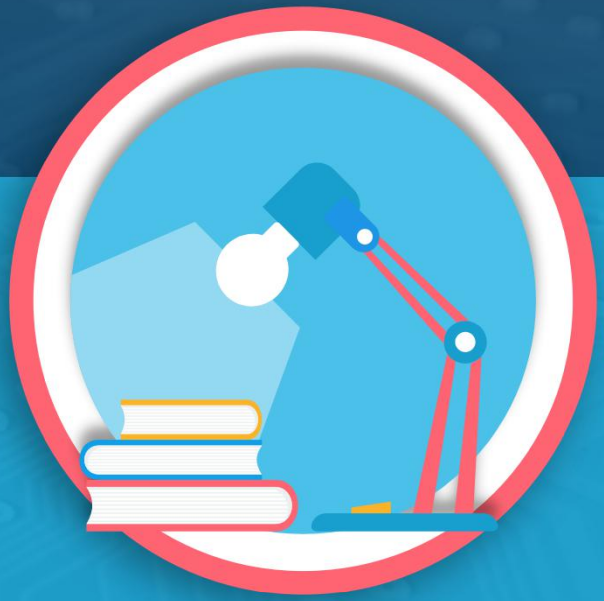
第一单元 点亮创客之路

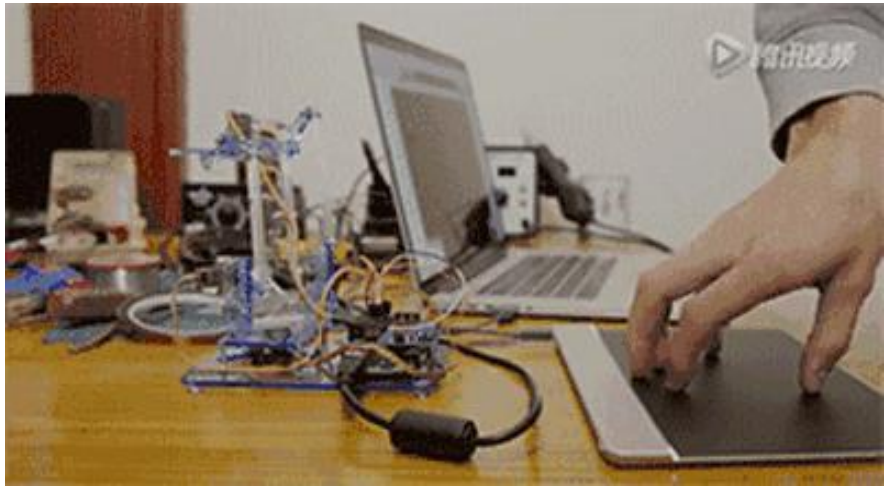
准备好了吗？

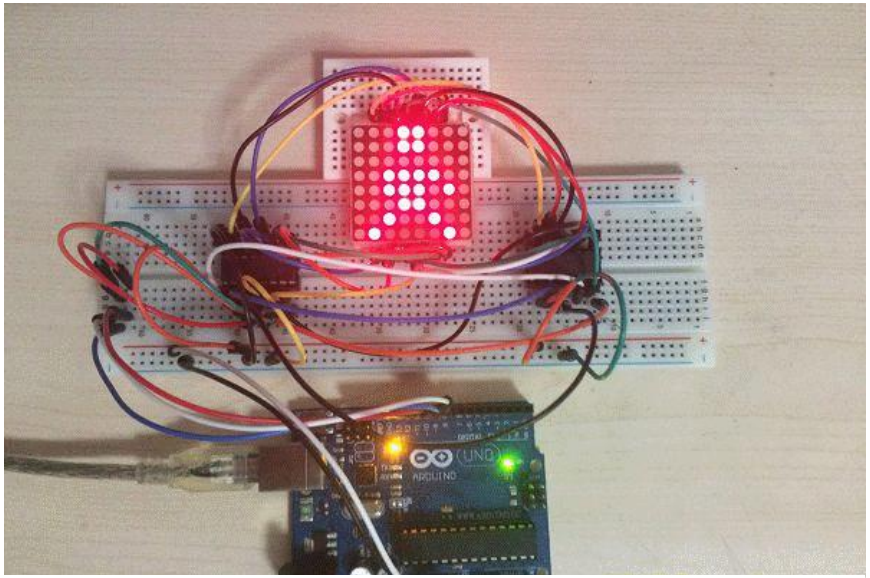
起始课



0 准备好了吗？

情境引入





$$8+4\div2=\ ?$$

0

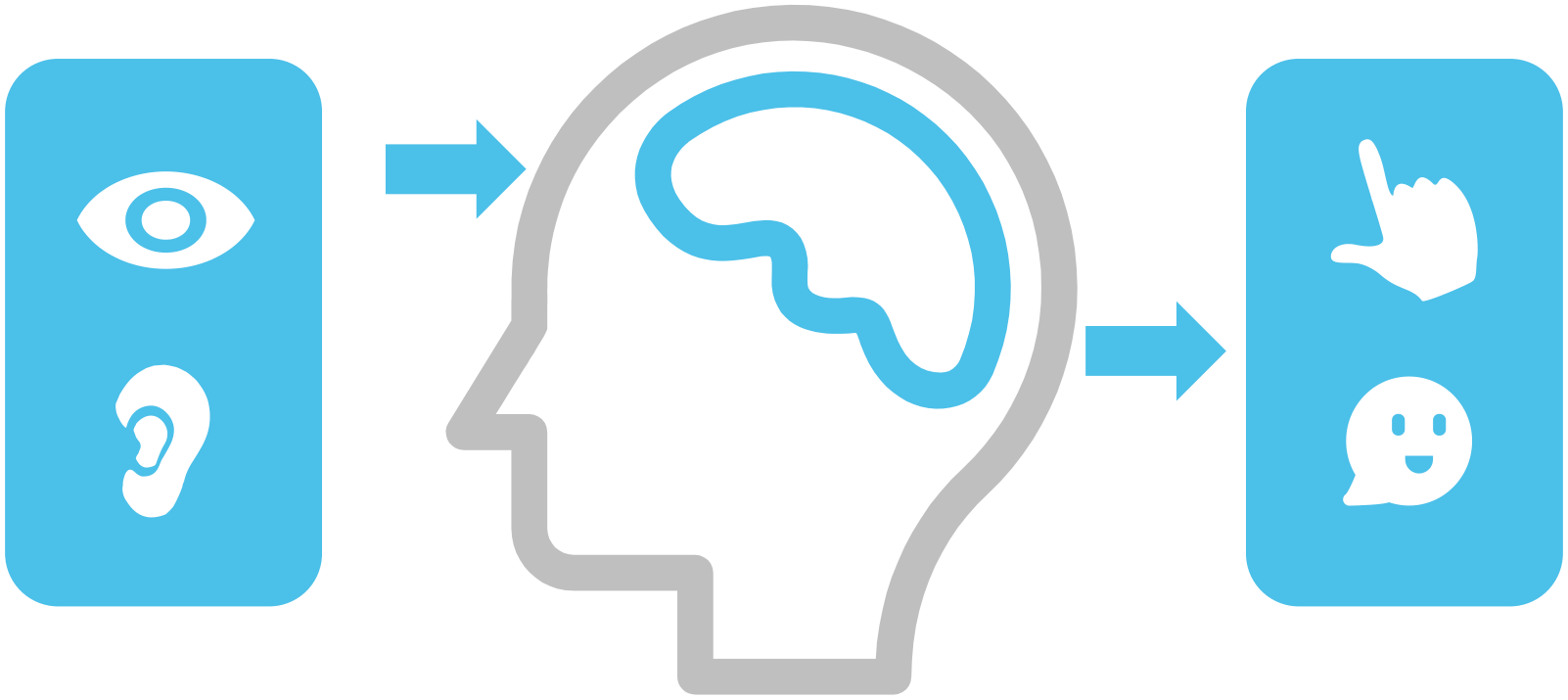
准备好了吗？

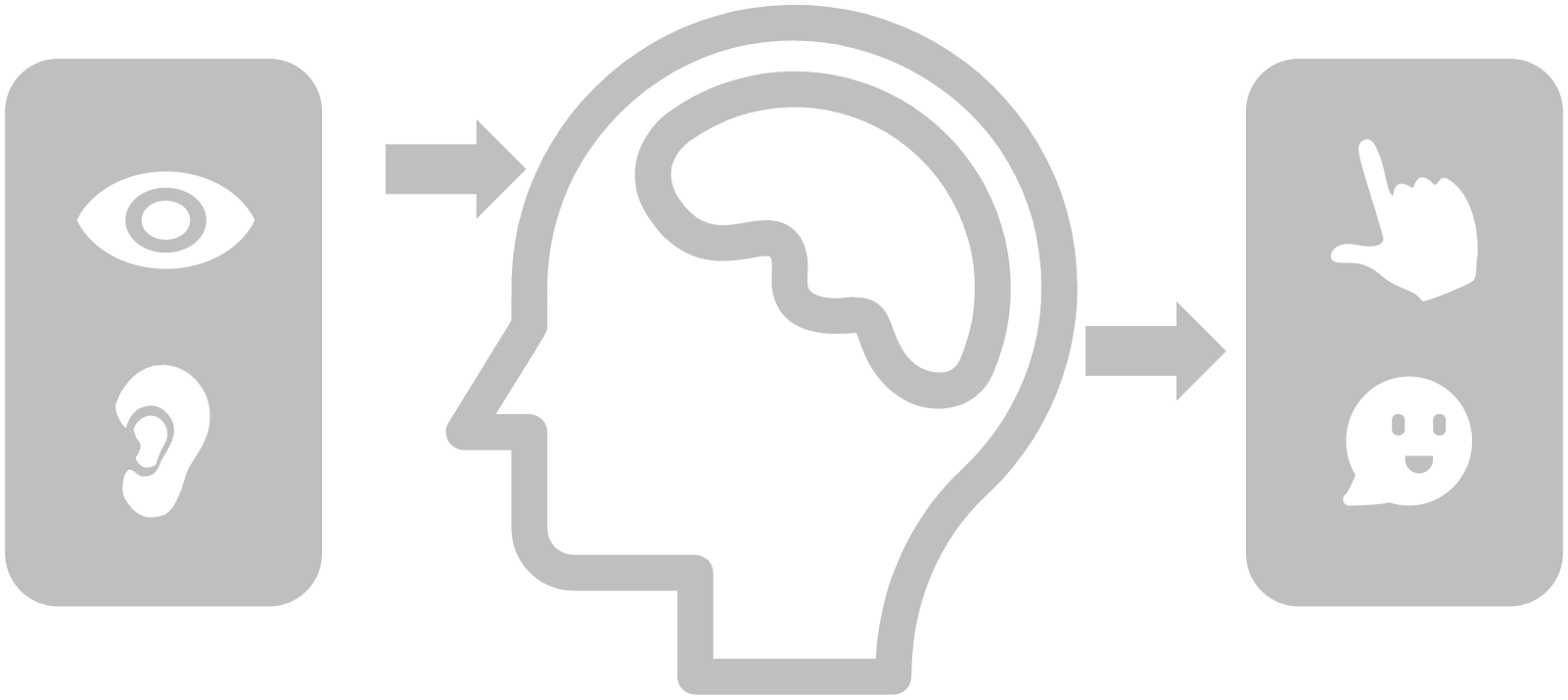
情境引入

$$8+4\div2=\ ?$$

0 准备好了吗？

认识Arduino

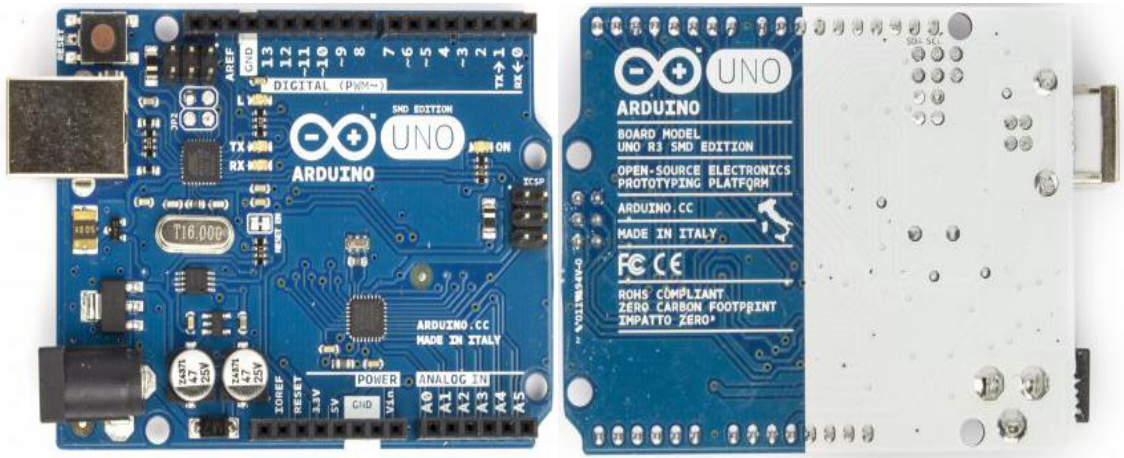




传感器　　控制板　　执行器

Massimo Banzi

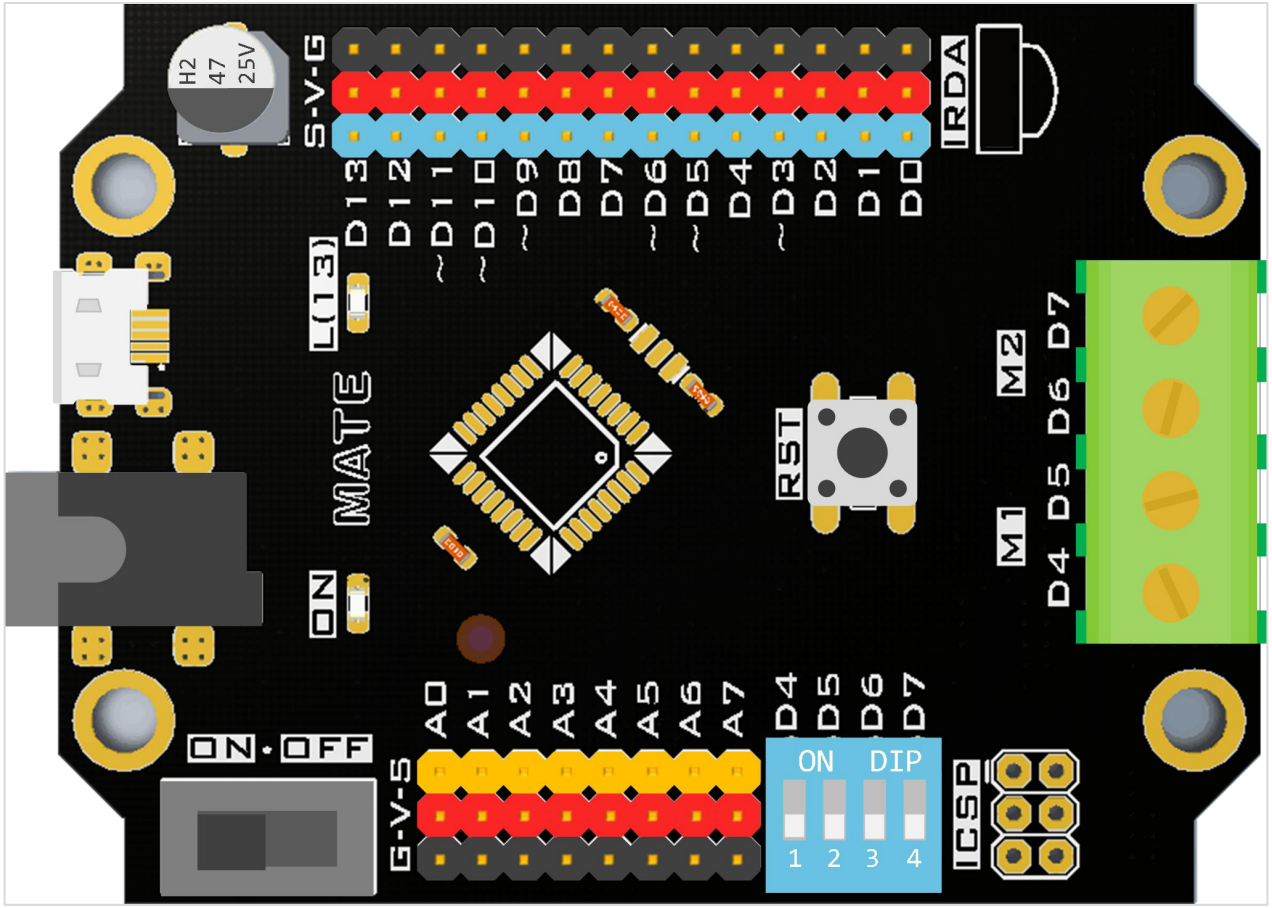
Arduino



Arduino Uno R3

准备好了吗？

认识Mate主控板



Mate 主控板

防止受伤

Mate主控板的管脚较为锐利，使用需小心

0 准备好了吗？

注意事项

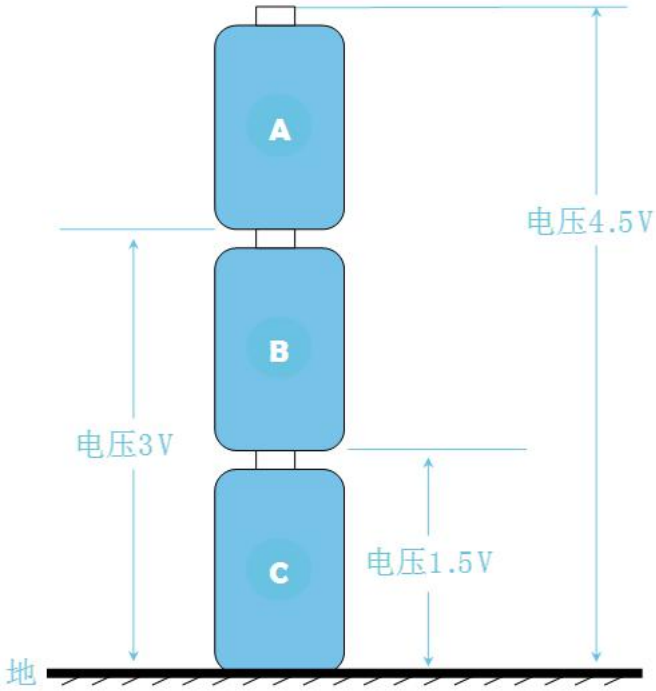
桌子上一定
不要放饮料

拿主控板时
尽量拿两侧

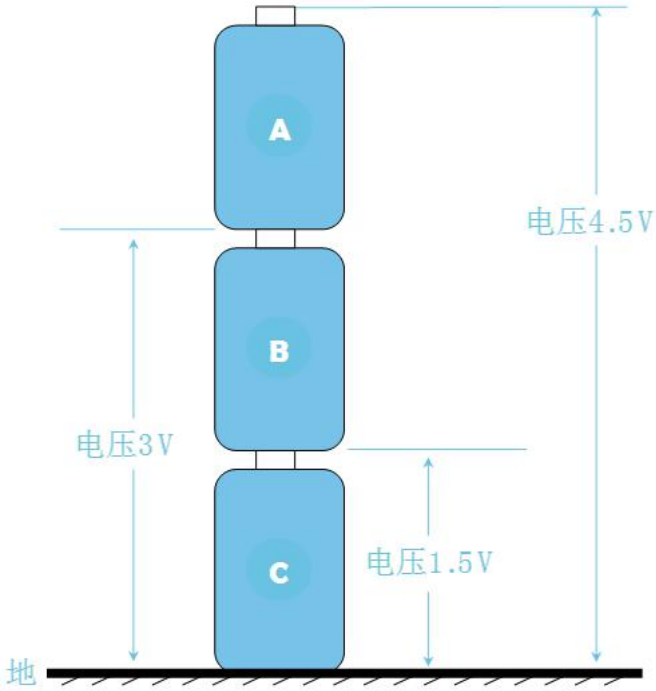
Mate主控板下方
最好垫一层绝缘材料

米思齐(Mixly)

- 自由电荷在导体中的定向移动形成了**电流(Current)**，电荷流动的前提是导体两端必须有电位差，电位差(也叫电势差)通常称作**电压(Voltage)**。



