



ARDUINO 学习手册

修订历史

版本	日期	原因
V1.0	2013/12/19	第一次发布
V1.1	2014/2/28	增加部分问题解决办法
V2.0	2014/7/9	增加提高版和至尊版例程

本套教程仅供本店或者本店代理售出的套件客户使用，鄙视其它店铺盗用视频的行为，请客户睁大眼睛，购买我们套件的享有在技术支持群咨询问题的权利，这是其它店铺给不了的，请善用这些权利。另外我们的套件视频会与时俱进，不断更新，现在已经录制第三期视频了。而其他盗用我们店铺视频大多还在使用我们第一个版本的视频，永远跟不上我们的步伐~



1. ARDUINO 简介

1.1 什么是 ARDUINO?

Arduino 是一个能够用来感应和控制现实物理世界的一套工具。它由一个基于单片机并且开放源码的硬件平台，和一套为 Arduino 板编写程序的开发环境组成。

Arduino 可以用来开发交互产品，比如它可以读取大量的开关和传感器信号，并且可以控制各式各样的电灯、电机和其他物理设备。Arduino 项目可以是单独的，也可以在运行时和你电脑中运行的程序（例如：Flash, Processing, MaxMSP）进行通讯。Arduino 板你可以选择自己去手动组装或是购买已经组装好的；Arduino 开源的 IDE 可以免费下载得到。

Arduino 的编程语言就像似在对一个类似于物理的计算平台进行相应的连线，它基于处理多媒体的编程环境。

1.2 为什么要选择 ARDUINO?




有很多的单片机和单片机平台都适合用做交互式系统的设计。例如：Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard 和其它等等提供类似功能的。所有这些工具，你都不需要去关心单片机编程繁琐的细节，提供给你的是一套容易使用的工具包。Arduino 同样也简化了同单片机工作的流程，但同其它系统相比 Arduino 在很多地方更具有优越性，特别适合老师，学生和一些业余爱好者们使用：

- 便宜 — 和其它平台相比，Arduino 板算是相当便宜了。最便宜的 Arduino 版本可以自己动手制作，即使是组装好的成品，其价格也不会超过 200 元。
- 跨平台 — Arduino 软件可以运行在 Windows, Macintosh OSX, 和 Linux 操作系统。大部分其它的单片机系统都只能运行在 Windows 上。
- 简易的编程环境 — 初学者很容易就能学会使用 Arduino 编程环境，同时它又能为高级用户提供足够多的高级应用。对于老师们来说，一般都能很方便的使用 Processing 编程环境，所以如果学生学习过使用 Processing 编程环境的话，那他们在使用 Arduino 开发环境的时候就会觉得很相似很熟悉。
- 软件开源并可扩展 — Arduino 软件是开源的，对于有经验的程序员可以对其进行扩展。Arduino 编程语言可以通过 C++ 库进行扩展，如果有人想去了解技术上的细节，可以跳过 Arduino 语言而直接使用 AVR C 编程语言（因为 Arduino 语言实际上是基于 AVR C 的）。类似的，如果你需要的话，你也可以直接往你的 Arduino 程序中添加 AVR-C 代码。
- 硬件开源并可扩展 — Arduino 板基于 Atmel 的 ATMEGA8 和 ATMEGA168/328 单片机。Arduino 基于 Creative Commons 许可协议，所以有经验的电路设计师能够根据需求设计自己的模块，可以对其扩展或改进。甚至是对于一些相对没有什么经验的用户，也可以通过制作试验板来理解 Arduino 是怎么工作的，省钱又省事。



1.3 本店 ARDUINO 套件标配的核心板对比

基本性能配置对比：

	Arduino UNO	Arduino UNO 加强版	Arduino Leonardo
外观			
主控型号	Atmega328p	Atmega328p	Atmega32u4
供电电压	5v	5v	5v
建议供电电压	DC7-12v	DC7-12v	DC7-12v
极限供电电压	DC6-20V	DC6-20V	DC6-20V
可用 IO	20	20	20
PWM	7	7	7
模拟输入 IO	6	6	12
Flash Memory	32KB	32KB	32KB
SRAM	2KB	2KB	2.5KB
EEPROM	1KB	1KB	1KB
串口	1 个	1 个	1 个硬件 com+1 个 usb 虚拟 com
USB 控制器	无	无	1 个
5V 输出端口	1	5	1
3.3v 电源输出最大电流	150mA	800mA	150mA
外扩 TTL 模块接口	无	有	无
系统驱动安装兼容性	Win7/win8 精简版装不上	所有系统都能装上	Win7/win8 精简版装不上

2 ARDUINO 学习基础准备工作

2.1 ARDUINO 开发环境简介

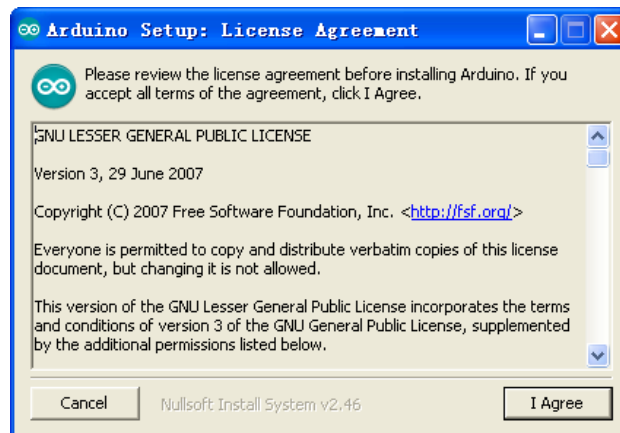


Arduino 的开发环境地址:

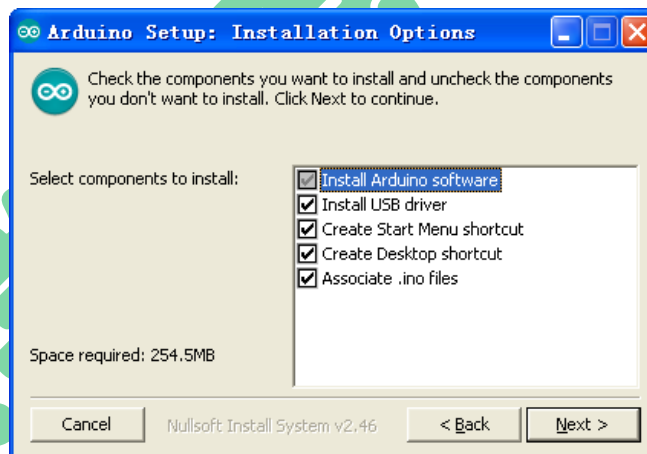
- ✓ 在 **Arduino 资料包\1.开发环境\arduino-1.0.5-windows.exe** 找到
- ✓ 在 Arduino 官网下载最新开发环境, 地址 <http://arduino.cc/en/Main/Software>

2.2 ARDUINO 开发环境安装

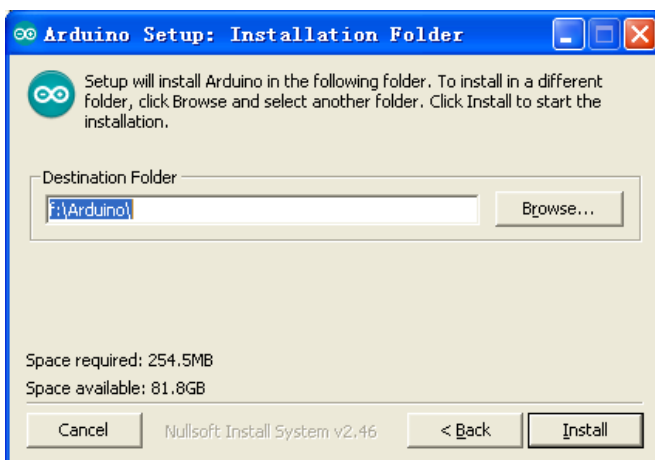
第一步:打开安装包 **arduino-1.0.5-windows.exe**



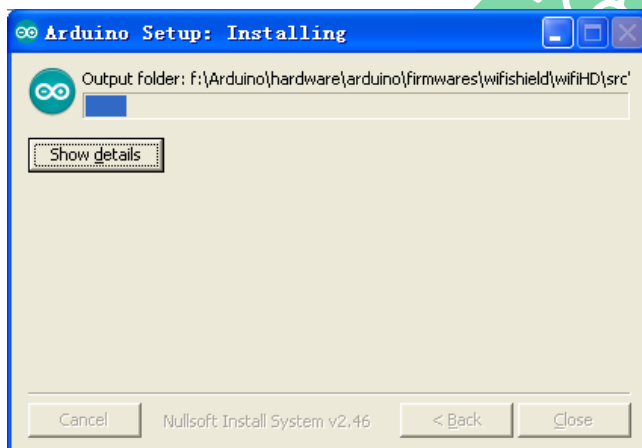
第二步: 点击 **I Agree**



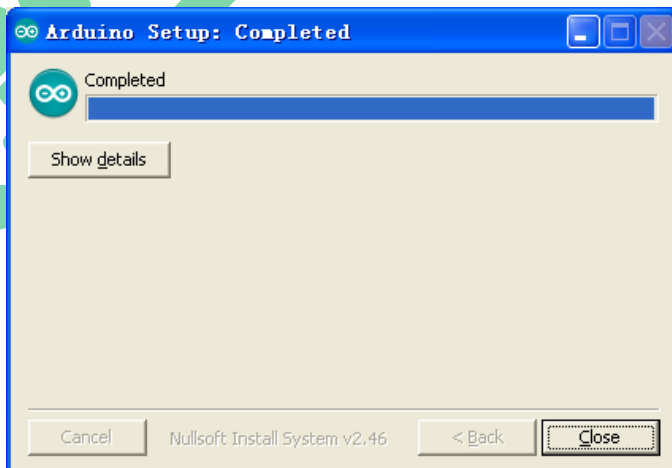
第三步: 点击 **NEXT**



第四步：选择自己的安装路径，点击 **Install**



第五步：等待安装完成



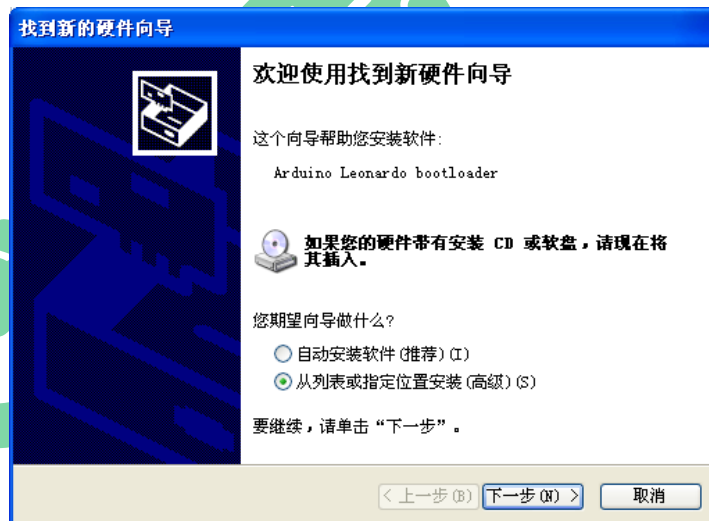
第六步：点击 **Close**



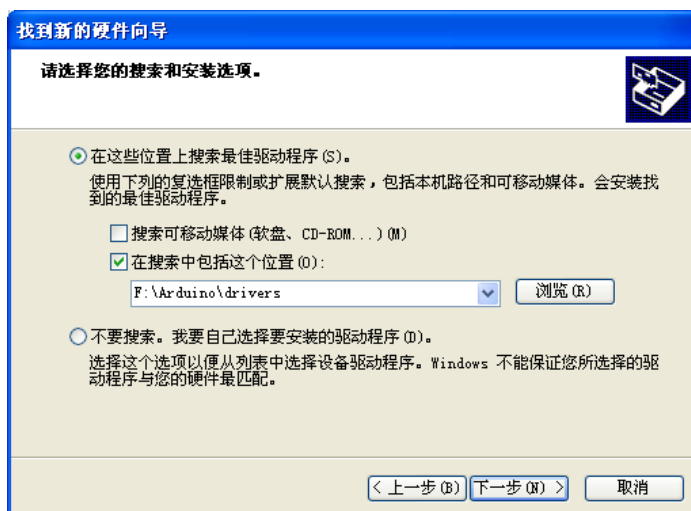
2.3 ARDUINO 驱动安装

2.3.1 ARDUINO 官方原版驱动安装

- 如果您是 XP 或者 Server2003 系统，按照以下教程安装。如果不能安装请解压缩 arduino 安装路径里面的\driver\Old_Arduino_Drivers.zip,然后按照以下教程安装即可。
XP 要用 SP3 以上的系统，SP2 可能会出现串口发送的数据不会显示的问题。
XP 对于 Arduino 官方的开发环境支持很稳定，但是对 Mind+的支持可能不太好。例如可能会出现串口监视器不能用的情况，请使用 arduino 官方开发环境的串口监视器看效果。
- 如果您是 WIN7 32/64BIT 系统，按照以下方法如果安装不上驱动，请重装系统，这个真没有别的好方法，我们对于 win7 也深恶痛绝。Arduino 与 win7 GHOST 版本的兼容性不好，请安装 WIN7 原版。或者更换与 Arduino 兼容性好的 XP 或者 WIN8 系统。
如果因为系统缺失部分文件 32bit 装不上驱动，可以先按照 Arduino Leonardo 资料包\1.开发环境\官方开发环境\win7 装不上驱动办法.zip 的方法解决。
若是 64bit 装不上驱动，也可以参考 32bit 的办法解决。缺失文件也已经放在文件夹下，要注意自己的系统对应补全。
- 如果您是 WIN8 系统，有的会自动安装驱动，如果没有识别出来驱动也请按照以下方法安装。
第一步：使用配套的 USB 线连接电脑和 Arduino 开发板。
此步骤若是 UNO 板子，显示为 Arduino UNO



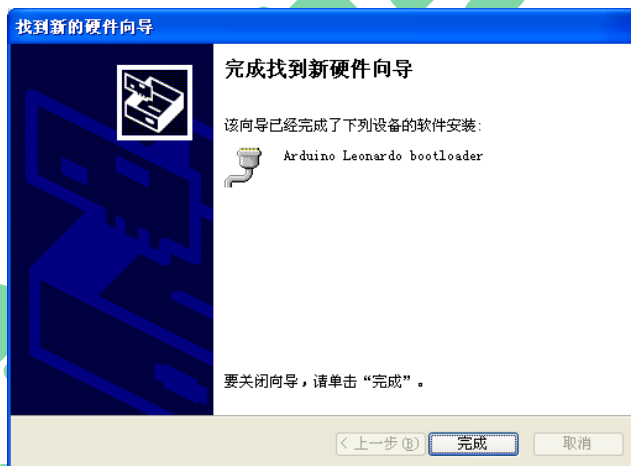
第二步：选择 从列表或指定位置安装（高级），点击 下一步



第三步：选择 在搜索中包括这个位置 ，选择 Arduino 开发环境安装目录下的 drivers

点击 下一步。（只有装过开发环境才有有驱动，亲！）

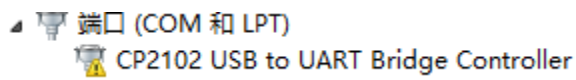
此步骤若为 UNO 板子，显示为 Arduino UNO



第四步：点击 完成 结束安装。

2.3.2 加强版 UNO 驱动安装

第一步：将 ArduinoUNO 增强版板子通过 USB 线与电脑相连。设备管理器里面出现以下界面





第二步：点击右键更新驱动，点击**浏览计算机以查找驱动程序软件**。

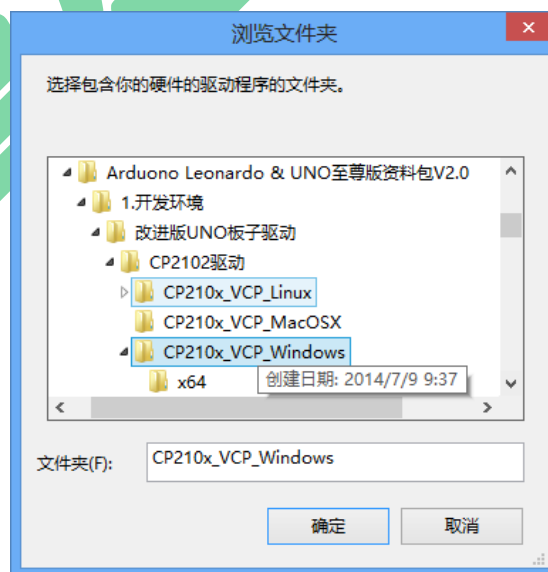


第三步：选择驱动所在文件夹，

WINDOWS 用户选择 WINDOWS,

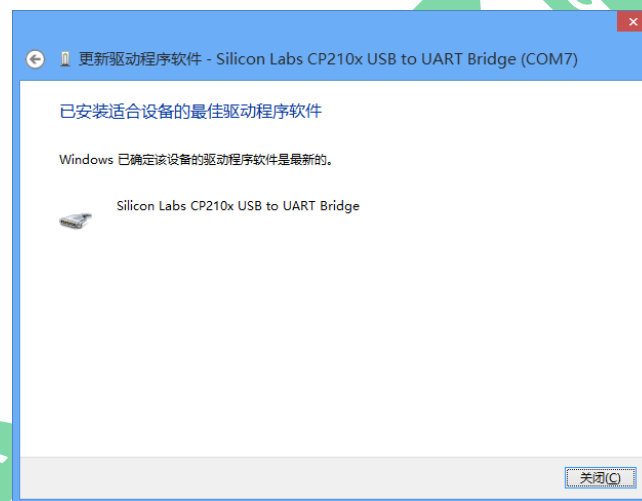
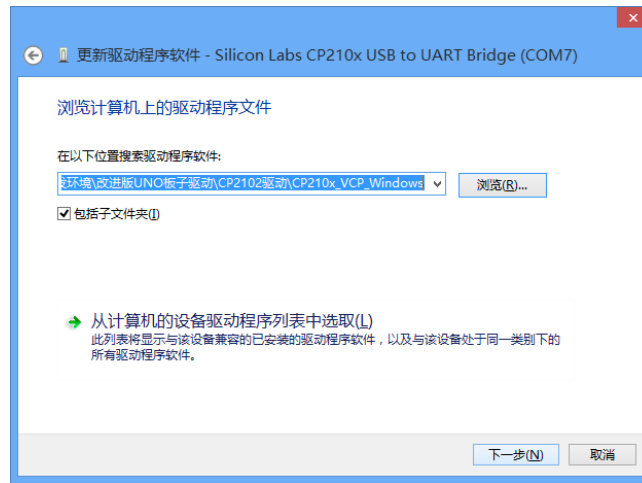
苹果电脑用户选择 MacOSX,

