How to Use Arduino UNO R3

Dec. 5, 2021

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Arduino Uno 是基于 ATmega328P 的微控制器板。 它有 14 个数字输入/输出引脚（其中 6 个可用作 PWM 输出）、6 个模拟输入、一个 16 MHz 陶瓷谐振器 (CSTCE16M0V53-R0)、一个 USB 连接、一个电源插孔、一个 ICSP 接头和一个复位按钮. 它包含支持微控制器所需的一切； 只需使用 USB 电缆将其连接到计算机或使用 AC-DC 适配器或电池为其供电即可开始使用。

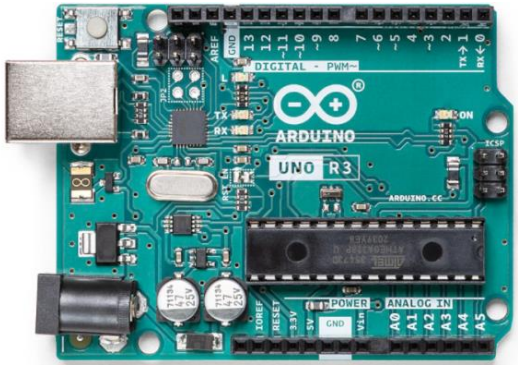


图1. **UNO R3**

I. Download UNO R3 Software IDE

https://www.arduino.cc/ensoftware

图2. **Arduino IDE**

II. 使用 UNO R3

1. 电脑和**UNO R3**用**USB**电缆连接. 这时候不用外接(9v)供电.
2. **USB**连接后在PC上被称为 Serial Port COMx. 比如在我的电脑上**USB**插上后被自动定义为 COM5.(图3)
3. 图3显示如何检查USB连接是否正常和显示当前**UNO R3 板**的有关信息.

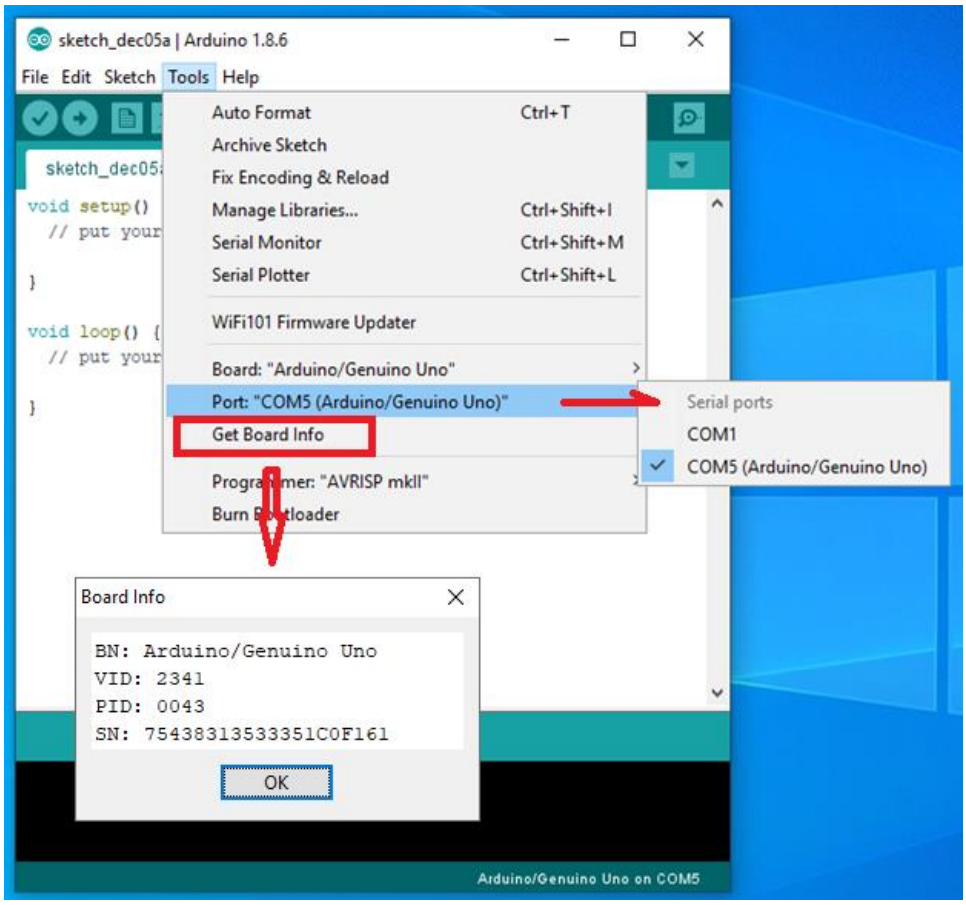


图3. 检查USB连接是否正常

III. 开发控制软件

1) UNO R3 Software IDE 写得非常方便使用. 如果是初学者,IDE本身就提供了很多 Sample Programs. 比如图4中的那个 File->Examples->Basics->Blink 程序可以作为第1个测试的程序. 因为

该程序特别简单,它不用接什么外部线路. 它的功能是令 UNO R3 板上的一个黄LED闪动.

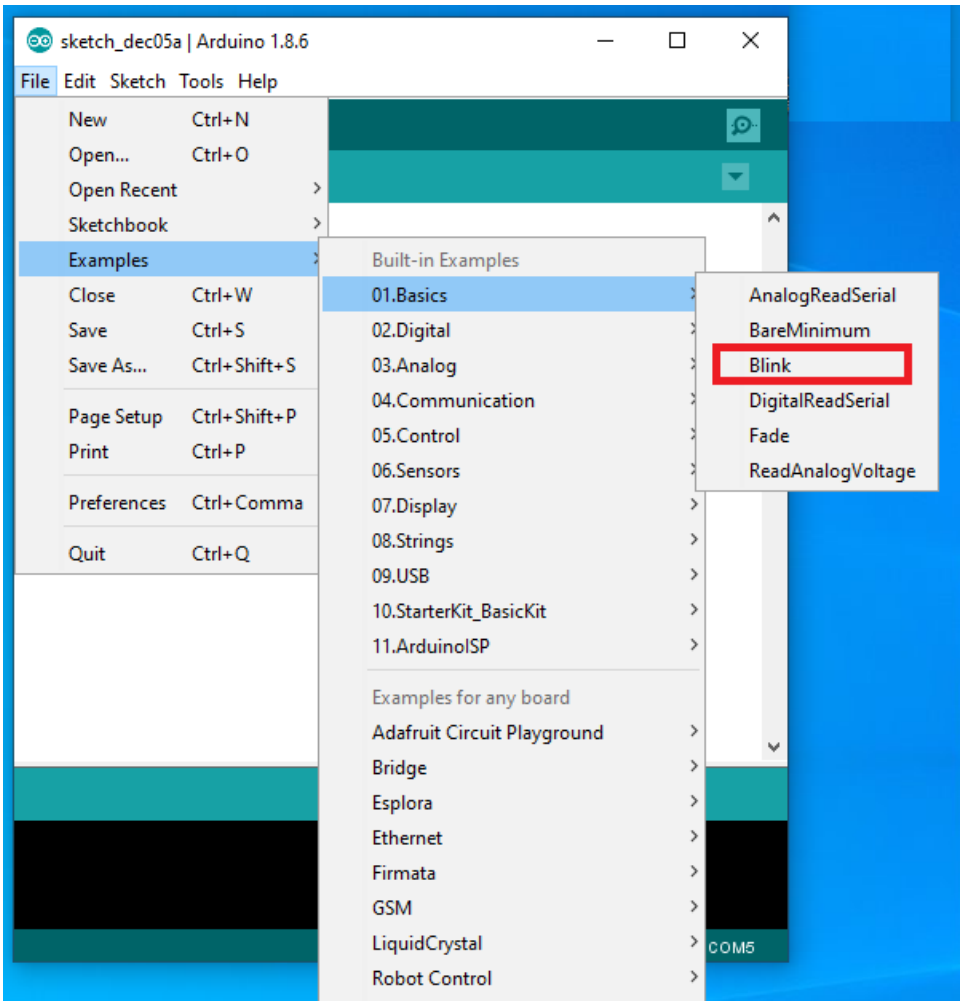


图4. 许多现成的软件可以供初学者使用

2) 编译程序和运行

编译程序和运行非常简单:

A) 先把程序装到 IDE 中. 如下图5, IDE装了Blink 程序.
B) 按一下图5中的 Load (红框) 就可完成(编译+运行).
C) 如果一切正常, 数秒钟以后Blink程序就开始执行. 这时可以看到UNO R3板上的黄色的 LED 开始闪动.

