cho thấy mạch bên trong của bàn phím này.

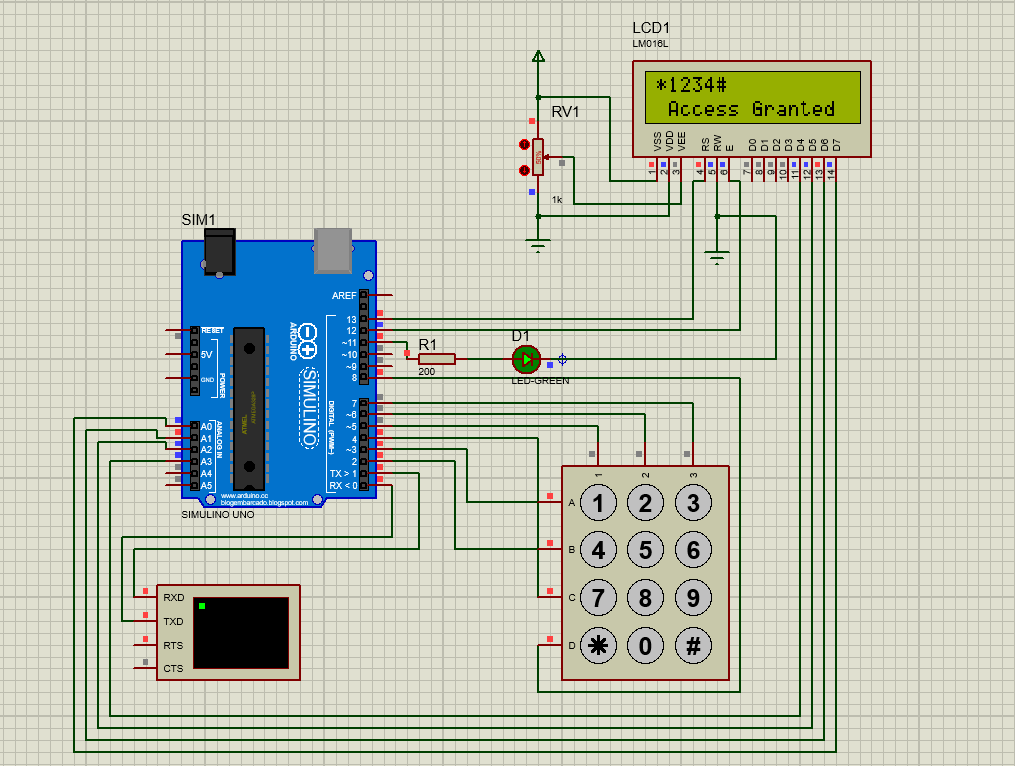


-LCD:

* Điện áp MAX : 7V
* Điện áp MIN : - 0,3V
* Điện áp ra mức thấp : <0.4V
* Điện áp ra mức cao : > 2.4
* Hoạt động ổn định : 2.7-5.5V
* Dòng điện cấp nguồn : 350uA - 600uA
* Nhiệt độ hoạt động : - 30 - 75 độ C



## Sơ đồ thiết kế:



## Mã lệnh chính:

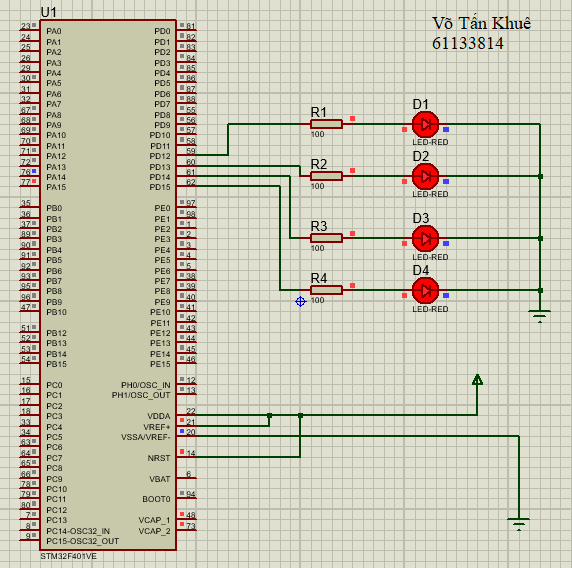
|  |
| --- |
| #include <LiquidCrystal.h> //các thư viện hỗ trợ  #include <Keypad.h>  const int rs = 13, en = 12, d4 = A3, d5 = A2, d6 = A1, d7 = A0;  LiquidCrystal lcd(rs, en, d4, d5, d6, d7);  const byte ROWS = 4; //4 hàng, 3 cột của key pad  const byte COLS = 3;  char keys[ROWS][COLS] = {  {'1','2','3'},  {'4','5','6'},  {'7','8','9'},  {'\*','0','#'}  };  byte rowPins[ROWS] = {3, 2, 4, 8}; //các chân kết nối hàng của key pad  byte colPins[COLS] = {5, 6, 7}; //các chân kết nói cột của kedpad  int LCDCol = 0;//cột của LCD  int LCDRow = 0;//Hàng cua Lcd  Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );  String v\_passcode="";//khởi tạo password  void setup(){  Serial.begin(9600);  lcd.begin(16, 2);  lcd.setCursor(LCDCol, LCDRow);  pinMode(11,OUTPUT);  }  void loop(){  char key = keypad.getKey();  if (key){  if ( LCDCol > 5 )//nếu nhập quá 5 kí tự thì tự động xuống dòng  {  ++LCDRow;  if (LCDRow>1)//reset lại Lcd  { LCDRow=0; LCDCol = 0 ; lcd.clear(); }  LCDCol = 0 ;  }  lcd.setCursor (LCDCol, LCDRow);  lcd.print(key);//hiện thị mật khẩu  ++LCDCol;  }  if (key != NO\_KEY){  v\_passcode = v\_passcode + key;  if(key=='\*')//bấm nút \* trên key pad để nhập nhật khẩu  { lcd.setCursor(1,2);  lcd.print("Enter Password");  v\_passcode="";  }  if(key=='#')//bấm nút # trên key pad để kiểm tra mật khẩu  {  Serial.println("Validate the Password");  if (v\_passcode=="1234#")  {  lcd.setCursor(1,2);  lcd.print("Access Granted");  digitalWrite(11,HIGH);  }  else  {  lcd.setCursor(1,2);  lcd.print("Access Denied");  delay(2000);  LCDRow=0; LCDCol = 0 ; lcd.clear();  }  }  }  } |