- 电压的大小可以类比水位相对于水平面的高低：处于高电平(Arduino主控板的最高电压为5V)的称为**正极**；处于低电平(一般规定0~0.25V)的称为**负极或接地**(Ground，简称GND)，这也是Arduino电压的基准参考点(称为“零电势点”)。



简单任务 点亮板载灯

认识新代码块

- “数字输出”代码块位于“输入/输出”模块分类中。
- 用于指定管脚电平值。
- 每个数字/模拟管脚（**A6,A7**除外）都支持数字输出。

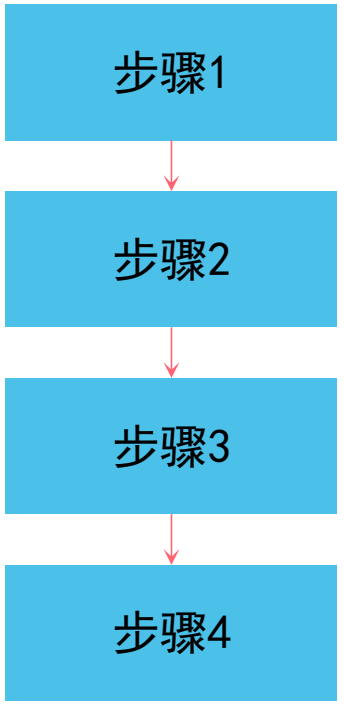
简单任务 点亮板载灯

认识新代码块

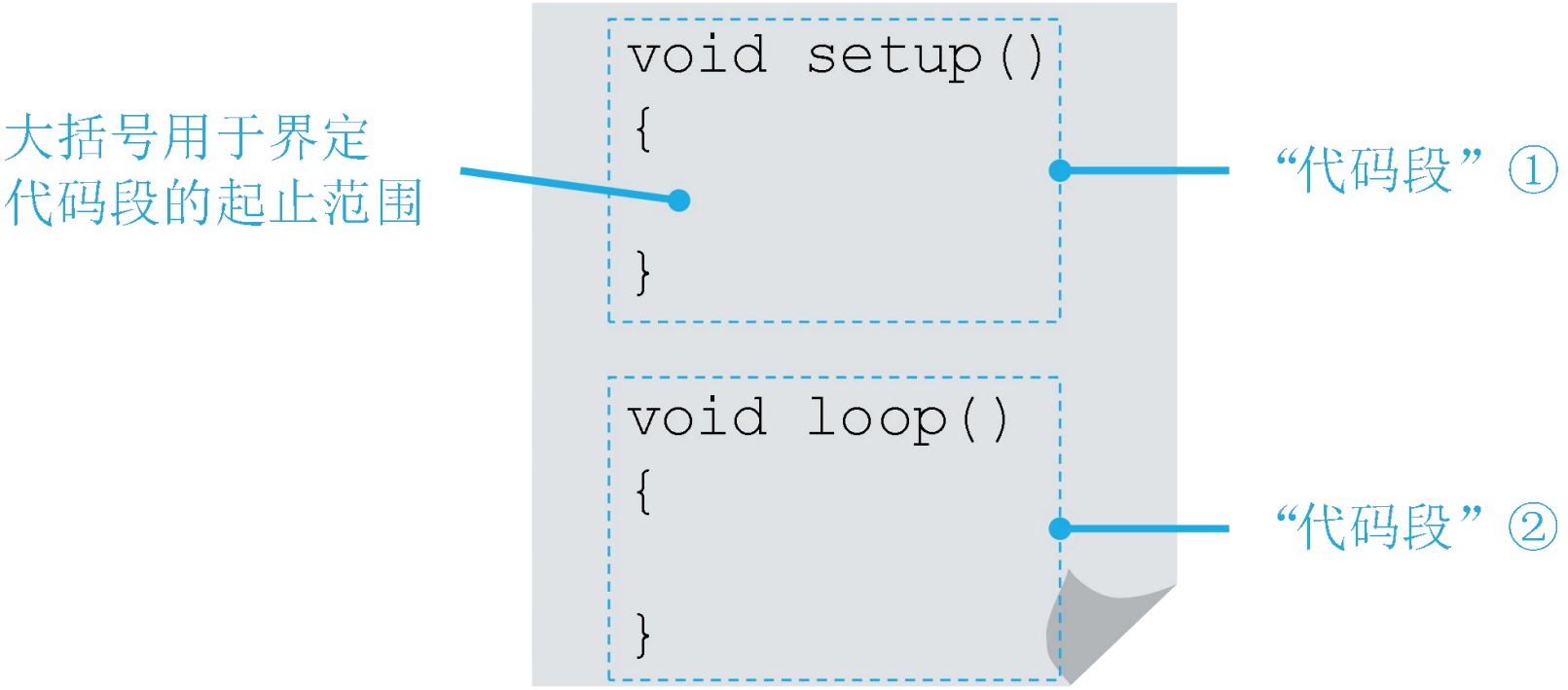
- “延时”代码块位于“控制”模块分类中。
- 在这段时间内，保持之前指令造成的状态，不做任何改变。

知识点讲解 顺序结构

- 在程序设计中，逐条书写、从上往下依次逐条执行的结构被称为顺序结构。
- 顺序结构程序设计是最简单的程序设计。

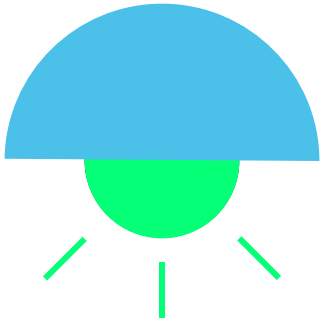


Arduino程序的架构



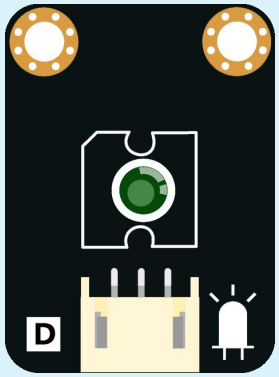
任务发布

使用绿色LED灯，编写程序实现如下效果：绿色LED灯以1秒为间隔不停闪烁。



0 准备好了吗？

模块清单

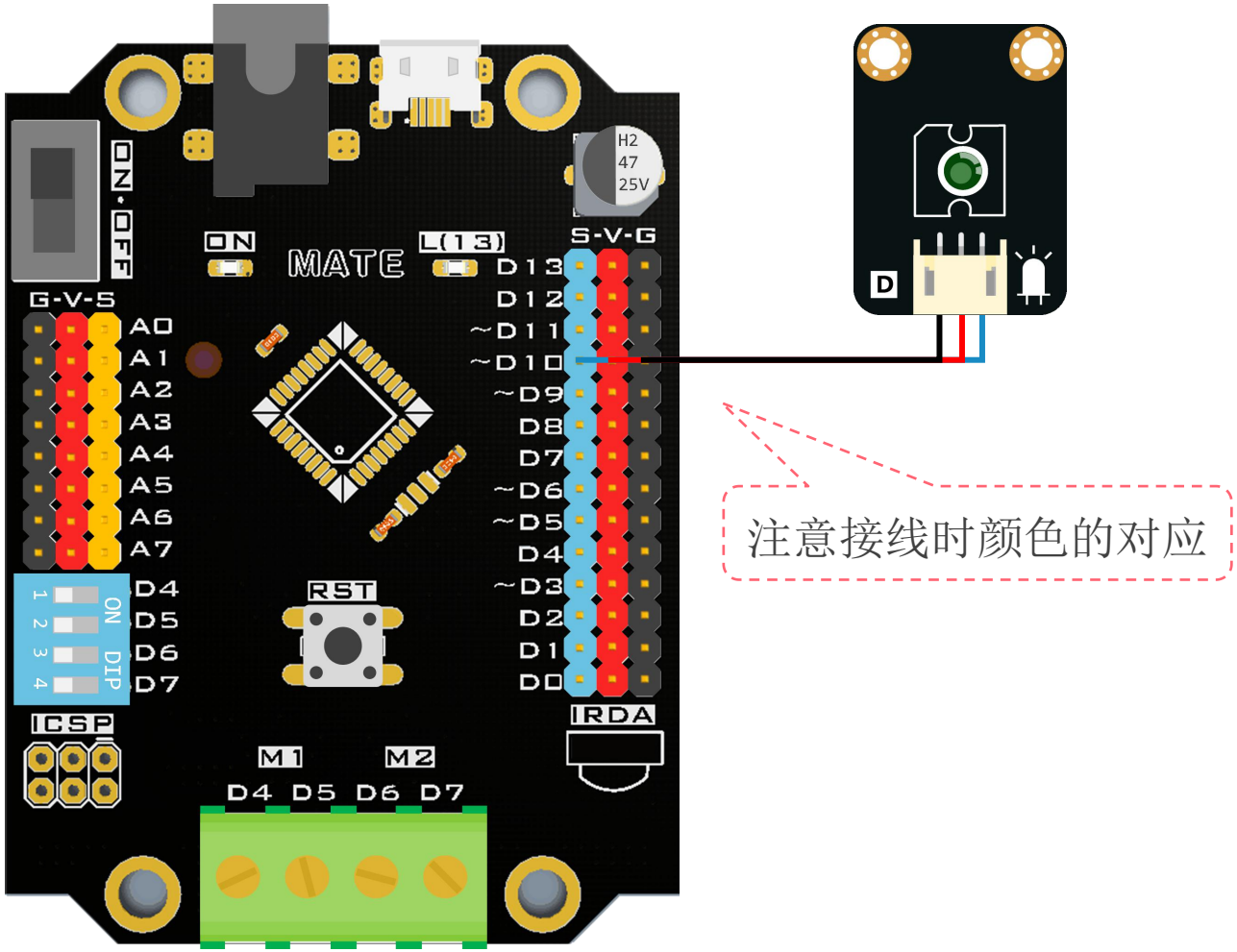


绿色LED×1

准备好了吗？

简单任务 点亮外部LED灯(2)

硬件连接



简单任务 点亮外部LED灯(2)

软件编写

课后练习

1. 请尝试同时点亮两个LED灯。

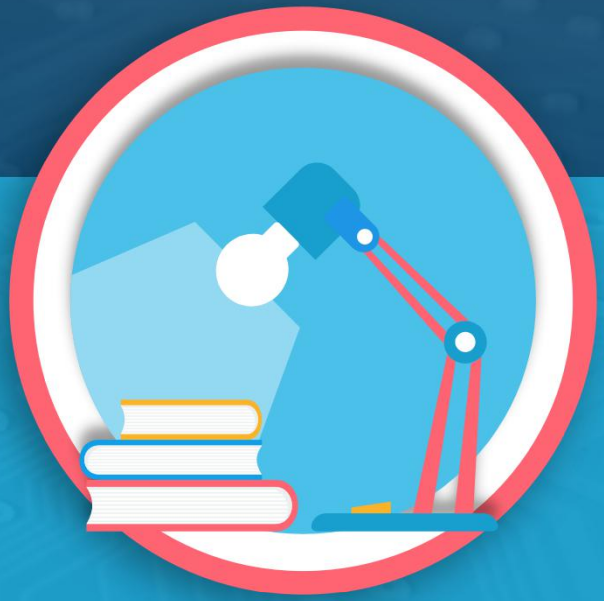
2. 请改变“延时”代码块中填写的数字，比如100、10、1，你发现了什么？请将你的发现填入下表。