Linux 用户选择 Linux

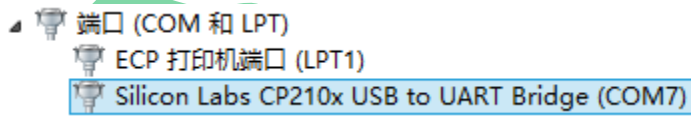




第四步：点击下一步



第五步：看端口里面显示出来 com 号，则安装驱动完成

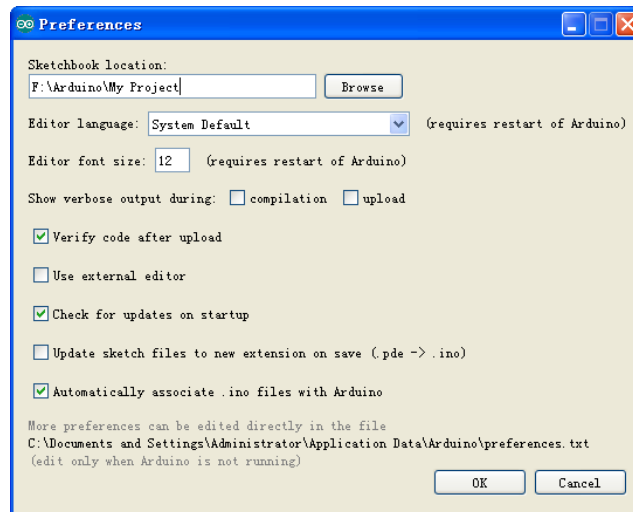


2.4 官方开发环境安装和使用

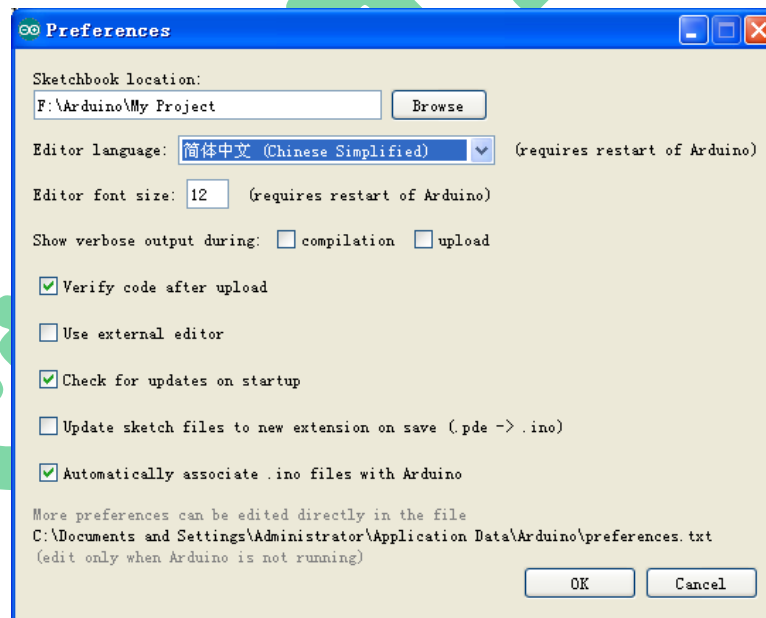
1) 打开桌面上的 Arduino 开发环境快捷方式



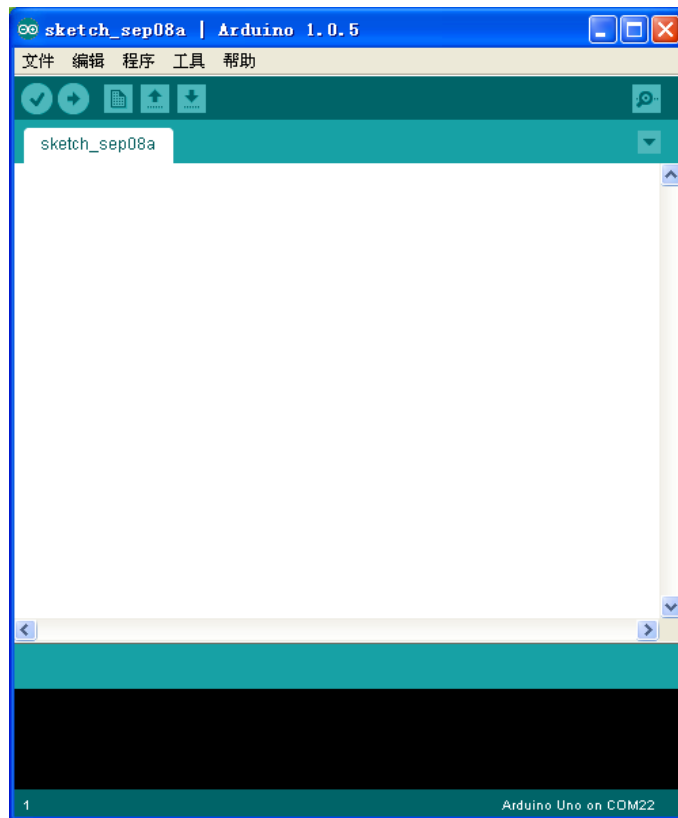
- 2) 选择 file 菜单下的 Preferred



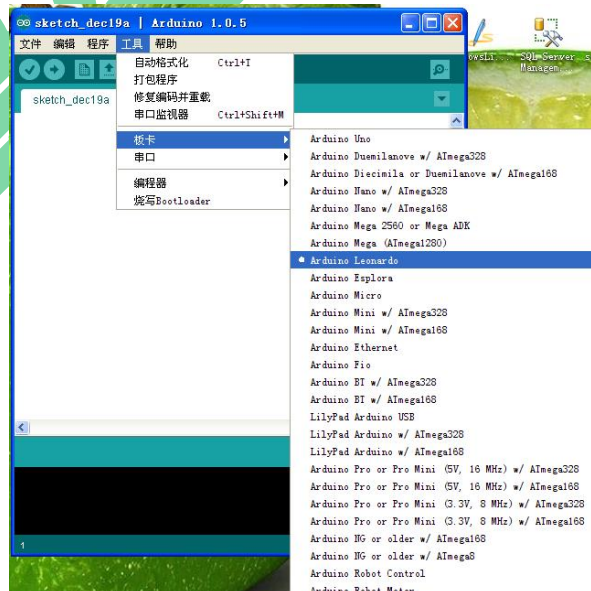
- 3) 选择简体中文开发环境，点 OK



- 4) 关闭开发环境，重新打开，切换到中文界面，是不是舒服多了？(*^__^*) 嘻嘻……

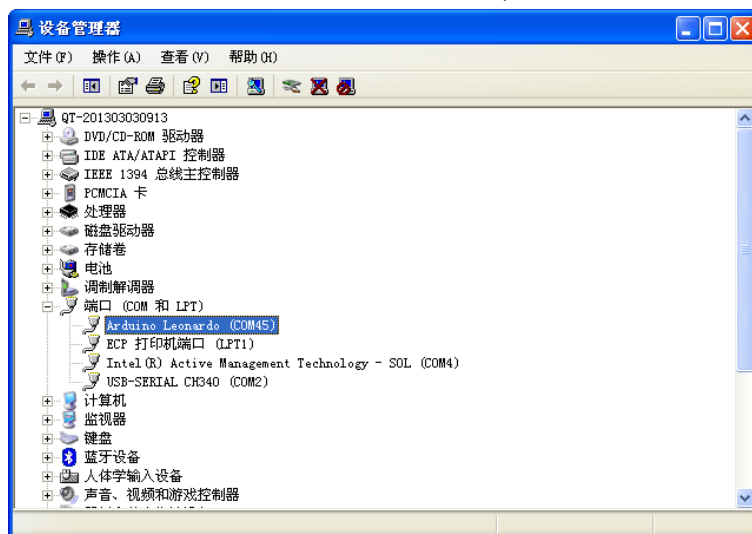


- 5) 在 **工具** 菜单下找到 **板卡**，选择自己的板子，
UNO 板子用户一定要选 **UNO**
ARDUINO Leonardo 用户选择 **Arduino Leonardo**，选错型号会下载不成功

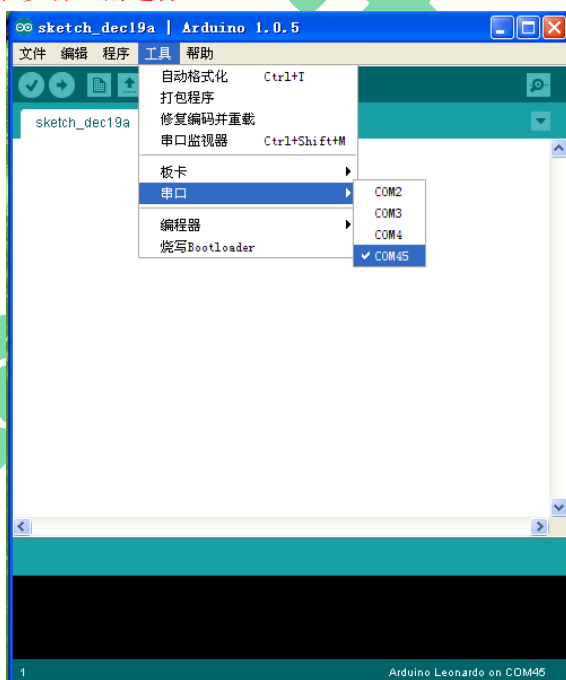




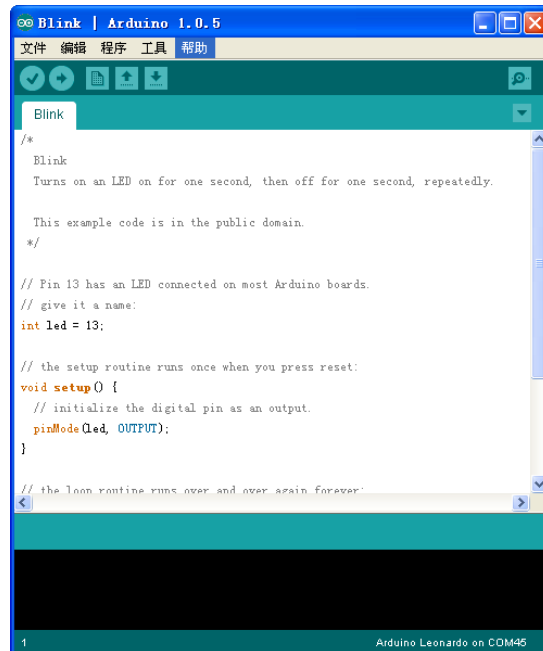
- 6) 选择串口，串口可以在 **我的电脑** 右键 **设备管理器** 查看，



- 7) 我的电脑是 COM45，在 Arduino 开发环境的 **工具** 菜单 **串口** 中选择 **COM45** 右下角出现 **您步骤 5 和步骤 7 的选择。**



- 8) 打开一个例程测试板子是否运转正常
文件 -> 示例 -> 01.Basics -> Blink



- 9) 点击 **编译** 点击 **下载**,
看开发板上的 LED 以 1 秒的频率闪烁。(恭喜您开发板的初步操作已经掌握)
您可以按照同样方法去测试其它的系统自带例程, 或者我们提供的例程了!

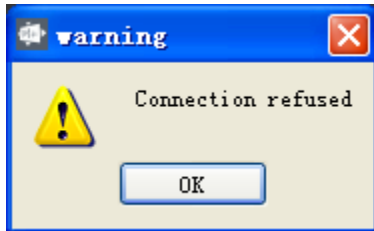


2.5 图形化编程开发环境安装和使用

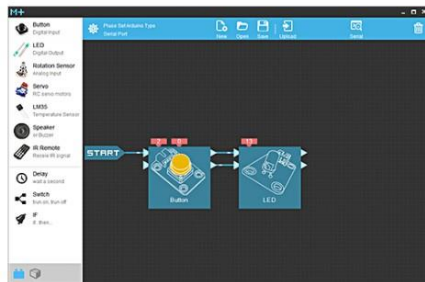
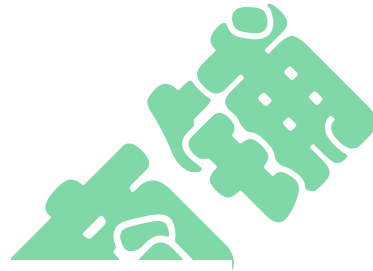


注意:

1. 因 Mind+ 软件较新,所以对 XP 系统的支持不完美,请尽量在 win7 或者 win8 电脑使用。
Xp 系统在安装中可能出现问题,或者出现串口监视器无法使用等问题。
2. 在使用过程中如出现如下警告,无需在意,该软件所有用户都会提示这个警告



- ✓ 图形化编程软件 Mind+,
可以在 [资料包\开发环境](#) 下找到安装包。
或者在官网 www.mindplus.cc 下载最新版本。



Mind+ 是一款对个人免费的Arduino编程软件

编程从未变得如此容易和轻松



不断增加的模块
现有58个丰富功能不同的模块
并且每个版本都会增加新的功能



全图形化界面
只用鼠标就可以完成全部操作



可自行定制的模块
如果你会C++ 你甚至可以自己添加新的模块
完成你想要的任何功能

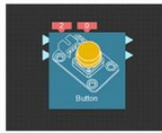


四步学会 Mind+使用

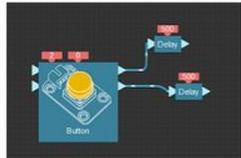
1.把Arduino插在电脑上



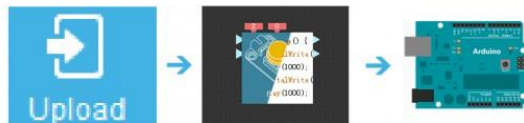
2.拖拽添加模块



3.给模块连线来构成逻辑



4.点击上传 把模块变成代码



向你的朋友展示你的成果 !!!



2.6 面包板连线绘图软件使用

面包板连接图绘制和打开使用 Fritzing。

1. 该软件是绿色版的（免安装），可以在[资料包\5.Arduino 面包板连线绘图软件\frtizing.2013.07.27.pc.zip](#) 找到。
2. 只要双击打开 **fritzing.exe** 一次，系统就会识别这个软件。就能打开绘制的连接图图纸。



..	文件夹	2013-7-27 2...	
bins	文件夹	2013-7-27 2...	
help	文件夹	2013-7-27 2...	
lib	文件夹	2013-7-27 2...	
parts	文件夹	2013-7-27 2...	
pdb	文件夹	2013-7-27 2...	
sketches	文件夹	2013-7-27 2...	
translations	文件夹	2013-7-27 2...	
fritzing.exe	应用程序	2013-7-27 2...	C2F609E5
LICENSE.CC-BY-SA	CC-BY-SA 文件	2011-2-17 1...	A1470D45
LICENSE.GPL2	GPL2 文件	2008-11-12 ...	F6BF552A
LICENSE.GPL3	GPL3 文件	2008-11-12 ...	80AF5E21
msvcpi100.dll	应用程序扩展	2012-10-4 1...	38F12A25
msvcpi100.dll	应用程序扩展	2012-10-4 1...	DB505E33
QtCore4.dll	应用程序扩展	2012-10-22 ...	7EA53B4F
QtGui4.dll	应用程序扩展	2012-10-22 ...	02430C34
QtNetwork4.dll	应用程序扩展	2012-10-22 ...	4A944E6C
QtSql4.dll	应用程序扩展	2012-10-22 ...	9028ED0E
QtSvg4.dll	应用程序扩展	2012-10-22 ...	77E9086F
QtXml4.dll	应用程序扩展	2012-10-22 ...	D9700645
README.txt	文本文档	2012-9-15 0...	222DD0C5

3. 用该软件可以打开 资料包\6.例程连线图原理图源文件 下的所有图纸。

2.7 关于例程和视频教程注意事项

本视频教程适合官方原版 Arduino UNO 板子、本店开发的 Arduino UNO 加强版，还有官方原版 Leonardo。但三块板子使用某些例程中存在差异。

其中 Leonardo 板子可以实现所有的例程。

UNO 的板子不支持例程 24,25.(原版 UNO 和增强版 UNO 在使用角度没区别)。

另外 UNO 和 Leonardo 板子在某些例程使用不太一样，视频中会着重讲解。若例程没区别，则选取其中一块板子讲解，不再使用 3 块板子演示。

3 实验例程

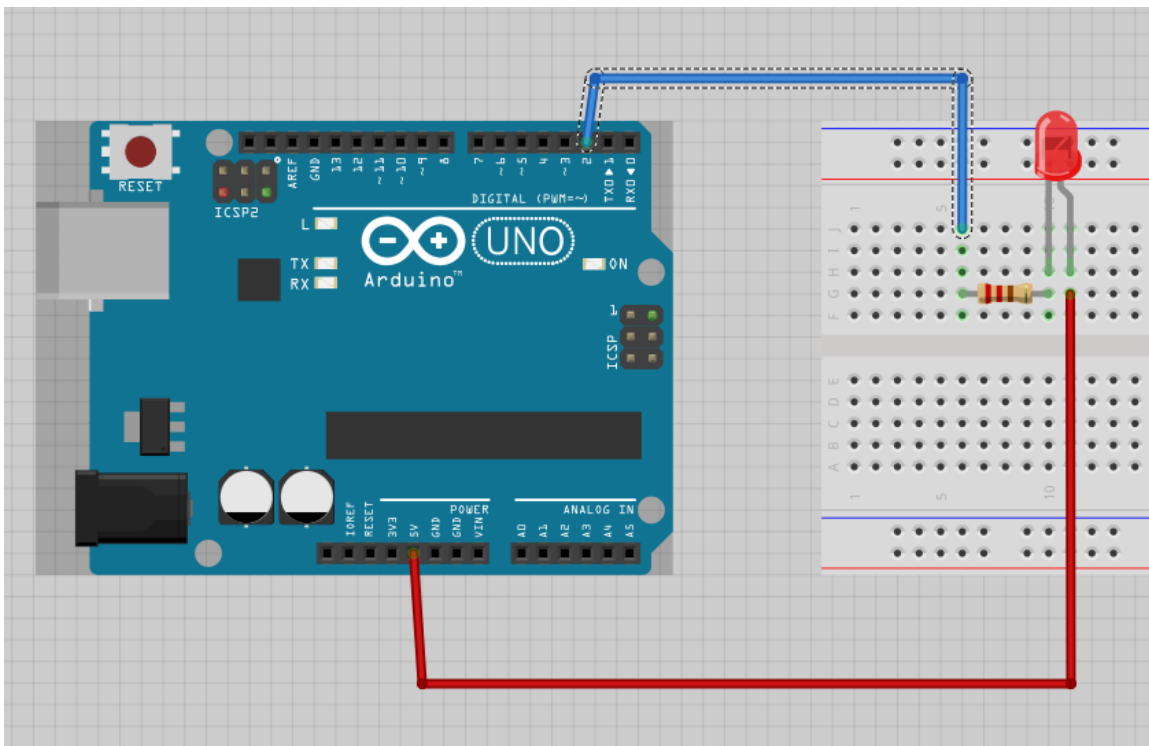
3.1 LED 闪烁实验

实验现象：LED 1 秒钟闪烁。

理论学习：

实验用到的新语句：

- ✓ `#define led 2` //用于设置板子上的 2 引脚的名字为 led
- ✓ `pinMode(led, OUTPUT);` //用于设置 LED 引脚为输出引脚
- ✓ `digitalWrite(led, HIGH);` //设置 LED 引脚输出高电平，点亮 LED
- ✓ `digitalWrite(led, LOW);` //设置 LED 引脚输出低电平，熄灭 LED



连接图

3.2 HELLO WORLD 实验

做为程序员编写所有程序的第一课，Hello World！是必须的一个环节，这一讲我们讲解一下如何使用 Arduino 的串口编写一句“Hello World！”，然后用 Arduino 发送给 PC 机。

理论学习：

该程序中用到 Arduino 程序里面最常见的几个函数：

- ✓ `void setup()` 该函数用于编写 Arduino 的初始化内容，本例程内设置通讯波特率就放在这里，对于通讯的设置只许设置一次就可以了，所以把 `Serial.begin(9600)` 放在这里。
- ✓ `Serial.begin(9600)` 用于设置串口通讯的波特率，这里设置为 9600
- ✓ `void loop()` 该函数是 Arduino 的主程序部分，编写 Arduino 的核心代码。这里通过串口向电脑发送一个字符串，字符串内容为“Hello World！”
- ✓ `Serial.println("Hello World!")` 用于填写需要发送的字符串的内容。
- ✓ `delay(1000);` 这是毫秒延时函数，延时 1000 毫秒，即 1 秒

3.3 按键控制 LED 亮灭



实验现象：本实验通过一个按键来控制一个发光二极管的亮灭。按键按一下 LED 点亮，再按一下 LED 熄灭。

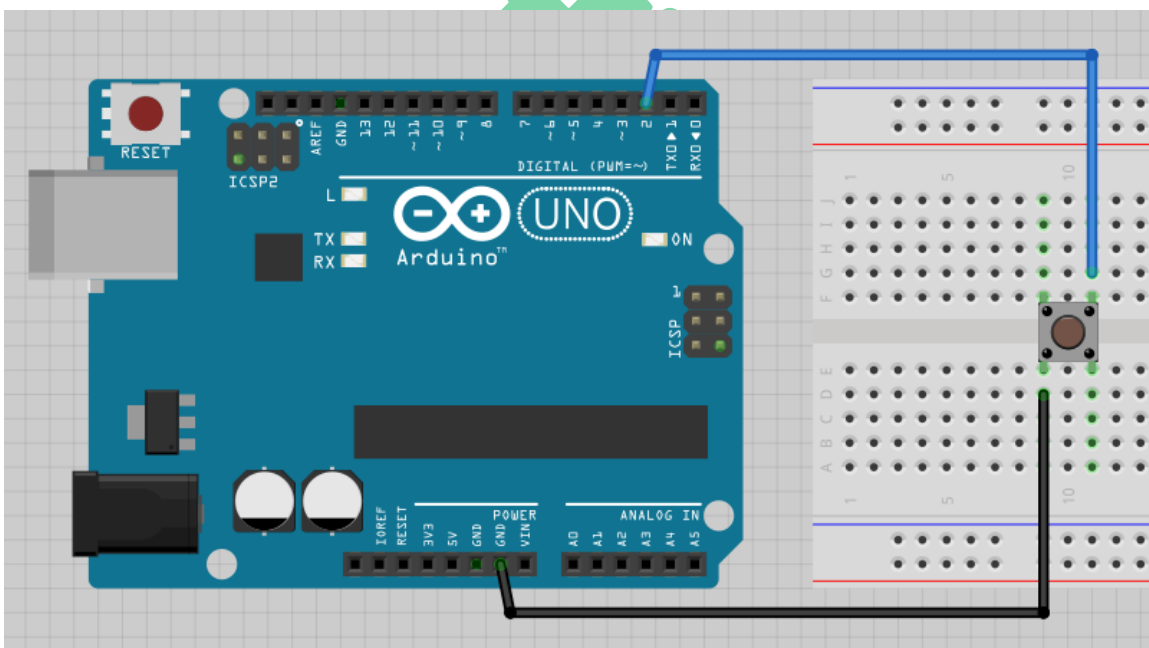
理论学习：

该实验主要难点：

- 如何扫描按键的状态？
- 如何防抖动？

原理讲解：

- ✓ **按键检测原理：**通过把 Arduino 的数字 IO 设置为输入状态来监控按键是否按下，当按键未按下时候因为有上拉电阻存在，读到的电平为 HIGH，当按键按下时候因为按键引脚接地，所以读回来的电平为 LOW，由此判断按键是否按下。
- ✓ **按键去抖动原理：**因为人手的机械动作使按键按下时候会产生大概 20ms 左右的按键抖动，如果 Arduino 在这 20ms 内去检测 IO 口的电平很可能会检测出来不稳定的信号。因此，在检测到低电平后延迟 20ms 再次检测可以起到软件去抖动的作用。程序中用到的 `delay(20);` 就是这个作用
- ✓ **松手检测原理：**所谓的松手检测原理更简单，就是程序无限循环等待按键松开，即让程序不停的去检测 IO 状态，如果恢复了 HIGH 就跳出循环。程序中这样写的 `while(digitalRead(KEY) == 0);`



连接图

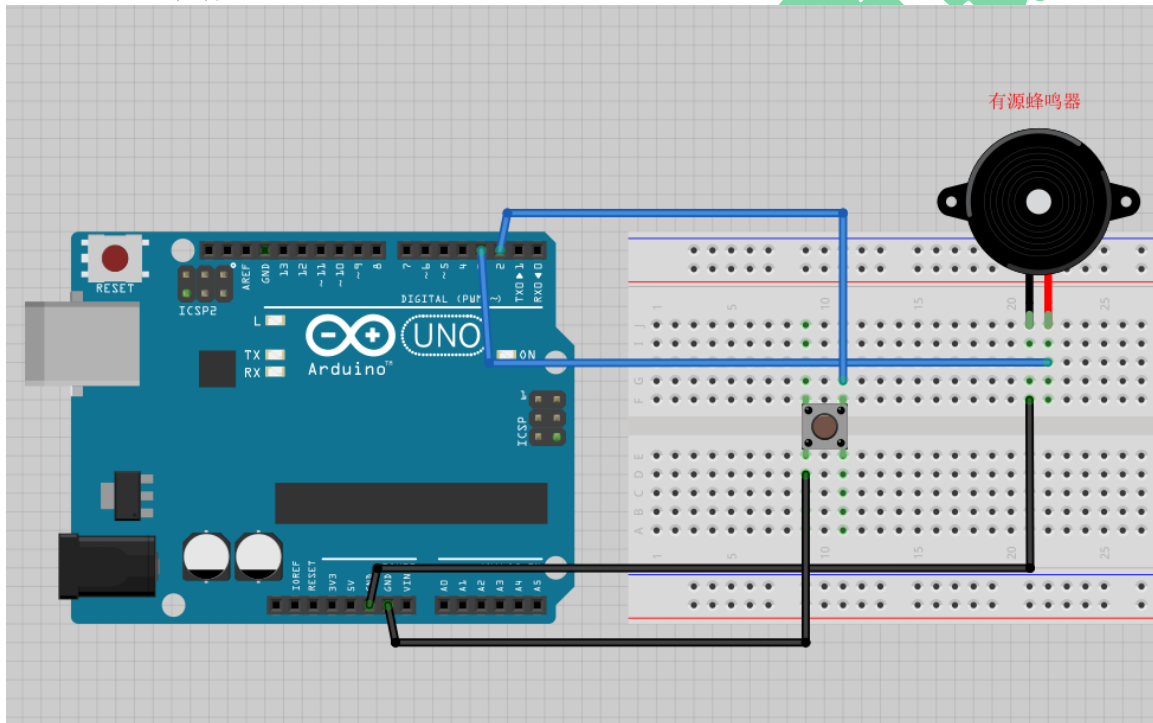
3.4 有源蜂鸣器实验

实验现象：每按一次按键，蜂鸣器滴一声

备注：如果想让有源蜂鸣器声音大一点，可以撕开蜂鸣器上的贴纸。

理论学习：

- ✓ 有源蜂鸣器与无源蜂鸣器的区别：
注意：这里的“源”不是指电源，而是指震荡源。
也就是说，有源蜂鸣器内部带震荡源，所以只要一通电就会叫。
而无源内部不带震荡源，所以如果用直流信号无法令其鸣叫。必须用 2K-5K 的方波去驱动它。