# Bài 13. Nháy 4 đèn Led thông qua STM32

## 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



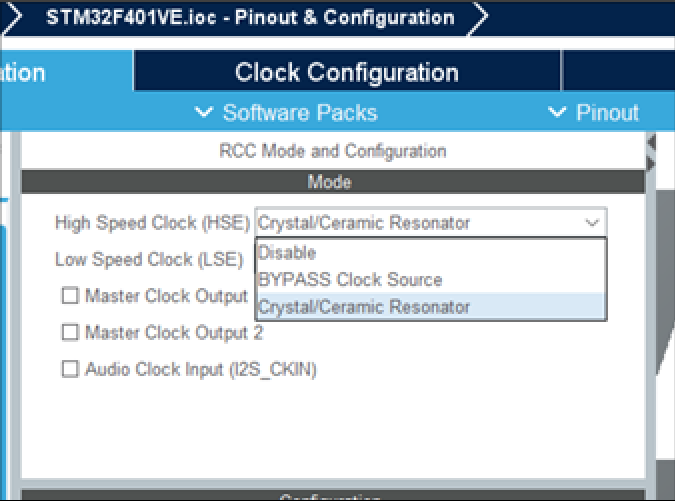
Hình . Sơ đồ mạch Proteus

## Đặc điểm của linh kiện

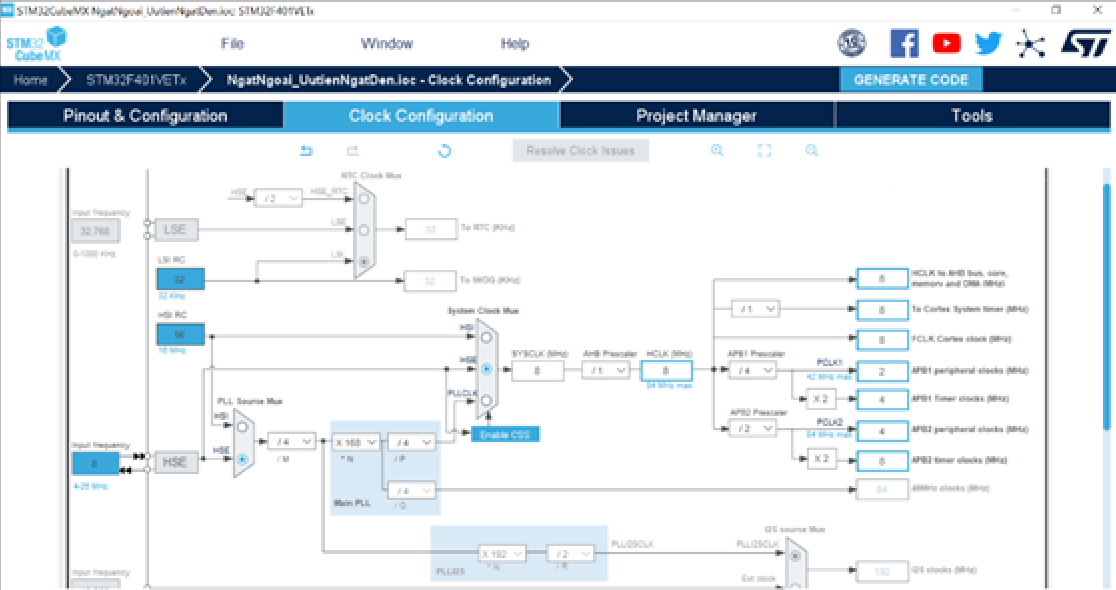
* 4 điện trở: 100 Ω
* 4 đèn Led
* 1 STM32F401VE
* Các cấu hình với CubeMX
* Ở Tab **PINOUT** chọn các chân PD12, PD13, PD14, PD15 có chức năng **“GPIO\_Output”**



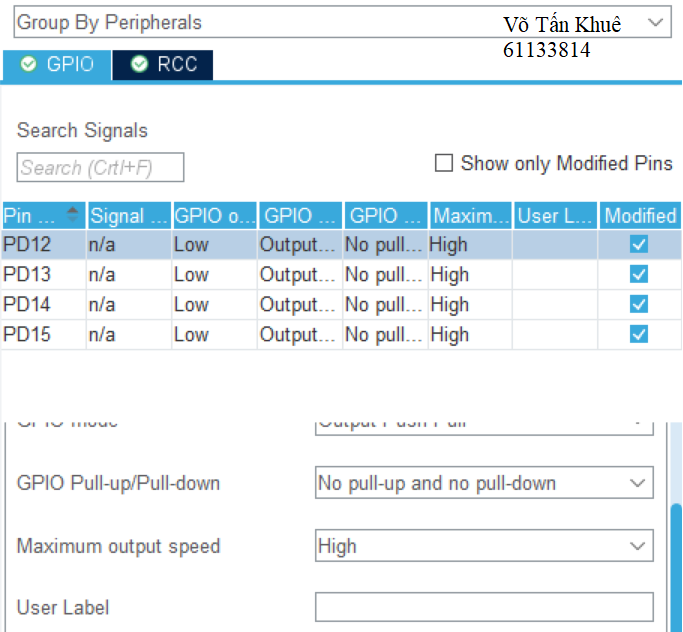
* Cấu hình chọn nguồn xung cho chip hoạt động với thạch anh ngoại được gắn sẵn trên board mạch **RCC → High Speed Clock (HSE)** và chọn “**Crystal/Ceramic Resonator”**



* Ở Tab **Clock Configuration** đặt **Input frequency** = 8 (thạch anh hàn sẵn trên board là loại 8 Mhz) và điền “8” tại mục “HCLK” (đây là tần số hoạt động tối đa của chip)



* Ở Tab **GPIO** ta chọn:
* **GPIO output level**: Low (cấu hình ban đầu cho các chân đang ở mức thấp)
* **GPIO mode**: Output Push Pull
* **GPIO Pull-up/Pull-down**: No pull-up and no pull-down (không cần điện trở kéo lên và kéo xuống)
* **Maximum output speed**: High



## Code chương trình

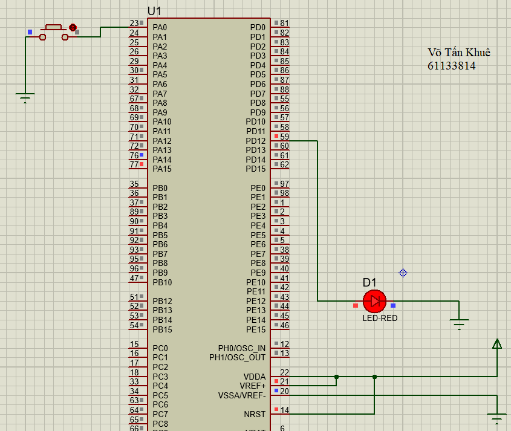
|  |
| --- |
| int main(void)  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  while (1)  {  //Hàm đảo trạng thái các chân 12,13,14,15  HAL\_GPIO\_TogglePin(GPIOD,GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15);  HAL\_Delay(1000); //delay 1s  }  } |

# Bài 14. Ngắt ngoài trên STM32

## Mô tả

Hệ thống được thiết kế thực hiện việc lập trình điều kiển ngắt ngoài bật/tắt đèn Led trên STM32F401VE. Chúng ta dùng chân PA0 để kết nối với nút bấm. Khi ta nhấn một nút nhấn thì phát sinh sự kiện ngắt ngoài gửi vào vi điều khiển, khi đó vi điều khiện triệu gọi một chương trình con phục vụ ngắt để thực hiện bật/tắt Led ở PA12.