间隔时间	现象
100ms	
10ms	
1ms	

第一单元 点亮创客之路

准备好了吗？

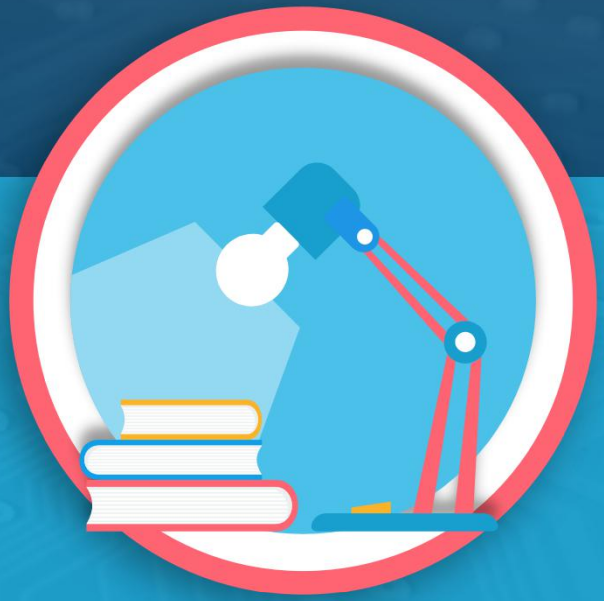
起始课



第一单元 点亮创客之路

一闪一闪亮晶晶

第1课



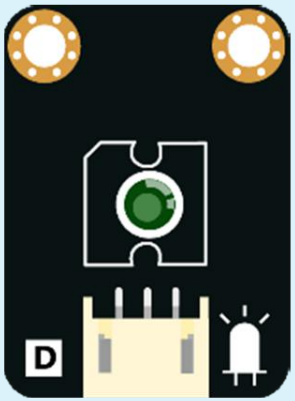
《创意电子入门》配套教学课件

1 一闪一闪亮晶晶

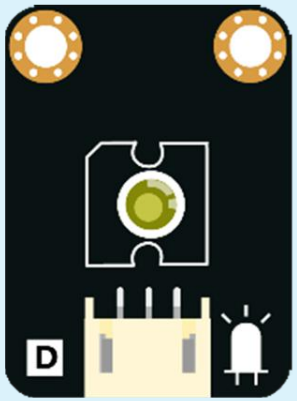
情境引入

1 一闪一闪亮晶晶

模块清单



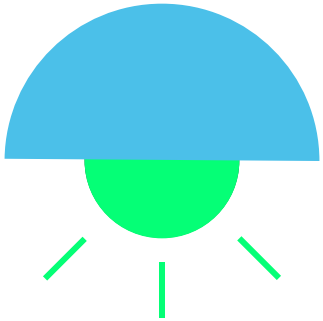
绿色 LED 灯×1



黄色 LED 灯×1

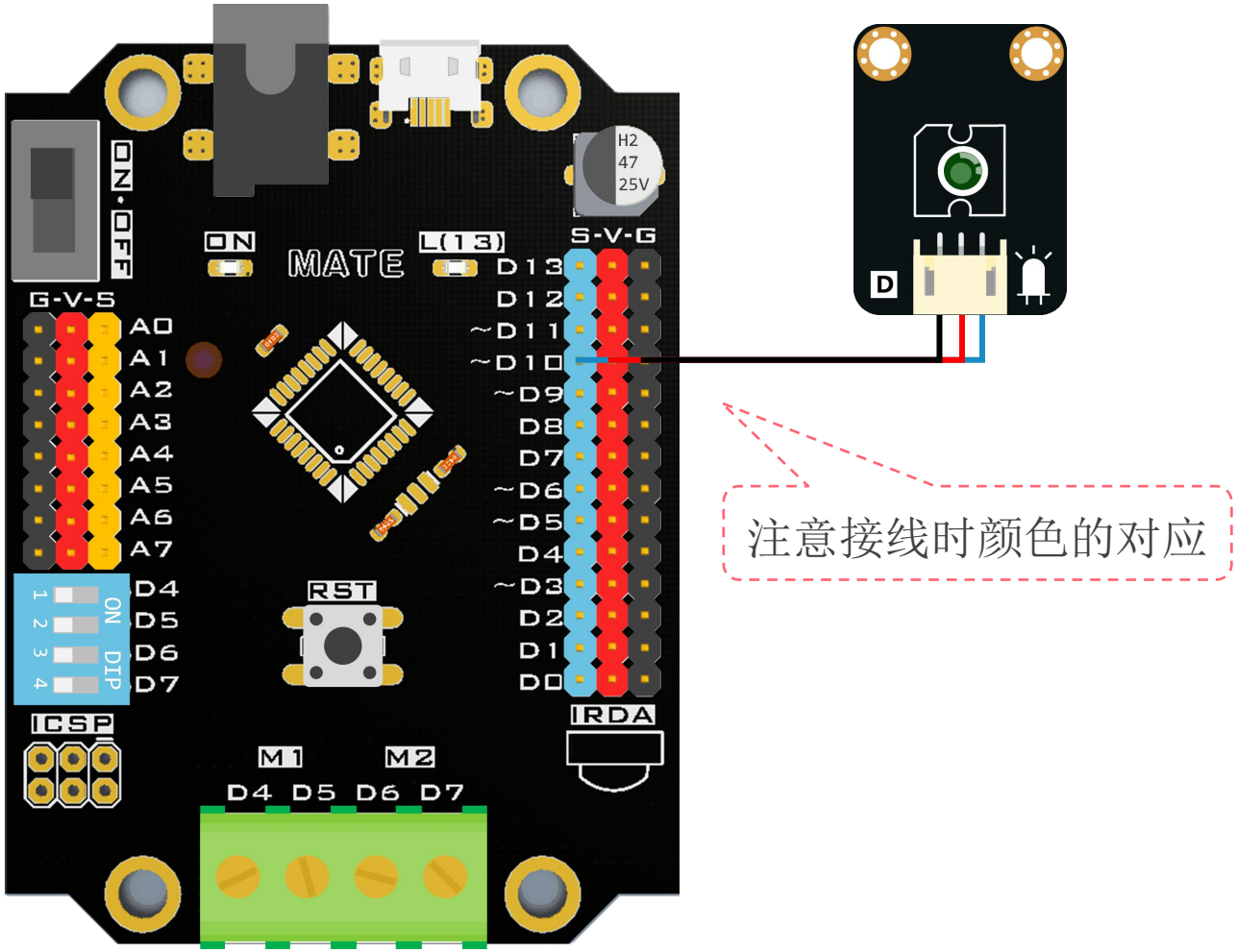
任务发布

使用绿色LED灯，编写程序实现如下效果：绿色LED灯以1秒为间隔不停闪烁。请同学们在模仿的过程中，注意体会引入“变量”来优化程序的好处。

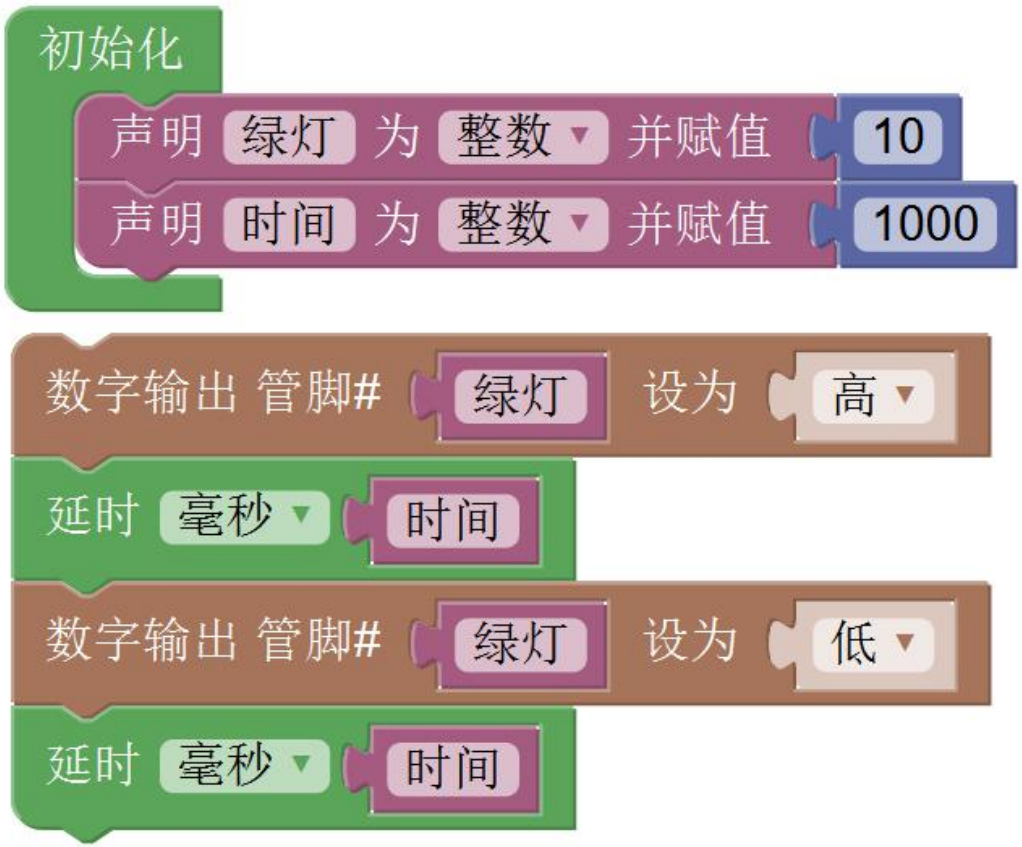


简单任务 点亮外部LED灯(2)

硬件连接



软件编写


初始化
声明 绿灯 为 整数 并赋值 10
声明 时间 为 整数 并赋值 1000
数字输出 管脚# 绿灯 设为 高
延时 毫秒 时间
数字输出 管脚# 绿灯 设为 低
延时 毫秒 时间

认识新代码块

- “数字”代码块位于“数学”模块分类中
- 用于放置一个确定的数字

认识新代码块

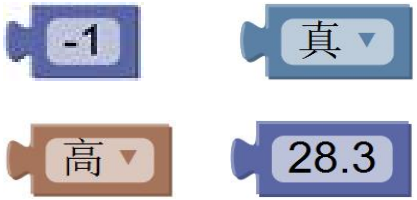
- “声明变量”代码块位于“变量”模块分类中
- 用于声明并初始化一个变量

认识新代码块

- “初始化”代码块位于“控制”模块分类中
- 用于存放只在程序开始运行时执行一次的代码块

知识点讲解 常量与变量

- **常量**用来直接表示数据的值或者内容。



- **变量**不但可以用来表示数据的值或内容，也可以用来存放不同的数据。
- 声明变量时必须要给它起个名字(叫做变量名)并指定它的**数据类型**。

「玩具盒」

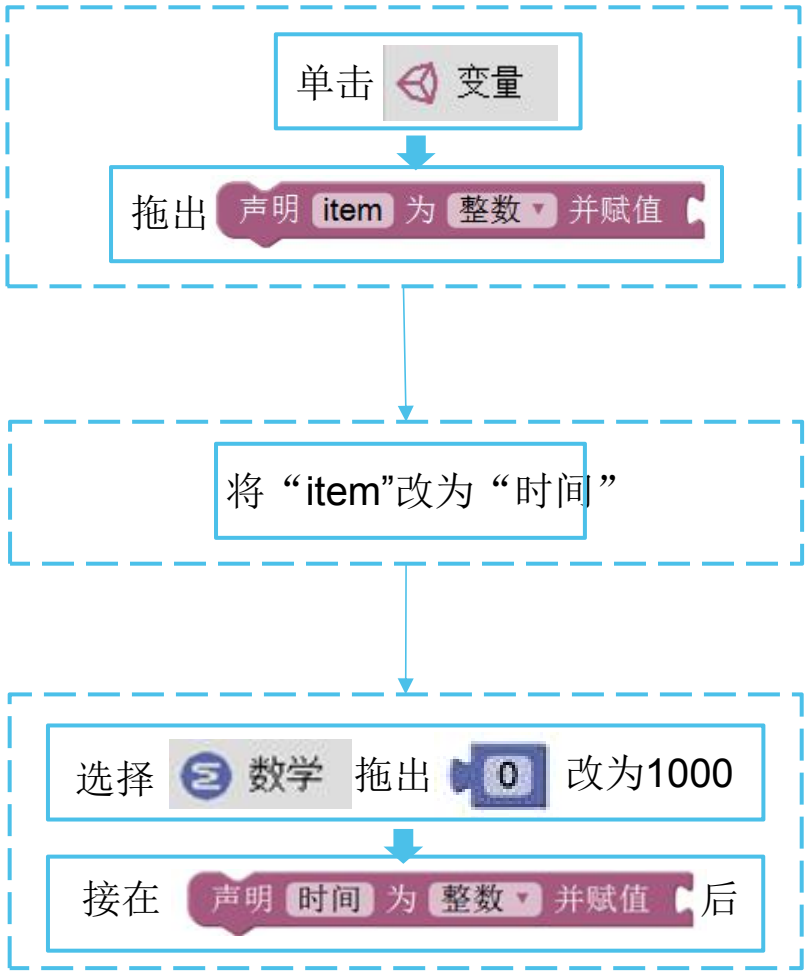
「工具盒」

知识点讲解 变量的声明与初始化

- 新建变量

- 变量重命名
  - ✓ 区分大小写
  - ✓ 尽可能使用有意义的文字
  - ✓ 避免使用有特殊意义的保留字

- 变量赋初值



- 整数型是在Mixly的应用中最常见的数据类型。
- 单击 整数 右侧的下拉菜单按钮，可以看到其他数据类型。

通过下表可以确定适合我们日后编写的程序所需数值范围的数据类型：

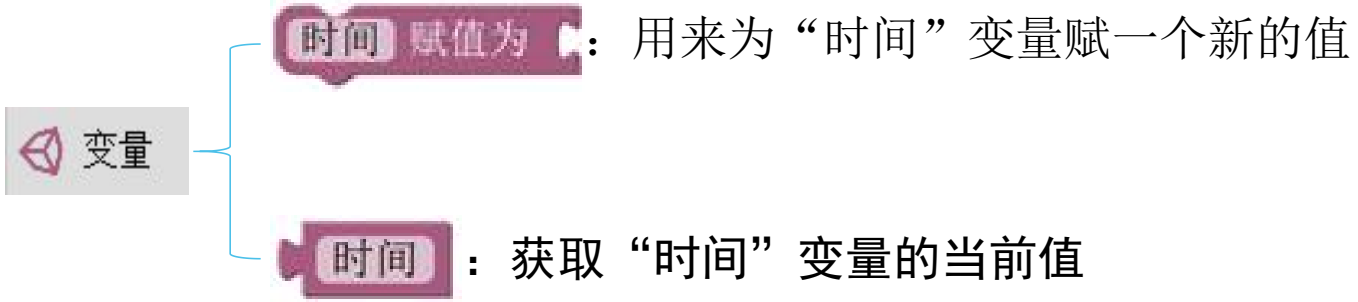
数据类型	取值范围	使用
整型	-32768~32767	表示范围内的任意整数值
长整型	-2147483648~2147483647	表示较大范围内的任意整数值
小数	-3.402×10^{38}~3.402×10^{38}	表示带小数的数字，用来近似真实世界的测量值
布尔	true(真,1)或false(假,0)	表示真假
字节	0~255	代表单个字节
字符	-128~127	代表单个字符，也表示范围内的整数值
字符串	——	表示一串字符

