.

# WU-Link 物联网套件

# 教程

好好搭搭在线

# 目　录

## 第一部分 硬件入门

# 第二部分 编程入门



# 第三部分 硬件进阶

# 第四部分 编程进阶



# 第五部分 物联网通讯

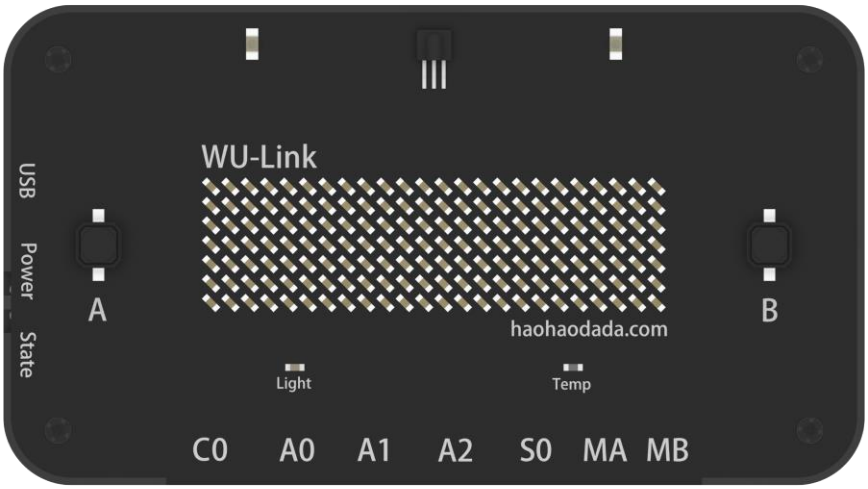
# 第一部分

# 硬件入门

# 第一节 初识 WU-Link 物联网套件

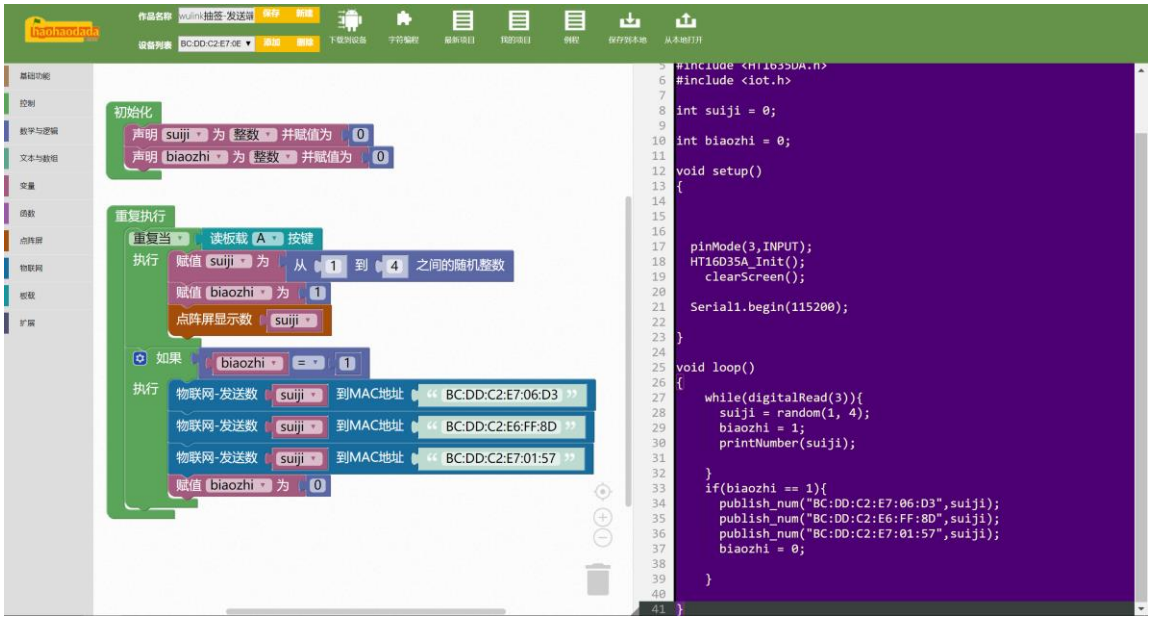
## 一、 认识 WU-Link 物联网套件

WU-Link 是基于云技术的一款物联网套件，集云编程、无线下载、物联网通讯功能于一体，既能编写兼容 Arduino 的应用程序，也能编写基于物联网的应用程序。




新一代图形化编程云平台，支持远程下载，无需连接数据线，只要 WU-Link 开机联网，无论身处何地，均可对其编程。全新的图形化编程云平台完全基于 Web 开发，跨系统支持，同时支持 PC、智能手机和平板电脑，编程体验更加灵活。

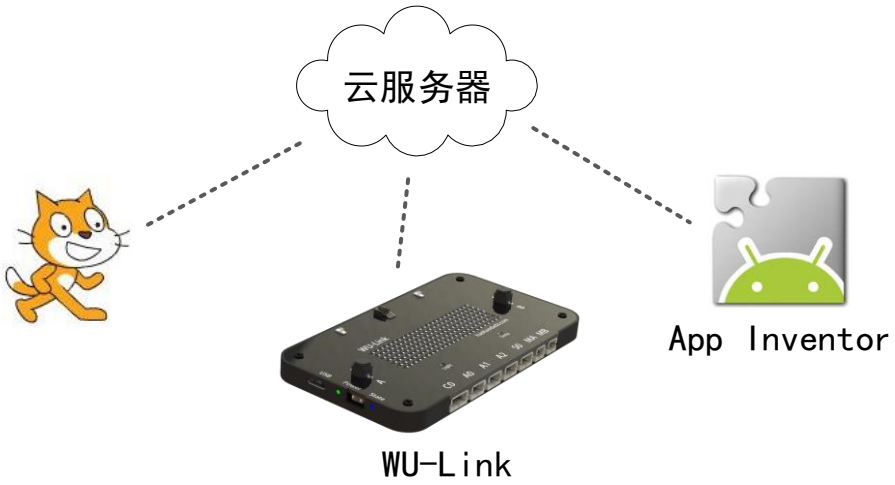
图形编程界面与字符编程界面之间方便切换、共存，可满足各种不同的学习场合。




WU-Link 能方便的实现与 Scratch、 App Inventor 及其它物联网设备的信息通讯，实现了 PC、智能手机、平板电脑和物联网硬件的多终端跨系统互联互通。

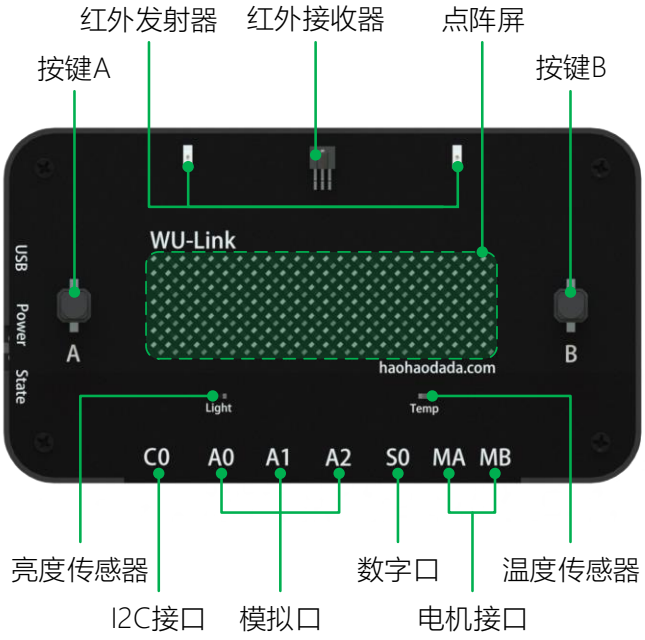
WU-Link 实现了《Scratch 编程》与《手机 APP 编程》等多平台编程课程的互联互通，能

让学生在创造性活动中不断的将新旧知识技能融会贯通。




WU-Link 除了物联网功能外，它本身是一款优秀的单机开发套件，内置亮度、温度、电子罗盘、加速度计、按键、蜂鸣器、红外接收、红外发射、7×24 点阵 LED 等输入输出器件以及 2200mAh 的锂电池。

扩展接口提供了一个 I2C 接口、三个模拟接口、一个数字接口以及两个电机驱动接口。能轻松实现大多数基于 Arduino 的应用开发。




接下来用一个案例，亲身体验 NOVA 电子积木的使用。

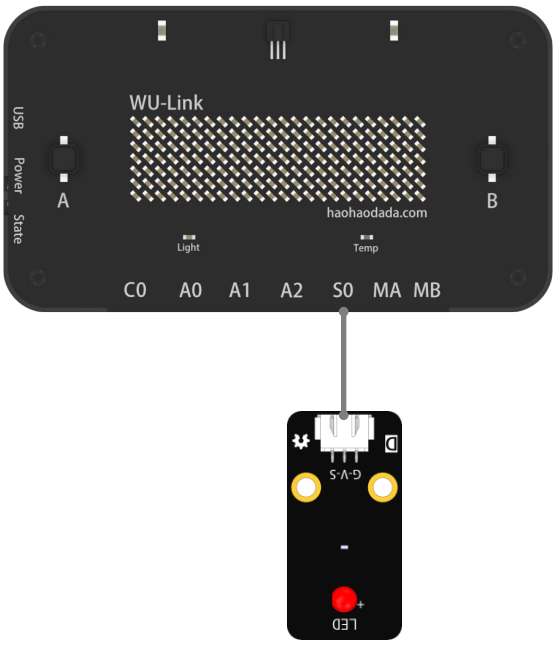
**示例 1-1：向世界说 Hello World!**

在 WU-Link 的点阵屏上显示“Hello World!”。

**元器件列表：**

WU-Link 主控板 ×1

**电路连接：**

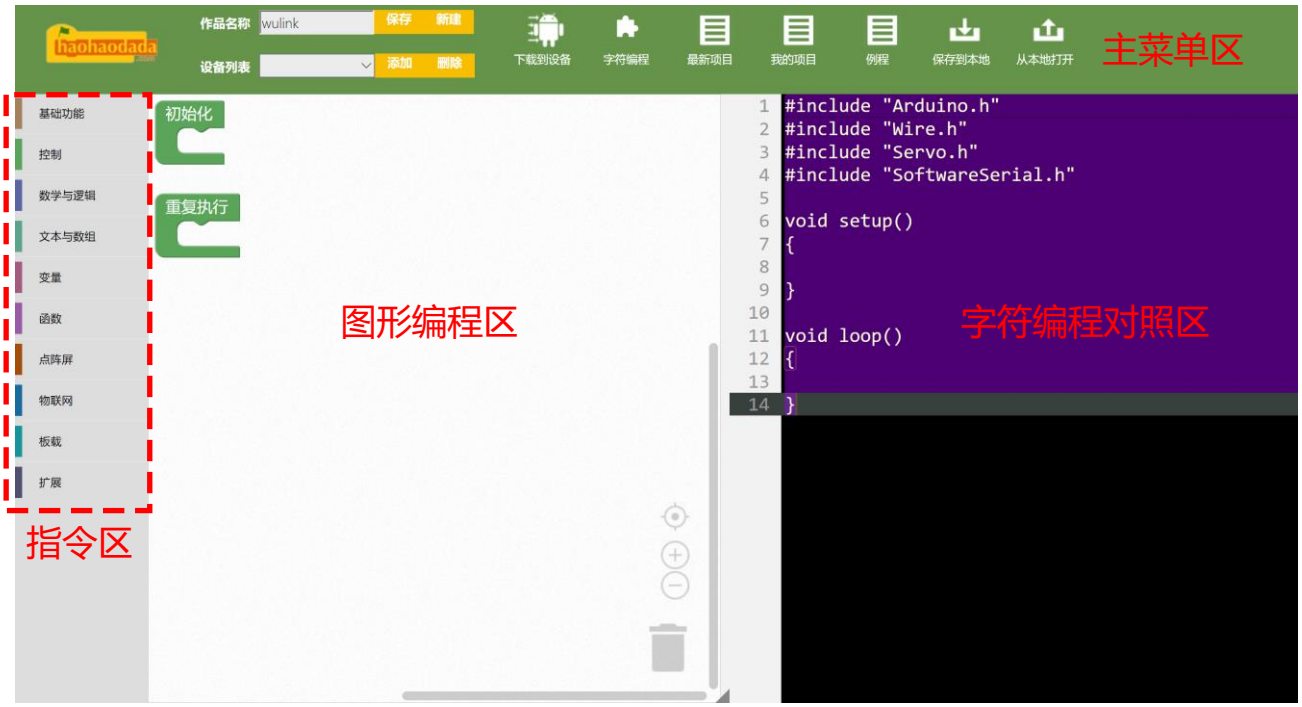



**网络配置**

WU-Link 设备需要通过连接本地的无线网络才能接入物联网中，所以在使用 WU-Link 编程之前需要进行网络配置。可以通过 Web 配置和微信小程序配置两种方式连入无线网络，详见附录三和附录四。

**进入编程界面：**

登入网址：“http://haohaodada.com/wulink”



编程界面分成四个大的区块：

主菜单区：修改作品名称、打开和保存作品、下载程序、切换图形/字符编程界面。

指令区：将所有指令分成了 10 个类目，用不同颜色区分。

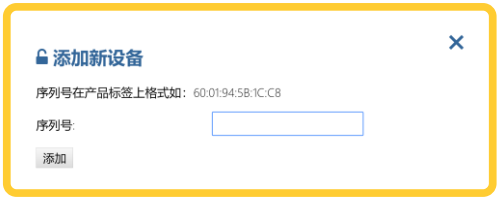
图形编程区：通过拖拽拼搭图形指令的方式编写程序的区域。

字符编程对照区：与图形程序对应的 C 语言程序，图形编程模式下不可修改，只可查看。

更详细的功能和操作说明详见附录五。

**绑定设备：**

点击“设备列表”右侧的“添加”按钮，添加新设备。


注意：输入 MAC 地址时，请将输入法切换到英文大写状态。

**编写第一个程序——点阵屏显示文本**

进入编程界面后，在图形程序编辑区中自动出现了“初始化”和“重复指令”两个图形指令，这两个图形指令是每一个程序所必需的。

**认识新指令——初始化（控制）**


初始化在程序中的作用一般是用于设定硬件启动时初始状态的，在程序开始后率先运行一次，之后不再运行。

初始化图形指令必须有且仅允许有一个，如果程序对初始状态无要求可以为空，即初始化图形指令内任何其他指令。

**认识新指令——重复执行（控制）**


重复执行指令的作用是让其内部的程序不断重复的运行，永无止境。在 WU-Link 硬件编程中，重复执行指令内部的程序被称为**主程序**。

主程序是一直重复运行的，位于“重复执行”指令之中。重复执行指令在程序中有且只

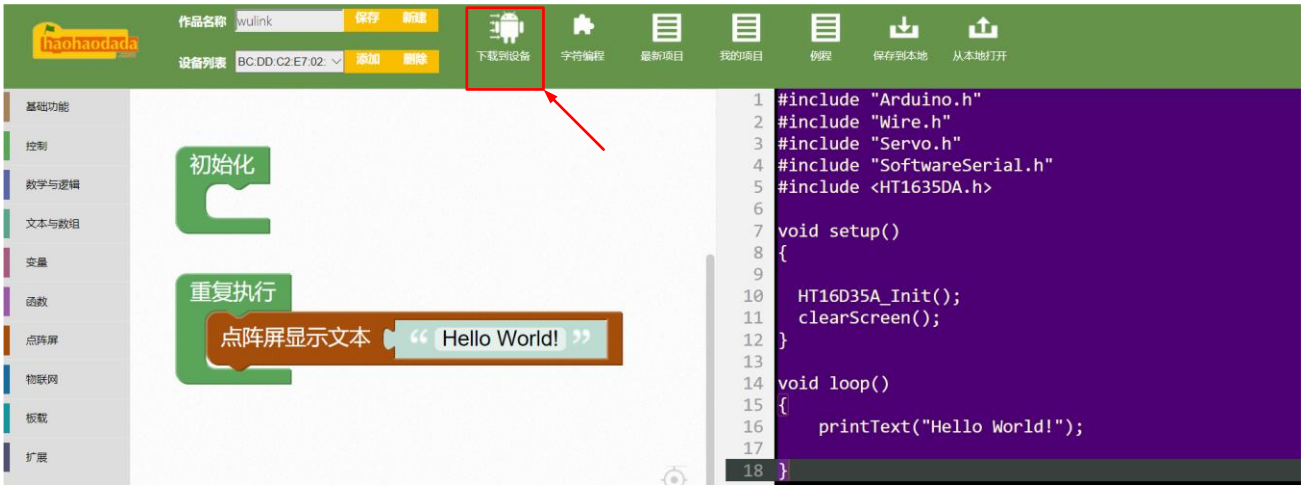
允许有一个。

**认识新指令——点阵屏文本显示指令（点阵屏）**


该指令可以设置 WU-Link 点阵屏显示指定的字母；默认显示“YES”，可以根据需要输入。由于 WU-Link 一屏只能显示四个字母，当输入字母数超过四个，就会以滚动方式显示。

**下载程序：**

点击 “下载到设备”按钮，将程序下载到 WU-Link 中。




**完成！**WU-Link 在向世界说你好！。

WU-Link 完整使用流程：


http://www.haohaodada.com/wu

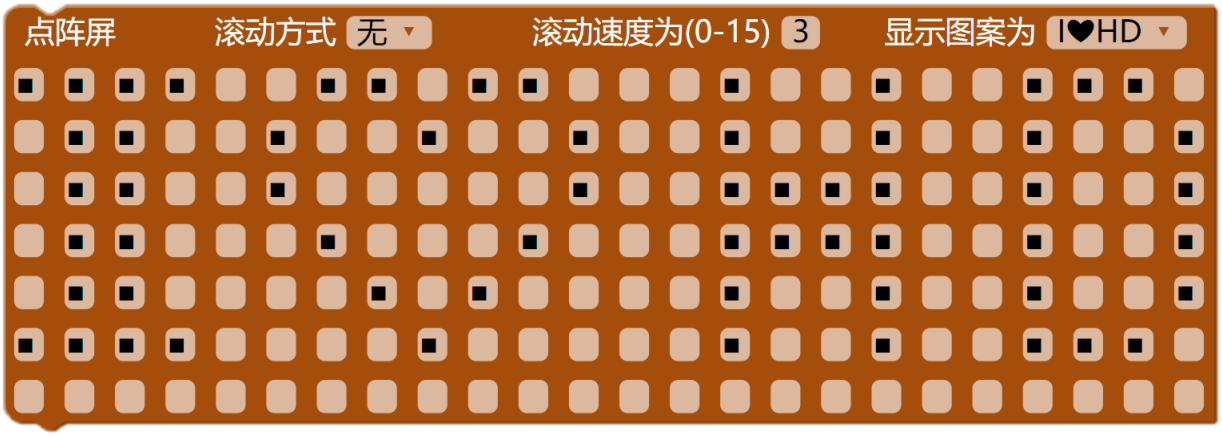
本教程后续内容仅对“搭建电路”和“编写程序”做详细说明。

# 第二节 玩转点阵屏

点阵屏在生活中随处可见，商业中心的外墙大屏幕、商店招牌、火车站候车信息等等。WU-Link 上集成有一个 7×24 的点阵屏，可以用来显示图案、文本和数字。

**示例 2-1：点阵屏绘制图案**

**认识新指令——点选设置点阵屏显示（点阵屏）**

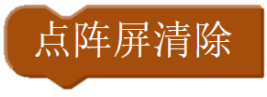



使用该指令可以设置 WU-Link 板载 LED 点阵屏显示图案。用鼠标单击指令中间的显示区，显示黑色■的格子表示选中、相应位置的 LED 被点亮；再次单击黑色■消失、相应位置的 LED 熄灭。

指令上方是图案显示属性设置区域；主要有以下几项设置内容：

- “滚动方式”设置：默认为“无”，单击下拉列表可以选择“↑”、“↓”、“←”、“→”四种滚动方向；
- “滚动速度”设置：默认是“3”，可以在方框内输入其它数值，数值的取值范围是（0，15），数值越小，滚动速度越快。
- “显示图案”设置：默认是“I♥HD”，可以单击下拉列表选择“雪花”、“全选”、“清零”。

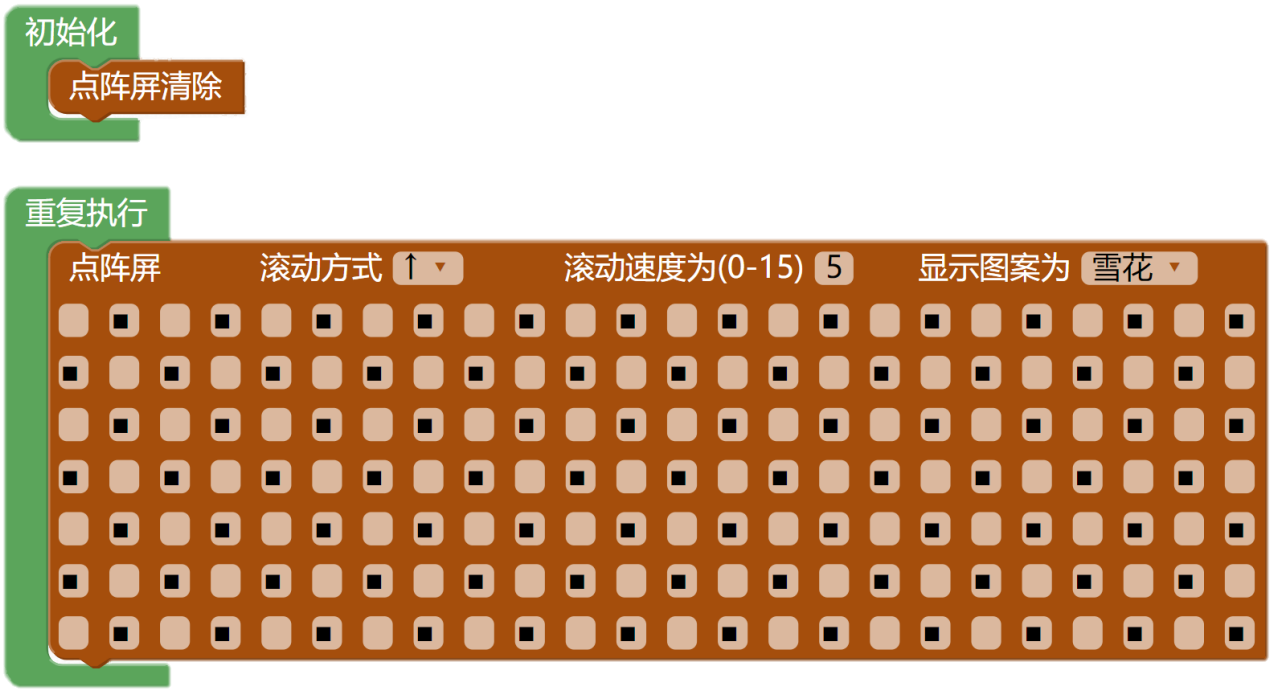
**认识新指令——点阵屏清除指令（点阵屏）**




使用这个指令可以清除 WU-Link 点阵屏原来的显示内容，为显示新的内容做准备。

当用到点阵屏时，通常会在初始化程序中加入一条点阵屏清除指令，以清除之前可能存留的显示信息。

**程序编写:**

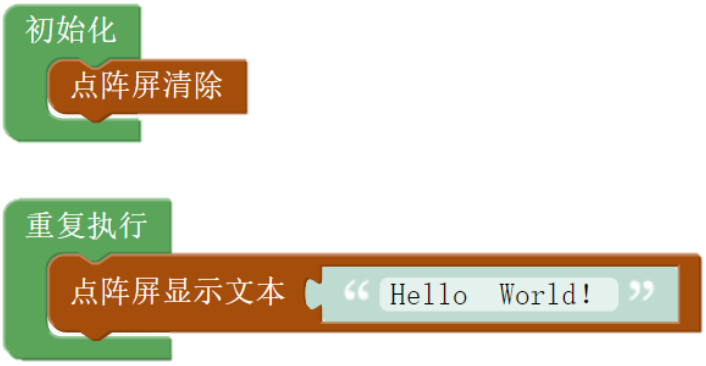



**示例 2-2：点阵屏显示文本**

**认识新指令——点阵屏文本显示指令（点阵屏）**


该指令可以设置 WU-Link 点阵屏显示指定的字母；默认显示“YES”，可以根据需要输入。由于 WU-Link 一屏只能显示四个字母，当输入字母数超过四个，就会以滚动方式显示。

**程序编写:**




点阵屏文本显示指令中的文本仅支持英文字母和数字，标点符号的输入也必须在英文输入法下进行，如上图程序中的感叹号。

**示例 2-3：点阵屏显示数字**

**认识新指令——点阵屏数值显示指令（点阵屏）**


该指令可以设置 WU-Link 点阵屏显示指定的数；默认显示“123”，可以根据需要输入。由于 WU-Link 一屏只能显示四个数字，当输入数字超过四个，会以滚动方式显示。

**程序编写:**


点阵屏数值显示指令可以用来读取各种传感器的数值，是一条使用频率非常高的指令。

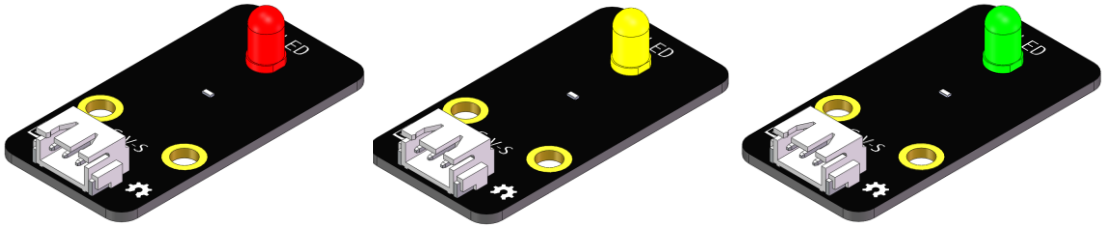
# 第三节 数字量输出及其控制

WU-Link 底部有外接扩展电子模块的接口，实现丰富的应用。这节课将用几个 LED 的案例让大家理解什么是数字量。

**示例 3-1：点亮 LED**

用 WU-Link 点亮一盏 LED。

**认识新硬件——LED 模块：**



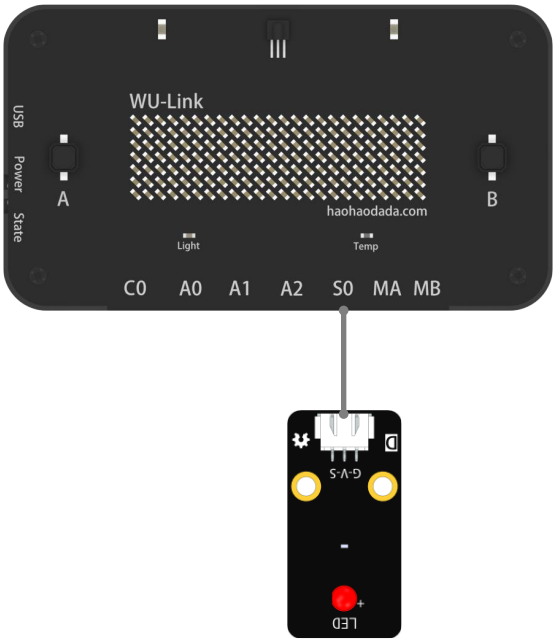
**LED 发光二极管：**由含镓（Ga）、砷（As）、磷（P）、氮（N）等的化合物制成。具有单向导通性，即 LED 有电流正向流入能点亮，反向流入或无电流则不亮。

**元器件列表：**

WU-Link 主控板 ×1

LED 模块（红） ×1

**电路连接：**



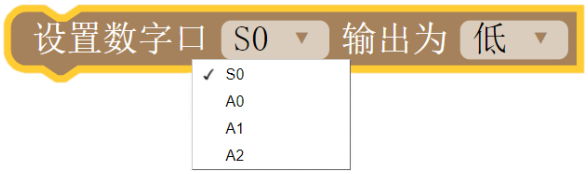

**认识新指令——设置数字量输出指令（基础功能）**

所以需要使用数字量相关指令，对于主控板 WU-Link 来说，点亮 LED 是由 WU-Link 输出

信号给 LED，因此使用“设置数字口…输出为…”指令：


**设置端口**

点击“S0”，弹出下拉菜单，选择数字口：




选择端口的原则是：**与硬件连接一致！**本例中 LED 模块是连接到 WU-Link 的 S0 接口，所以指令选择“S0”。

输出可选择“低”或“高”，指的是输出为“低电平”或“高电平”，低电平关灭 LED，高电平点亮 LED，所以本例中选择“高”


**程序编写：**


**示例 3-2：LED 的闪烁**

**元器件列表：**

同示例 3-1

**电路连接：**

同示例 3-1

**认识新指令——延时指令（控制）**

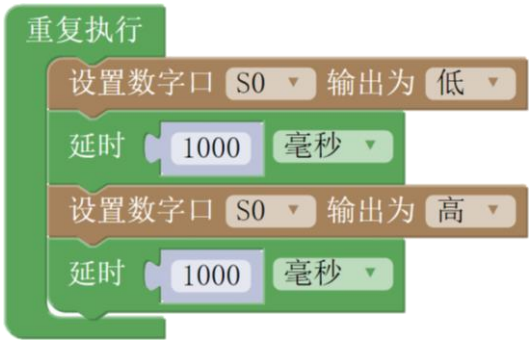

延时指令的作用是让程序暂停执行一段时间，时间的长短可以设置，单位可选择毫秒或微秒。

**程序分析:**

我们来仔细思考下“闪烁”这个词，闪烁是灯亮一段时间，之后灭一段时间，如此循环往复。那么，“闪烁”应该被拆分成“亮”、“灭”、“持续一段时间”这些动作的组合。其中，“亮”和“灭”可以通过“设置数字口”指令实现：


“持续一段时间”需要使用延时指令。

**程序编写:**




**示例 3-3：交通灯（红绿灯）**

**元器件列表：**

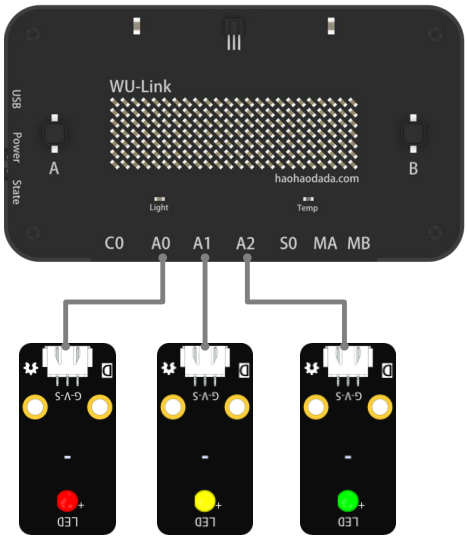
WU-Link 主控板 ×1

LED 模块（红） ×1

LED 模块（黄） ×1

LED 模块（绿） ×1

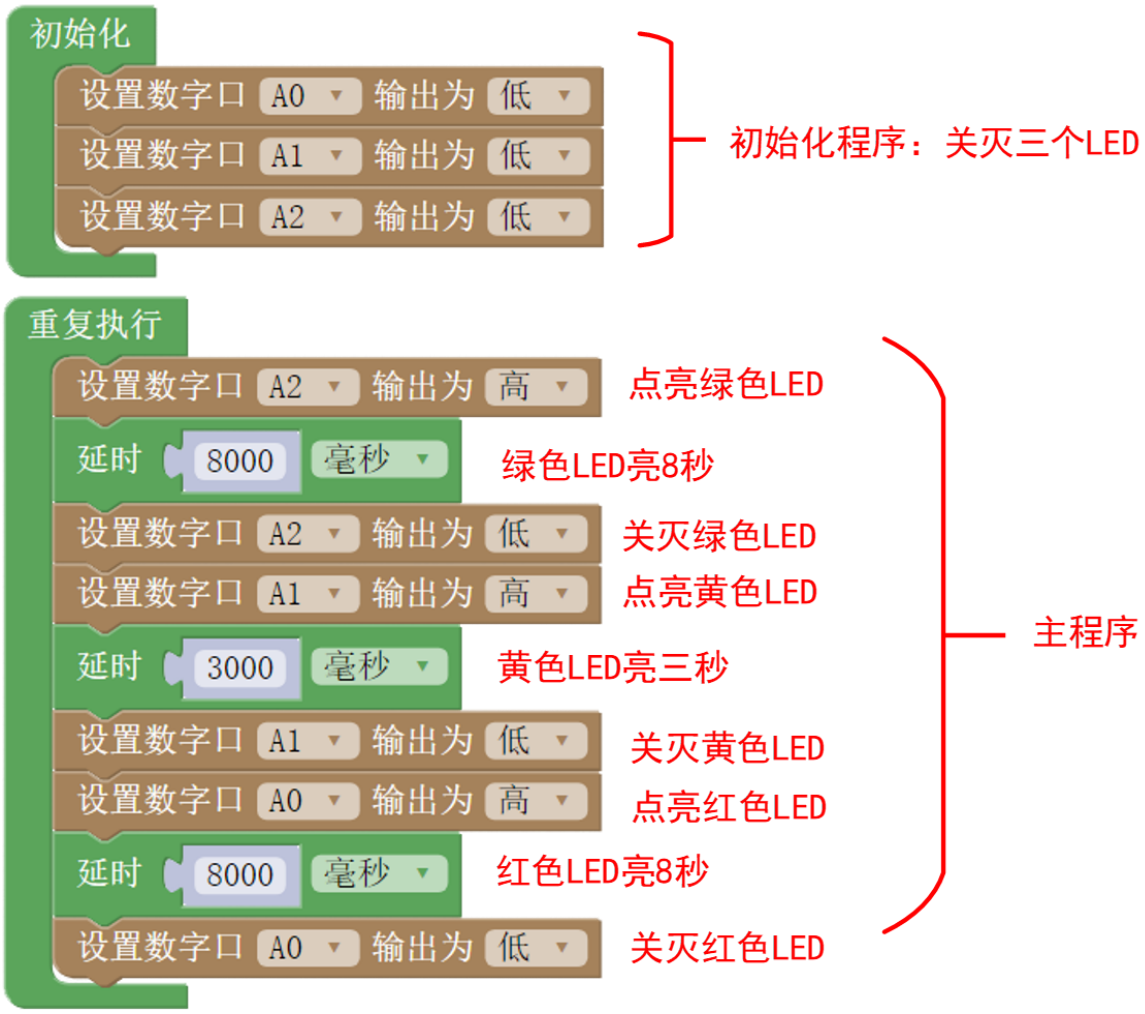
**电路连接：**



**程序分析：**

同学们可以去仔细观察或回想一下交通路口的红绿灯，它们是如何运作的：红黄绿三盏灯依次亮灭，各持续一段时间。程序的编写要点是用不同的端口控制不同的 LED 模块，实现依次亮灭的目的。

**程序编写：**



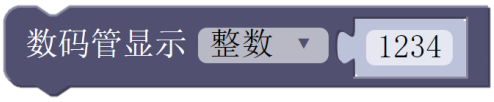
# 第四节 数字显示屏

右图这种计算器相信大家都不陌生，这种液晶显示屏只能显示数字，简称为**数码管**。相对于电视或手机屏幕，这种显示屏使用上更加简单，当创客项目中仅需要显示数字而不需要显示文字、图片信息时，可以使用数码管来简单快速的实现。

**认识新硬件——数码管**

数码管模块可以用于显示数字，最多显示 4 位数字，只能连接到 WU-Link 上的 C0 接口。

**认识新指令——数值显示指令（扩展）**




整体数字显示指令可以方便的将四位以内的数字显示到数码管上。显示整数选择“整数”模式，显示带小数位的数字（浮点数）选择“小数”模式。

**认识新指令——数码管清除指令（扩展）**


数码管清除指令用于将数码管的显示全部清除，即将数码管上所有的 LED 关灭。

**示例 4-1：用数码管显示数字**

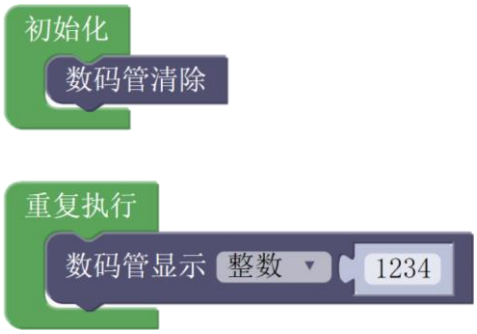
**元器件列表：**

1. WU-Link 主控板 ×1
2. 数码管模块 ×1

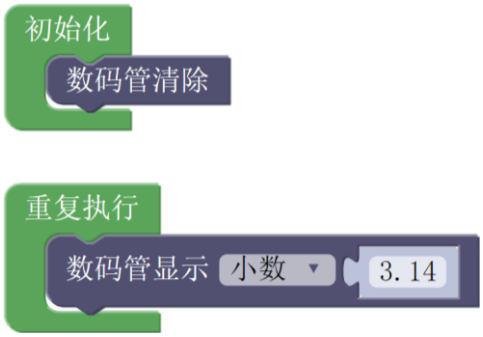
**电路连接：**


**Haohaodada 程序编写:**

**显示一个整数：**




**显示一个小数：**



# 第五节 模拟量输入与采集

本次课将带大家认识**传感器**，传感器是一种检测装置，能感受到被测量的信息，并能将感受到的信息，按一定规律变换成为电信号或其他所需形式的信息输出，以满足信息的传输、处理、存储、显示、记录和控制等要求。

从传感器输入的数值看，一种是只有两个取值“0”和“1”的传感器，称为数字量传感器；另一种是有一个取值范围，如 0 到 100、0 到 1023，称为模拟量传感器。

**数字量传感器**

**详见第十一课《数字量输入与条件判断指令》**

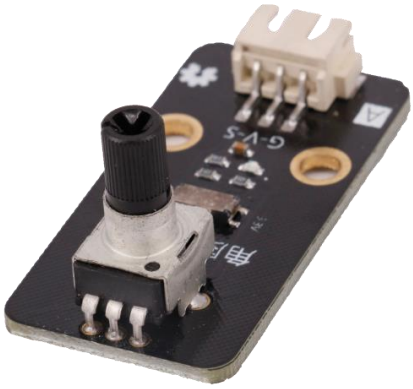
**模拟量接口传感器**

模拟量是指在一定范围连续变化的量，比如温度、亮度、距离、重量等。

模拟量传感器就是采集这类物理量，并将其转换为电压或电流信号的传感器。

**认识新模块和新指令：**

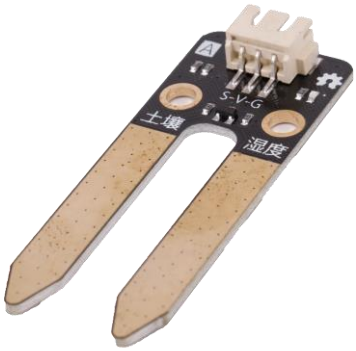
**电位计模块：**一种旋转变阻器，可以用作旋钮，可以感知角度的变化。




**亮度传感器模块：**用来检测当前的光照强度。

**声音传感器模块：**用来检测当前环境的声音强度。


**土壤湿度模块：**用来检测土壤的湿度情况。




以上四种模块都是模拟量传感器模块，可以直接用基础功能类目中的“读模拟量”指令读取：


主控板上能读取模拟量传感器的接口是 A0~A2。

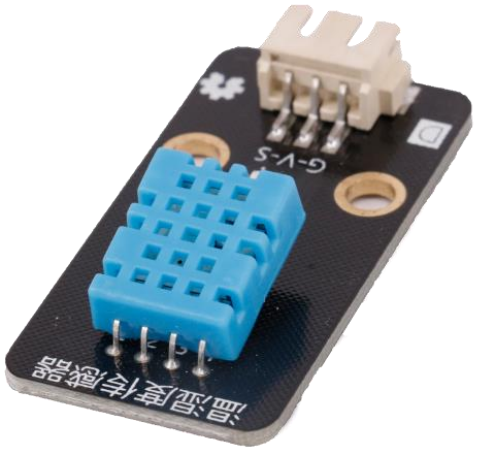
模拟量传感器模块输送到主控板中的是电压值，而不是物理量常用的单位如摄氏度。

另外还有一类传感器，它测量的是模拟量如距离、温度、湿度，但是输送给主控板的信号不是电压值，所以不能用读模拟口指令读取数值。好好搭搭为这类传感器设计了专用的指令。

**超声波测距传感器：**通过发射接收超声波来测量超声波模块与障碍物距离。超声波测距传感器共设计够两种，图片和对应的指令如下：


**温湿度传感器：**用来检测当前环境的温度和湿度。下图中的蓝色塑料块即是温湿度传感器的核心器件——DHT11，所以对应的指令名称为“读 DHT11”。

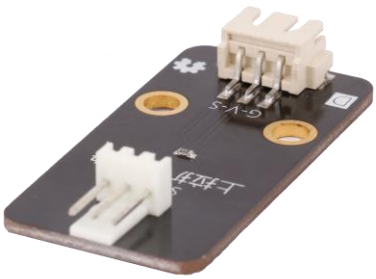
注意不能直接放入水中测量水温！





用这些传感器配合 WU-Link 的点阵屏或数码管可以制作各式各样的测量仪表，如超声波测距仪、光照强度仪、噪音计

**DS18B20 温度传感器：**温度传感器是含有 DS18B20 探测器的金属管温度计，抗干扰能力强，精度高且外部有橡胶管能防水。测量温度范围:-55℃+125℃，该传感器不能直连 WU-Link，需先连接上拉模块，再连接 WU-Link。