## Sơ đồ thiết kế

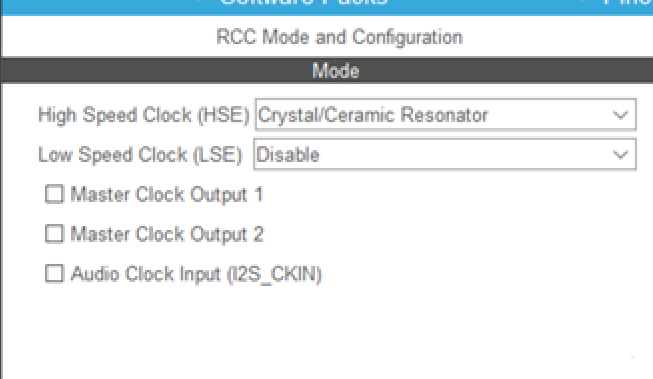


Hình . Sơ đồ mạch Proteus

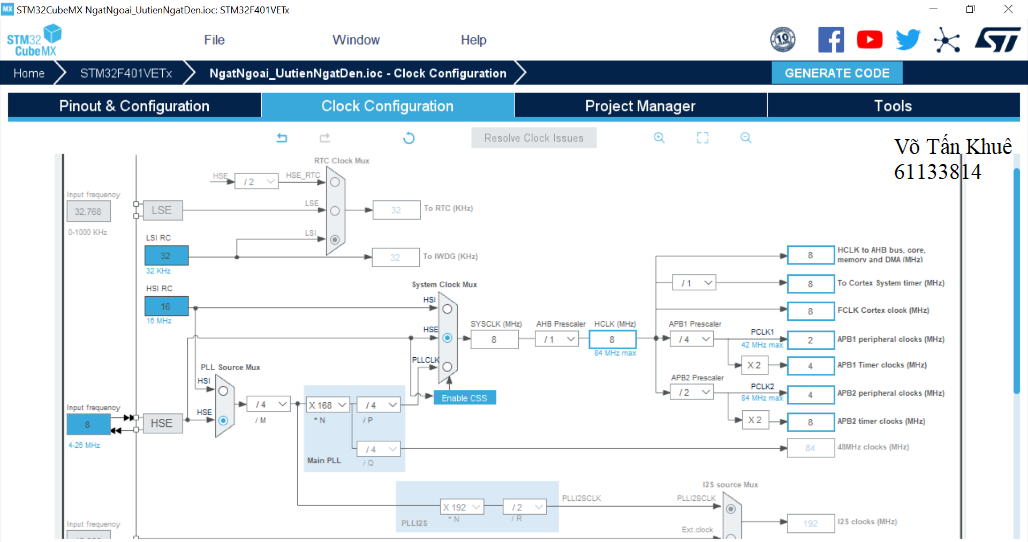
## Đặc điểm của linh kiện

* 1 đèn Led
* 1 Nút bấm
* 1 STM32F401VE
* Các cấu hình với CubeMX
* Chúng ta cấu hình thạch anh và xung Clock cho Chip:

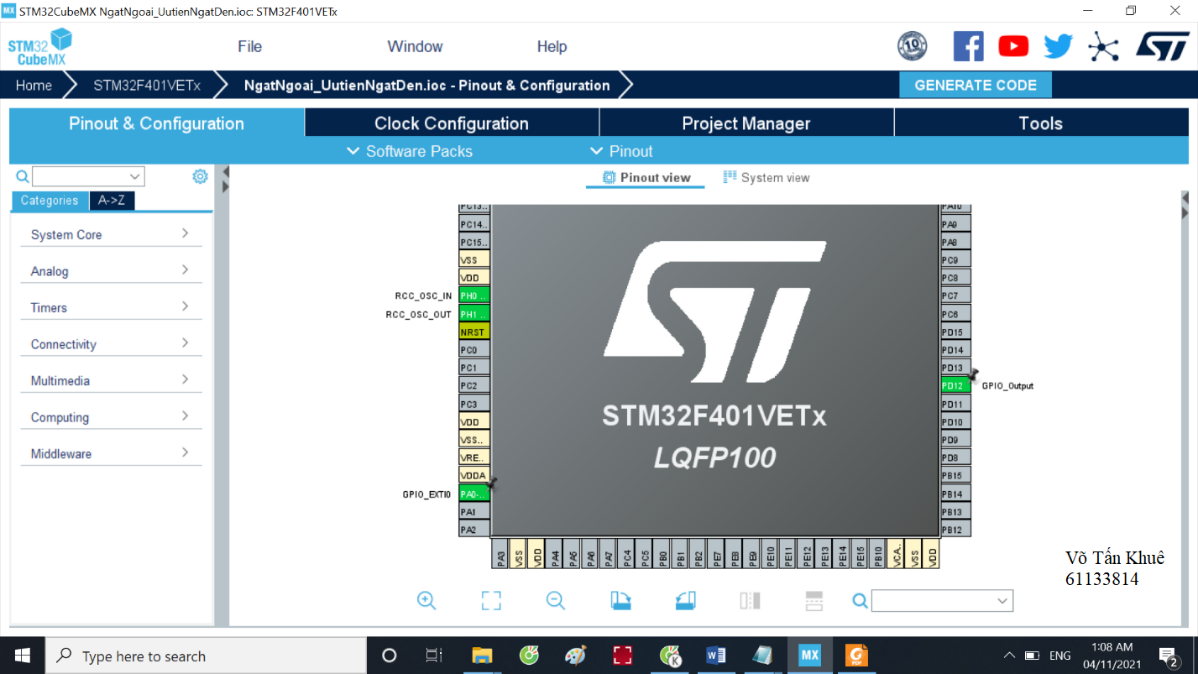
Tại mục **RCC → High Speed Clock (HSE)** và chọn **“Crystal/Ceramic Resonator”** Chức năng này sẽ giúp chip hoạt động với thạch anh ngoại được gắn sẵn trên board mạch.