- 声明变量并赋初值的操作在程序一开始就要执行；
- 初始化只需要执行一次；
- 这类操作我们应该将其放到 

模块中。

例如：

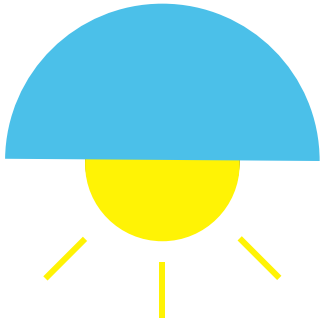
知识点讲解 变量的使用

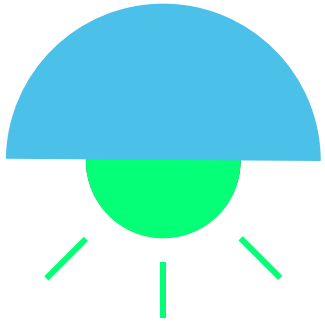


将每盏灯亮的时间赋值成1000毫秒：

任务发布

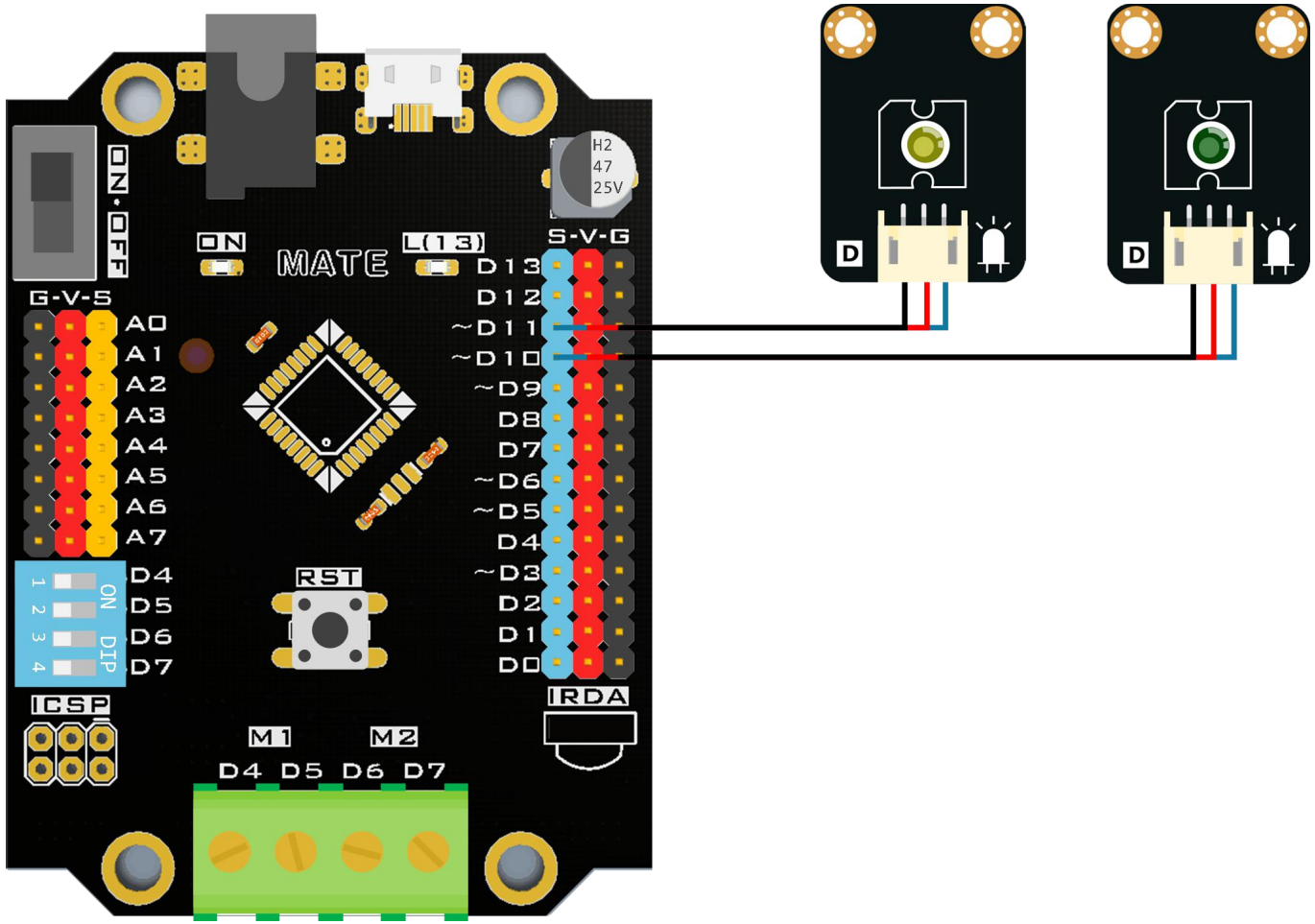
请使用两个LED灯，编写程序，完成如下的实验效果：两个LED灯交替亮灭，每2秒钟完成一组交替。