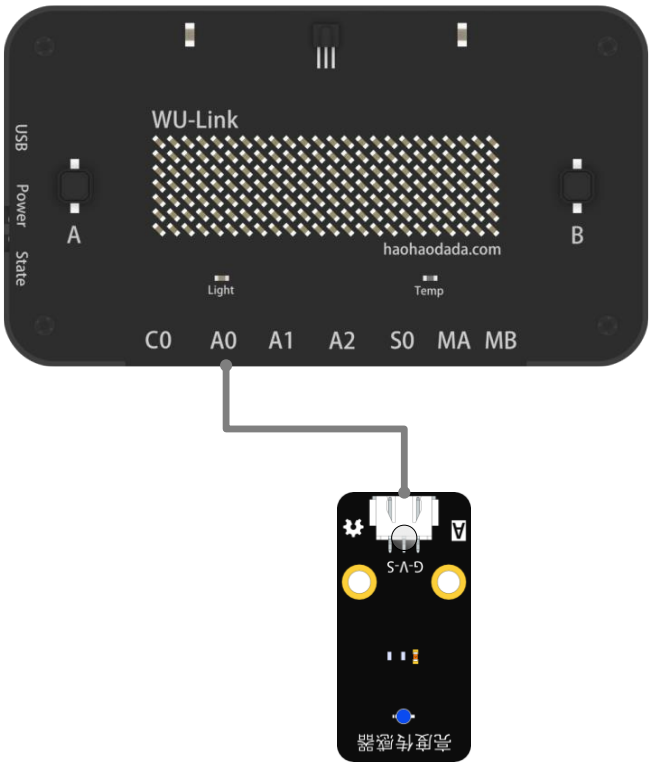
**示例 5-1：环境光检测仪**

在 WU-Link 点阵屏上实时的显示环境光的数值

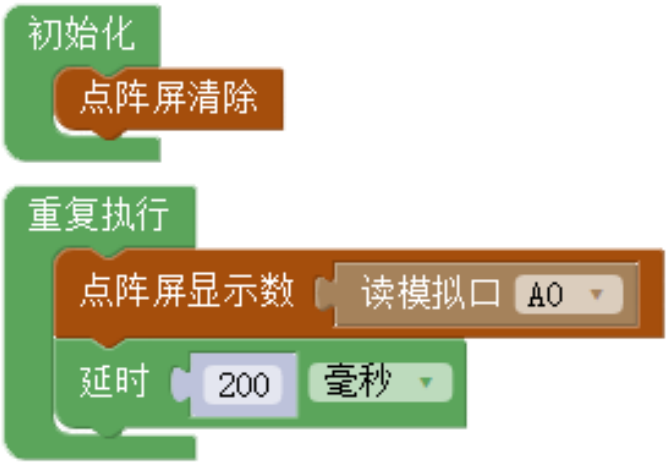
**元器件列表：**

1. WU-Link 主控板×1
2. 亮度传感器模块×1

**电路连接：**




**程序编写：**




以上程序即可实现环境光的检测，同学们试着测量各种情况下的亮度值。

| 环境情况 | 显示数值 |
| --- | --- |
| 白天房间里 | |
| 用阴影遮挡 | |
| 直接盖上 | |
| 在灯光下 | |
| 在阳光下 | |

将传感器换成声音传感器，程序不变，即可制作环境声检测仪。同样，同学们也试着测量下各种环境中的声音响度值吧

| 环境情况 | 显示数值 |
| --- | --- |
| 比较安静的房间里 | |
| 2 人正常对话 | |
| 大声喊叫 | |
| 教室里 | |
| 上体育课的操场上 | |

**示例 5-2：超声波测距仪**

在 WU-Link 点阵屏上实时的显示测量的距离值

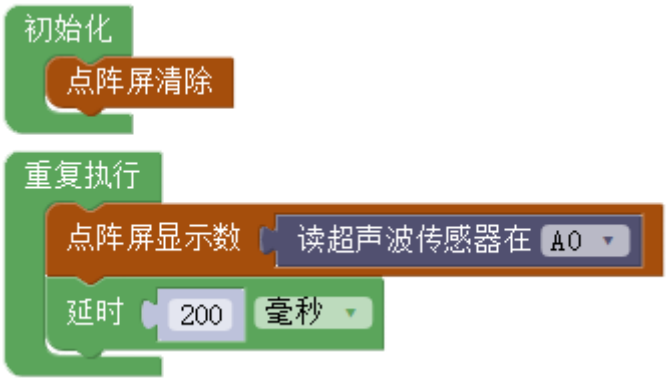
**元器件列表：**

1. WU-Link 主控板×1
2. 超声波测距模块×1

**电路连接：**


**程序编写:**

这里需要用超声波测距传感器模块的专用指令。



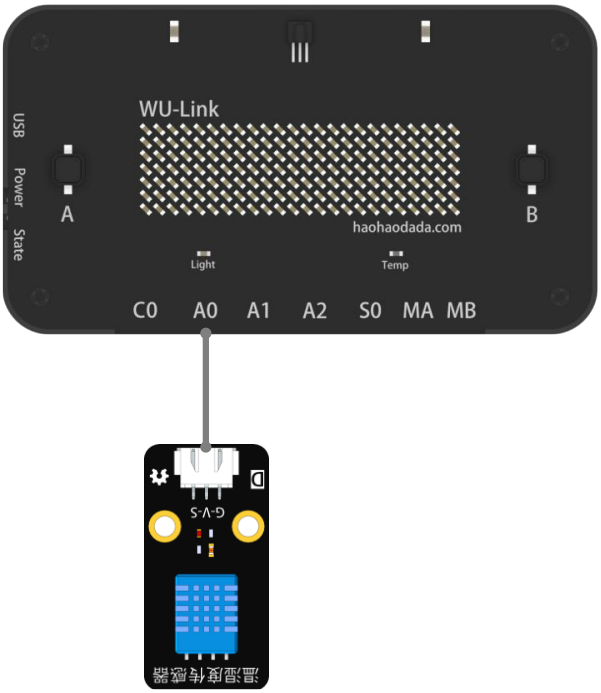

**示例 5-3：温湿度计**

用数码管实时的温度值/湿度值

**元器件列表：**

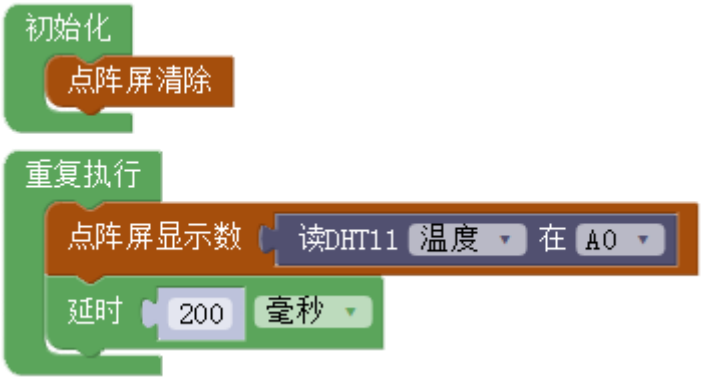
1. WU-Link 主控板 × 1
2. 温湿度传感器 × 1

**电路连接：**




**程序编写：**

这里需要用温湿度传感器的专用指令——读 DHT11。



# 第六节 模拟量的输出

在第一、三课中同学们已经掌握控制 LED 亮灭的方法，那么 LED 除了开（完全点亮）和关（完全熄灭）两种状态之外，是不是能做到 LED 亮了，但是比完全点亮暗一些呢?

手机、平板可以说是大家使用最多的电子设备，大家会在晚上关灯后调低显示屏的亮度以保护眼睛，在白天强光下调高亮度以便看的更清楚。这些显示屏的背光光源都是 LED，LED 的调光是如何实现的呢？先从 LED 的闪烁说起。

**示例 6-1：LED 调光**

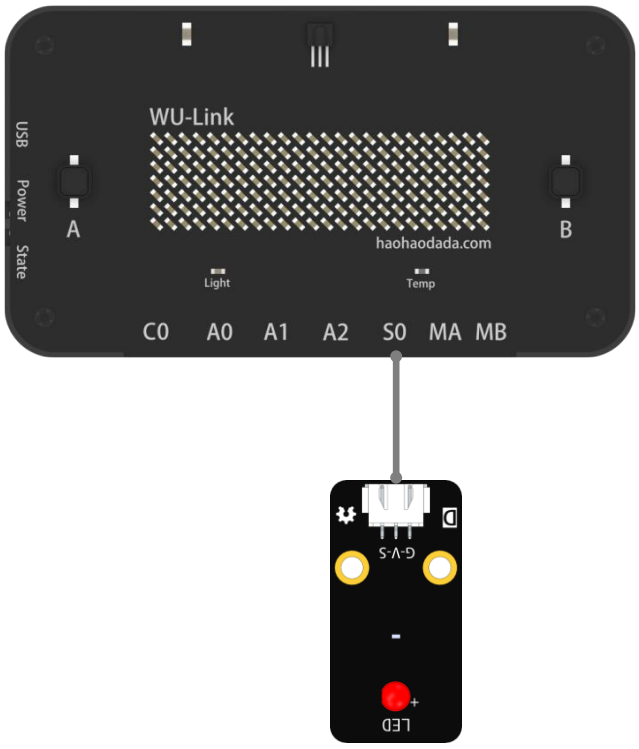
在编程实现 LED 的调光之前，同学们可以先回顾下示例 2-1：LED 的闪烁。

**元器件列表：**

WU-Link 主控板 × 1

LED 模块 × 1

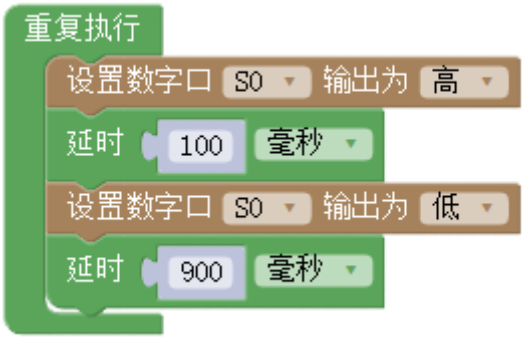
**电路连接：**




LED 灯闪烁的程序已经实现，接下来做几个小实验：

**小实验 1：LED 灯亮 100 毫秒，LED 灭 900 毫秒**

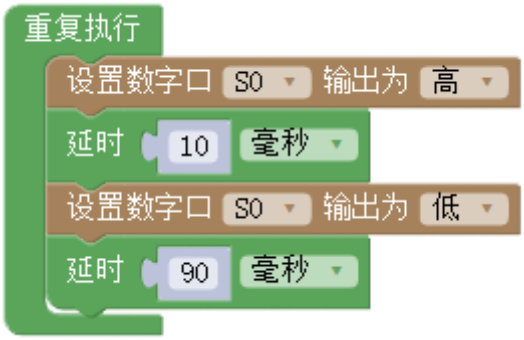
程序：（初始化程序为空，图中省略）




实验 1 直观结果：LED 闪烁，熄灭时间比亮起时间长。

**小实验 2：LED 灯亮 10 毫秒，LED 灭 90 毫秒**

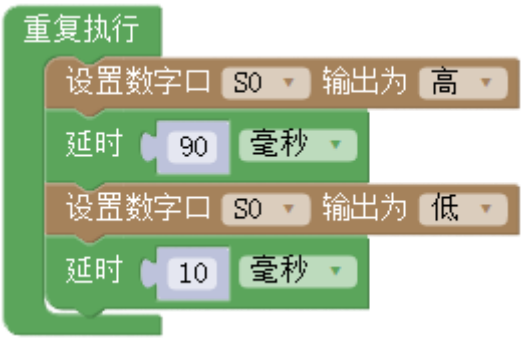
程序：（初始化程序为空，图中省略）




实验 2 直观结果：LED 快速闪烁，注意与实验 3 的直观结果做比较。

**小实验 3：LED 灯亮 90 毫秒，LED 灭 10 毫秒**

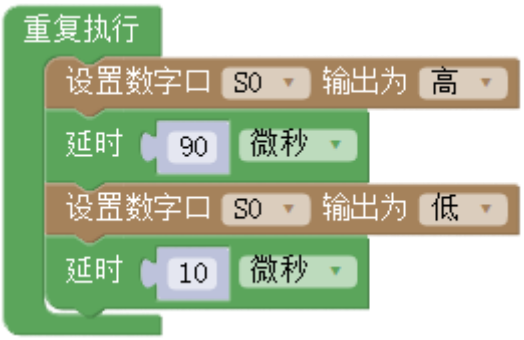
程序：（初始化程序为空，图中省略）




实验 3 直观结果：LED 快速闪烁，看上去比实验 2 亮多了。

**小实验 4：LED 灯亮 90 微秒，LED 灭 10 微秒**

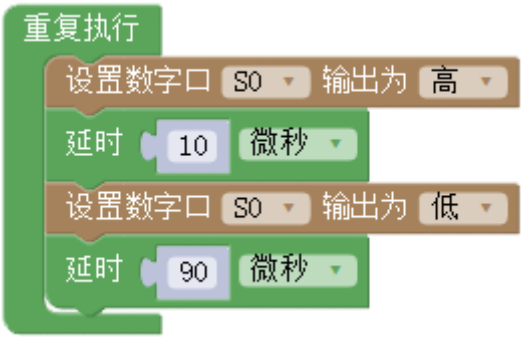
程序：（初始化程序为空，图中省略）




实验 4 直观结果：LED 看上去完全不闪了，注意与实验 5 直观结果做比较。

**小实验 5：LED 灯亮 10 微秒，LED 灭 90 微秒**

程序：（初始化程序为空，图中省略）



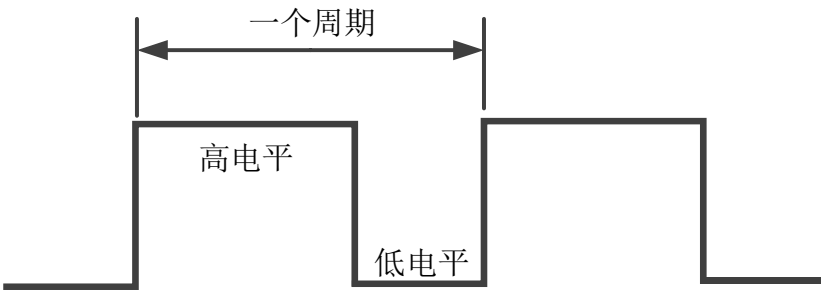

实验 5 直观结果：LED 看上去完全不闪，但是比实验 4 的亮度低很多。

大家可以再试试“0 100”“20 80”、“40 60”、“50 50”、“60 40”、“80 20”、“100 0”不同的组合，延时单位为微秒。可以看到这些参数组合的结果：亮度都不同。

**结论：LED 的调光是通过在一定的时间间隔内，调节点亮和熄灭 LED 的相对时间来实现的。**

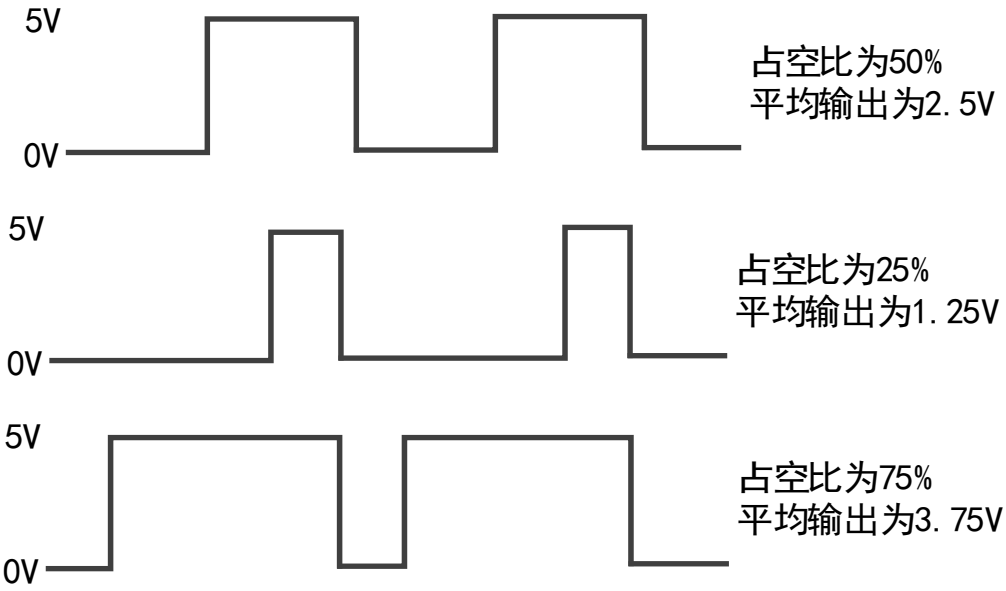
**认识新概念——PWM**

类似 LED 调光这种通过调节供电“通”和“断”的相对时间来改变平均输出大小方法，在工程上称为 PWM（脉冲宽度调制）。




其中，一个周期中高电平持续时间与低电平持续时间的比值，称为占空比。

如高电平电压为 5V，占空比 50%的 PWM 信号的平均电压为 2.5V；占空比 25%的 PWM 信号的平均电压为 1.25V；占空比 75%的 PWM 信号的平均电压为 3.75V。




因为 WU-Link 主控器运行程序是单线程的，如果在复杂程序里要调光，运行其他部分程序的时间也会被计算到点亮或熄灭 LED 的延时时间中去，导致亮度调节不准确。所以在复杂程序中需要实现调光功能时，需要调用 **PWM 输出指令**。

**认识新指令——设置 PWM 输出指令（基础功能）**


PWM 输出指令能调用主控板的内部定时器，来实现平均输出的调节，无需使用延时指令。WU-Link 主控板上能输出 PWM 信号的端口是 S0。

PWM 输出指令的数值范围为 0 到 255，输出数值为 0 时，输出电压为 0V，即一直为低电平，相当于数字口输出设置为 0；输出数值为 255 时，输出电压为 5V，即一直为高电平，相当于数字口输出设置为 1。

利用 PWM 输出指令来设置 LED 的亮度的程序为：


同学们试试不同的输出值，看看 LED 的亮度变化。

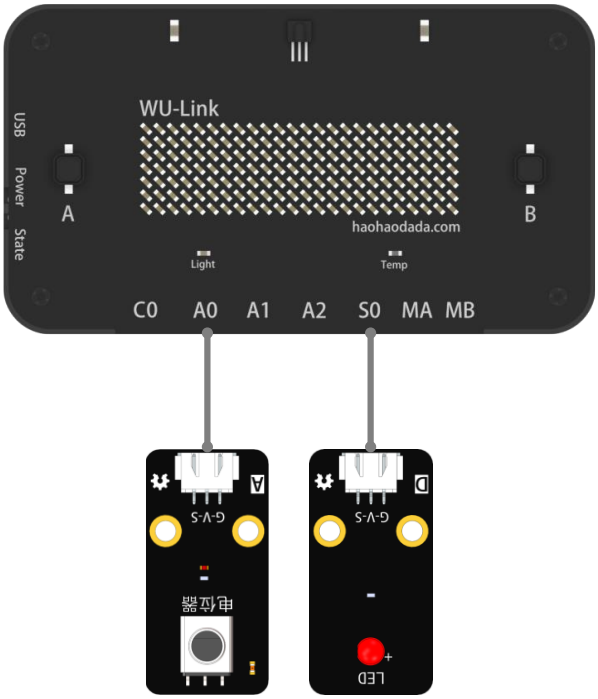
**示例 6-2：旋钮调光**

通过前面的内容，了解了模拟量输入和模拟量输出都是范围取值，那么如何用模拟量输入控制模拟量输出呢？

**元器件列表：**

1. WU-Link 主控板 ×1
2. 电位器模块 ×1
3. LED 模块 ×1

**电路连接：**



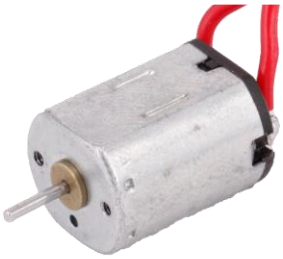

**程序编写：**


其中电位器的取值范围为 0 到 4095，而 PWM 输出的取值范围为 0 到 255，所以从电位器模块读取到的模拟值需要经过换算，让其最大值不超过 255，最小值不小于 0。

# 第七节 电机的驱动

**电动机**是将电能转换为机械能的一种执行器，通常简称为**电机**。让机构旋转起来，最简单直接的方法就是使用电机。

**认识新模块——电机**



N20 电机（直流电机）

**认识新指令——电机输出指令（扩展）**

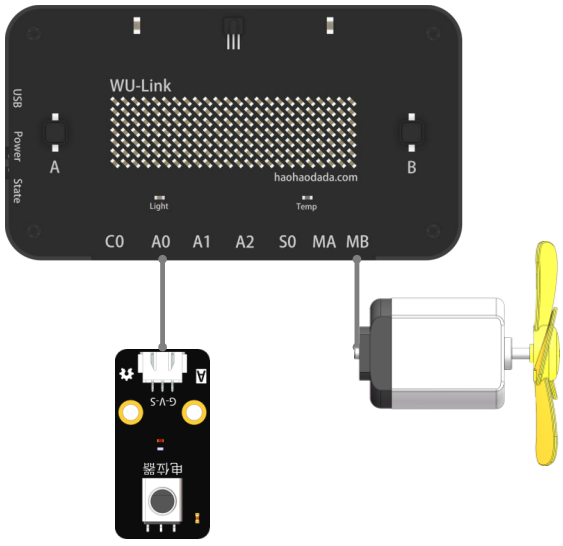

该指令的数值输出范围均为-255 到 255。能为电机提供驱动信号的端口为 MA、MB。

**示例 7-1：可调速电风扇**

**元器件列表：**

1. WU-Link 主控板 × 1
2. 电位器模块 ×1
3. N20 电机（装风扇叶片） × 1

**电路连接：**



程序编写

# 第八节 蜂鸣器

蜂鸣器模块用于发出一定频率的电子声。我们用它编制一首乐曲吧!

声音的三个主观属性分别是音量（响度)、音调和音色。音量指人耳感受到的声音强弱；音调指人的听觉能分辨一个声音的调子高低的程度；音色指声音的感觉特性，即根据不同的音色，即使在同一音高和同一声音强度的情况下，也能区分出是不同乐器或人发出的。

蜂鸣器的驱动电流（PWM 占空比）决定了蜂鸣器发声的音量；蜂鸣器的工作频率（PWM 频率）决定了蜂鸣器发声的音调；蜂鸣器的内部结构与发声原理决定了蜂鸣器发声的音色。

**认识新模块——蜂鸣器**

**认识新指令——设置 PWM 频率指令（基础功能）**

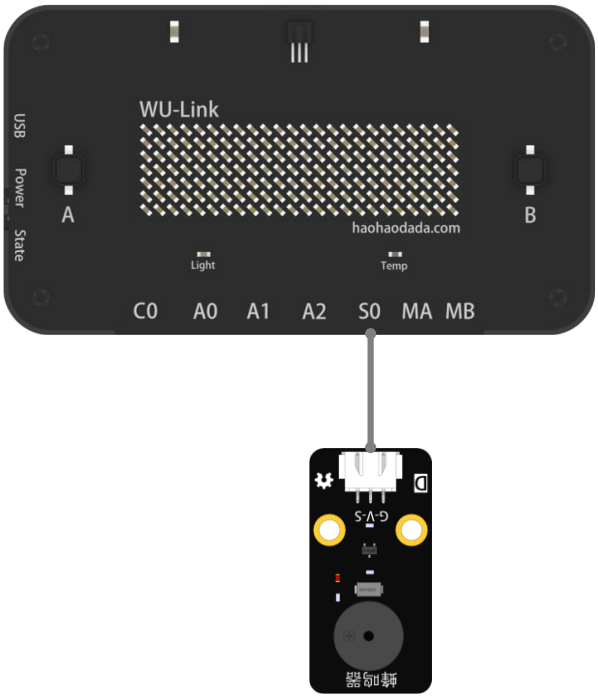

设置 PWM 频率指令是用于修改 PWM 周期的指令，从而实现修改蜂鸣器的音调。

**示例 8-1：两只老虎**

**元器件列表：**

1. WU-Link 主控板 ×1
2. 蜂鸣器模块 ×1

**电路连接：**




**程序编写:**

**蜂鸣器鸣响程序范式：**


**发出 C 调音符**

我们听到的音乐，每个音符都有固定的频率，比如 C 调音符相对应的频率如下图所示：

| C 调音符 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 频率 | 262 | 293 | 329 | 349 | 392 | 440 | 494 |
| C 调音符 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 频率 | 523 | 586 | 658 | 697 | 783 | 879 | 987 |
| C 调音符 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 频率 | 1045 | 1171 | 1316 | 1393 | 1563 | 1755 | 1971 |

《两只老虎》的简谱如下：


要播放一首乐曲，除了要发出确定的音调，还需要有节拍的配合。如上图中


两拍的时间长度是一拍的 2 倍，半拍的时间长度是一拍的一半。

一拍的时间长度没有固定的限制，可以是 0.5 秒也可以是 2 秒，时间越短节奏越快。那么在程序中通过设置蜂鸣器模块的“持续时间”来实现某个音的拍数

那么第一节的蜂鸣器程序为：

接下来开始编写曲子吧！

**认识新指令——板载蜂鸣器设置指令（板载）**

WU-Link 主控板内置有一个蜂鸣器，在开机、联网、下载程序时都能听到它发出的声音，它也可以由用户编程来发出声音，相关图形指令位于板载类目中：


设置板载蜂鸣器频率指令


设置板载蜂鸣器音调指令

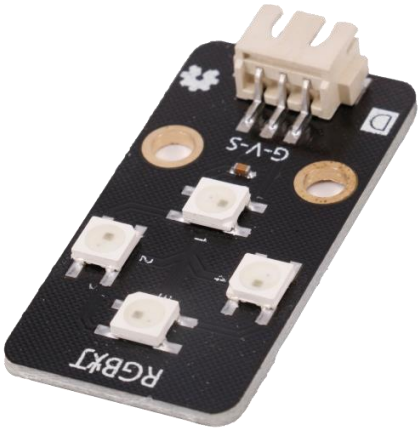

设置板载蜂鸣器音量指令

# 第九节　RGB 七彩灯

前面课程用到的 LED，只能发出固定颜色的灯光，当需要发出更多颜色的光时，可以使用 RGB 模块。很炫哦!

RGB 色彩模式是工业界的一种颜色标准，是通过对红(R)、绿(G)、蓝(B)三个颜色通道的变化以及它们相互之间的叠加来得到各式各样的颜色，这个标准几乎包括了人类视力所能感知的所有颜色，是目前运用最广的颜色系统之一。

**认识新模块——RGB 模块**



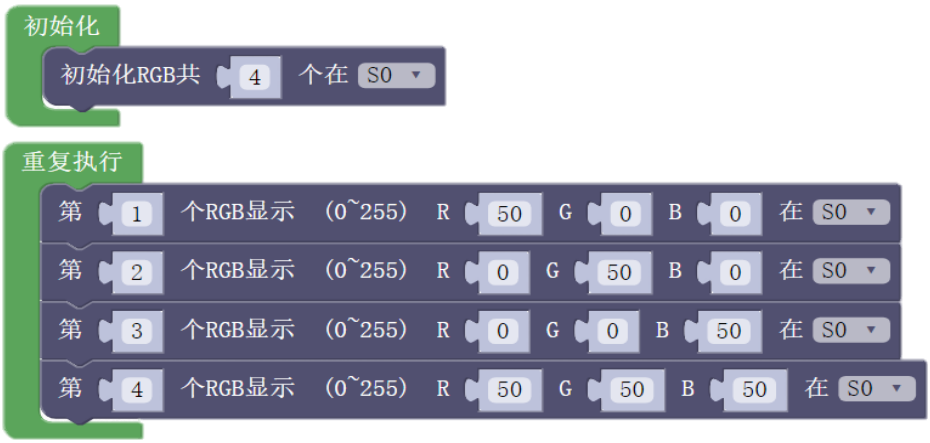

**认识新指令——RGB 初始化指令（扩展）**


RGB 初始化指令必须放在初始化程序中，用于告知 WU-Link 主控板接入的 RGB 模块上有几个 RGB 灯。套件中的 RGB 模块上有 4 个 RGB 灯，所以该指令参数默认为 4，无需修改，如果接入的 RGB 模块上有 6 个 RGB 灯，则需将参数改为 6。

**认识新指令——发送 RGB 数据指令（扩展）**


发送 RGB 数据指令是四个参数框，第一个参数框填入要点亮的 RGB 灯的编号；第二个至第四个参数框分别填入红（R）、绿（G）、蓝（B）三种颜色灯的参数值，允许填入的数值范围为 0-255。

**RGB 模块点亮程序范式：**



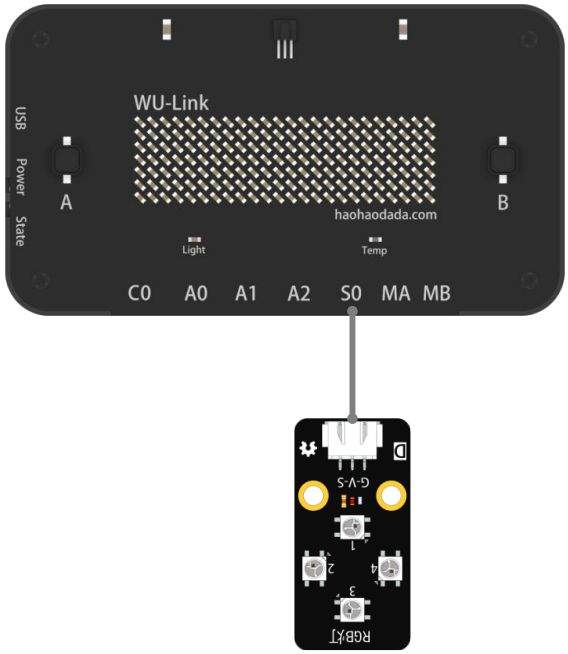

**示例 9-1：RGB 交通灯**

用 RGB 模块实现交通灯的红黄绿灯变换。

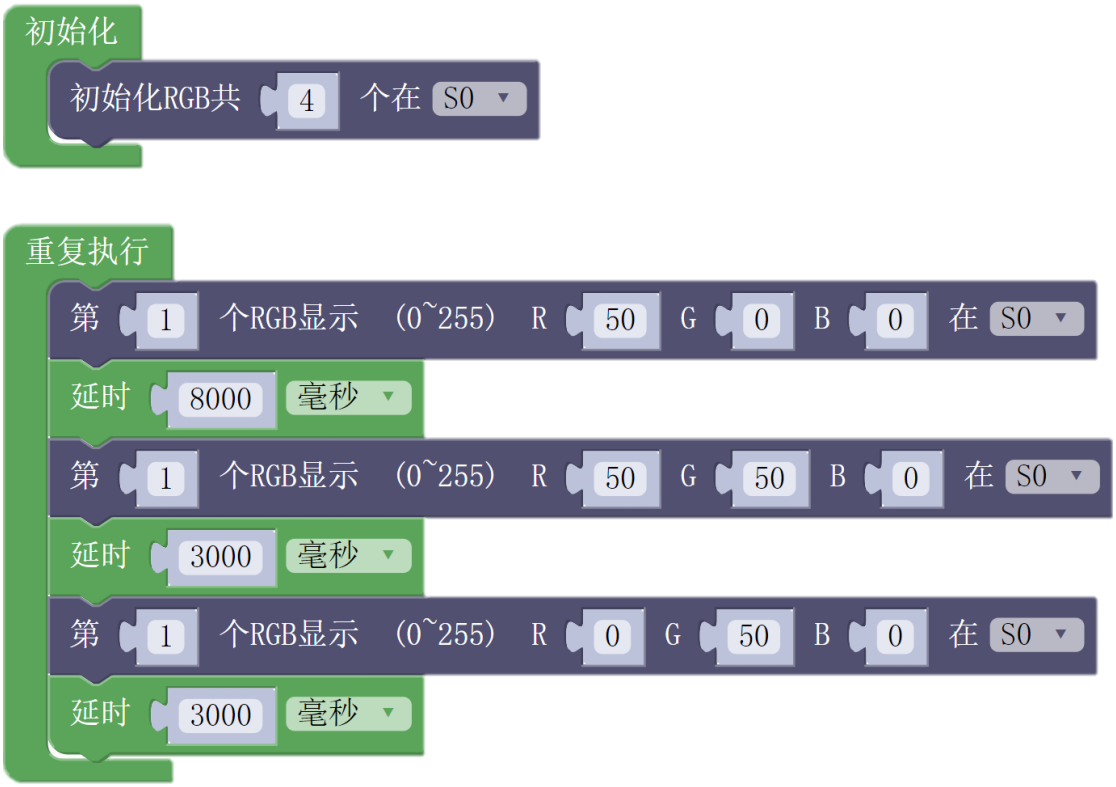
**元器件列表：**

1. WU-Link 主控板 ×1
2. RGB 模块 ×1

**电路连接：**




**程序编写：**


初始化
初始化RGB共 4 个在 S0
重复执行
第 1 个RGB显示 (0~255) R 50 G 0 B 0 在 S0
延时 8000 毫秒
第 1 个RGB显示 (0~255) R 50 G 50 B 0 在 S0
延时 3000 毫秒
第 1 个RGB显示 (0~255) R 0 G 50 B 0 在 S0
延时 3000 毫秒

# 第二部分

# 编程入门

# 第十节 编程基础知识介绍

程序是一套人与计算机沟通的语言，既然是语言就有它一套独特的表达方式。

学习编程很像学习一门外语。然而，程序语言比任何一种人类语言都要古板，任何规范之外的表达方式都会被认为是错误的。这套规则来自一代又一代程序设计师的设定，随着时间的流逝，渐渐成为了计算机的行业规范。不了解程序语言的规范，是写不出正确程序的。

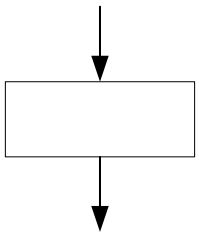
**程序流程图**

无论是代码程序还是图形化程序，当程序复杂到一定程度之后，都会变得难以读懂。工程上会利用各种图形来表达程序的结构和运行顺序，其中程序流程图是最简单最流行的一种。

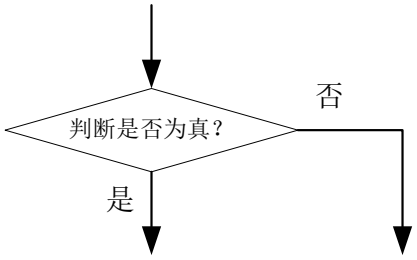
程序流程图中最主要的三种符号：


**路径**表示程序之间连接与流转关系，路径互相之间不能交叉。

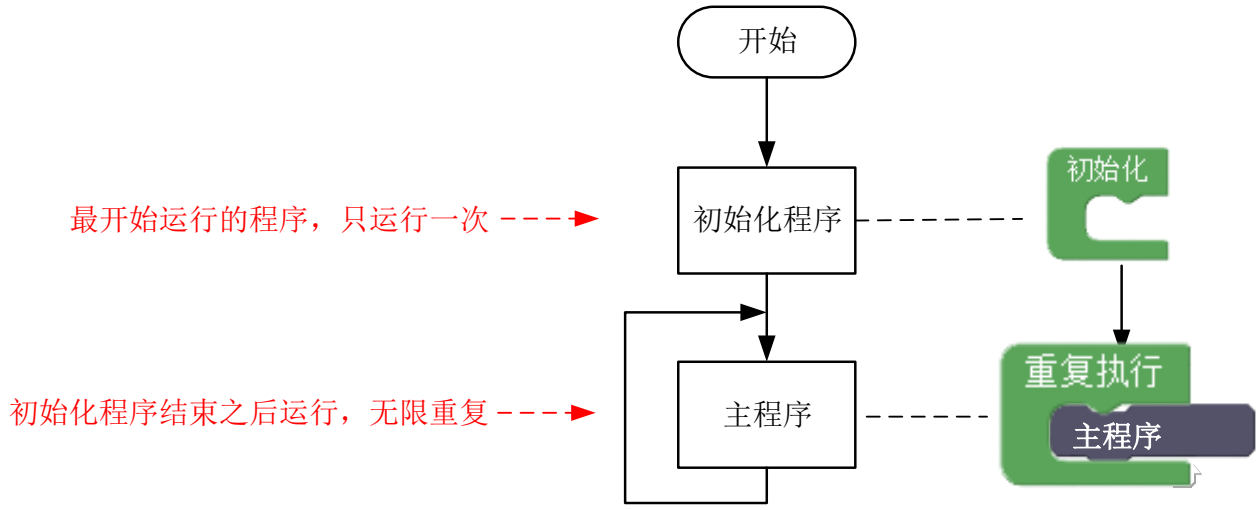
**执行框**代表程序中的一个处理或者步骤，它只能接一条流入路径和一条流出路径，如：



**判断框**是对一个条件进行判断抉择，它只能有一条流入路径，如果为“真”走一条流出路径，如果为“假”则走另一条流出路径，不可能同时往两条流出路径走，如：



WU-Link 的程序框架包括两部分：初始化程序和主程序。程序运行的流程如下：




**初始化程序**在程序中的作用一般是用于设定硬件启动时初始状态的，位于“主程序”之前，在程序开始后率先运行一次，之后不再运行。例如示例三交通灯程序中，初始化程序的作用就是使接通电源时，三个 LED 都处于熄灭状态。

**主程序**是一直**重复运行**的，位于“重复执行”指令之中。重复执行指令在程序中有且只允许有一个。

# 第十一节 数字量输入与条件判断指令

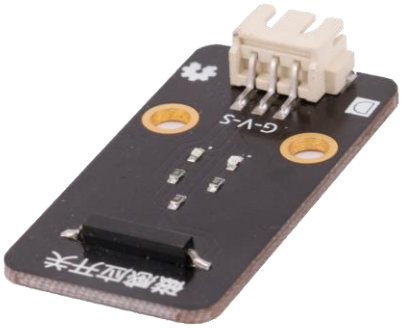
**数字量传感器**

数字量是只有“1”和“0”两种状态的量，比如：“是”与“否”、“开”与“关”、“真”与“假”，这些都是非此即彼的，都是数字量，也被称为开关量，只不过计算机系统只能处理数字信息，所以用“1”和“0”来指代。

