BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA/VIỆN: CÔNG NGHỆ THÔNG TIN**

**---****---**



**BÁO CÁO LẬP TRÌNH NHÚNG**



**Giảng viên hướng dẫn: Mai Cường Thọ**

**Sinh viên thực hiện: Nguyễn Quốc Châu**

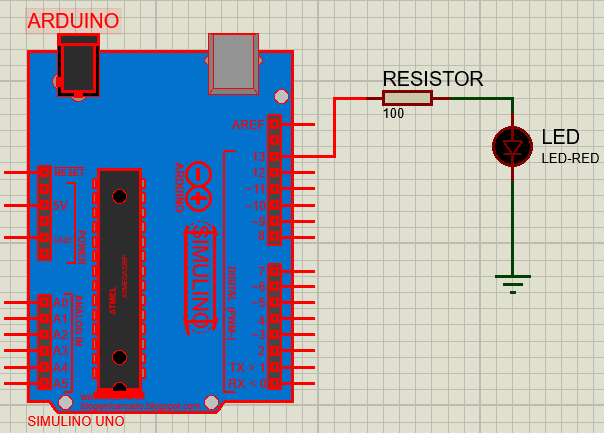
**Mã số sinh viên: 61130073**

# Bài 1: Led nhấp nháy

## Mô tả

Bài này thực hiện việc lập trình điều khiển một đèn Led bật/tắt trong khoảng thời gian 1,5 giây, đèn Led được kết nối vào chân số 13 của board mạch.

## Sơ đồ mạch



Hình 1. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino Uno
* 1 điện trở: 100 **Ω**
* 1 led

## Code chương trình

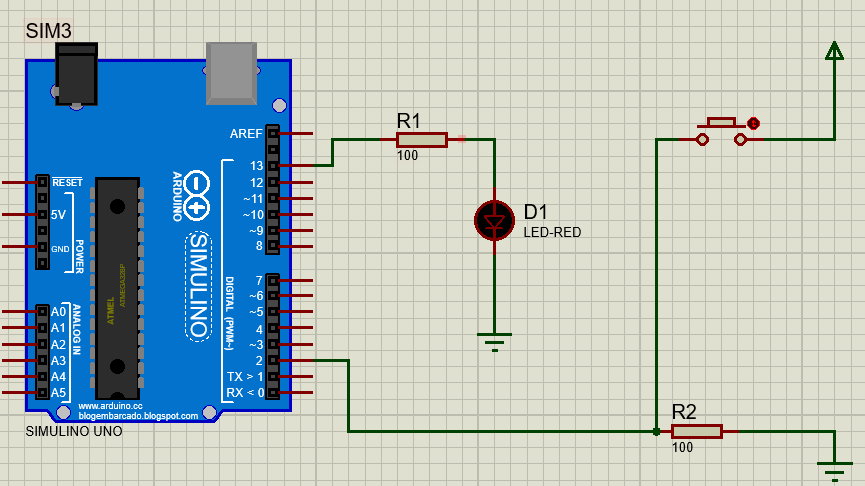
|  |
| --- |
| int LedPin =13; // Thay đổi cổng chân Led  void setup() {  pinMode(LedPin, OUTPUT);  }  void loop() {  digitalWrite(LedPin, HIGH); //Bật Led sáng  delay(1500); // Để Led sáng 1,5 giây  digitalWrite(LedPin, LOW); //Tắt Led  delay(1500); // Để Led tắt 1,5 giây  } |

# Bài 2: Led sáng khi bấm phím

## Mô tả

Bài này thực hiện việc lập trình điều khiển một đèn Led bật/tắt bằng bấm phím trong khoảng thời gian 1 giây, đèn Led được kết nối vào chân số 13 của board mạch.

## Sơ đồ mạch



Hình 2. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino
* 2 điện trở: 100 Ω
* 1 led
* 1 nút bấm

## Code chương trình

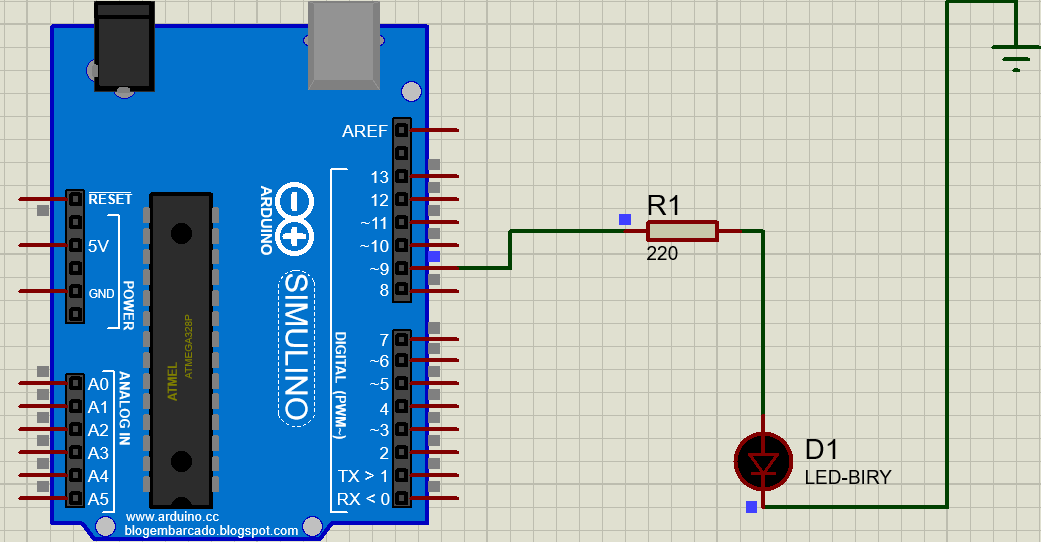
|  |
| --- |
| int LedPin =13; // Thay đổi cổng chân Led  int buttonPin = 2; // Thay đổi cổng chân pin của nút bấm  int x =0; // Thái ban đầu của nút bấm  **void setup()** {  pinMode(LedPin, OUTPUT);  pinMode(buttonPin, INPUT);  }  **void loop()** {  x = digitalRead(buttonPin); // đọc trạng thái của giá trị nút bấm:  // kiểm tra xem nút có được nhấn không. Nếu đúng như vậy, x là CAO:  if (x == HIGH) {  digitalWrite(LedPin, HIGH); // bật LED sáng  } else {  digitalWrite(LedPin, LOW); // tắt LED  }  delay(10); // độ trễ nút bấm  } |

# Bài 3: Led sáng dần

## Mô tả

Bài này thực hiện việc lập trình điều khiển một đèn Led sáng dần trong khoảng thời gian 30 mili giây, đèn Led được kết nối vào chân số 9 của board mạch.