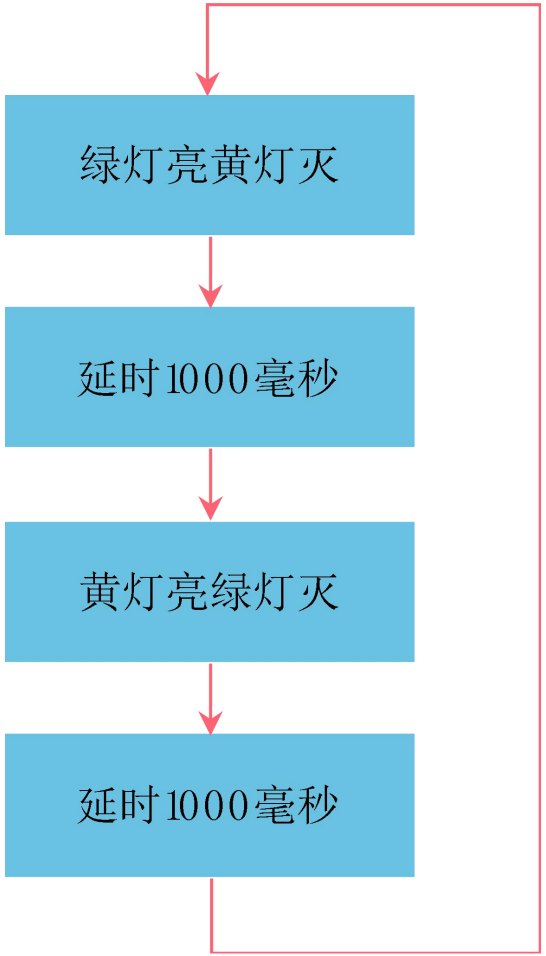


可扩展任务 一闪一闪亮晶晶

硬件连接



编程思路

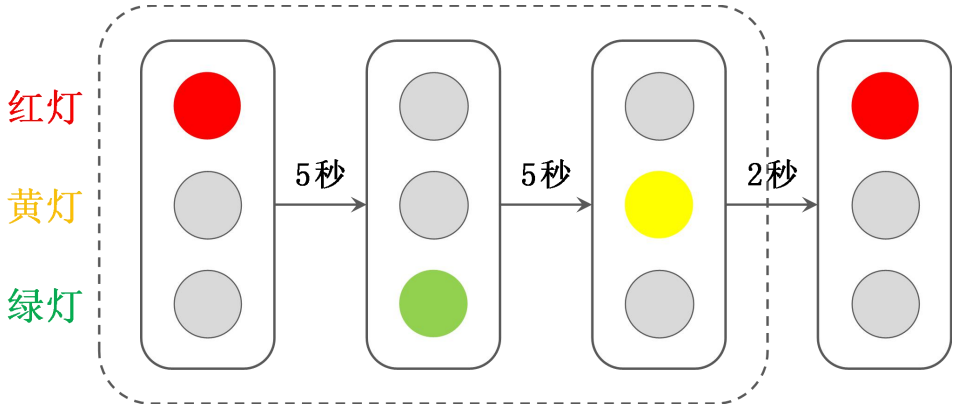


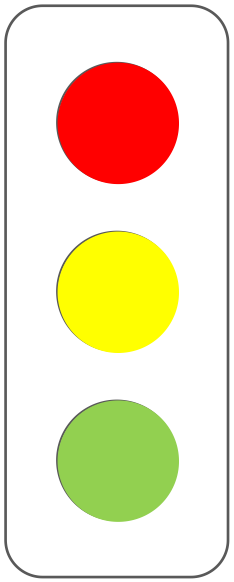
可扩展任务 一闪一闪亮晶晶

软件编写

任务发布

请使用三个LED灯，编写程序，模拟交通信号灯的效果。

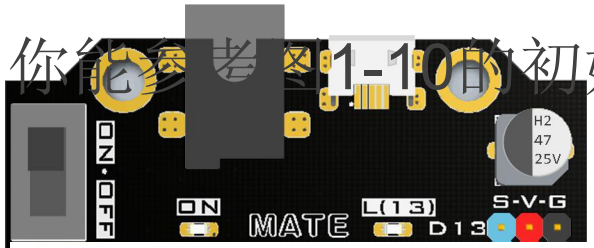


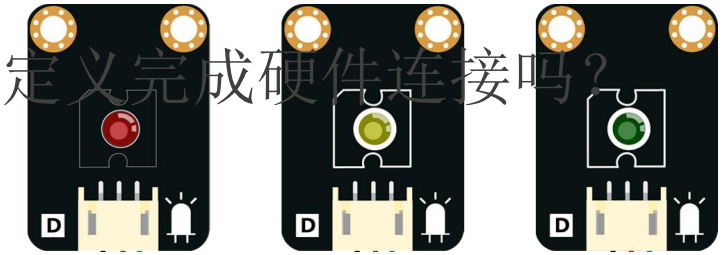


自主扩展任务 简单交通信号灯

硬件连接

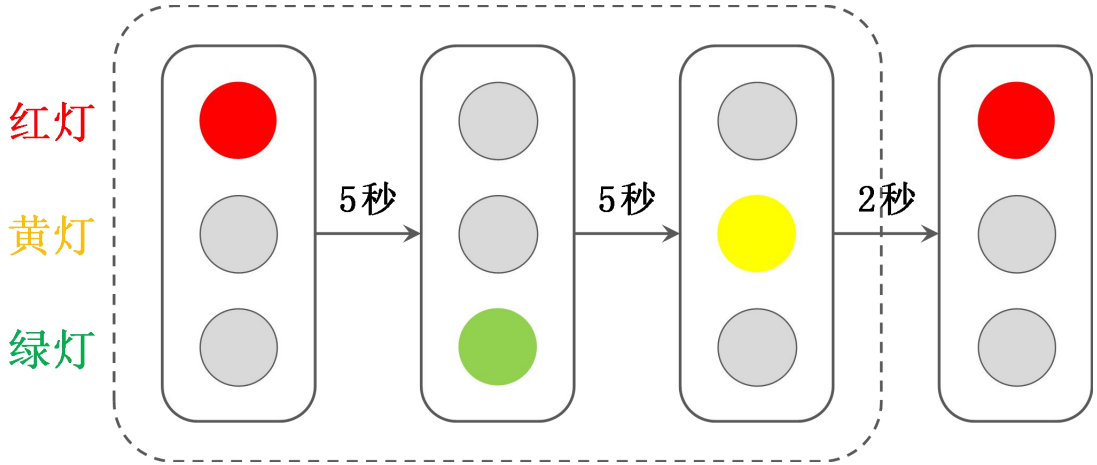
你能参考图1-10的初始化定义完成硬件连接吗？

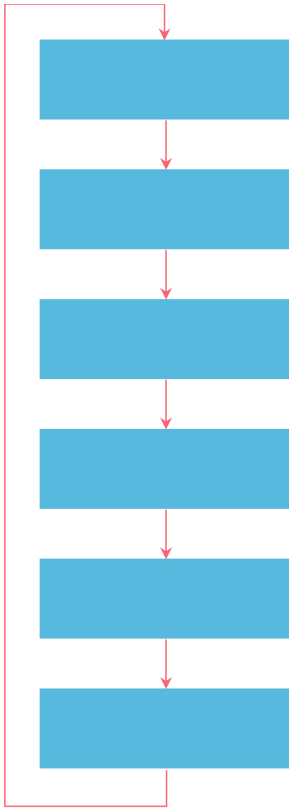




软件编写

你能否结合下面的红绿灯工作示意图，参考可扩展任务的“编程思路”单元，将右侧的程序框图补充完整呢？





课后练习

1. 制作一个流水灯，按下图①→②→③→④的顺序依次点亮。

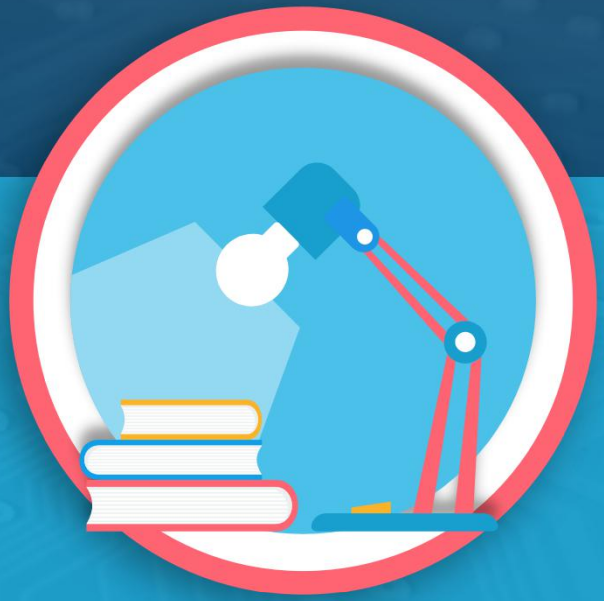
①　②　③　④

2. 在上个任务的基础上，按上图①→②→③→④→③→②的顺序依次点亮。

第一单元 点亮创客之路

延时灯

第2课



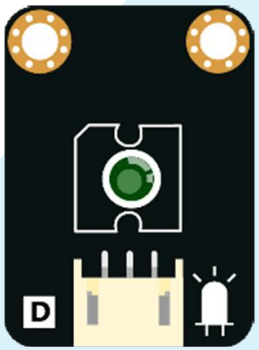
《创意电子入门》配套教学课件

2 延时灯

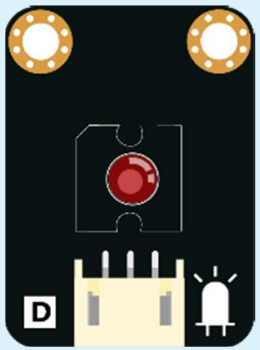
情境引入

2 延时灯

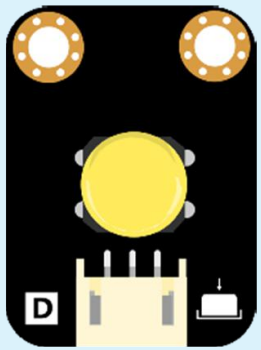
模块清单



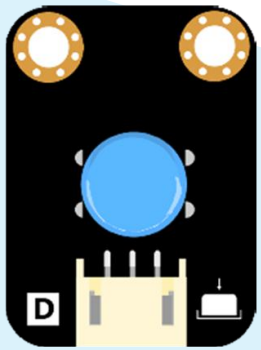
绿色 LED 灯×1



红色 LED 灯×1



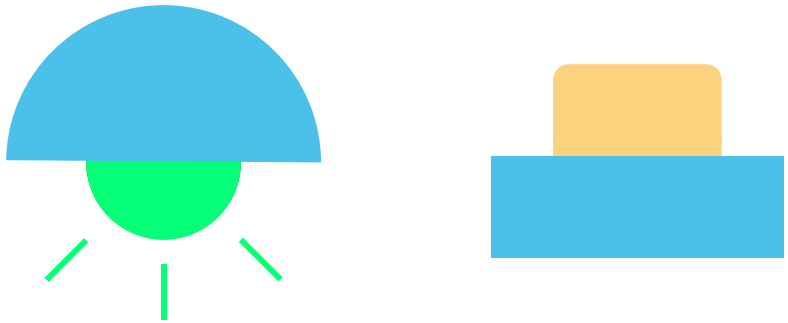
黄色按钮×1



蓝色按钮×1

任务发布

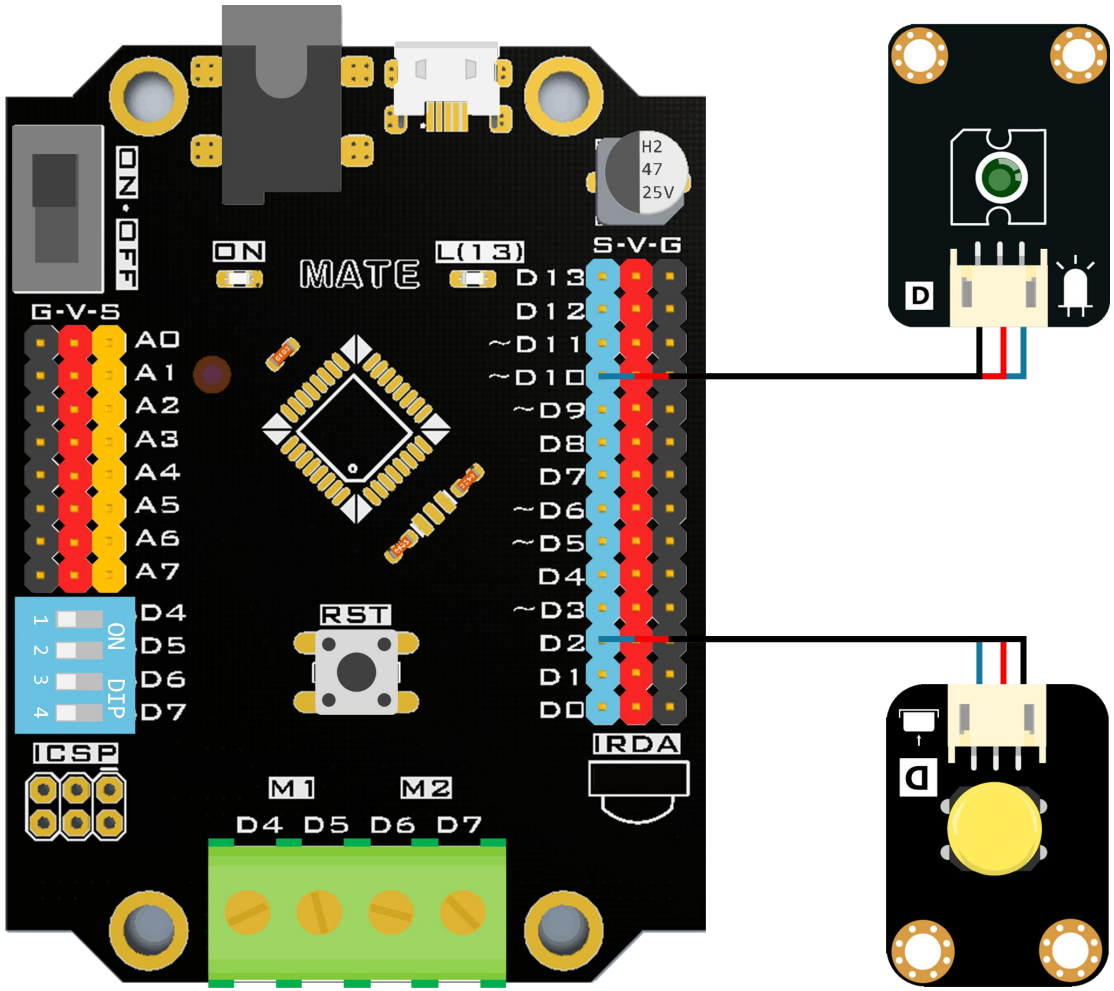
使用按钮和LED灯，编写程序实现以下实验效果：程序上传后，按下按钮时，LED灯点亮；松开按钮时，LED灯熄灭。



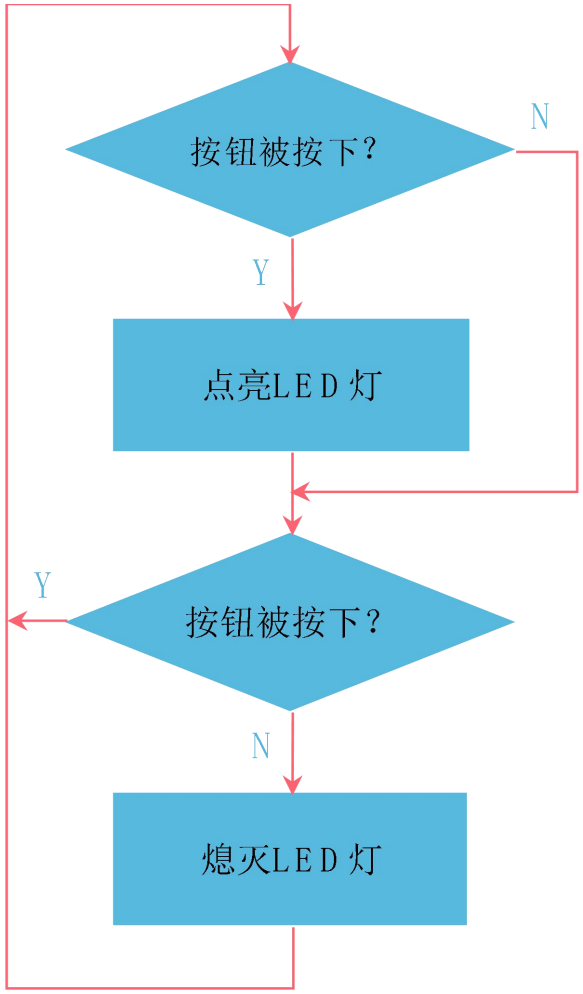
2 延时灯

简单任务 简单按键灯

硬件连接



编程思路



软件编写

2 延时灯

简单任务 简单按键灯

认识新代码块

- “数字输入”代码块位于“输入/输出”模块分类中
- 用于获取指定管脚的电平值，有“高”和“低”两种状态。

认识新代码块

- “如果”代码块位于“控制”模块分类中。
- “执行”右侧的口插入当判断条件为“真”时执行的代码；当判断条件的结果为“假”时，则不执行这段代码。

认识新代码块

- “比较运算”代码块位于“逻辑”模块分类中。
- 包括大于、等于、小于、小于等于、不等于、大于等于等多种逻辑判断运算。

认识新代码块

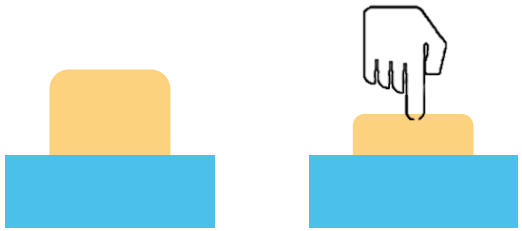
- “高/低”代码块位于“输入/输出”模块分类中。
- 默认值为高。

知识点讲解 数字输入

- Arduino的**所有数字和模拟管脚**都能读取/输出高与低电平。
- 用来读取管脚电平值的块叫做**数字输入**。
- 输入值超过电源电压(5V)的一半，就得到高电平，否则得到低电平。

从2号数字管脚读取的电平值

- 按钮是一种常用的控制电器元件，常用来接通或断开“控制电路”(其中电流很小)。
- 按钮可以连接在主控板的除了0、1之外的任何一个管脚上。

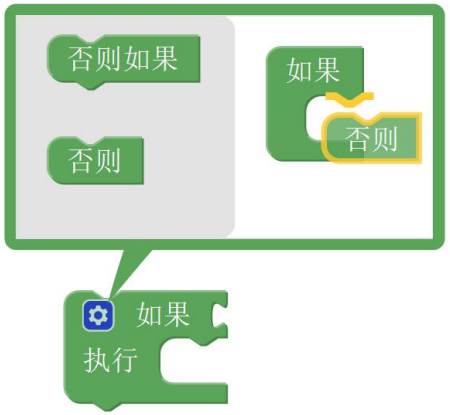


【低电平】 【高电平】

- 程序中，依照**是否满足条件来决定是否执行特定指令**的控制结构叫做“条件结构”。即：“如果……就……”，其对应的代码块为 

。

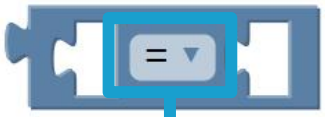
知识点讲解 条件结构的进阶使用



【在“如果-执行”中拖出“否则”】

【使用“如果-否则”改写程序】

知识点讲解 比较运算符

比较运算块  在  模块分类下

点击，得到

6个比较运算符：

- ✓ 用来比较块左右两侧的值的大小关系。
- ✓ 条件判断中，经常会用到比较运算符。
- ✓ 比较之后的结果会返回**真**(代表**条件成立**)或**假**(代表**条件不成立**）

- Mixly中涉及到的比较运算符和说明请参见下表：

符号	说明	符号	说明
=	如果两者相等则条件成立	≠	如果两者不等则条件成立
<	如果左边小于右边则条件成立	>	如果左边大于右边则条件成立
≤	如果左边不大于右边则条件成立	≥	如果左边不小于右边则条件成立

2 延时灯

知识点讲解 “高/低”“真/假”与“1/0”

- 在Mixly中，“高”“真”“1”这三个概念是一致的；“低”“假”“0”这三个概念也是一致的。

返回值为“高”时

结果为真

- 将简单按键台灯的程序简化：

结束

重新审视图2- 11的代码并思考下面的问题：

1. 按钮按下时，2号管脚读取到_____电平，此时10号管脚应输出_____电平。

✓ 从2号管脚读取的电平值与10号管脚应输出的电平值相同。

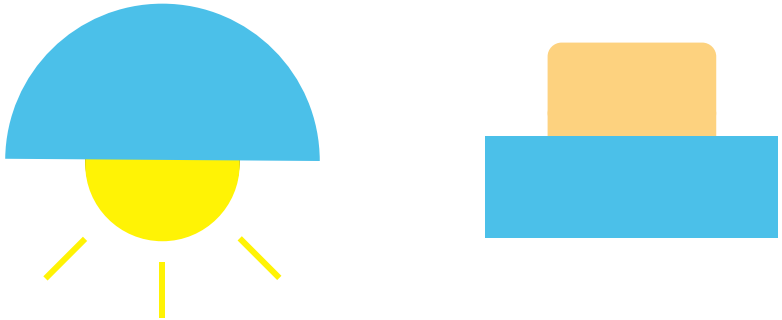
✓ 语句之所以能够工作是因为它在不断地被重复执行。

2. 按钮抬起时，2号管脚读取到_____电平，此时10号管脚应输出_____电平。

简单按键台灯优化后程序：

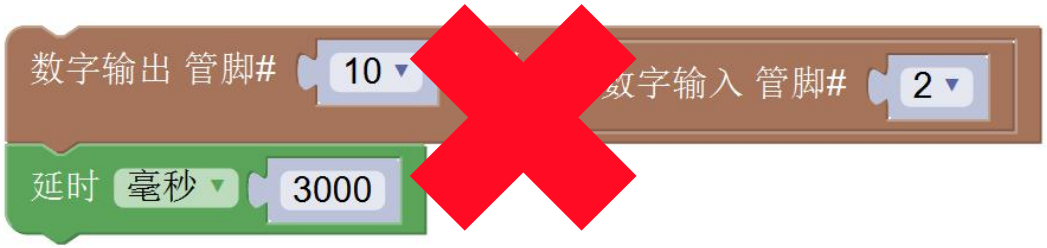
任务发布

请使用按钮和LED灯，编写程序，完成如下的实验效果：按下按钮，LED灯立刻被点亮；松开按钮3秒后，LED灯熄灭。



编程思路

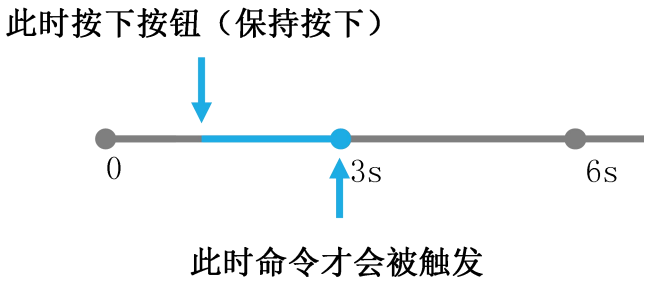
- 下面的程序是否可以达到实验效果？

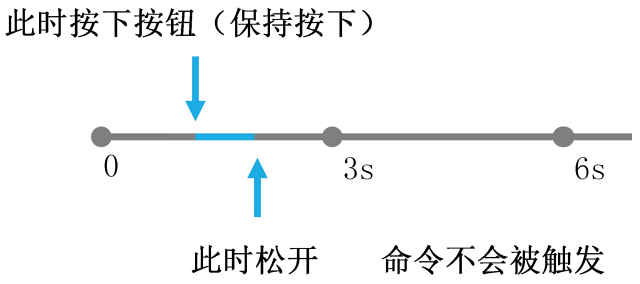


- 在Arduino中，程序会被循环执行。除了数字输出语句执行的那一刻外，其余的时间程序都被延时语句“锁住”了。

编程思路

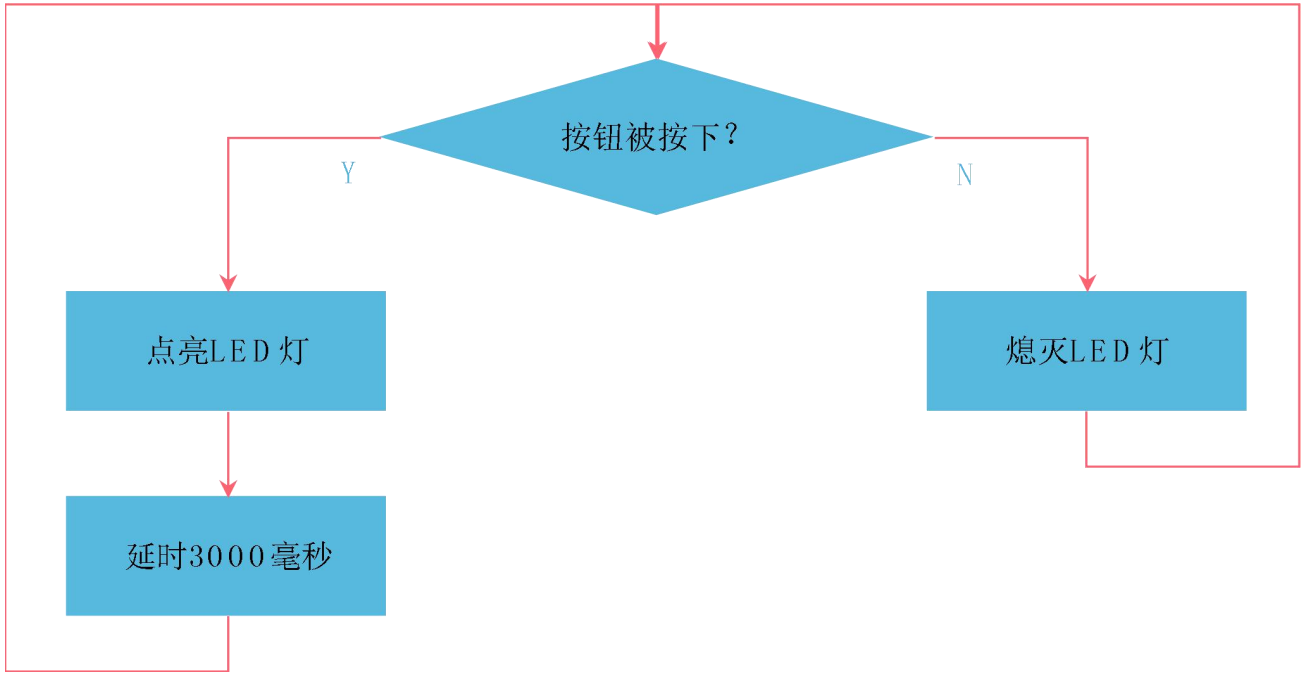
- 在Arduino中，程序会被循环执行。除了数字输出语句执行的那一刻外，其余的时间程序都被延时语句“锁住”了。