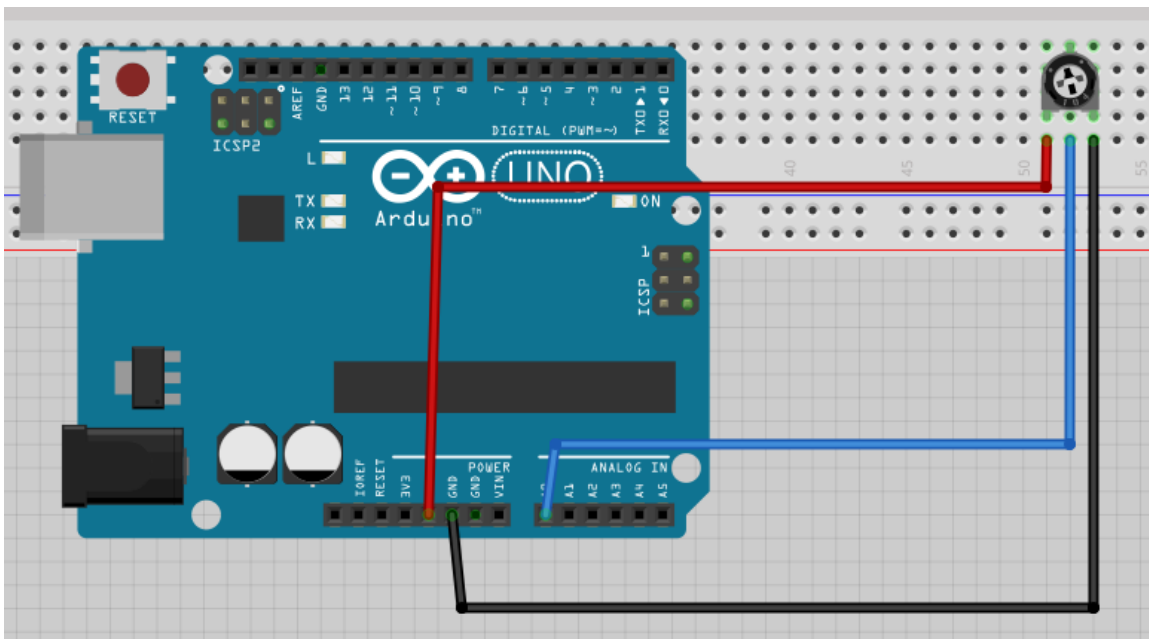
连接图

3.5 电位器实验

实验现象：调节电位器，串口发送电位器调节值。

理论学习：

- 本实验用到的电位器为 10K，即 PIN1 和 PIN3 两个引脚之间的电压为 10K，PIN2 和任意两个引脚之间的电阻是通过调节电位器旋钮改变的。
- 本实验主要学习模拟 IO 的读取。



连接图

3.6 PWM 调光

实验现象：电位器控制 LED 的亮度。

理论学习：

- ✓ 学会 PWM 的使用：**Pulse Width Modulation** 脉冲宽度调制，简称脉宽调制。是利用微处理器的数字输出来对模拟电路进行控制的一种非常有效的技术，广泛应用在从测量、通信到功率控制与变换的许多领域中。

脉冲宽度调制（PWM）是一种对模拟信号电平进行数字编码的方法，由于计算机不能输出模拟电压，而只能输出 0V 或 5V 的数字电压值，（0V 为 0；5V 为 1）所以通过高分辨率计数器，利用方波的占空比被调制的方法对一个具体模拟信号的电平进行编码。但 PWM 信号仍然是数字的，因为在给定的任意时刻，直流供电要么是 5V（数字值为 1），要么是 0V（数字值为 0）。电压或电流源以一种通(ON)、断(OFF)的重复脉冲序列加到模拟负载上，只要带宽足够，任何模拟值都可以使用 PWM 进行编码。

- ✓ 输出的电压值是通过通和断的时间进行计算的，计算公式为：

输出电压 = (接通时间 / 脉冲时间) * 最大电压值

- ✓ PWM 的三个基本参数：
 - 1、脉冲宽度变化幅度（最小值/最大值）
 - 2、脉冲周期（1 秒内脉冲频率个数的倒数）
 - 3、电压高度（例如：0V-5V）

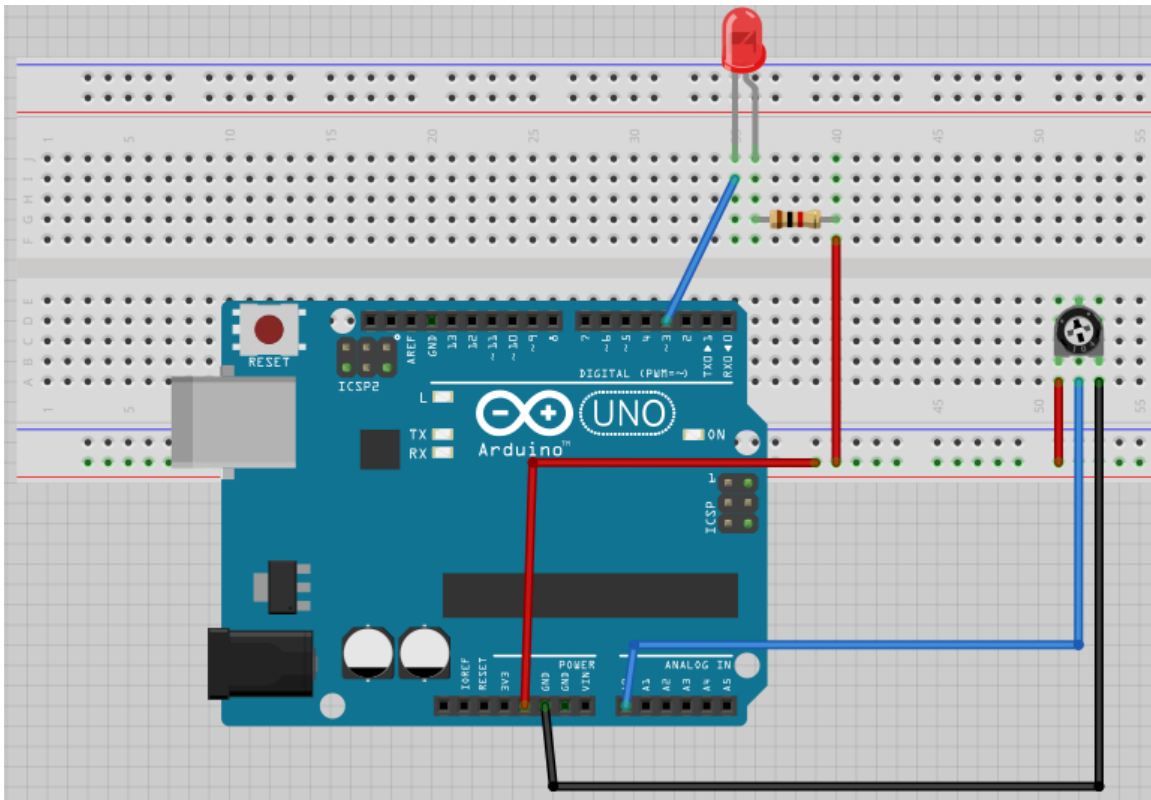
Arduino UNO 控制器上有 6 个 PWM 接口分别是数字接口 3、5、6、9、10、11

- ✓ Arduino 中的设置 PWM 的语句：

```
analogWrite (pin,value) ;
```

// pin: 用于输入数值的引脚。

//value: 占空比: 0 (完全关闭) 到 255 (完全打开) 之间。



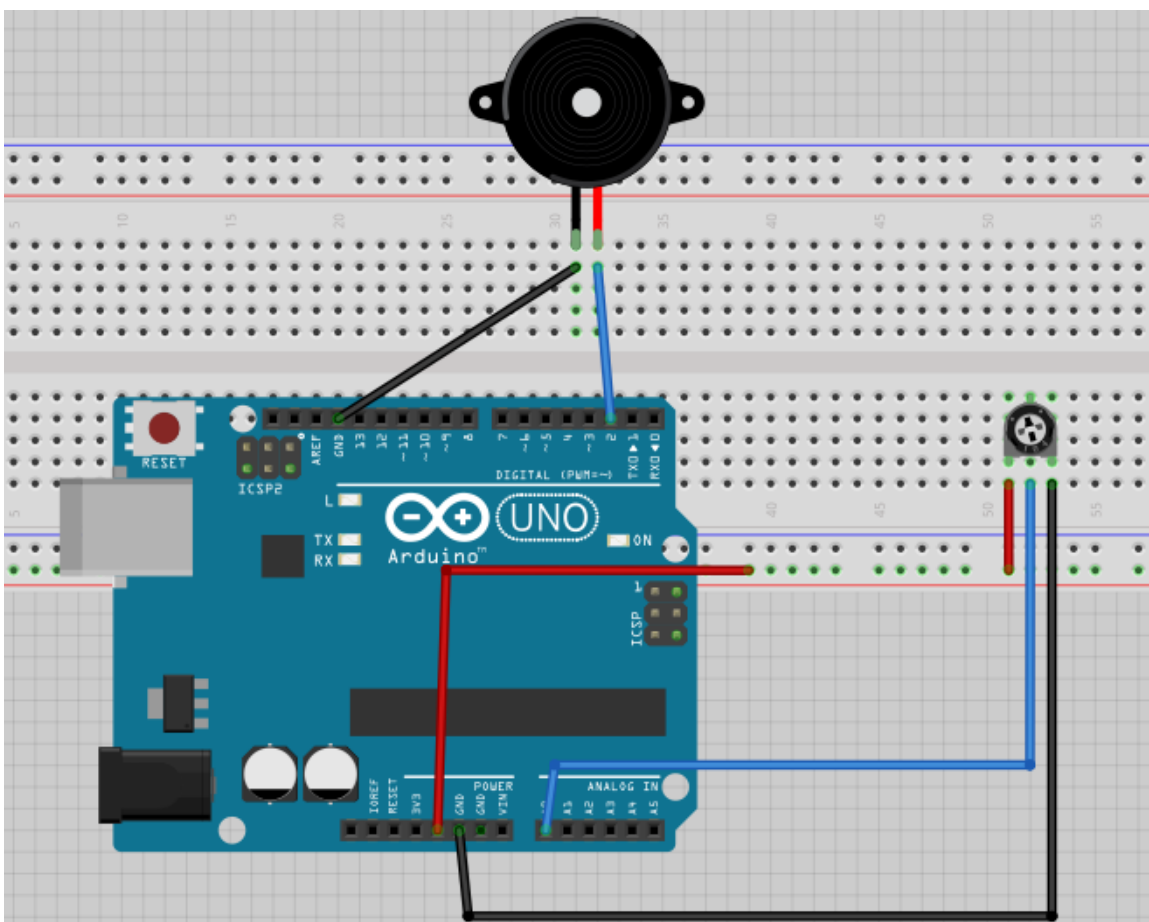
连接图

3.7 无源蜂鸣器实验（在家玩 DJ）

实验现象：用电位器调节无源蜂鸣器发出不同声音。

理论学习：

无源蜂鸣器和实际的音响的信号是一样的，是靠频率驱动的，学会无源蜂鸣器可以直接把音响信号接到自己家的音箱上。在家自己做 DJ 不是梦哦~~



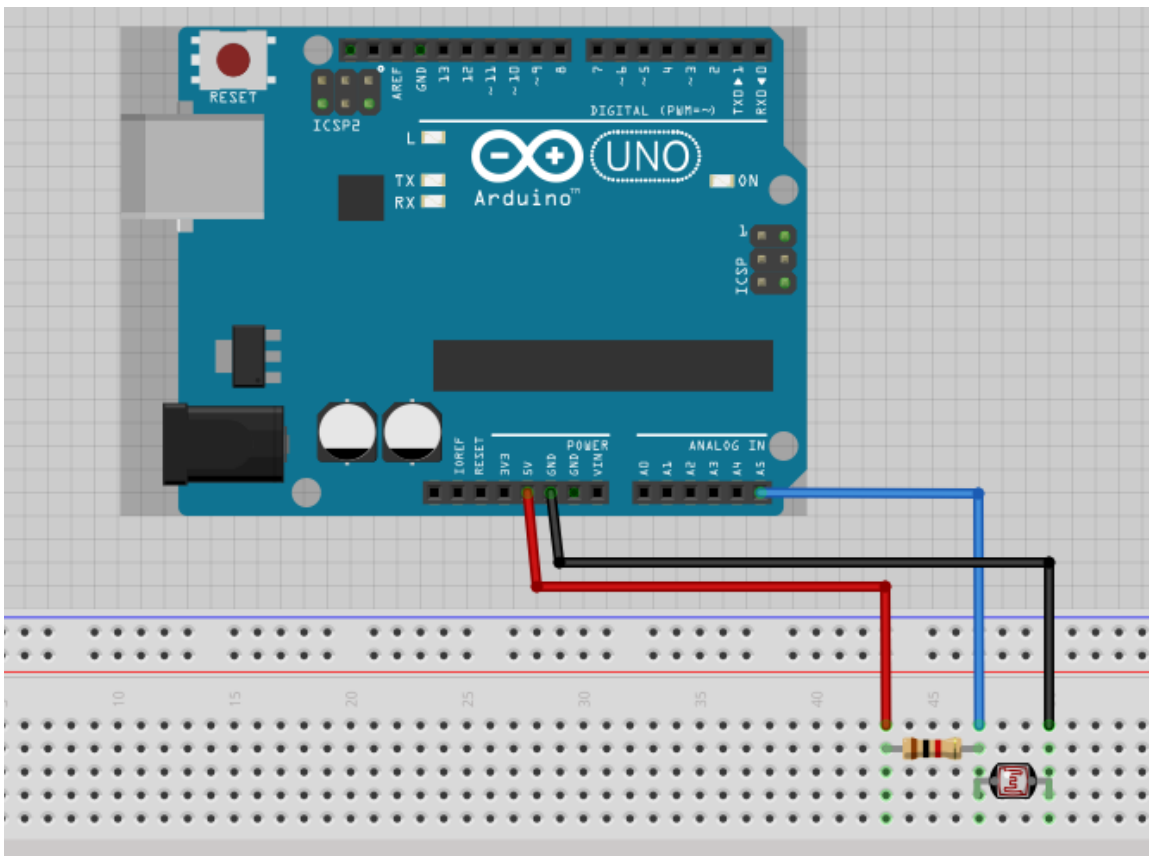
连接图

3.8 光控 LED 实验（天黑请开灯）

实验现象：当室内光线充足时候 LED 关闭，当室内光线变暗时候点亮 LED。

理论学习：

- ✓ 光敏电阻的特性是光敏电阻的阻值随着光照强度的变大而减小。换句话说，光线越亮电阻越小，光线越暗电阻越大。
- ✓ 本实验选用 1K 电阻和光敏电阻串联，根据电阻分压原理，光线越暗，光敏电阻两端的电压越大。
- ✓ 本实验通过 A5 引脚检测光敏电阻两端的电压值来检测光强。



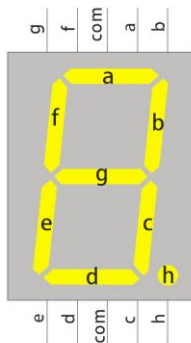
连接图

3.9 一位共阴数码管实验

实验现象：数码管依次显示 0-9。

理论学习：

- ✓ 数码管是一种半导体发光器件，其基本单元是发光二极管。按发光二极管单元连接方式可分为共阳极数码管和共阴极数码管。共阳数码管是指将所有发光二极管的阳极接到一起形成公共阳极(COM)的数码管，共阳数码管在应用时应将公共极 COM 接到+5V，当某一字段发光二极管的阴极为低电平时，相应字段就点亮，当某一字段的阴极为高电平时，相应字段就不亮。共阴数码管是指将所有发光二极管的阴极接到一起形成公共阴极(COM)的数码管，共阴数码管在应用时应将公共极 COM 接到地线 GND 上，当某一字段发光二极管的阳极为高电平时，相应字段就点亮，当某一字段的阳极为低电平时，相应字段就不亮
- ✓ 套餐内给用户配备有一个一位共阴数码管。
- ✓ 共阴数码管的引脚示意图：



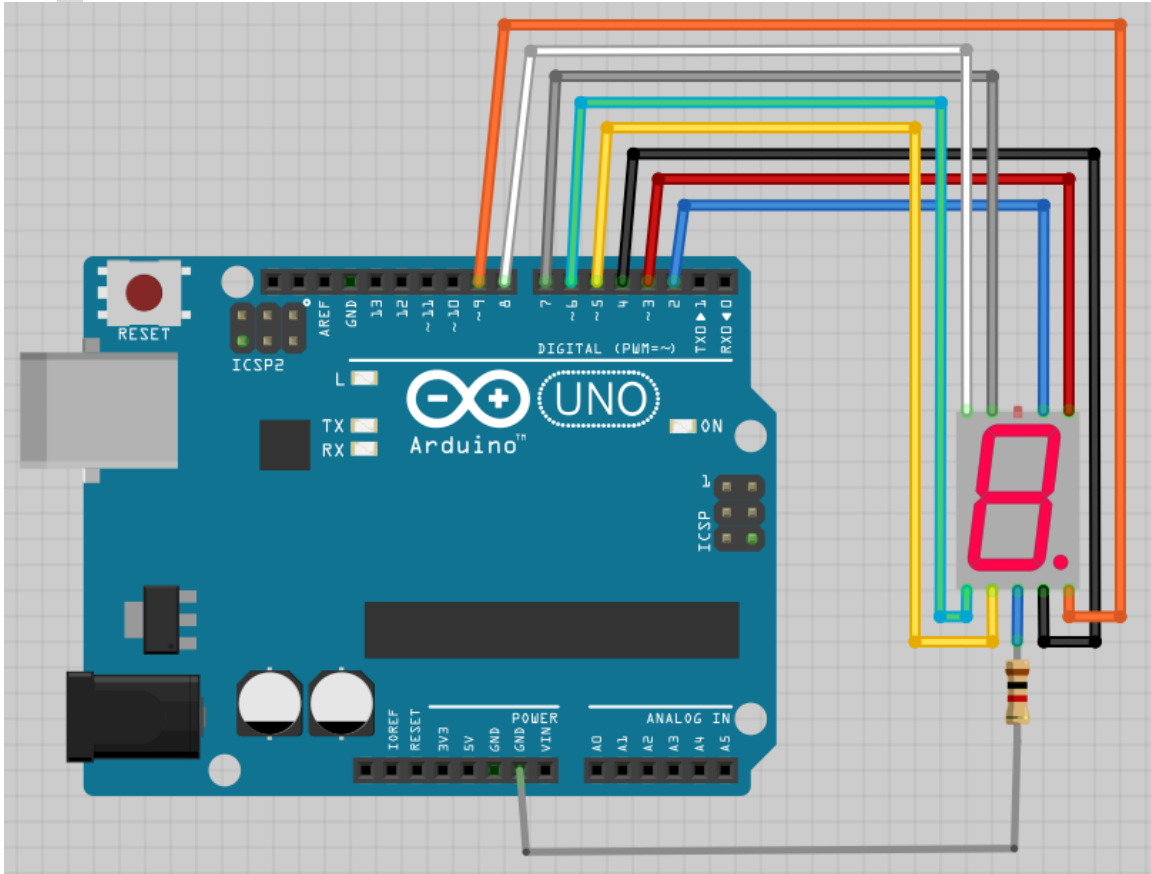
- ✓ 显示原理：如果想点亮 a 段的 LED，需要 a 引脚提供高电平，com 口提供低电平。建议在 COM 引脚串联一个电阻用于分压限流。
- ✓ 共阴数码管显示 0-9 时候的码值：

	h	g	f	e	d	c	b	a
0	0	0	1	1	1	1	1	1
1	0	0	0	0	0	1	1	0
2	0	1	0	1	1	0	1	1
3	0	1	0	0	1	1	1	1
4	0	1	1	0	0	1	1	0
5	0	1	1	0	1	1	0	1
6	0	1	1	1	1	1	0	1
7	0	0	0	0	0	1	1	1
8	0	1	1	1	1	1	1	1
9	0	1	1	0	1	1	1	1

将以上存放在数组中

```
unsigned char table[10][8] =
{
  {0, 0, 1, 1, 1, 1, 1, 1}, //0
  {0, 0, 0, 0, 0, 1, 1, 0}, //1
  {0, 1, 0, 1, 1, 0, 1, 1}, //2
  {0, 1, 0, 0, 1, 1, 1, 1}, //3
  {0, 1, 1, 0, 0, 1, 1, 0}, //4
  {0, 1, 1, 0, 1, 1, 0, 1}, //5
  {0, 1, 1, 1, 1, 1, 0, 1}, //6
  {0, 0, 0, 0, 0, 1, 1, 1}, //7
}
```

```
{0, 1, 1, 1, 1, 1, 1, 1}, //8
{0, 1, 1, 0, 1, 1, 1, 1} //9
};
```