## Sơ đồ mạch



Hình 3. Sơ đồ kết nối của hệ thống.

## Linh kiện

* 1 mạch Arduino Uno
* 1 điện trở: 100 Ω
* 1 led

## Code chương trình

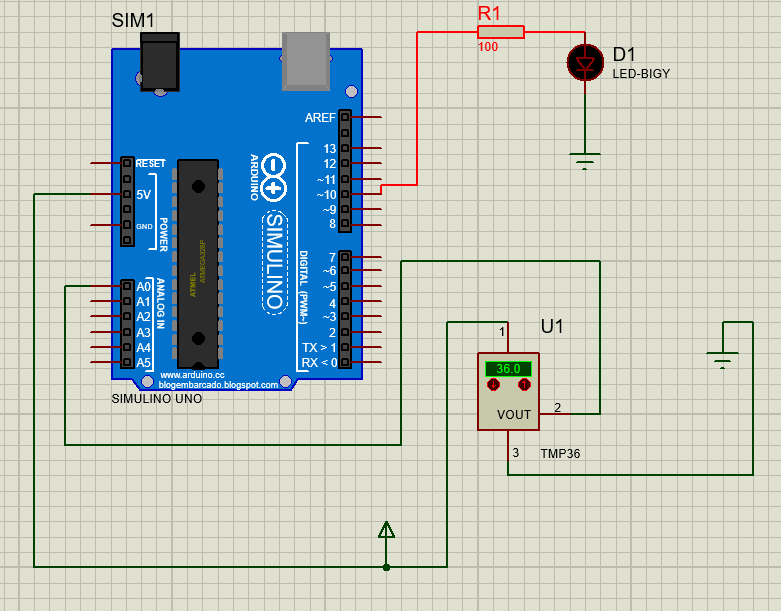
|  |
| --- |
| int brightness =0;  int LedPin =9; // Thay đổi cổng chân Led  **void setup()** {  pinMode(LedPin, OUTPUT);  }  **void loop()** {  // Vòng lặp đèn sáng  for ( brightness = 0; brightness <=255; brightness +=5)  {  analogWrite(LedPin,brightness);  delay(30); // độ trễ đèn sáng  }  for ( brightness = 255; brightness <=0; brightness -=5)  {  analogWrite(LedPin,brightness);  delay(30); // độ trễ đèn sáng  }  } |

# Bài 4: Led sáng với cảm biến nhiệt độ

## Mô tả

Bài này thực hiện việc lập trình điều khiển một đèn Led sáng khi nhiệt độ lên quá 36 độ, đèn Led được kết nối vào chân số 10 và cảm biến nhiệt độ TMP36 được kết nối chân 2-A0 của board mạch và chân 2 cảm biến nhiệt độ nối với nguồn 5 Volt.

## Sơ đồ mạch



Hình 4. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino Uno
* 1 điện trở: 100 Ω
* 1 led
* 1 cảm biến nhiệt độ TMP36

## Code chương trình

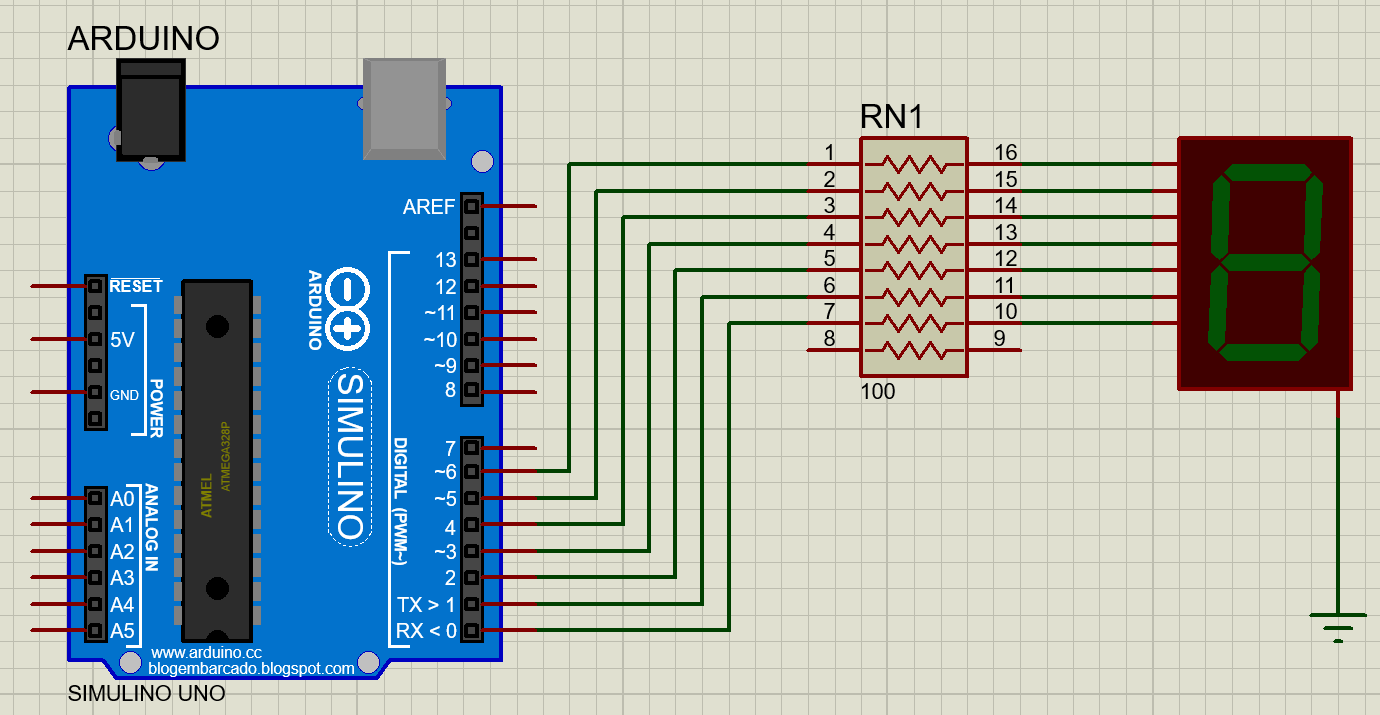
|  |
| --- |
| int LedPin =10; // Thay đổi cổng chân Led  **void setup()**  {  pinMode(LedPin, OUTPUT);  Serial.begin(96000); // khởi tạo kết nối nối tiếp ở 9600 bit mỗi giây  }  **void loop()**  {  int x = analogRead(A0);  int t = map ( x , 0 , 410 , -50 , 150);  if ( t >=36) //Nhiệt độ lớn hơn 36 thì đèn sáng và ngược lại  digitalWrite(LedPin,HIGH);  else  digitalWrite(LedPin, LOW);  delay(100); // độ trễ đèn sáng  } |