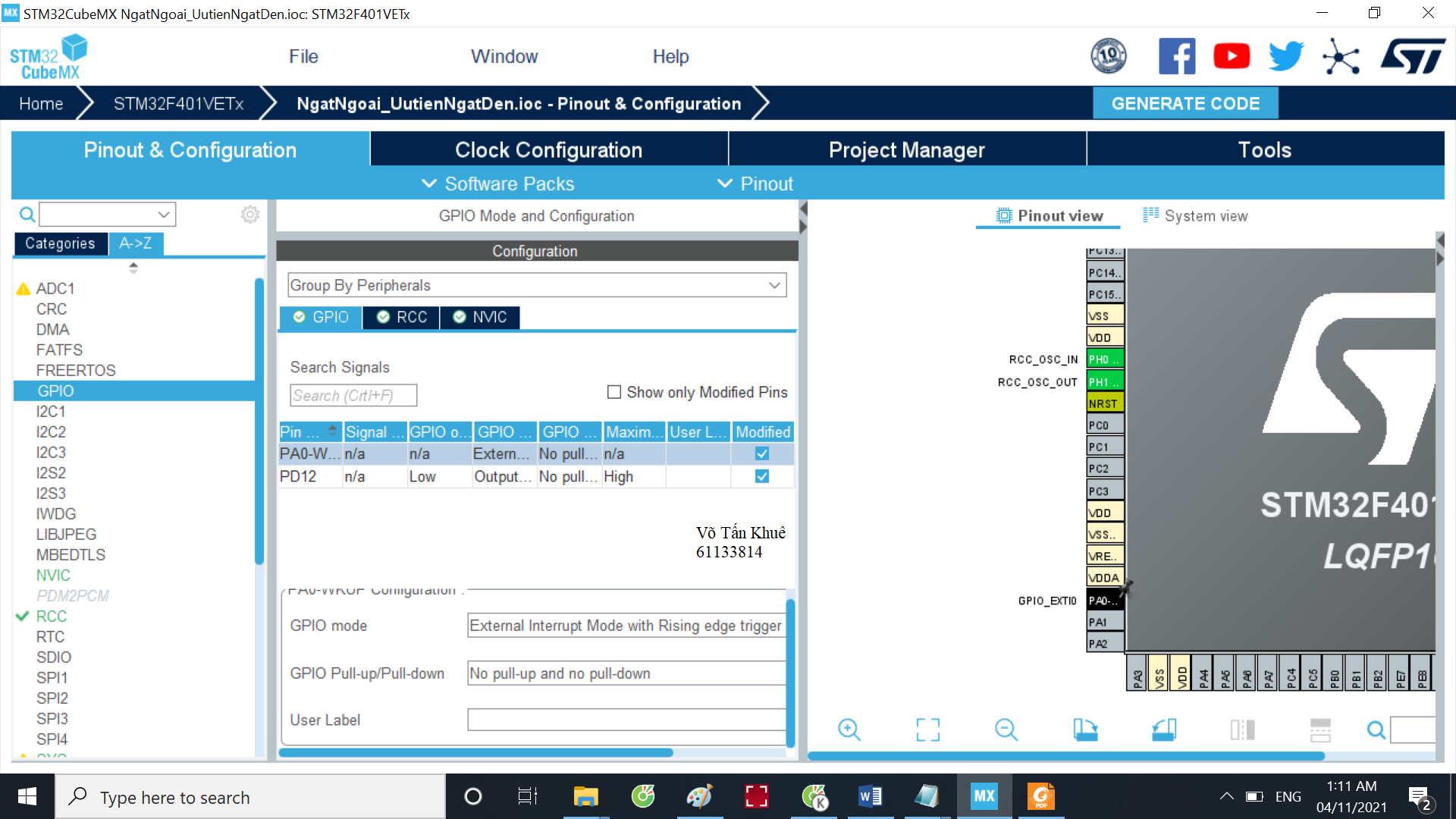
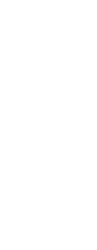
* Ở Tab **Clock Configuration** đặt **Input frequency** = 8 (thạch anh hàn sẵn trên board là loại 8 Mhz) và điền “8” tại mục “HCLK” (đây là tần số hoạt động tối đa của chip)



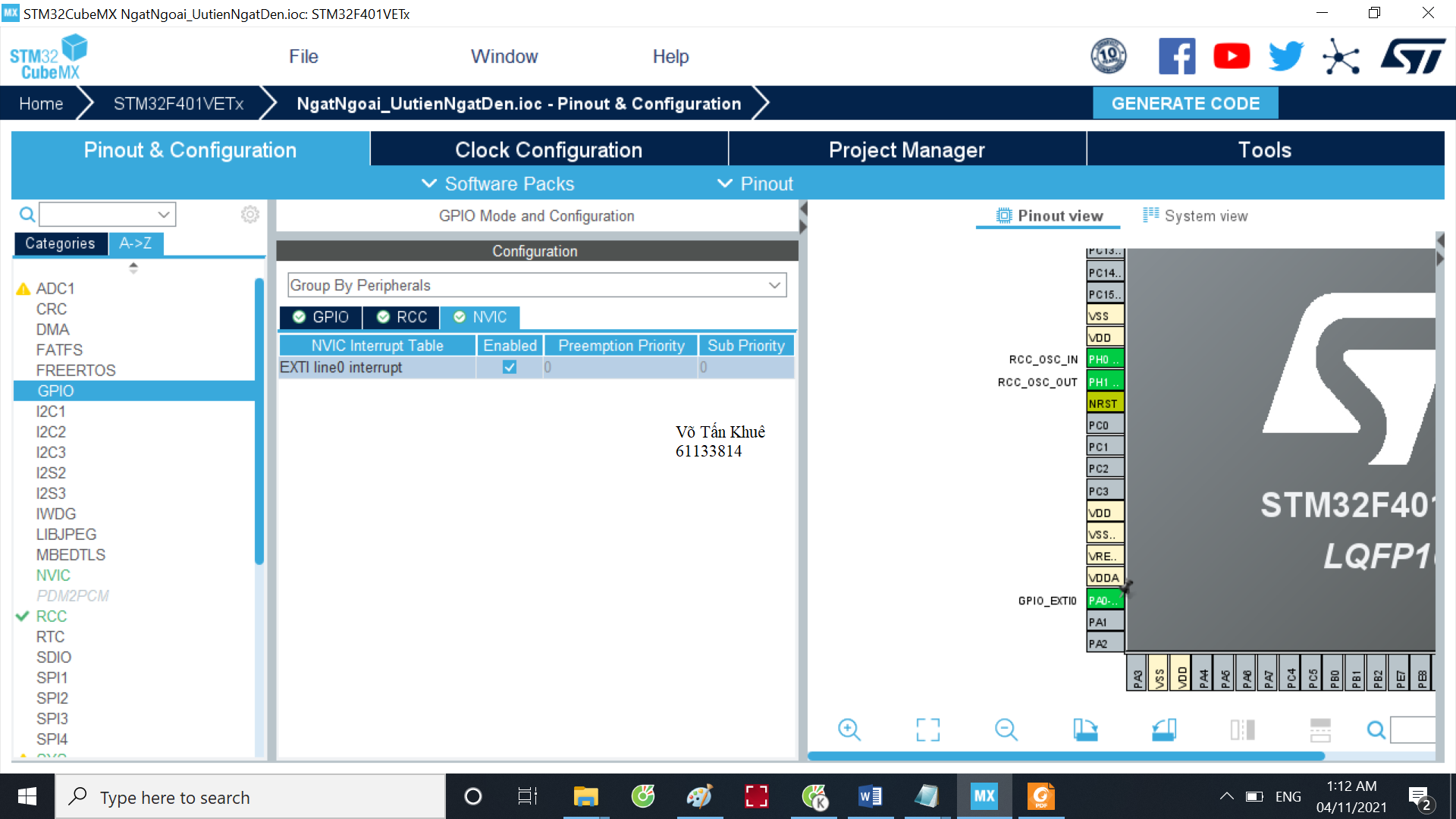
* Ở Tab **PINOUT**, mình sẽ cấu hình cho chân PA0 hoạt động với chức năng GPIO\_EXTI0 và cấu hình chân PD12 ở chế độ **“GPIO\_Output”**để quan sát sự hoạt động của ngắt.