连接图

3.10 四位共阴数码管实验

实验现象：数码管 1-4 位分别显示 1、2、3、4。

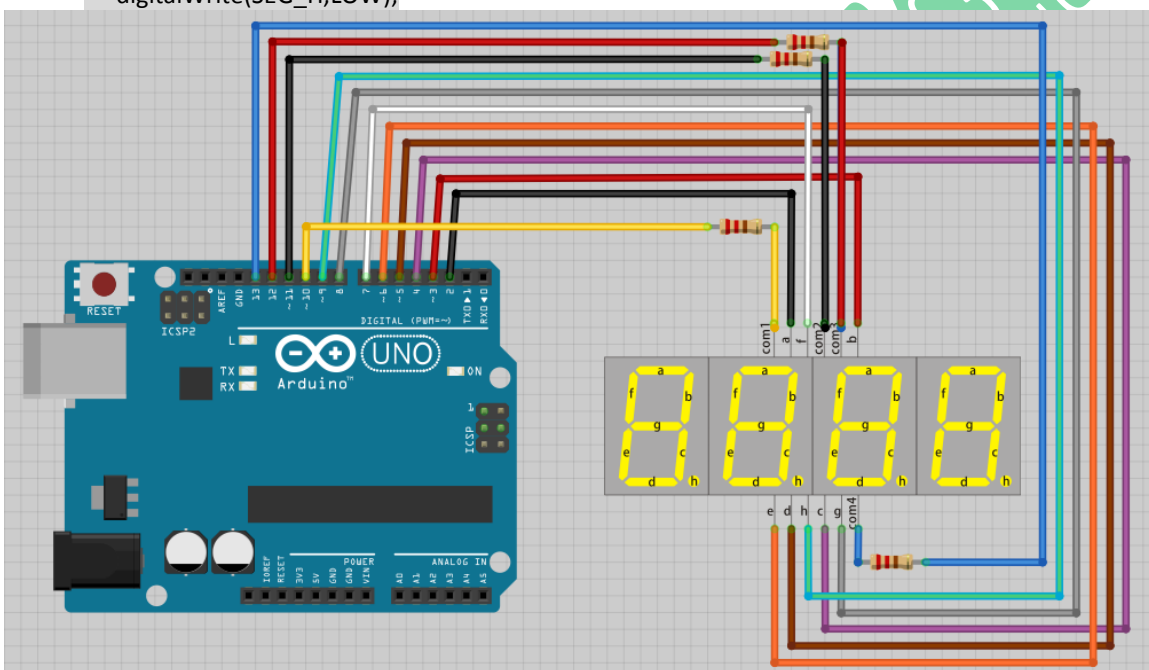
理论学习：

- ✓ 数码管动态扫描：动态显示的特点是将所有位数码管的段选线并联在一起，由位选线控制是哪一位数码管有效。这样一来，就没有必要每一位数码管配一个锁存器，从而大大地简化了硬件电路。选亮数码管采用动态扫描显示。所谓动态扫描显示即轮流向各位数码管送出字形码和相应的位选，利用发光管的余辉和人眼视觉暂留作用，使人的感觉好像各位数码管同时都在显示。

通俗点说：动态扫描就是一位一位显示数码管的速度加快，速度达到几个 ms 以内。

- ✓ 动态扫描时候去除余晖(显示模糊): 在动态扫描时候切换位选时候先把段选的输出清零, 然后再切换位选, 这样可以有效去除余晖。程序中 `void Display(unsigned char com, unsigned char num)` 函数中最前面的程序即为去除余晖的代码:

```
digitalWrite(SEG_A,LOW); //去除余晖
digitalWrite(SEG_B,LOW);
digitalWrite(SEG_C,LOW);
digitalWrite(SEG_D,LOW);
digitalWrite(SEG_E,LOW);
digitalWrite(SEG_F,LOW);
digitalWrite(SEG_G,LOW);
digitalWrite(SEG_H,LOW);
```

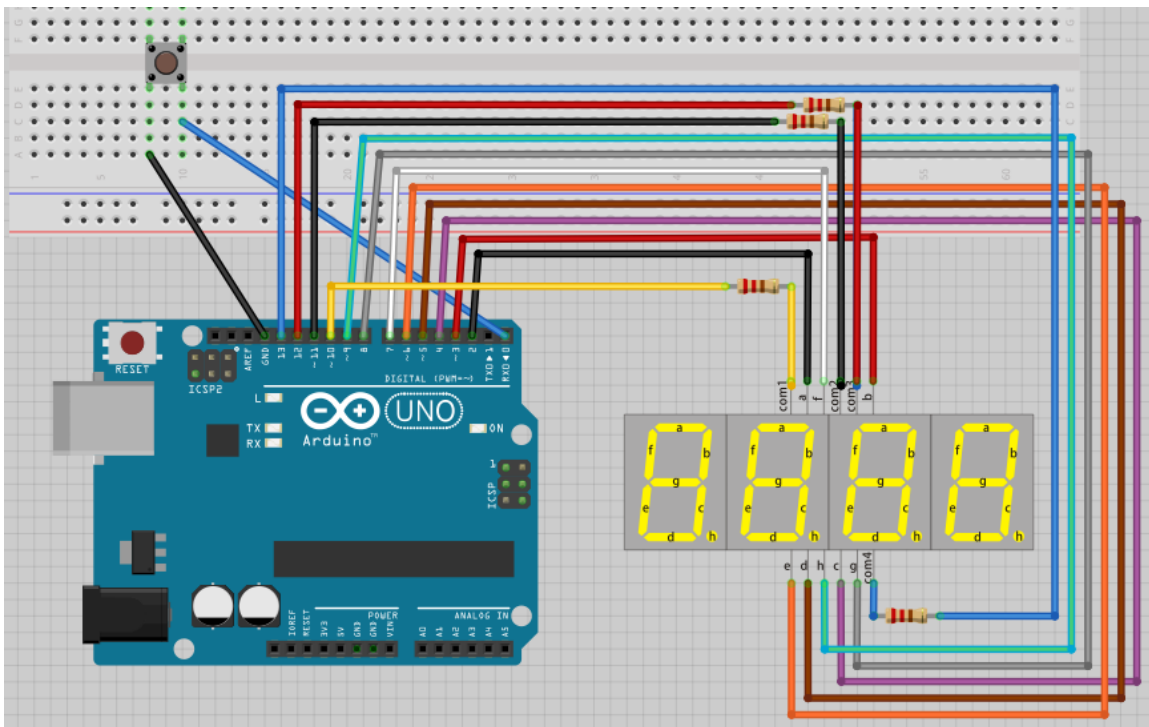


连接图

3.11 累加器实验

实验现象: 每当按下一下按键, 数码管显示值加一。

理论学习: 熟悉四位数码管显示原理, 回顾以前实验中按键的写法。编写出需要的实验效果。



连接图

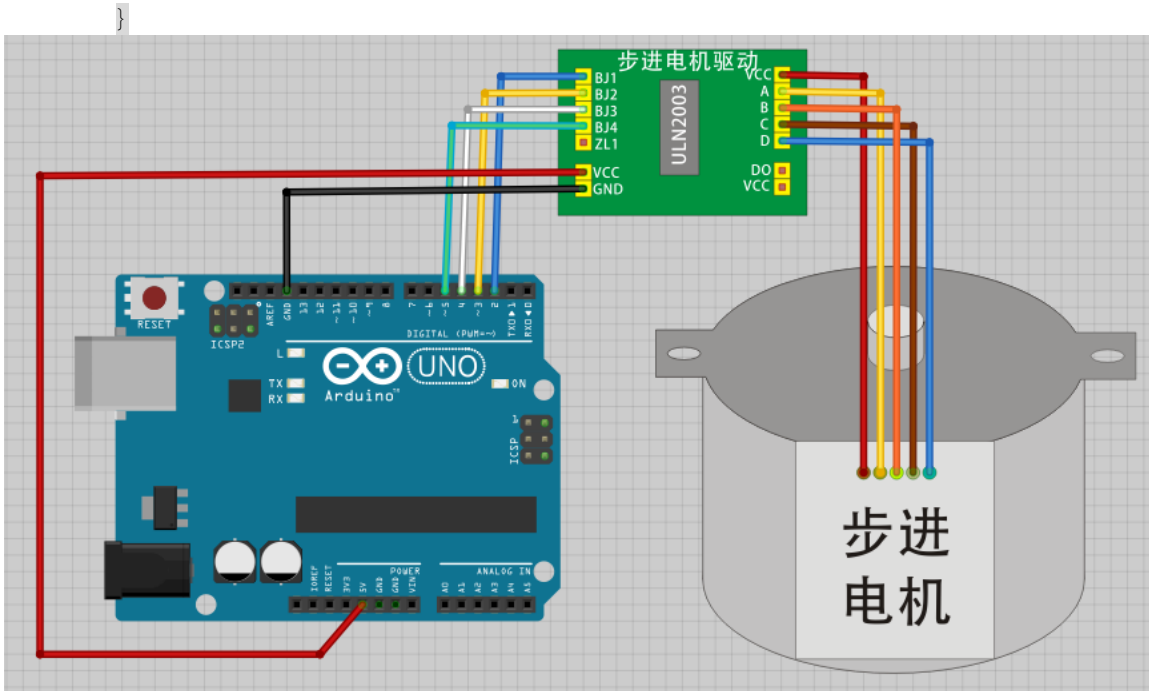
3.12 步进电机驱动实验

实验现象：步进电机旋转。

理论学习：

- ✓ 28BYJ-48 步进电机空载耗电在 50mA 以下，带 64 倍减速器，输出力矩比较大，可以驱动重负载，极适合开发板使用。注意：此款步进电机带有 64 倍减速器，与不带减速器的步进电机相比，转速显得较慢，为方便观察，可在输出轴处粘上一片小纸板。其中红色线为 VCC，其余 4 个为 4 个相位。
- ✓ 使用 ULN2003 达林顿驱动芯片驱动步进电机，板载 4 个 LED，可以指示相位状态。
- ✓ 步进电机相位控制，如果选择相位 A，单片机给驱动板 BJ1 引脚高电平，其它 BJ2/BJ3/BJ4 引脚低电平，达林顿管对应引脚输出电平会翻转，达林顿管输出低电平时配合 VCC，可以驱动 A 相位。代码示例：

```
void Phase_A()
{
    digitalWrite(A1,HIGH); //A1引脚高电平
    digitalWrite(B1,LOW);
    digitalWrite(C1,LOW);
    digitalWrite(D1,LOW);
}
```



连接图

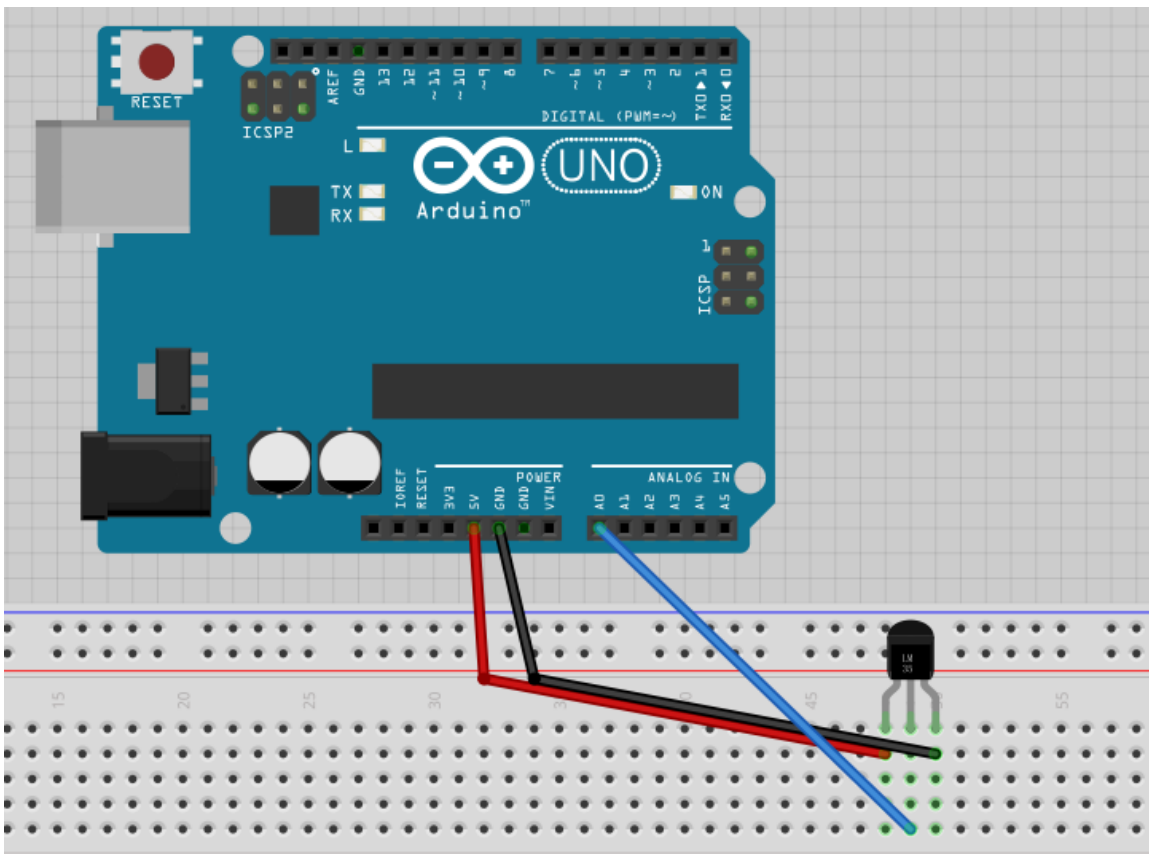
3.13 温度传感器 LM35 实验

实验现象：通过 LM35 测量温度值，通过串口发送给电脑。

理论学习：LM35 使用非常普遍，他使用内部补偿机制，输出可以从 0℃ 开始。封装为 T0992，工作电压 4—30V。而且在上述电压范围内，芯片的工作电流不超过 60ua。根据产品使用手册，得知 LM35 传感器的输出电压与摄氏温度呈线性关系，0℃ 时输出为 0V，每升高 1℃，输出电压增加 10mv。

计算过程：

1. 假设 AD 读取出来的值为 val，那么该 AD 值对应的电压为： $val * 5 / 1023 * 1000 = 4.8876 * val$ ，单位 mV。
2. 对应的温度为 $4.8876 * val / 10 = 0.48876 * val$ ，单位为℃。



连接图

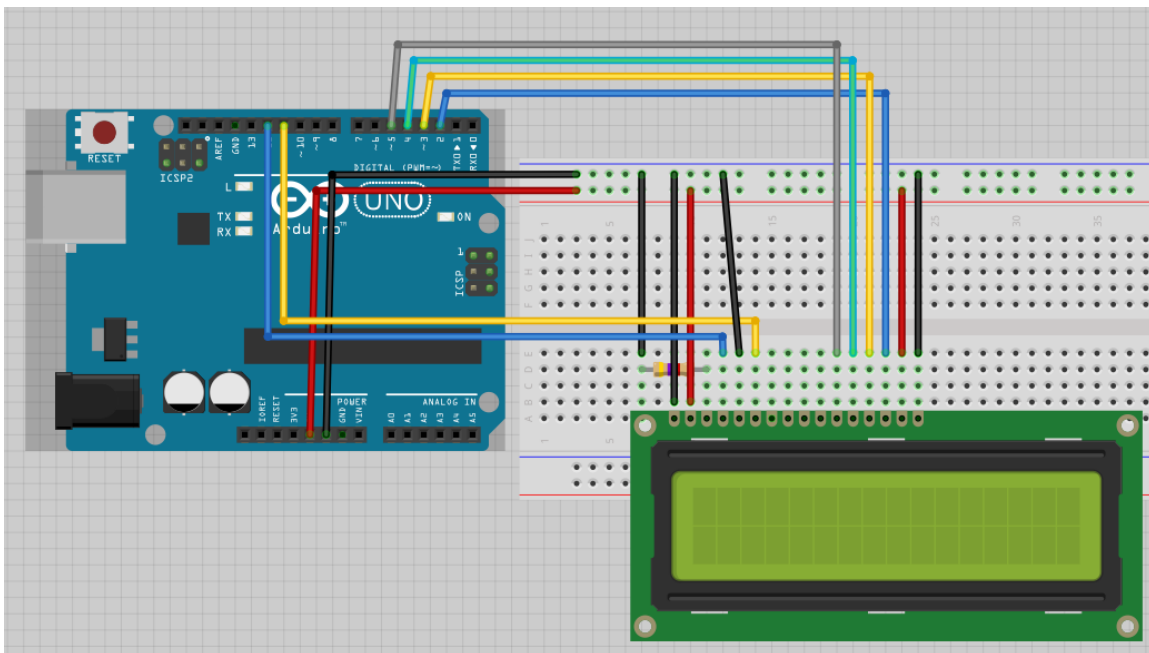
3.14 液晶 LCD1602 实验

实验现象:

- ✓ 液晶开机画面显示 Welcome to use!
- ✓ 读取 LM35 温度传感器的值，显示在液晶上。

理论学习:

- ✓ LCD1602 液晶是一款通用简易液晶，可以显示 ASCII 码的英文字母、数字和标点符号（不能显示中文）。
- ✓ LCD1602 总共可以显示 2 行*16 个字符。



连接图

3.15 滚珠开关实验

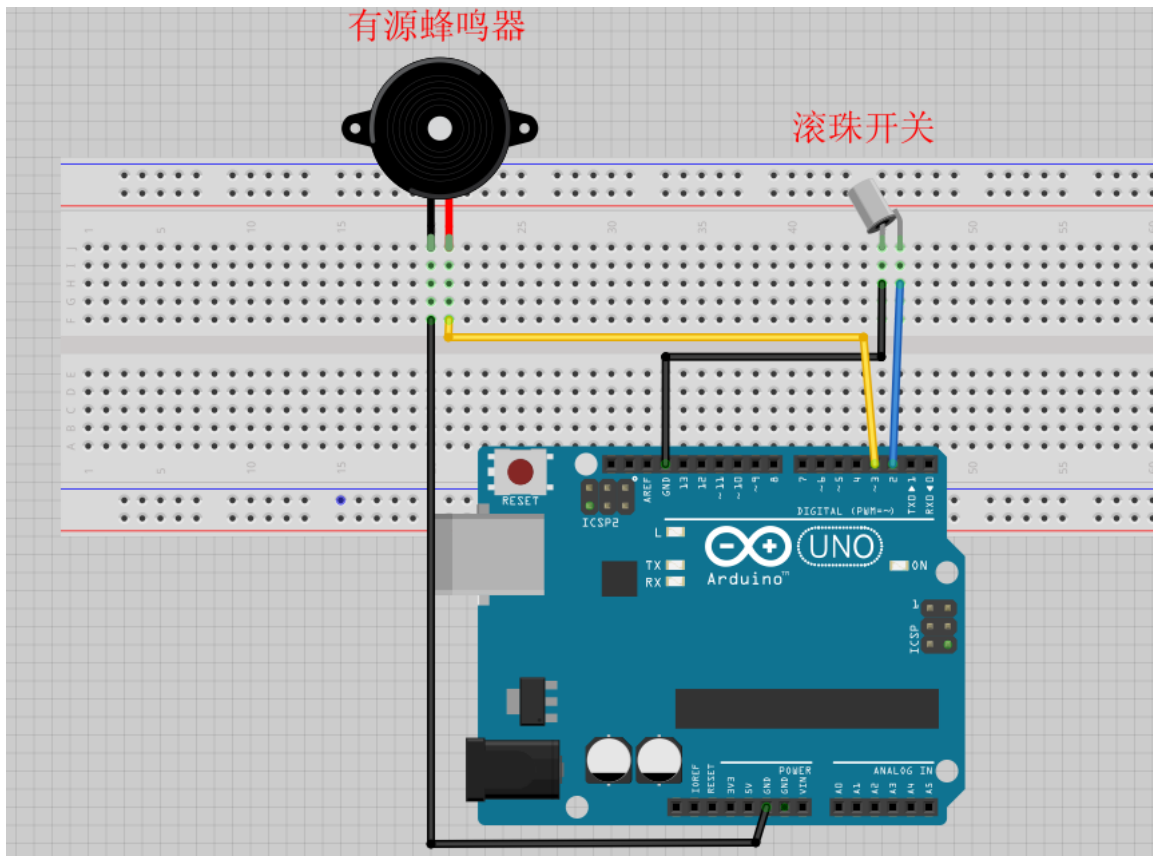
实验现象:

利用滚珠开关特性，编写一个电动车防盗报警设备。

理论学习:

滚阻开关, 当有振动发生时候, 滚珠开关的两个引脚会瞬间接通。Arduino 通过中断方式获取到该信号, 进行蜂鸣器报警。

熟练掌握 Arduino 的中断编写方式: `attachInterrupt(interrupt,function,mode);`



连接图

外部中断在不同的Arduino型号上位置也不同，只有外部中断发生在以下型号的外部中断引脚标号。

型号	int.0	int.1	int.2	int.3	int.4	int.5
UNO\Ethernet	2	3				
Mega2560	2	3	21	20	19	18
Leonardo	3	2	0	1		
Due	所有IO口均可					

不同型号的 **arduino** 的中断位于数字 IO 的引脚不同，请参照上图编写程序：

举例说明：数字 IO 的 2 引脚，UNO 板子是外部中断 0，而 Leonardo 则是外部中断 1。

对应本程序中以下两条语句

```
attachInterrupt(0,BuzzerDi,FALLING);
//沿触发中断函数 BuzzerDi,
```

```
//设置 ARDUINO UNO 中断 0（数字 IO 2），下降
```

```
attachInterrupt(1,BuzzerDi,FALLING);
//这和 UNO 有所不同，请知悉！
```

```
//设置 ARDUINO Leonardo 中断 1（数字 IO 2），
```