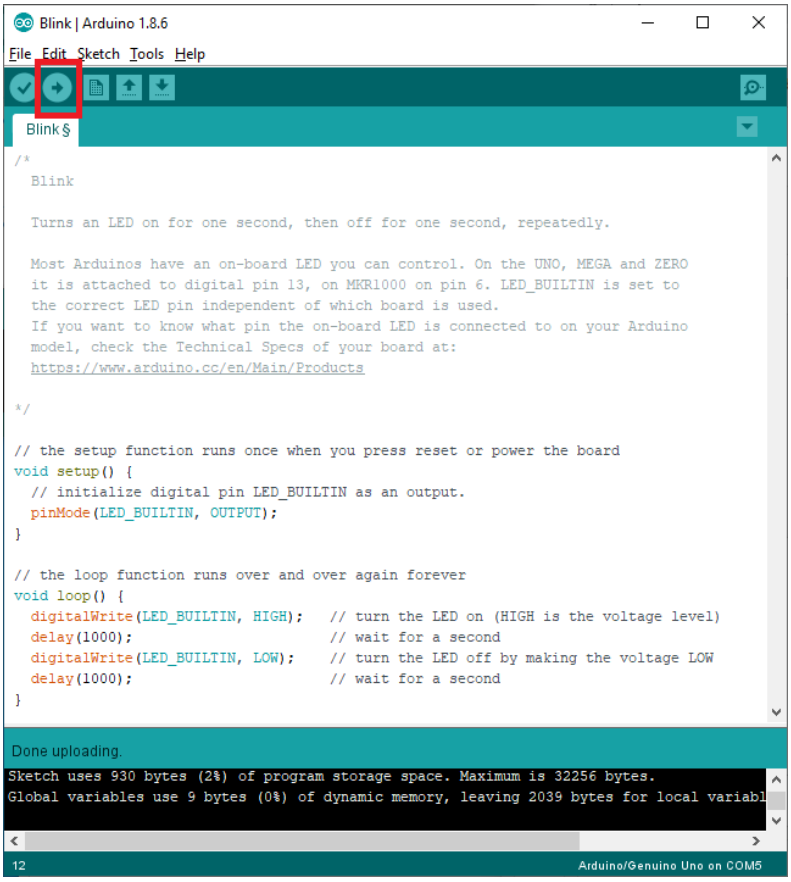


图5. Blink 程序. 按Load(红框)就可编译,运行

IV. 两个有用的工具: Serial Monitor & Serial Plotter

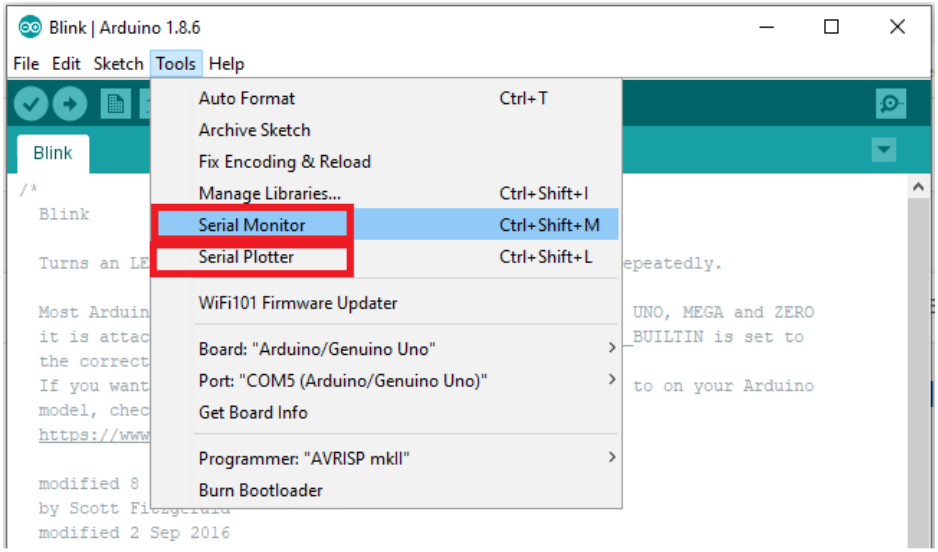


图6. Serial Monitor & Serial Plotter

这里是一个例子表明如何来使用这两个工具.
重要:在使用这两个Serial工具以前, 必须要在 setup()中加上一行程序 Serial.begin(9600);这样才能打开串行接口!

```
void setup() {
// put your setup code here, to run once:
Serial.begin(9600); //Open serial port
}
```

1) Sample Program -该程序使用 Serial.print() & Serial.println()这两个命令来显示不同的输出值. 这些变化的输出值可以在Serial Monitor 上面看到也可以在 Serial Plotter 上看到.

```
//----Begin Code (copy from here)----
//Variables
int ledPin = 6; //This is a variable. We will discuss this after the code segment!
int pwmVal = 255; //PWM range is 0 - 255. Change this value and re-upload to see
the difference!
int currVal = 0;
String serialOut = "Current Value: ";
void setup() {
        // put your setup code here, to run once:
        pinMode(ledPin, OUTPUT); //Set the pin we chose above as OUTPUT.
        Serial.begin(9600); //Open serial port
        }
void loop() {
// put your main code here, to run repeatedly:
        analogWrite(ledPin, pwmVal); //analogWrite is the arduino function for PWM.
        We are telling the Arduino to apply a PWM signal to ledPin, with the value
        of pwmVal which we set at the top of this code.

         if (pwmVal <= 0) { //If value is 0
        pwmVal = 255; //Reset the value to max ready to begin again
        }
        else { //Otherwise if value is not 0
        pwmVal--; //Decrease value by 1
        }

        Serial.print(serialOut); //Print "Current Value: "
        Serial.println(pwmVal); //Add the value to the end of the last line. NOTE
        this command is println, which adds a newline character at the end of the
        line. This saves us having to add another "\n".
}
//----End Code (copy to here)----
```

2) Serial Monitor

Serial Monitor 用于显示程序中 Serial.print() & Serial.println() 的输出值，也可以输入键盘的数据.

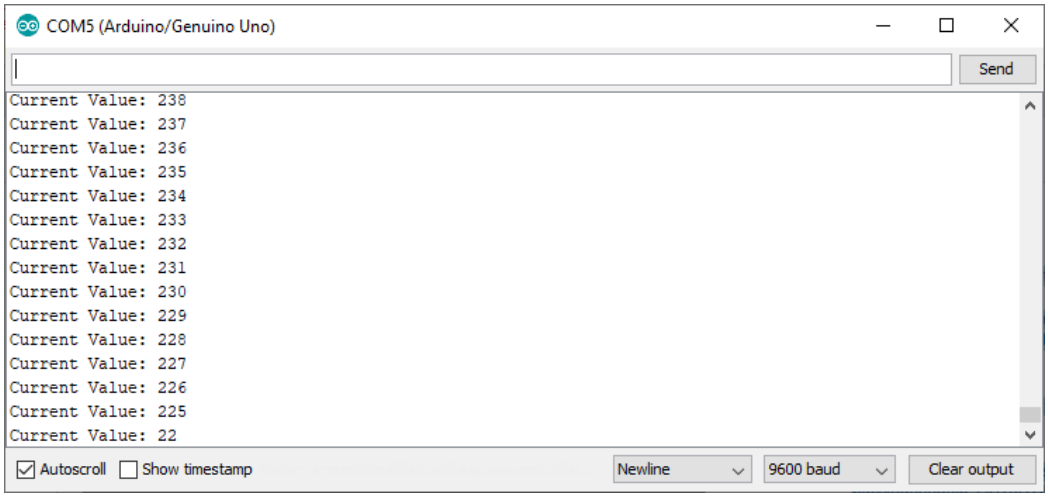


图7. Serial Monitor 用于显示程序中的输入/输出值

3) Serial Plotter

Serial Plotter 把程序中 Serial.print() & Serial.println() 的数据画成图表.

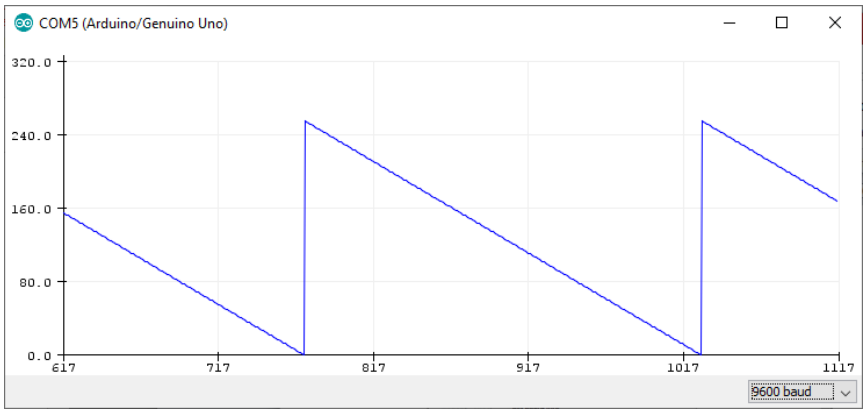


图8. Serial Plotter 把程序中的数据画成图表.

V. 一些有趣的 UNO R3 Projects

1) RGB LED 随机彩色灯

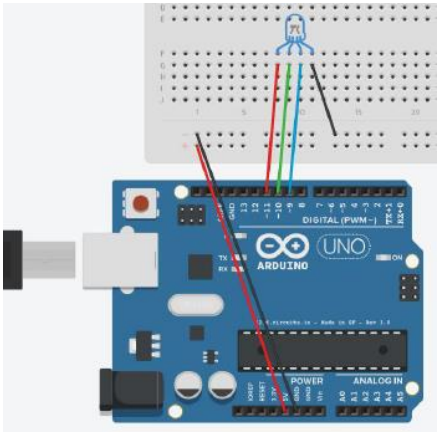
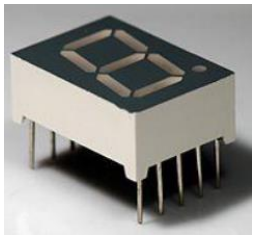


图8. 随机彩色灯

```
//----Begin Code (copy from here)----
//Variables:
int rPin = 11;
int gPin = 10;
int bPin = 9; //Set the PWM pins to be used on the arduino
int rVal = 0;
int gVal = 0;
int bVal = 0; //Set the values to 0 to begin with
void setup() {
// put your setup code here, to run once:
//No setup for this project.
}
void loop() {
// put your main code here, to run repeatedly:
analogWrite(rPin, rVal);
analogWrite(bPin, bVal);
analogWrite(gPin, gVal); //Apply PWM output to each leg of the RGB LED, with
the value stored in the corresponding variable.
rVal = random(0,255);
gVal = random(0,255);
bVal = random(0,255); //Randomise the variables to get a random color each
time
delay(500); //Delay before changing colors, so we can see each change.
}
//----End Code (copy to here)----
```