# Bài 5: Led 7 đoạn

## Mô tả

Bài này thực hiện việc lập trình điều khiển mạch Led 7 đoạn sáng hình số 3, đèn Led được kết nối vào các chân A-6,B-5,C-4,D-3,E-2,F-1,G-0 của board mạch.

## Sơ đồ mạch



Hình 5. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino Uno
* 1 bảng mạch điện trở: 100 Ω
* 1 bảng mạch led 7 đoạn

## Code chương trình

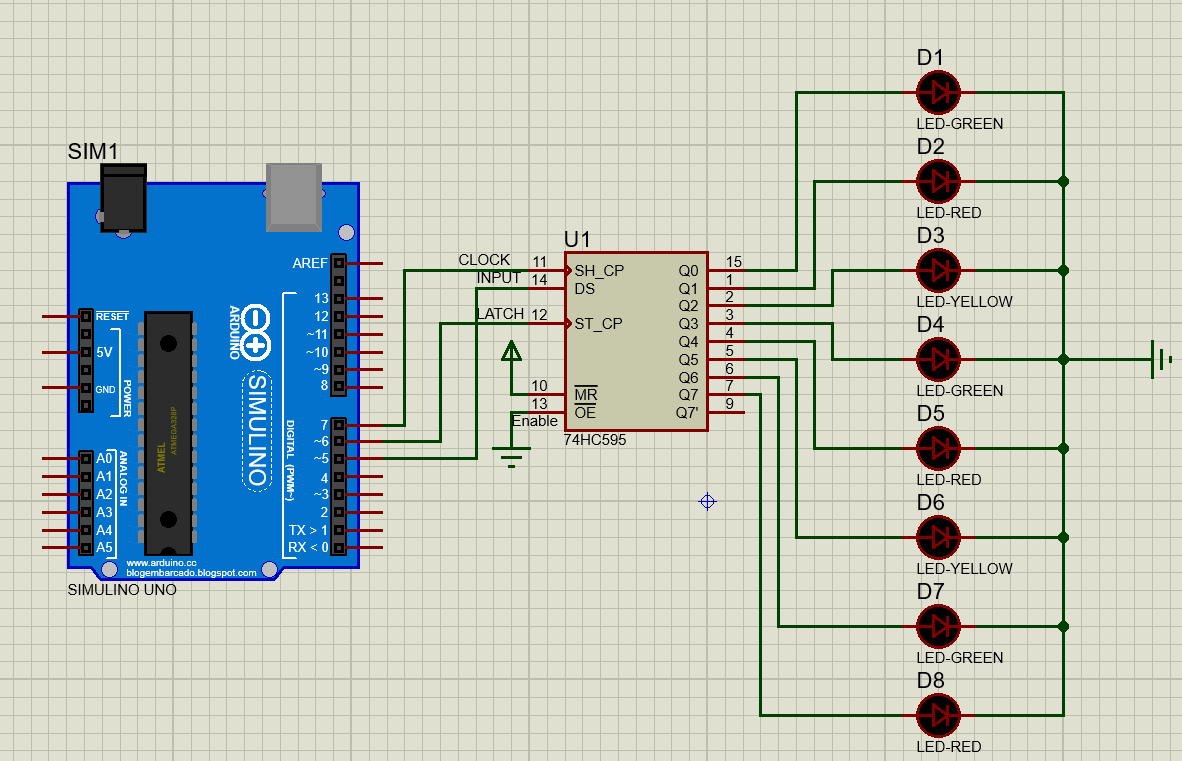
|  |
| --- |
| int a=6 , b=5 , c=4, d=3, e=2, f=1, g=0; // Các chân đèn gắn với mạch  **void setup()**{  pinMode(a, OUTPUT); pinMode(b, OUTPUT);  pinMode(c, OUTPUT); pinMode(d, OUTPUT);  pinMode(e, OUTPUT); pinMode(g, OUTPUT);  pinMode(f, OUTPUT); }  **void BA()**{  digitalWrite (a,HIGH); digitalWrite (b,HIGH);  digitalWrite (c,HIGH); digitalWrite (d,HIGH);  digitalWrite (g,HIGH); digitalWrite (e,LOW);  digitalWrite (f,LOW); }  **void loop()**{  BA();  } |

# Bài 6: Led sáng tuần tự

## Mô tả

Bài này thực hiện việc lập trình điều khiển mạch Led sáng tuần tự, đèn Led được kết nối IC 74HC595 và được kết nối các chân 5,6,7 của board mạch.