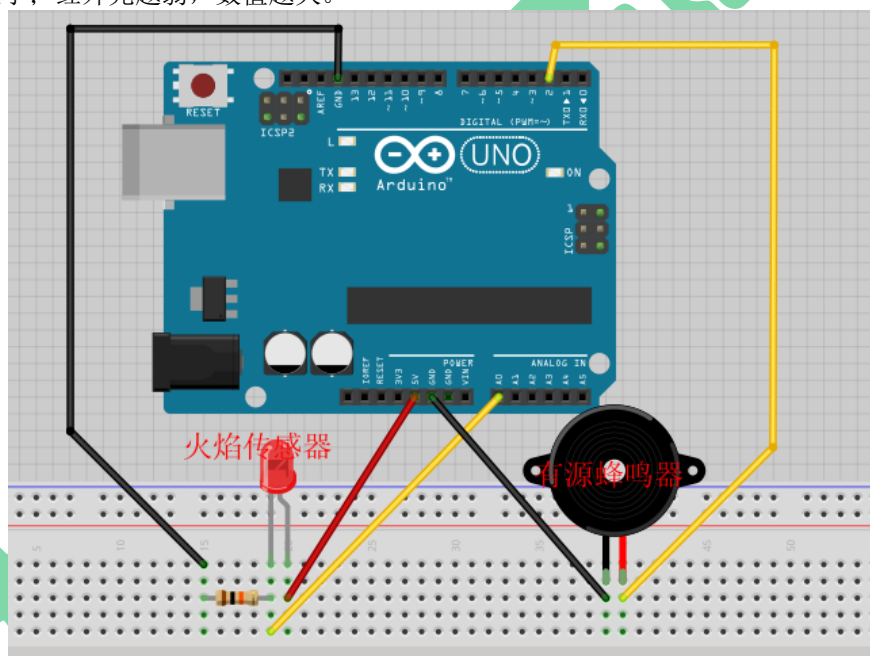
3.16 火焰传感器实验

实验现象:

检测到有火源报警。

理论学习:

- ✓ 远红外火焰传感器可以用来探测火源或其它一些波长在 700 纳米~1000 纳米范围内的热源。在机器人比赛中，远红外火焰探头起着非常重要的作用，它可以用作机器人的眼睛来寻找火源或足球。利用它可以制作灭火机器人、足球机器人等。
- ✓ 远红外火焰传感器能够探测到波长在 700 纳米~1000 纳米范围内的红外光，探测角度为 60°，其中红外光波长在 880 纳米附近时，其灵敏度达到最大。远红外火焰探头将外界红外光的强弱变化转化为电流的变化，通过 A/D 转换器反映为 0~1023 范围内数值的变化。外界红外光越强，数值越小；红外光越弱，数值越大。



连接图

3.17 三基色 LED 七彩跳变

实现现象: 本实验使用 UNO 驱动一颗 RGB 三基色 LED 产生七色光的变化。

备注: 引脚最长的为共阳的正极, 剩下 3 个分别为红色、绿色、蓝色。

理论学习: 三基色组合七色光原理:

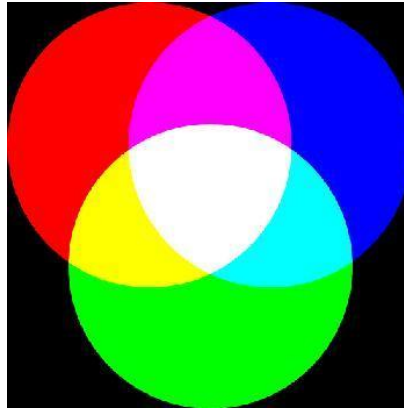


红色+绿色=黄色

绿色+蓝色=青色

红色+蓝色=紫色

红色+绿色+蓝色=白色



总共由 3 种基色“红绿蓝”组合出来七色光“红绿蓝青紫黄白”。

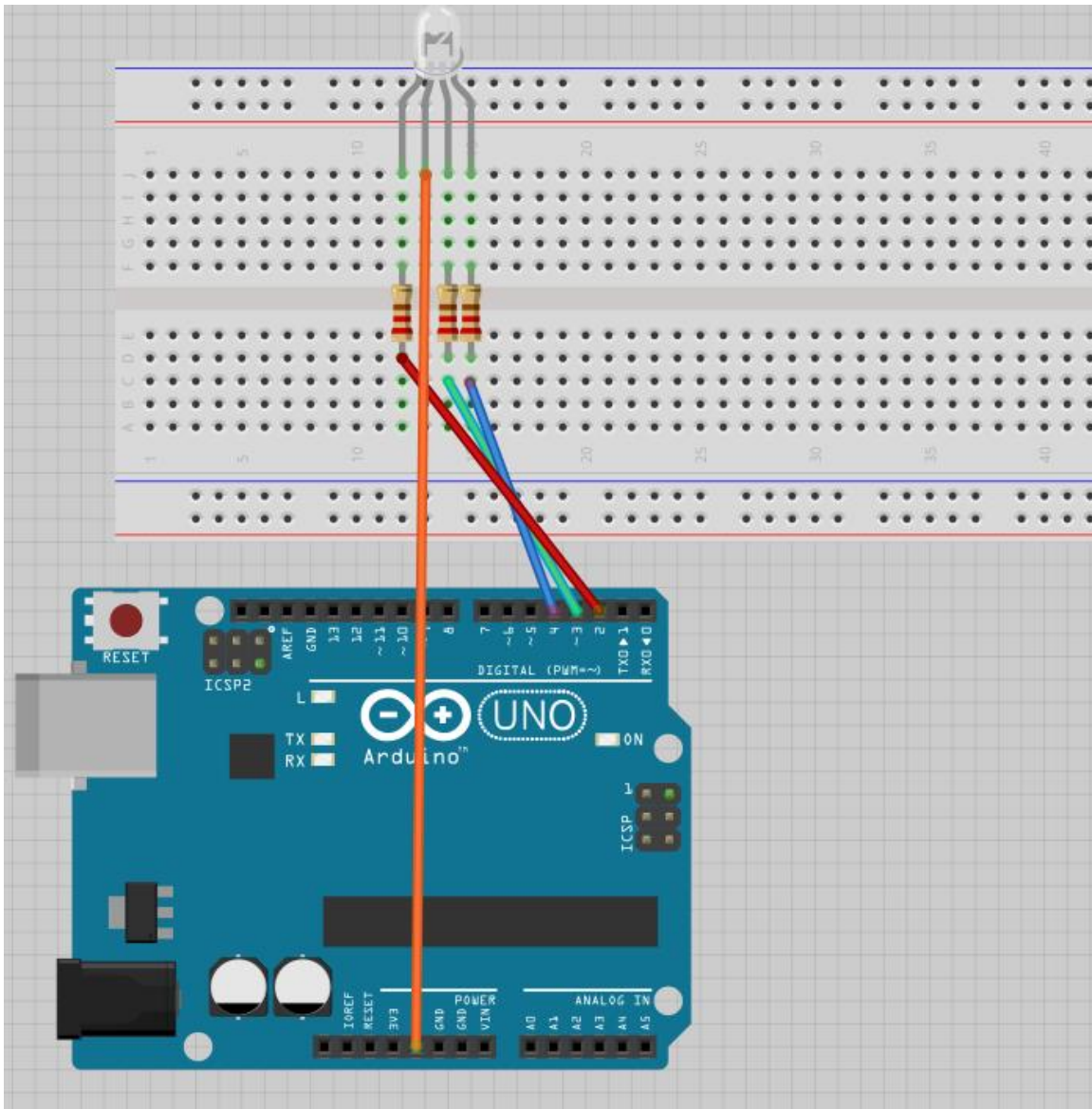
大家明白了三基色的原理就很容易的去编写这个例程了。我们想让哪个颜色 LED 点亮就把对应控制引脚的 IO 设置为 LOW 就可以了。（因为这个 LED 是共阳极的哦!）

试验中使用 switch 语句来选择颜色

```
enum{Color_R,Color_G,Color_B,Color_RG,Color_RB,Color_GB,Color_RGB}; //枚举所有颜色
//红, 绿, 蓝, 黄, 紫, 青, 白
void Change_Color(unsigned char data_color) //选择颜色函数
{
    switch(data_color)
    {
        case Color_R: //红色
            digitalWrite(LED_R,LOW);
            digitalWrite(LED_G,HIGH);
            digitalWrite(LED_B,HIGH);
            break;
        case Color_G: //绿色
            digitalWrite(LED_R,HIGH);
            digitalWrite(LED_G,LOW);
            digitalWrite(LED_B,HIGH);
            break;
        case Color_B: //蓝色
            digitalWrite(LED_R,HIGH);
            digitalWrite(LED_G,HIGH);
            digitalWrite(LED_B,LOW);
            break;
        case Color_RG: //黄色
            digitalWrite(LED_R,LOW);
            digitalWrite(LED_G,LOW);
            digitalWrite(LED_B,HIGH);
            break;
        case Color_RB: //紫色
            digitalWrite(LED_R,LOW);
            digitalWrite(LED_G,HIGH);
            digitalWrite(LED_B,LOW);
            break;
        case Color_GB: //青色
            digitalWrite(LED_R,HIGH);
            digitalWrite(LED_G,LOW);
            digitalWrite(LED_B,LOW);
            break;
        case Color_RGB: //白色
            digitalWrite(LED_R,LOW);
            digitalWrite(LED_G,LOW);
            digitalWrite(LED_B,LOW);
            break;
    }
}
```



```
        break;
    case Color_RB:                                //紫色
        digitalWrite(LED_R,LOW);
        digitalWrite(LED_G,HIGH);
        digitalWrite(LED_B,LOW);
        break;
    case Color_GB:                                //青色
        digitalWrite(LED_R,HIGH);
        digitalWrite(LED_G,LOW);
        digitalWrite(LED_B,LOW);
        break;
    case Color_RGB:                               //白色
        digitalWrite(LED_R,LOW);
        digitalWrite(LED_G,LOW);
        digitalWrite(LED_B,LOW);
        break;
    default:
        break;
}
```



连接图

3.18 通过按键切换 LED 颜色

实验现象: 按键每次按下 LED 切换一种颜色, 依次为红, 绿, 蓝, 黄, 紫, 青, 白;

理论学习:

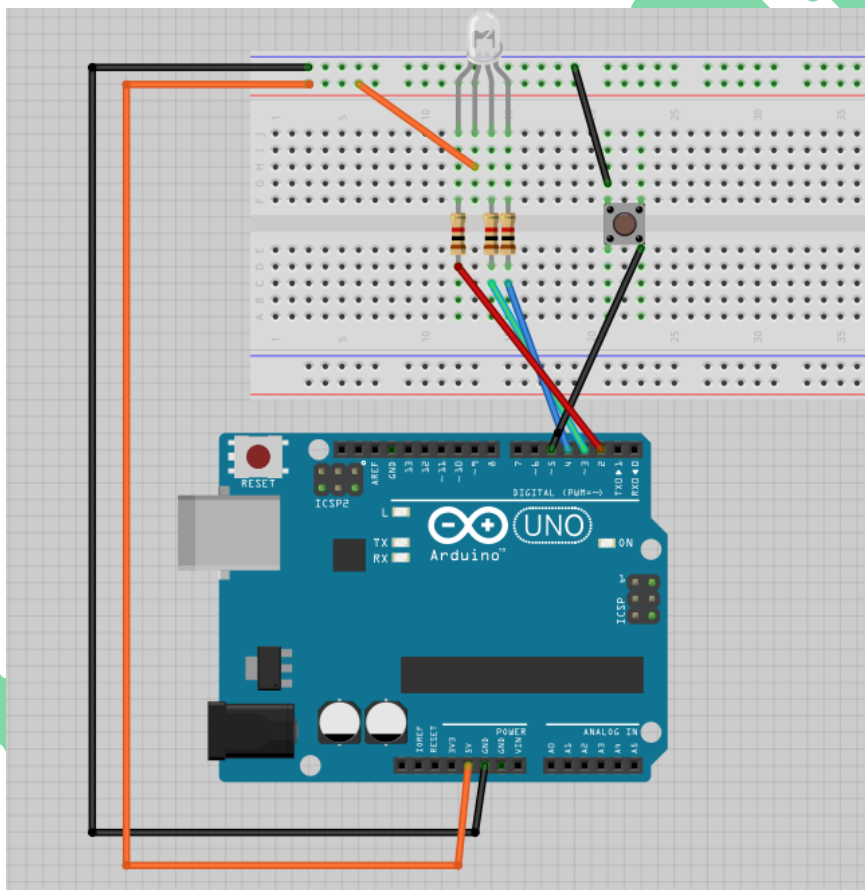
- ✓ 学会枚举语句的应用
- ```
enum{Color_R,Color_G,Color_B,Color_RG,Color_RB,Color_GB,Color_RGB};
```

相当于  $\text{Color\_R} = 0, \text{Color\_G} = 1, \dots, \text{Color\_RGB} = 6$ , 使用枚举语句使程序更易懂, 阅读起来更简单舒服。

这样在选择颜色时候, 例如选择白色  $\text{Change\_Color}(\text{Color\_RGB})$ ; 就可以了, 如果写  $\text{Change\_Color}(6)$  当然也能看懂, 但是需要花费很大的精力。

- ✓ 因为要 0-6 这 7 种颜色循环切换, 因此当切换到 白色 时候要跳回 红色。程序中使用以下语句切换:

```
Color_Value++; //颜色变量+1
if(Color_Value == Color_RGB+1) //当颜色变量 == 7
{
 Color_Value = Color_R; //颜色变量清零, 切换为红色
}
```



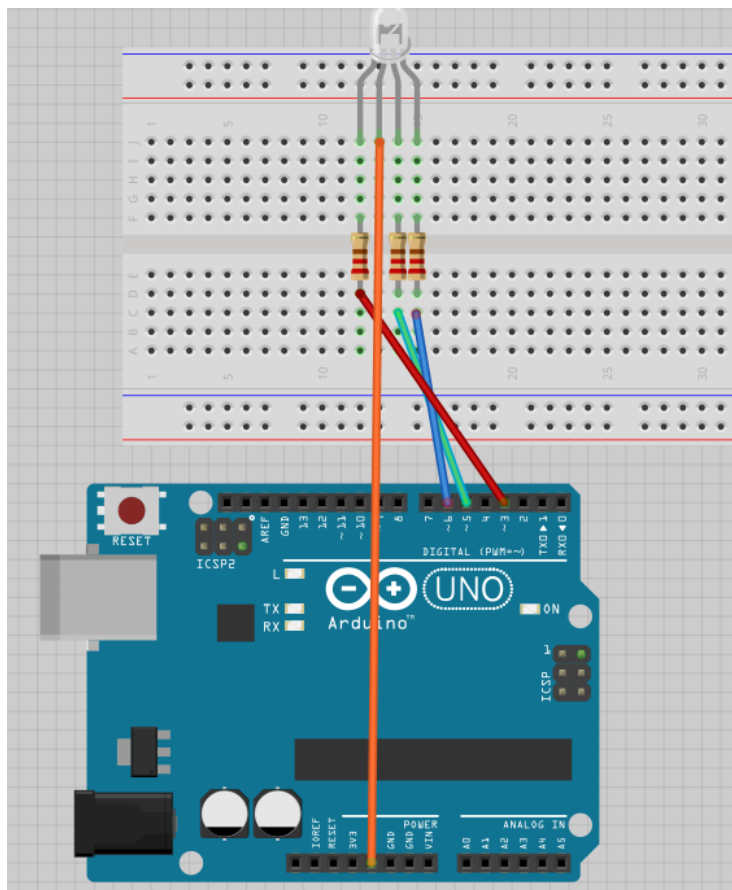
连接图

### 3.19 三彩渐变实验

**实验现象:** 控制三基色 LED 中的红绿蓝三种颜色分别慢慢变亮, 再慢慢变暗。



理论学习：熟练运用 PWM 原理，实现颜色渐变效果。



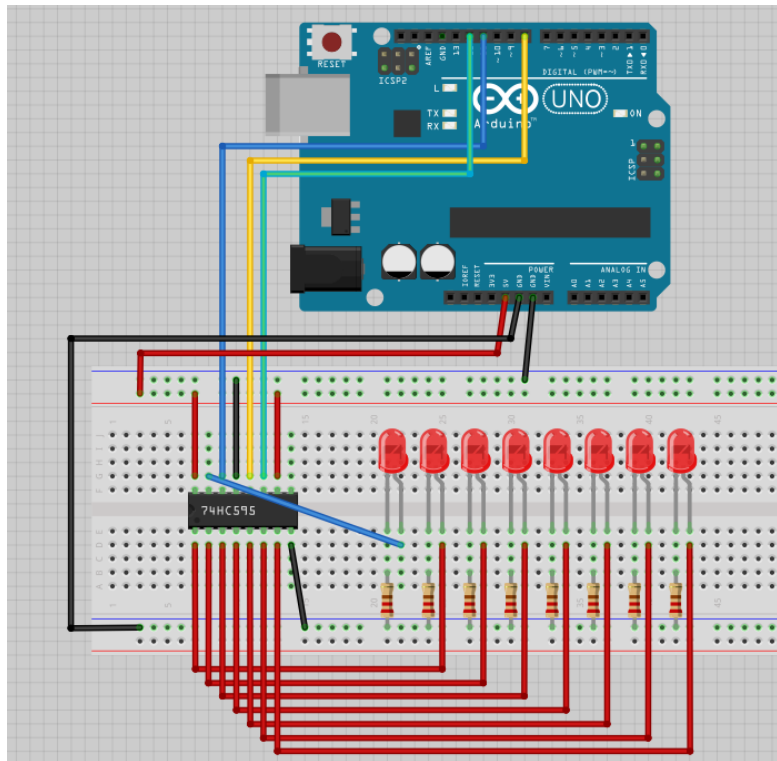
连接图

### 3.20 74HC595 实验

实验现象：利用 74HC595 芯片实现 LED 流水灯效果。

理论学习：

- 掌握 74HC595 的工作原理。
- 实现串行数据转换为并行数据的控制模式。
- 学会 shiftout 语句的用法。



连接图

### 3.21 点阵实验

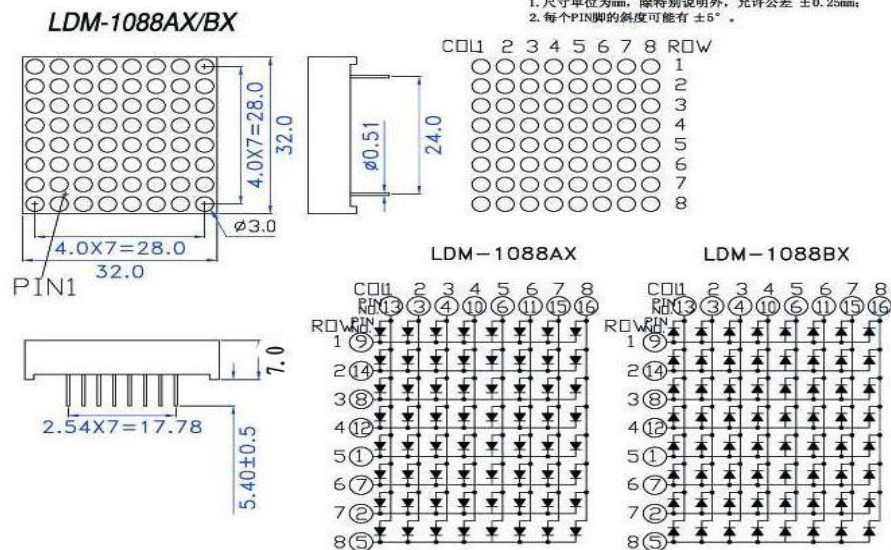
**实验现象：**利用 8\*8 点阵显示出心跳动画。

**理论学习：**

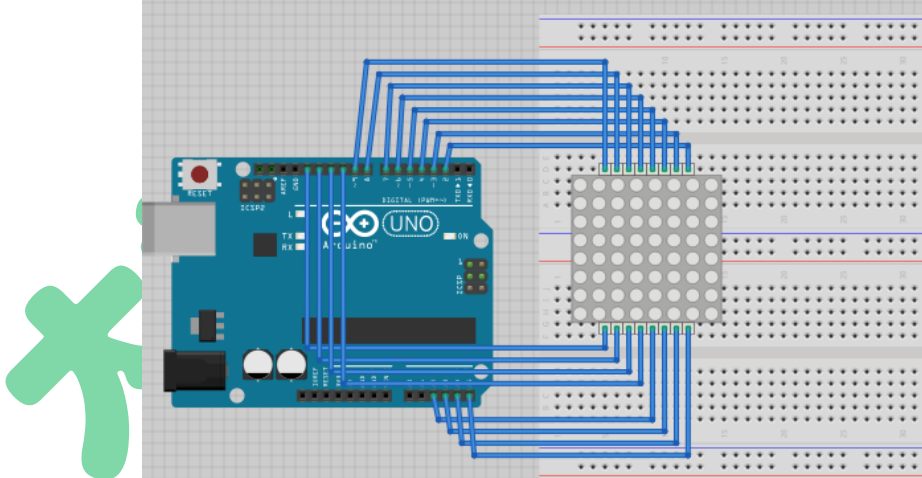
- 本套件配备点阵为共阳点阵（行控制为 LED 阳极）。代码仅适用于共阳点阵。
- 共阳点阵的引脚图



2. 面包板使用注意事项



1. 面包板是可以自己组合的，
2. 套件内配备两块一样的面包板
3. 因受本实验中点阵宽度限制，必须使用两块面包板搭配使用
4. 两块面包板按照如图方式级联：  
将其中一块面包板的电源/地的那行，折弯90度  
将两块面包板级联成如图所示的外观，即可做本实验



连接图

### 3.22 9 克舵机

**实验现象：**通过调整电位器来改变舵机旋转的角度。

(有些电脑的 USB 的驱动能力不足，不能驱动起来舵机，请使用 DC9V 的电源适配器给 arduino 供电。)