可扩展任务 延时灯

编程思路



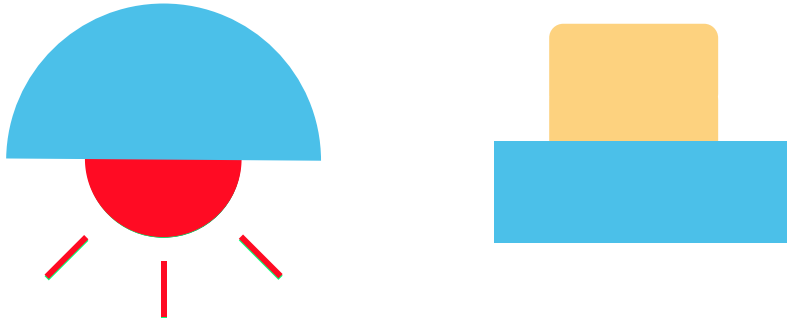
2 延时灯

可扩展任务 一闪一闪亮晶晶

软件编写

任务发布

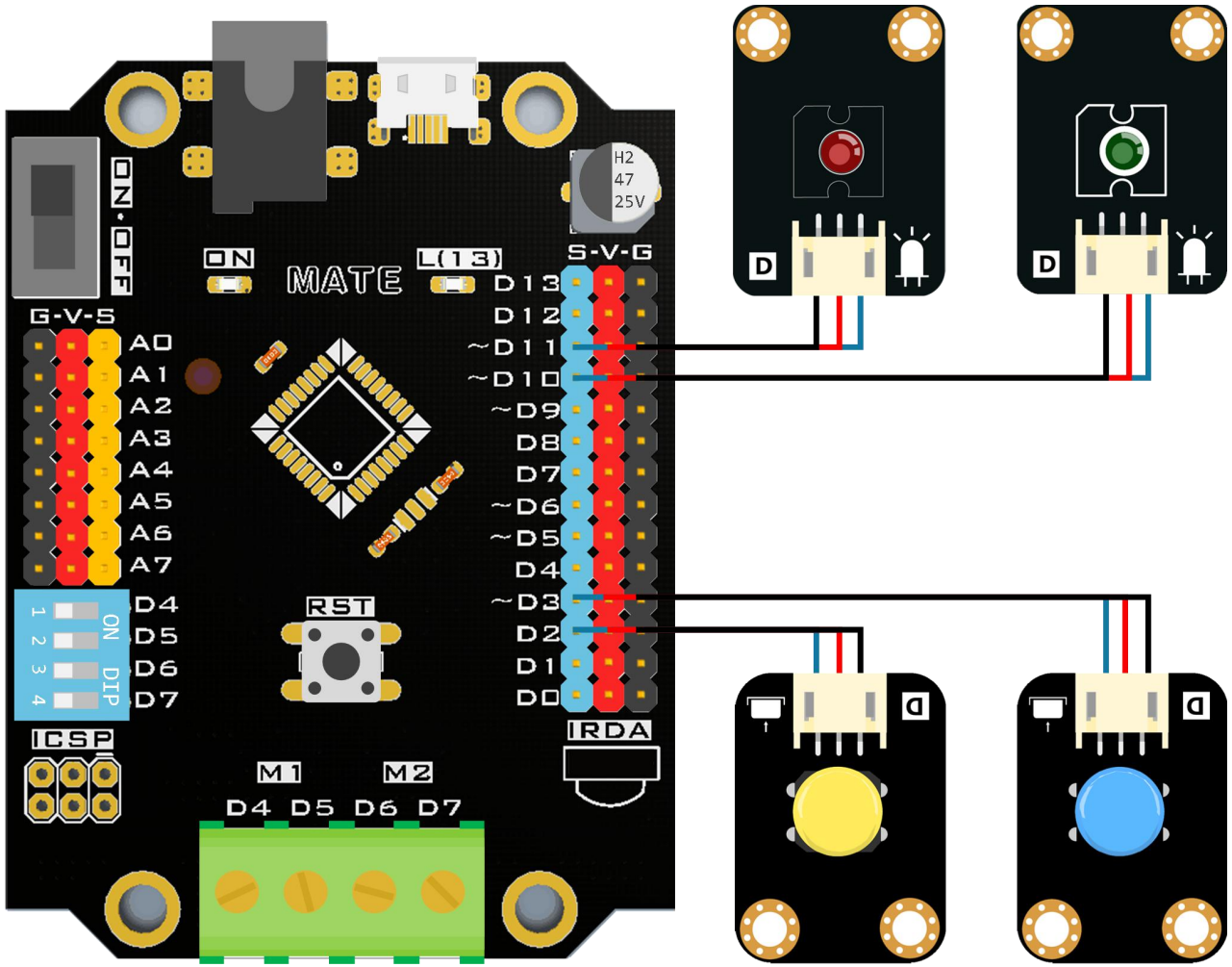
请使用多个按钮，编写程序，实现第一个按钮按下红灯亮绿灯灭，第二个按钮按下红灯灭绿灯亮的效果。



2 延时灯

自主扩展任务 模拟收费站

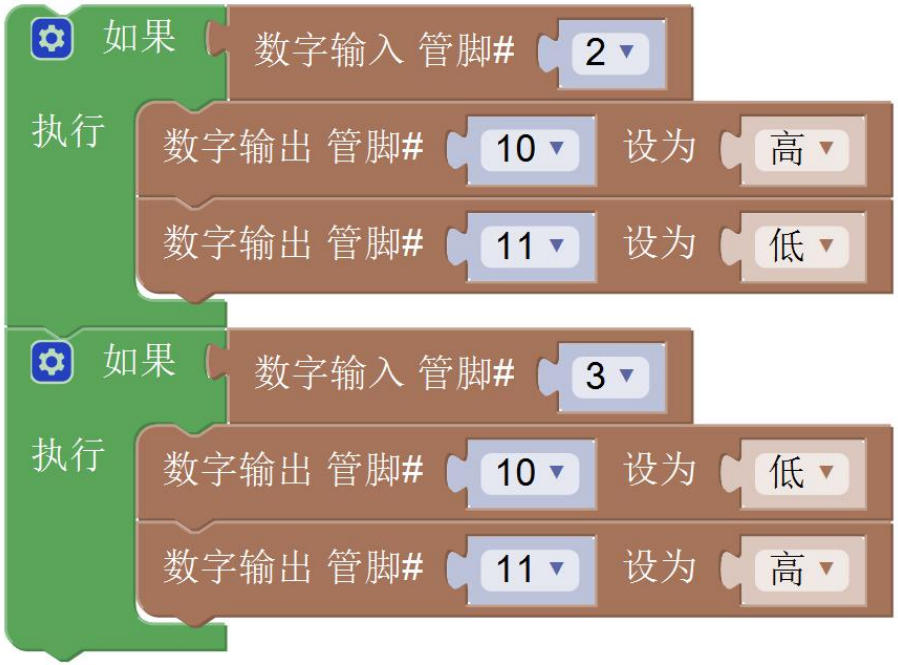
硬件连接



2 延时灯

自主扩展任务 模拟收费站

软件编写



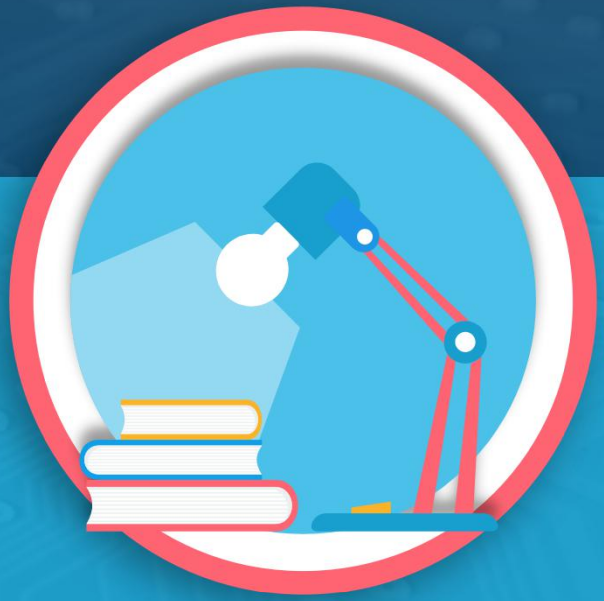
1. 请使用多个按键，实现一个按钮控制灯亮，另一个按钮控制灯灭。

2. 在练习1的基础上，让控制灯灭的按钮按下后3秒钟灯再熄灭。

第一单元 点亮创客之路

流光沙漏

第3课



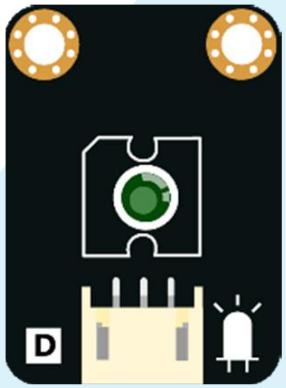
《创意电子入门》配套教学课件

3 流光沙漏

情境引入

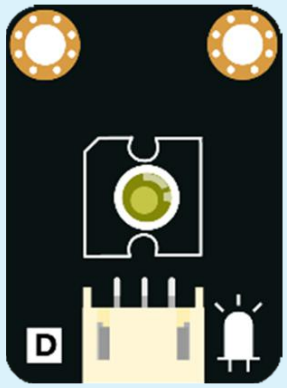
3 流光沙漏

模块清单



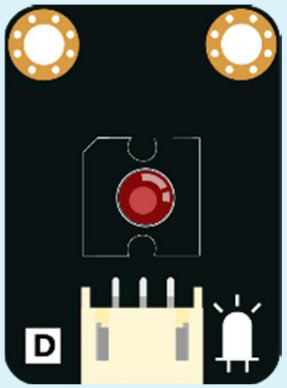
绿色 LED 灯

×1



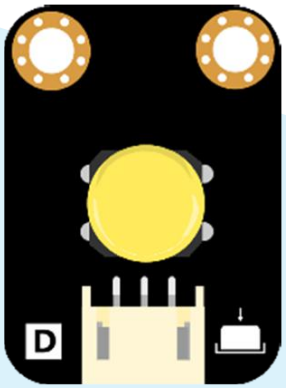
黄色 LED 灯

×1



红色 LED 灯

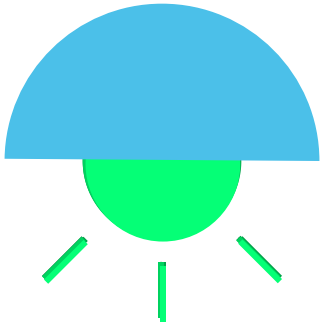
×1



黄色按钮×1

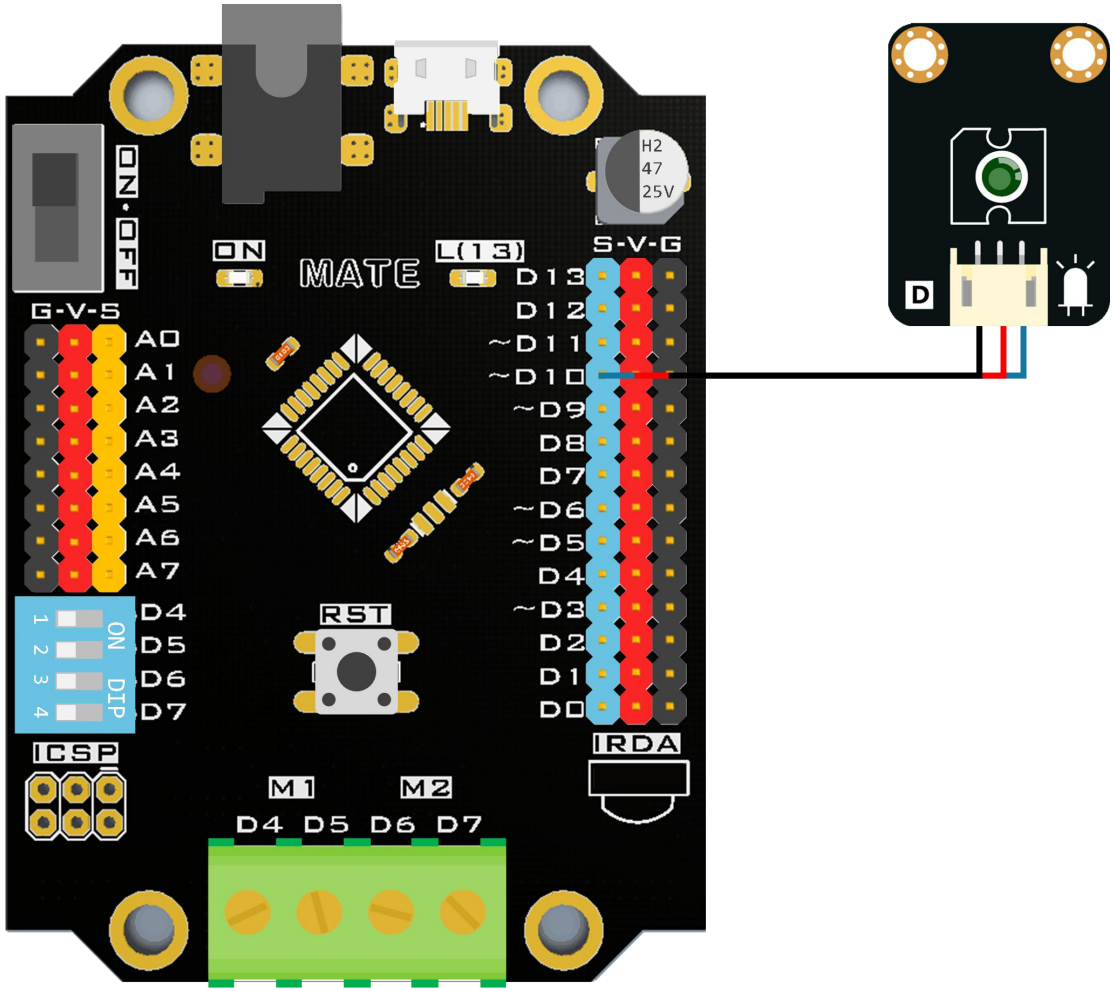
任务发布

呼吸灯的灯光在微电脑的控制下，可以完成由暗到亮再由亮到暗的逐渐变化的过程，感觉像是在呼吸。请同学们使用LED灯，编写程序完成如下的实验效果：程序上传后，LED灯慢慢变亮。