2) 7 Segments LED Display - 该程序显示 0-F 16进制数



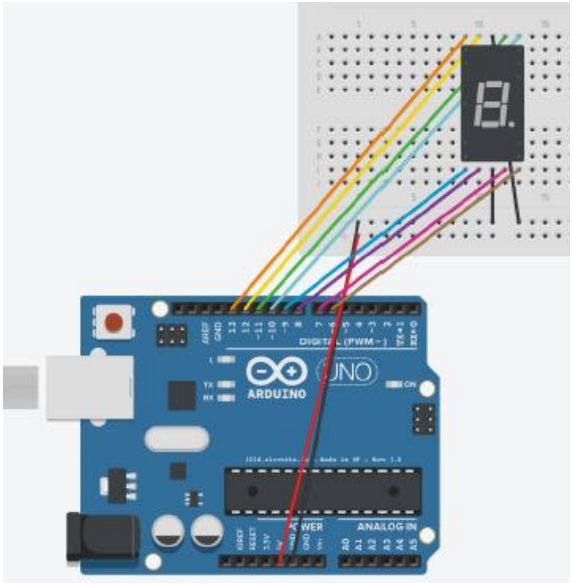


图9. 7 Segments LED

```
//----Begin Code (copy from here)----
//Variables:
int pin_a = 6;
int pin_b = 7;
int pin_c = 8;
int pin_d = 9;
int pin_e = 10;
int pin_f = 11;
int pin_g = 12;
int pin_h = 13;
int delayVar = 500;
void setup() {
// put your setup code here, to run once:
Serial.begin(9600); //Opens the Serial port and sets the data rate
Serial.print("Serial Port Started..\n");
pinMode(pin_a, OUTPUT);
pinMode(pin_b, OUTPUT);
pinMode(pin_c, OUTPUT);
pinMode(pin_d, OUTPUT);
pinMode(pin_e, OUTPUT);
pinMode(pin_f, OUTPUT);
pinMode(pin_g, OUTPUT);
pinMode(pin_h, OUTPUT);
}
void loop() {
```

```
// put your main code here, to run repeatedly:
ch_0();
Serial.print("Show #0\n");
delay(delayVar);
Serial.print("Show #1\n");
ch_1();
delay(delayVar);
Serial.print("Show #2\n");
ch_2();
delay(delayVar);
Serial.print("Show #3\n");
ch_3();
delay(delayVar);
Serial.print("Show #4\n");
ch_4();
delay(delayVar);
Serial.print("Show #5\n");
ch_5();
delay(delayVar);
Serial.print("Show #6\n");
ch_6();
delay(delayVar);
Serial.print("Show #7\n");
ch_7();
delay(delayVar);
Serial.print("Show #8\n");
ch_8();
delay(delayVar);
Serial.print("Show #9\n");
ch_9();
delay(delayVar);
Serial.print("Show #A\n");ch_a();
delay(delayVar);
Serial.print("Show #B\n");
ch_b();
delay(delayVar);
Serial.print("Show #C\n");
ch_c();
delay(delayVar);
Serial.print("Show #D\n");
ch_d();
delay(delayVar);
Serial.print("Show #E\n");
ch_e();
delay(delayVar);
Serial.print("Show #F\n");
ch_f();
delay(delayVar);
delay(delayVar);
```

```
}
void ch_a()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_b()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_c()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, LOW);
}
void ch_d()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, LOW);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, HIGH);
}
void ch_e()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
```

```
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_f()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_1()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, LOW);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, LOW);
}
void ch_2()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, HIGH);
}
void ch_3()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, LOW);
```

```
digitalWrite(pin_h, HIGH);
}
void ch_4()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, LOW);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_5()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_6()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_7()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, LOW);
}
void ch_8()
{
digitalWrite(pin_a, LOW);
```

```
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_9()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_0()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, LOW);
}
//----End Code (copy to here)----
```

3) Remote Control 遥控信号接收器

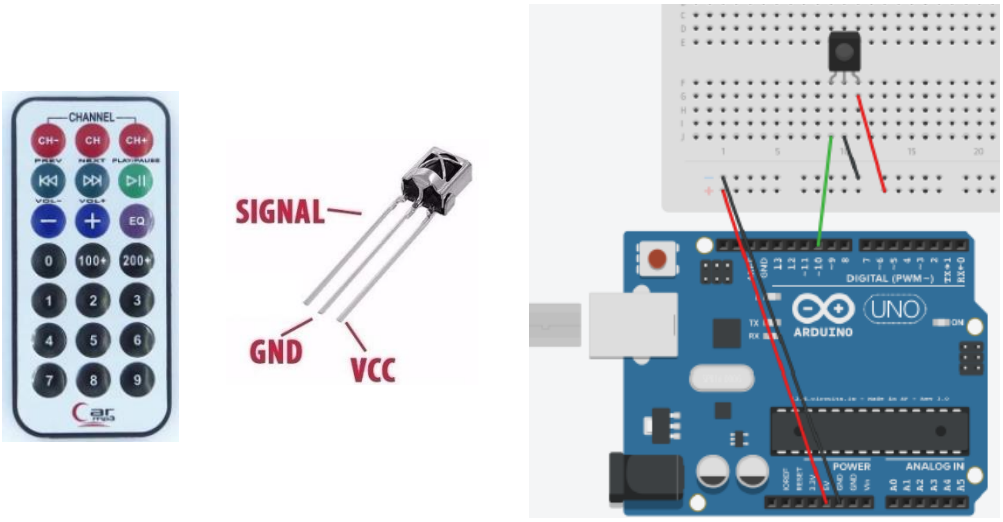
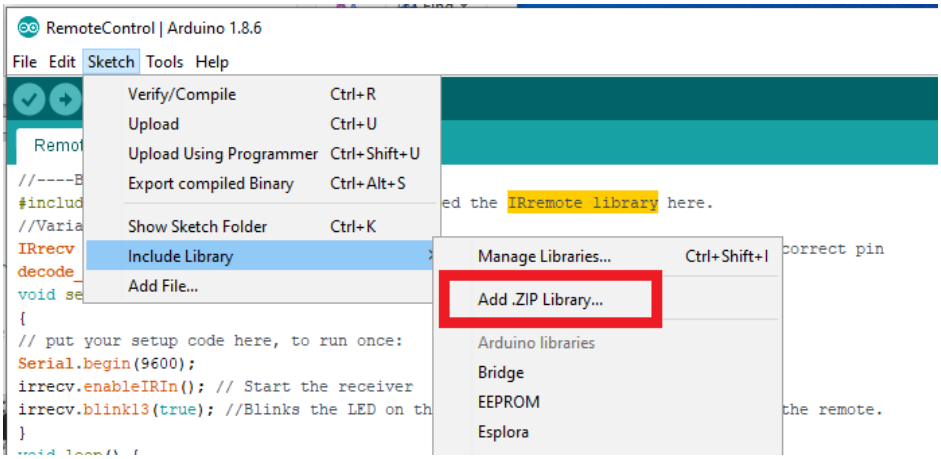


图10. Remote Control with IR Receive VS-1838B

在这个 Project 中要用到一个 3rd Party Library, ‘IRremote’ Library. 下面描述如何下载和安装.

A) Download IRremote library from:
https://www.pjrc.com/teensy/arduino_libraries/IRremote.zip

B) Install library:



NOTE: *In this project we are using a library for the first time. The library will download as a ZIP file. You should then extract the files to your Arduino installation folder/libraries*
eg: C:\Program Files (x86)\Arduino\libraries
If you have problems with this step google search “Using libraries with Arduino” for easy to follow guides.

```
//----Begin Code (copy from here)----
#include <IRremote.h> //Notice we have included the IRremote library here.
//Variables:
IRrecv irrecv(10); //Set up the IR Reciever, call it irrecv and attach it to
the correct pin
decode_results results; //Set up a variable to hold the results
void setup()
{
// put your setup code here, to run once:
Serial.begin(9600);
irrecv.enableIRIn(); // Start the receiver
irrecv.blink13(true); //Blinks the LED on the Arduino board as it gets data
from the remote.
}
void loop() {
// put your main code here, to run repeatedly:
if (irrecv.decode(&results)) {
String tmp = (String)results.value;
if (tmp == "4294967295") //This value tells the Arduino that the previous
button is still active.
{
Serial.println("Button still active."); //Report to the serial monitor the
last button was still held down
}
else {
Serial.println(results.value, HEX); //A new button has been pressed with the
value shown.
}
irrecv.resume(); // Receive the next value
}
}
//----End Code (copy to here)----
```

4) 遥控信号接收器 + 显示遥控器按的号码

该程序只是简单地把上面两个程序 （7 Segments LED Display +遥控信号接收器）合在一起. 但是因为两个程序输入输出的地方有些重复，所以下面有两处程序以及接线要改动：

图10 Pin 10 -> Pin 3

图9 Pin 13 -> Pin 5

//IRrecv irrecv(10); //Set up the IR Reciever, call it irrecv and attach it to the correct pin

```
IRrecv irrecv(3);     // William: 10 is used for 7-Segment LED. Now set it to 3.

//int pin_h = 13;     // William: Pin-13 probably is used for indicating remote signal is
                      //  received. Now set it to 5
int pin_h = 5;
```

这里要说明的是遥控接收器接到的遥控信号是十六进制(HEX)显示的,但要转换成十进制字符才能比较. 比如: 按遥控器上面数字2, 程序内接受到的16进制是 FF18E7, 要转换成十进制字符 16718055 才能比较. 见下面例子:

```
if (tmp == "16718055") {  // FF18E7 = 16718055
  Serial.println("This is Button 2");
  ch_2();
}
```