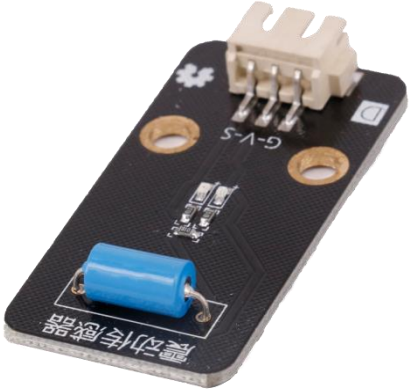
**认识数字量传感器**

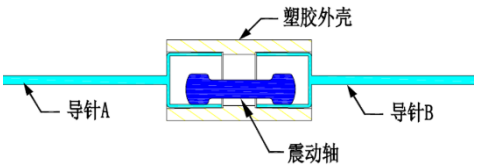
**按钮开关：**感知外界有否物体按压它。

**磁感应开关：**磁感应开关的敏感元件是干簧管，通常由两个软磁性材料做成的、无磁时断开的金属簧片触点。用于检测附近是否有磁性物体靠近。

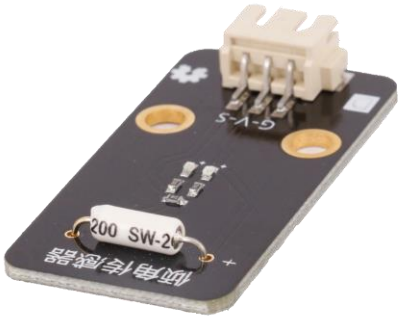


**震动开关：**没有振动时，震动轴呈静止状态，导针 A 与导针 B 两端则为接通状态，当有震动时，震动轴会运动，导针 A 与导针 B 之间会有瞬间的断开，实现震动触发的作用。





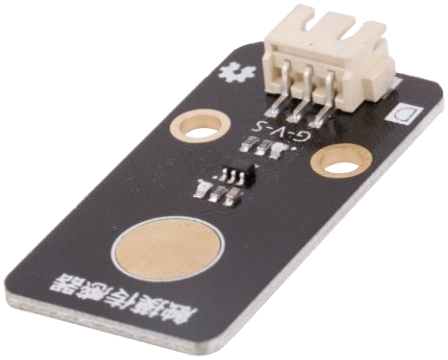
**倾斜开关：**垂直悬挂的倾斜开关探头在受到外力作用且偏离垂直位置 17 度以上时，倾斜开关内部的金属球触点动作，常闭触点断开。当外力撤消后，倾斜开关回复到垂直状态，金属球触点复又闭合。倾斜开关可以用于检测某个确定角度的状态。




**人体红外感应开关：**人体辐射的红外线中心波长为 9~10--um，而探测元件的波长灵敏度在 0.2~20--um 范围内几乎稳定不变。在传感器顶端开设了一个装有滤光镜片的窗口，这个滤光片可通过光的波长范围为 7~10--um，正好适合于人体红外辐射的探测，而对其它波长的红外线由滤光片予以吸收，这样便形成了一种专门用作探测人体辐射的红外线传感器。


**触摸开关：**这是一个基于电容感应的触摸开关模块。人体或金属在传感器金属面上的直接触碰会被感应到。

除了与金属面的直接触摸，隔着一定厚度的塑料、玻璃等材料的接触也可以被感应到，感应灵敏度随接触面的大小和覆盖材料的厚度有关。

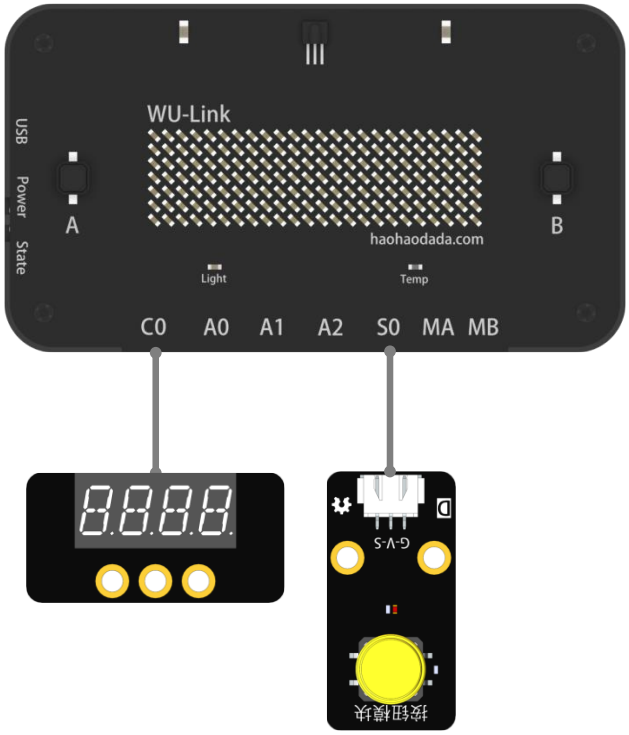


**示例 11-1：用点阵屏显示开关量传感器的状态，以按钮开关为例。**

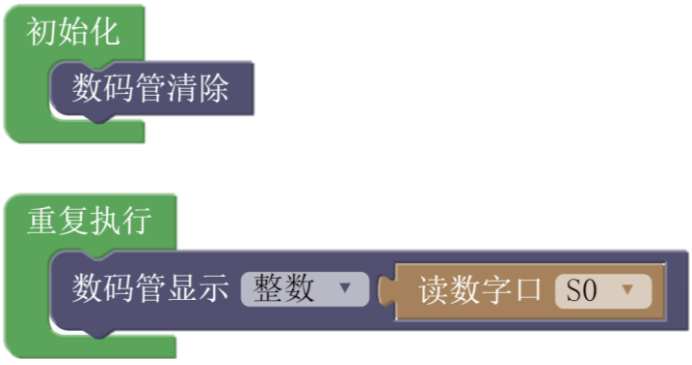
**元器件列表：**

1. WU-Link 主控板 ×1
2. 按钮开关模块 ×1

**电路连接：**




**程序编写：**



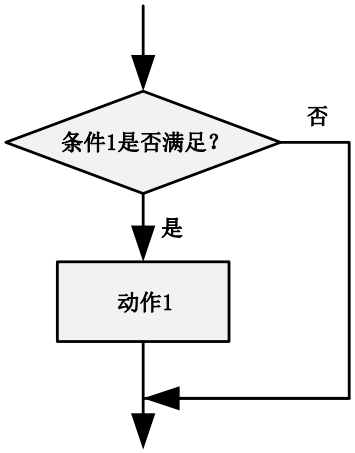

上传程序后，可以看到当按钮按下时，点阵屏显示 1，当按钮松开时，点阵屏显示 0。可以把按钮开关换成其他开关量传感器模块，试试看。

**条件判断指令**

数字量传感器的使用，通常都要配合条件判断指令来使用，即需要判断数字量传感器输入的数值是“1”还是“0”：如果是“1”，则认为是“真”；如果是“0”，则认为是“假”。

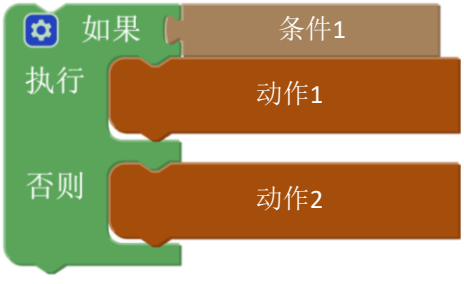
**认识新指令——如果-执行指令（控制）**

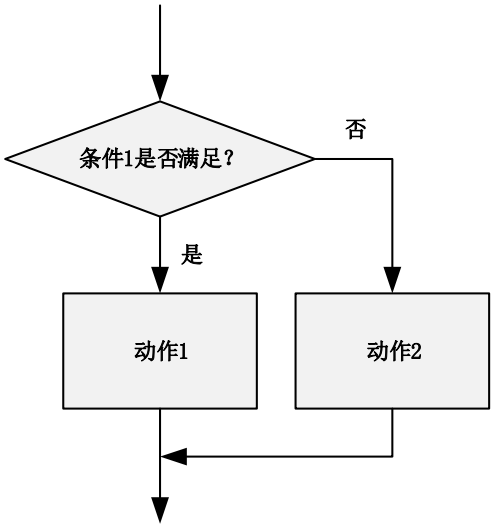

**功能说明：**如果条件 1 成立，则执行动作 1，否则不执行程序 1

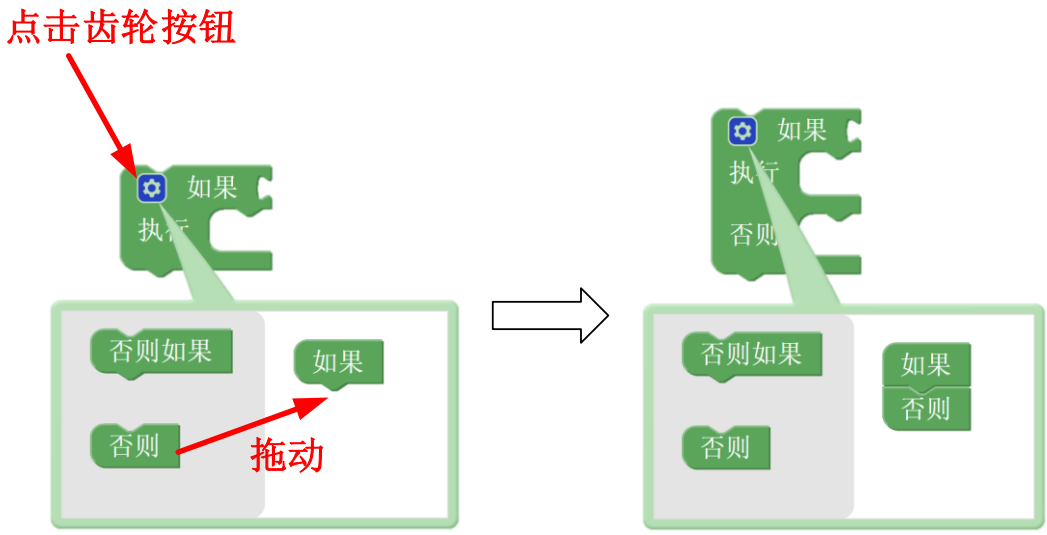

**认识新指令——如果-执行-否则指令（控制）**


**功能说明：**如果条件 1 成立，则执行动作 1 不执行动作 2，否则不执行动作 1 执行动作 2。



如果-执行-否则指令是由如果执行指令变形而来的，变形操作如下：

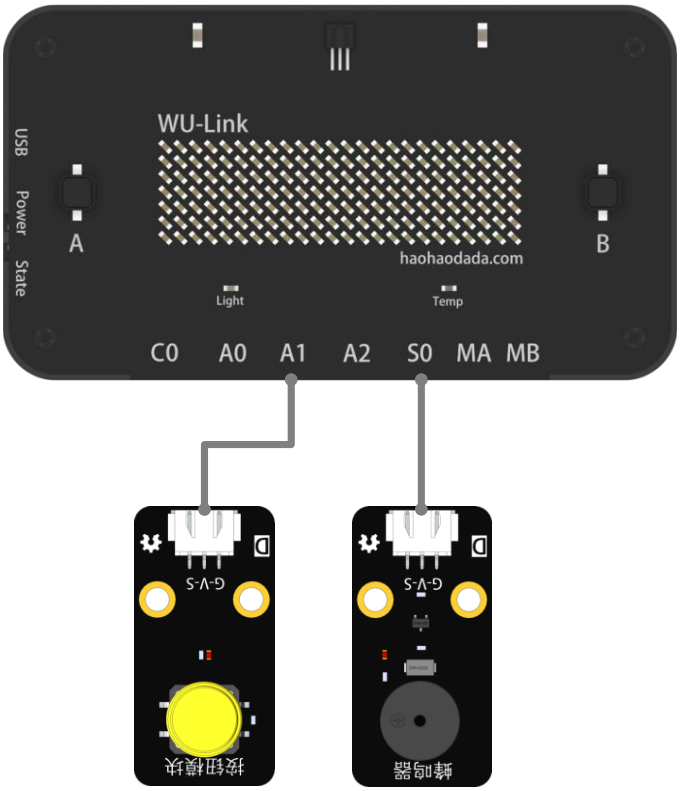



**示例 11-2：门铃**

**元器件列表：**

1. WU-Link 主控板 ×1
2. 按键模块 ×1
3. 蜂鸣器模块 ×1

**电路连接：**



**程序编写:**


如果-执行指令


如果-那么-否则指令

条件判断指令是硬件编程中最重要的指令。能否灵活巧妙的使用它，可以直接的表现出一个程序员的逻辑思维能力。

后续的课程会不断的使用到条件判断指令，会有更多精妙的应用！

# 第十二节 变量

**概念：**

**变量**可以保存程序运行时用户输入的数据和特定的运算结果。

在程序中使用变量几个主要作用是：

（1）提升程序可读性；

（2）方便批量修改程序数据；

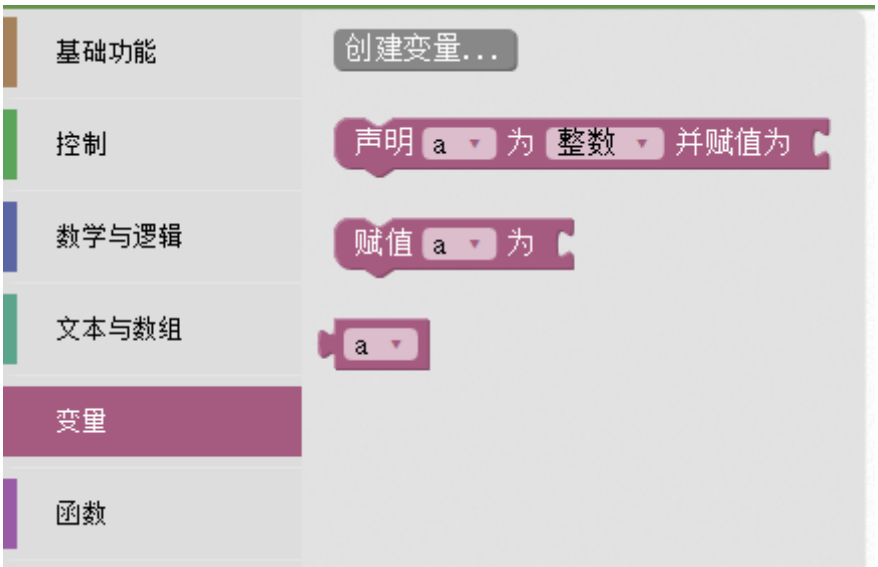
（3）存放程序中间运算结果。

变量的运用是编程最重要的能力之一，能否灵活恰当的使用变量能直接体现对编程理解的深度。

**新建变量：**



新建完成后，再次“变量”类目，可以看到所有已建的变量。




**认识新指令——赋值指令（变量）**

用于将用户要输入的数据或中间运算结果写入到变量中。


**认识新指令——变量调用指令（变量）**

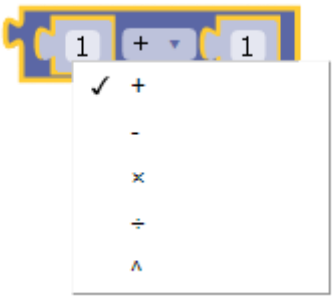
当需要处理变量时使用。


变量的作用要在运算、逻辑判断、循环、函数等构成复杂程序逻辑的场合才能真正的体现，所以变量的相关案例和解释将在后续的章节中详细说明。

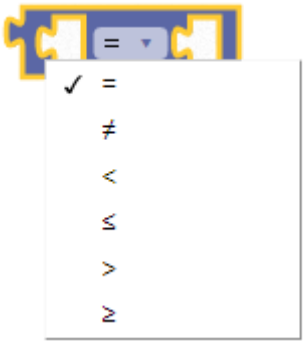
# 第十三节 数学与逻辑运算

在程序编写中经常需要对变量或数值进行换算或比较，这时可以调用数学与逻辑运算类指令。

## 认识数学运算指令：




## 认识逻辑运算指令：




比较指令的运行结果为“真”或“假”，即正确的为真，错误的为假。例如：

比较指令常配合“如果...那么...”指令使用。

## 认识布尔运算指令：


且运算

或运算

非运算

布尔运算指令是对真值（真或假）进行操作的指令，达到多个条件的协同判断的目的。

| 布尔运算真值表 | | |
|---|---|---|
|  真 |  假 |  假 |
|  真 |  真 |  假 |
|  假 |  真 | |

**示例 13-1：华氏温度计**

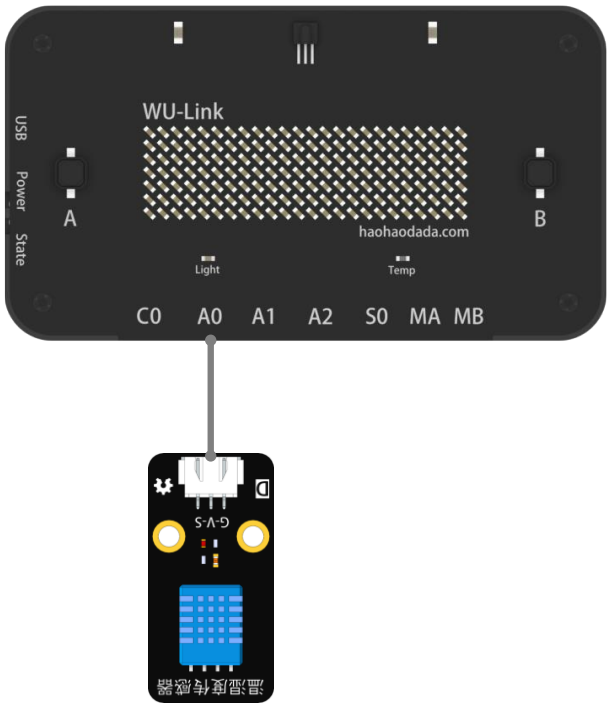
在日常生活中，常见的温度单位又两种：摄氏度和华氏度，前者是我国使用的温度单位（°C），后者是美国使用的温度单位（°F），两者的关系是：华氏度=摄氏度×1.8+32

通过程序实现：数码管显示摄氏度和华氏度，并用按键做为输入实现切换。

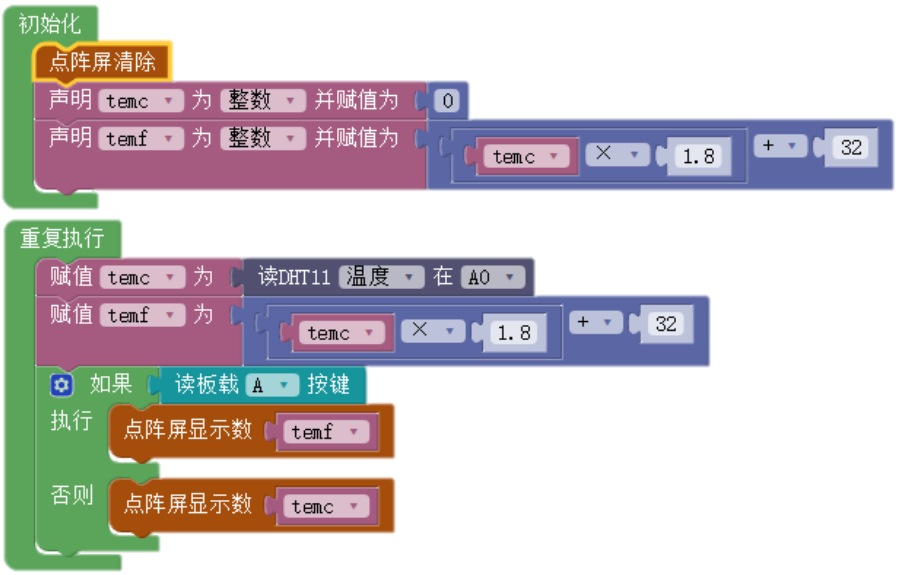
**元器件列表：**

1. WU-Link 主控板 ×1
2. 温湿度传感器模块 ×1

**电路连接：**




**程序编写：**




其中，变量 temc 和 temf 分别指代摄氏度和华氏度。这里变量的作用是提升程序的可读性和存放中间计算结果。

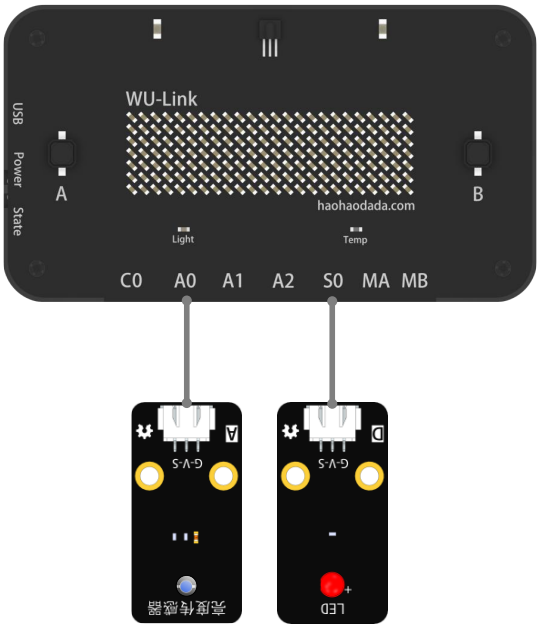
**示例 13-2：光控灯（光敏开关版）**

白天（光照强度高），LED 自动熄灭；夜晚（光照强度低），LED 亮起，LED 只需开关两种状态。

**元器件列表：**

1. WU-Link 主控板 ×1
2. LED 模块 ×1
3. 亮度传感器模块 ×1

**电路连接：**



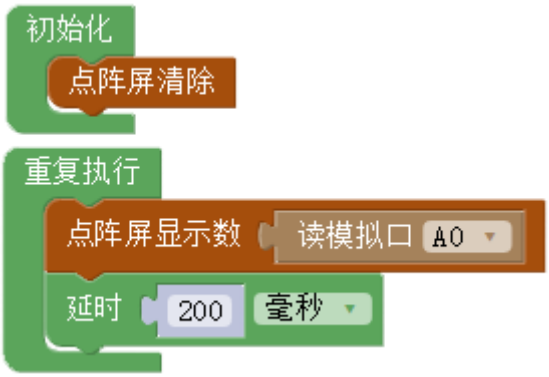
**程序编写:**

第 1 步

测量夜晚需要开灯时的光照强度，观察数码管的显示值并记录下来，这个值就是用于比较的阈值。

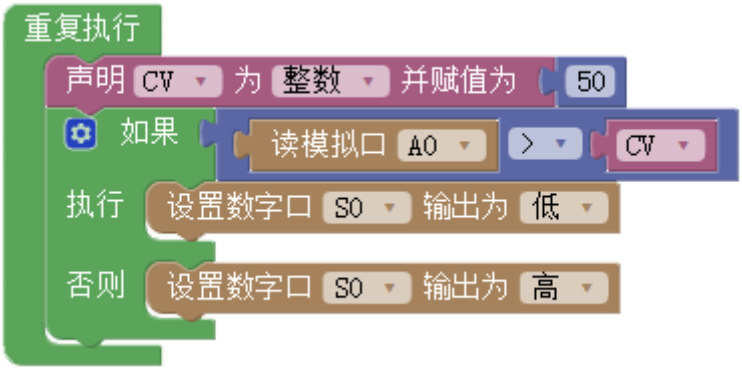
**阈值（Critical Value）**是指两个相邻定义域的边界。比如常说的白天和黑夜的边界，春天和夏天的边界，白色和灰色的边界。有的阈值是约定俗成的，如春天和夏天边界是立夏这一天；有的阈值则是可以自定义的，如冷和热，亮和暗，每个人的标准都不一样。

模拟量输入与阈值进行比较大小，即可将模拟量输入传感器视为一个开关。




第 2 步

根据的第 1 步测量到的阈值，本例中使用的阈值为 50，编写光控开关的程序：



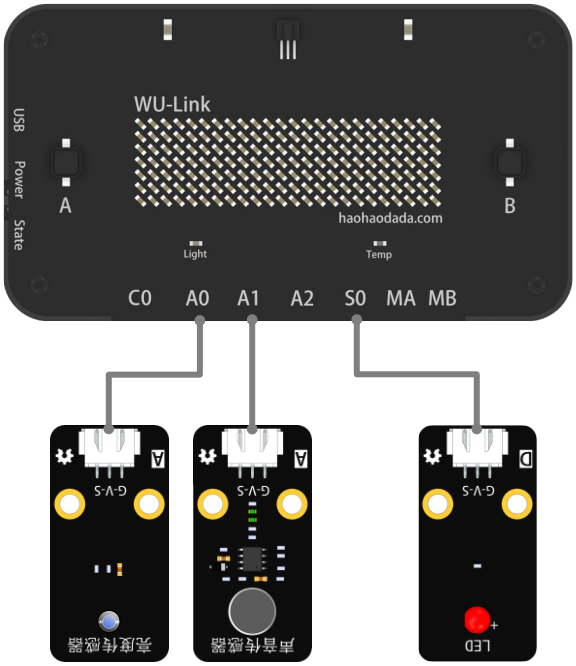

**示例 13-3：夜间声控灯**

只有在夜间才能通过声音点亮的灯

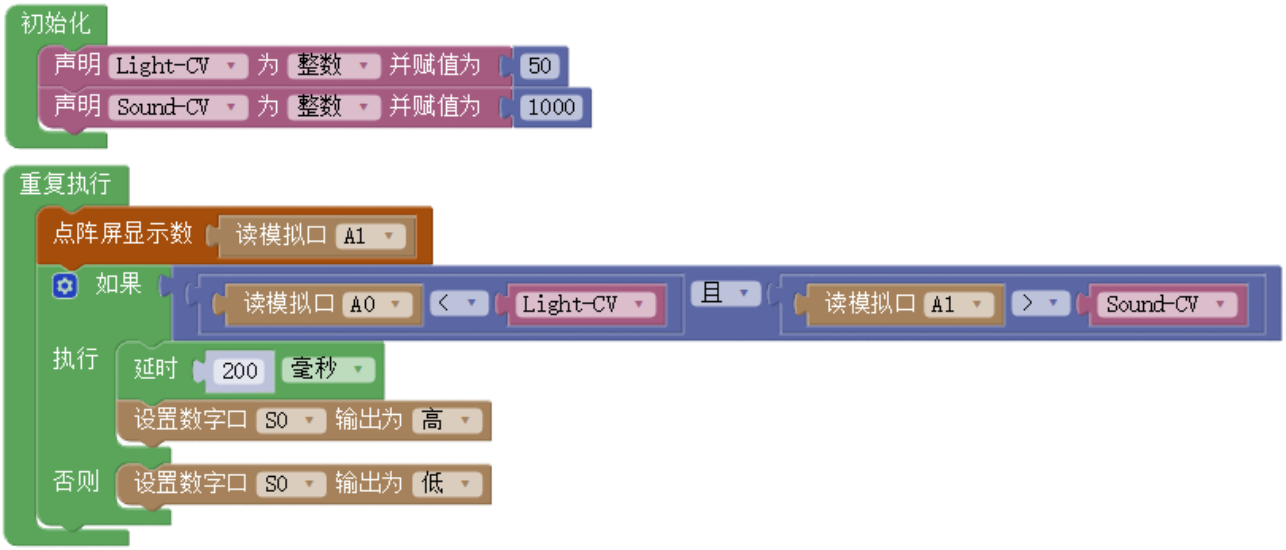
**元器件列表：**

1. WU-Link 主控板 ×1
2. 亮度传感器模块 ×1
3. 声音传感器模块 ×1

电路连接：


WU-Link
USB
Power
State
A
B
haohaodada.com
Light
Temp
C0 A0 A1 A2 S0 MA MB
光度传感器
声音传感器
LED

程序编写：


初始化
声明 Light-CV 为 整数 并赋值为 50
声明 Sound-CV 为 整数 并赋值为 1000
重复执行
点阵屏显示数 读模拟口 A1
如果
读模拟口 A0 < Light-CV 且 读模拟口 A1 > Sound-CV
执行
延时 200 毫秒
设置数字口 S0 输出为 高
否则
设置数字口 S0 输出为 低

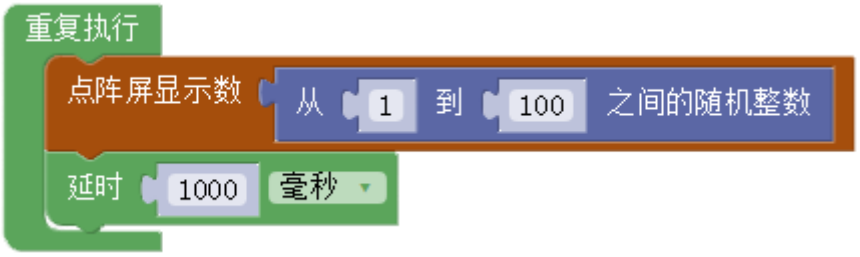
# 第十四节 随机数

在编写程序时，尤其是设计游戏和模拟实验时，你有可能需要生成随机数让程序富有变化，这时可以使用随机数指令。

**认识随机数指令**

用于随机从某数值范围中抽取一个数（整数）。


指令中的两个参数指定随机数的取值范围。随机数指令每运行一次，产生一个随机数，产生的这个数即为该指令的运行结果。试运行以下程序并分析：



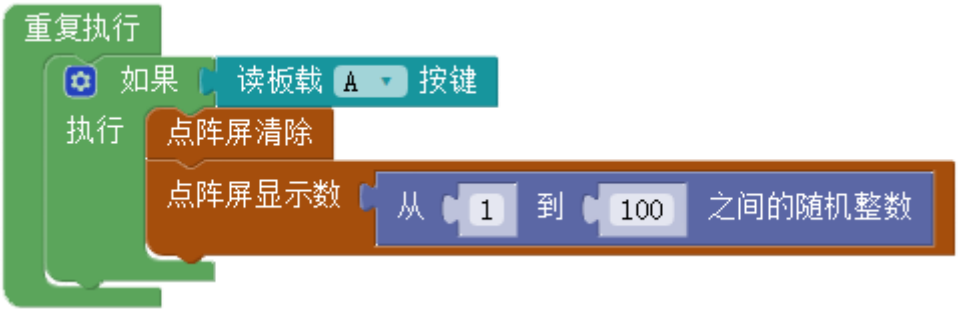

**示例 14-1：摇号机（基本版）**

从学号尾号 01 到 100 之间抽取一个号码，多次抽取，号码可能出现重复。

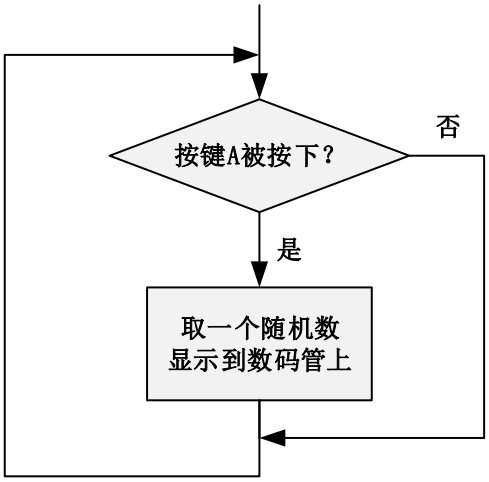
**元器件列表：**

1. WU-Link 主控板 × 1

**程序编写:**




**程序说明:**




（1）当按键 A 被按下，不断运行随机数指令，显示到数码管上，看到不断滚动的数字；

（2）当按键 A 被松开，不运行随机数指令，数码管数字不更新，看到一个确定的数字。

# 第十五节 串口打印

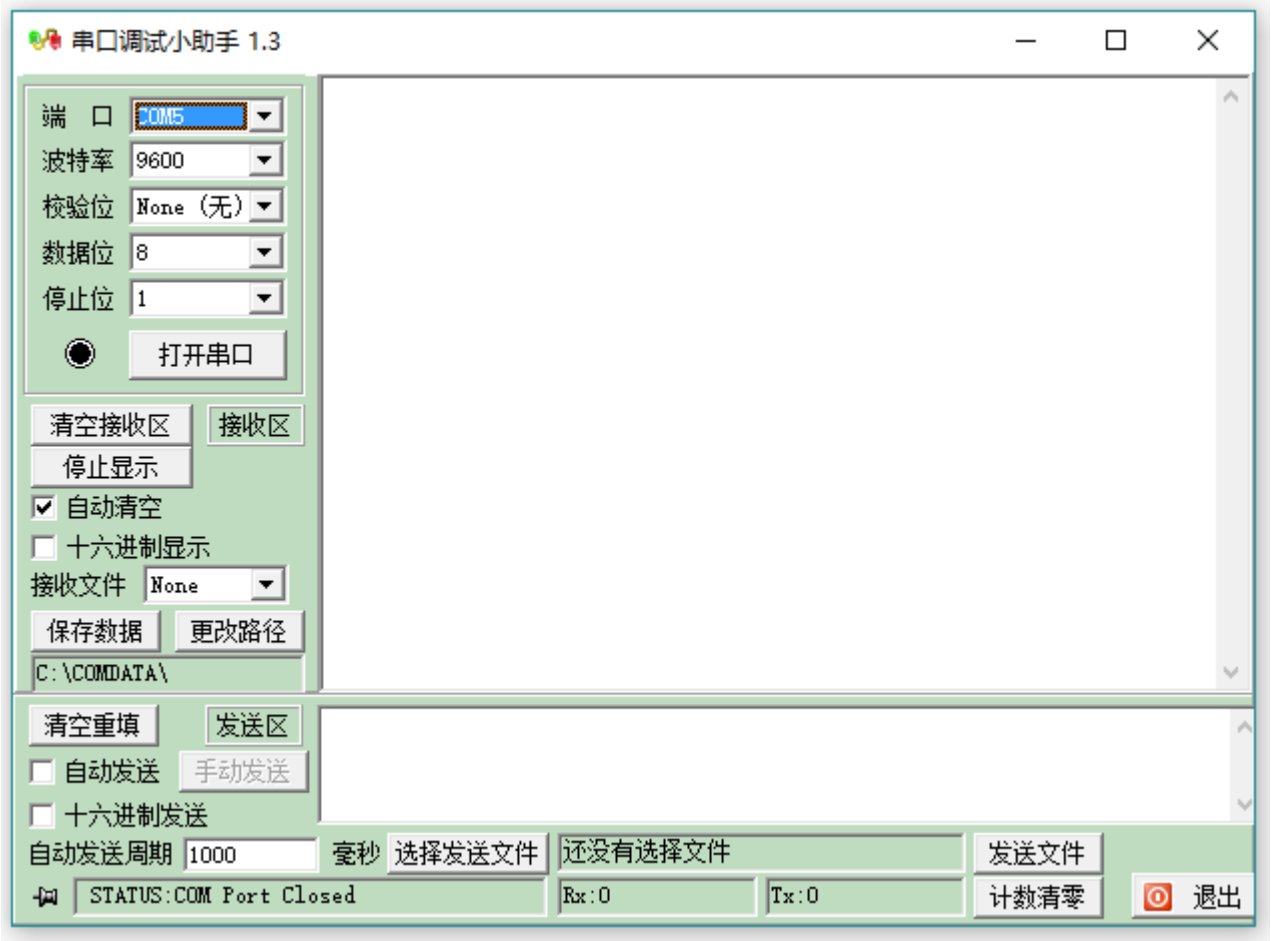
在调试程序时，程序员经常需要查看各种参数或变量的数值，这时可以利用串口助手，将这些信息在电脑屏幕上显示出来。

**认识串口助手**

在“硬件下载插件”中点击“打开串口助手”按钮。


硬件下载助手的下载地址在好好搭搭网站的“资源”->“好好搭搭插件下载”中：http://www.haohaodada.com/art_show.php?id=53

串口调试小助手界面：




**波特率**是指通过串口发送数据的速度，即单位时间内发送数据的个数。单位：bps（每秒位数）。

**认识串口速率设置指令**


**波特率**是指通过串口发送数据的速度，即单位时间内发送数据的个数。单位：bps（每秒位数）。

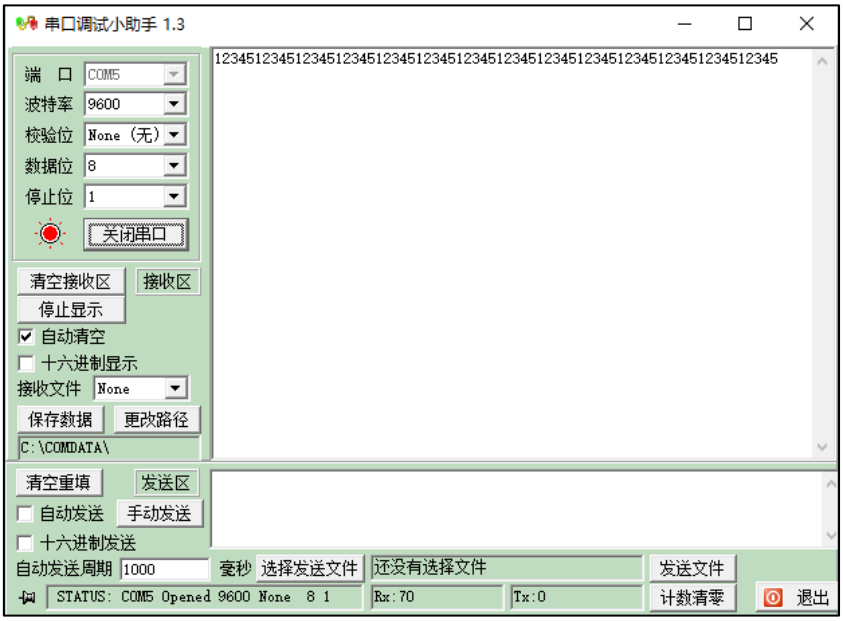
常用的波特率有：1200、2400、4800、9600、19200、38400、57600、115200。NOVA 串口的默认波特率为 9600。

程序中设置的“串口速率”一定要与串口助手中的“波特率”一致，是正常实现串口通讯的必要条件！

**认识串口输出指令**


与数码管显示指令类似，串口输出指令是将数据显示到串口助手中。

重复运行串口输出指令的效果是：




串口输出指令输出的数据，不会自动换行，重复执行之后数据之间会首尾相连，所以串口输出指令常用于连接多个数据打印时使用，如果只输出一个数据，建议使用“串口输出并换行”指令。