* Cấu hình **GPIO** đối với Chân **P12**, chọn **Maximum output speed: High**, đối với chân **PA0** ta chọn **GPIO mode** là **External Interrupt Mode with Rising edge trigger detection**, dễ dàng nhận ra ở thời điểm nút nhả thì chân PA0 có mức ưu logic là 0, khi ta ấn nút thì chân PA0 lên mức logic 1.



* Sang phần **NVIC** chúng ta sẽ chọn **Enable** cho **EXTI line0 interrupt**



## Code chương trình

Trong hàm void **HAL\_GPIO\_EXTI\_Callback (uint16\_t GPIO\_Pin)** ta sẽ viết như sau:

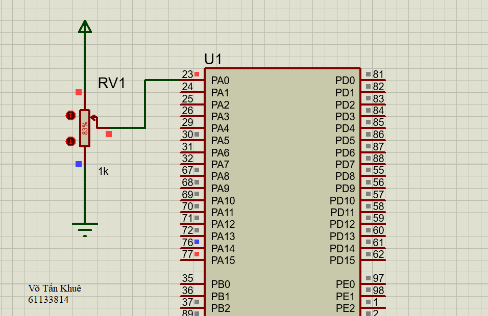
|  |
| --- |
| //Câu lệnh if (GPIO\_Pin == GPIO\_PIN\_0) sẽ giúp kiểm tra, phân luồng, phát hiện ngắt có //đúng đang sinh ra có phải ở chân 0 hay không.  void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin)  {  if(GPIO\_Pin == GPIO\_PIN\_0)  { HAL\_GPIO\_TogglePin(GPIOD, GPIO\_PIN\_12); }  } |

# Bài 15. Chức năng ADC trên vi điều khiển STM32F4

## Mô tả

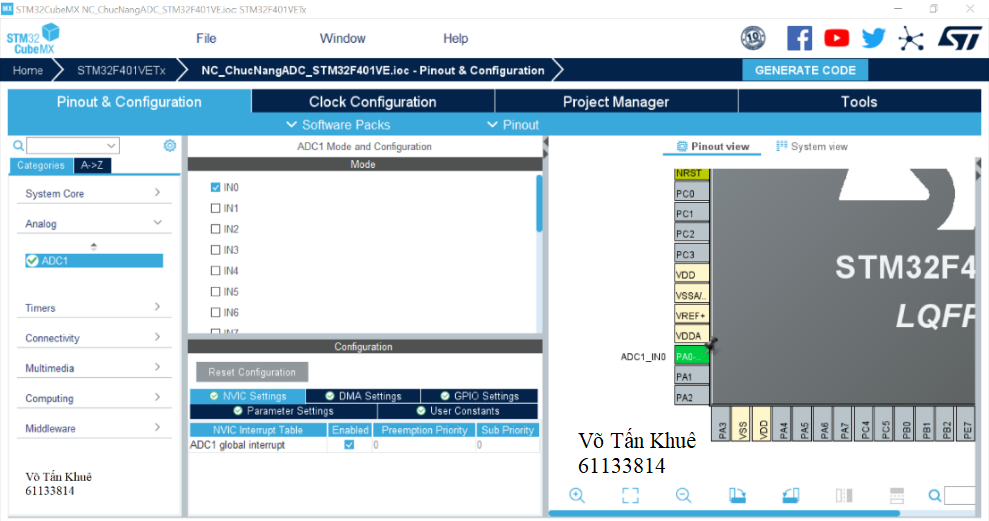
ADC (Analog-to-Digital Converter) bộ chuyển đổi tín hiệu tương tự - số là thuật ngữ nói đến sự chuyển đổi một tín hiệu tương tự thành tín hiệu số để dùng trong các hệ thống số (Digital Systems) hay vi điều khiển.