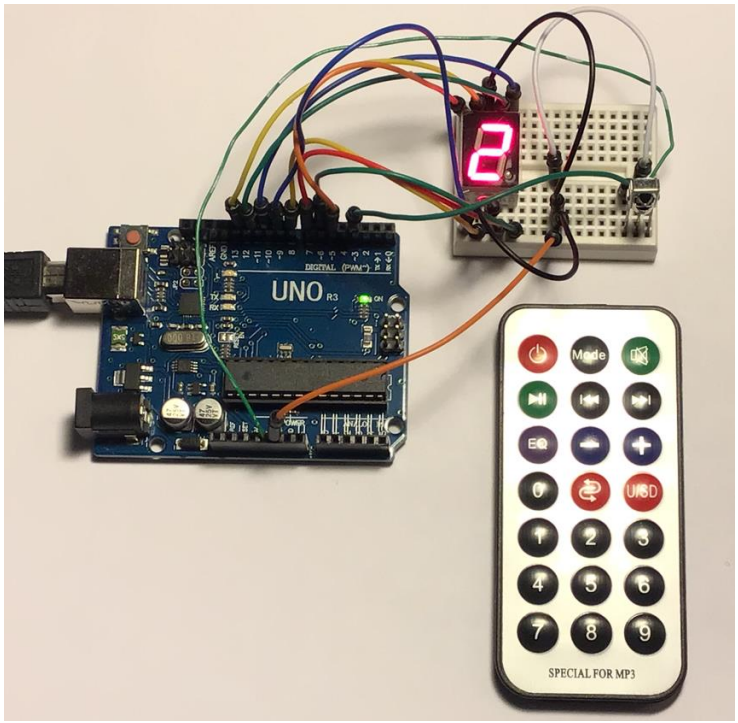


图11. 遥控信号接收器 + 显示遥控器按的号码

```
//----Begin Code (copy from here)----
#include <IRremote.h> //Notice we have included the IRremote library here.
//Variables:
```

```
//IRrecv irrecv(10); //Set up the IR Reciever, call it irrecv and attach it to the
correct pin
IRrecv irrecv(3);    // William: 10 is used for 7-Segment LED. Set it to 3.
decode_results results; //Set up a variable to hold the results

int pin_a = 6;
int pin_b = 7;
int pin_c = 8;
int pin_d = 9;
int pin_e = 10;
int pin_f = 11;
int pin_g = 12;

//int pin_h = 13;   // William: 13 probably is used for indicating remote signal is
received. Now set it to 5
int pin_h = 5;

//int delayVar = 1000;

void setup()
{
// put your setup code here, to run once:
Serial.begin(9600);
irrecv.enableIRIn(); // Start the receiver
irrecv.blink13(true); //Blinks the LED on the Arduino board as it gets data from the
remote.

pinMode(pin_a, OUTPUT);
pinMode(pin_b, OUTPUT);
pinMode(pin_c, OUTPUT);
pinMode(pin_d, OUTPUT);
pinMode(pin_e, OUTPUT);
pinMode(pin_f, OUTPUT);
pinMode(pin_g, OUTPUT);
pinMode(pin_h, OUTPUT);
}
void loop() {
// put your main code here, to run repeatedly:
if (irrecv.decode(&results)) {
String tmp = (String)results.value;
if (tmp == "4294967295") //This value tells the Arduino that the previous button is
still active.
{
Serial.println("Button still active."); //Report to the serial monitor the last
button was still held down
//Serial.println(results.value, HEX);
}
else {
```

```
Serial.println(results.value, HEX); //A new button has been pressed with the value
shown.

if (tmp == "16724175") {     // FF30CF = 16724175
  Serial.println("This is Button 1");
  ch_1();
}

if (tmp == "16718055") {      // FF18E7 = 16718055
  Serial.println("This is Button 2");
  ch_2();
}

if (tmp == "16743045") {      // FF7A85 = 16743045
  Serial.println("This is Button 3");
  ch_3();
}

if (tmp == "16716015") {      // FF10EF = 16716015
  Serial.println("This is Button 4");
  ch_4();
}

if (tmp == "16726215") {     // FF38C7 = 16726215
  Serial.println("This is Button 5");
  ch_5();
}

if (tmp == "16734885") {      // FF5AA5 = 16734885
  Serial.println("This is Button 6");
  ch_6();
}

if (tmp == "16728765") {      // FF42BD = 16728765
  Serial.println("This is Button 7");
  ch_7();
}

if (tmp == "16730805") {      // FF4AB5 = 16730805
  Serial.println("This is Button 8");
  ch_8();
}

if (tmp == "16732845") {      // FF52AD = 16732845
  Serial.println("This is Button 9");
  ch_9();
}

if (tmp == "16738455") {      // FF6897 = 16738455
```

```
  Serial.println("This is Button 0");
  ch_0();
}
}
irrecv.resume(); // Receive the next value
}
}


void ch_a()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_b()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_c()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, LOW);
}
void ch_d()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
```

```
digitalWrite(pin_f, LOW);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, HIGH);
}
void ch_e()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_f()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_1()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, LOW);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, LOW);
}
void ch_2()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, LOW);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, HIGH);
}
void ch_3()
```

```
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, HIGH);
}
void ch_4()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, LOW);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_5()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_6()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, LOW);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_7()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, LOW);
digitalWrite(pin_d, LOW);
```

```
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, LOW);
digitalWrite(pin_h, LOW);
}
void ch_8()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_9()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, LOW);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, HIGH);
}
void ch_0()
{
digitalWrite(pin_a, LOW);
digitalWrite(pin_b, HIGH);
digitalWrite(pin_c, HIGH);
digitalWrite(pin_d, HIGH);
digitalWrite(pin_e, HIGH);
digitalWrite(pin_f, HIGH);
digitalWrite(pin_g, HIGH);
digitalWrite(pin_h, LOW);
}
//----End Code (copy to here)----
```

5)Character analysis 字符分析

这个程序告诉你怎么使用字符串的各种功能. 在Serial Monitor 上面键入一个字符马上显示这个字符的各种特性. 这是众多 Exsamples 中的一个.

File->Examples->Strings-> CharacterAnalysis

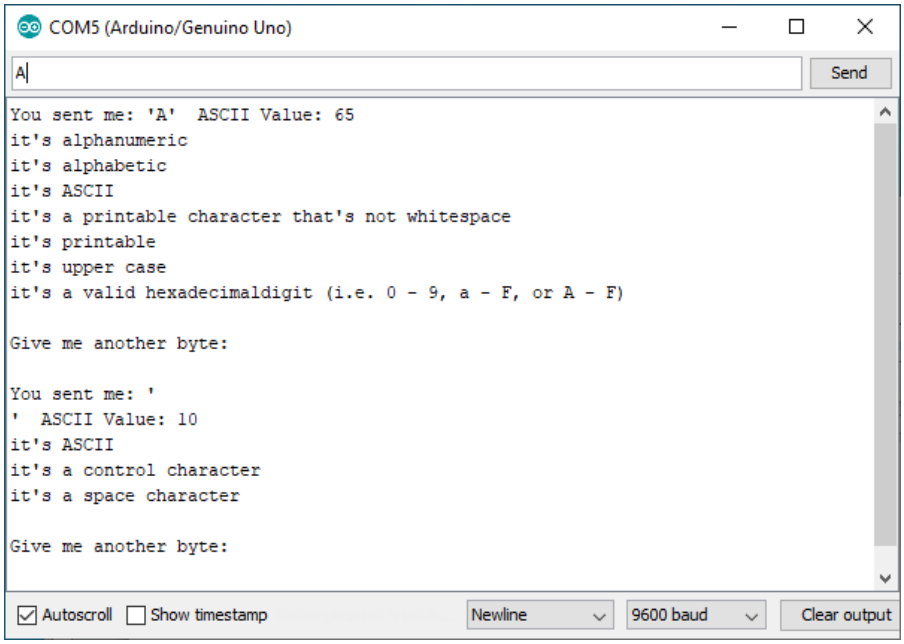


图12. Character analysis operators

```
/*
  Character analysis operators

  Examples using the character analysis operators.
  Send any byte and the sketch will tell you about it.

  created 29 Nov 2010
  modified 2 Apr 2012
  by Tom Igoe

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/CharacterAnalysis
*/

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for native USB port only
  }
```

```
  // send an intro:
  Serial.println("send any byte and I'll tell you everything I can about it");
  Serial.println();
}

void loop() {
  // get any incoming bytes:
  if (Serial.available() > 0) {
    int thisChar = Serial.read();

    // say what was sent:
    Serial.print("You sent me: \'");
    Serial.write(thisChar);
    Serial.print("\'  ASCII Value: ");
    Serial.println(thisChar);

    // analyze what was sent:
    if (isAlphaNumeric(thisChar)) {
      Serial.println("it's alphanumeric");
    }
    if (isAlpha(thisChar)) {
      Serial.println("it's alphabetic");
    }
    if (isAscii(thisChar)) {
      Serial.println("it's ASCII");
    }
    if (isWhitespace(thisChar)) {
      Serial.println("it's whitespace");
    }
    if (isControl(thisChar)) {
      Serial.println("it's a control character");
    }
    if (isDigit(thisChar)) {
      Serial.println("it's a numeric digit");
    }
    if (isGraph(thisChar)) {
      Serial.println("it's a printable character that's not whitespace");
    }
    if (isLowerCase(thisChar)) {
      Serial.println("it's lower case");
    }
    if (isPrintable(thisChar)) {
      Serial.println("it's printable");
    }
    if (isPunct(thisChar)) {
```

```
      Serial.println("it's punctuation");
    }
    if (isSpace(thisChar)) {
      Serial.println("it's a space character");
    }
    if (isUpperCase(thisChar)) {
      Serial.println("it's upper case");
    }
    if (isHexadecimalDigit(thisChar)) {
      Serial.println("it's a valid hexadecimaldigit (i.e. 0 - 9, a - F, or A -
F)");
    }

    // add some space and ask for another byte:
    Serial.println();
    Serial.println("Give me another byte:");
    Serial.println();
  }
}
```

6) How to Control a 4-digit 7-segment LED Display with an Arduino

http://www.learningaboutelectronics.com/Articles/4-digit-7-segment-LED-circuit-with-an-arduino.php

7段LED显示屏的原理很简单：(参见图13,14)

A,B,C,D,E,F,G 7段分别用来构成数字. 这里有四个数字,每个数字的7段两极管的的正极连在一起. 这个公共端就是本数字的启动端. 在显示四位数字时, 程序扫描四个启动端, 使得四个数字在时序上分别显示. 因为扫描频率快, 眼睛跟不上闪动于是就觉得四位数字是稳定同时显示.