**认识串口输出并换行指令**

串口输出并换行指令也是将数据显示到串口助手中，相比串口输出指令，它会在输出数据结束之后，自动添加一个换行符。

重复运行串口输出并换行指令的效果是：

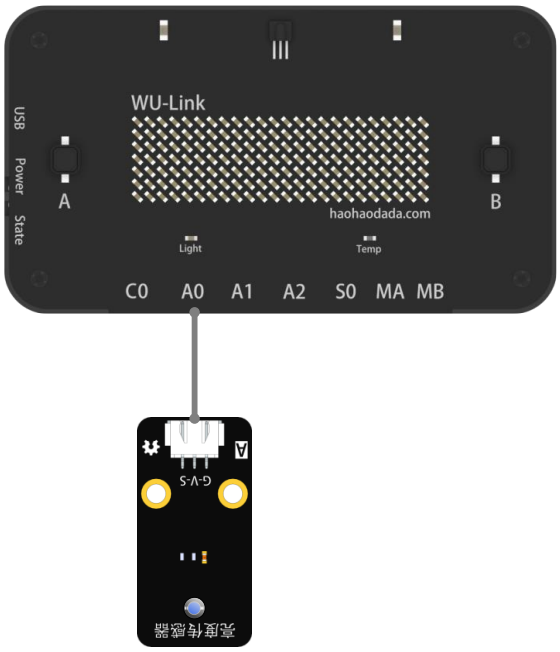

**示例 15-1：串口打印单个传感器数值**

在串口监视器上显示亮度值。

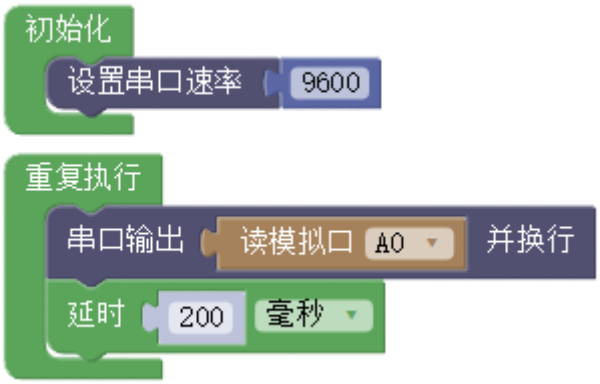
**元器件列表：**

1. WU-Link 主控板 × 1
2. 亮度传感器模块 × 1

**电路连接：**



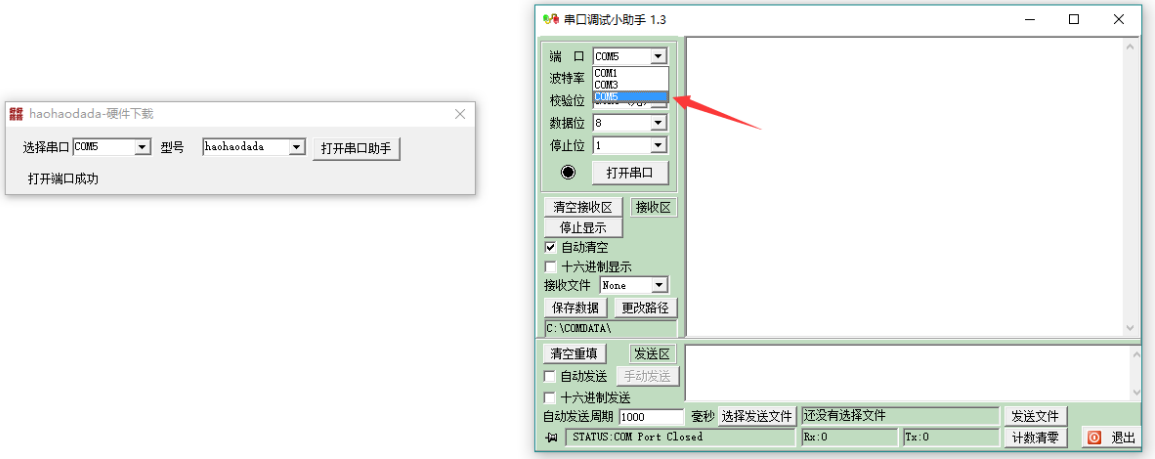
**程序编写:**




待上传完成之后，打开串口助手。注意！不要断开 USB 连接。

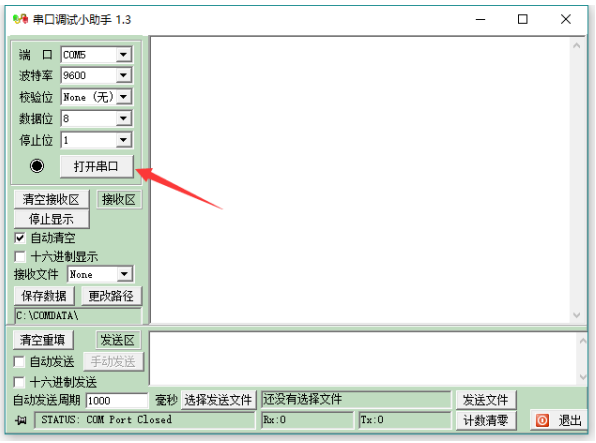
需完成如下三项操作，串口助手方可将数据显示出来。

（1）端口选择。端口选择与“硬件下载插件”一致。如本例中硬件下载插件中的串口为 COM5，则串口助手中端口选择也为 COM5；




（2）波特率检查。串口助手中的波特率要与好好搭搭程序中的串口速率一致。本例中选用默认的 9600。

（3）点击“打开串口”按钮，之后，便可在接收区看到数据的显示了。




打开串口前


打开串口后

注意！注意！注意！如果要再次编译下载程序或断开 USB 连接，必须关闭串口！

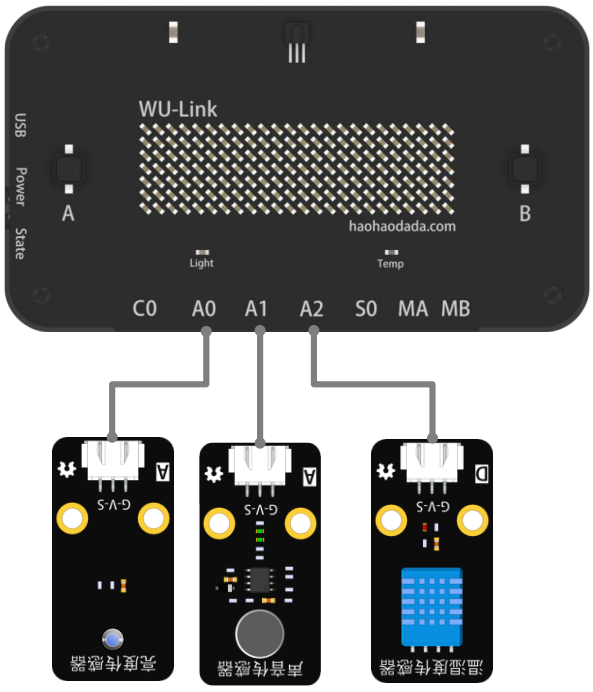
**示例 15-2：串口打印多个传感器数值**

在串口监视器上显示亮度值、声音值、温度值和湿度值。

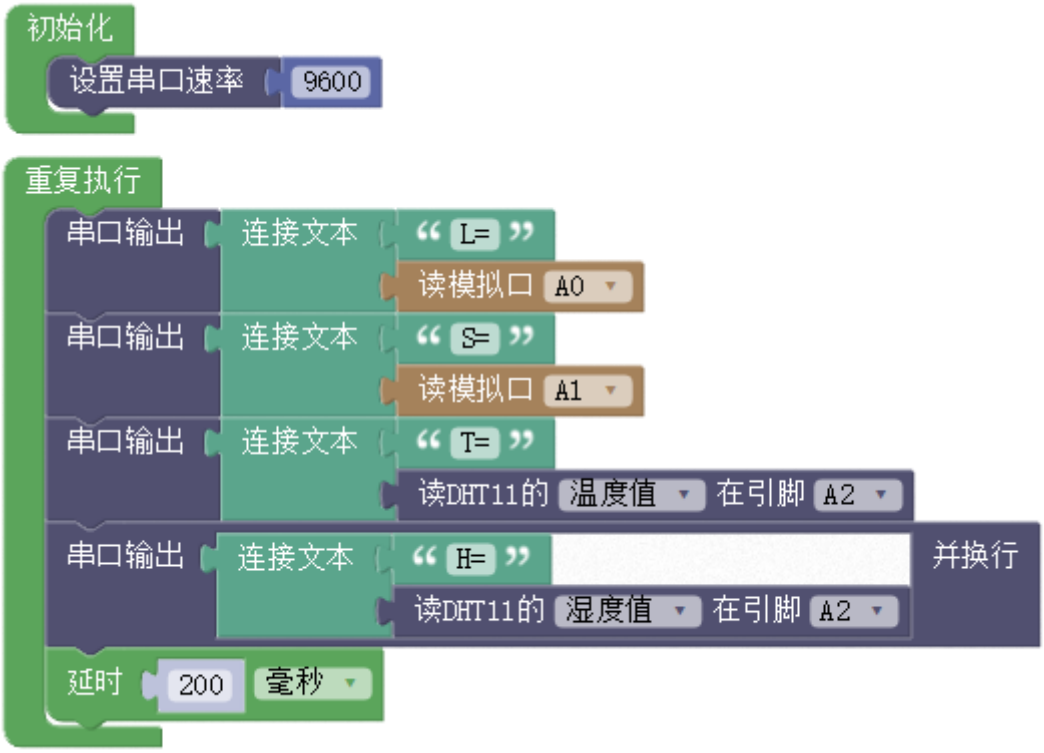
**元器件列表：**

1. WU-Link 主控板 ×1
2. 亮度传感器模块 ×1
3. 声音传感器模块 ×1
4. 温湿度传感器模块 ×1

**电路连接：**



**程序编写：**



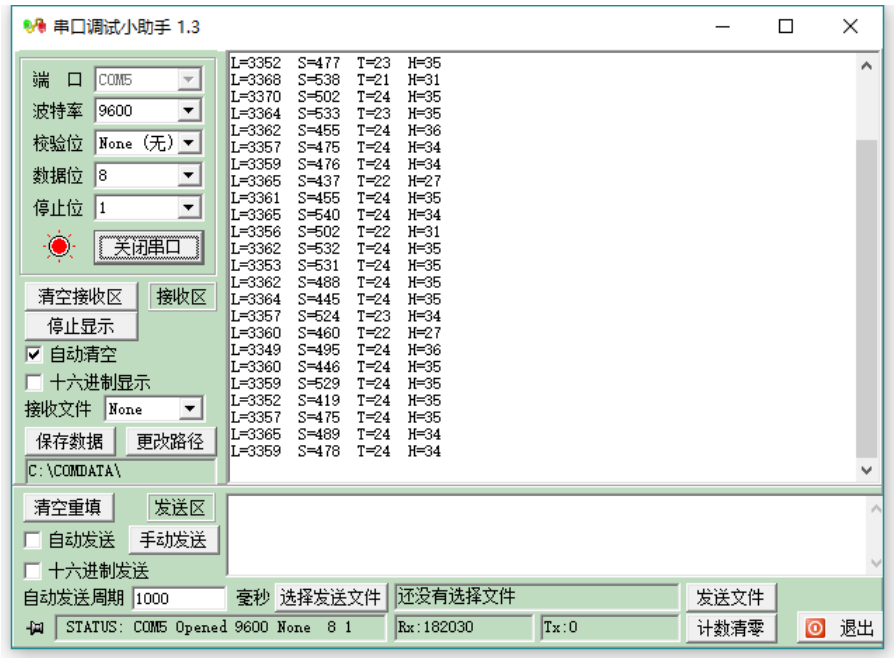

其中，

（1）L 为 Light 光；S 为 Sound 声音；T 为 Temperature 温度；H 为 Humidity 湿度。

（2）为了让所有输出信息，在同一行，所以最后一条输出指令用“串口输出并换行”指令，之前的全部用“串口输出”指令。

（3）为让不同相同信息互相隔开，可以用空格符，如：

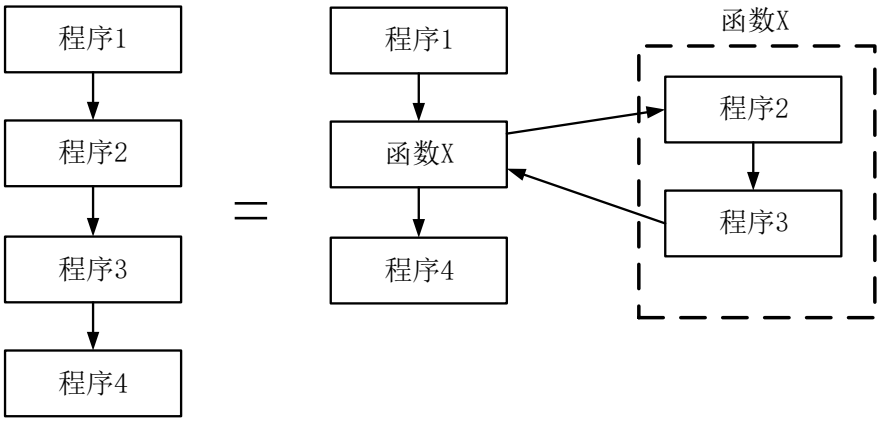

打印结果为：


串口调试小助手 1.3
端 口 COM5
波特率 9600
校验位 None (无)
数据位 8
停止位 1
关闭串口
清空接收区
接收区
停止显示
自动清空
十六进制显示
接收文件 None
保存数据
更改路径
C:\COMDATA\
L=3352 S=477 T=23 H=35
L=3368 S=538 T=21 H=31
L=3370 S=502 T=24 H=35
L=3364 S=533 T=23 H=35
L=3362 S=455 T=24 H=36
L=3357 S=475 T=24 H=34
L=3359 S=476 T=24 H=34
L=3365 S=437 T=22 H=27
L=3361 S=455 T=24 H=35
L=3365 S=540 T=24 H=34
L=3356 S=502 T=22 H=31
L=3362 S=532 T=24 H=35
L=3353 S=531 T=24 H=35
L=3362 S=488 T=24 H=35
L=3364 S=445 T=24 H=35
L=3357 S=524 T=23 H=34
L=3360 S=460 T=22 H=27
L=3349 S=495 T=24 H=36
L=3360 S=446 T=24 H=35
L=3359 S=529 T=24 H=35
L=3352 S=419 T=24 H=35
L=3357 S=475 T=24 H=35
L=3365 S=489 T=24 H=34
L=3359 S=478 T=24 H=34
清空重填
发送区
自动发送
手动发送
十六进制发送
自动发送周期 1000 毫秒
选择发送文件
还没有选择文件
发送文件
STATUS: COM5 Opened 9600 None 8 1
Rx:182030
Tx:0
计数清零
退出

# 第十六节 新建函数

函数的作用是将主程序中的一部分程序段封装起来，运行主程序运行到函数位置时，调用函数内部的指令，运行完之后，回到主程序，继续运行之后的程序。当程序复杂到一定程度之后，一部分功能相对独立或会被多次使用的程序段通常会采用新建函数的方式简化程序，提升程序的可读性。

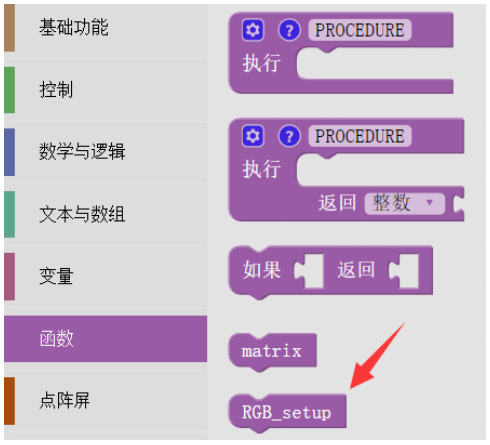
下图两个程序等效：



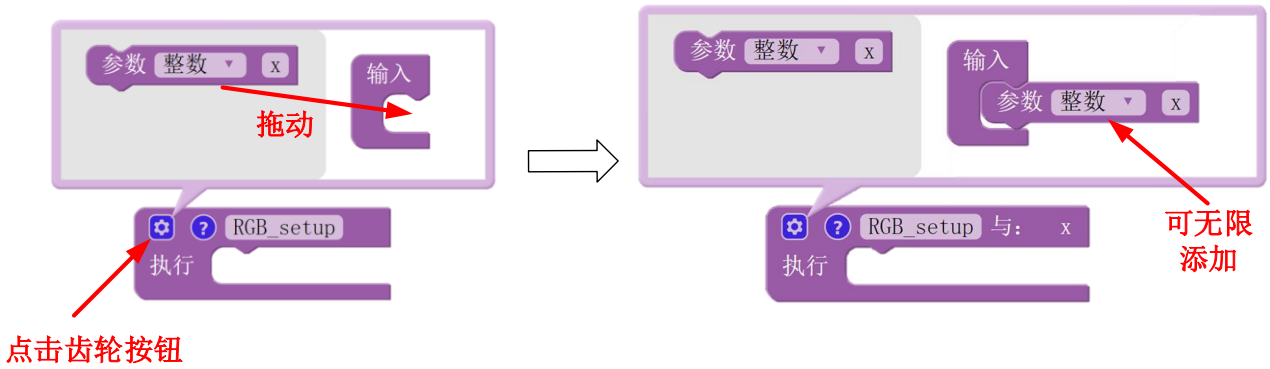

**认识新指令——新建函数（无返回值）**


将该指令拖到图形编程区，修改函数名称即可创建一个新的函数，例如将函数名改为“RGB_setup”：

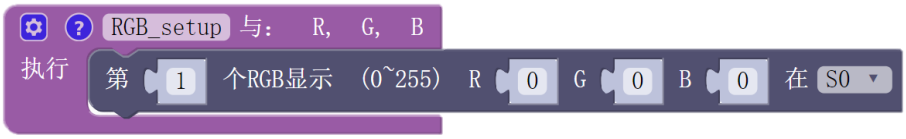

再次点开“函数”类目，即可找到新的函数“RGB_setup”。



当函数在调用时需要放入不同的参数时，需要给函数添加参数框。操作如下：




参数名称可自定义修改，但是不能为中文。RGB 显示程序段放入函数如下：

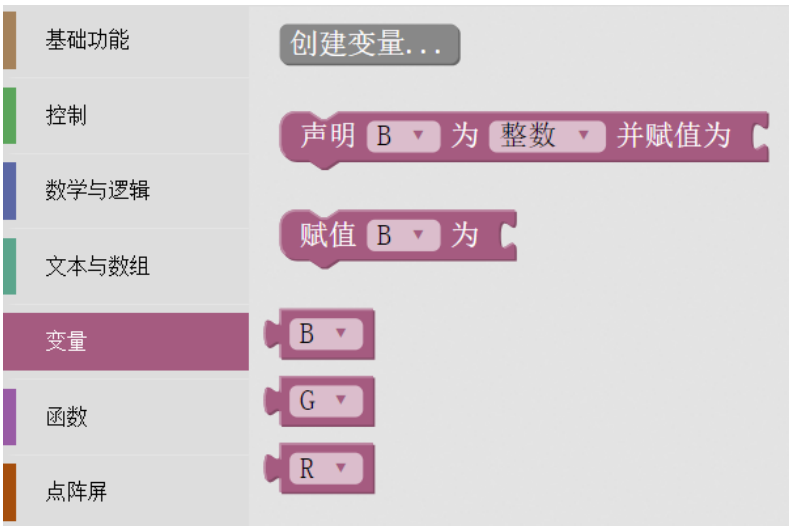



此时，点开“函数”类目，可以看到 RGB_setup 函数变为：

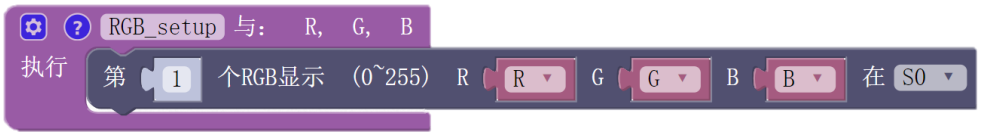

RGB_setup 函数的 R、G、B 三个缺口可以接入数值或其它变量。

但此时还有个问题是当 RGB_setup 函数被调用时，运行的程序是：


与参数 R、G、B 无关，所以上图程序中需加入参数 R、G、B。点开“变量”类目，可以看到三个变量 R、G、B。



将这里的变量 R、G、B 分别拖入 RGB 显示的三个参数框中：




此时，一个可以传递参数的函数完成了。

注意：因为这里参数 R、G、B 虽然是在“变量”类目中，但是因为没有被声明，所以变量 R、G、B 不能在主程序中被使用，否则会编译失败。

**示例 16-1：RGB 交通灯（函数版）**

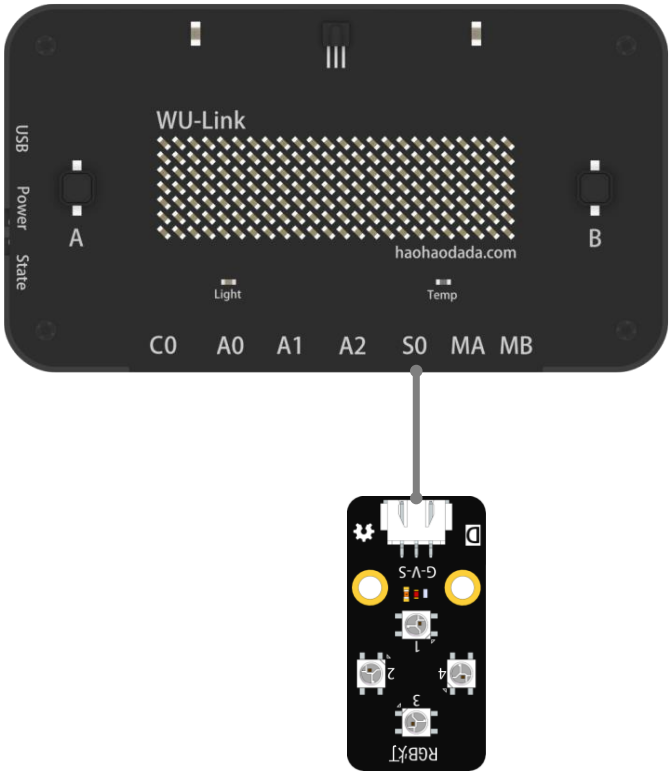
在示例 9-1 中，只用了 RGB 模块上的一个 RGB 灯实现交通灯的应用。本例用四个 RGB 灯一起来实现该应用。

要点亮 RGB 模块上的 4 个灯，要用 4 条语句，每个动作都要使用四条语句，则程序会变得非常冗长。巧妙地使用函数能很好的简化程序。

**元器件列表：**

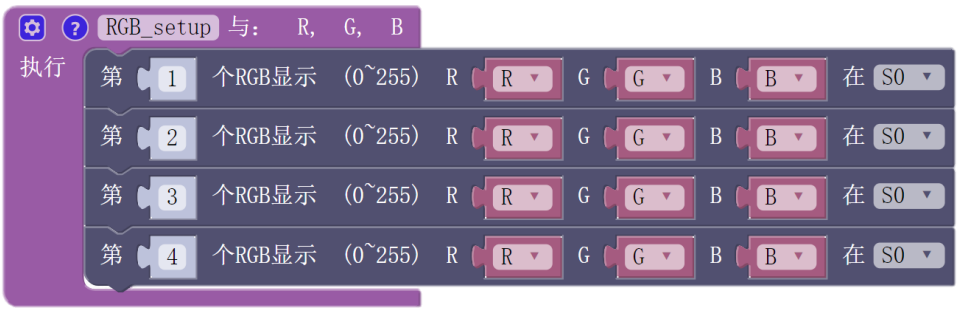
1. WU-Link 主控板 ×1
2. RGB 模块 ×1

**电路连接：**

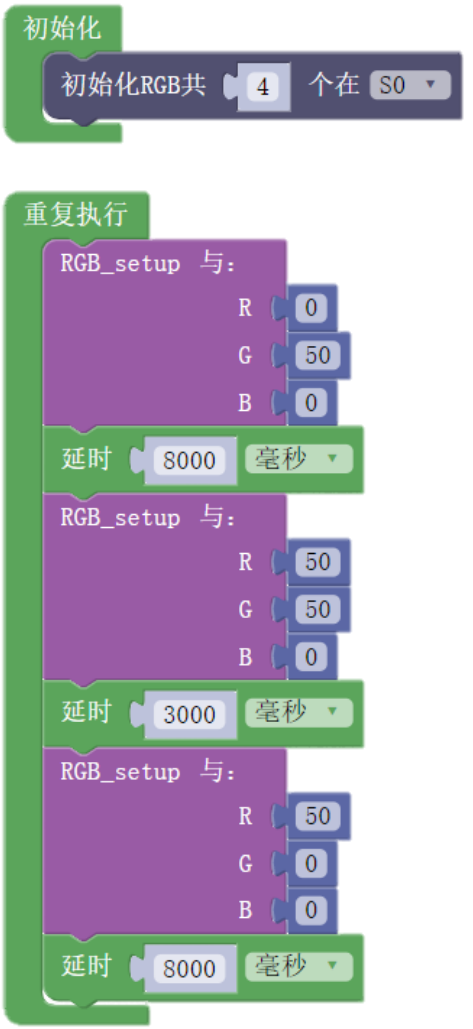


**程序编写：**

新建一个函数 RGB_setup：




程序为：



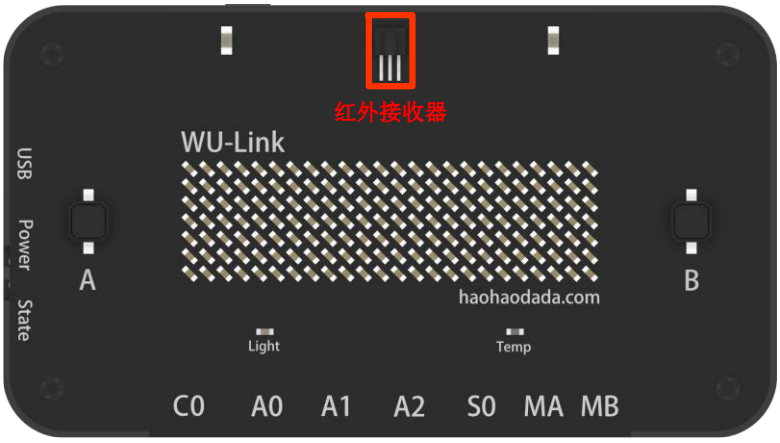
# 第三部分

# 硬件进阶

# 第十七节 红外遥控接收

**红外遥控**是广泛应用与生活电器的远程控制方式，需要一个红外信号发射器，一个红外信号接收器。

在 WU-Link 点阵屏的上方，板载了一个红外接收器，使得 WU-Link 能够接收到红外遥控器发出的信息。




**认识新硬件——红外遥控器**

红外信号发射器，简称“遥控器”，不同的按键按下，会发送不同信息，称为键值。

**认识新指令——读取板载红外接收值指令**


使用这个指令可以读取 WU-Link 板载红外接收模块接收到的红外数值。

**认识新指令——红外按键值指令**


红外按键值指令是为了方便用户使用制作的图形化指令，可以通过下拉菜单来选择对应

的按键。

注意：该指令只支持官方配套遥控器，其它遥控器的使用方法详见 17-2。

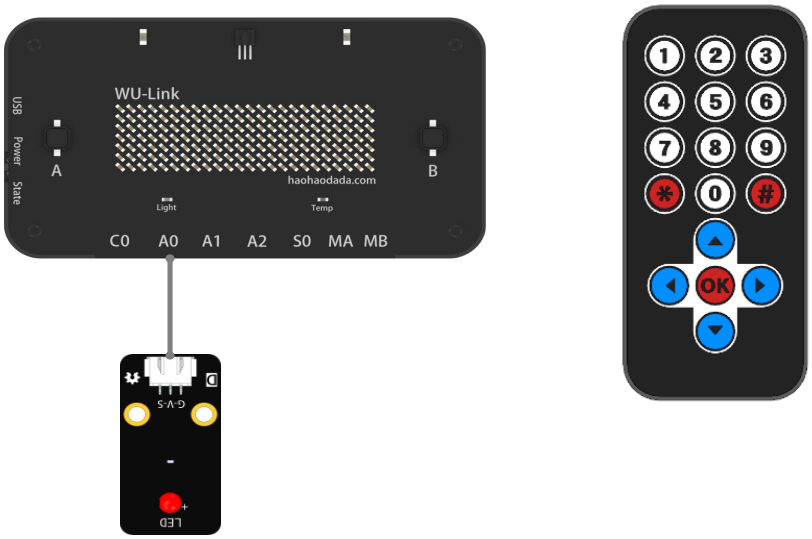
**示例 17-1：红外遥控 LED**

用红外遥控方式控制 LED 的亮灭

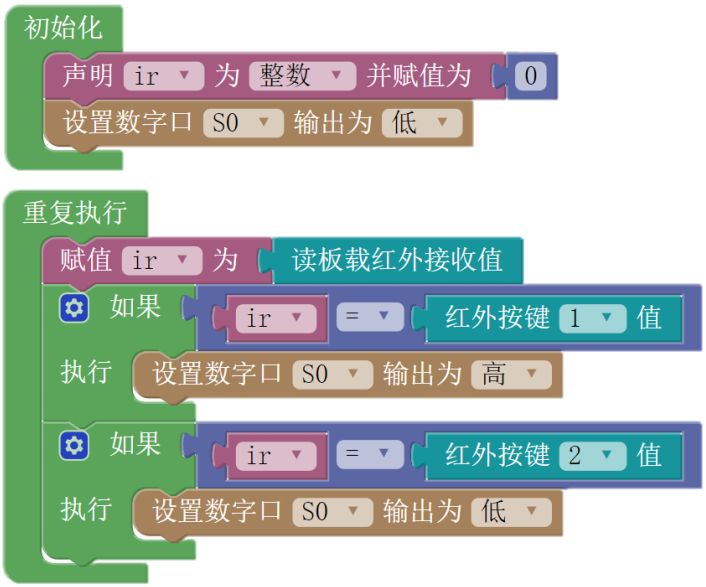
**元器件列表：**

1. WU-Link 主控板 ×1
2. LED 模块 ×1
3. 红外遥控器 ×1

**电路连接：**




**程序编写：**




示例程序地址：http://haohaodada.com/wulink/index.php?id=1889

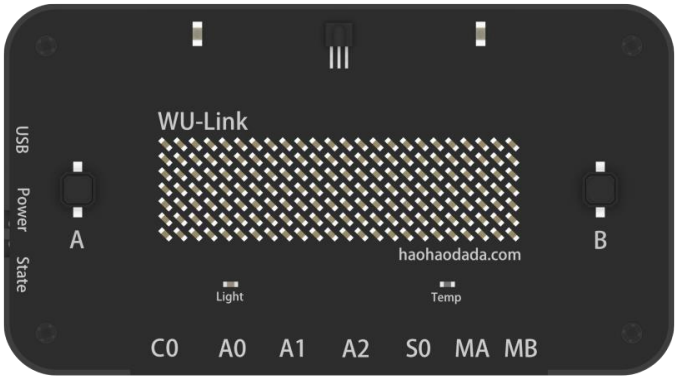
**示例 17-2：用点阵屏显示红外遥控器的按键值**

红外接收模块除了支持官方配套遥控器外，也支持大多数生活中的遥控器，如电视遥控器、空调遥控器、投影机遥控器。

**元器件列表：**

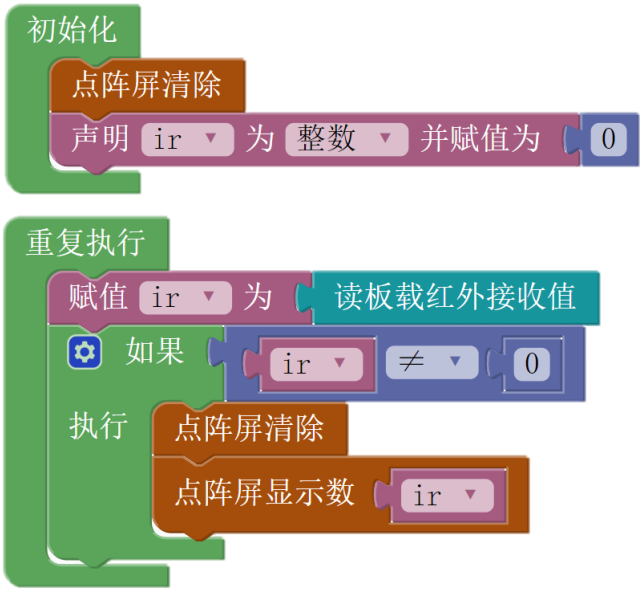
1. WU-Link 主控板 ×1
2. 红外遥控器 ×1

**电路连接：**





**程序编写：**




示例程序地址：http://haohaodada.com/wulink/index.php?id=1890

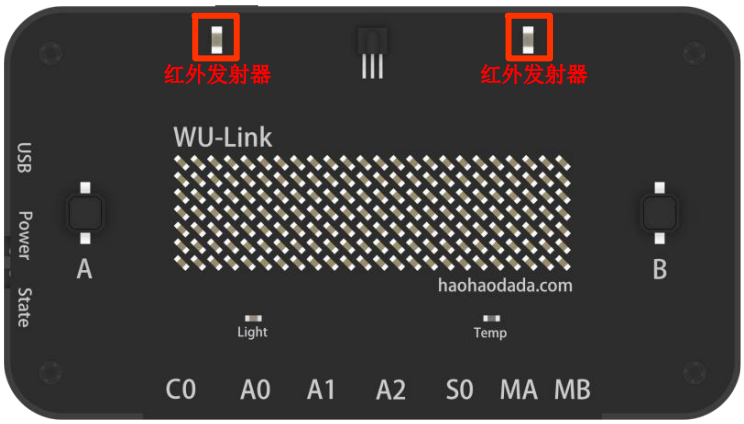
当没有开始操作红外遥控器时，点阵屏上面显示空白；而当按下红外遥控器上的按键，板载红外接收器接收的红外按键值在点阵屏上会接收到有且只有一个值，之后就算一直按住按键不放，也只会显示该值。

通过上述方法，把遥控器的各个按键的值记录下来，如本例中用的某款电视遥控器中的“确定”键的值为 87，“返回”键的值为 175，利用这两个按键控制 LED 的亮灭，示例 16-2 的程序变为：


示例程序：http://haohaodada.com/wulink/index.php?id=1888

# 第十八节 红外遥控发射

在 WU-Link 点阵屏的上方，板载了两个红外发射模块，使得 WU-Link 能够发射红外编码信息。




**认识新指令——板载红外发射指令**


该指令可以让 WU-Link 板载红外发射模块发射指定的红外数值，数值在参数框输入。

**示例 18-1：用 WU-Link 红外遥控控制 LED**

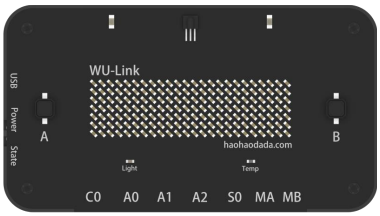
用一块 WU-Link 红外遥控方式控制另外一块 WU-Link 外接 LED 的亮灭

**元器件列表：**

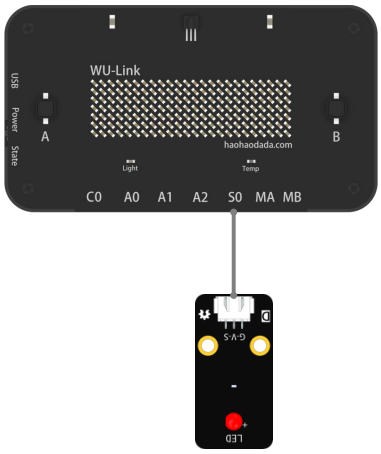
1. WU-Link 主控板 × 2
2. LED 模块 × 1

**电路连接：**

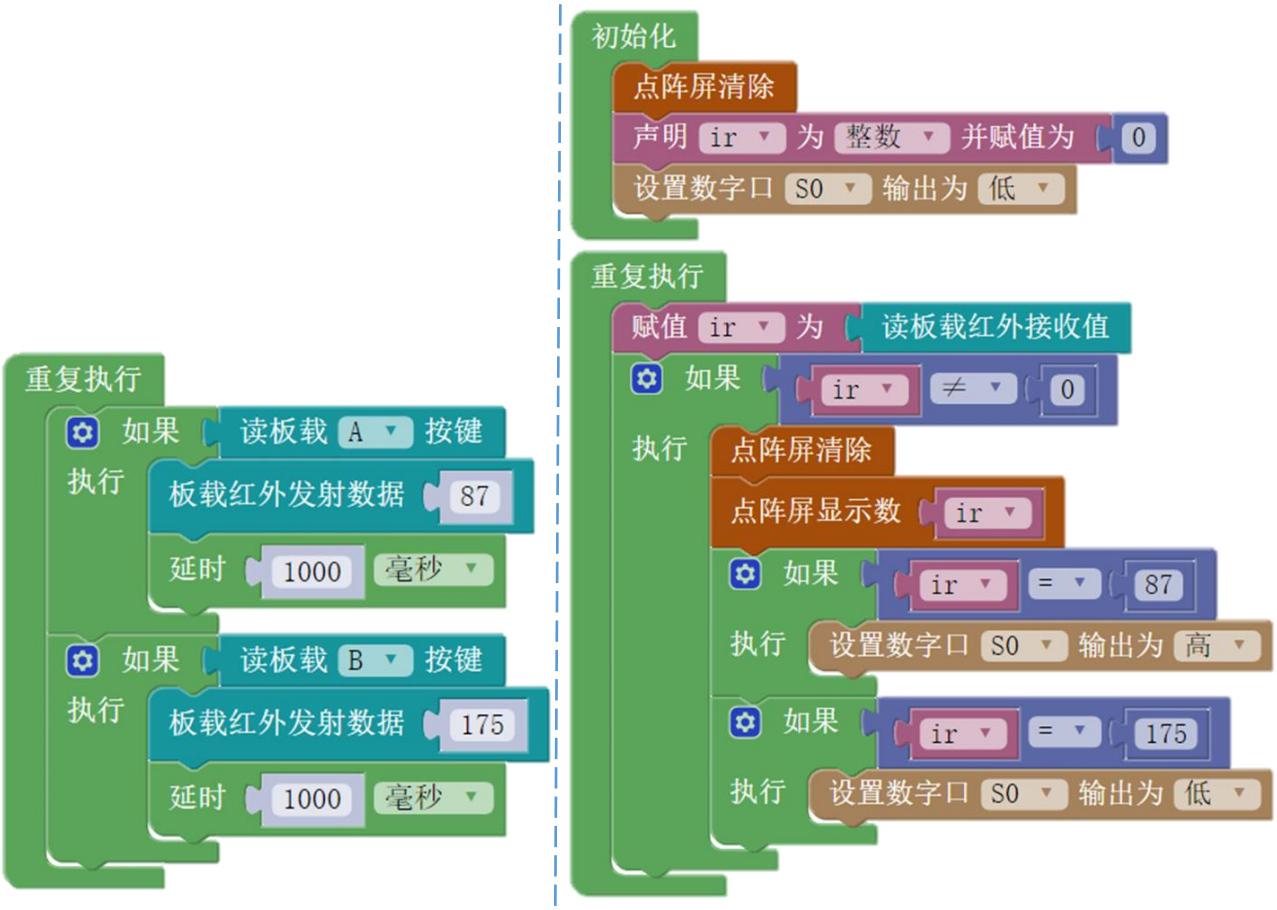
红外发射




红外接收



**程序编写:**

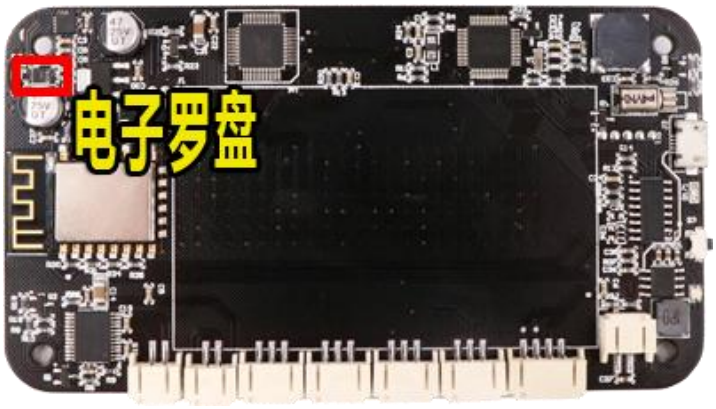



红外发射程序　　红外接收程序

发射示例程序地址：http://haohaodada.com/wulink/index.php?id=1891
接收示例程序地址：http://haohaodada.com/wulink/index.php?id=1892

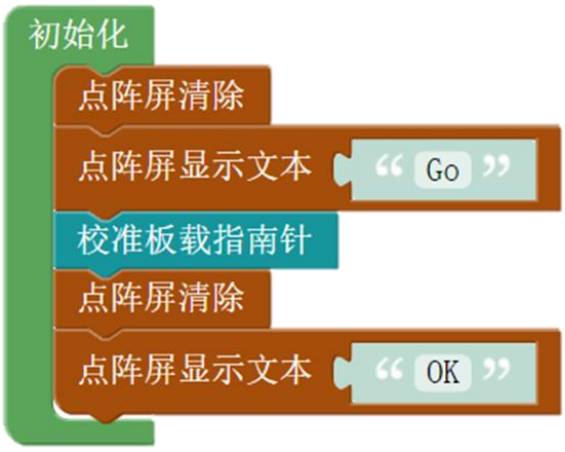
# 第十九节 电子罗盘

**电子罗盘**是使用磁阻传感器制作的，WU-Link 内部的电路板上就集成了一块电子罗盘芯片（如下图所示），使得 WU-Link 也具有识别方向的功能。




**认识新指令——校准板载指南针指令**

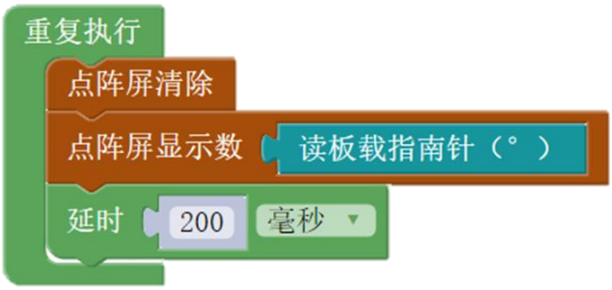

由于地球磁场非常微弱，电子罗盘容易受到各种电子产品的干扰。为了提高数据准确性，电子罗盘在使用前一般都需要校准，让电子罗盘的数值更加准确。具体校准方法如下：




http://haohaodada.com/wulink/index.php?id=1893

**认识新指令——读板载指南针角度指令**


用于读取板载电子罗盘的角度值。取值范围（0-360），正北方是 0 度，正东方是 90 度，正南方是 180 度，正西方是 270 度。一般可以采用在点阵屏上显示方向角度值的形式指示方向。具体操作方法如下：