简单任务 呼吸灯

硬件连接



简单任务 呼吸灯

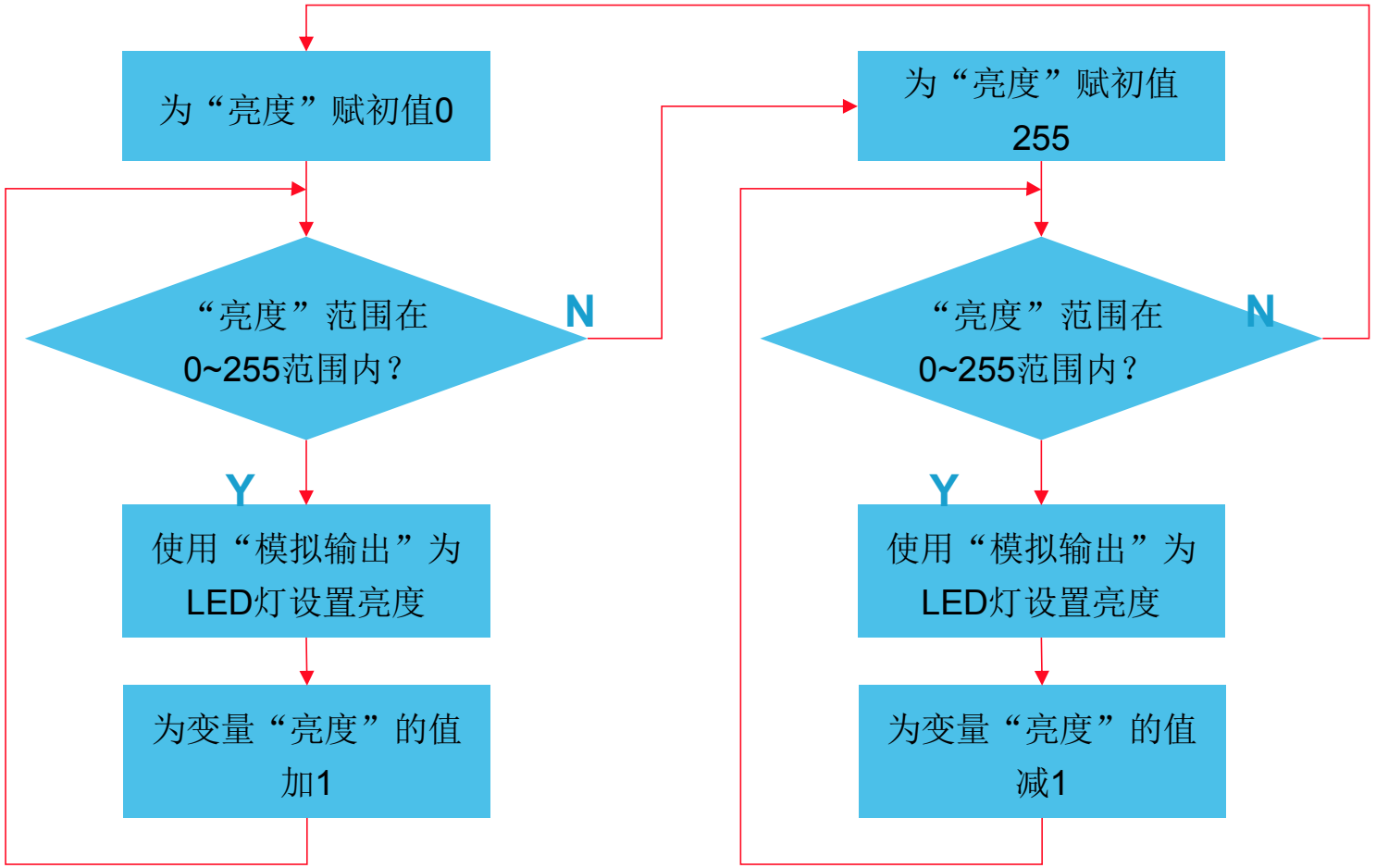
认识新代码块

- 模拟数值介于**0~255之间**，对应输出0~5V之间的仿真模拟电压值。

认识新代码块

- “使用i从1到10步长为1”代码块位于“控制”模块分类中。
- 其作用是：为变量(如i)从起始数(如1)到结尾数(如10)按指定的步长(间隔，如1)赋值，并执行指定的模块。

编程思路


为“亮度”赋初值0
“亮度”范围在0~255范围内？
N
Y
使用“模拟输出”为LED灯设置亮度
为变量“亮度”的值加1
为“亮度”赋初值255
“亮度”范围在0~255范围内？
N
Y
使用“模拟输出”为LED灯设置亮度
为变量“亮度”的值减1

软件编写


使用 亮度 从 0 到 255 步长为 1
执行
模拟输出 管脚# 10 赋值为 亮度
延时 毫秒 5
使用 亮度 从 255 到 0 步长为 -1
执行
模拟输出 管脚# 10 赋值为 亮度
延时 毫秒 5

- 很多情况下，程序需要重复执行一个程序段。这种结构被称为循环结构，这个被重复执行的程序段被称为循环体。

此块可用于实现循环结构。

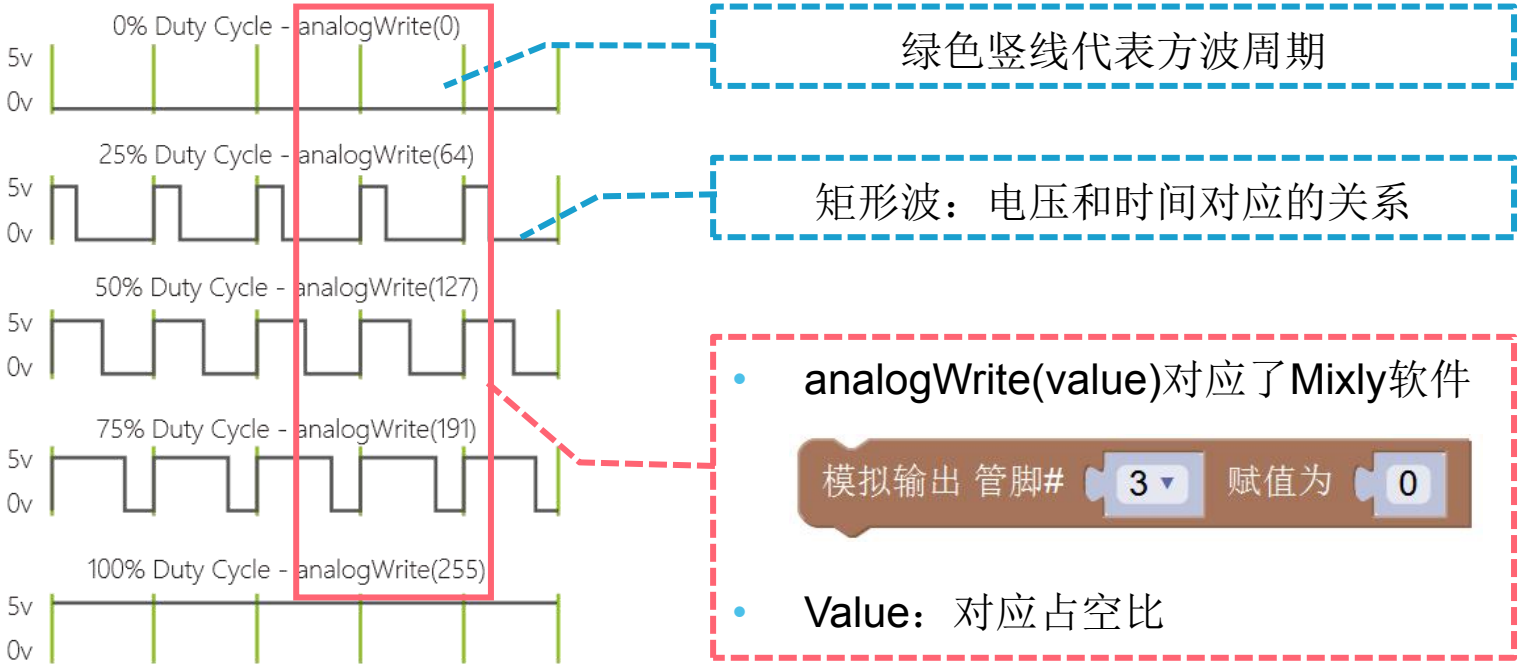
思考：标红程序块将被执行几次？

- 请尝试分别将下列代码上传到主控板上，观察效果：

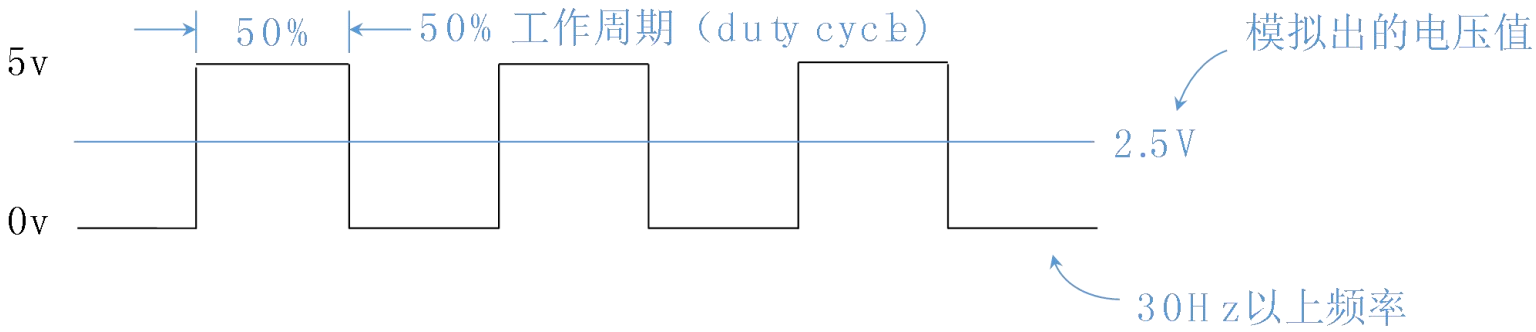
- **左侧LED灯亮度 > 右侧LED灯亮度**

知识点讲解 模拟输出

- Arduino可以通过**脉宽调制技术(PWM)**来控制LED灯的明亮度。
- 在Mate主控板上，有六个数字管脚标有“~”，这些管脚都具有PWM功能。

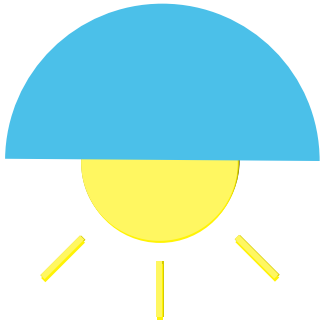


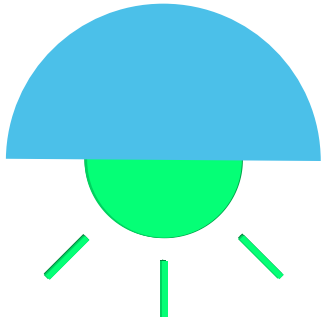
- 下图示例中占空比为50%：



任务发布

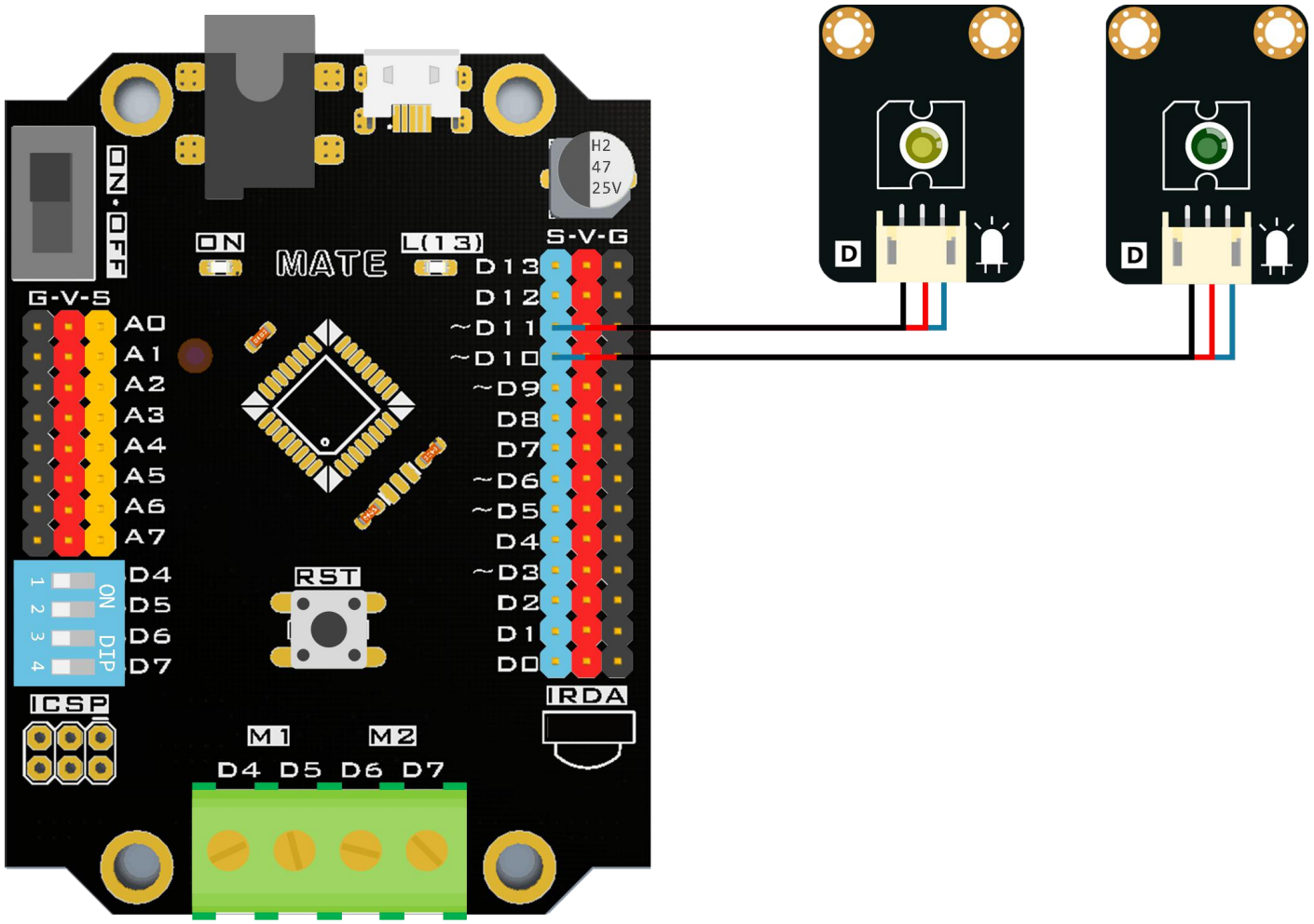
流光沙漏的两个LED灯的灯光互补，即当一个逐渐变暗时，另一个逐渐变亮。请同学们使用两个LED灯，编写程序，用LED灯的亮度模拟沙漏中的沙子。LED灯越亮表示沙子越多。