参见图13和图14,其中 6,8,9,12 端即是四个启动端. 因为显示两极管电流比较大, 所以在下面的应用线路中加了一个三极管作大电流驱动器

重要: 7段LED显示屏两极管可以有不同的正负极性. 公共端可以是两极管的正极也可以是负极. 图16中用的是正极公共端. 如果是负极公共端(如7段LED 5461AS), 图16中的所有驱动三极管都要改为图15右边所示. 另外程序中也有一句话要修改:

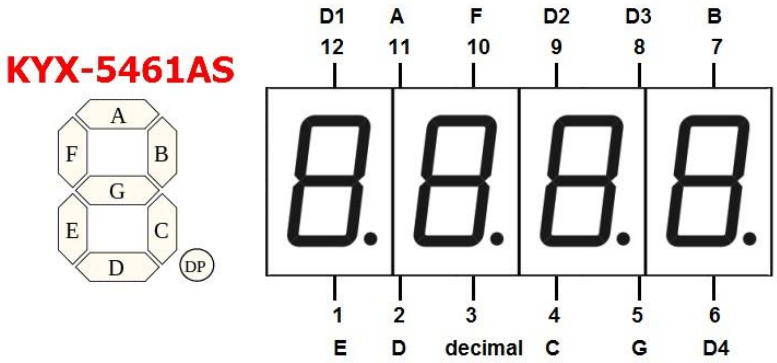


图13. 7-segment LED Display

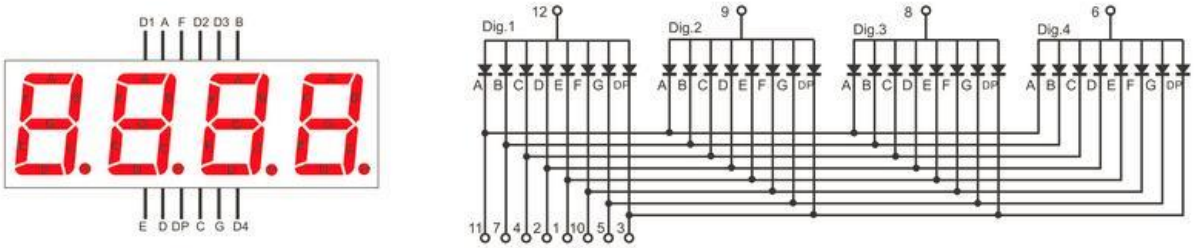


图14. 7-segment LED Display 7段两极管

```
isBitSet= ! isBitSet; //remove this line if common cathode display
```

也就是说,把这句话加上 ‘//’ 注解掉就可以:

```
// isBitSet= ! isBitSet; //remove this line if common cathode display
```

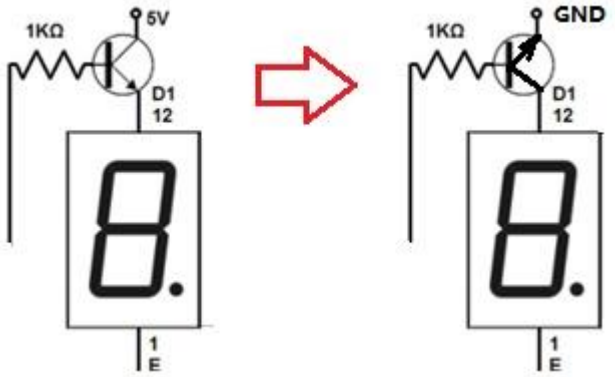


图15. 如果是负极公共端, 图16中的所有三极管都要改为右边所示

In this circuit, we will show how to display numerals on a 4-digit 7-segment display just by directing using any arduino microcontroller with no other external chips such as LED display drivers.

Simply by using transistors to each of the digits of an LED display, we can have sufficient current going to each digit of the LED display to drive them.

In this circuit, we're going to connect an analog sensor to the arduino microcontroller, simply a potentiometer because it's very easy to deal with. The value of the potentiometer will be shown on the 4-digit 7-segment display, depending of course on the resistance it is offering.

Since we are using a 4-digit LED display, the values can vary from 0 to 9999, 0 being the lowest value and 9999 being the highest. When the potentiometer is offering 0Ω of resistance, the value is 0. As the resistance that the potentiometer offers increases, the value steadily increases. When the potentiometer offers its full, maximum resistance, the value is 9999 (or close to it).

So this is a very good circuit for showing how an arduino can directly control a 4-digit 7-Segment LED display simply with only using transistors and resistors as additional components.

The transistor we use is a 2N3904 transistor. But really any NPN bipolar junction transistor such as the 2N2222 transistor also.

4-Digit 7-Segment LED Display Circuit with an Arduino

The 4-Digit 7-Segment LED circuit that we will build with an arduino microcontroller is shown below.

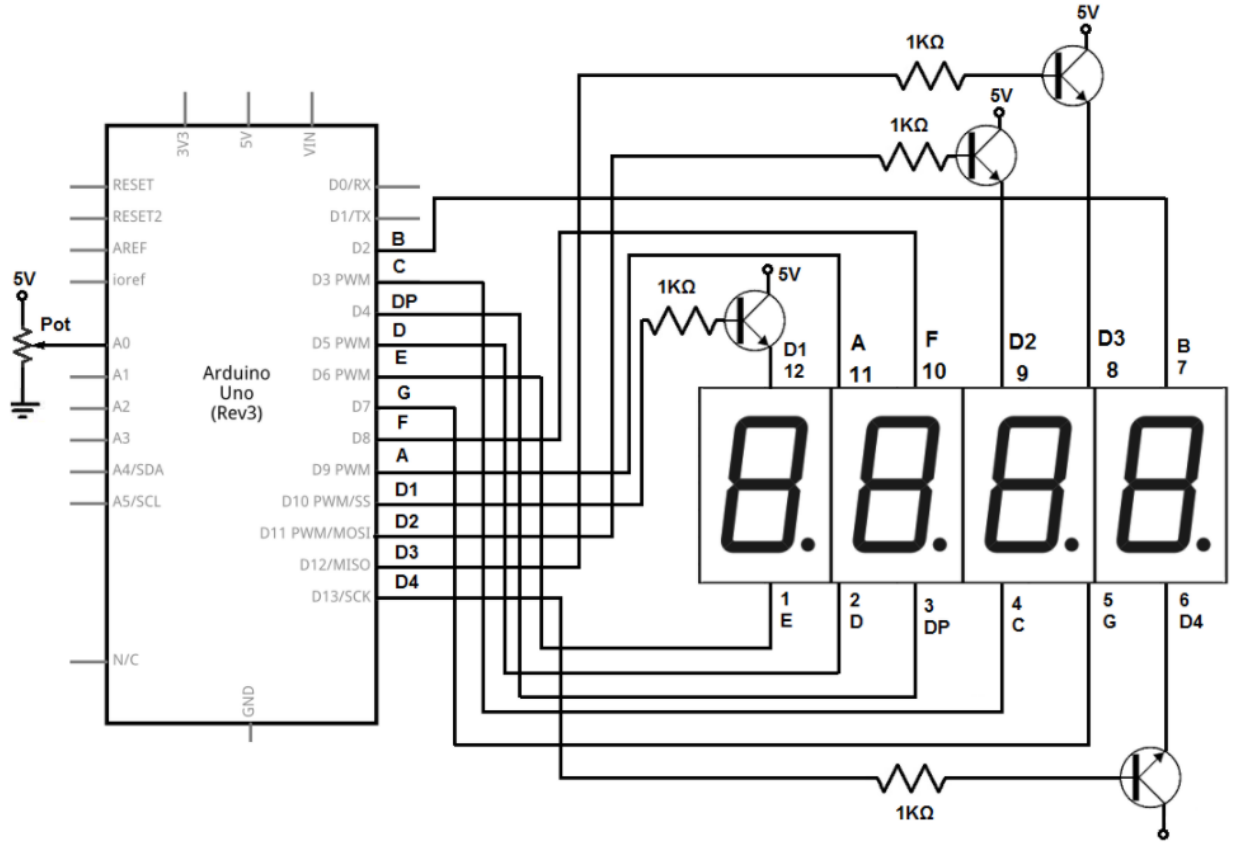


图16. UNO R3 7-segment LED Display

This code will display a number between 0 and 9999 based on the value read from the potentiometer.

The first line declares the sensorPin and initializes it to 0 because the sensor, which is the potentiometer, will be connected to analog pin 0.

The next block of code defines all the numeral values. A 7 segment LED display is made up of 8 segments. Being made up of 8 LED segments, in order to show a number on the LED display, we have to enter in which LEDs turn on in the display to show that number. All the values in this block represent the numerals based on the lit LEDs.

The next line defines the array, segmentPins[]. This array defines the pin connections for the segments of the LED display. This defines the pin connections dp, G, F, E, D, C, B, A.

The next line declares the variable, numberofDigits, as 4, since there are 4 digits in the LED display.

The next line is another array that defines digitPins[]. This array defines the pin connections for the digit pins, D1, D2, D3, and D4.

In the setup() function, the first for loop defines the segmentPins all as outputs. It goes through each of the 8 segment pins and declares each one as an output.

The second for loop, similarly, defines all the digit pins as outputs. It goes through each of the 4 digit pins and declares each one as an output.

The next block if the loop() function. In this function, we read the value from analog pin 0 which is the value read from the wiper terminal of the potentiometer; we store this number in the value variable. We then display this number on the LED display with the showNumber() function with the value variable as the parameter.