**示例 19-1：用电子罗盘制作指南针**

在 WU-Link 上显示方向的角度值，有时候不太直观，可以采用在点阵屏上显示“东、南、西、北”这四个方向文本的形式显示实际方向。

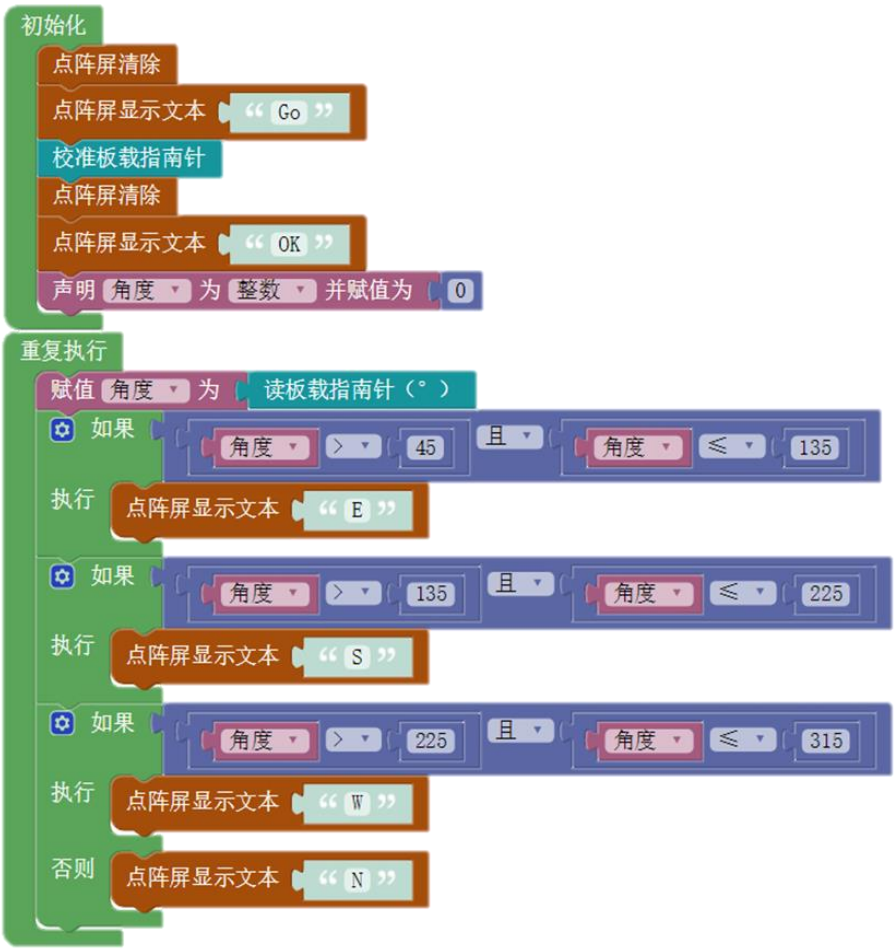
**元器件列表：**

WU-Link 主控板 × 1

**电路连接：**

略

**程序编写:**

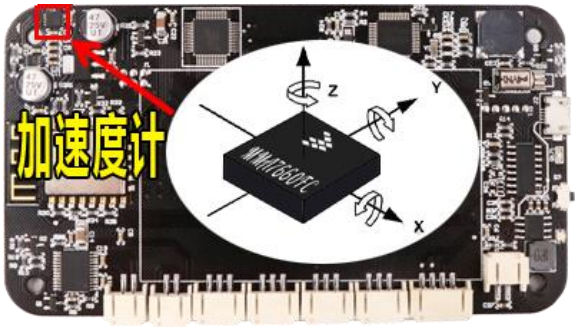



示例程序地址：http://haohaodada.com/wulink/index.php?id=1895

# 第二十节 加速度计

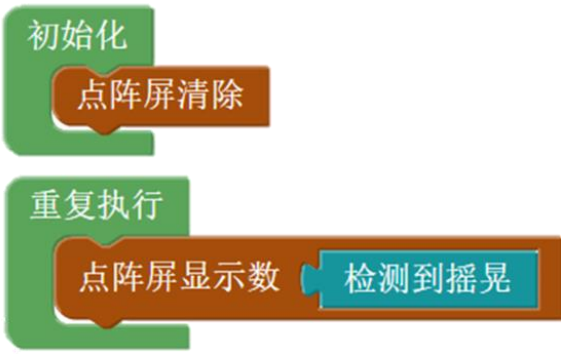
**加速度计**就是能够检测到物体速度变化快慢并转化、输出相应数值的传感器。在汽车安全系统、地震检测、游戏控制、手机计步器等很多方面都有广泛的应用。

WU-Link 在主板上集成的加速度计型号是“MMA7660FC(如下图所示)。它的大小是 3 × 3 × 0.9(毫米)，可以检测 X、Y、Z 三个方向所受到的加速度大小，检测精度是±1.5 个重力加速度。




**认识新指令——检测到摇晃指令**


用于检测 WU-Link 有没有被晃动。如果被晃动，返回值为“1”；否则，返回值为“0”。




**认识新指令——读板载加速度计指令**


可以读取 WU-Link 在左右(X)、垂直(Y)、前后(Z)三个方向上的运动变化数值，从而更为精确的了解 WU-Link 的运动状态。

**示例 20-1：摇号机（摇晃版）**

用加速度计控制的摇号机

**元器件列表：**

WU-Link 主控板 × 1

**电路连接：**

略

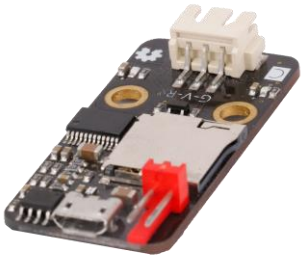
**程序编写：**


示例程序地址：http://haohaodada.com/wulink/index.php?id=1928

# 第二十一节 MP3 音乐播放

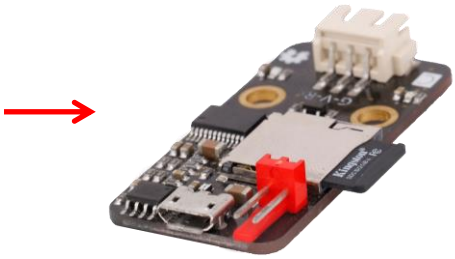
**MP3 音乐播放器**在生活中随处可见，被集成到各种各样的电子设备中，如：手机、平板电脑、汽车、智能手表。创意作品的制作中恰当的使用 MP3 播放模块能极大提升作品的表现力。

**认识新硬件——MP3 模块和 microSD 卡**

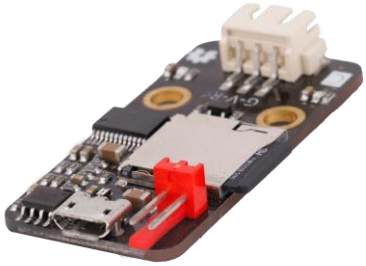


MP3 模块

microSD 卡



MP3 模块+SD 卡组合中



MP3 模块+SD 卡组合完成

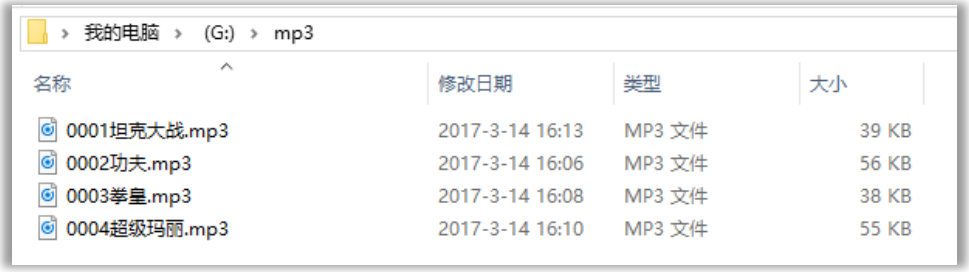
**认识新硬件——扬声器**

**SD 卡存放 MP3 文件**

将装有 SD 卡的 MP3 模块通过 USB 线与电脑相连，在 SD 卡根目录下新建一个名叫“mp3”的文件夹，将 MP3 音乐文件存入。


mp3 文件的文件名必须为“XXXX+任意字符”，如“0001 坦克大战”、“1578 魂斗罗”。文

件名的前四位数字即是该 MP3 文件的曲目编号，根据曲目编号可以播放指定的 MP3 文件。




**认识新指令——MP3 操作指令**


MP3 操作指令共有播放、循环播放、随机播放、暂停、下一曲、上一曲、音量加、音量减和停止九种操作模式。

（1）“播放”模式：继续播放 MP3 文件。如 MP3 模块处于暂停状态，运行“播放”程序，则继续播放；如 MP3 模块处于停止状态，运行“播放”程序，则重头开始播放。

（2）“循环播放”模式：重头开始按顺序播放 MP3 文件，当最后一首 MP3 文件播放结束后，又从第一首开始播放。不论 MP3 模块处于什么状态，运行“循环播放”程序，则重头开始播放。

（3）“随机播放”模式：重头开始乱序播放 MP3 文件。不论 MP3 模块处于什么状态，运行“随机播放”程序，则播放第一首 MP3 文件，之后按乱序播放 MP3 文件。

（4）“暂停”程序：让播放暂停，下一次运行“播放”程序，则从暂停位置继续播放。

（5）“下一曲”和“上一曲”：切换歌曲播放。当“顺序播放”时，按顺序切换歌曲；当“随机播放”时，按乱序切换歌曲。

（6）“音量加”和“音量减”：调节音量。

（7）“停止”程序：停止播放，下次播放重头开始。

**认识新指令——MP3 播放曲目指令**


该指令可以直接选取对应曲目文件播放，如参数为“1”，则播放文件名开头为“0001”的文件，如参数为“13”，则播放文件名开头为“0013”的文件。

## 示例 21-1：红外遥控 MP3 播放器

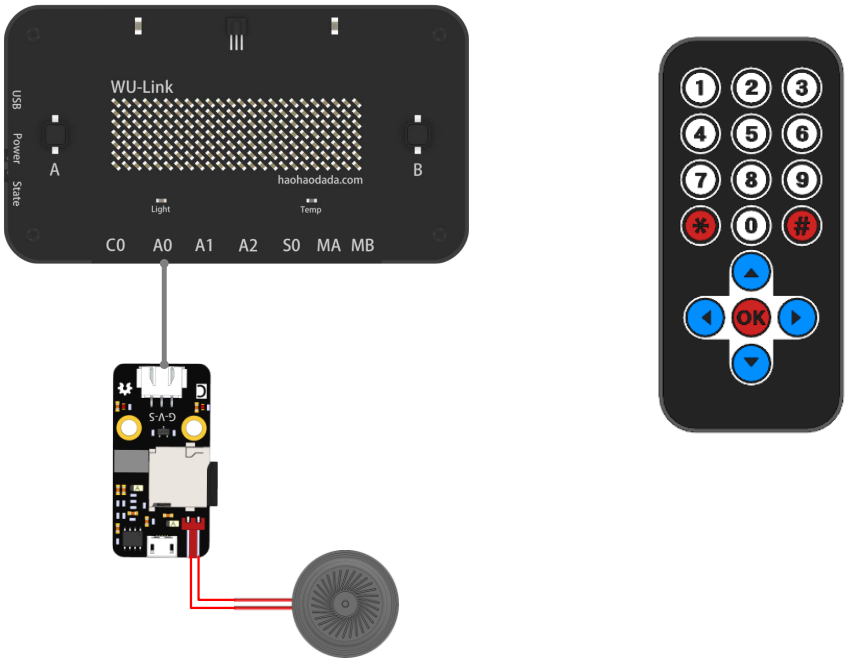
红外遥控器按键对应的 MP3 播放功能：

| 按键 | 模式 | 按键 | 模式 |
|---|---|---|---|
| 1～9 | 播放对应曲目 | OK | 播放 |
| * | 随机播放 | # | 循环播放 |
| ↑ | 音量加 | ↓ | 音量减 |
| → | 下一曲 | ← | 上一曲 |
| 0 | 停止 | | |

**元器件列表：**

1. WU-Link 主控板 ×1
2. MP3 模块 ×1
3. microSD 卡×1
4. 扬声器 ×1
5. 红外遥控器 ×1

**电路连接：**



程序编写:


播放控制相关程序


曲目选择相关程序

http://haohaodada.com/wulink/index.php?id=1899

# 第二十二节 炫酷点阵屏

点阵屏绘制图案、显示文本和显示数在第二节中已经介绍，在之后的多个案例中也经常被用到。本节将使用点阵屏坐标类指令，实现更炫酷的应用

**认识点阵写入点指令**


通过该指令中输入的位置可以选取要点亮的点。

**认识点阵屏删除点指令**


通过该指令中输入的位置可以选取要写灭的点。

**认识点阵显示点指令**


单独运行写入点指令或删除点指令不能直接点亮或更新点阵屏显示，必须运行点阵显示指令。

**示例 22-1：按坐标点亮和熄灭点阵点阵屏**

一行一行的把点阵点亮，又一行一行的把点阵熄灭，如此往复

**元器件列表：**

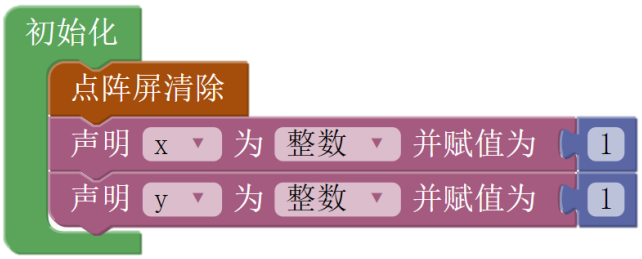
WU-Link 主控板 × 1

**电路连接：**

略

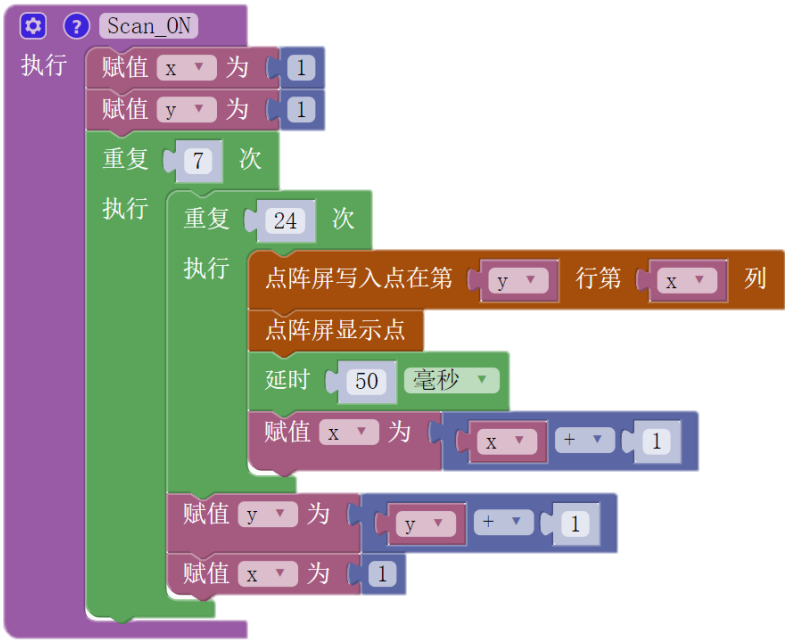
**程序编写:**

初始化程序：

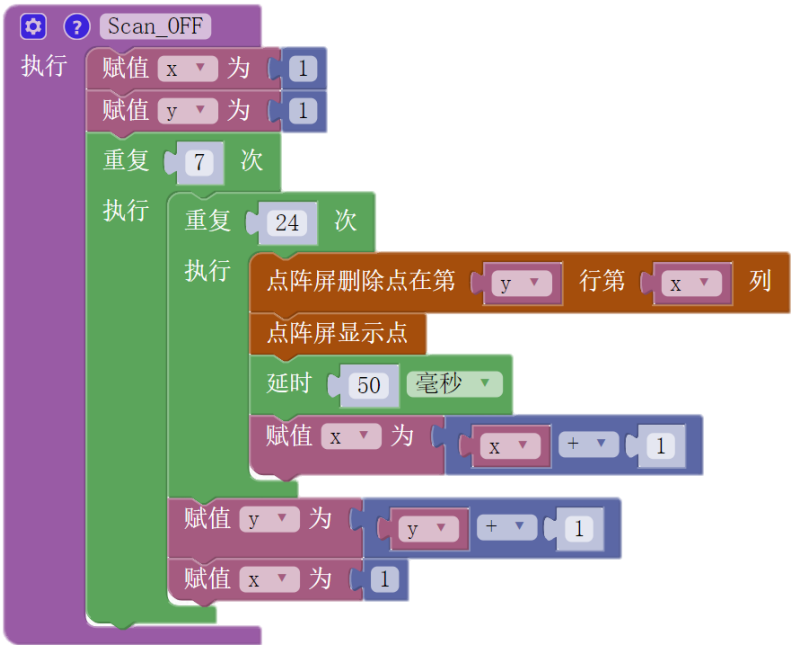



主程序：


逐行点亮函数：




逐行熄灭函数：




http://haohaodada.com/wulink/index.php?id=1929

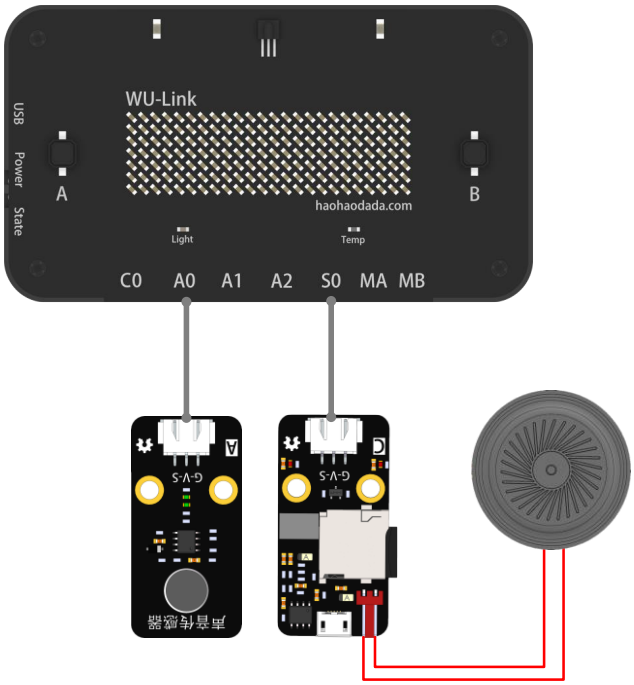
**示例 22-2：动感音乐播放器（柱状图）**

控制 MP3 模块播放音乐，利用声音传感器采集声音信号，将音乐的律动显示到点阵屏上。

**元器件列表：**

1. WU-Link 主控板 × 1
2. 声音传感器模块× 1
3. MP3 模块（含 SD 卡）× 1
4. 扬声器 × 1

**电路连接：**




**程序编写:**

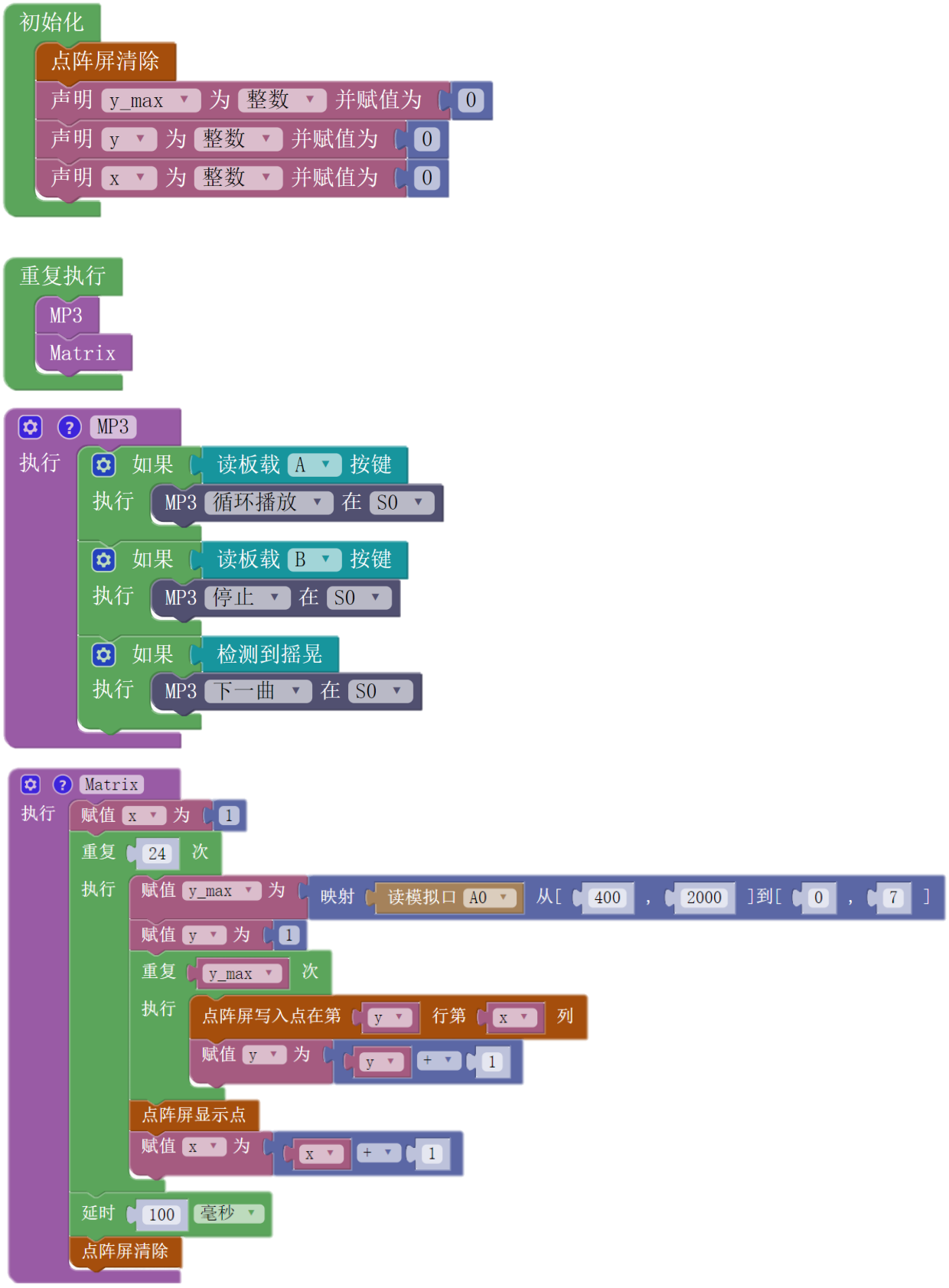
打开音频播放器，当然也可以把 MP3 模块集成进去用于发声。先用串口助手观察声音的波形情况。


安静时


播放音乐时

从波形看，最大值约为 2000，最小值约为 400。正好分为 7 个区间，与点阵模块 Y 坐标的 7 个点对应。

音乐动感点阵屏的程序如下：




示例程序地址：http://haohaodada.com/wulink/index.php?id=1930

# 第四部分

# 编程进阶

# 第二十三节 计时器

之前的项目案例多为单一功能，不存在多个功能之间产生冲突的问题。如一个复杂应用中需要有 LED 闪烁这个小功能，如果使用第三节中介绍 LED 闪烁程序的方法实现，等待指令将导致整个程序的响应能力变的极差。这节的内容将介绍解决复杂程序中的冲突问题。

**认识新指令——计时器指令**


计时器的值由主控芯片自动产生，单位为毫秒，按时间自动增加，完全不占用程序运行的时间等待指令的运行会使整个程序停止，如果这时输入有改变是读取不到的。这是两者最大的区别。

计时器指令需要配合两个变量一起使用，取名为 t1 和 t2。其中 t1 负责读取系统运行时间，t2 用于与 t1 进行比较。

如希望某事件发生一次的时间间隔为 1 秒（1000 毫秒）：

（1） 让 t1 的值与系统运行时间始终保持一致；

（2） 当 t1-t2 的值大于 1000（毫秒）时，则让 t2 的值更新为 t1 的值，否则不更新。

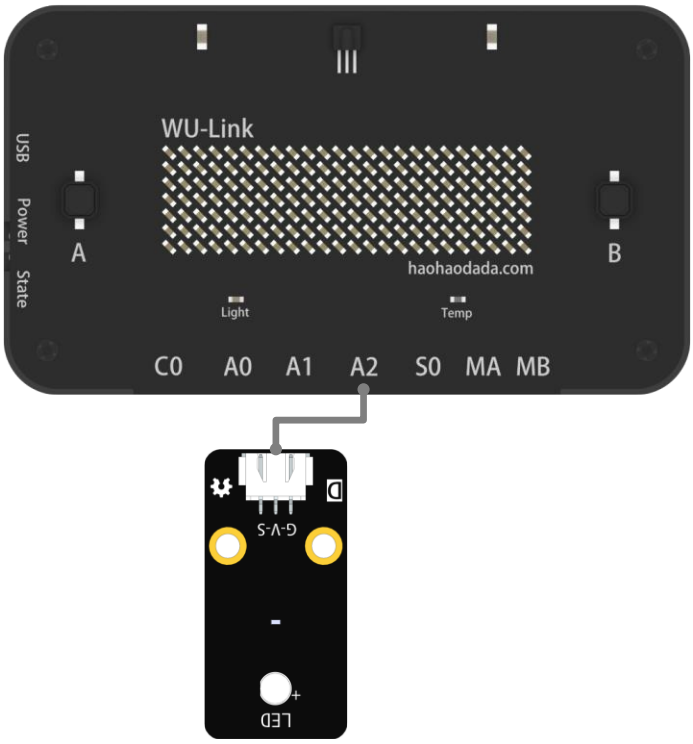
**示例 23-1：长按点亮 LED，松开熄灭 LED**

长按 A 按键 3 秒以上才能点亮 LED，短按按键则马上熄灭 LED。

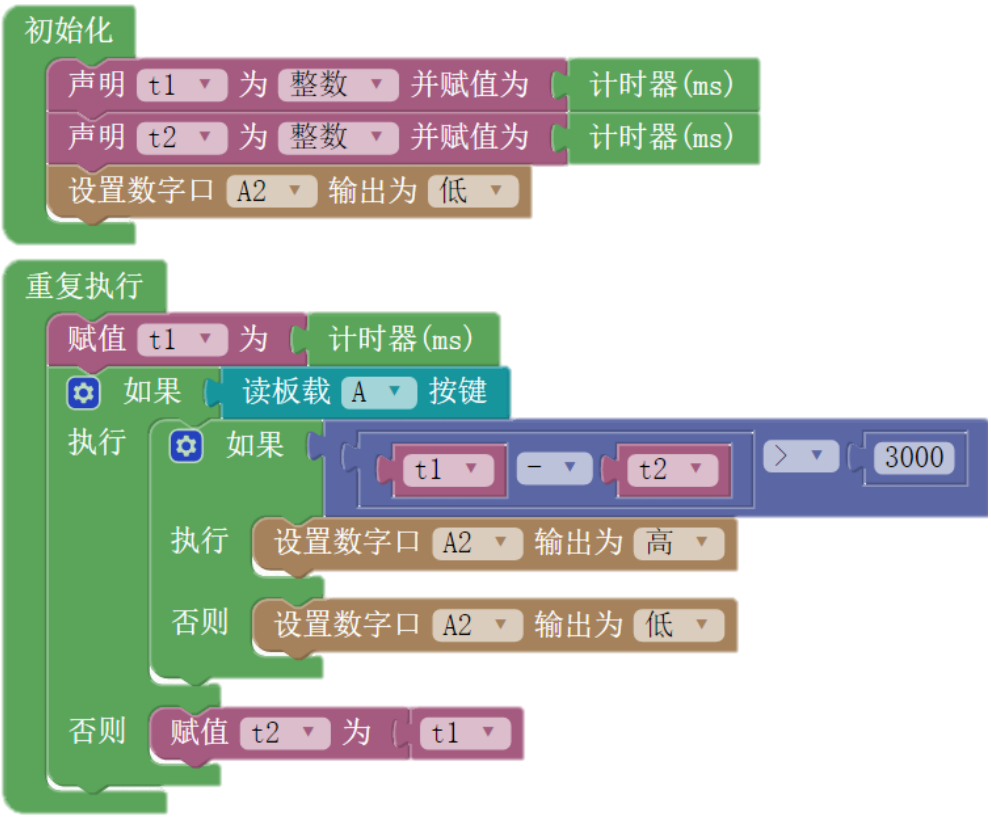
**元器件列表：**

1. WU-Link 主控板 × 1
2. LED 模块 × 1

**电路连接：**




**程序编写：**




示例程序地址：http://haohaodada.com/wulink/index.php?id=1823

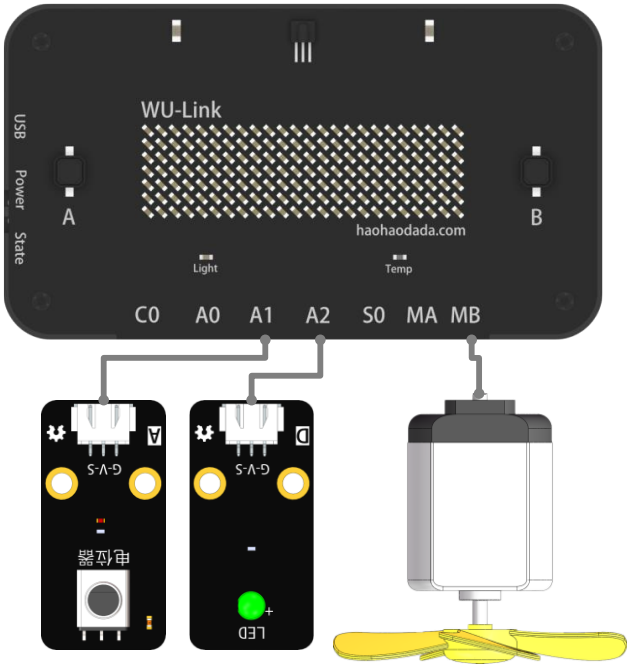
**示例 23-2：LED 闪烁的同时，用电位器控制风扇的转速（等待指令）**

用等待指令编写程序

**元器件列表：**

1. WU-Link 主控板 ×1
2. 电机 ×1
3. 风扇叶片 ×1
4. 电位器模块 ×1
5. LED 模块 ×1

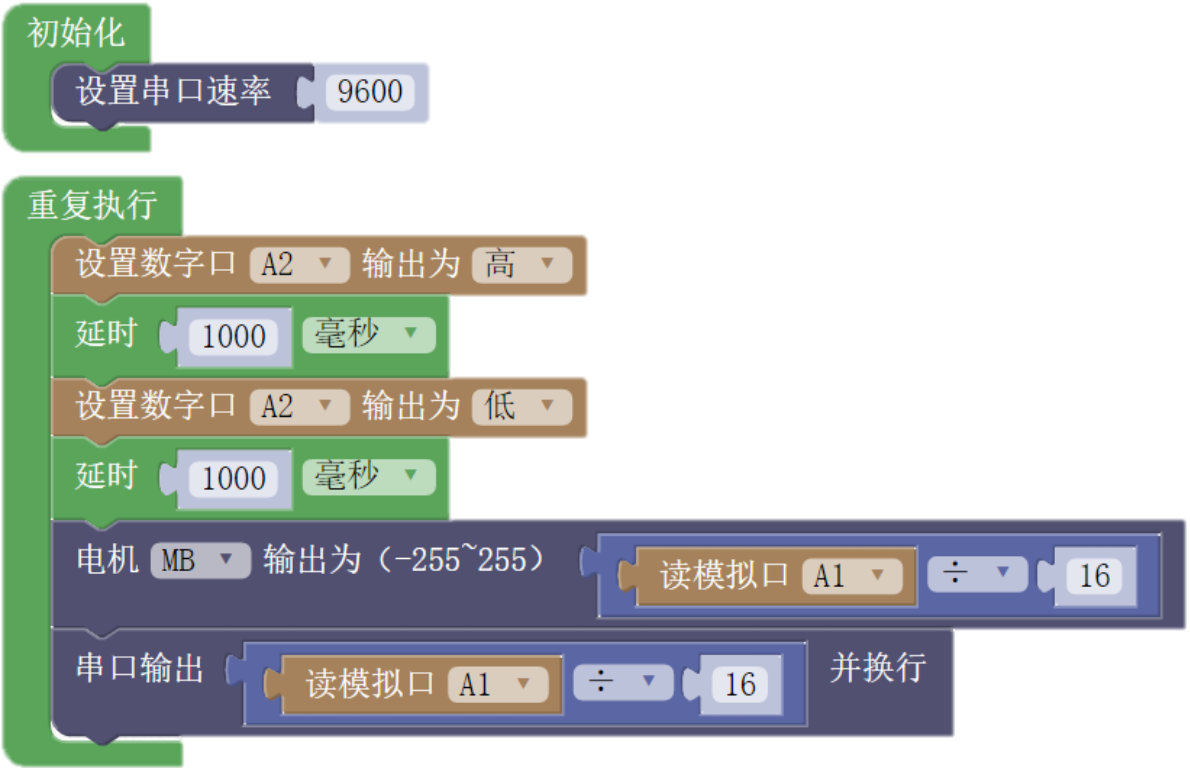
**电路连接：**




**程序编写:**

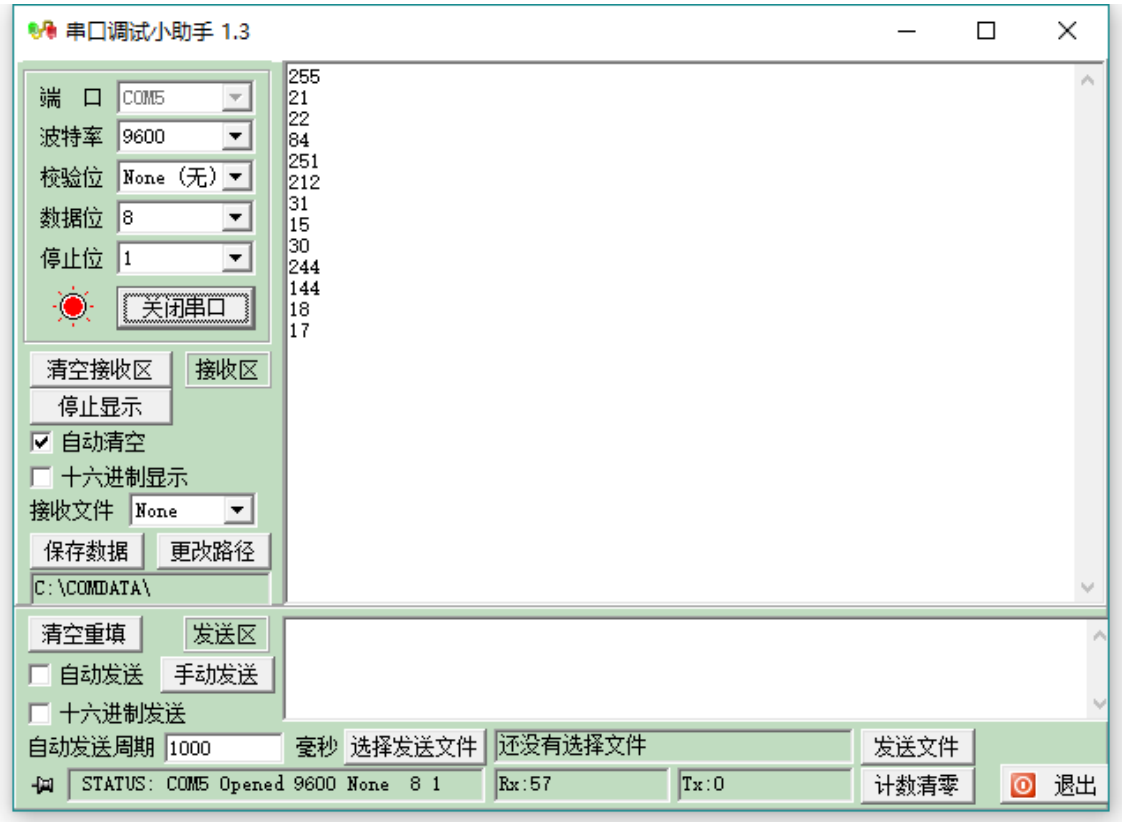
用等待指令实现 LED 的闪烁:

在加上电位器控制风扇的程序，同时用串口助手将电位器的值打印出来：




**程序运行结果：**

打开串口助手，观察电位器的值。



发现电位器的值每两秒打印一次，风扇转速变化也不平滑。出现这个结果正是因为等待指令运行时，电位器的变化是读取不到的。接下来用计时器指令实现 LED 闪烁试试看。

**示例 23-3：LED 闪烁的同时，用电位器控制风扇的转速（系统运行时间）**

用计时器指令编写程序

**元器件列表：**

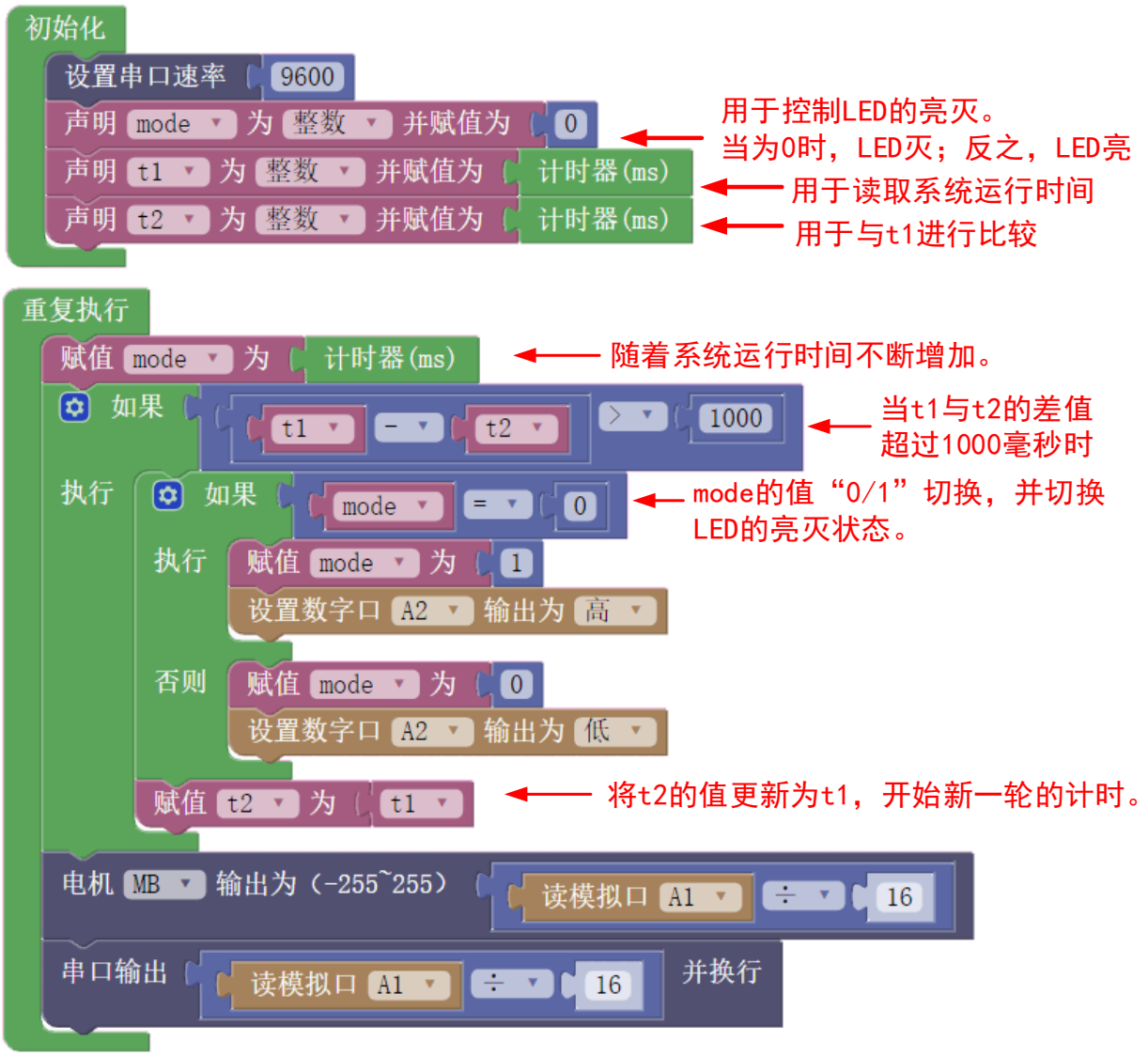
同示例 23-2。

**电路连接：**

同示例 23-2。

**程序编写：**

用计时器指令实现 LED 的闪烁，需定义 mode、t1 和 t2 三个变量。




示例程序地址：http://haohaodada.com/wulink/index.php?id=1824

**程序运行结果：**

打开串口监视器，观察电位器的值，这时可以看到打印值刷新非常快。电机转速的控制也非常流畅。


在复杂程序中，应避免使用等待指令，所有需要用到时间间隔的程序，都可以用计时器指令实现。

# 第二十四节 输入计数及其应用

要实现诸如记录按键按下次数，或通过按下松开按键一次切换输出器件状态，需要巧用变量来实现。

**示例 24-1：记录按键按下松开的次数，并显示到点阵屏上**

按下 A 按键一次，不论按下多长时间，只要不松开，点阵屏上的数字只增加 1。

**元器件列表：**

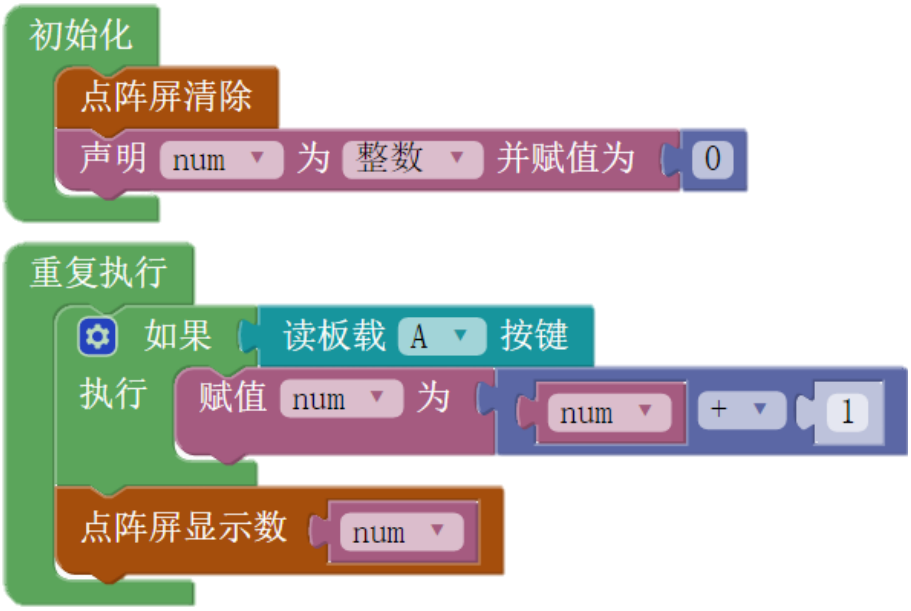
1. WU-Link 主控板 ×1

**电路连接：**

略

**程序编写：**

大家可能首先想到的是下面这样的程序：

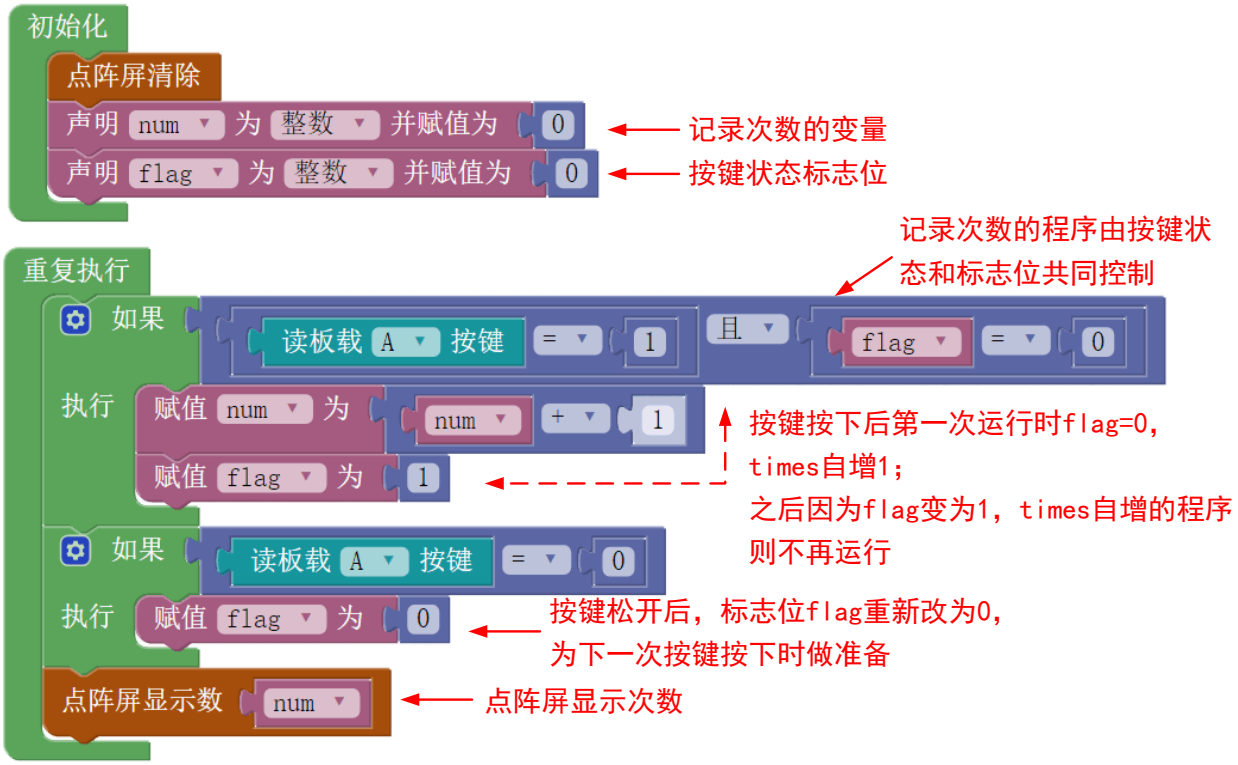



这个程序的运行结果是按下一次，数码管显示的数字增加的非常快。即使手速再快，每按一次，数码管显示的数值都会至少增加几十。

大家应该都能分析出程序错误的原因：按键被按下的时间里，程序运行了很多次，变量 num 自增了很多次。

所以，问题的关键在于按键被按下时，如何让计数程序只运行一次。

这需要声明一个变量，用于记录按键的松开和按下状态，工程上这样的变量被称为“标志位”。



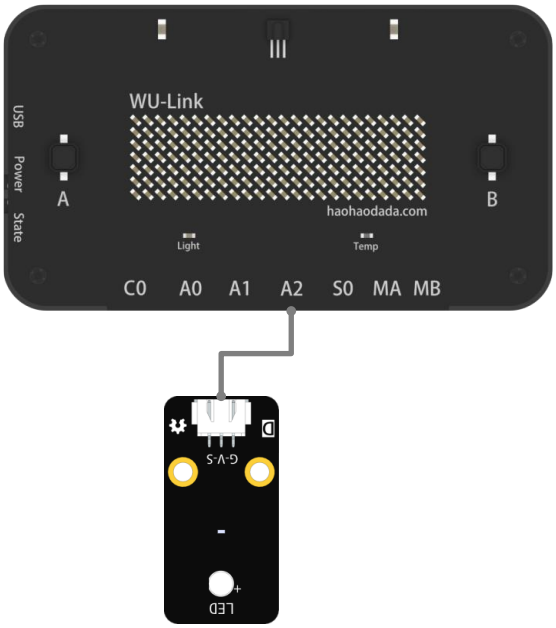

**示例 24-2：点控 LED**

A 按键按下松开一次，LED 亮；再按下松开一次，LED 灭；如此往复。

**元器件列表：**

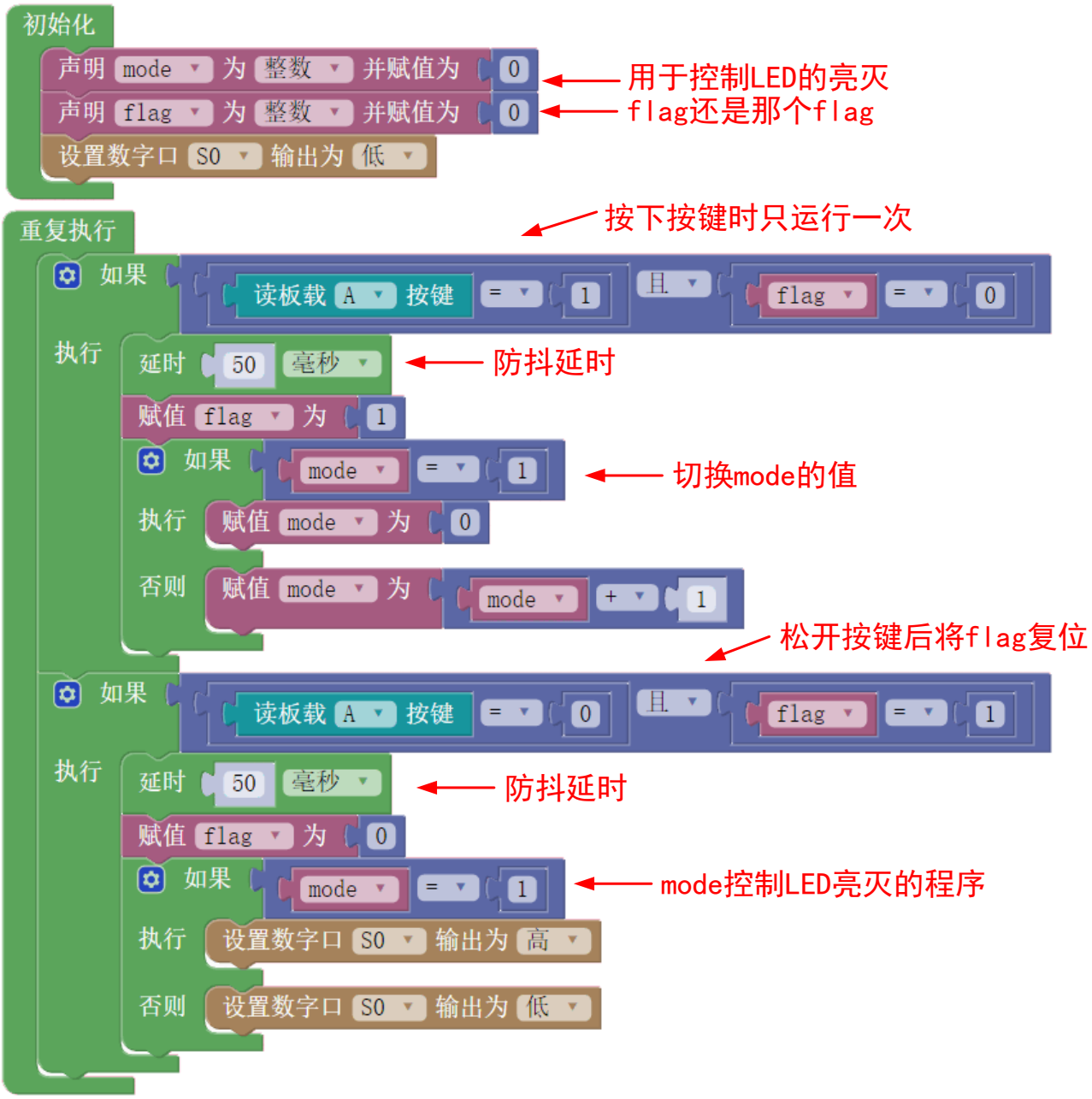
1. WU-Link 主控板 ×1
2. LED 模块 ×1

**电路连接：**

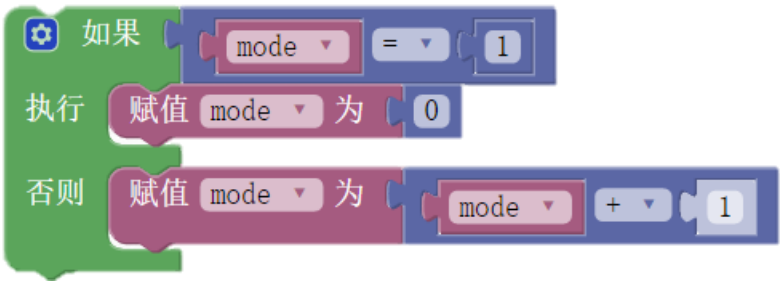


**程序编写：**

根据示例 24-1，可以轻松的理解下面的程序：




其中，按键按下部分的程序如下：



这是更通用的方式，当按键控制的模式不只有亮和灭两种状态时，依然可用。具体应用请看示例 24-3。

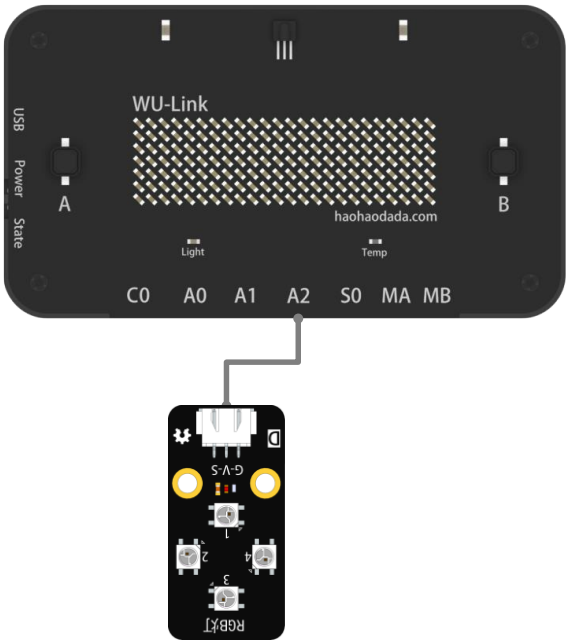
**示例 24-3：按键控制切换 RGB 灯的颜色**

按键按下松开一次，RGB 灯的颜色切换一次。这种应用在家里的吸顶灯中非常常见。

**元器件列表：**

1. WU-Link 主控板 × 1
2. RGB 模块× 1

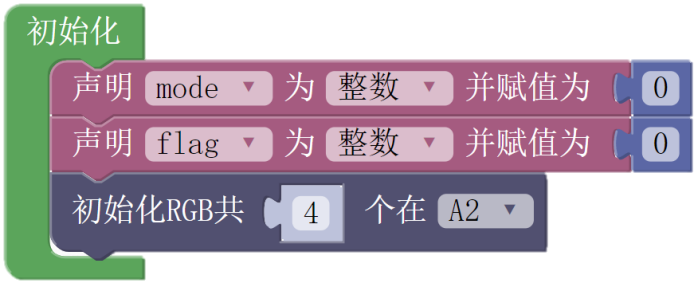
**电路连接：**



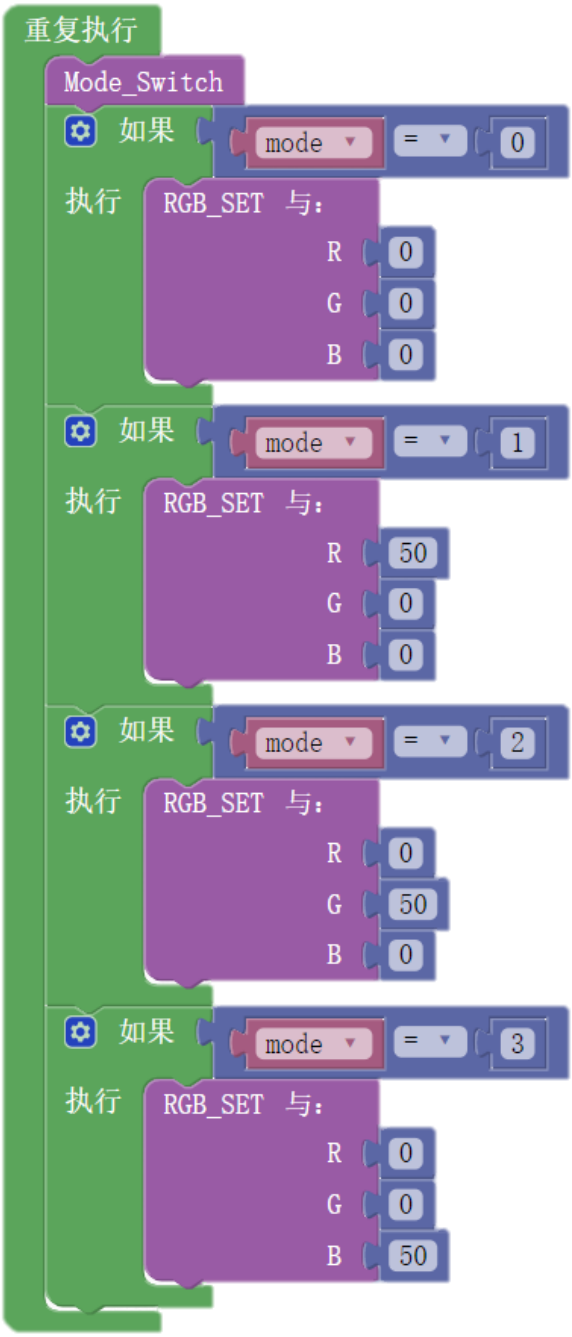

**程序编写：**

只需在示例 24-2 程序的基础上稍作修改即可。

初始化程序：

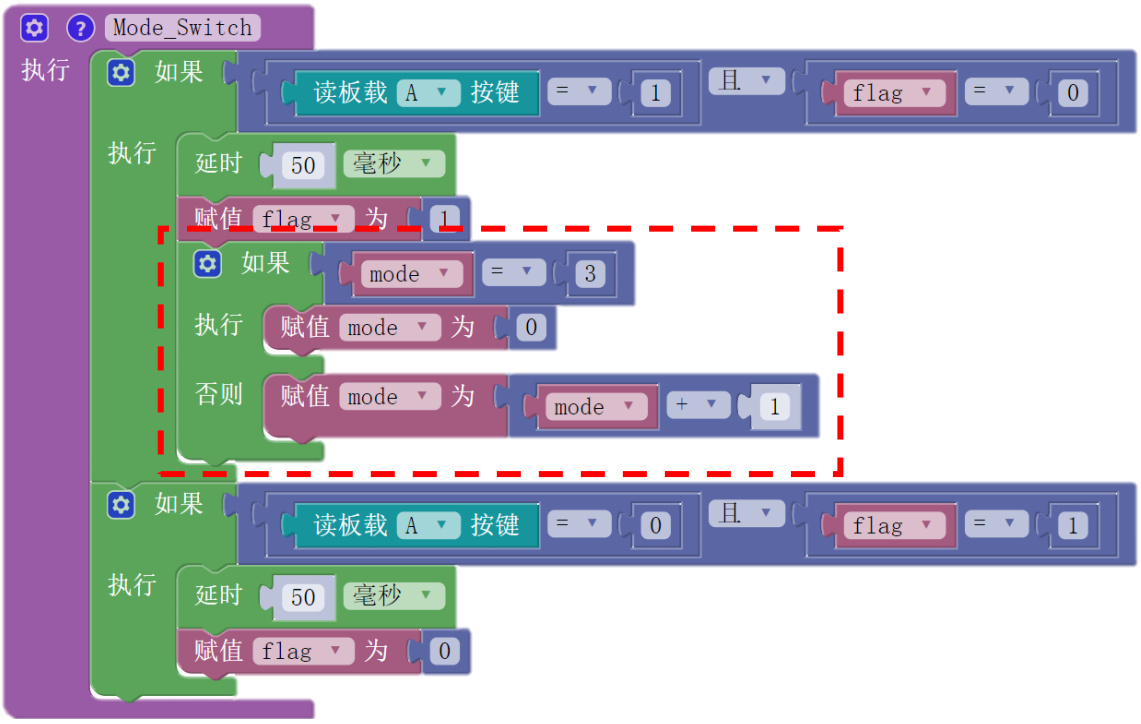


主程序：




同时点亮 RGB 模块 4 个灯的函数：

模式切换函数：




示例程序地址：http://www.haohaodada.com/wulink/index.php?id=1931

注意虚线红框的程序，与示例 24-2 的程序基本相同，只是 mode 的取值范围变为了 0、1、2、3。

以上程序中的输入计数用的都是按键这种数字量输入信号，如果换成模拟量信号呢？聪明的同学肯定已经想到办法了。

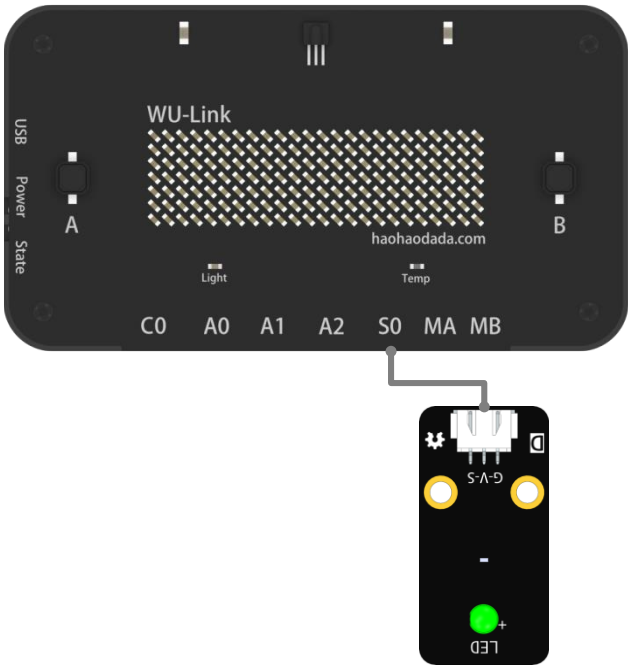
**示例 24-4：亮度传感器控制 LED 灯的亮灭（点控）**

摸一次板载亮度传感器，LED 亮，再摸一次，LED 灭，如此往复。

**元器件列表：**

1. WU-Link 主控板 × 1
2. 亮度传感器模块 × 1
3. LED 模块× 1

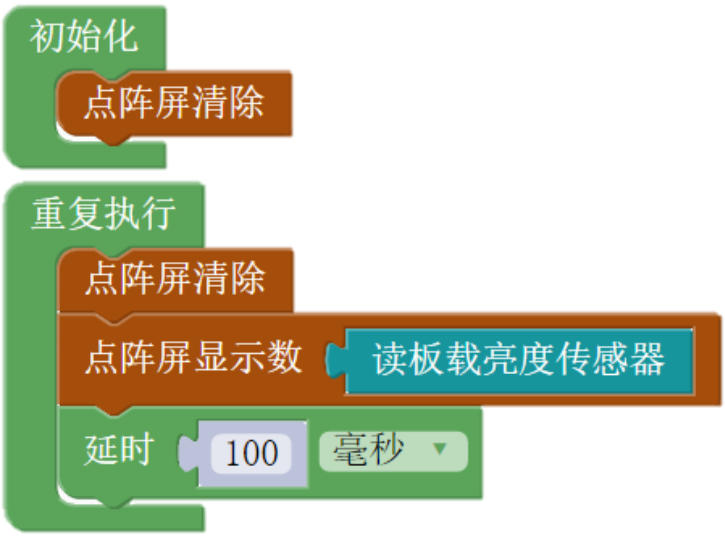
**电路连接：**




**程序编写:**

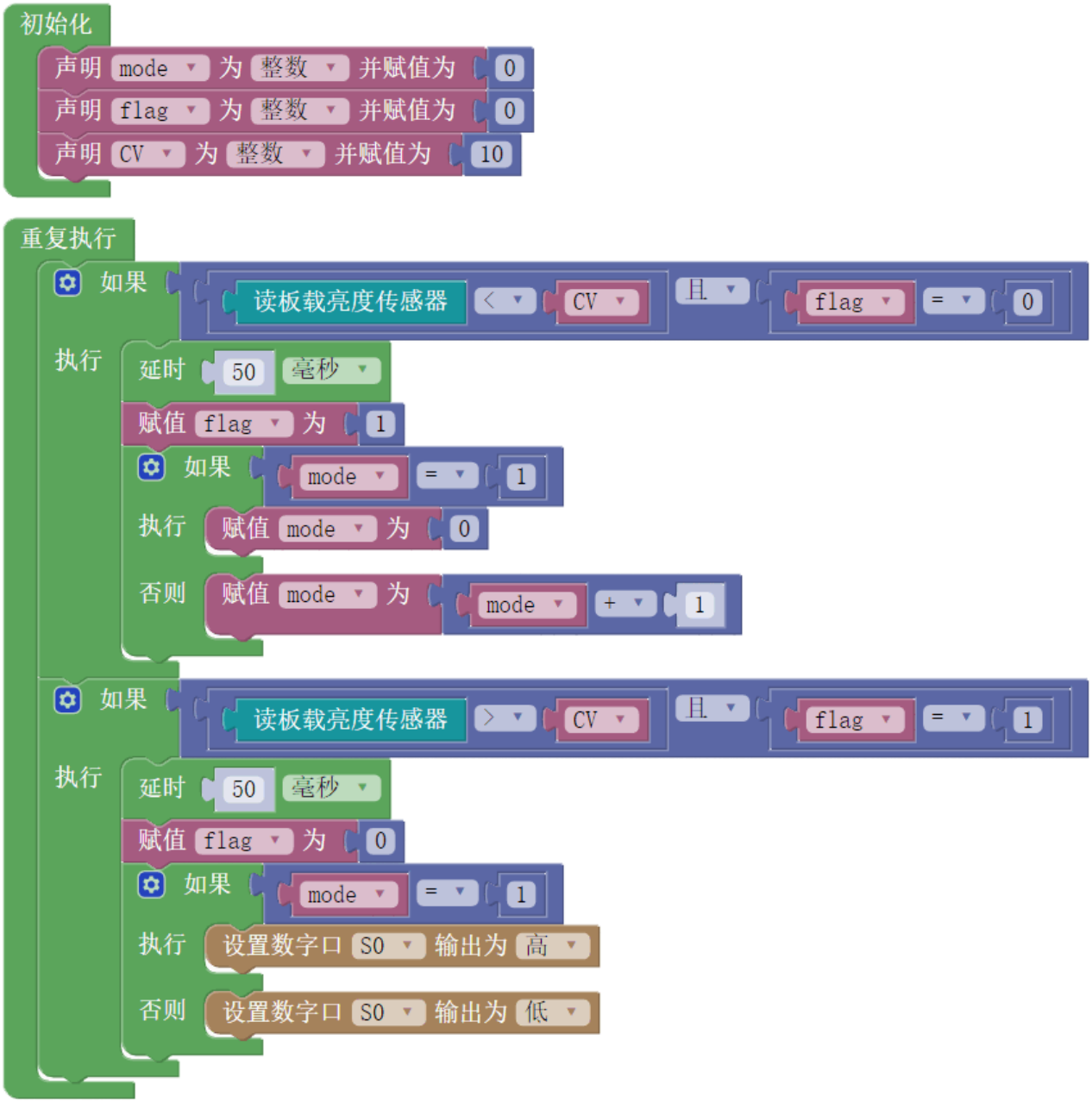
只需在示例 24-2 程序的基础上稍作修改，即将按下按钮替换为亮度值小于阈值；将松开按钮替换为亮度值大于阈值即可。所以需要先测定这个阈值。

定义一个变量 CV 为认定亮度传感器是否被触摸的阈值（Critical Value），这个阈值应该设为多少，需要做一次测定。通过点阵屏显示板载亮度传感器的数值，检测手按住和松开时的亮度值。




阈值的选取可以从 10 到 20 之间选取一个，本例中选取的阈值为 10。

那么程序为：




示例程序地址：http://haohaodada.com/wulink/index.php?id=1829

**通过以上四个案例可以总结出：利用“标志位”的方法，可以让持续型信号触发一次跳变信号，避开重复运行程序导致信号多次跳变的问题。**

在第二十一节中，初步介绍了 MP3 模块的使用，其示例 21-1 是用红外遥控方式控制 MP3 模块，因为红外遥控信号不是持续型信号，所以重复运行程序不会造成任何影响。接下来将介绍按键、传感器这类持续型信号如何控制 MP3 模块。

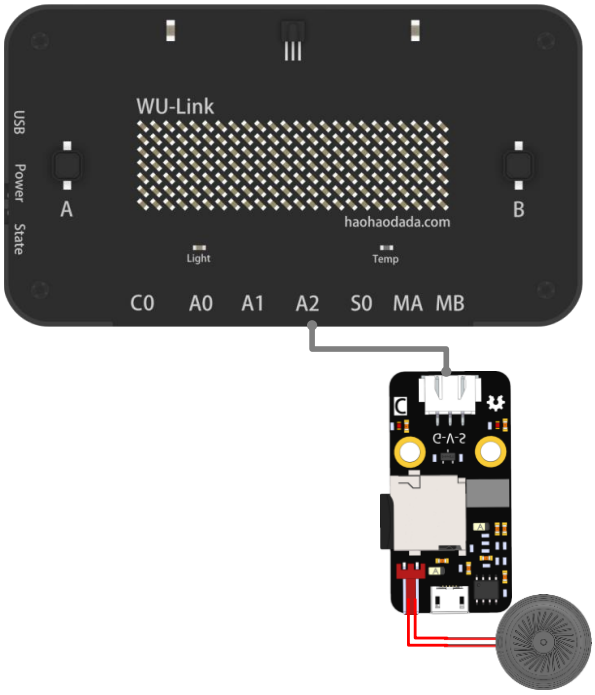
**示例 24-5：用两个按键控制 MP3 的上一曲/下一曲切换**

按一次按键 A，切换为上一曲；按一次按键 B，切换为下一曲。

**元器件列表：**

1. WU-Link 主控板 ×1
2. MP3 模块 ×1
3. microSD 卡×1
4. 扬声器 ×1

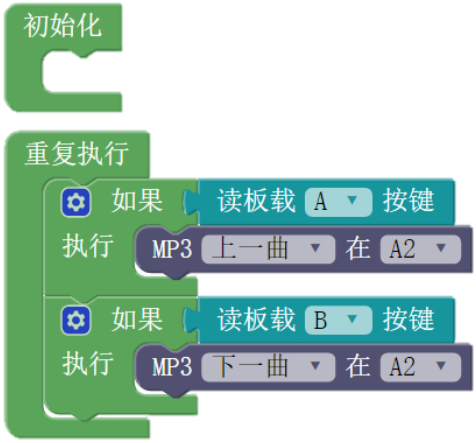
**电路连接：**



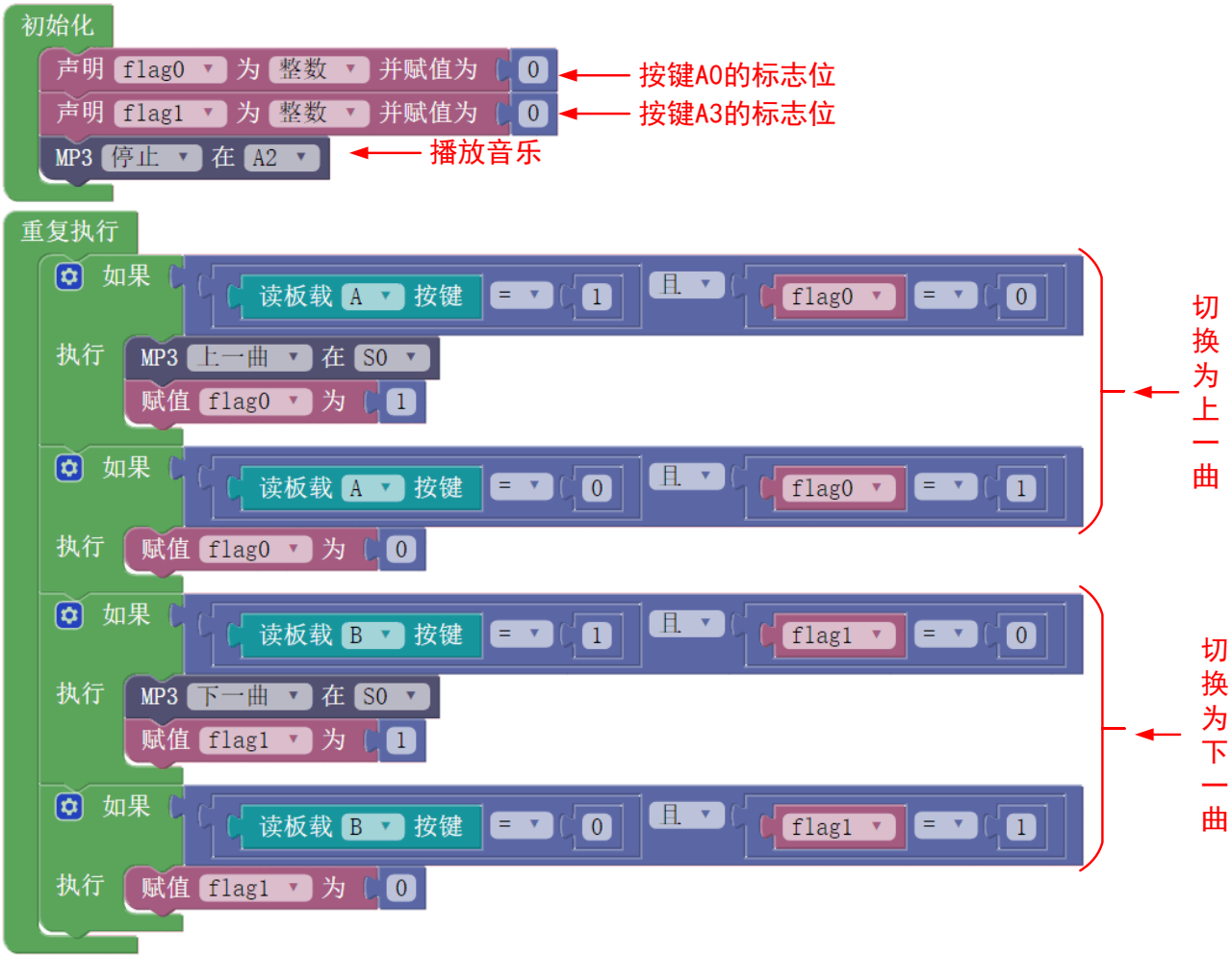

**程序编写:**

利用 MP3 操作指令


最容易想到的程序肯定是：



回想示例 24-1 记录按键按下松开次数的程序，就会发现以上程序的问题，按一次按键 A0，“上一曲”将会运行很多次，按一次按键 A1 同理。如果要实现按下松开按键一次，歌曲只向前或向后切换一次，那么可以仿造示例 24-2 的方式来实现。

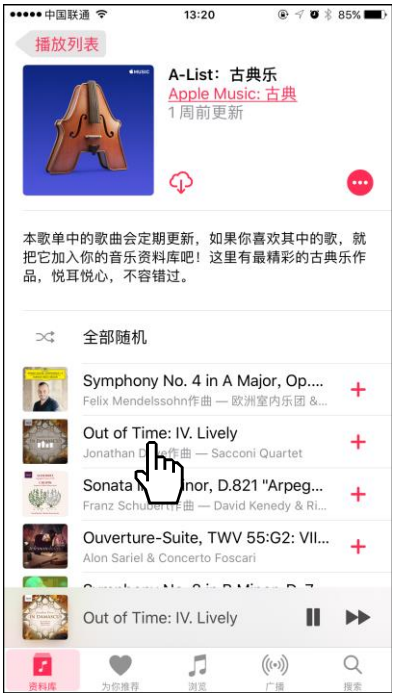



示例程序地址：http://haohaodada.com/wulink/index.php?id=1830

除了“上一首”和“下一首”切换歌曲之外，MP3 模块的按曲目播放和音量加减调节程序块同样不能直接调用，反复运行。

音量加和音量减为什么不能直接调用，原因与上一曲/下一曲一样，会在短时间内运行很多次。

指定曲目播放为什么不能直接调用呢？我们来做一个实验，打开你手机上的音乐 APP，反复点击选取其中一首歌曲，歌曲的进度条是会反复会到歌曲的 0 分 0 秒处。正确的做法是点击选择一次，之后不再点击。




同样，使用“指定曲目播放”程序块时，也应该是运行一次之后不再运行，而不是反复运行。

# 第二十五节 数组

经过之前的大量案例，相信大家对变量的作用有了基本的了解。当需要很多变量，且这些变量是相关的，那么用数组将这些变量归类，并利用数组的方式存取变量值，将极大的简化程序。

数组，相当于一组带有编号的变量集合，编号本身也可以作为变量处理。

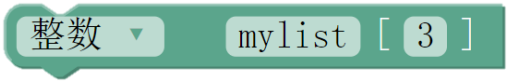
**功能描述：**

所谓数组，就是相同数据类型的元素按一定顺序排列的集合。

数组在程序中最大的作用是，可以把数据一个一个的放入到有编号的格子里，取用的时候也可以直接根据编号来调取。如下图 0 号格存放了数字 255、1 号格存放了数字 127 等等。

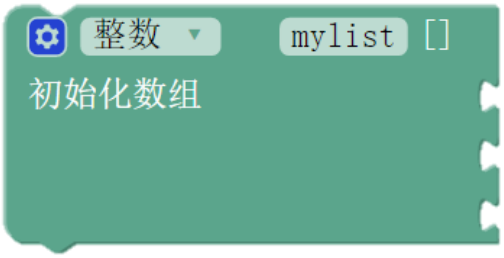
| 数组名 | 255 | 127 | 191 | 223 | 95 | 159 |
|---|---|---|---|---|---|---|
| 元素标号 | 1 | 2 | 3 | 4 | 5 | 6 |

**认识新指令——数组初始化指令**




该指令用于初始化数组，放在初始化程序中。和变量一样，需选择正确的数据类型。数组名和数组长度均可自定义，数组名不能为中文，数组长度不能为 0。

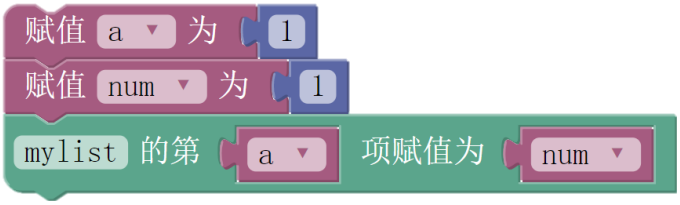
**认识新指令——数组初始化赋值指令**




该指令可以在初始化的同时进行赋值，数组长度为添加的元素个数。可以点击“齿轮”按钮进行元素的添加和删减，操作与函数的参数添加删减相同。

**认识数组元素赋值指令**


标号和元素值都可以用变量来替换。




**认识数组元素读取指令**


用于读取某个数组元素的值，在参数框中填入数值。

数组元素的赋值和读取，其编号和元素值都能代入变量，而变量能很方便的进行自增减，这样便给程序带了可迭代性，这是多个无前后关系的变量所不具备的功能。

**示例 24-1：抽签机随机取数不重复**

按下松开一次按键，点阵屏显示一个数字，每次显示的数字均不重复。显示完毕之后点阵屏显示 8888。

**元器件列表：**

1. WU-Link 主控板 ×1

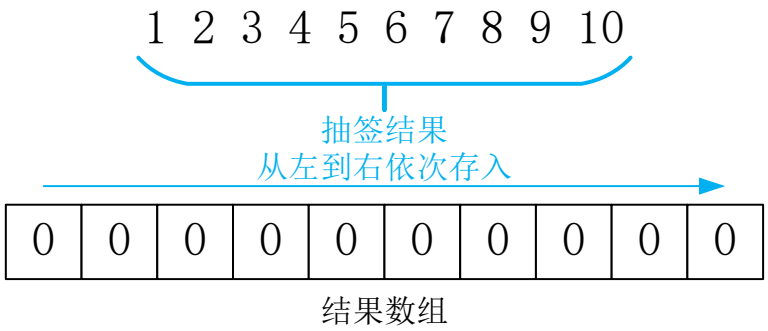
**电路连接：**

略：

**程序编写:**

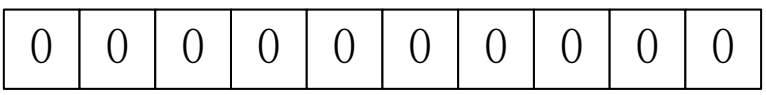
**方案 1：检查法**

**方法概述：**

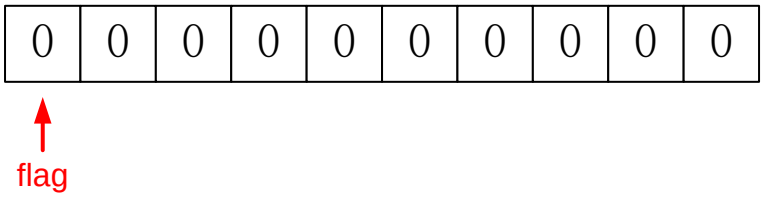



**程序分析：**

第一步，建立一个 10 位的空数组：

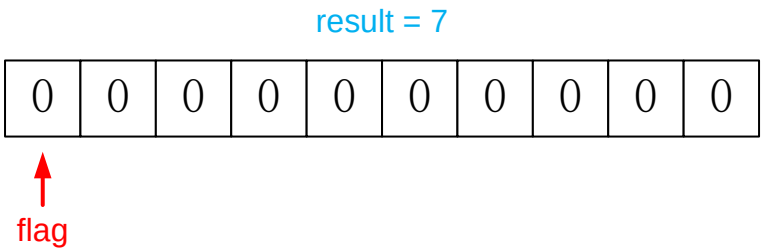


第二步，定义一个变量 flag，用来指向数组的一个元素，初始时指向标号 0 的元素：

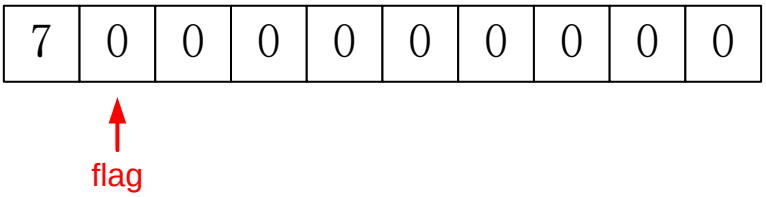



第三步：按下按钮，只要不松开，不断的取出随机数，将结果暂时存放在一个变量 result 里，将变量 result 去与数组中从标号 0 元素到标号 flag-1 的元素一一比较。