## 理论学习:

- ✓ 标配 9g 舵机引脚，黄色为信号线，红色为电源，棕色为地线。
- ✓ 舵机是一种位置伺服的驱动器，具有闭环控制系统的机电结构，由小型直流电机、变速齿轮组、可调电位器、控制板等部件组成由于可以方便地控制舵机旋转的角度（舵角，但是舵角一般不超过  $180^\circ$ ），因此，舵机在要求角度不断变化的控制系统中得到了广泛应用。
- ✓ 舵机在工作中，控制器发出脉冲宽度调制（PWM）信号给舵机，获得直流偏置电压。舵机内部有一个基准电路，产生周期为 20ms，宽度为 1.5ms 的基准信号，将获得的直流偏置电压与电位器的电压比较，获得电压差输出到电机驱动芯片，驱动芯片根据电压差的正负控制电机的正反转。
- ✓ 舵机转动的角度是通过调节 PWM 信号的占空比来实现的，标准 PWM 信号的周期固定为 20ms，理论上脉宽（脉冲的高电平部分）范围在 1ms~2ms 之间，但实际上脉宽可以在 0.5ms~2.5ms 之间，脉宽和舵机的转角  $0^\circ \sim 180^\circ$  相对应。如以脉宽为 0.5ms~2.5ms 范围控制舵机的角度转动，转动范围为  $0^\circ \sim 180^\circ$ 。
- ✓ 小型舵机的工作电压一般为 4.8V 或 6V，转速也不是很快，所以假如更改角度控制脉宽太快时，舵机可能反应不过来。如果需要更快速的反应，就需要更高的转速了。要精确的控制舵机，其实没有那么容易，很多舵机的位置等级有 1024 个，那么，如果舵机的有效角度范围为  $180^\circ$  的话，其控制的角度精度是可以达到  $180^\circ/1024 \approx 0.18^\circ$ ，如果假定脉宽为 0.5ms~2.5ms 范围，则要求的脉宽控制精度为  $(2.5-0.5)\text{ms}/1024 \approx 2 \mu\text{s}$ 。
- ✓ 舵机分别用 0.5ms~2.5ms 之间的脉宽来对应 0 到  $180^\circ$  左右的角度，且转动的角度与脉宽呈线性关系，则舵机每转动  $1^\circ$ ，对应的脉宽为  $(2.5-0.5)\text{ms}/180^\circ$ ，该值除不尽，因此，用一个除不尽的脉冲宽度控制舵机转动，显然转动角度的精度很难控制，为此，实验中以接近 2.5ms 且能够整除 180 的值最为脉宽的变化范围，则取脉宽的范围为 0.5ms~2.48ms，此时，舵机每转动  $1^\circ$ ，则脉宽变化  $(2.48-0.5)\text{ms}/180=11 \mu\text{s}$ 。因此，定义脉宽与转动角度之间的关系为：

$$\text{pulsewidth}=(\text{angle} * 11)+500$$

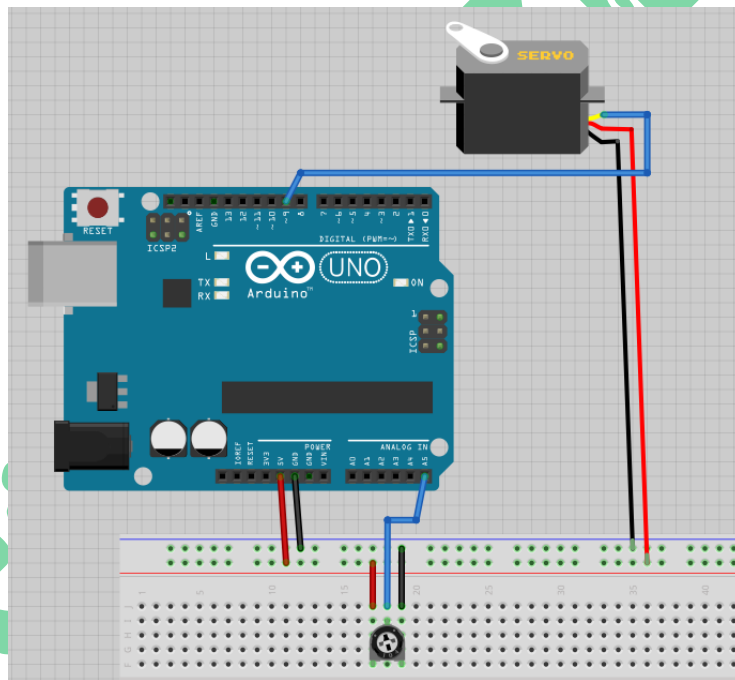
- ✓ 根据这个公式可以写出来设置转动角的函数，然后写出第一个例程。

```
void pulse(int angle)
{
 pulswidth=int ((angle*11)+500);
 digitalWrite(PWM_pin,HIGH);
 delayMicroseconds(pulswidth);
 digitalWrite(PWM_pin,LOW);
 delay(20-pulswidth/1000);
}
```

- ✓ 当然还有更简单的办法：调用伺服电机控制 lib，

```
#include <Servo.h> //引入 lib
Servo myservo; // 创建一个伺服电机对象
#define potpin A5 // 设定连接可变电阻的模拟引脚
int val; // 创建变量，储存从模拟端口读取的值（0 到 1023）
```

```
void setup()
{
 myservo.attach(9); // 9 号引脚输出电机控制信号
 //仅能使用 9、10 号引脚
}
void loop()
{
 val = analogRead(potpin);
 // 读取来自可变电阻的模拟值（0 到 1023 之间）
 val = map(val, 0, 1023, 0, 179); // 利用“map”函数缩放该值，得到伺服电机需
 要的角度（0 到 180 之间）
 myservo.write(val); // 设定伺服电机的位置
 delay(15); // 等待电机旋转到目标角度
}
```



连接图

### 3.23 红外遥控实验

**实验现象：**当按下遥控器上某个按键，串口输出该按键的名称。

**理论学习：**

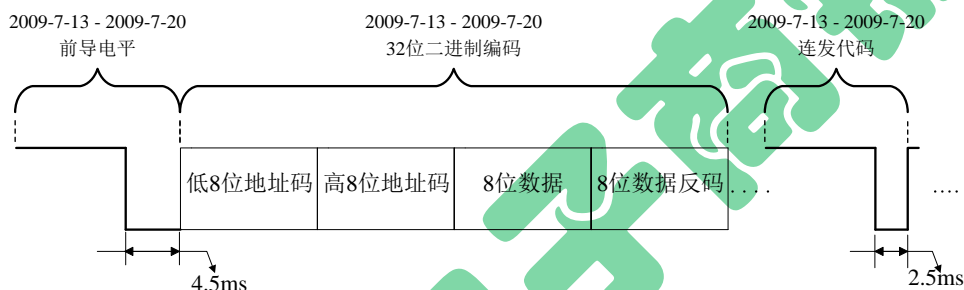


- ✓ 本实验使用红外接收头 VS1838+红外遥控器。VS1838 使用 NEC 码编码格式，
- ✓ NEC 码格式：

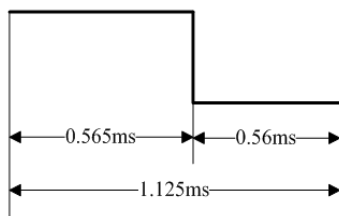
- 1.使用 38 kHz 载波频率
- 2.引导码间隔是 9 ms + 4.5 ms
- 3.使用 16 位客户代码
- 4.使用 8 位数据代码和 8 位取反的数据代码

**备注：红外接收头上得到信号和下面的波形高低电平相反**

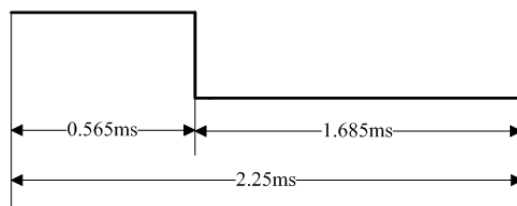
NEC遥控发送数据格式



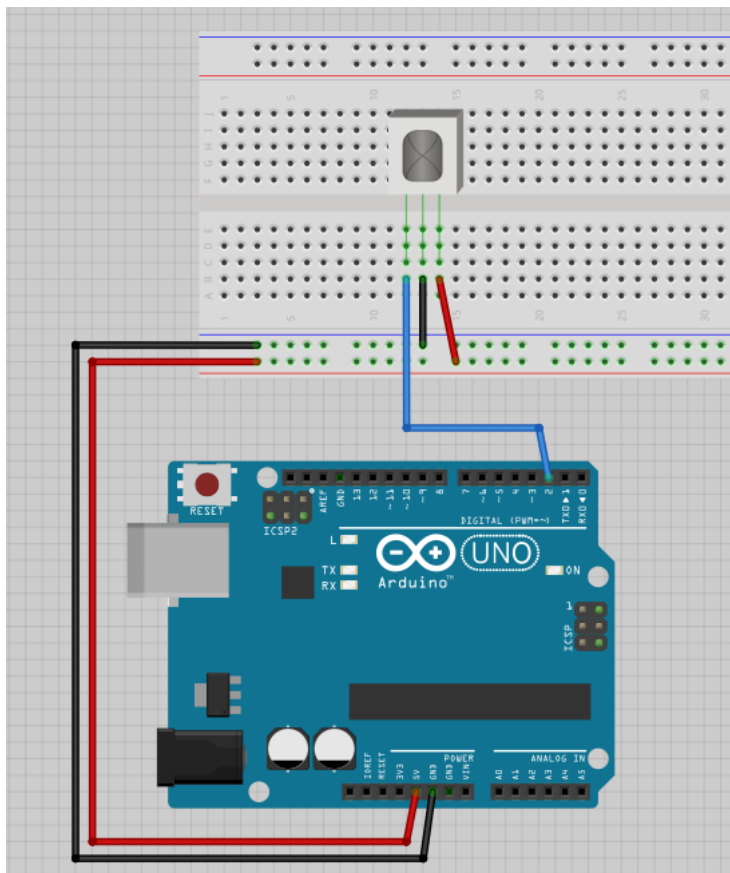
5 二进制 0 和 1 的表示方法：



遥控发射码“0”



遥控发射码“1”



连接图

### 3.24 USB 键盘模拟实验

**实验现象：**模拟键盘上的←和→按键，控制“是男人就下一百层”游戏。

**理论学习：**

利用 Leonardo 板子的 USB 控制器，模仿一个 USB 键盘。

关于 USB 键盘的资料，请查看官方网站：

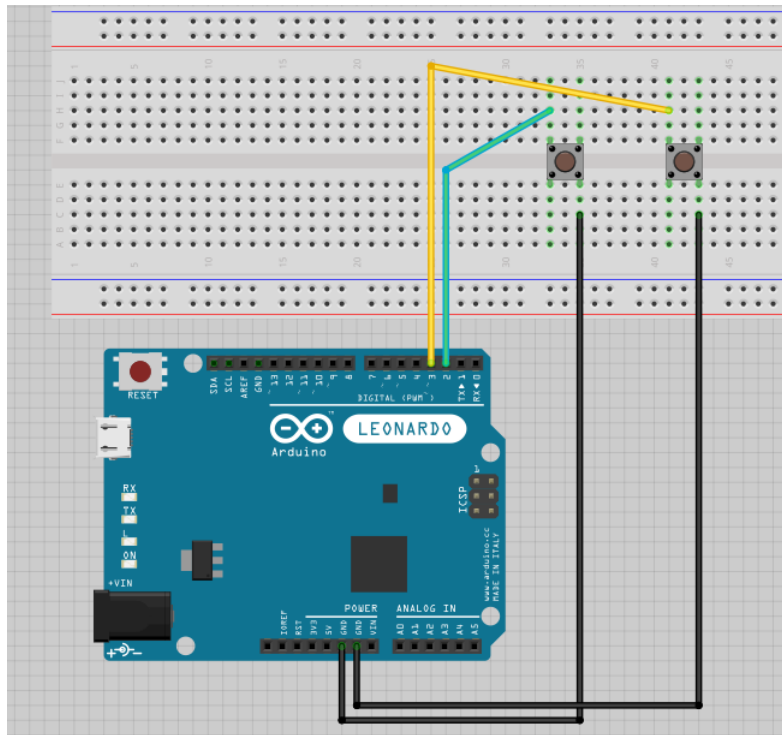
<http://arduino.cc/en/Reference/KeyboardWrite>

关于键盘上的按键名称和键值，请查看：

<http://arduino.cc/en/Reference/KeyboardModifiers>

关于 ASCII 码表，请查看：

<http://www.asciitable.com/>



连接图

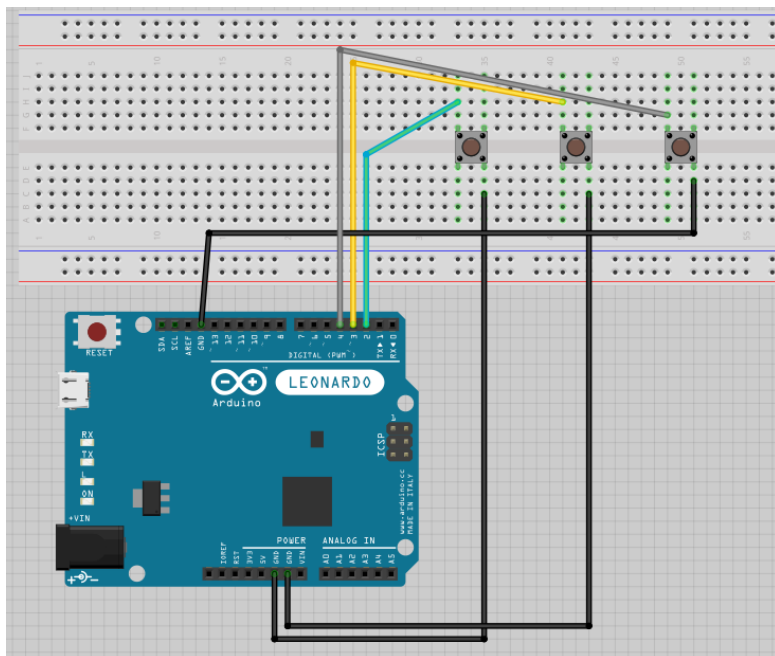
### 3.25 虚拟鼠标实验

**实验现象：**按键模拟鼠标向左移动，向右移动，点击左键等操作。

**理论学习：**

学会 `Mouse.move` 和 `Mouse.click` 的使用。

详情查看：<http://arduino.cc/en/Tutorial/ButtonMouseControl>



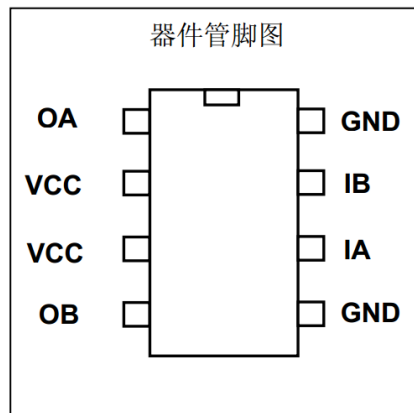
连接图

### 3.26 直流电机驱动实验

**实验现象：**本实验采用小型直流电机驱动 L9110 驱动一个小型直流电机。

**理论学习：**

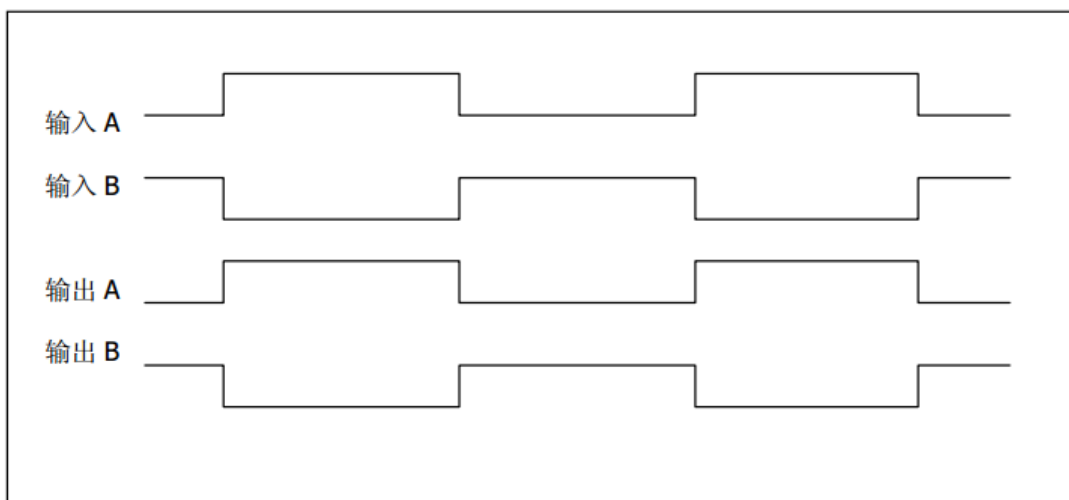
L9110 是为控制和驱动电机设计的两通道推挽式功率放大专用集成电路器件，将分立电路集成在单片 IC 之中，使外围器件成本降低，整机可靠性提高。该芯片有两个 TTL/CMOS 兼容电平的输入，具有良好的抗干扰性；两个输出端能直接驱动电机的正反向运动，它具有较大的电流驱动能力，每通道能通过 750~800mA 的持续电流，峰值电流能力可达 1.5~2.0A；同时它具有较低的输出饱和压降；内置的钳位二极管能释放感性负载的反向冲击电流，使它在驱动继电器、直流电机、步进电机或开关功率管的使用上安全可靠。L9110 被广泛应用于玩具汽车电机驱动、步进电机驱动和开关功率管等电路上。

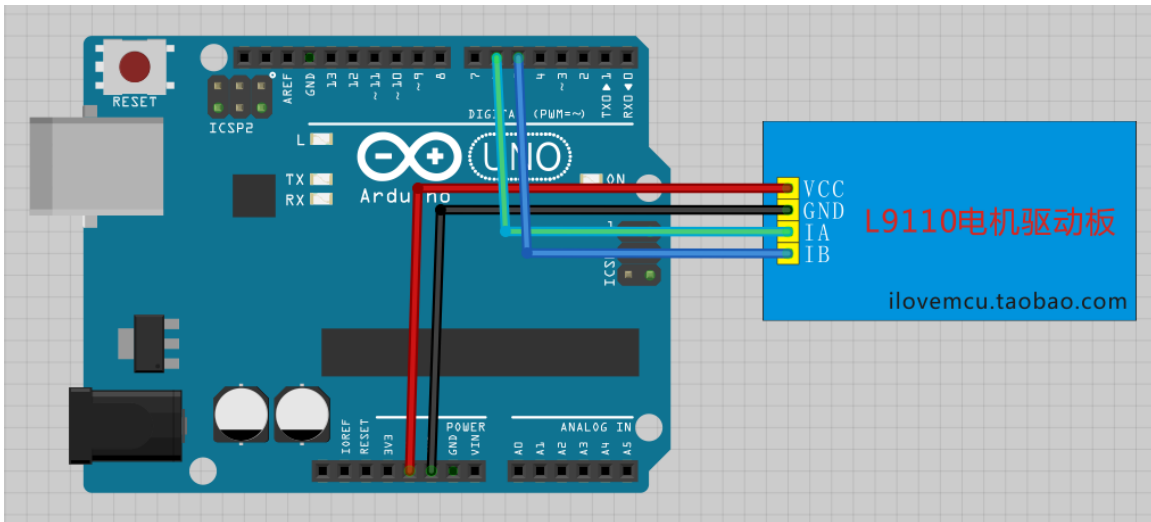


逻辑关系:

| IA | IB | OA | OB |
|----|----|----|----|
| H  | L  | H  | L  |
| L  | H  | L  | H  |
| L  | L  | L  | L  |
| H  | H  | L  | L  |

管脚波形图:



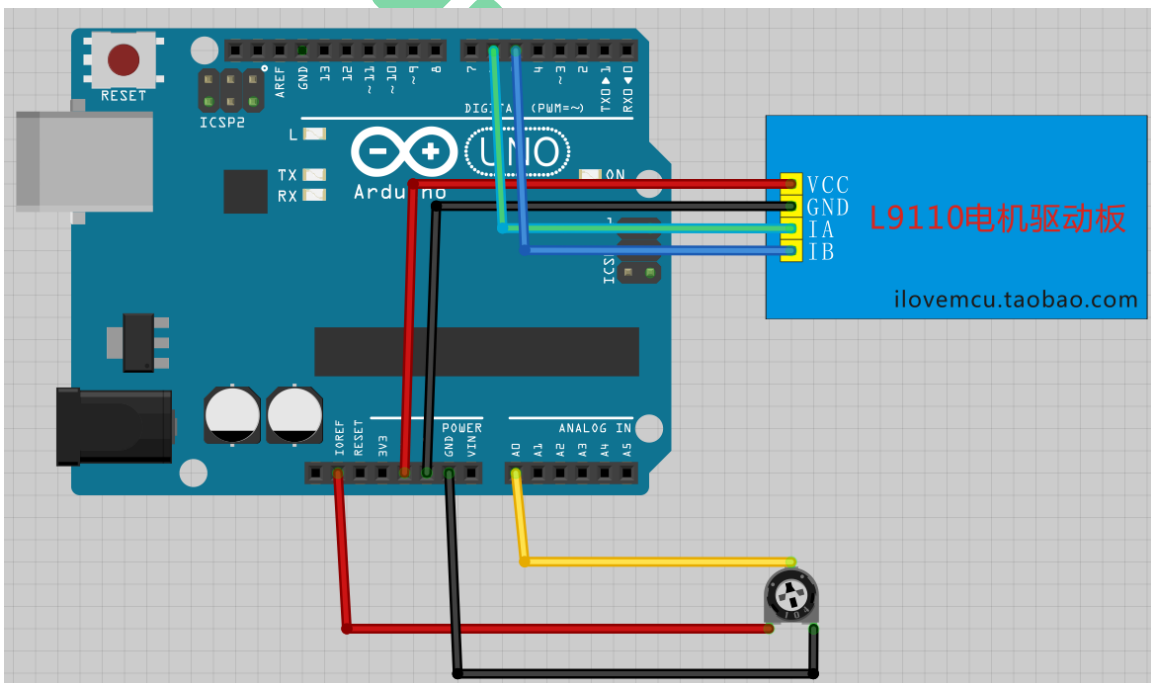


连接图

### 3.27 风扇转速调节实验

**实验现象：**使用电位器调节直流电机转速。

**理论学习：**利用之前学过的电位器模拟值采样和 PWM 输出实验，结合上一讲直流电机驱动，达到电机调速的效果。





## 连接图

这里给大家讲解一个小技巧,5V 接口不够用时候,板子上的IOREF 引脚实际电路内部和5v 是短路的,可以当5v 引脚用。(5v 的单片机IO 的参考肯定是5v 喽~就这个道理!)

### 3.28 串口接收数据实验

**实验现象:** PC 发送一个字符, arduino 板子接收到后返回给 PC.

**理论学习:** 利用 `if (Serial.available() > 0)` 可以查询有没有串口数据输入。

利用 `Serial.read();` 可以将接收到的一个字节的的数据读取出来。

学会利用官方例程编写代码。

### 3.29 串口中断实验

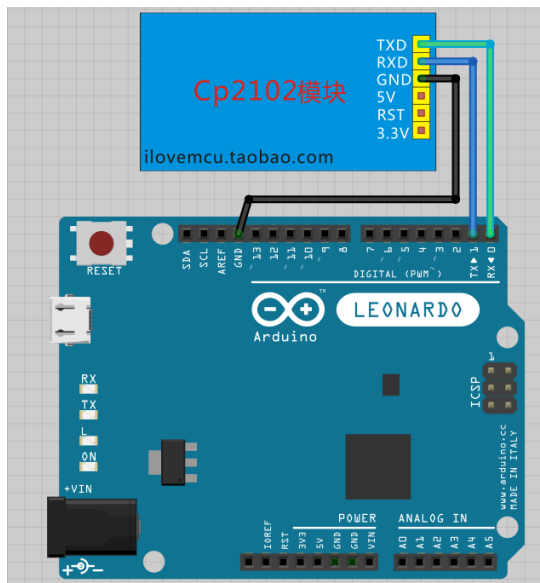
**实验现象:** 通过“串口调试助手”发送一串字符串, arduino 板子接收到后返回给 PC.