The next 2 blocks of code define the showNumber() and showDigit() define the 2 functions needed to show the value that the potentiometer reads on the LED display. We won't go into them now but they're needed in order to actually display the value on the LED display.

```
/*
   This code will display a number between 0 and 9999 based on the value read from
   the potentiometer.
*/
const int sensorPin= 0;//The analog sensor is connected to analog pin 0 of the arduino

//ABCDEFG,dp
const int numeral[10]= {
B11111100, //0
B01100000, //1
B11011010, //2
B11110010, //3
B01100110, //4
B10110110, //5
B00111110, //6
B11100000, //7
B11111110, //8
B11100110, //9
};

//pins for decimal point and each segment
//dp, G, F, E, D, C, B, A
const int segmentPins[]= { 4, 7, 8, 6, 5, 3, 2, 9};
```

```
const int numberofDigits=4;
const int digitPins[numberofDigits] = { 10,11,12, 13}; //digits 1, 2, 3, 4

void setup()
{
  for (int i=0; i < 8; i++)
      pinMode(segmentPins[i], OUTPUT); //set segment and DP pins to output

//sets the digit pins as outputs
  for (int i=0; i < numberofDigits; i++)
      pinMode(digitPins[i], OUTPUT);
}

void loop()
{
  int value= analogRead(sensorPin);
  showNumber(value);
}

void showNumber (int number)
{
  if (number == 0)
     showDigit (0, numberofDigits-1); //display 0 in the rightmost digit
  else
    {
     for (int digit= numberofDigits-1; digit >=0; digit--)
       {
         if (number > 0)
          {
            showDigit(number % 10, digit);
            number= number/10;
          }
       }
     }
}

//Displays given number on a 7-segment display at the given digit position
void showDigit (int number, int digit)
{
   digitalWrite(digitPins[digit], HIGH);
   for (int segment= 1; segment < 8; segment++)
     {
       boolean isBitSet= bitRead(numeral[number], segment);

       isBitSet= ! isBitSet; //remove this line if common cathode display
       digitalWrite(segmentPins[segment], isBitSet);
     }
   delay(5);
   digitalWrite(digitPins[digit], LOW);
}
```

7) 4x4 KeyPad 输入数字和符号

图12. 4x4 KeyPad

该程序用 4x4 KeyPad 输入数字和符号，显示在 Serial Monitor 上面. 程序需要装入 Mark Stanley 的 KeyPad Library 见下图:

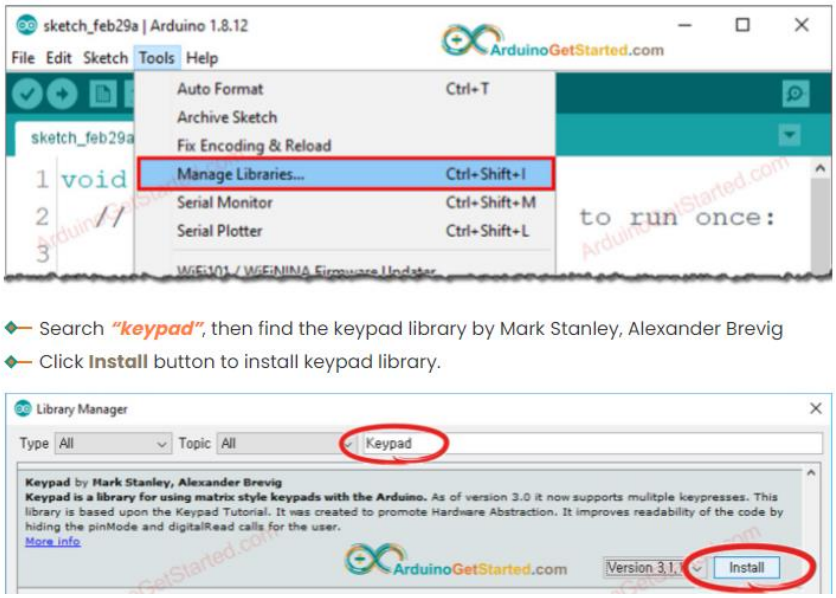


图12. Install Mark Stanley 的 KeyPad Library

```
#include <Keypad.h>

const int ROW_NUM = 4; //four rows
const int COLUMN_NUM = 4; //four columns

char keys[ROW_NUM][COLUMN_NUM] = {
  {'1','2','3', 'A'},
  {'4','5','6', 'B'},
  {'7','8','9', 'C'},
  {'*','0','#', 'D'}
};

byte pin_rows[ROW_NUM] = {9, 8, 7, 6}; //connect to the row pinouts of the keypad
byte pin_column[COLUMN_NUM] = {5, 4, 3, 2}; //connect to the column pinouts of the keypad
```

```
Keypad keypad = Keypad( makeKeymap(keys), pin_rows, pin_column, ROW_NUM, COLUMN_NUM );

void setup(){
  Serial.begin(9600);
}

void loop(){
  char key = keypad.getKey();

  if (key){
    Serial.println(key);
  }
}
```

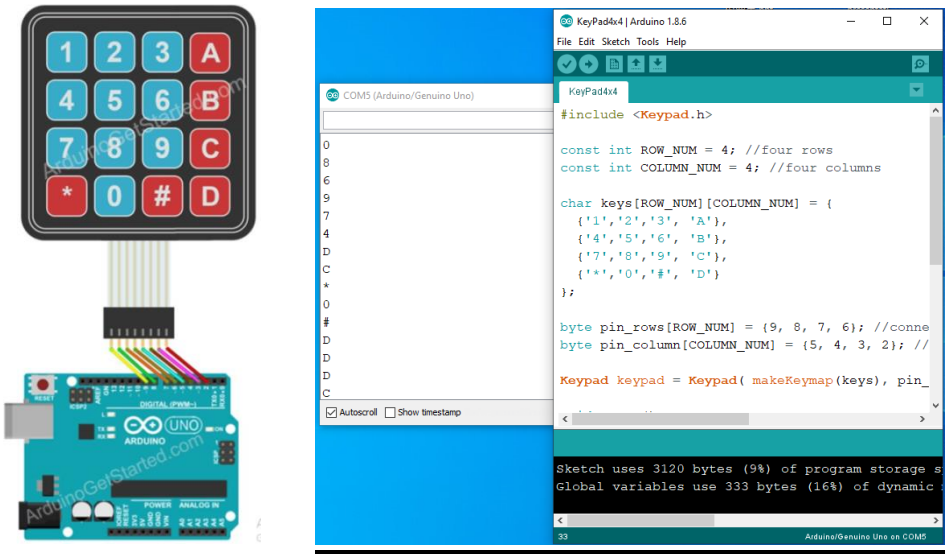


图14. 4x4 KeyPad 接线图

8) 4x4 KeyPad Calculator 计算器

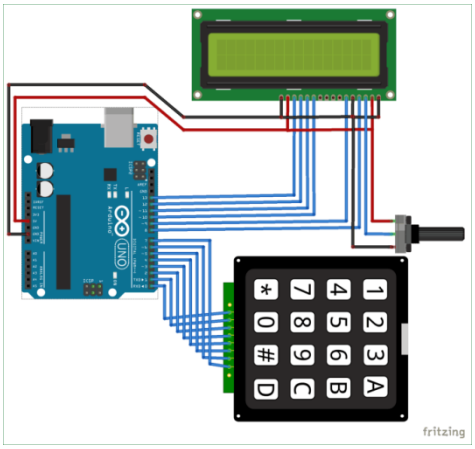


图15. 计算器

https://circuitdigest.com/microcontroller-projects/arduino-calculator-using-4x4-keypad

Since the keypad used here does not have proper markings on it I have assumed the Alphabets to be operators as listed below

Character on Keypad	Assumed to be
"A"	Addition (+)
"B"	Subtraction (-)
"C"	Multiplication (*)
"D"	Division (/)
"*"	Clear (C)
"#"	Equals (=)

```
/*
 * Arduino Keypad calculator Program
 */
#include <LiquidCrystal.h> //Header file for LCD
from https://www.arduino.cc/en/Reference/LiquidCrystal
#include <Keypad.h> //Header file for Keypad from https://github.com/Chris--A/Keypad
const byte ROWS = 4; // Four rows
const byte COLS = 4; // Three columns
// Define the Keymap
char keys[ROWS][COLS] = {
  {'7','8','9','D'},
  {'4','5','6','C'},
  {'1','2','3','B'},
  {'*','0','#','A'}
};
byte rowPins[ROWS] = { 0, 1, 2, 3 };// Connect keypad ROW0, ROW1, ROW2 and ROW3 to these
Arduino pins.
byte colPins[COLS] = { 4, 5, 6, 7 }; // Connect keypad COL0, COL1 and COL2 to these Arduino
pins.
Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS ); //  Create the Keypad
const int rs = 8, en = 9, d4 = 10, d5 = 11, d6 = 12, d7 = 13; //Pins to which LCD is
connected
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
 long Num1,Num2,Number;
 char key,action;
 boolean result = false;

void setup() {
```

```
  lcd.begin(16, 2); //We are using a 16*2 LCD display
  lcd.print("DIY Calculator"); //Display a intro message
  lcd.setCursor(0, 1);   // set the cursor to column 0, line 1
  lcd.print("-CircuitDigest"); //Display a intro message
   delay(2000); //Wait for display to show info
    lcd.clear(); //Then clean it
}
void loop() {

key = kpd.getKey(); //storing pressed key value in a char
if (key!=NO_KEY)
DetectButtons();
if (result==true)
CalculateResult();
DisplayResult();
}
void DetectButtons()
{
     lcd.clear(); //Then clean it
    if (key=='*') //If cancel Button is pressed
    {Serial.println ("Button Cancel"); Number=Num1=Num2=0; result=false;}

     if (key == '1') //If Button 1 is pressed
    {Serial.println ("Button 1");
    if (Number==0)
    Number=1;
    else
    Number = (Number*10) + 1; //Pressed twice
    }

     if (key == '4') //If Button 4 is pressed
    {Serial.println ("Button 4");
    if (Number==0)
    Number=4;
    else
    Number = (Number*10) + 4; //Pressed twice
    }

     if (key == '7') //If Button 7 is pressed
    {Serial.println ("Button 7");
    if (Number==0)
    Number=7;
    else
    Number = (Number*10) + 7; //Pressed twice
    }

    if (key == '0')
    {Serial.println ("Button 0"); //Button 0 is Pressed
    if (Number==0)
```

```
Number=0;
else
Number = (Number*10) + 0; //Pressed twice
}

 if (key == '2') //Button 2 is Pressed
{Serial.println ("Button 2");
 if (Number==0)
Number=2;
else
Number = (Number*10) + 2; //Pressed twice
}

 if (key == '5')
{Serial.println ("Button 5");
 if (Number==0)
Number=5;
else
Number = (Number*10) + 5; //Pressed twice
}

 if (key == '8')
{Serial.println ("Button 8");
 if (Number==0)
Number=8;
else
Number = (Number*10) + 8; //Pressed twice
}

if (key == '#')
{Serial.println ("Button Equal");
Num2=Number;
result = true;
}

 if (key == '3')
{Serial.println ("Button 3");
 if (Number==0)
Number=3;
else
Number = (Number*10) + 3; //Pressed twice
}

 if (key == '6')
{Serial.println ("Button 6");
if (Number==0)
Number=6;
else
Number = (Number*10) + 6; //Pressed twice
```

```
    }

     if (key == '9')
    {Serial.println ("Button 9");
    if (Number==0)
    Number=9;
    else
    Number = (Number*10) + 9; //Pressed twice
    }
      if (key == 'A' || key == 'B' || key == 'C' || key == 'D') //Detecting Buttons on
Column 4
  {
    Num1 = Number;
    Number =0;
    if (key == 'A')
    {Serial.println ("Addition"); action = '+';}
     if (key == 'B')
    {Serial.println ("Subtraction"); action = '-'; }
     if (key == 'C')
    {Serial.println ("Multiplication"); action = '*';}
     if (key == 'D')
    {Serial.println ("Devesion"); action = '/';}
    delay(100);
  }

}
void CalculateResult()
{
  if (action=='+')
    Number = Num1+Num2;
  if (action=='-')
    Number = Num1-Num2;
  if (action=='*')
    Number = Num1*Num2;
  if (action=='/')
    Number = Num1/Num2;
}
void DisplayResult()
{
  lcd.setCursor(0, 0);   // set the cursor to column 0, line 1
  lcd.print(Num1); lcd.print(action); lcd.print(Num2);

  if (result==true)
  {lcd.print(" ="); lcd.print(Number);} //Display the result

  lcd.setCursor(0, 1);   // set the cursor to column 0, line 1
  lcd.print(Number); //Display the result
}
```