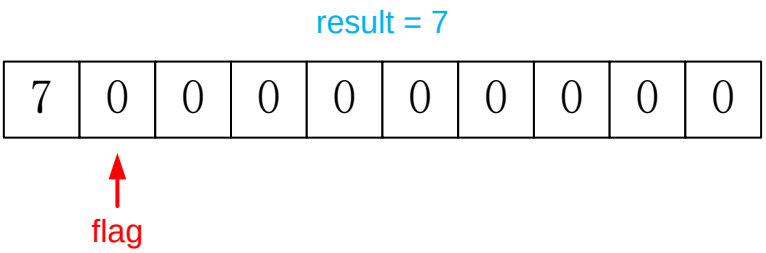
**推演：** 当 flag=0 时，标号 0 元素的值为 0，result 的值为 1~10 中的一个，必然不相等，将 result 的值存入数组标号 0 元素中，假设其值为 7。



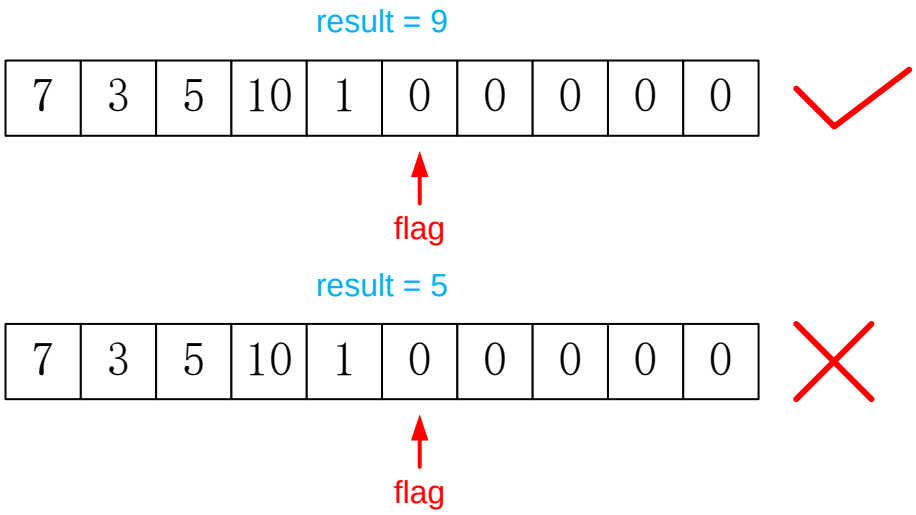

flag 加 1，指向后一位，为下一次抽取做准备。




此时 flag=1。按下按键，result 继续从 1~10 中随机取出一个，如果 result 取中的仍然是 7，则这个结果是不符合要求的，必须重新抽取。问题是如何判断结果是否符合要求？




当 flag=1 时，result 要与标号 0 元素进行比较，如果相等，则不符合要求，继续抽取；当 flag=6 时，result 要与标号 0 元素到标号 5 元素一一比较。推而广之，即当 flag=n 时，result 需要跟数组前 n 个元素都进行比较，确认 result 与它们都不同，才是符合要求的结果。




可以用一个循环程序块，让 result 去与标号 0 的元素比较，一直比较到标号 flag-1 元素。

定义一个变量 check，初始值为 0，result 每比较一个元素，如果不相等，则 check 的值加 1，如果相等，则 check 的值不变。那么如果 result 的值与已抽出数字全部不同时，则 check 的值等于 flag+1，如果 result 的值与已抽出数字有相同时，则 check 的值必然不等于 flag+1。所以 check 的结果可以用来判断抽出的 result 是否符合要求。

最后，当 10 个数字全部抽取出来之后，数码管可以显示 8888，已提醒用户抽签结束。

**图形化程序如下：**

初始化，定义相关变量和数组


主程序：

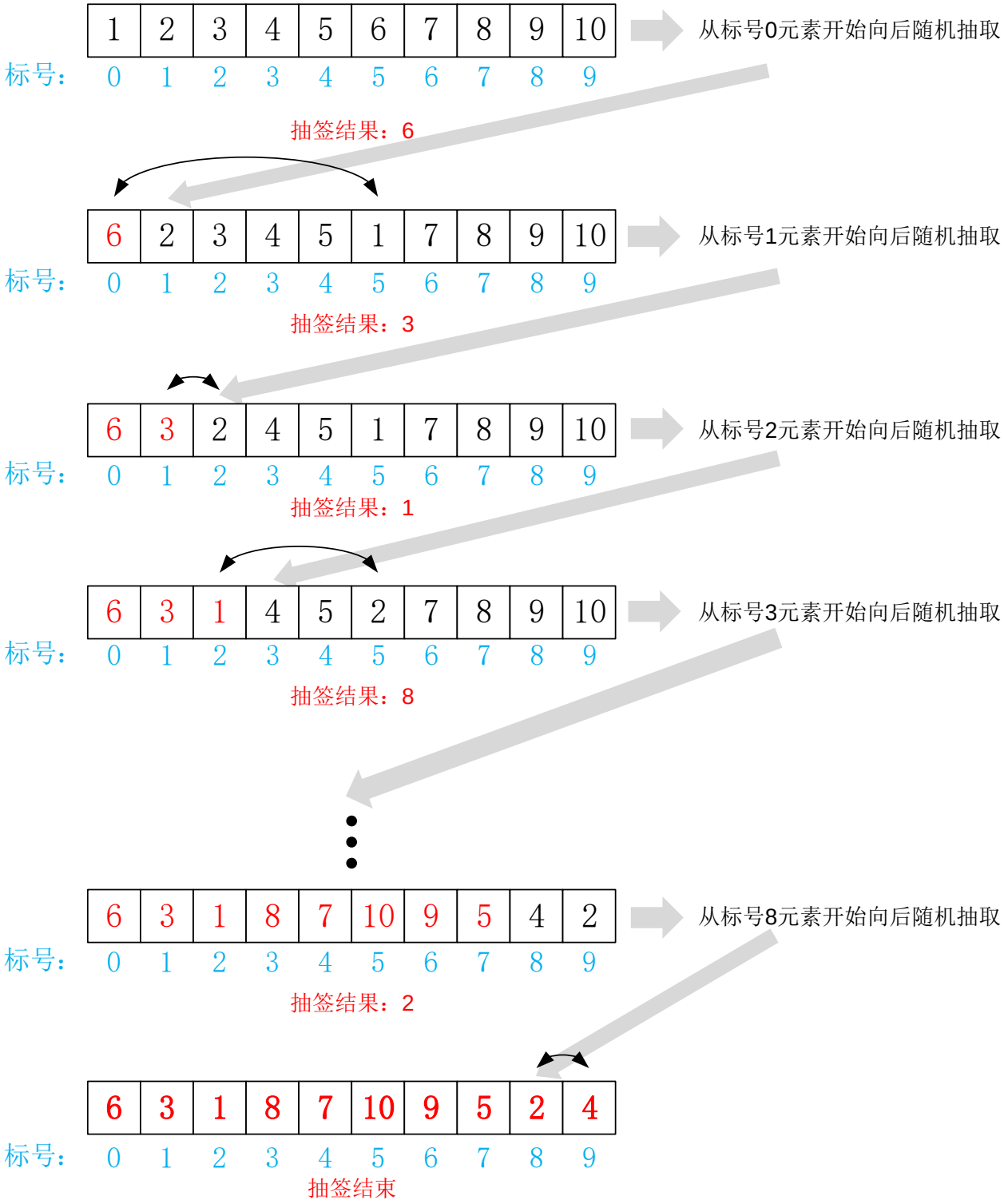

示例程序地址：http://www.haohaodada.com/wulink/index.php?id=1832

**项目要求二——方案 2：交换法**

**方法概述：**

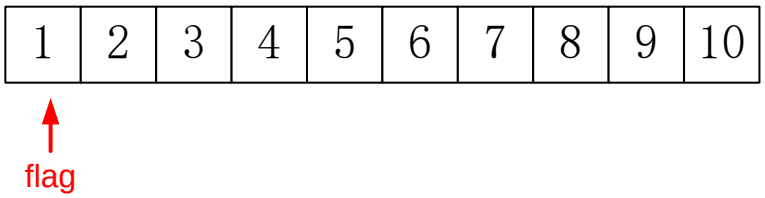
先在数组按顺序存入 1～10 十个数字；

再不停的抽取数字，，并进行调换。举例说明：

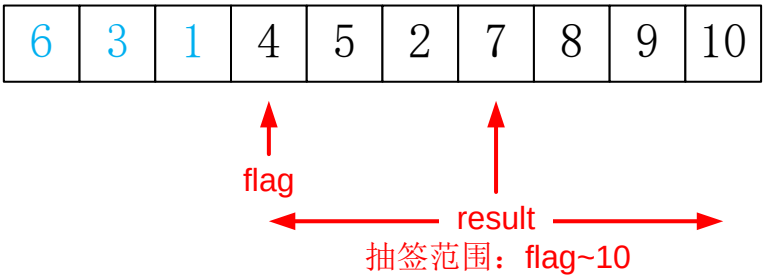


**程序分析：**

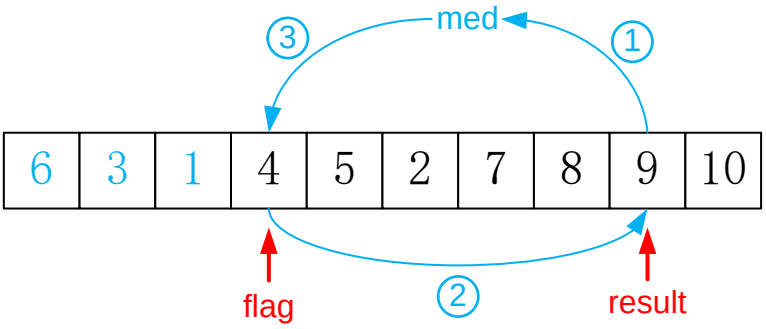
第一步，建立一个 10 位数组，把 1~10 十个数字按顺序存放进去。同方法 1 一样，定义一个数组标号变量 flag，初始值为 0。




第二步：按下按钮，只要不松开，不断的从第 flag 元素到第 10 个元素中取出随机数，存入变量 result，并把数组中标号为 result 的元素显示在数码管上。




第三步：交换数值，定义一个中间变量 med 来帮助实现 flag 和 result 指向的元素进行交换。

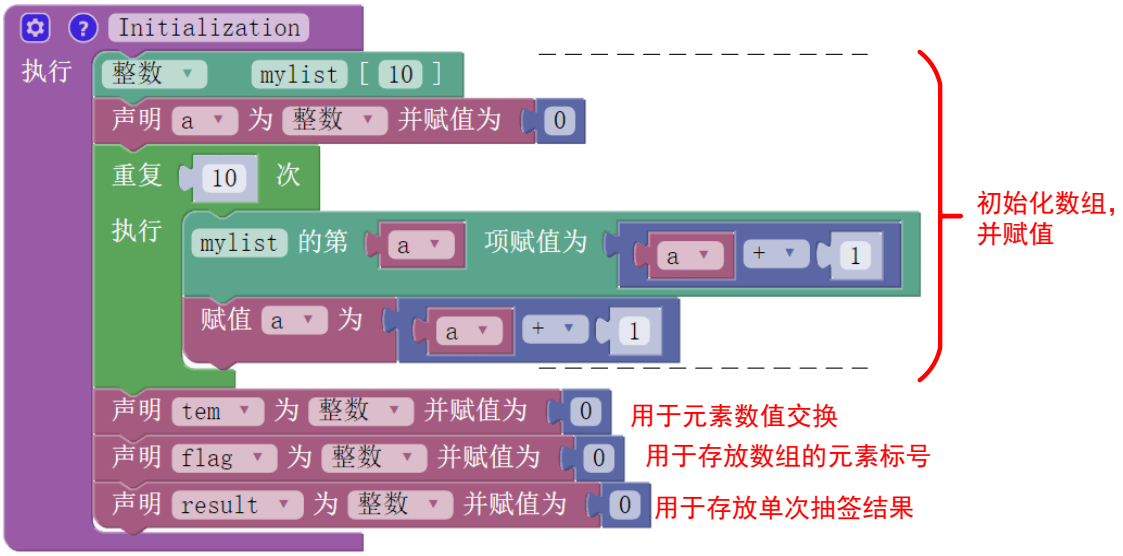



第四步：flag 的值加 1，进行下一次的抽取。

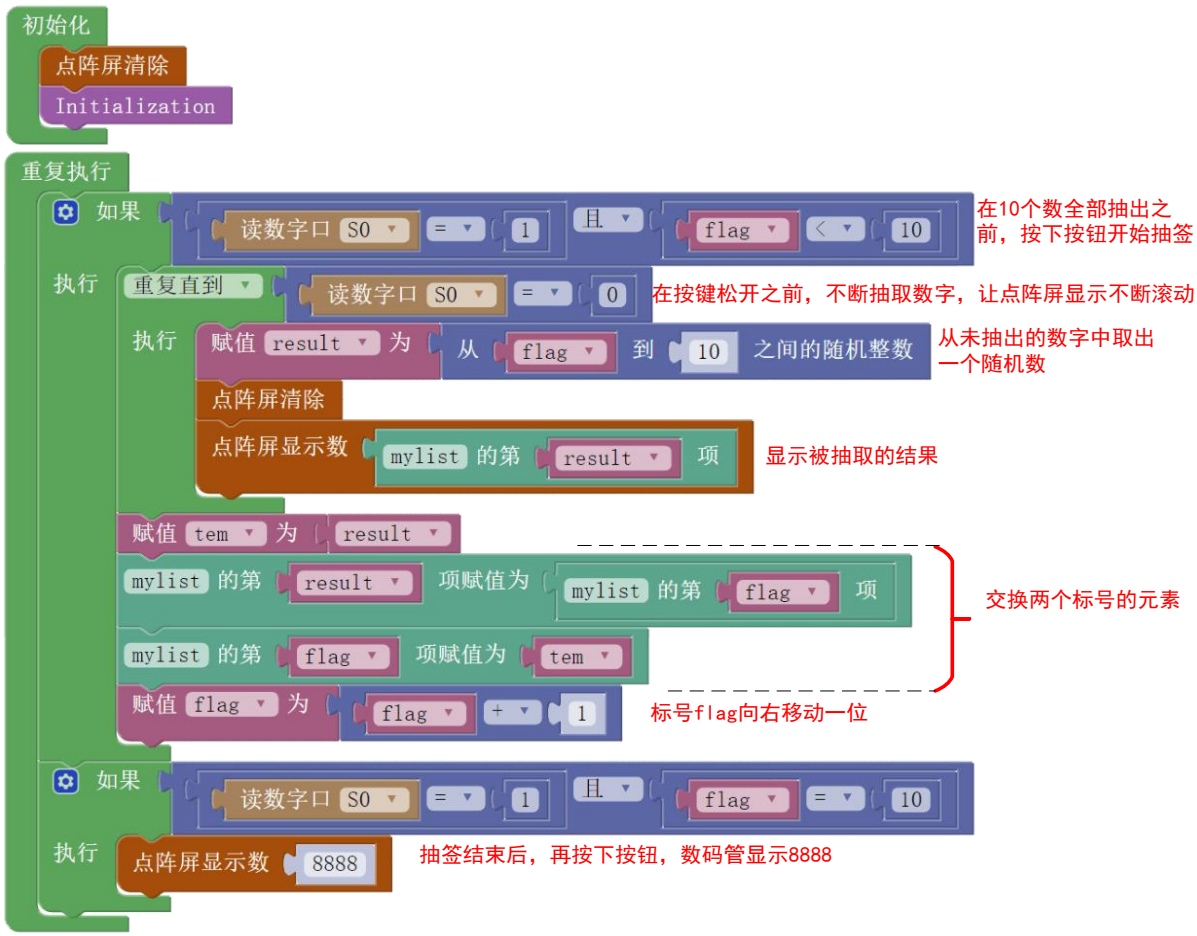
第五步：全部抽取结束，flag 的值为 11，再按一下按钮，数码管显示 8888。

**图形化程序如下：**

数组初始化函数：




主程序：




示例程序地址：http://haohaodada.com/wulink/index.php?id=1839

# 第五部分

# 物联网通讯

# 第二十五节 物联网基础知识介绍

之前的课程内容均是基于 WU-Link 的单机应用，信息的采集、处理、输出均由单个控制核心完成，接下来的内容将介绍多控制核心工作时如何通过物联网传递信息。

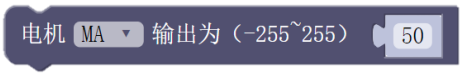
**物联网**

物联网就是物物相连的互联网。英文名称是：Internet of things，英文缩写是“IoT”。物联网的核心和基础仍然是互联网，但在互联网的基础上扩展和延伸到了任何物品，在物品与物品之间通讯和信息交换。物联网是继计算机、互联网之后信息技术领域的第三次重大变革。

**数据类型**

在计算机行业，工程师会将各种数据进行分类，以方便程序的处理。数据类型分很多种，这里不做一一赘述，只介绍两种物联网通讯中用到的两种数据类型：数值和字符串

**数值：**单个数值信息只包换一个数值，它可以用于数学运算、逻辑运算。在之前课程中传感器采集的数据、驱动器的设置参数均为数字型信息，如：


**字符串：**由英文字母、数字、符号组合而成，其优点是单条信息能包含的信息量更丰富，且可读性更强。

如小明向小红说分别传递两条数值型信息和两条文本型信息：

| 信息 | 数值 | 字符串 |
|---|---|---|
| 第一条 | 12 | Jim is 12 years old |
| 第二条 | 15 | Peter is 15 years old |

同样是传递两条信息，如果是数值，接收方会不知道数值的具体指代，所以在物联网通讯中不可避免的将用到字符串。

**消息**

在物联网中传递着各种各种各样的信息，有数值、“变量”、字符串，在 WU-Link 编程环境中统称为消息。

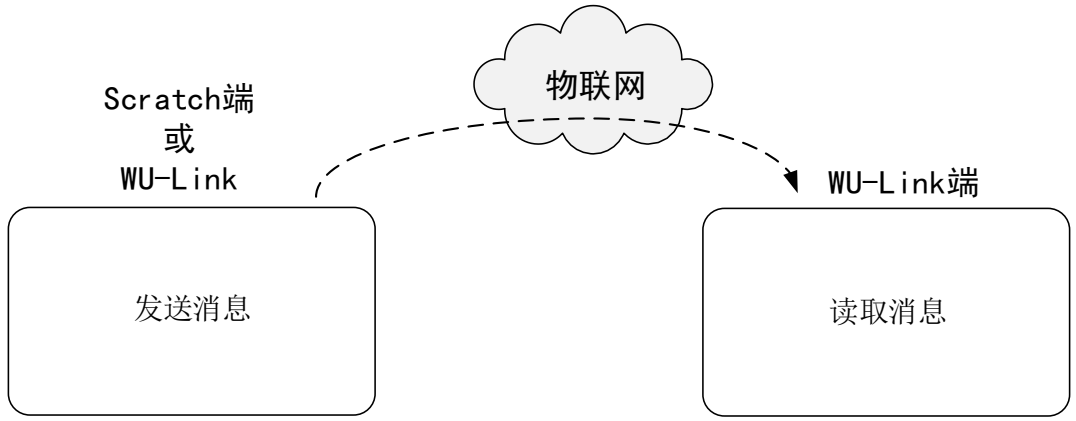
**MAC 地址**

MAC 地址用来定义网络设备的位置，又称硬件地址，一个网络设备只有一个 MAC 地址。

**物联网通讯方式**

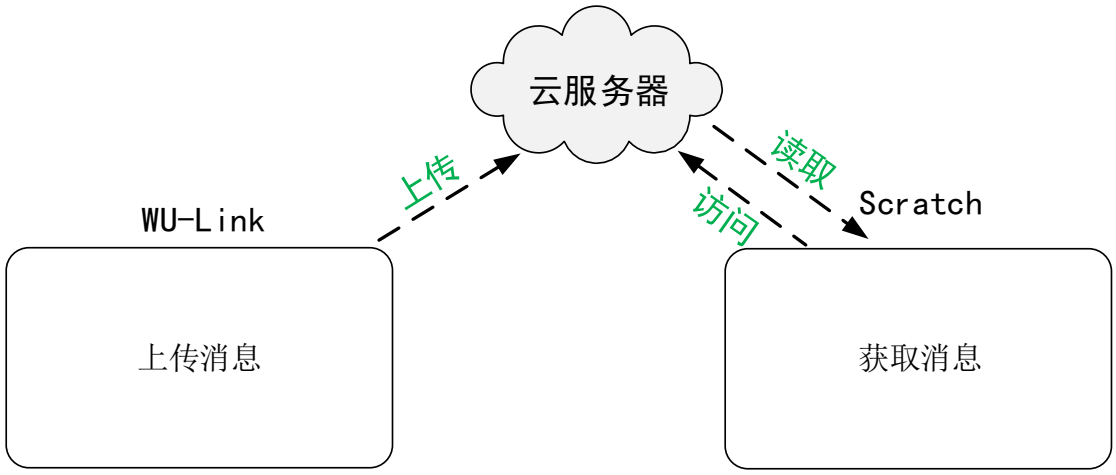
通过物联网向 WU-Link 发送数据，必须设置接收设备的 MAC 地址。消息发出后，通过物联网直接将消息送达到指定 MAC 地址的 WU-Link 设备中，WU-Link 使用“读取消息”指令即可把消息读取出来。

通过物联网向 WU-Link 发送数据就像是快递送货上门，其程序与通讯流程图示如下：



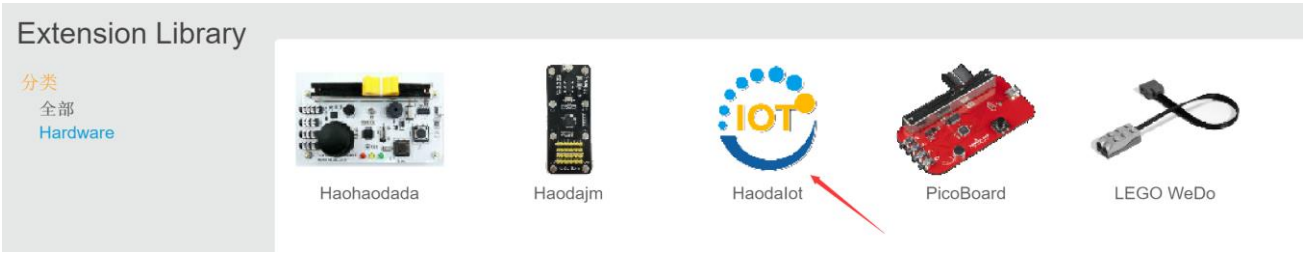

Scratch 端的程序因为是在 PC 端浏览器中运行，无法获取 PC 的 MAC 地址，所以 WU-Link 在发送消息时无法指定哪个 Scratch 程序接收，只能将消息上传到物联网中，由 Scratch 根据发送端 WU-Link 的 MAC 地址到物联网中获取。

通过物联网向 Scratch 发送数据就像是快递送达自提点之后由收件人来取，其程序与通讯流程图示如下：



# 第二十六节 Scratch 向 WU-Link 发消息

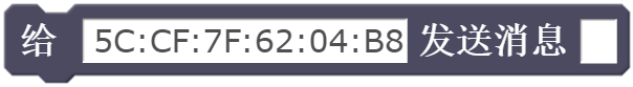
Scratch 中的物联网相关图形指令，需要在“更多模块”中添加扩展，并选择“Haodalot”模块。




**示例 26-1：物联网点读机（Scratch 向 WU-Link 发送字符串）**

在 Scratch 中点击角色，WU-Link 点阵屏上显示对应的英文单词

**认识新指令——向指定 MAC 地址设备发送消息（Scratch 端）**




该指令有两个参数框，第一个参数框是目标设备的 MAC 地址，可以手动输入，也可以用 ctrl+c（复制）和 ctrl+v（粘贴）的方式输入；第二个参数框为要发送的文本信息，不支持中文。

**认识新指令——物联网检测到消息指令（WU-Link 端）**


该指令可以检测有没有发送给本台 WU-Link 的消息。如果有消息，指令的返回值为“1”，也就是逻辑值为“真”；如果没有消息，指令的返回值为“0”，也就是逻辑值为“假”。

**认识新指令——物联网读取消息指令（WU-Link 端）**


该指令可以读取最新一条发送给本台 WU-Link 的字符串信息。在使用该指令之前，必须先判断是否检测到信息。

**元器件列表：**

1. WU-Link 主控板 ×1

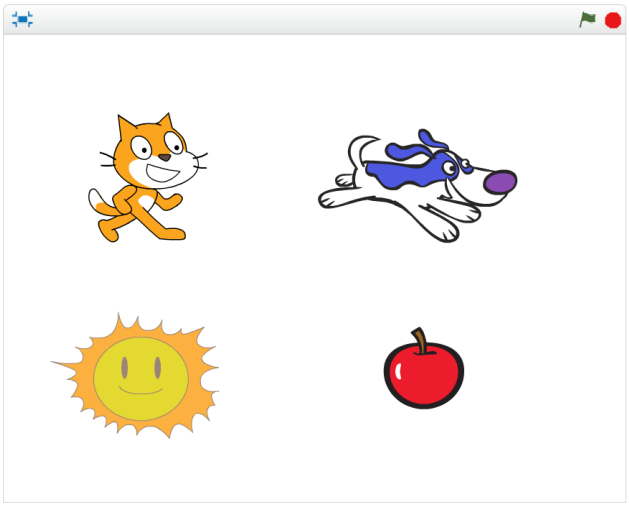
**电路连接：**

略

**程序编写：**

Scratch 端的程序：

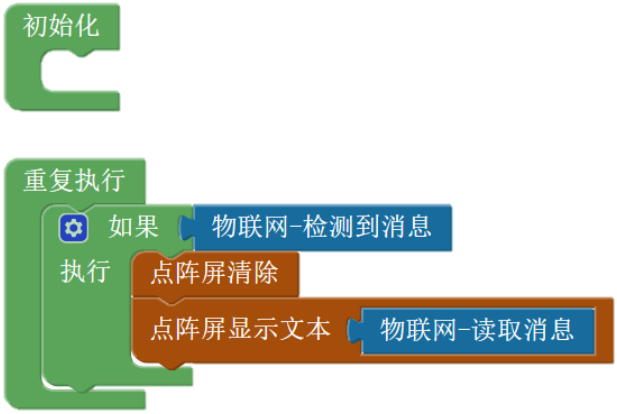
Scratch 舞台：



Scratch 角色及其脚本为：

| 角色 | 脚本 |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

WU-Link 端的程序为：




以上程序的运行结果是，每点击角色一次，WU-Link 的点阵面板显示对应的单词一次。

**示例 26-2：物联网亮度调节灯（Scratch 向 WU-Link 发送单个变量）**

在 Scratch 中调节一个个变量的滑杆值，远程控制 WU-Link 外接 RGB 模块的亮度。

因为网络通讯中只能传递字符串数据，所以这里约定：

**发送端发送“变量”必须为一个形如“XX=YY”的字符串**

其中 XX 是字符串中的“变量”名称，不能为中文，YY 是字符串中的数值，必须为整数。

**认识新指令——物联网读取消息中指定变量的值（WU-Link 端）**


该指令可以从消息中读取指定“变量”的值。第一个参数框中可以嵌入“物联网-读取消息”指令或一个字符串型变量；第二个参数框用于输入指定“变量”的名称，必须与 Scratch 端发送的“变量”名称完全一致！

例：

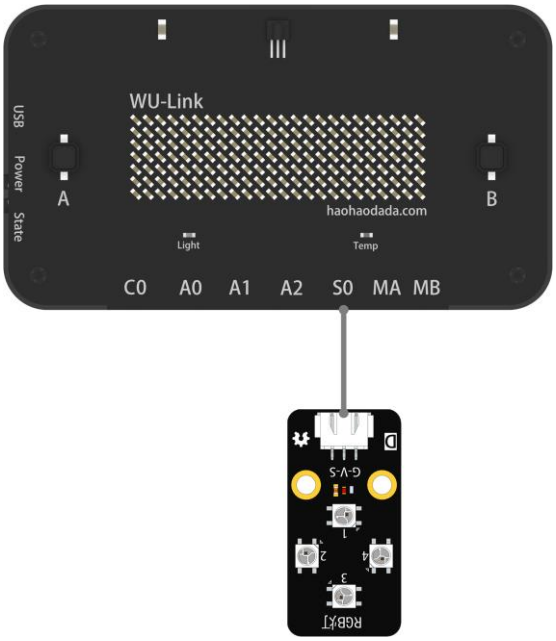

使用上图指令时，如果有接收到形如“XX=YY”的消息，则返回 YY（数值）；如果接收到的消息中不含“XX=”，则返回数值 0。

所以，为了避免混淆，则应保证发送的消息“XX=YY”中，YY≠0。

**元器件列表：**

1. WU-Link 主控板 ×1
2. RGB 模块 ×1

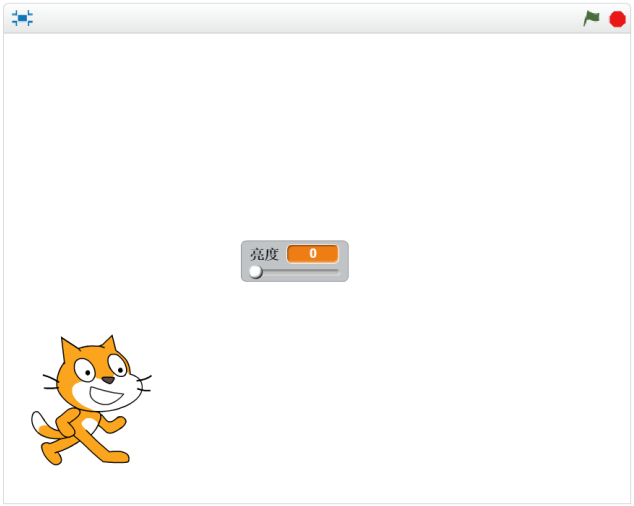
**电路连接：**




**程序编写：**

Scratch 端的程序：

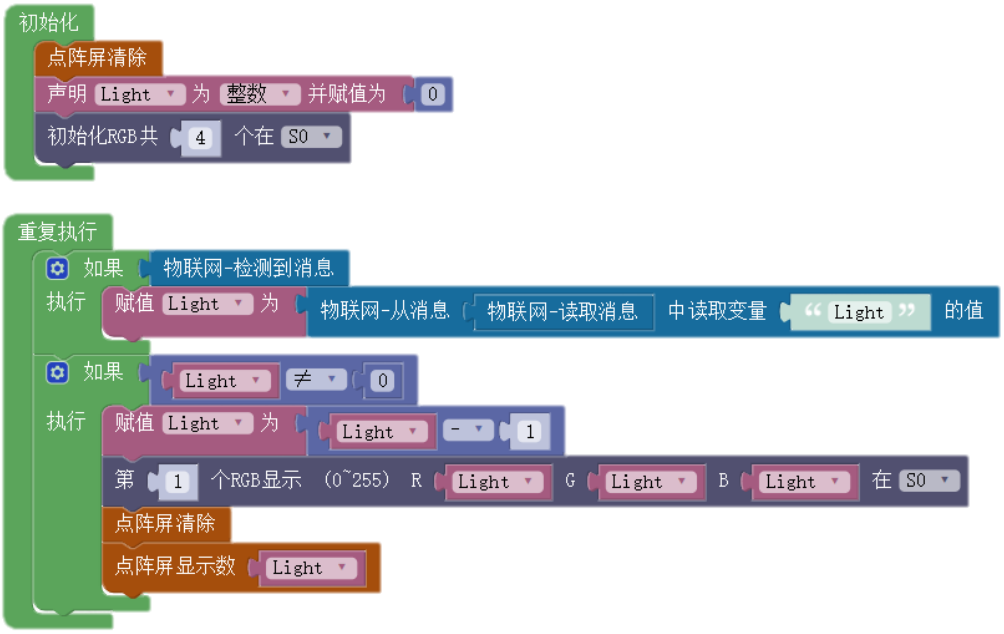
Scratch 舞台：




Scratch 角色及其脚本为：

| 角色 | 脚本 |
| --- | --- |
|  |  |

WU-Link 端的程序为：




**程序解析：**

在上述程序中，看到 Scratch 端程序中发送的“变量 Light”不是“亮度”而是“亮度+1”：


又看到 WU-Link 端程序在收到消息后，让变量 Light-1：


之所以做这样的处理，是因为 Scratch 端“亮度”变量的取值范围为 0-100，加 1 之后取值范围变为 1-101，发送的“变量”数值不会为 0，可以避免与其它消息产生的混淆。WU-Link 端程序再将“Light”变量的值减 1，则将取值范围变回 0-100，“Light”变量的值可以取到 0，进而实现 RGB 灯的熄灭。

**示例 26-3：物联网调色灯（Scratch 向 WU-Link 发送多个变量）**

在 Scratch 中调节三个变量的滑杆值，远程控制 WU-Link 外接 RGB 模块的颜色。

**元器件列表：**

1. WU-Link 主控板 ×1
2. RGB 模块 ×1

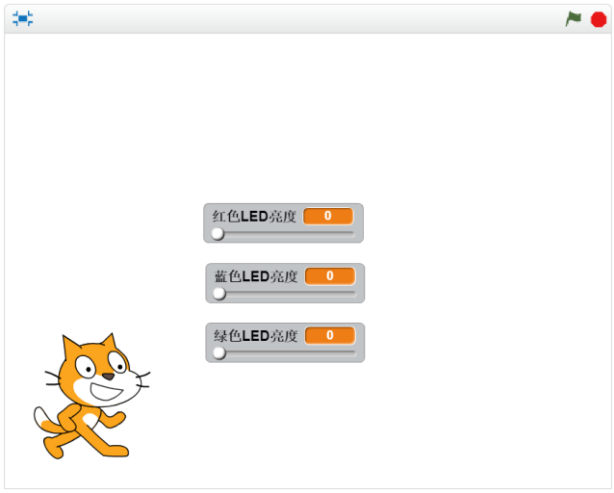
**电路连接：**

与示例 26-2 相同

**程序编写：**

Scratch 端的程序：

Scratch 舞台：




Scratch 角色及其脚本为：

| 角色 | 脚本 |
| --- | --- |
|  |  |



WU-Link 端的程序为：


其中变量 R、G、B 为数值型变量，变量 MSG 为字符串型变量。


**程序解析:**

当物联网通讯中传递多个“变量”，与传递单个“变量”，在程序上的处理的不同点为：必须创建一个字符串型变量，把读取到的消息存储下来。

因为每使用一次“物联网-读取消息”指令，物联网中等待接收的消息就会被更新一次，如不用字符型变量存储消息的程序为：

上述程序运行一个周期，使用“物联网-读取消息”指令 6 次，即物联网中等待接收的消息更新了 6 次，如果按下表顺序接收消息，则变量 R、G、B 分别接收到的值为：

| 消息顺序 | 消息内容 | 运行的程序 | 变量 R、G、B 的值 |
|---|---|---|---|
| 第 1 条 | R=42 | *从消息中获取“R”的值 | R=50 G=50 B=50 |
| 第 2 条 | G=83 | 从消息中获取“R”的值 | R=0 G=50 B=50 |
| 第 3 条 | B=67 | *从消息中获取“G”的值 | R=0 G=50 B=50 |
| 第 4 条 | R=42 | 从消息中获取“G”的值 | R=0 G=0 B=50 |
| 第 5 条 | G=83 | *从消息中获取“B”的值 | R=0 G=0 B=50 |
| 第 6 条 | B=67 | 从消息中获取“B”的值 | R=0 G=0 B=67 |

备注：（1）假设接收消息之前变量 R、G、B 的值均为 50；（2）“运行的程序”列中标“*”的为判断是否为 0 的程序，未标“*”的为赋值程序。

从上表可以看出，程序运行一个周期，除变量 B 侥幸获取正确外，变量 R、G 的值都获取错误。即如果消息的内容和要获取的“变量”不一致时，有效信息就会丢失，并造成错误的通讯。

如果换做用字符串型变量存储消息的程序，则程序运行一个周期的结果是：

| 消息内容 | 字符串串变量内容 | 运行的程序 | 变量 R、G、B 的值 |
|---|---|---|---|
| R=42 | R=42 | *从消息中获取“R”的值 | R=42 G=50 B=50 |
| | R=42 | 从消息中获取“R”的值 | R=42 G=50 B=50 |
| | R=42 | *从消息中获取“G”的值 | R=42 G=50 B=50 |
| | R=42 | 从消息中获取“G”的值 | R=42 G=50 B=50 |
| | R=42 | *从消息中获取“B”的值 | R=42 G=50 B=50 |
| | R=42 | 从消息中获取“B”的值 | R=42 G=50 B=50 |

从上表看出，虽然程序运行一个周期，只更新了一个变量 R 的值，但是变量 R 的值是稳定的正确获取了，只需再执行两个周期，变量 G、B 的值也能正常获取。

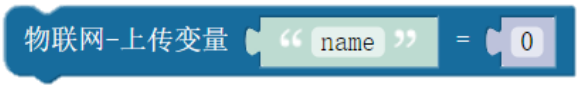
综上，当物联网通讯中传递多个“变量”时，必须先用字符串型变量将消息存储起来。

# 第二十七节 WU-Link 向 Scratch 发消息

**示例 27-1：物联网环境监测（WU-Link 向 Scratch 发送变量）**

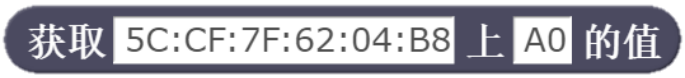
WU-Link 上传板载亮度传感器和温度传感器的值，Scratch 从物联网中获取亮度值和温度值并显示出来。

**认识新指令——物联网上传变量指令（WU-Link 端）**




该指令有两个参数框，第一个参数框是“变量”的名称，不能为中文；第二个参数框为“变量”的数值，必须为整数。

**认识新指令——获取指定设备的指定变量值（Scratch 端）**




该指令能获取指定 MAC 地址设备上传的指定变量的值，第一个参数框输入指定设备的 MAC 地址；第二个参数框输入指定“变量”的名称，必须与 WU-Link 端上传的“变量”名称完全一致！

**元器件列表：**

1. WU-Link 主控板 ×1

**电路连接：**

略

**程序编写：**

WU-Link 端的程序为：

Scratch 端的程序：

Scratch 舞台：


Scratch 角色及其脚本为：

| 角色 | 脚本 |
| --- | --- |
|  | 当 被点击<br>重复执行<br>将 亮度 设定为 获取 BC:DD:C2:E7:0E:99 上 Light 的值<br>说 在 教学楼的亮度是： 后面添加 亮度<br>等待 1 秒 |
|  | 当 被点击<br>重复执行<br>将 温度 设定为 获取 BC:DD:C2:E7:0E:99 上 Temp 的值<br>说 在 在 教学楼的温度是： 后面添加 温度 后面添加 ℃<br>等待 1 秒 |