## Sơ đồ thiết kế



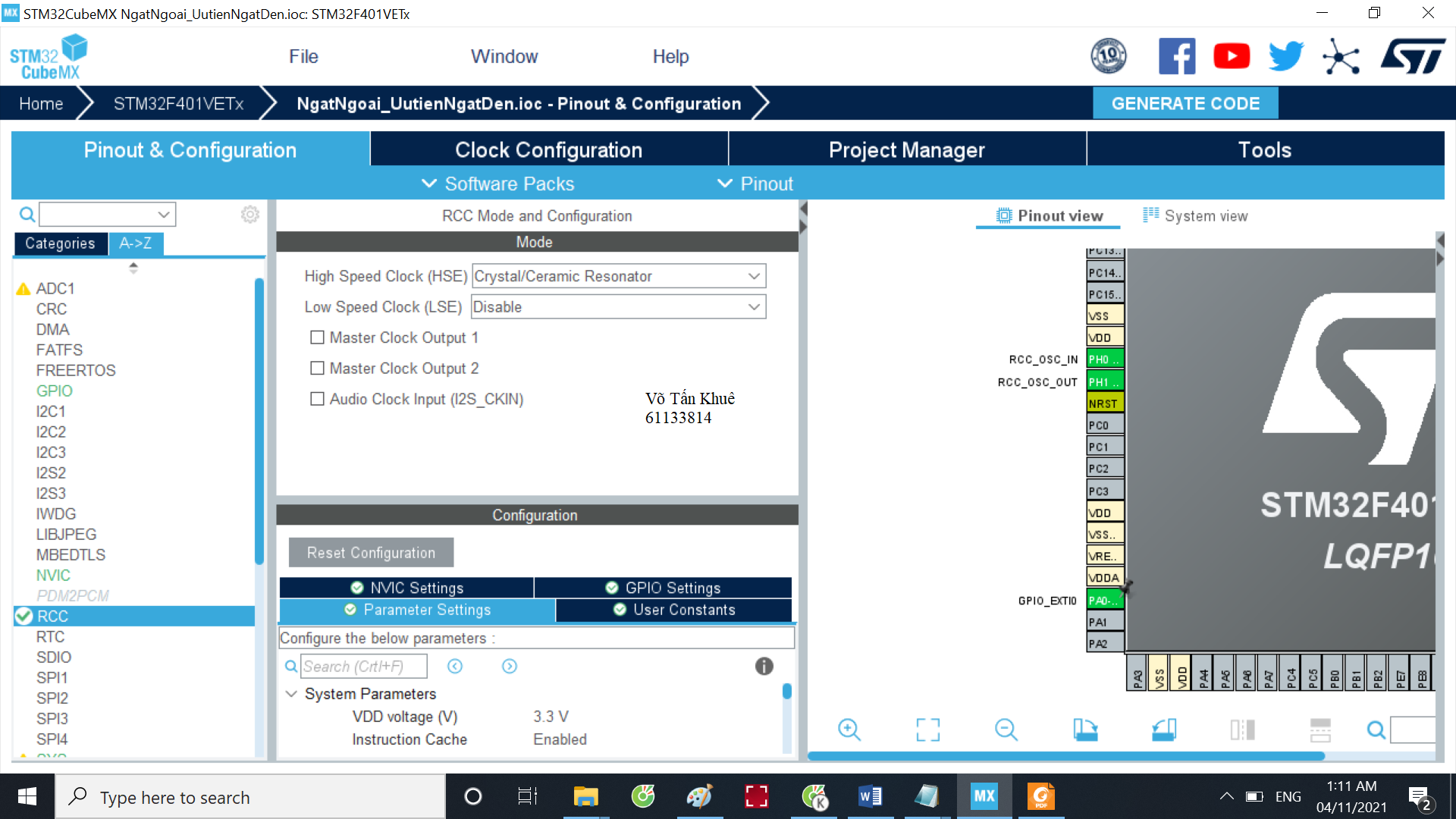
## Đặc điểm của linh kiện

* 1 Biến trở
* 1 STM32F401VE
* **Các cấu hình với CubeMX**
* Tại mục **“Analog”**, và chọn ADC1 và click chọn IN0, thao tác này đã kích hoạt ADC1 channel 0 trên chip. Hoặc chúng ta có thể chọn chân PA0 và chọn chế độ ADC1\_IN0.

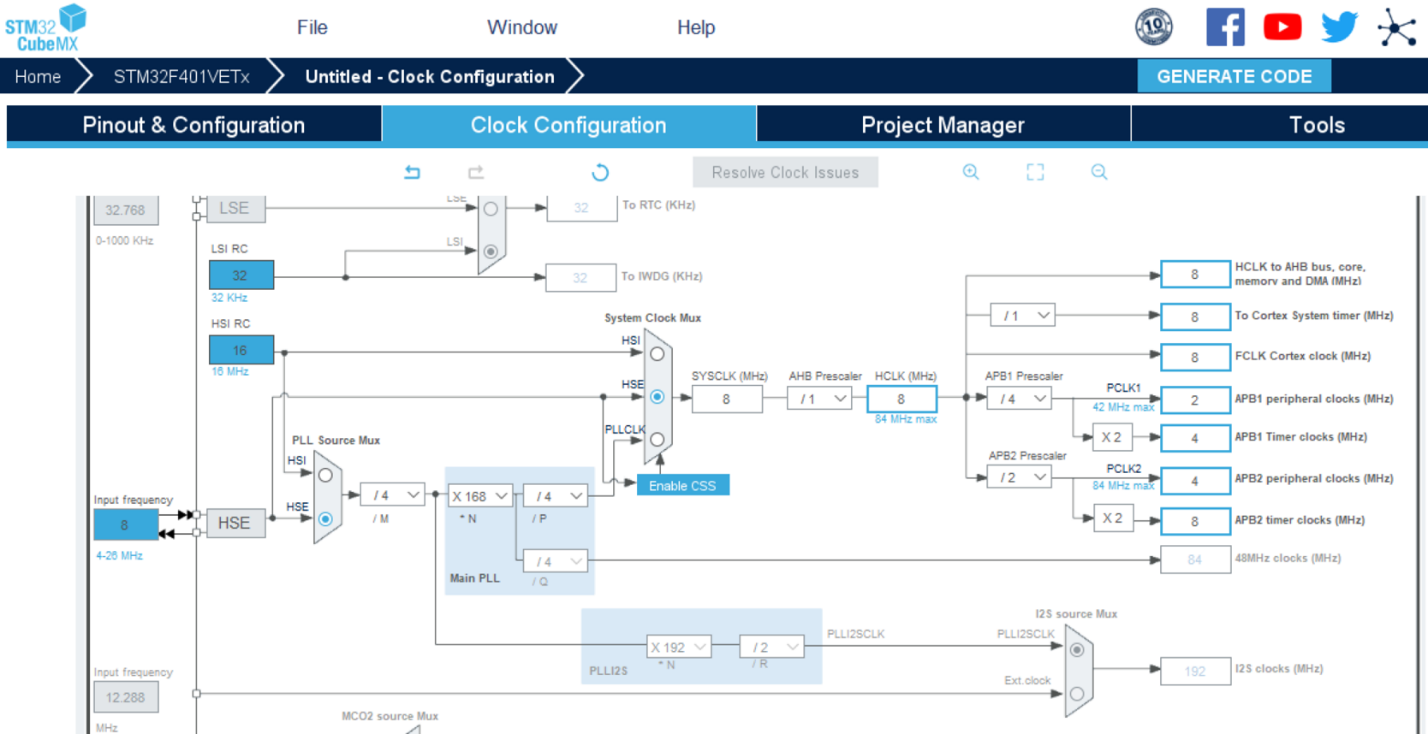


* Chúng ta cấu hình thạch anh và xung Clock cho Chip:

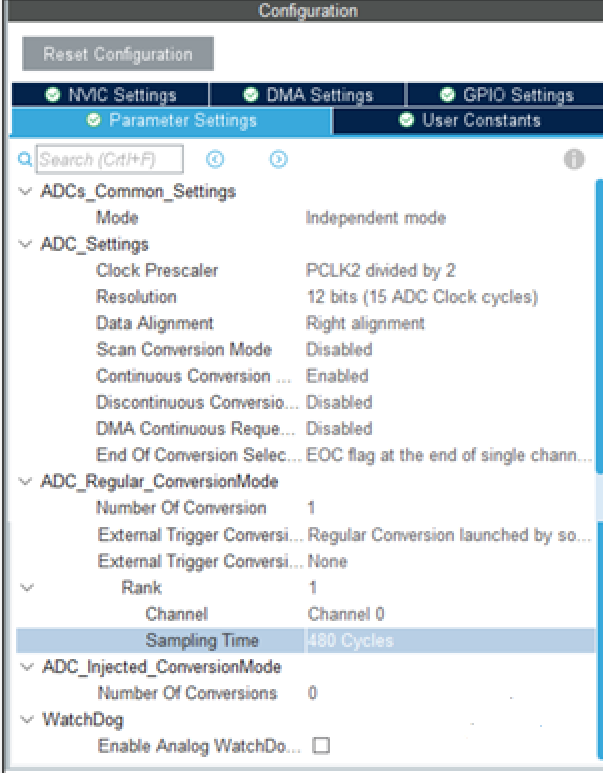
Tại mục **RCC → High Speed Clock (HSE)** và chọn **“Crystal/Ceramic Resonator”** Chức năng này sẽ giúp chip hoạt động với thạch anh ngoại được gắn sẵn trên board mạch.