9) 使用 LCD（液晶显示器）

WH1602 V2 Character 16x2

1. 5x8 dots includes cursor
2. Built-in controller (ST 7066 or Equivalent)
3. +5.0V power supply
4.1/16 duty cycle

Pin NO.	Symbol	Function
1	VSS	Ground
2	VDD	Supply voltage for logic
3	VO	Contrast Adjustment
4	RS	H: Data L: Instruction
5	R/$\overline{W}$	H: read data L: write data
6	E	Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	A	LED +
16	K	LED -

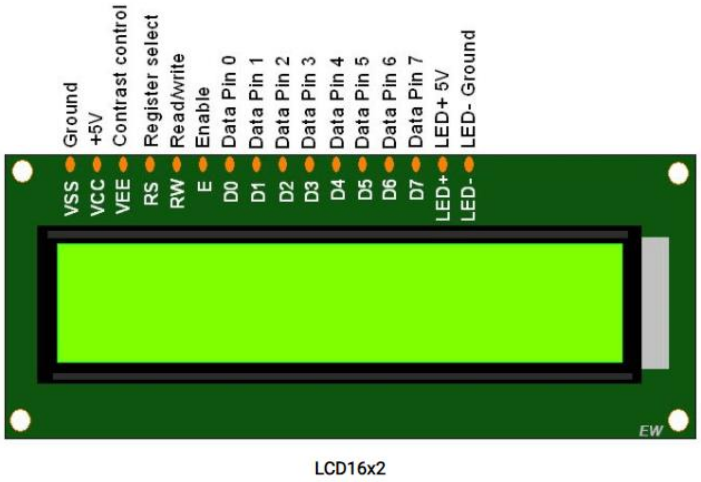


图16. WH1602 V2 液晶显示器

16x2 LCD Pinout Configuration

Pin #	Pin Name	Description
1	Vss (Ground)	Ground pin connected to system ground
2	Vdd (+5 Volt)	Powers the LCD with +5V (4.7V – 5.3V)
3	VE (Contrast V)	Decides the contrast level of display. Grounded to get maximum contras
4	Register Select	Connected to Microcontroller to shift between command/data register
5	Read/Write	Used to read or write data. Normally grounded to write data to LCD
6	Enable	Connected to Microcontroller Pin and toggled between 1 and 0 for data acknowledge
7	Data Pin 0	Data pins 0 to 7 forms a 8-bit data line. They can be connected to Microcontroller to send 8-bit data. These LCD's can also operate on 4-bit mode in such case Data pin 4,5,6 and 7 will be left free.
8	Data Pin 1	
9	Data Pin 2	
10	Data Pin 3	
11	Data Pin 4	
12	Data Pin 5	
13	Data Pin 6	
14	Data Pin 7	
15	LED Positive	Backlight LED pin positive terminal
16	LED Negative	Backlight LED pin negative terminal

小结：16x2 LCD 共有16个外节点：

电源: Pin 1 (Vss) 接地, Pin 2 (Vdd) +5v,
显示对比度: Pin 3 (VE). 接地显示最大对比度.
命令/数据: Pin 4 (Register Select). H :Data, L:Instruction
读/写: Pin 5 (Read/Write). H:Read, L:Write
启用: Pin 6 (Enable)
数据输入: Pin 7 - Pin 14 (Data 0 - Data 7)
背景光LED电源: Pin 15 (Positive, +5v), Pin 16(Negative)

以下程序显示两行字并且不断左移:
第1行: SunFounder
第2行: hello, world!
左移, 显示命令是:

```
lcd.scrollDisplayLeft();  //Scrolls the contents of the display one space to the left.
lcd.print(array1[positionCounter1]);  // Print a message to the LCD.
```

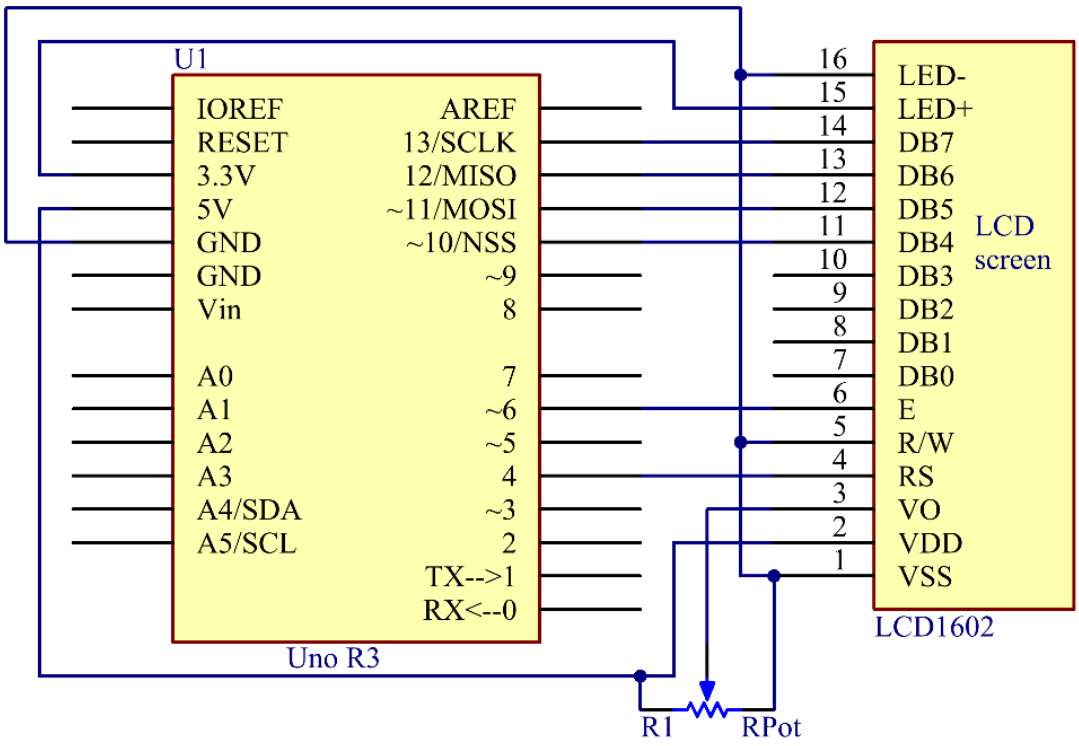


图16. 16x2 LCD Project 线路图. RPot(10K)

```
//LCD1602
//You should now see your LCD1602 display the flowing characters "SUNFOUNDER" and "hello,
world"
//Email:info@primerobotics.in
//Website:www.primerobotics.in
//2015.5.7
#include <LiquidCrystal.h>// include the library code
/*********************************************************/
char array1[]=" SunFounder               ";  //the string to print on the LCD
char array2[]="hello, world!             ";  //the string to print on the LCD
int tim = 500;  //the value of delay time
// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(4, 6, 10, 11, 12, 13);
/*********************************************************/
void setup()
{
  lcd.begin(16, 2);  // set up the LCD's number of columns and rows:
}
/*********************************************************/
void loop()
{
    lcd.setCursor(15,0);  // set the cursor to column 15, line 0
    for ( int positionCounter1 = 0; positionCounter1 < 26; positionCounter1++)
    {
      lcd.scrollDisplayLeft();  //Scrolls the contents of the display one space to the
left.
      lcd.print(array1[positionCounter1]);  // Print a message to the LCD.
      delay(tim);  //wait for 250 ms
    }
    lcd.clear();  //Clears the LCD screen and positions the cursor in the upper-left
corner.
    lcd.setCursor(15,1);  // set the cursor to column 15, line 1
    for (int positionCounter2 = 0; positionCounter2 < 26; positionCounter2++)
    {
      lcd.scrollDisplayLeft();  //Scrolls the contents of the display one space to the
left.
      lcd.print(array2[positionCounter2]);  // Print a message to the LCD.
      delay(tim);  //wait for 250 ms
    }
    lcd.clear();  //Clears the LCD screen and positions the cursor in the upper-left
corner.
}
/*********************************************************/
```