## Sơ đồ mạch



Hình 6. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino Uno
* 8 Led
* 1 mạch IC 74HC595

## Code chương trình

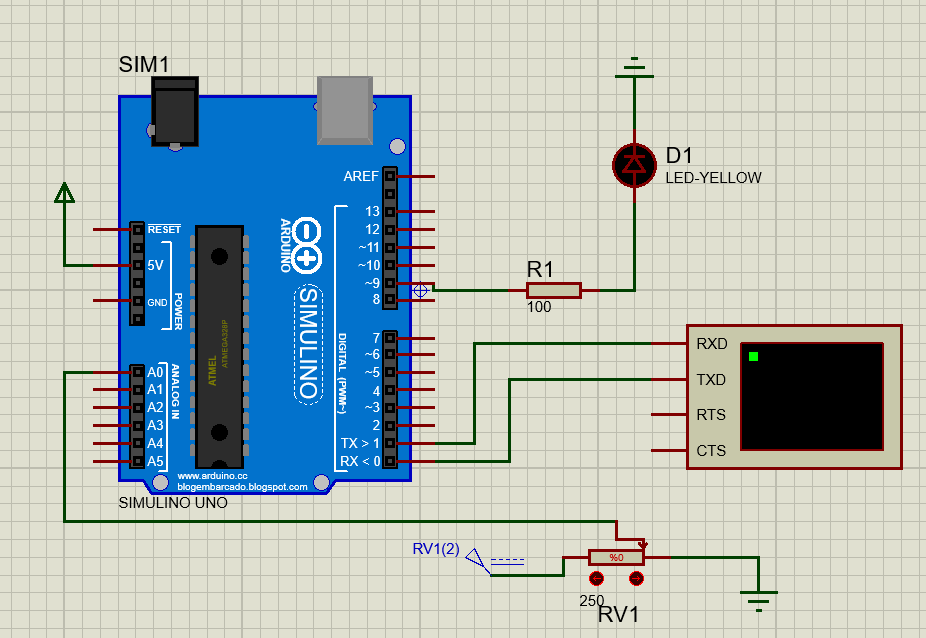
|  |
| --- |
| // Chân đèn được kết nối thông qua mạch IC 74HC595 tới board mạch  #define \_clock 7  #define \_latch 6  #define \_data 5  **void setup()** {  pinMode( \_clock,OUTPUT);  pinMode( \_latch,OUTPUT);  pinMode( \_data,OUTPUT);  }  **void loop()** {  for ( int i=0; i< 256; i++)  {  digitalWrite(\_latch, LOW);  shiftOut(\_data, \_clock, LSBFIRST,i); // Đọc 8 bit xuất theo thứ tự  digitalWrite(\_latch, HIGH);  delay(100); //Độ trễ đèn  }  } |

# Bài 7: Led sáng bằng chiết áp

## Mô tả

Bài này thực hiện việc lập trình điều khiển mạch Led thông qua chiết áp được nối vào A0 và chân đèn nối vào chân 9 của board mạch.

## Sơ đồ mạch



Hình 7. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino Uno
* 1 Led
* 1 chiết áp

## Code chương trình

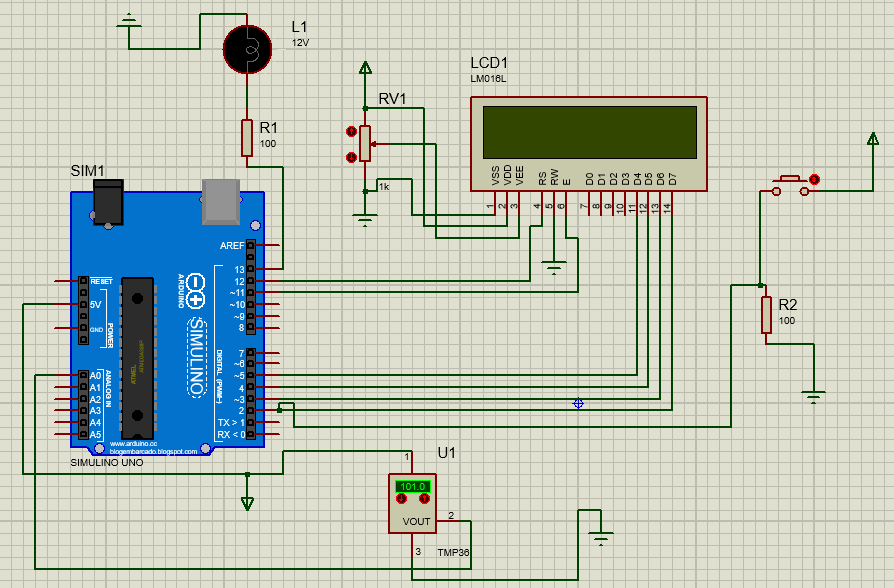
|  |
| --- |
| int pinLed = 9; // Thay đổi cổng chân Led  int poten = A0; // Chân chiết áp  **void setup()** {  pinMode(poten,INPUT);  pinMode(pinLed, OUTPUT);  Serial.begin(9600);  }  int readA0 = 0;  **void loop()** {  readA0 = analogRead(poten);  Serial.println(readA0);  digitalWrite(pinLed,HIGH);  delay(readA0);  digitalWrite(pinLed,LOW);  delay(readA0);  } |

# Bài 8: Hiện nhiệt độ bằng LCD và sáng đèn

## Mô tả

Bài này thực hiện việc lập trình điều khiển màn hình LCD hiện nhiệt độ của TMP36 và một nút bấm khi bấm với điều kiện nhiệt độ trên 100 sẽ sáng đèn.