**理论学习:** 所谓中断顾名思义就是停止下来办理某些事情。如果没有中断的话, arduino 是一直运行的 loop 内的代码, 一遍一遍重复运行。当有中断产生时候, 单片机会停止 loop 的代码, 开始运行中断服务函数的代码, 运行一遍中断服务函数后继续回到 loop 内接着刚才运行的代码运行。

Arduino 开放的串口中断函数写法: `SerialEvent();`

<http://arduino.cc/en/Tutorial/SerialEvent>

因为 SerialEvent 是基于硬件串口的, 所以模拟的串口是不可以使用 SerialEvent 的。通俗点讲, UNO 的 USB 接口连接的串口 (数字 IO 的 0 和 1) 可以使用 SerialEvent。Leonardo 的 USB 接口连接的串口 (使用 USB 虚拟的 COM), 不能使用 SerialEvent, 但是数字 IO 的 0 和 1 是真实的硬件串口, 可以使用 SerialEvent, 这里要注意, leonardo 的数字 IO 的 0 和 1 是调用的 Serial1 类, 而 SerialEvent 的标准写法应该是 SerialEvent1。而 Serial1 需要用外部的串口电路去验证功能, 详细请按照视频教程操作使用 CP2102 usb-ttl 模块来测试 SerialEvent1。



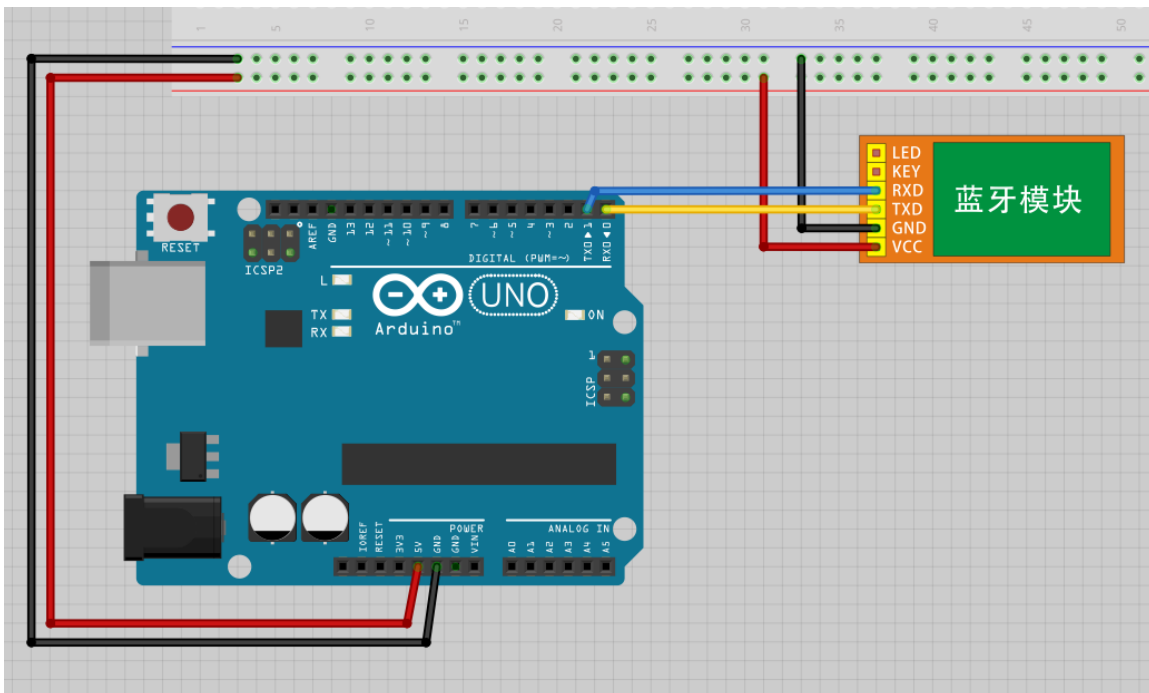
## Leonardo 和 cp2102 模块连接图

### 3.30 手机和蓝牙模块通讯实验

**实验现象：**使用安卓手机控制继电器的开关。

理论学习:

- ✓ 本实验采用本店开发的 BT-HC05 蓝牙模块，该模块主从一体，可以软件设置是主机还是从机。（发货默认从机，波特率 9600，PIN 密码 1234）。和手机连接使用从机模式不需要额外设置。
- ✓ 该模块很简单，默认就是串口的透传模式，只要连接好线就当无线串口用就行了。
- ✓ 原来 **arduino** 和手机通讯也如此简单！



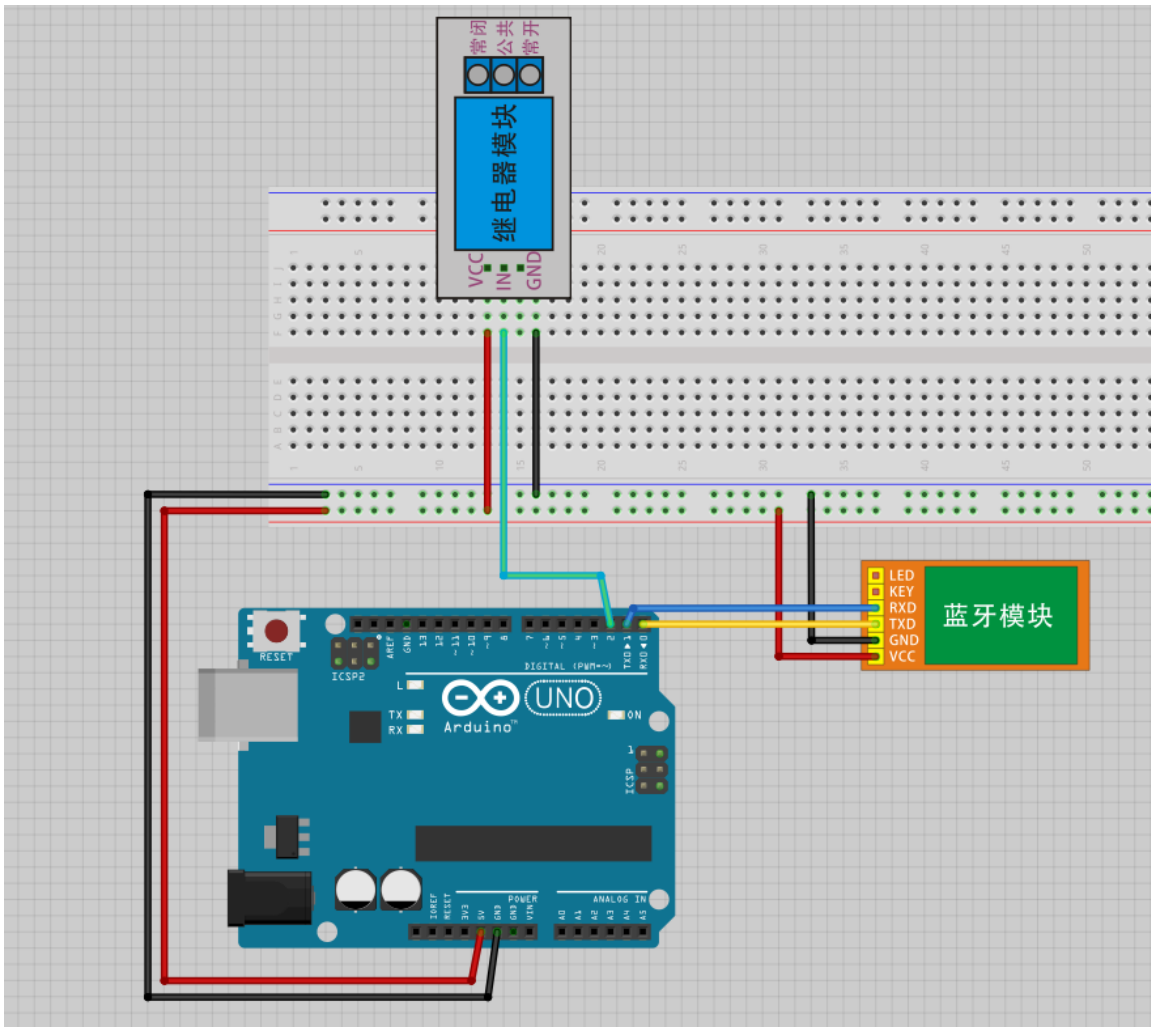
连接图

### 3.31 手机蓝牙控制继电器实验

**实验现象：**利用手机无线开关继电器

**理论学习：**

- ✓ 电磁继电器一般由铁芯、线圈、衔铁、触点簧片等组成的。只要在线圈两端加上一定的电压，线圈中就会流过一定的电流，从而产生电磁效应，衔铁就会在电磁力吸引的作用下克服返回弹簧的拉力吸向铁芯，从而带动衔铁的动触点与静触点（常开触点）吸合。当线圈断电后，电磁的吸力也随之消失，衔铁就会在弹簧的反作用力返回原来的位置，使动触点与原来的静触点（常闭触点）释放。这样吸合、释放，从而达到了在电路中的导通、切断的目的。对于继电器的“常开、常闭”触点，可以这样来区分：继电器线圈未通电时处于断开状态的静触点，称为“常开触点”；处于接通状态的静触点称为“常闭触点”。继电器一般有两股电路，为低压控制电路和高压工作电路。
- ✓ 继电器一般是低压控制高压的一种装置，顾一般需要通过三极管隔离驱动，本套餐内模块使用 NPN 的三极管驱动，当控制引脚给高电平时继电器常开端吸合/D2 点亮，低电平时，继电器常开端断开/D2 熄灭。



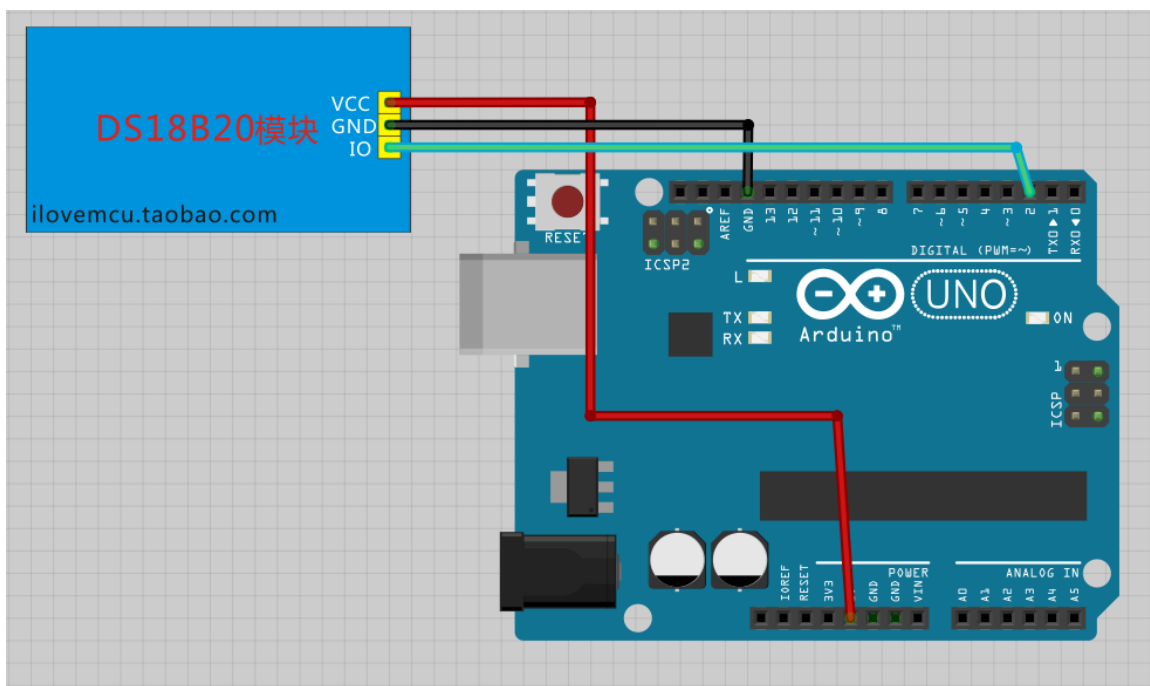
连接图

### 3.32 温度传感器 DS18B20 实验

**实验现象：**Arduino 读取温度传感器 DS18B20 的值，然后计算出当前温度，通过串口发送给 PC，PC 串口调试助手显示。

**理论学习：**

- ✓ DS18B20 数字温度传感器采用单总线模式，和 MCU 通讯仅需要一个 IO，测温范围-55~125℃
  - ✓ 温度数字量转换时间 200ms（典型值），即每次读取的时间间隔最好大于 200ms
- 其它细节的指标请查看 DS18B20 手册（[Arduino 资料包\7.芯片资料\DS18B20 中文资料](#)）



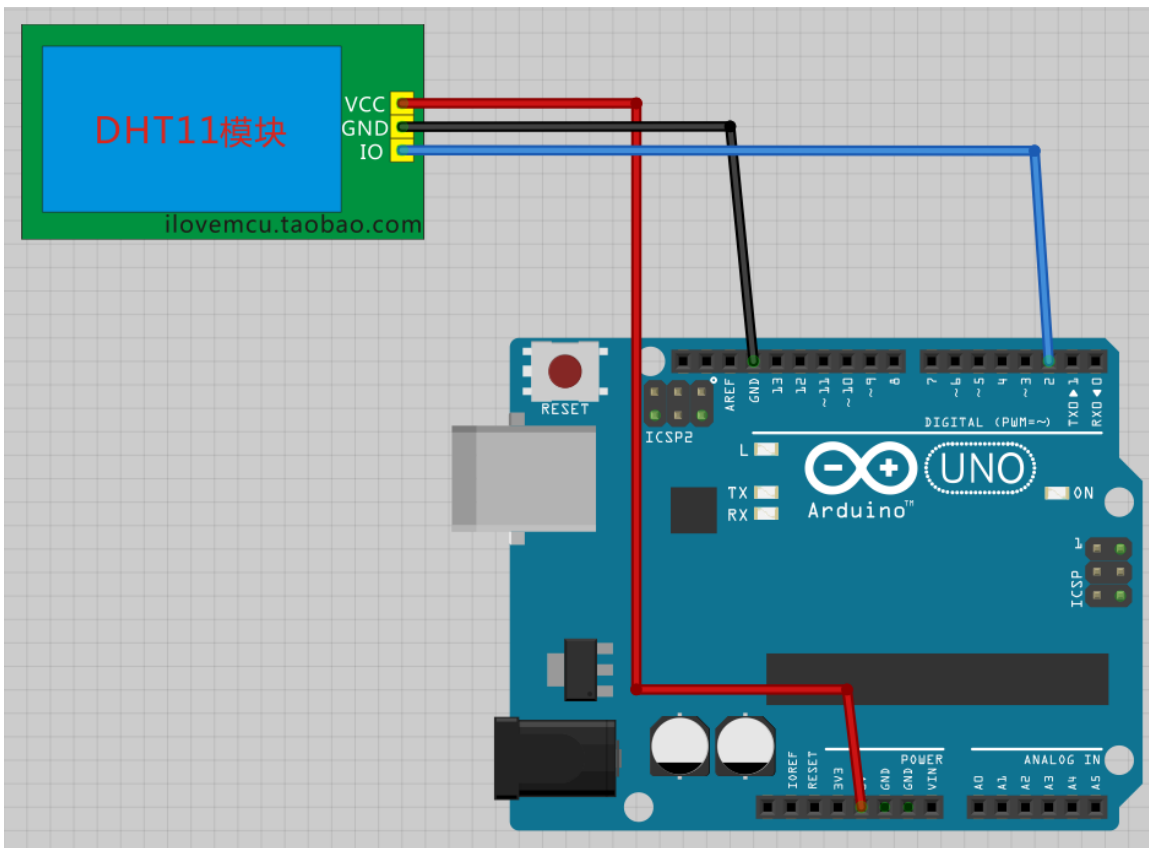
连接图

### 3.33 温湿度传感器 DHT11 实验

**实验现象：**读取 DHT11 温湿度传感器的值,然后发送给 PC 通过串口显示出来。

**理论学习：**

- ✓ DHT11 是一款单总线的温湿度传感器。  
供电电压： 3.3~5.5V DC  
输出： 单总线数字信号  
测量范围： 湿度 20-90%RH， 温度 0~50℃  
测量精度： 湿度+5%RH， 温度+2℃  
分辨率： 湿度 1%RH， 温度 1℃
- ✓ DHT11 传感器的初始化时序和读取 0/1 的时序要求请参考 DHT11 说明书  
([Arduino 资料包\7.芯片资料\DHT11 说明书](#))
- ✓ 本程序通过调用头文件DHT11.h内的DHT11\_Read();来读取当前的温湿度值，湿度值存放在 HUMI\_Buffer\_Int中，温度值存放在TEM\_Buffer\_Int中，然后通过通过串口发送出来。



连接图

### 3.34 实时时钟 DS1302 实验

#### 实验现象：

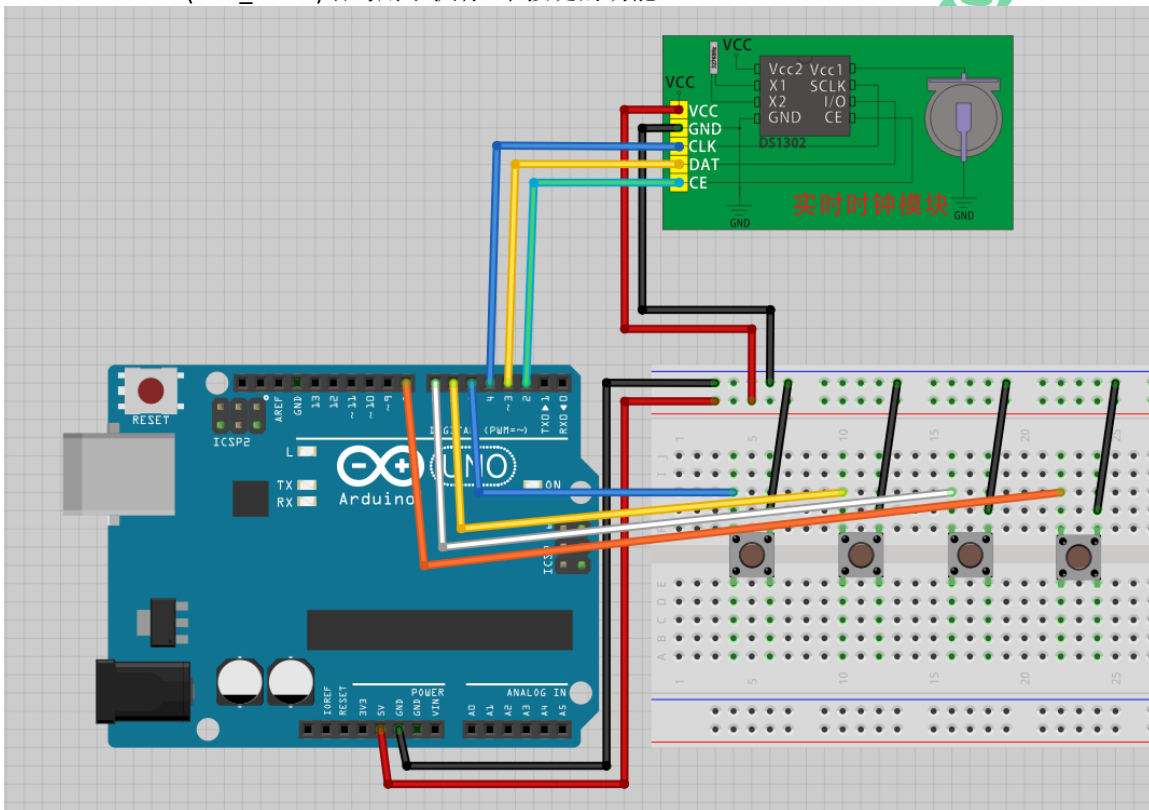
- ✓ 通过 DS1302 的读取获取年/月/日/星期/时/分/秒的值，然后串口显示在 PC 上。
- ✓ 可以通过 4 个按键设定当前的时间。
- ✓ 按键 1（设置）：按一下进入设置状态。再按一下退出设置状态。
- ✓ 按键 2（切换）：按一下切换一种设置变量，分别可以设置年/月/日/星期/时/分/秒
- ✓ 按键 3（增一）：可以对当前的设置变量进行加一操作。
- ✓ 按键 4（减一）：可以对当前的设置变量进行减一操作。

#### 理论学习：

- ✓ DS1302 是美国 DALLAS 公司推出的一种高性能、低功耗的实时时钟芯片，附加 31 字节静态 RAM，采用 SPI 三线接口与 CPU 进行同步通信，并可采用突发方式一次传送多个字节的时钟信号和 RAM 数据。实时时钟可提供秒、分、时、日、星期、月和年，一个月小与 31 天

时可以自动调整，且具有闰年补偿功能。工作电压宽达 2.5~5.5V。采用双电源供电（主电源和备用电源），可设置备用电源充电方式，提供了对后背电源进行涓细电流充电的能力。细节参数请查看 DS1302 中文资料（[Arduino 资料包\7.芯片资料\DS1302 中文资料](#)）。

- ✓ 程序通过调用头文件DS1302.h里面的DS1302\_GetTime(&DS1302Buffer)函数获取当前的实时时钟，存放在结构体变量DS1302Buffer中，然后通过Display\_RTCC()函数把实时时钟通过串口发送出来。
- ✓ Scan\_KEY();语句用于扫描4个按键的按键动作。
- ✓ switch( KEY\_NUM )语句用于执行4个按键的功能。