* Ở Tab **Clock Configuration** đặt **Input frequency** = 8 (thạch anh hàn sẵn trên board là loại 8 Mhz) và điền “8” tại mục “HCLK” (đây là tần số hoạt động tối đa của chip)



* Thiết lập các thông số trong Parameter Settings, trong đó:
* **Resolution:** như đã giới thiệu ở trên, độ phân giải càng cao, quá trình chuyển đổi sẽ càng chính xác, vì vậy chúng ta chọn “12 bits (15 ADC Clock cycles”.
* **Data Alignment:** vì độ phân giải là 12 bit, nên chúng ta cần 1 thanh ghi 16 bit để lưu dữ liệu, như vậy sẽ còn dư 4 bit. Chúng ta chọn “Right alignment” tức là lưu 12 bit dữ liệu dịch về phía bên phải của thanh ghi 16 bit này.
* **Continuous Conversion Mode:** cho phép các quá trình chuyển đổi diễn ra liên tục, chúng ta sẽ “enable” chức năng này.
* **Sampling Time:** thời gian lấy mẫu, nếu thông số này càng lớn, độ chính xác càng lớn nhưng bù lại quá trình chuyển đổi sẽ diễn ra lâu hơn.



## Code chương trình

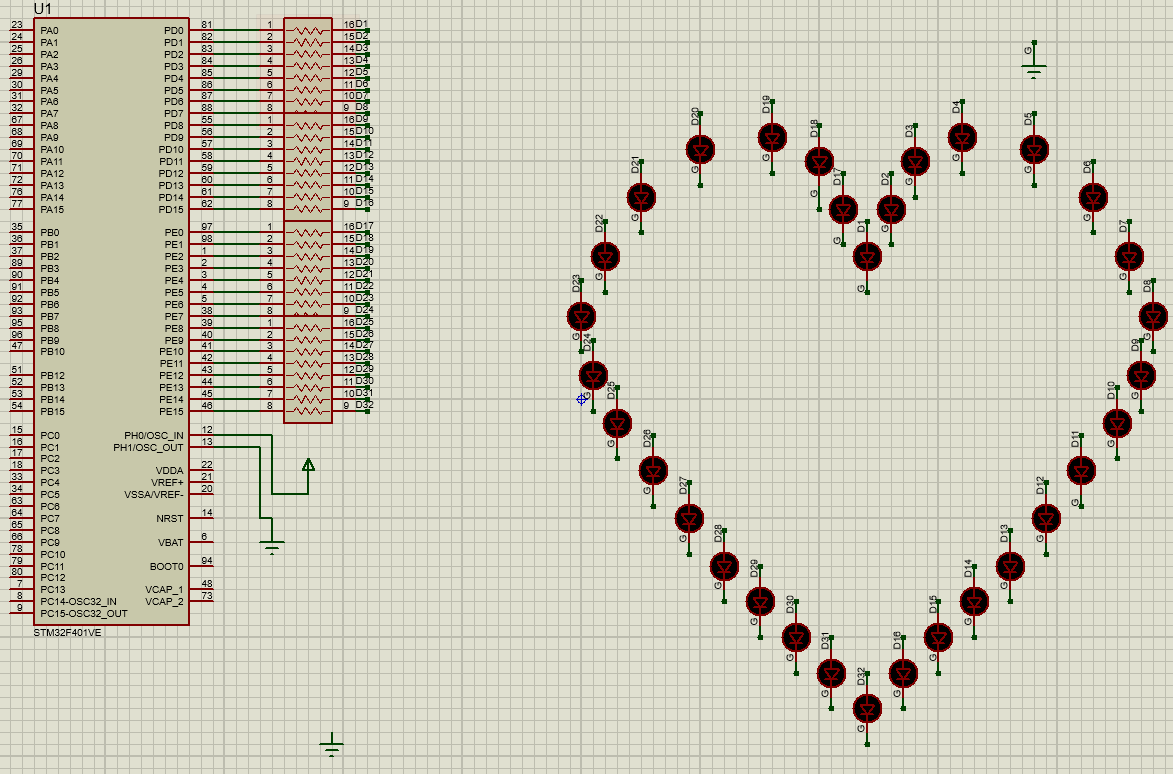
|  |
| --- |
| void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin)  {  if(GPIO\_Pin == GPIO\_PIN\_0)  { HAL\_GPIO\_TogglePin(GPIOD, GPIO\_PIN\_12); }  } |

# Bài 16. Sáng Led hình trái tim bằng STM32

## Mô tả

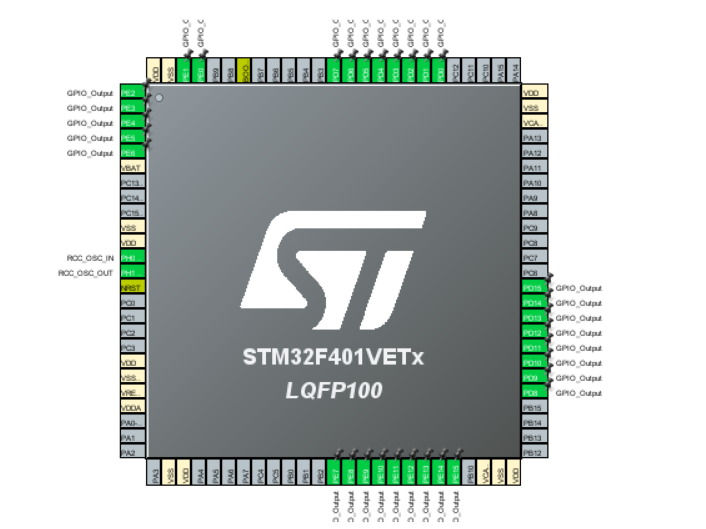
Hệ thống được thiết kế thực hiện việc lập trình điều kiển bật/tắt 20 đèn Led được sắp xếp thành hình trái tim, các đèn Led được kết nối vào STM32 thông qua 3 qua điện trở 16 cổng.

## Sơ đồ thiết kế



## Đặc điểm của linh kiện

* 34 đèn Led
* 4 bộ điện trở 16 cổng 150Ω
* 1 STM32F401VE
* **Các cấu hình với CubeMX**
* Ở Tab **PINOUT** chọn chân màu xanh như trong hình bên dưới thiết lập chức năng **“GPIO\_Output”**