## Sơ đồ mạch



Hình 8. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 mạch Arduino Uno
* 1 Lamp
* 2 điện trở: 100 Ω
* 1 POT
* 1 cảm biến nhiệt độ TMP36
* 1 LCD

## Code chương trình

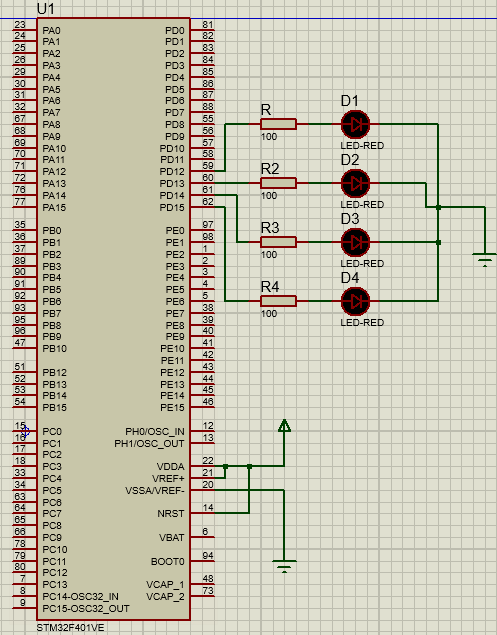
|  |
| --- |
| #include <LiquidCrystal.h>  #include <SoftwareSerial.h>  LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  int sensorTMPPin;// chân analog kết nối tới cảm biến TMP36  int y = 0;  int pinled = 13;  int button = 2;  **void setup()** {  pinMode(pinled, OUTPUT);  pinMode(button, INPUT);  Serial.begin(96000);  lcd.begin(16, 2); // thiết lập số cột và số hàng của màn hình LCD  lcd.print("Nhiet do"); // In thông báo ra màn hình LCD.  Serial.println("Nhiet do");  }  **void loop()** {  int x = analogRead(A0);  int sensorTMPPin = map ( ((x -20)\*3.04),0,1023, -40, 125);  lcd.setCursor(0, 0);  lcd.print("Nhiet do");  lcd.setCursor(0, 1);  lcd.print(sensorTMPPin);  lcd.print("C"); // ký tự sau nhiệt độ  Serial.println(sensorTMPPin);  delay( 1000);  lcd.clear();  batden();  }  **void batden()**{  int z = analogRead(A0);  int t = map ( z , 0 , 410 , -50 , 150);  y = digitalRead(button);  // kiểm tra xem nút có được nhấn không. Nếu đúng như vậy, x là CAO:  if (y == HIGH && t>=100) {  digitalWrite(pinled, HIGH); // turn LED on:  } else {  digitalWrite(pinled, LOW); // turn LED off:  }  delay(10); // độ trễ button  } |

# Bài 9: Nháy Led sáng bằng vi điều khiển STM32F401VE

## Mô tả

Bài này thực việc lập trình Proteus điều khiển nháy 4 Led trong 0.1s bằng vi điều khiển STM32F401VE . Với các chân Led được kết nối với các chân PD12,PD13,PD14,PD15 ở mức thấp và tần số đầu vào và ra là 8Hz của vi điều khiển.

## Sơ đồ mạch



Hình 9. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 vi điều khiển STM32F401VE
* 4 Led
* 4 điện trở: 100 Ω

## Code chương trình

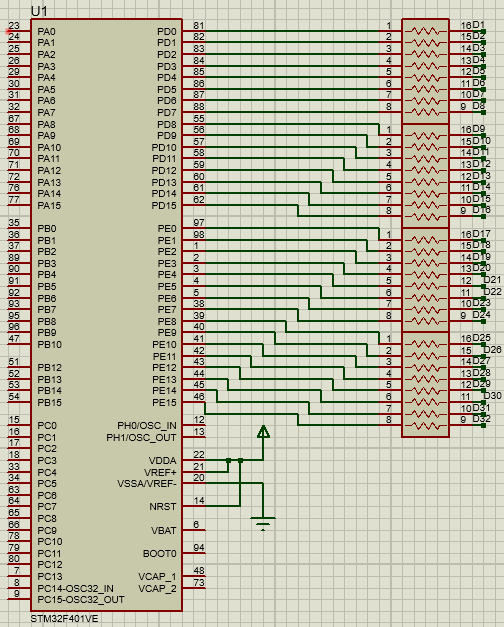
|  |
| --- |
| **#include "main.h"**  **#include "stm32f4xx\_hal.h"**  **void SystemClock\_Config(void);**  **static void MX\_GPIO\_Init(void);**  **int main(void)**  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  while (1)  {  // Thiết lập các chân Led  HAL\_GPIO\_TogglePin(GPIOD,GPIO\_PIN\_12);  HAL\_GPIO\_TogglePin(GPIOD,GPIO\_PIN\_13);  HAL\_GPIO\_TogglePin(GPIOD,GPIO\_PIN\_14);  HAL\_GPIO\_TogglePin(GPIOD,GPIO\_PIN\_15);  HAL\_Delay(100);  }  }  **void SystemClock\_Config(void)**  {  RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};  RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};  \_\_HAL\_RCC\_PWR\_CLK\_ENABLE();  \_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG(PWR\_REGULATOR\_VOLTAGE\_SCALE2);  RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSI;  RCC\_OscInitStruct.HSIState = RCC\_HSI\_ON;  RCC\_OscInitStruct.HSICalibrationValue = RCC\_HSICALIBRATION\_DEFAULT;  RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;  if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)  {  Error\_Handler();  }  RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK  |RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;  RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_HSI;  RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;  RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV2;  RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV2;  if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0) != HAL\_OK)  {  Error\_Handler();  }  }  **static void MX\_GPIO\_Init(void)**  {  GPIO\_InitTypeDef GPIO\_InitStruct = {0};  \_\_HAL\_RCC\_GPIOC\_CLK\_ENABLE();  \_\_HAL\_RCC\_GPIOH\_CLK\_ENABLE();  \_\_HAL\_RCC\_GPIOD\_CLK\_ENABLE();  \_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();  HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15, GPIO\_PIN\_SET);  GPIO\_InitStruct.Pin = GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15;  GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;  HAL\_GPIO\_Init(GPIOD, &GPIO\_InitStruct);  }  **void Error\_Handler(void)**  {  \_\_disable\_irq();  while (1)  {  }  }  #ifdef USE\_FULL\_ASSERT |