连接图

### 3.35 烟雾传感器 MQ-2 实验

**实验现象:** 烟雾传感器把烟雾浓度值转换为模拟电压值输出,Arduino 通过 AO 引脚读取模拟电压值,通过串口发送给 PC,当模拟电压值超过模块的比较器设定值的时候,模块 DO 引脚输出低电平,Arduino 监控到 DO 引脚拉低信号时候,板载 LED 点亮报警,串口输出 Alarm 信号。报警的阈值可以通过调节板载的电位器调整大小。

理论学习:

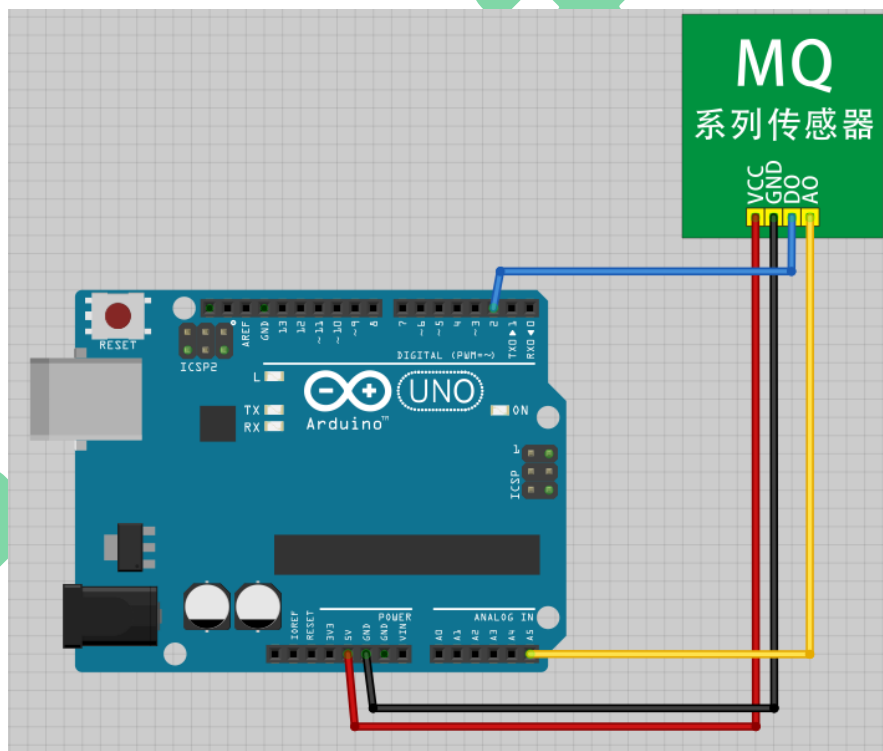
- ✓ MQ 系列传感器常见型号:



- MQ-2 烟雾传感器
- MQ-3 酒精传感器
- MQ-4 甲烷、天然气传感器
- MQ-5 煤气传感器
- MQ-6 液化气体传感器
- MQ-7 一氧化碳传感器
- MQ-8 氢气传感器
- MQ-9 一氧化碳传感器

✓ 模块使用说明:

- 1.VCC/GND 之间加上 5v 电压;
- 2.等待大概 30s 传感器预热后, 读取模拟输出 AO 引脚电压;
- 3.通过调节电位器,改变 LED 报警阈值
- 4.当浓度值大于设定值时候, 模块上 LED 指示灯点亮,同时 D0 引脚输出低电平。



连接图

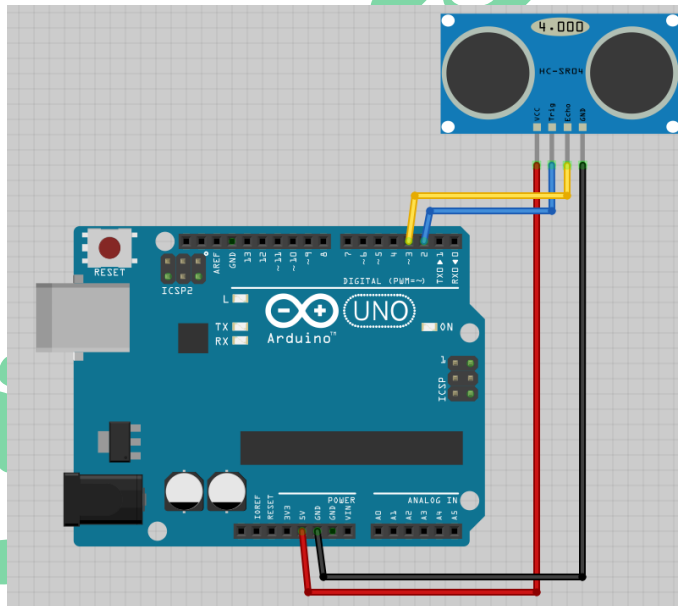
### 3.36 超声波测距



实验现象：串口输出超声波模块和格挡物体之间的距离，单位为 cm

理论学习：

- ✓ 超声波测距原理：
  - (1)采用 trig 触发测距，给至少 10us 的高电平信号；
  - (2)模块自动发送 8 个 40khz 的方波，自动检测是否有信号返回；
  - (3)有信号返回，通过 echo 输出高电平，高电平持续的时间就是距离的 2 倍；
  - (4)超声波从发射到返回的时间。测试距离=(高电平时间\*声速(340M/S))/2
- ✓ 测距程序核心代码 pulseIn(pin, value) 函数：读取一个引脚的脉冲（HIGH 或 LOW）。例如，如果 value 是 HIGH，pulseIn()会等待引脚变为 HIGH，开始计时，再等待引脚变为 LOW 并停止计时。返回脉冲的长度，单位微秒。如果在指定的时间内无脉冲函数返回。此函数的计时功能由经验决定，长时间的脉冲计时可能会出错。计时范围从 10 微秒至 3 分钟。（1 秒=1000 毫秒=1000000 微秒），**请注意单位为 us。**  
pin:你要进行脉冲计时的引脚号。  
value:要读取的脉冲类型，HIGH 或 LOW。



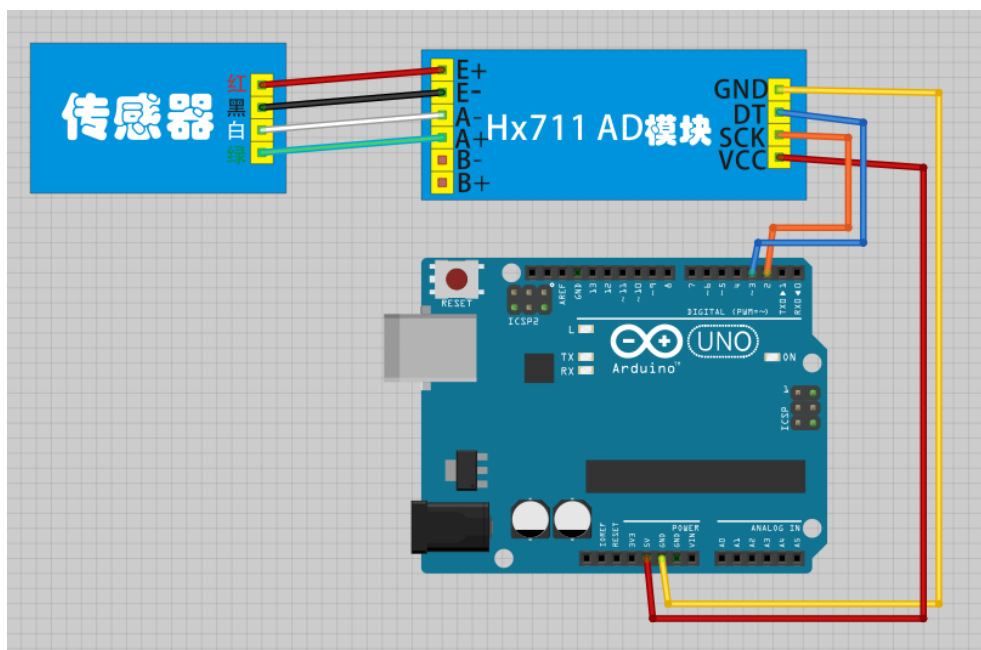
连接图

### 3.37 ARDUINO 电子秤制作

实验现象：通过砝码校准好电子秤参数，串口输出砝码值（误差±1g）。

理论学习：

- ✓ HX711 AD 模块是高精度 24bit AD，内部带 128 倍增益，可方便测量 mV 电压。
- ✓ 3Kg 压力拉力传感器分辨率 1mV/V (例如传感器供电 5v，传感器最大输出 5mV)。
- ✓ Arduino 把接收到的 HX711 的串行数据，转换为砝码的重量，通过串口输出显示。



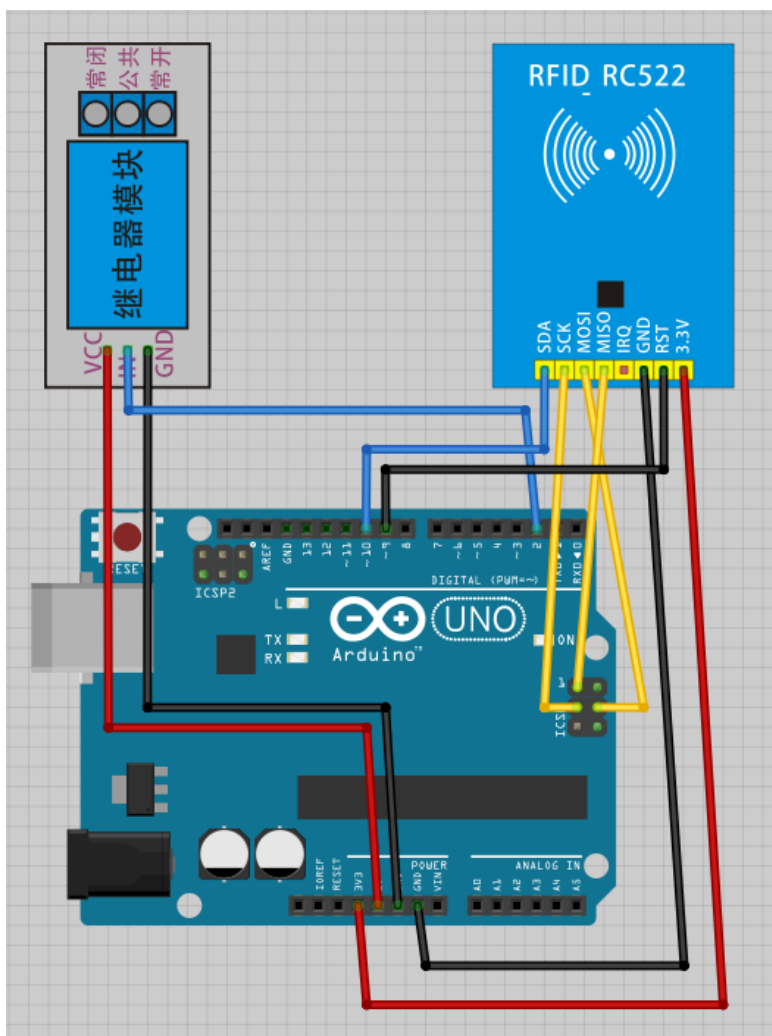
连接图

### 3.38 RFID 实验

**实验现象：**可以通过配置的两个磁卡，让继电器模块吸合。

**理论学习：**本模块采用 Philips MFRC522 原装芯片设计读卡电路，使用方便，成本低廉，适用于设备开发、读卡器开发等高级应用的用户、需要进行射频卡终端设计/生产的用户。本模块可直接装入各种读卡器模具。模块采用电压为 3.3V,通过 SPI 接口简单的几条线就可以直接与用户任何 CPU 主板相连接通信,可以保证模块稳定可靠的工作、读卡距离远。

**特别注意：**UNO 板子和 leonardo 板子的 SPI 接口不一样，UNO 板子的 SPI 和数字 IO 的 11、12、13 复用，leonardo 的不复用。



连接图

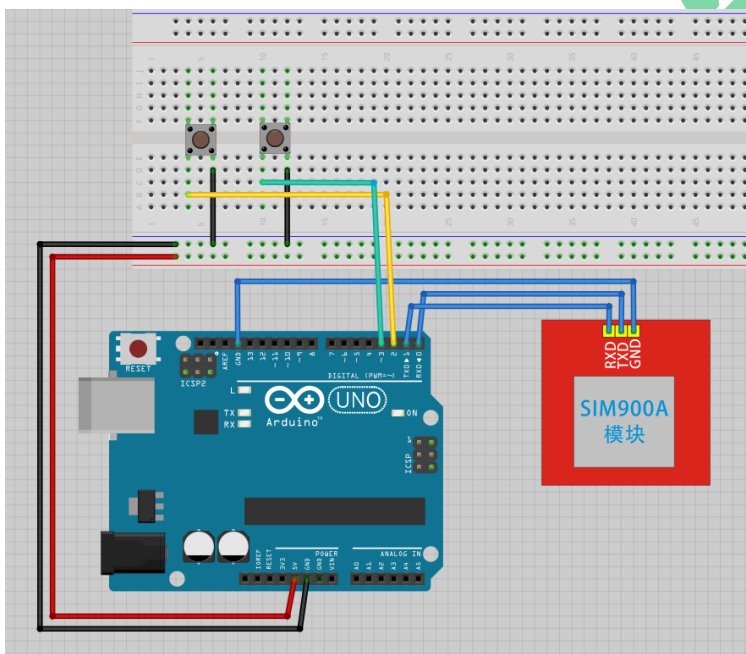
### 3.39 GSM 模块 SIM900A 拨打电话实验

**实验现象：**按下按键 1 后，SIM900 向指定手机号拨打电话。按下按键 2，挂断电话。

**理论学习：**

- ✓ 学会使用 CP2102 来调试 SIM900A 模块
- ✓ ATD 拨打电话指令。
- ✓ ATH 挂断电话指令。
- ✓ 每条指令后面要加回车（\r\n）。
- ✓ 模块使用顺序：

1. 将一张移动手机卡插到模块背面的 SIM 卡插槽。
2. 使用标配的 9v 电源给 SIM900A 模块供电,打开模块上的供电电源。  
按下 SIM900A 模块上的 ON/OFF 按键,给模块开机。等待 LED 指示灯 STA 注册到网络 (64ms 亮/3000ms 灭)
3. 下载例程到 Arduino uno (一定要在未连接 RX/TX 引脚时候下载程序,否则会下载程序不成功)。
4. 按照连接图连接各导线。
5. 按面包板上按键 1, 拨打指定手机号码。
6. 按面包板上按键 2, 挂断电话。



连接图

### 3.40 GSM 模块 SIM900A 发送短信

**实验现象:** 按下面包板上按键后, 向指定手机发送一条短信, 短信内容为 “Hello World! ”。

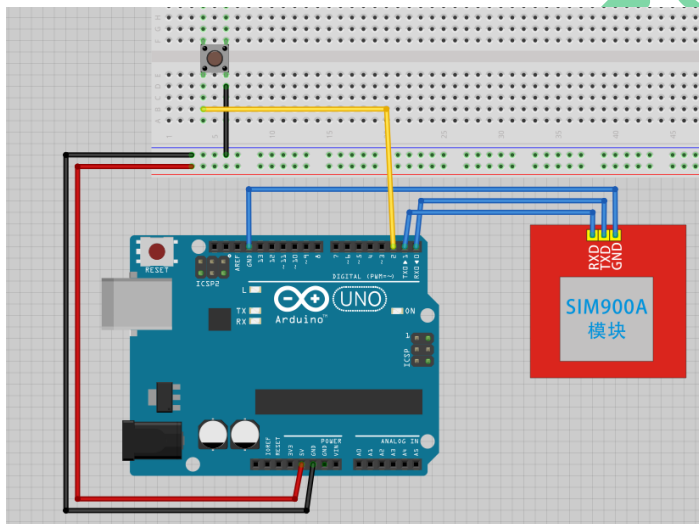
**理论学习:**

- ✓ AT+CSCS="GSM" 设置为缺省字符集
- ✓ AT+CMGF, 用于设置短消息模式, SIM900A 支持 PDU 模式和文本 (TEXT) 模式等 2 种模式, 发送: AT+CMGF=1, 即可设置为文本模式。  
本短信发送英文短信, 因此使用 AT+CMGF=1
- ✓ AT+CMGS, 用于发送短信, 在 "GSM" 字符集下, 最大可以发送 180 个字节的英文字符,

在"UCS2"字符集下，最大可以发送 70 个汉字（包括字符/数字）。

✓ 模块使用顺序：

1. 将一张移动手机卡插到模块背面的 SIM 卡插槽。
2. 使用标配的 9v 电源给 SIM900A 模块供电,打开模块上的供电电源。按下 SIM900A 模块上的 ON/OFF 按键，给模块开机。等待 LED 指示灯 STA 注册到网络（64ms 亮 /3000ms 灭）
3. 下载例程到 Arduino UNO（一定要在未连接 RX/TX 引脚时候下载程序，否则会下载程序不成功）。
4. 按照连接图连接各导线。
5. 按面包板上按键后，向指定手机发送一条短信。



连接图

### 3.41 卫星 GPS 信息获取和解析

实验现象：

- ✓ Arduino 接收到 GPS 模块发来的信息直接转发给 PC。
- ✓ Arduino 把 GPS 信息进行处理解析为有用信息，然后发送给 PC。

理论学习：

- ✓ GPS 模块是接收卫星的信息然后用串口发送出来的装置，本店 GPS 采用 U-BLOX 模组，自带可充电后备电池（以支持温启动或热启动，后备电池在主电源断电后，可以维持半小时左右的 GPS 接收数据保存）；
- ✓ 模块通过串口与外部系统连接，串口波特率：4800、9600（默认）、38400、57600 等不同速率；



✓ 兼容 5V/3.3V 单片机系统，可以非常方便的与您的产品进行连接。

✓ 最常用指令解析：

➤ \$GPGGA (GPS 定位信息, Global Positioning System Fix Data)

\$GPGGA 语句的基本格式如下 (其中 M 指单位 M, hh 指校验和, CR 和 LF 代表回车换行, 下同)：

**\$GPGGA,(1),(2),(3),(4),(5),(6),(7),(8),(9),M,(10),M,(11),(12)\*hh(CR)(LF)**

- (1) UTC 时间, 格式为 hhmmss.ss;
- (2) 纬度, 格式为 ddmm.mmmmm (度分格式);
- (3) 纬度半球, N 或 S (北纬或南纬);
- (4) 经度, 格式为 dddmm.mmmmm (度分格式);
- (5) 经度半球, E 或 W (东经或西经);
- (6) GPS 状态, 0=未定位, 1=非差分定位, 2=差分定位;
- (7) 正在使用的用于定位的卫星数量 (00~12)
- (8) HDOP 水平精确度因子 (0.5~99.9)
- (9) 海拔高度 (-9999.9 到 9999.9 米)
- (10) 大地水准面高度 (-9999.9 到 9999.9 米)
- (11) 差分时间 (从最近一次接收到差分信号开始的秒数, 非差分定位, 此项为空)
- (12) 差分参考基站标号 (0000 到 1023, 首位 0 也将传送, 非差分定位, 此项为空)

举例如下：

**\$GPGGA,023543.00,2308.28715,N,11322.09875,E,1,06,1.49,41.6,M,-5.3,M,,\*7D**

➤ \$GPRMC (推荐定位信息, Recommended Minimum Specific GPS/Transit Data)

\$GPRMC 语句的基本格式如下：

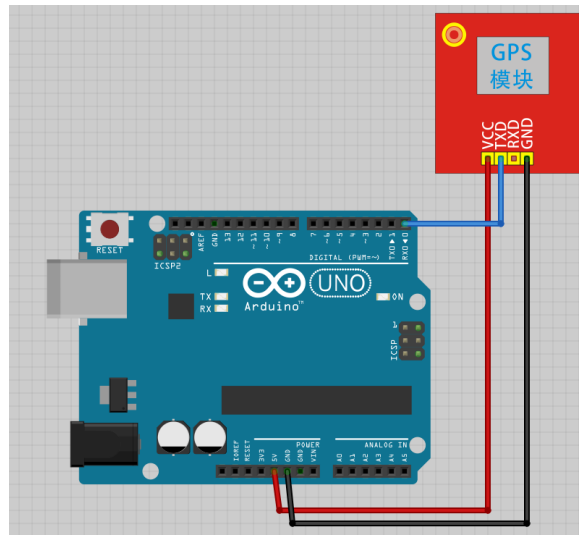
**\$GPRMC,(1),(2),(3),(4),(5),(6),(7),(8),(9),(10),(11),(12)\*hh(CR)(LF)**

- (1) UTC 时间, hhmmss (时分秒)
- (2) 定位状态, A=有效定位, V=无效定位
- (3) 纬度 ddmm.mmmmm (度分)
- (4) 纬度半球 N (北半球) 或 S (南半球)
- (5) 经度 dddmm.mmmmm (度分)
- (6) 经度半球 E (东经) 或 W (西经)
- (7) 地面速率 (000.0~999.9 节)
- (8) 地面航向 (000.0~359.9 度, 以真北方为参考基准)
- (9) UTC 日期, ddmmyy (日月年)
- (10) 磁偏角 (000.0~180.0 度, 前导位数不足则补 0)
- (11) 磁偏角方向, E (东) 或 W (西)
- (12) 模式指示 (A=自主定位, D=差分, E=估算, N=数据无效)

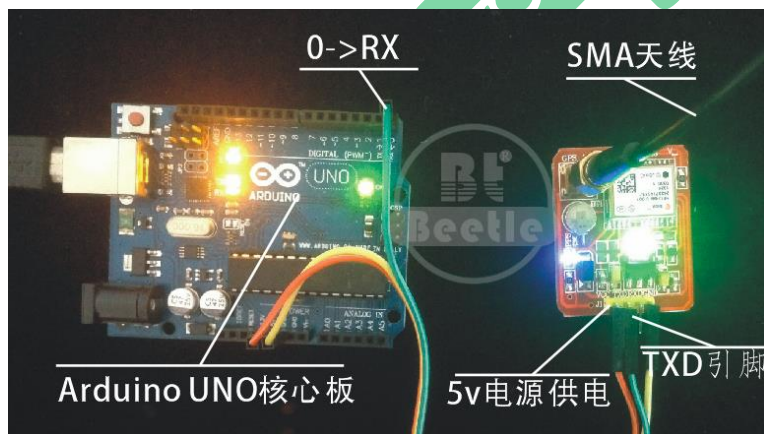
举例如下：

**\$GPRMC,023543.00,A,2308.28715,N,11322.09875,E,0.195,,240213,,,A\*78**

➤ 其它指令解析请参考 GPS 手册。



连接图



实物效果图