## Code chương trình

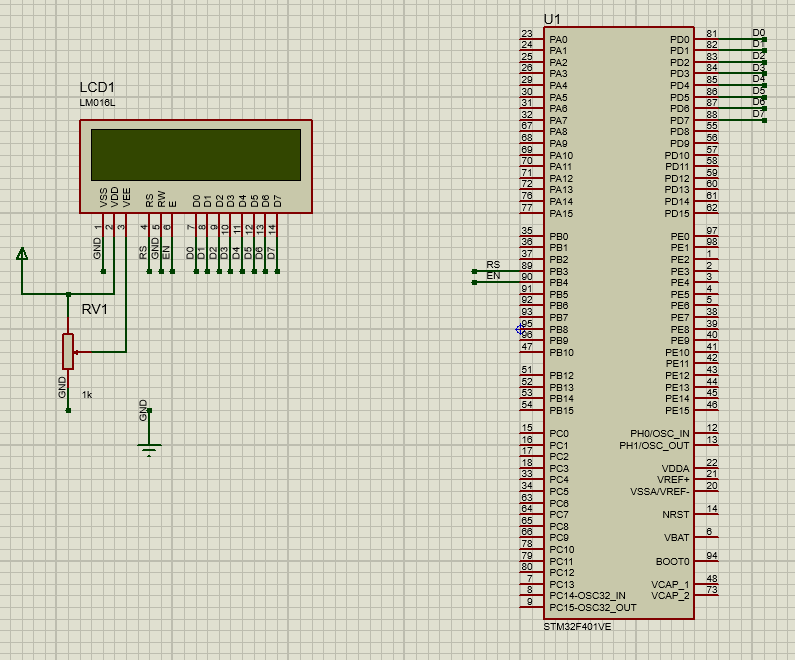
|  |
| --- |
| //Xây dựng các hàm tắt đèn, sáng đèn, sáng lần lượt  void SangHetDen(); //Bật hết tất cả các đèn  void TatHetDen(); //Tắt hết tất cả các đèn  void SangLanLuot(); //Bật sáng từng đèn, giữ nguyên không tắt  void SangLanLuot2(); //Nháy từng đèn  int main(void)  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  while (1)  { //Chương tình chính  SangHetDen();  HAL\_Delay(1000);  TatHetDen();  HAL\_Delay(1000);  SangLanLuot();  HAL\_Delay(1000);  SangLanLuot2();  }  }  void SangHetDen(){  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_2, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_3, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_4, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_5, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_6, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_7, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_8, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_9, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_10, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_11, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_12, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_13, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_14, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_15, GPIO\_PIN\_SET);  //Sáng đèn ở các cổng PE  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_0, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_2, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_3, GPIO\_PIN\_SET);  }  void TatHetDen(){  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_1, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_2, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_3, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_4, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_5, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_6, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_7, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_8, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_9, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_10, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_11, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_12, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_13, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_14, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_15, GPIO\_PIN\_RESET);  // Tắt đèn ở các cổng PE  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_0, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_1, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_2, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_3, GPIO\_PIN\_RESET);  }  void SangLanLuot(){  TatHetDen();  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_2, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_3, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_4, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_5, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_6, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_7, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_8, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_9, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_10, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_11, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_12, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_13, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_14, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_15, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_0, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_2, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_3, GPIO\_PIN\_SET);  HAL\_Delay(100);  }  void SangLanLuot2(){  TatHetDen();  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_1, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_2, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_2, GPIO\_PIN\_RESET);    HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_3, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_3, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_4, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_4, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_5, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_5, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_6, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_6, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_7, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_7, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_8, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_8, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_9, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_9, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_10, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_10, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_11, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_11, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_12, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_12, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_13, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_13, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_14, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_14, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_15, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_15, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_0, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_0, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_1, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_2, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_2, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_3, GPIO\_PIN\_SET);  HAL\_Delay(100);  HAL\_GPIO\_WritePin (GPIOE, GPIO\_PIN\_3, GPIO\_PIN\_RESET);  } |

# Bài 17. Điều khiển hiển thị lên màn hình LCD trên mạch STM32F401VE

## Mô tả

- Mạch lập trình mô tả việc hiển thị chữ lên màn hình LCD, thực hiện trên mạch STM32F401VE

## Sơ đồ thiết kế



*Hình 21. Sơ đồ mạch Proteus*

## Đặc điểm của linh kiện

* Mạch STM32F401VE
* 1 màn hình LCD
* 1 điện trở 100Ω

## Code chương trình

|  |
| --- |
| #include "main.h"  void SystemClock\_Config(void);  static void MX\_GPIO\_Init(void);  // Ham gui data/cmd ra 8 chan cua LCD  void send8BitLCD(char D) {  // Dem D and so hoc voi 2^i de biet bit thu i =0 hay khac ko  // tim gia tri cac bit  int b0,b1,b2, b3,b4, b5, b6, b7;  if ((D & 1) == 0) b0=0; // 1== 2^0  else b0 =1;  if ((D & 2) == 0) b1=0; // 2== 2^1  else b1 =1;  if ((D & 4) == 0) b2=0; // 4== 2^2  else b2 =1;  if ((D & 8) == 0) b3=0; // 8== 2^3  else b3 =1;  if ((D & 16) == 0) b4=0; // 16== 2^4  else b4 =1;  if ((D & 32) == 0) b5=0; // 32== 2^5  else b5 =1;  if ((D & 64) == 0) b6=0; // 64== 2^6  else b6 =1;  if ((D & 128) == 0) b7=0; // 128== 2^7  else b7 =1;  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_0, b0);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_1, b1);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_2, b2);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_3, b3);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_4, b4);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_5, b5);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_6, b6);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_7, b7);  }  // Ham gui lenh  void sendCMD2LCD(char cmd) {  //B1. Done  //B2. Dat chan RS =0, de noi rang cmd là lenh  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3, GPIO\_PIN\_RESET);  //B3. Gui 8 bit CMD vao 8 pin  send8BitLCD(cmd);  //B4. Enable cho cmd-->lcd  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_4, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_4, GPIO\_PIN\_SET);  //  HAL\_Delay(1);  }  // Ham gui ky tu hien thi  void sendChar2LCD(char Char) {  //B1. Done  //B2. Dat chan RS =1,  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3, GPIO\_PIN\_SET);  //B3. Gui 8 bit CMD vao 8 pin  send8BitLCD(Char);  //B4. Enable cho cmd-->lcd  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_4, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_4, GPIO\_PIN\_SET);  //  HAL\_Delay(1);  }  void sendString2LCD(char \*str) {  for (int i=0; str[i] != '\0'; i++) {  sendChar2LCD(str[i]);  }  }  //Chuyen so thanh chuoi so  int itoa(int value,char \*ptr)  {  int count=0,temp;  if(ptr==NULL)  return 0;  if(value==0)  {  \*ptr='0';  return 1;  }  if(value<0)  {  value\*=(-1);  \*ptr++='-';  count++;  }  for(temp=value;temp>0;temp/=10,ptr++);  \*ptr='\0';  for(temp=value;temp>0;temp/=10)  {  \*--ptr=temp%10+'0';  count++;  }  return count;  }  int main(void)  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();    // Xoa noi dung tren LCD  sendCMD2LCD(0x01);  // Bat hien thi man hinh, tat con tro  sendCMD2LCD(0x0C);  // Test thu chuoi Hello  sendString2LCD("TranKhaiHoan");  // Xuong dong 2  sendCMD2LCD(0x38); // CHE DO 2 DONG  sendCMD2LCD(0xC0); // TRO XUONG DONG 2  sendString2LCD("NGUYEN THANH HUNG");  HAL\_Delay(1000);    while (1)  {  }  } |