# Bài 10: Nháy led trái tim xuôi và ngược

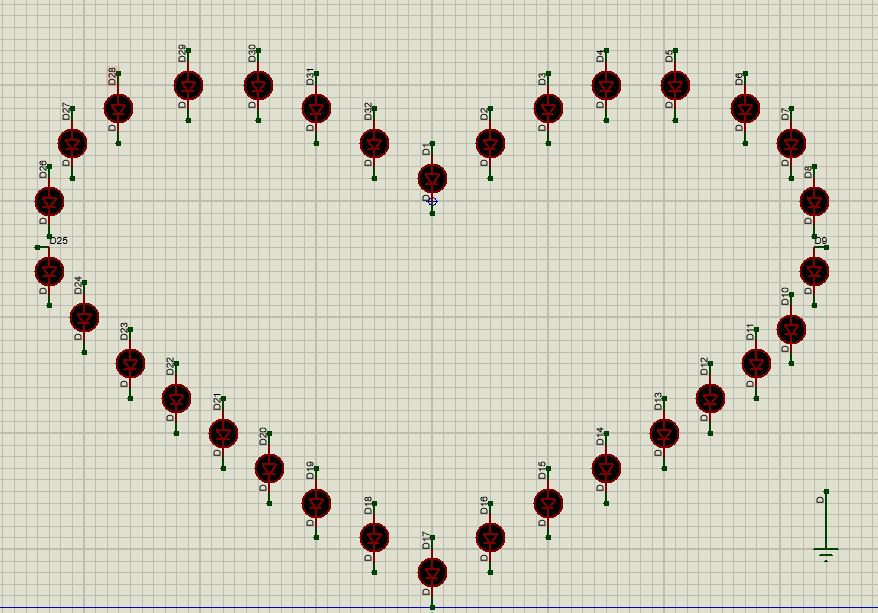
## Mô tả

Bài này thực việc lập trình Proteus điều khiển nháy Led hình trái tim xuôi và ngược trong 0.1s bằng vi điều khiển STM32F401VE . Với các chân Led được kết nối với tất cả các chân D,E ở mức thấp và tần số đầu vào và ra là 8Hz của vi điều khiển.

## Sơ đồ mạch



Hình 10a. Sơ đồ kết nối của hệ thống



Hình 10b. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 vi điều khiển STM32F401VE
* 32 Led
* 4 mạch điện trở DIPIS: 150 Ω

## Code chương trình

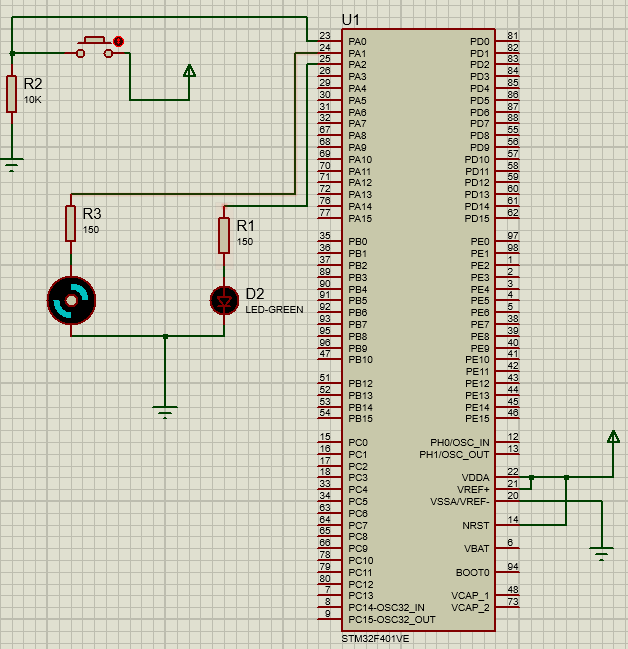
|  |
| --- |
| **#include "main.h"**  // Tạo biến luu tru chân led  Int hieuungled[]={0x0001,0x0002,0x0004,0x0008,0x0010,0x0020,0x0040,0x0080,  0x0100,0x0200,0x0400,0x0800,0x1000,0x2000,0x4000,0x8000};  **void SystemClock\_Config(void);**  **static void MX\_GPIO\_Init(void);**  **void SangHetLed();**  **void TatHetLed();**  **void SangXuoi();**  **void SangNguoc();**  **void NhapNhay();**  **int main(void)**  {  HAL\_Init();  SystemClock\_Config();  MX\_GPIO\_Init();  while (1)  {  TatHetLed();  HAL\_Delay(10);  SangHetLed();  HAL\_Delay(1000);  TatHetLed();  HAL\_Delay(100);  SangXuoi();  HAL\_Delay(1000);  TatHetLed();  HAL\_Delay(1000);  SangNguoc();  HAL\_Delay(1000);  NhapNhay();  HAL\_Delay(1000);  }  }  **void SangXuoi()**  {  for ( int i=0;i<=15;i++)  {  // Sáng Led chân D tuần tự và xuôi  HAL\_GPIO\_TogglePin ( GPIOD, hieuungled[i]);  HAL\_Delay(100);  }  for ( int i=0;i<=15;i++)  {  // Sáng Led chân E tuần tự và xuôi  HAL\_GPIO\_TogglePin ( GPIOE, hieuungled[i]);  HAL\_Delay(100);  }  }  **void SangNguoc()**  {  for ( int i=15;i>=0;i--)  {  // Sáng Led chân E tuần tự và ngược  HAL\_GPIO\_TogglePin ( GPIOE, hieuungled[i]);  HAL\_Delay(100);  }  for ( int i=15;i>=0;i--)  {  // Sáng Led chân D tuần tự và ngược  HAL\_GPIO\_TogglePin ( GPIOD, hieuungled[i]);  HAL\_Delay(100);  }  }  **void NhapNhay()**  {  // Nháy led trong vòng 0.5s  for ( int i=0;i<=3;i++)  {  SangHetLed();  HAL\_Delay(500);  TatHetLed();  }  }  **void SangHetLed()**  {  // SÁNG ÐÈN CÔNG D  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);  // SANG ÐÈN CÔNG E  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);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_4, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_5, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_6, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_7, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_8, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_9, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_10, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_11, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_12, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_13, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_14, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_15, GPIO\_PIN\_SET);  }  **void TatHetLed()**  {  // TAT ÐÈN CÔNG D  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);  // TAT ÐÈN CÔNG E  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);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_4, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_5, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_6, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_7, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_8, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_9, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_10, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_11, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_12, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_13, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_14, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin ( GPIOE, GPIO\_PIN\_15, GPIO\_PIN\_RESET);  }  **void SystemClock\_Config(void)**  {  RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};  RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};  \_\_HAL\_RCC\_PWR\_CLK\_ENABLE();  \_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG(PWR\_REGULATOR\_VOLTAGE\_SCALE2);  RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSE;  RCC\_OscInitStruct.HSEState = RCC\_HSE\_ON;  RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;  if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)  {  Error\_Handler();  }  RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK  |RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;  RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_HSE;  RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV1;  RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV1;  RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;  if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0) != HAL\_OK)  {  Error\_Handler();  }  }  **static void MX\_GPIO\_Init(void)**  {  GPIO\_InitTypeDef GPIO\_InitStruct = {0};  \_\_HAL\_RCC\_GPIOE\_CLK\_ENABLE();  \_\_HAL\_RCC\_GPIOD\_CLK\_ENABLE();  HAL\_GPIO\_WritePin(GPIOE, GPIO\_PIN\_2|GPIO\_PIN\_3|GPIO\_PIN\_4|GPIO\_PIN\_5  |GPIO\_PIN\_6|GPIO\_PIN\_7|GPIO\_PIN\_8|GPIO\_PIN\_9  |GPIO\_PIN\_10|GPIO\_PIN\_11|GPIO\_PIN\_12|GPIO\_PIN\_13  |GPIO\_PIN\_14|GPIO\_PIN\_15|GPIO\_PIN\_0|GPIO\_PIN\_1, GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_8|GPIO\_PIN\_9|GPIO\_PIN\_10|GPIO\_PIN\_11  |GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15  |GPIO\_PIN\_0|GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3  |GPIO\_PIN\_4|GPIO\_PIN\_5|GPIO\_PIN\_6|GPIO\_PIN\_7, GPIO\_PIN\_SET);  GPIO\_InitStruct.Pin = GPIO\_PIN\_2|GPIO\_PIN\_3|GPIO\_PIN\_4|GPIO\_PIN\_5  |GPIO\_PIN\_6|GPIO\_PIN\_7|GPIO\_PIN\_8|GPIO\_PIN\_9  |GPIO\_PIN\_10|GPIO\_PIN\_11|GPIO\_PIN\_12|GPIO\_PIN\_13  |GPIO\_PIN\_14|GPIO\_PIN\_15|GPIO\_PIN\_0|GPIO\_PIN\_1;  GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;  HAL\_GPIO\_Init(GPIOE, &GPIO\_InitStruct);  GPIO\_InitStruct.Pin = GPIO\_PIN\_8|GPIO\_PIN\_9|GPIO\_PIN\_10|GPIO\_PIN\_11  |GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15  |GPIO\_PIN\_0|GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3  |GPIO\_PIN\_4|GPIO\_PIN\_5|GPIO\_PIN\_6|GPIO\_PIN\_7;  GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;  HAL\_GPIO\_Init(GPIOD, &GPIO\_InitStruct);  }  **void Error\_Handler(void)**  {  \_\_disable\_irq();  while (1)  {  }  }  #ifdef USE\_FULL\_ASSERT |