**示例 27-2：物联网传感器曲线绘制（WU-Link 向 Scratch 发送变量）**

使用 DS18B20 温度传感器测量热水温度，并在 Scratch 中绘制热水在常温环境下的冷却温度曲线，时间总长为 30 分钟，时间间隔为 5 秒一次。

**元器件列表：**

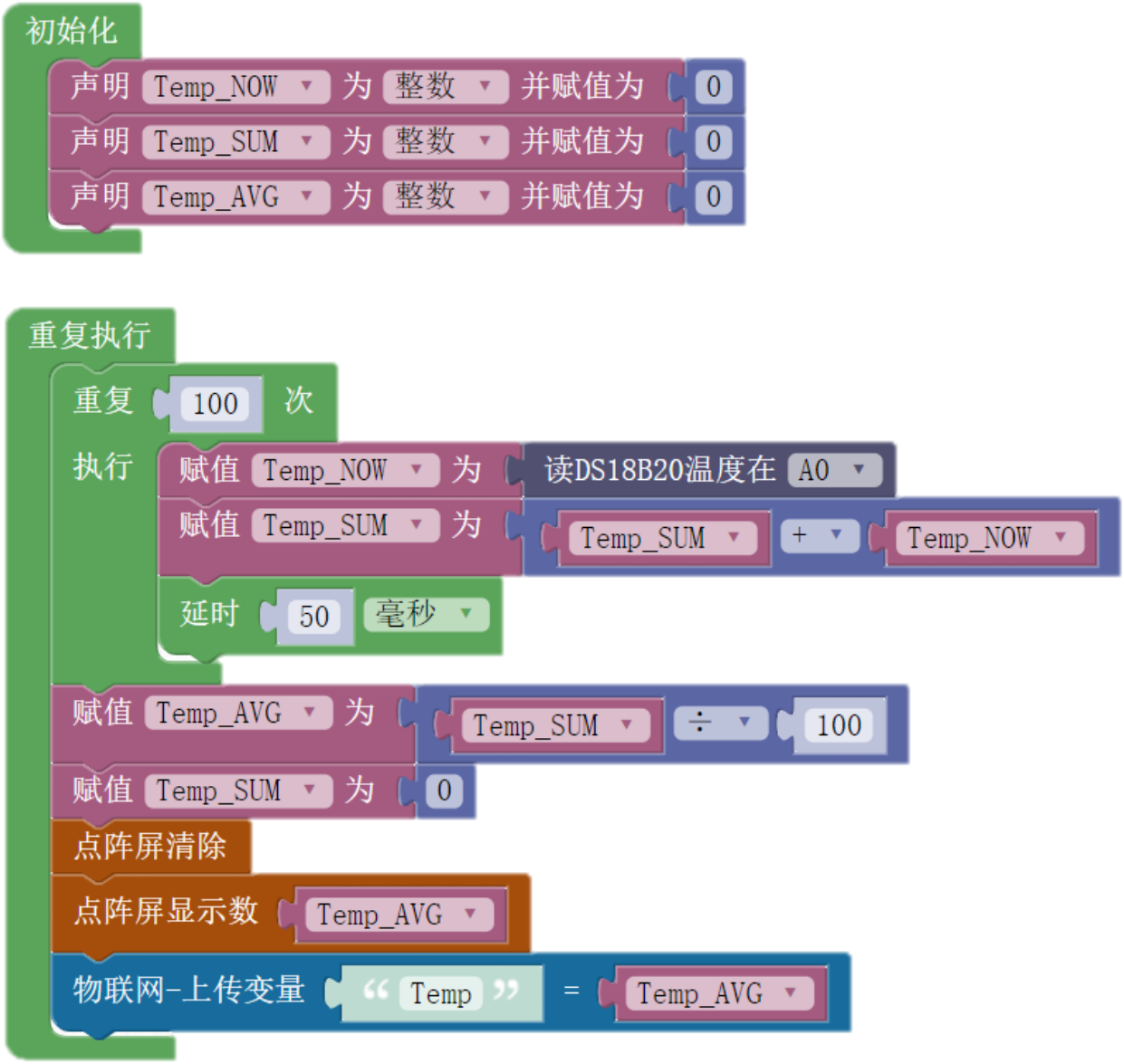
1. WU-Link 主控板 ×1

**电路连接：**

略

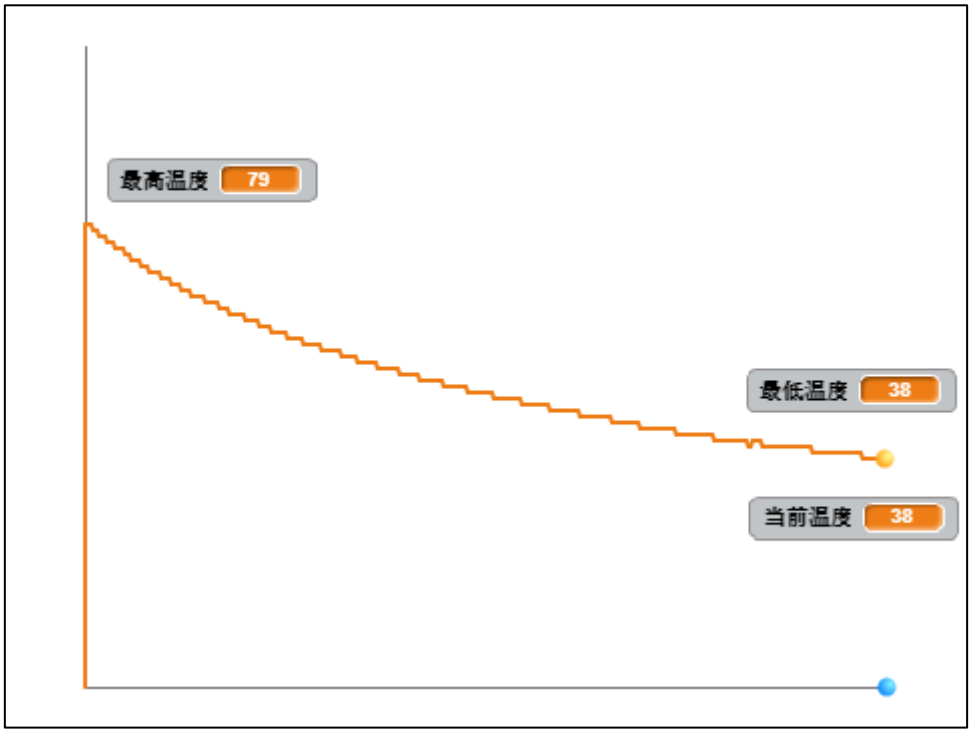
**程序编写：**

WU-Link 端的程序为：



Scratch 端的程序：

Scratch 舞台：




Scratch 角色及其脚本为：

| 角色 | 脚本 |
| --- | --- |
|  绘制坐标系 |   |

绘制曲线

```
当 被点击
将角色的大小设定为 20
抬笔
移到 x: -200 y: -160
将画笔的颜色设定为
将画笔的大小设定为 2
落笔
将 x 设定为 -200
重复执行 400 次
    将 当前温度 设定为 获取 BC:DD:C2:E6:FF:85 上 Temp 的值
    将 y 设定为 当前温度 * 3 - 160
    移到 x: x y: y
    将变量 x 的值增加 1
    等待 5 秒
```

```
当 被点击
将 最高温度 设定为 0
将 最低温度 设定为 100
重复执行
    如果 最高温度 < 当前温度 那么
        将 最高温度 设定为 当前温度
    如果 最低温度 > 当前温度 那么
        将 最低温度 设定为 当前温度
```

# 第二十八节 WU-Link 向 WU-Link 发消息

**示例 28-1：物联网呼叫器（WU-Link 单对单通讯）**

发送端 WU-Link 按下 A 键时，点阵屏显示“Call”并发出约定消息；

接收端 WU-Link 接收到约定消息，点阵屏显示“Hello!”，蜂鸣器响、振动马达启动。

**认识新指令——物联网发送字符串（WU-Link 端）**


该指令有两个参数框，第一个参数框是要发送的字符串，不能为中文；第二个参数框为目标 WU-Link 设备的 MAC 地址。

**认识新指令——物联网发送数值（WU-Link 端）**


该指令有两个参数框，第一个参数框是要发送的数值，必须为数字；第二个参数框为目标 WU-Link 设备的 MAC 地址。

**认识新指令——物联网发送变量（WU-Link 端）**


该指令有三个参数框，第一个参数框是要发送的“变量”名称，不能为中文；第二个参数框是要发送的数值，必须为数字；第三个参数框为目标 WU-Link 设备的 MAC 地址。

**元器件列表：**

1. WU-Link 主控板 ×2

**电路连接：**

略

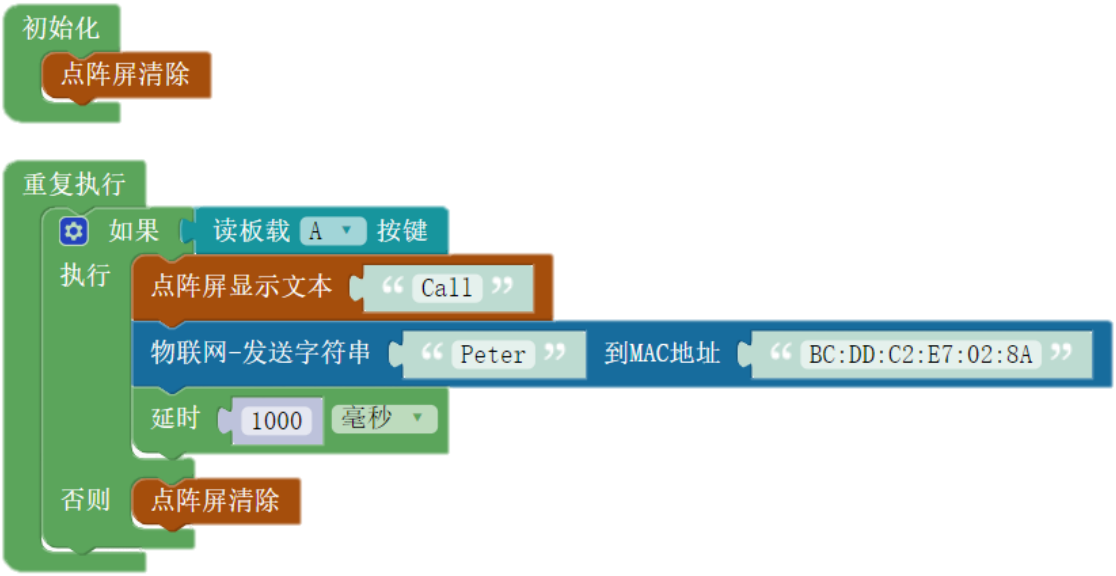
**程序编写：**

WU-Link 与 WU-Link 之间通过物联网通讯，有发送字符串、数值、“变量”三种方式，本示例可以用任意一种方式实现。

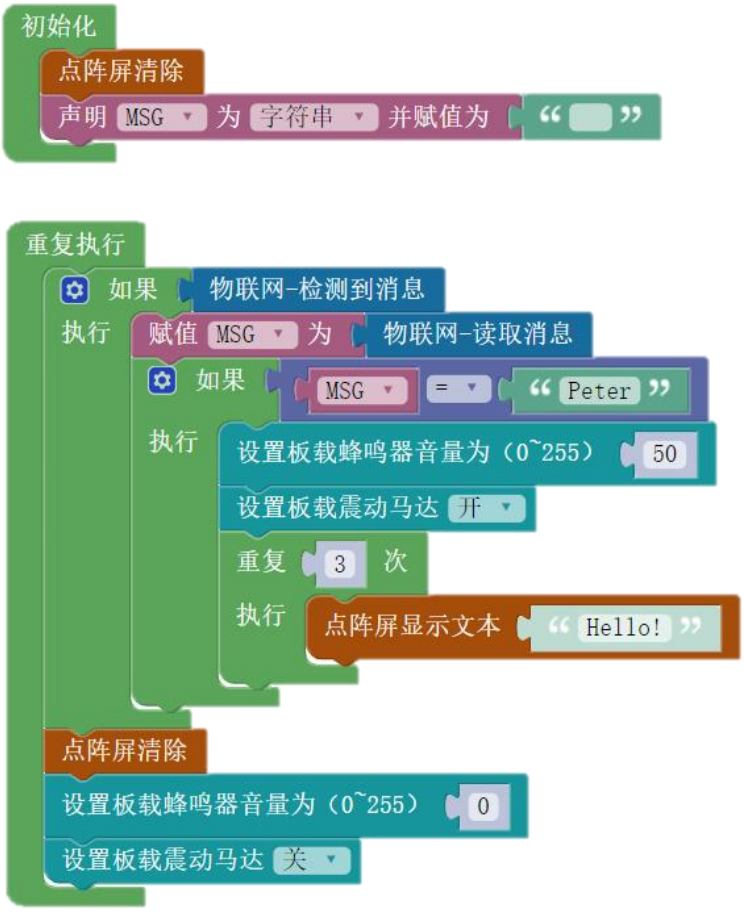
以下分别用三种方式实现本示例的应用。

**WU-Link 之间通过字符串传递消息**

发送端 WU-Link 的程序为：

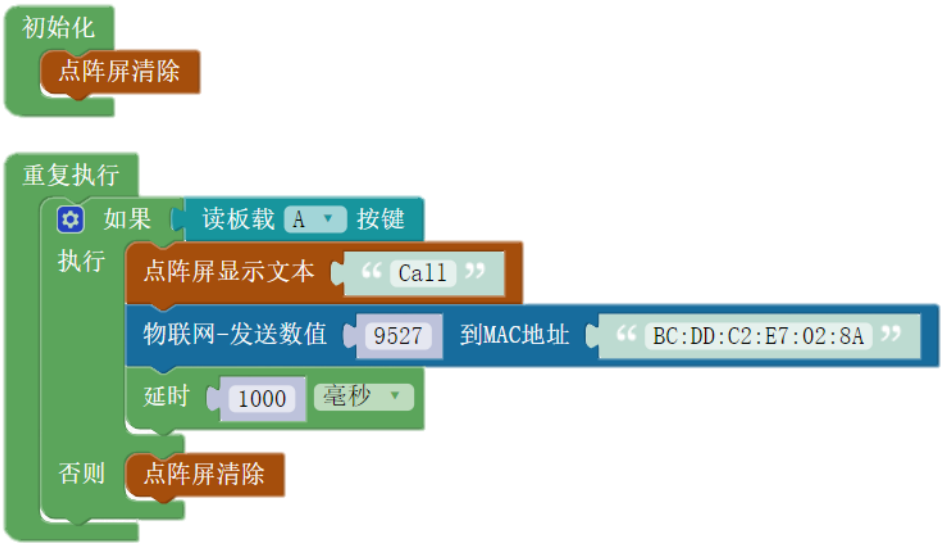



接收端 WU-Link 的程序为：

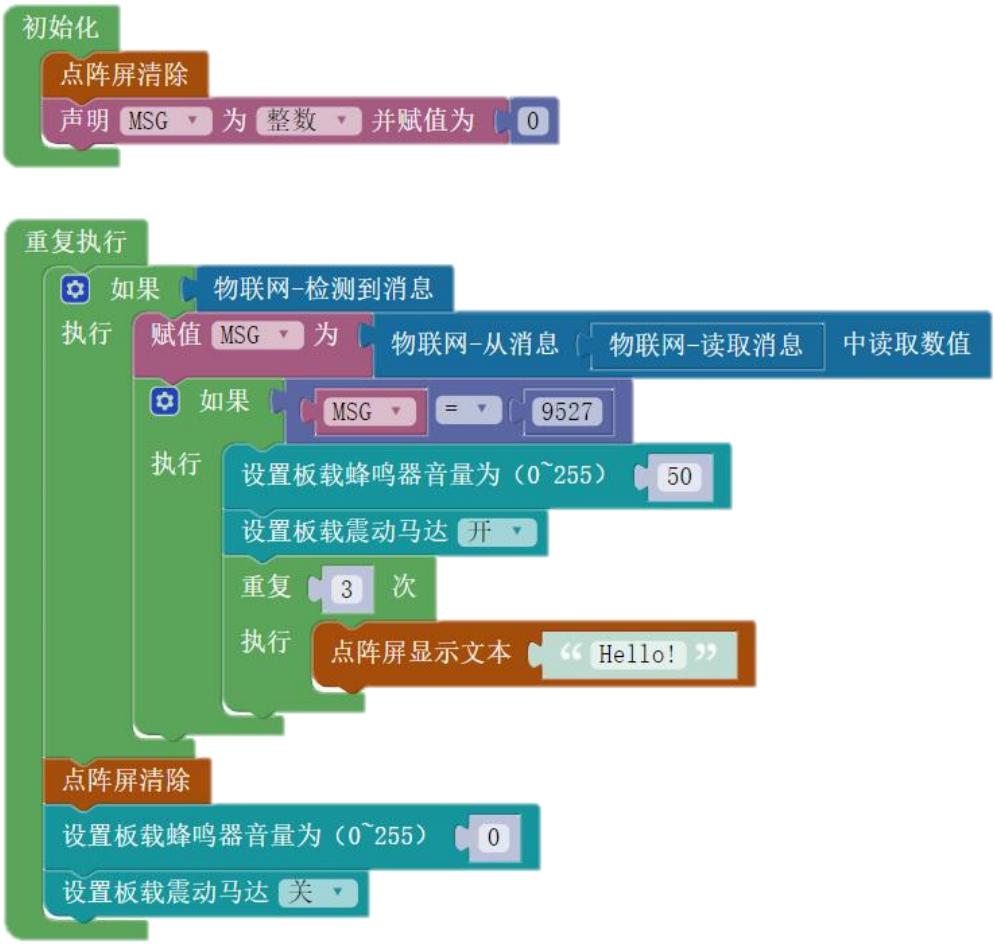


**WU-Link 之间通过数值传递消息**

发送端 WU-Link 的程序为：

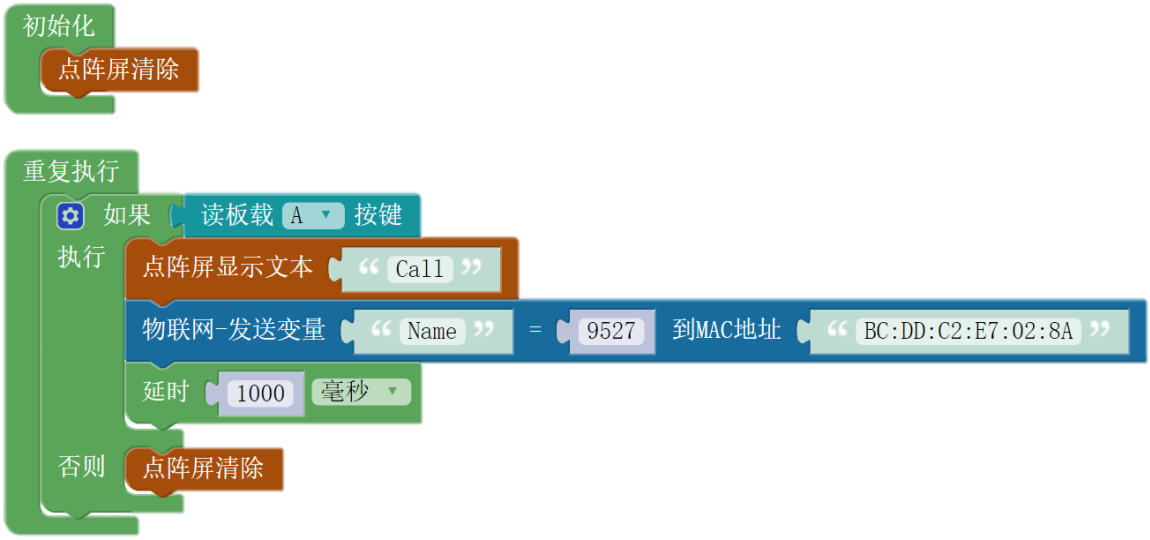



接收端 WU-Link 的程序为：

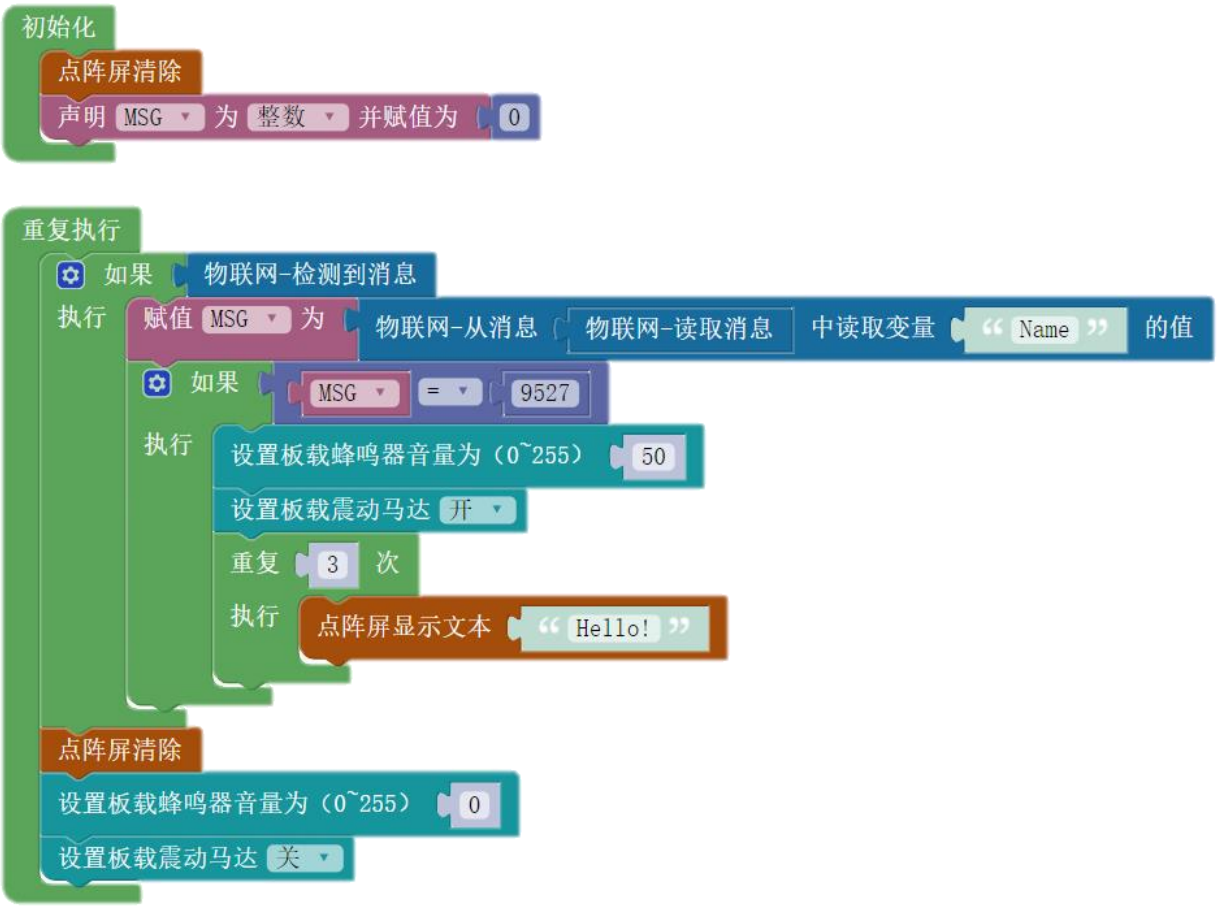


**WU-Link 之间通过“变量”传递消息**

发送端 WU-Link 的程序为：




接收端 WU-Link 的程序为：



**示例 28-2：物联网抽签器（WU-Link 单对多通讯）**

有一位主持人和编号分别为 1、2、3 的三位观众，他们手中各有一台 WU-Link。

主持人快速摇动自己的 WU-Link，停止之后，显示抽签结果，并将结果发送给三位观众。

观众的 WU-Link 上显示被抽中的编号。

**元器件列表：**

1. WU-Link 主控板 ×4

**电路连接：**

略

**程序编写：**

本例采用数值方式进行通讯。

发送端 WU-Link（主持人）的程序为：


以上程序将摇晃取随机数的动作与发送消息分开，即主持人摇晃 WU-Link 时，并不发送消息。

其中，变量 Flag 的可以看做一个虚拟“标记”。

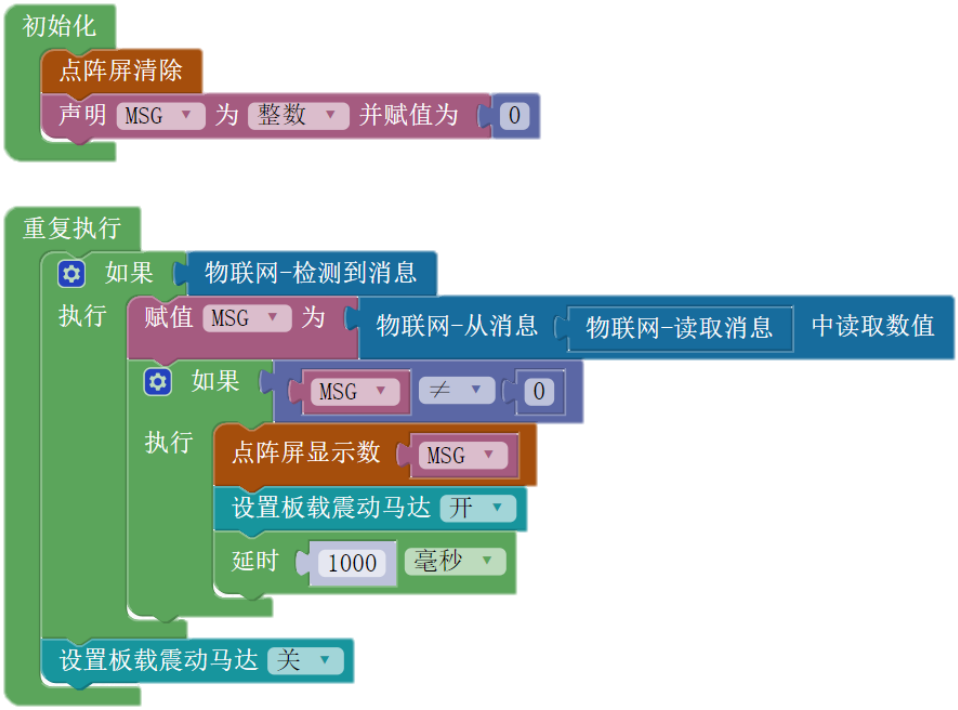
（1）程序启动时，变量 Flag 被初始化为 0；

（2）当主持人不停的摇晃 WU-Link 时，变量 Flag 被设置为 1，用于标记摇晃结束。

（3）如果变量 Flag 为 1，则代表主持人的 WU-Link 被摇晃了一次，摇晃结束后，即可根据 Flag=1，发送消息；

（4）在发送完消息之后，将变量 Flag 设置为 0，即可结束消息的发送，等待下一次的摇晃。

接收端 WU-Link（观众）的程序为：




观众的 WU-Link 接收到消息，振动马达起动，并把数值显示出来。

**示例 28-3：物联网抢答器（WU-Link 多对单通讯）**

有一位主持人和编号分别为 1、2、3 的三位选手，他们手中各有一台 WU-Link。

（1） 主持人按下自己 WU-Link 上的 A 键，并口头发出抢答开始指令；

（2） 之后三位选手可以按下自己的 A 键向主持人的 WU-Link 发送带有自己编号的消息；

（3） 主持人的 WU-Link 将第一个接收到的消息显示出来，之后不再接收任何消息；

（4） 主持人可以再次按下 A 键，重置，等待下一回合的抢答。

**元器件列表：**

1. WU-Link 主控板 ×4

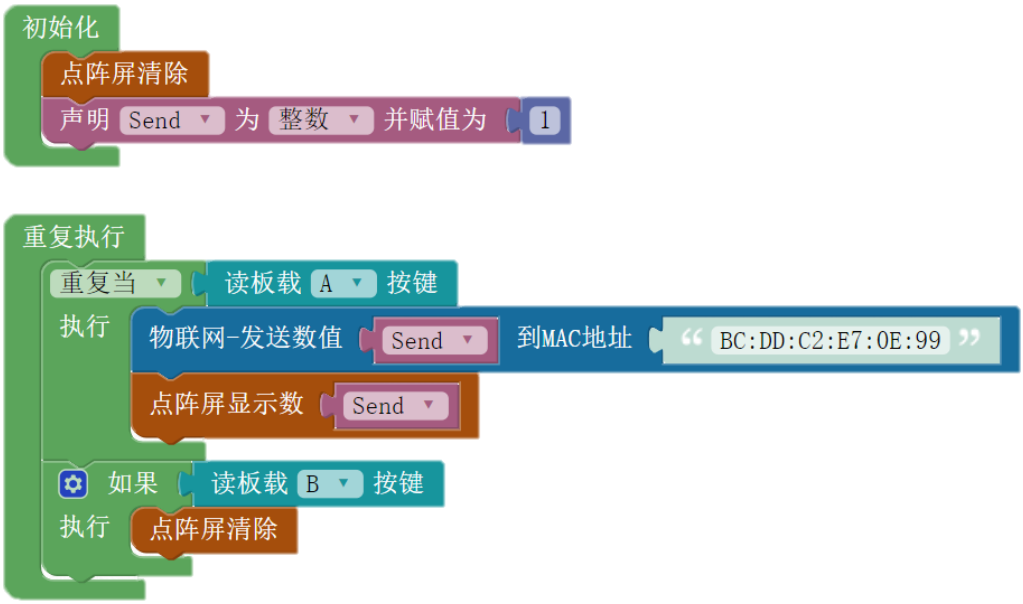
**电路连接：**

略

**程序编写：**

本例采用数值方式进行通讯。

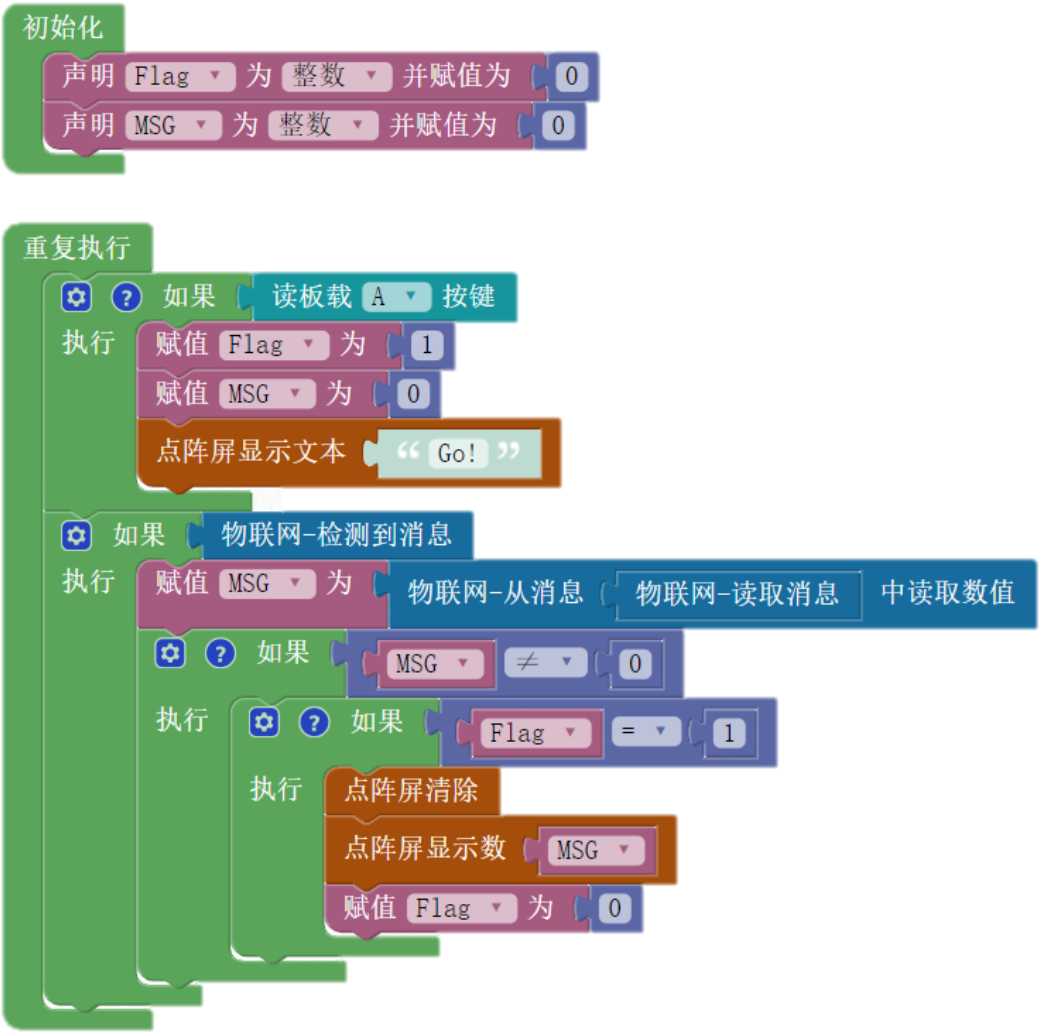
发送端 WU-Link（选手）的程序为：




其中，变量 Send 的值为选手的编号，1 号选手的 Send 值为 1，2 号选手的 Send 值为 2，2 号选手的 Send 值为 3，所以三个发送端程序略有不同，上图为 1 号选手的程序。

选手可以按下 B 键将点阵屏显示清除，这样下次按下 A 键时的再次显示可以作为 A 键是否按下的视觉反馈。

接收端 WU-Link（主持人）的程序为：




上图程序的变量 Flag 同样是一个虚拟“标记”。

（1）程序启动时，变量 Flag 被初始化为 0；

（2）当主持人按下 A 键后，变量 Flag 被设置为 1，用于标记可接收信息状态。

（3）可接收信息状态下，在接收到第一条有效信息，点阵屏将信息显示出来之后，将变量 Flag 设置为 0，标记为不可接收信息状态，点阵屏的显示信息不再更新；

（4）再次按下 A 键，又可将变量 Flag 设置为 1，进行下一回合的抢答。

# 附录一：WU-Link 工作状态指示


**1. Web 配置模式：**电源指示灯常亮，状态指示灯红色常亮，蜂鸣器响一声；

**2. 微信配置模式（联网失败状态）：**电源指示灯常亮，状态指示灯红色慢闪，蜂鸣器“滴-滴-…”每隔 0.5 秒响一次；

**3. 联网成功状态：**电源指示灯常亮，状态指示灯蓝色常亮，蜂鸣器不响；

**4. 低电量状态：**未连接 USB 线时，电源指示灯闪烁；

**5. 充电未充满状态：**连接 USB 线时，电源指示灯闪烁；

**6. 充电已充满状态：**连接 USB 线时，电源指示灯常亮；

**7. 关机状态：**电源指示灯不亮，状态指示灯不亮。

# 附录二：设备操作方式

**1.开机：**关机状态下，单击电源键；

**2.关机：**开机状态下，快速双击电源键；

**3.重启程序：**开机状态下，单击电源键；

**4.清空程序：**关机状态下，按住 A 键，再单击电源键；


**5.进入 web 配置模式：**关机状态下，按住 B 键，再单击电源键。

# 附录三：WU-Link 的网络连接设置（web 配置方式）

## 第一步，让 WU-Link 进入配置模式


进入无线路由模式的组合键

按上图方式操作，电源灯亮起，状态指示灯为红色常亮，蜂鸣器响一声，进入配置模式。

## 第二步，让手机或 PC 连接 WU-Link 路由器

手机和 PC 可以像连接常规无线路由器一样进行连接，具体操作不用多说。

在无线网络连接中找到一个以“haohaodada”8 个字母开头后接 12 位 MAC 地址的路由器，对它进行连接，连接密码为：haohaodada。


1. 搜索到 WU-Link 路由器

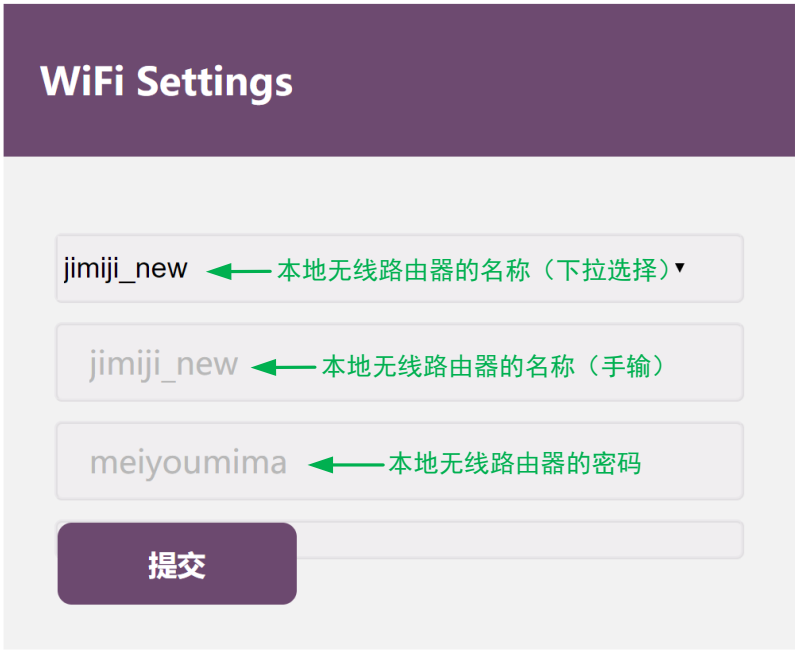

2. 连接 WU-Link 成功

右图为 iPhone 的无线网路连接界面，其中“haohaodada5c:cf:7f:65:e1:d9”即为 WU-Link。点击进行连接，输入密码“haohaodada”，即可连接成功。

**第三步，设置 WU-Link 的网络连接**

该步骤是告诉 WU-Link 如何连接本地的无线路由器。

打开浏览器，输入网址：“192.168.4.1”。选择或输入本地的无线路由器名称。



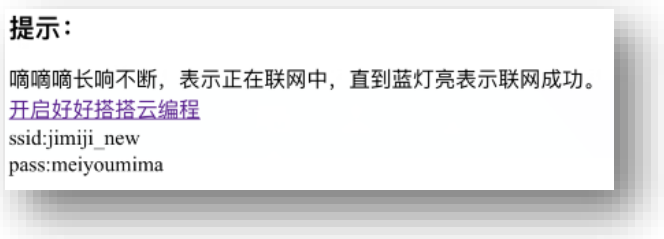

浏览器设置界面

其中，本地无线路由器的名称可以通过两种方式输入：

第一种，如果 WU-Link 能搜索到本地路由器，将出现在第一行下拉列表中，可以直接选取，第二行的名称也会变为选取的路由器名称。

第二种，如果 WU-Link 无法搜索到本地路由器，即未出现在第一行下拉列表中，可以在第二行直接输入路由器的名称，进行连接。该方法适合用于周围只有一台智能手机可用的情况下，进行联网操作。

点击“提交”按钮之后，浏览器出现如下界面。




等待一段时间，状态指示灯变为蓝色常亮之后，代表 WU-Link 已成功连入互联网。之后的操作与 web 配置方式相同，即浏览器登录“http://haohaodada.com/wulink”，登录好好搭搭账号，绑定设备 MAC 地址。

# 附录四：WU-Link 的网络连接设置（微信公众号配置方式）

**第一步，让 WU-Link 进入微信配置模式**

单击电源键开机，WU-Link 会试图去连接本地无线路由器，当连接失败后，直接进入等待微信配置模式，此时，状态指示灯为红色慢闪状态，蜂鸣器“滴-滴-…”每隔 0.5 秒响一次。

**第二步， “好好搭搭”微信公众号配置无线**


直接扫描设备背后标签上的二维码，关注好好搭搭微信公众号，也可以在微信中搜索“好好搭搭”公众号。


关注“好好搭搭”微信公众号　　点击“WU-Link”-“WIFI配置”　　点击“开始配置”　　点击“连接”

点击“WU-Link”-“WIFI 配置”，之后输入本地 WIFI 密码，点击“连接”按钮。

等待一段时间，状态指示灯变为蓝色常亮之后，代表 WU-Link 已成功连入互联网！