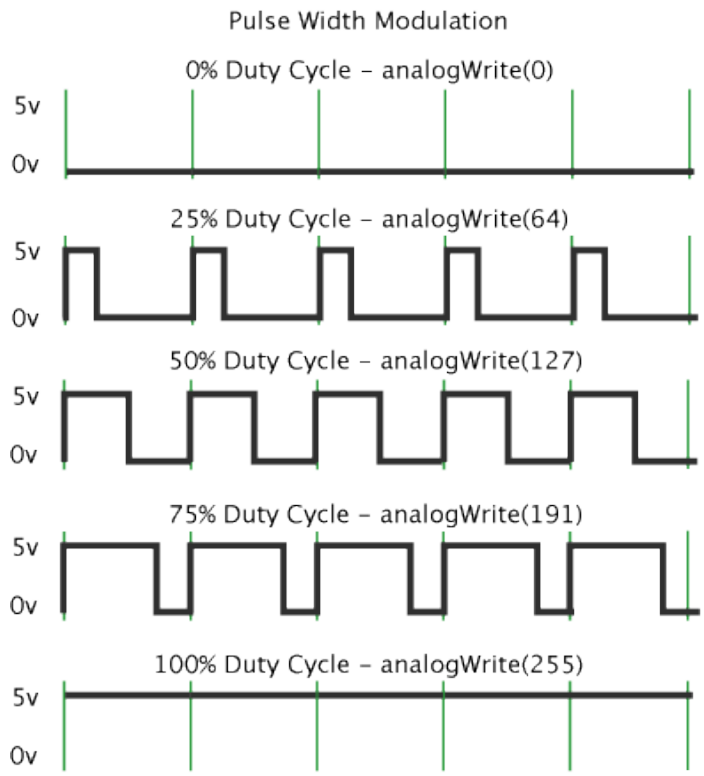
Appendix

What is PWM

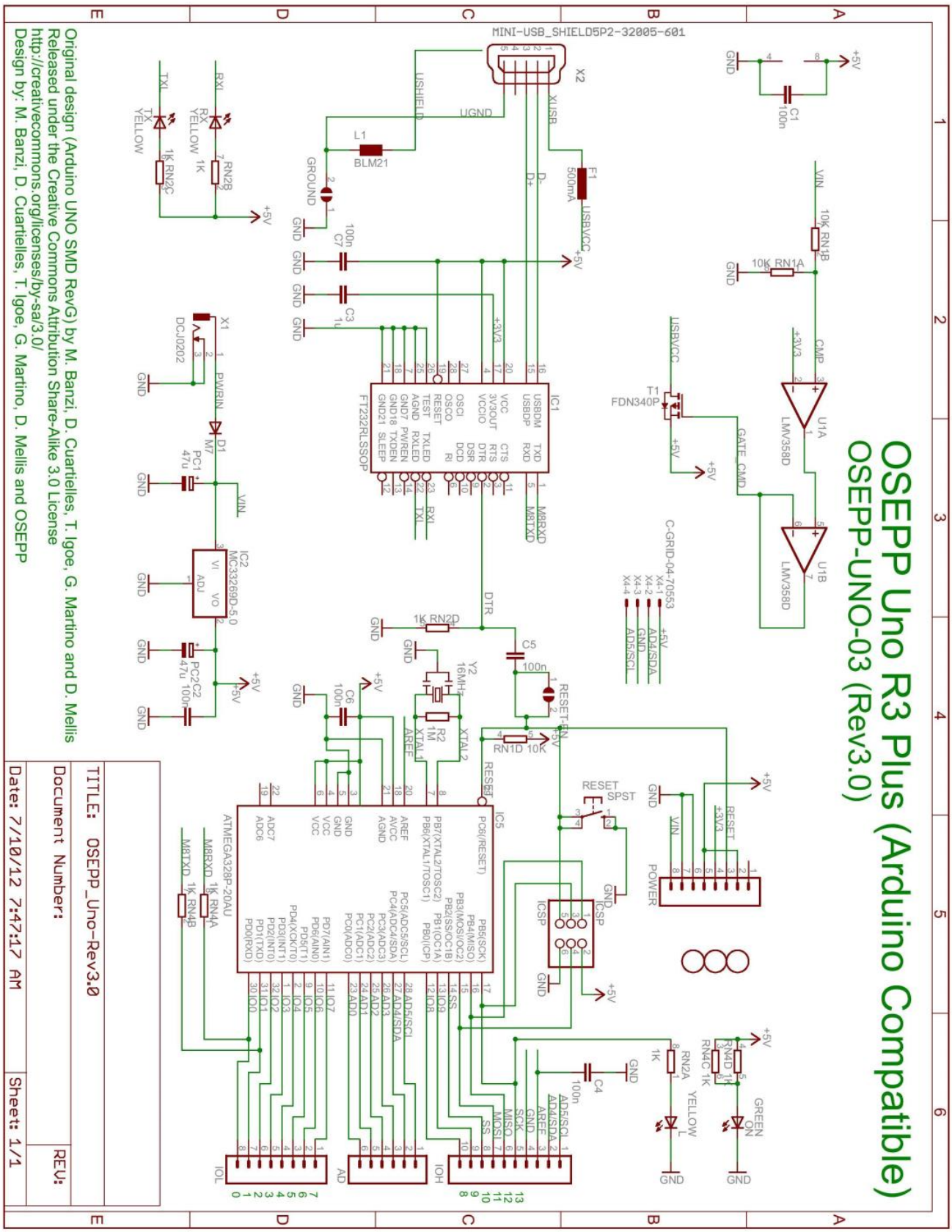
The Fading example demonstrates the use of analog output (PWM) to fade an LED. It is available in the File->Sketchbook->Examples->Analog menu of the Arduino software.

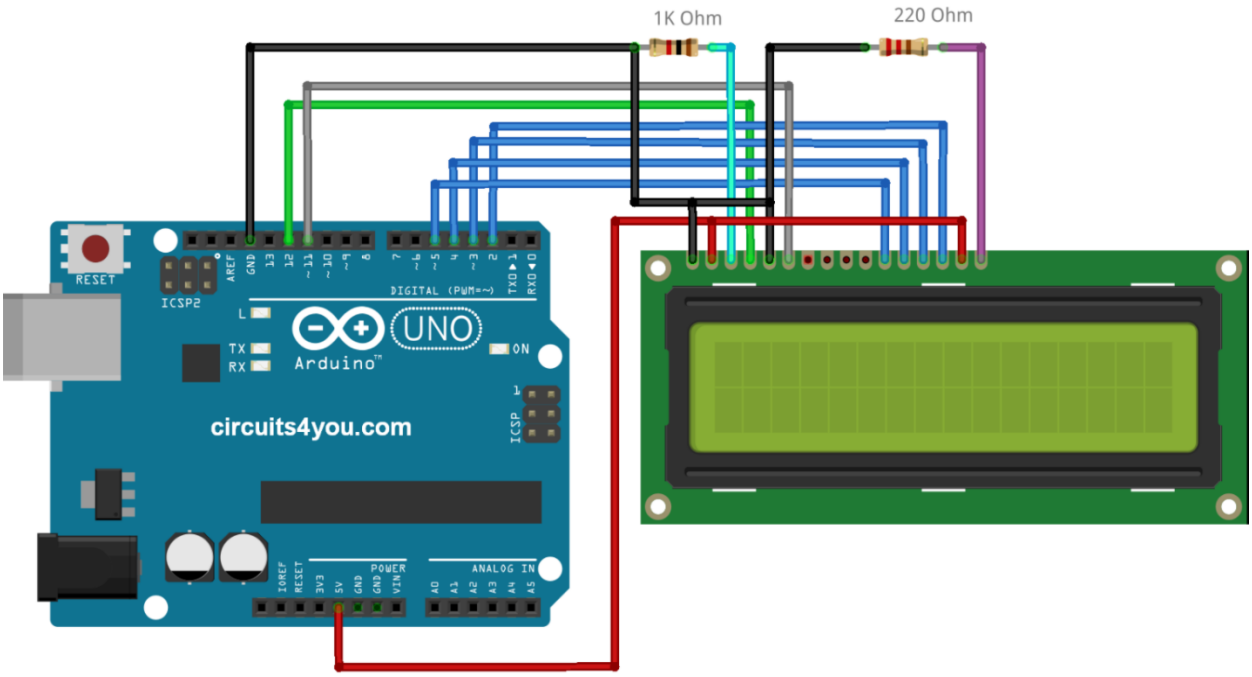
Pulse Width Modulation, or PWM, **is a technique for getting analog results with digital means.** Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between the full Vcc of the board (e.g., 5 V on Uno, 3.3 V on a MKR board) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change, or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED for example, the result is as if the signal is a steady voltage between 0 and Vcc controlling the brightness of the LED.

In the graphic below, the green lines represent a regular time period. This duration or period is the inverse of the PWM frequency. In other words, with Arduino's PWM frequency at about 500Hz, the green lines would measure 2 milliseconds each. A call to analogWrite() is on a scale of 0 - 255, such that analogWrite(255) requests a 100% duty cycle (always on), and analogWrite(127) is a 50% duty cycle (on half the time) for example.



Once you get this example running, grab your arduino and shake it back and forth. What you are doing here is essentially mapping time across the space. To our eyes, the movement blurs each LED blink into a line. As the LED fades in and out, those little lines will grow and shrink in length. Now you are seeing the pulse width.


OSEPP Uno R3 Plus (Arduino Compatible)
OSEPP-UNO-03 (Rev3.0)
Original design (Arduino UNO SMD RevG) by M. Banzi, D. Cuartielles, T. Igoe, G. Martino and D. Mellis
Released under the Creative Commons Attribution Share-Alike 3.0 License
http://creativecommons.org/licenses/by-sa/3.0/
Design by: M. Banzi, D. Cuartielles, T. Igoe, G. Martino, D. Mellis and OSEPP
TITLE: OSEPP_Uno-Rev3.0
Document Number:
REV:
Date: 7/10/12 7:47:17 AM
Sheet: 1/1
MINI-USB_SHIELD5P2-32005-601
FT232RLSSOP
ATMEGA328P-20AU
MC33269D-5.0
FDN340P
LMV358D
C-GRID-04-7053
DCJ0202
BLM21
500mA
16MHz
RESET SPST
ICSP
POWER
YELLOW
GREEN
ON


1K Ohm
220 Ohm
RESET
ICSP2
UNO
Arduino
TX
RX
ON
circuits4you.com
ICSP
DIGITAL (PWM=~)
POWER
ANALOG IN