# Bài 11: Sáng Led và Chạy Motor bằng bấm phím

## Mô tả

Bài này thực việc lập trình Proteus điều khiển phím bấm sáng Led và chạy Motor bằng vi điều khiển STM32F401VE . Với chân Led, Motor và phím bấm được kết nối với chân A1,A2,A0 ở mức thấp và tần số đầu vào và ra là 8Hz của vi điều khiển. Khi bấm phím 1 lần sẽ chạy Motor và bấm phím lần 2 Led sẽ sáng.

## Sơ đồ mạch



Hình 11. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 vi điều khiển STM32F401VE
* 1 Led
* 3 điện trở: 150 Ω
* 1 Motor
* 1 phím bấm

## Code chương trình

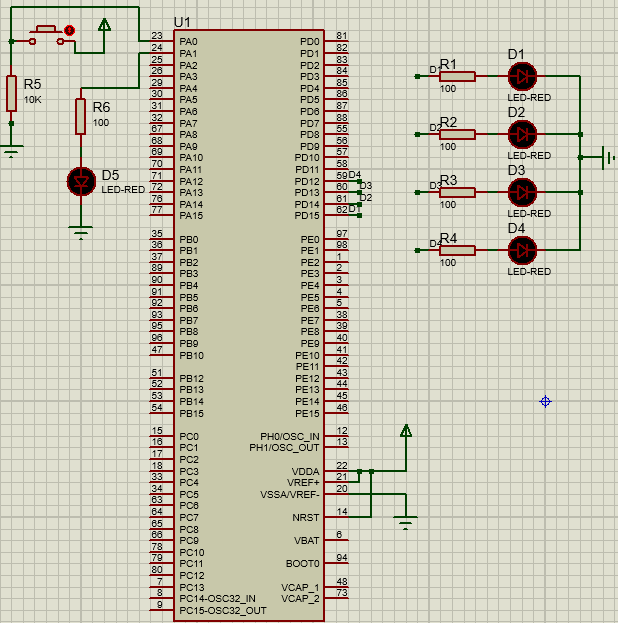
|  |
| --- |
| **#include "main.h"**  **void SystemClock\_Config(void);**  **static void MX\_GPIO\_Init(void);**  **unsigned** trangthai=0;// trạng thái nút bấm  **void nutnhan()**  {  if((HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0)==0))  {  HAL\_Delay(20);  if((HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0)==0))  {  if(trangthai<3)  {  trangthai++;  }  else  {  trangthai=1;  }  // vòng cho trang thai nut bam  while(HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0)==0){}  } }  }  **int main(void)**  {  **HAL\_Init();**  **SystemClock\_Config();**  **MX\_GPIO\_Init();**  while (1)  {  nutnhan();  // chạy motor  if(trangthai==1)  {  HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_1,GPIO\_PIN\_SET);  }  //sáng led  if(trangthai==2)  {  HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_2,GPIO\_PIN\_SET);  }  // tắt motor và led  if(trangthai==3)  {  HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_1,GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_2,GPIO\_PIN\_RESET);  }  }  }  **void SystemClock\_Config(void)**  {  RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};  RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};  \_\_HAL\_RCC\_PWR\_CLK\_ENABLE();  \_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG(PWR\_REGULATOR\_VOLTAGE\_SCALE2);  RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSI;  RCC\_OscInitStruct.HSIState = RCC\_HSI\_ON;  RCC\_OscInitStruct.HSICalibrationValue = RCC\_HSICALIBRATION\_DEFAULT;  RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;  if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)  {  Error\_Handler();  }  RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK  |RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;  RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_HSI;  RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV2;  RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV1;  RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;  if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0) != HAL\_OK)  {  Error\_Handler();  }  }  **static void MX\_GPIO\_Init(void)**  {  GPIO\_InitTypeDef GPIO\_InitStruct = {0};  \_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();  HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1|GPIO\_PIN\_2, GPIO\_PIN\_RESET);  GPIO\_InitStruct.Pin = GPIO\_PIN\_0;  GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);  GPIO\_InitStruct.Pin = GPIO\_PIN\_1|GPIO\_PIN\_2;  GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;  HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);  }  **void Error\_Handler(void)**  {  \_\_disable\_irq();  while (1)  {  }  }  #ifdef USE\_FULL\_ASSERT |

# Bài 12: Sáng 5 Led và 1 Led bằng bấm phím

## Mô tả

Bài này thực việc lập trình Proteus điều khiển sáng 1 Led với phím bấm và 4 Led còn lại sáng mặc định bằng vi điều khiển STM32F401VE . Với chân Led và phím bấm được kết nối với chân D15,D12,D13,D14,A1,A0 ở mức thấp và tần số đầu vào và ra là 8Hz của vi điều khiển. Khi bấm phím Led sẽ sáng.

## Sơ đồ mạch



Hình 12. Sơ đồ kết nối của hệ thống

## Linh kiện

* 1 vi điều khiển STM32F401VE
* 5 Led
* 6 điện trở: 150 Ω
* 1 phím bấm

## Code chương trình

|  |
| --- |
| **#include "main.h"**  **void SystemClock\_Config(void);**  **static void MX\_GPIO\_Init(void);**  **unsigned** trangthai=0; // trang thai nut bâm  **void nutnhan()**  {  if((HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0)==0))  {  HAL\_Delay(20);  if((HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0)==0))  {  if(trangthai<2)  {  trangthai++;  }  else  {  trangthai=1;  }  // vòng cho trang thai nut bam  while(HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0)==0){}  }  }  }  **int main(void)**  {  **HAL\_Init();**  **SystemClock\_Config();**  **MX\_GPIO\_Init();**  while (1)  {  // Sáng 4 Led các chân 15,14,13,12  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_15,GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_14,GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_13,GPIO\_PIN\_SET);  HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_12,GPIO\_PIN\_SET);  nutnhan();  if(trangthai==1)  {  HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_1,GPIO\_PIN\_SET);  }  else  {  HAL\_GPIO\_WritePin(GPIOA,GPIO\_PIN\_1,GPIO\_PIN\_RESET);  }  }  }  **void SystemClock\_Config(void)**  {  RCC\_OscInitTypeDef RCC\_OscInitStruct = {0};  RCC\_ClkInitTypeDef RCC\_ClkInitStruct = {0};  \_\_HAL\_RCC\_PWR\_CLK\_ENABLE();  \_\_HAL\_PWR\_VOLTAGESCALING\_CONFIG(PWR\_REGULATOR\_VOLTAGE\_SCALE2);  RCC\_OscInitStruct.OscillatorType = RCC\_OSCILLATORTYPE\_HSI;  RCC\_OscInitStruct.HSIState = RCC\_HSI\_ON;  RCC\_OscInitStruct.HSICalibrationValue = RCC\_HSICALIBRATION\_DEFAULT;  RCC\_OscInitStruct.PLL.PLLState = RCC\_PLL\_NONE;  if (HAL\_RCC\_OscConfig(&RCC\_OscInitStruct) != HAL\_OK)  {  Error\_Handler();  }  RCC\_ClkInitStruct.ClockType = RCC\_CLOCKTYPE\_HCLK|RCC\_CLOCKTYPE\_SYSCLK  |RCC\_CLOCKTYPE\_PCLK1|RCC\_CLOCKTYPE\_PCLK2;  RCC\_ClkInitStruct.SYSCLKSource = RCC\_SYSCLKSOURCE\_HSI;  RCC\_ClkInitStruct.AHBCLKDivider = RCC\_SYSCLK\_DIV2;  RCC\_ClkInitStruct.APB1CLKDivider = RCC\_HCLK\_DIV1;  RCC\_ClkInitStruct.APB2CLKDivider = RCC\_HCLK\_DIV1;  if (HAL\_RCC\_ClockConfig(&RCC\_ClkInitStruct, FLASH\_LATENCY\_0) != HAL\_OK)  {  Error\_Handler();  }  }  **static void MX\_GPIO\_Init(void)**  {  GPIO\_InitTypeDef GPIO\_InitStruct = {0};  \_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();  \_\_HAL\_RCC\_GPIOD\_CLK\_ENABLE();  HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_RESET);  HAL\_GPIO\_WritePin(GPIOD, GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15, GPIO\_PIN\_RESET);  GPIO\_InitStruct.Pin = GPIO\_PIN\_0;  GPIO\_InitStruct.Mode = GPIO\_MODE\_INPUT;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);  GPIO\_InitStruct.Pin = GPIO\_PIN\_1;  GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;  HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);  GPIO\_InitStruct.Pin = GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15;  GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;  GPIO\_InitStruct.Pull = GPIO\_NOPULL;  GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_LOW;  HAL\_GPIO\_Init(GPIOD, &GPIO\_InitStruct);  }  **void Error\_Handler(void)**  {  \_\_disable\_irq();  while (1)  {  }  }  #ifdef USE\_FULL\_ASSERT |