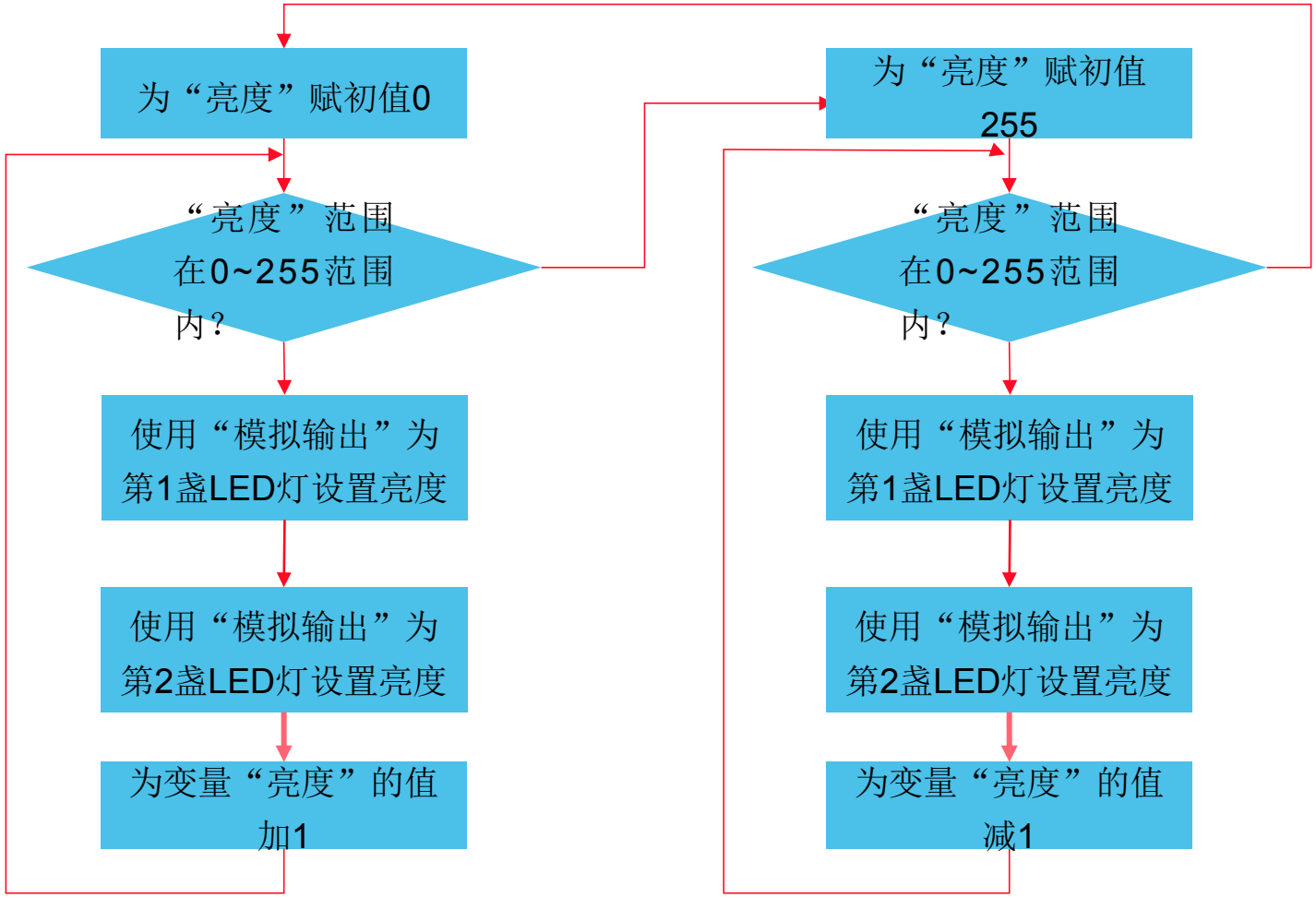


可扩展任务 流光沙漏

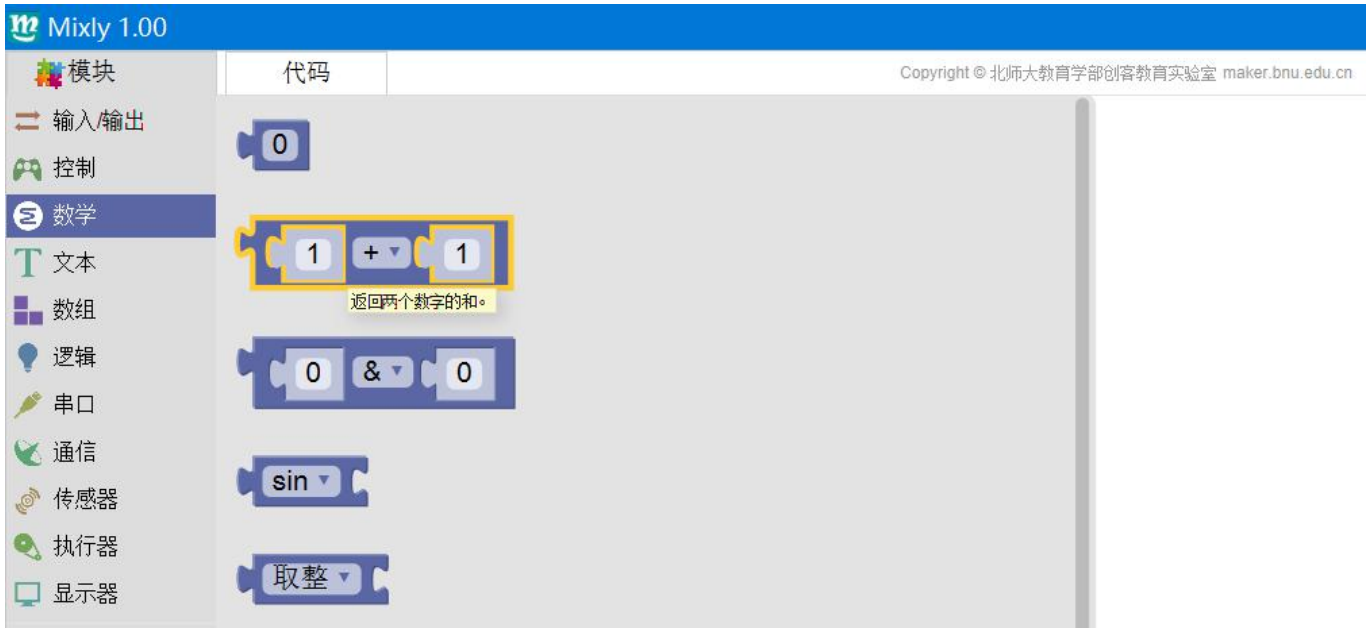
硬件连接



编程思路


为“亮度”赋初值0
“亮度”范围在0~255范围内？
使用“模拟输出”为第1盏LED灯设置亮度
使用“模拟输出”为第2盏LED灯设置亮度
为变量“亮度”的值加1
为“亮度”赋初值255
“亮度”范围在0~255范围内？
使用“模拟输出”为第1盏LED灯设置亮度
使用“模拟输出”为第2盏LED灯设置亮度
为变量“亮度”的值减1

认识新代码块



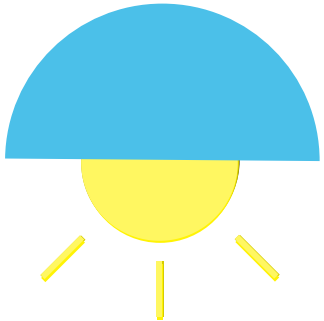
- “运算”代码块位于“数学”模块分类中。运算代码块支持6种运算：加(+)、减(-)、乘(×)、除(÷)、取余数(%)和幂运算(^)。

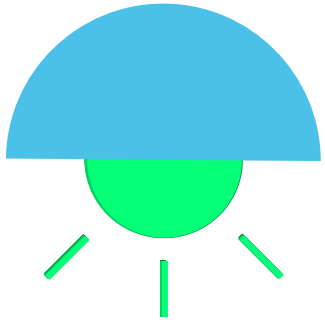
软件编写


使用 亮度 从 0 到 255 步长为 1
执行
模拟输出 管脚# 10 赋值为 亮度
模拟输出 管脚# 11 赋值为 255 - 亮度
延时 毫秒 5
使用 亮度 从 255 到 0 步长为 -1
执行
模拟输出 管脚# 10 赋值为 亮度
模拟输出 管脚# 11 赋值为 255 - 亮度
延时 毫秒 5

拓展思考

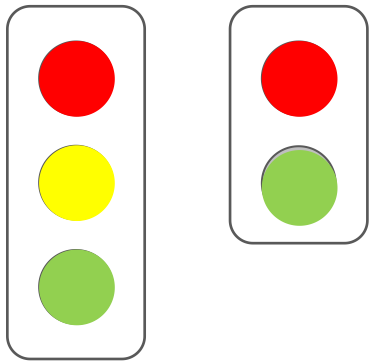
- 沙漏运动的方向和速度各是由什么变量控制的？





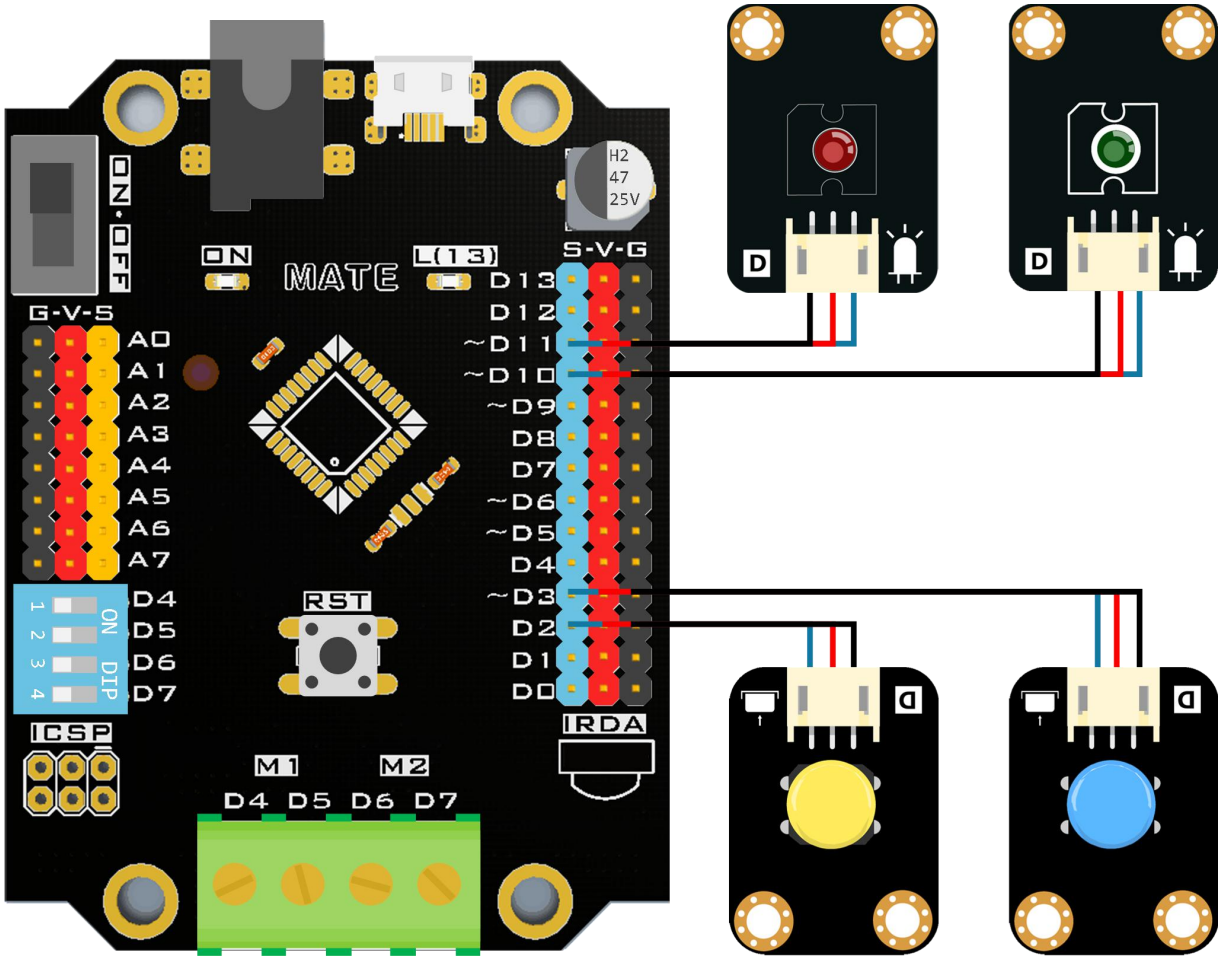
任务发布

- 使用五个LED灯。其中三个(红、黄、绿)代表汽车灯，另外两个(红、绿)代表对向的行人灯，例如：东西向为汽车灯，则南北向为行人灯。
- ①请参考第2课自主扩展任务的思路，完成两组红绿灯程序的编写。
- ②加入行人干预：只有当按下按钮(接在2号管脚)时，行人灯才会由红变绿。

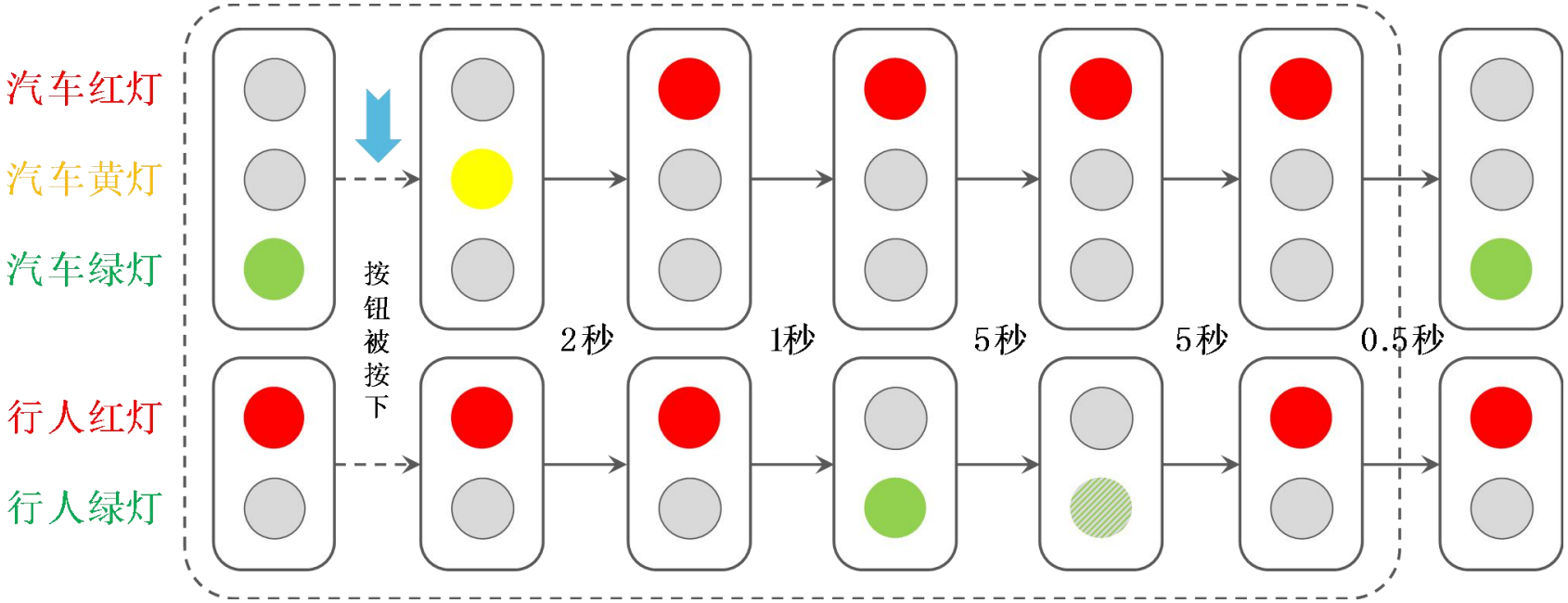


自主扩展任务 可控交通灯

硬件连接



编程思路



3 流光沙漏

自主扩展任务 可控交通灯

软件编写

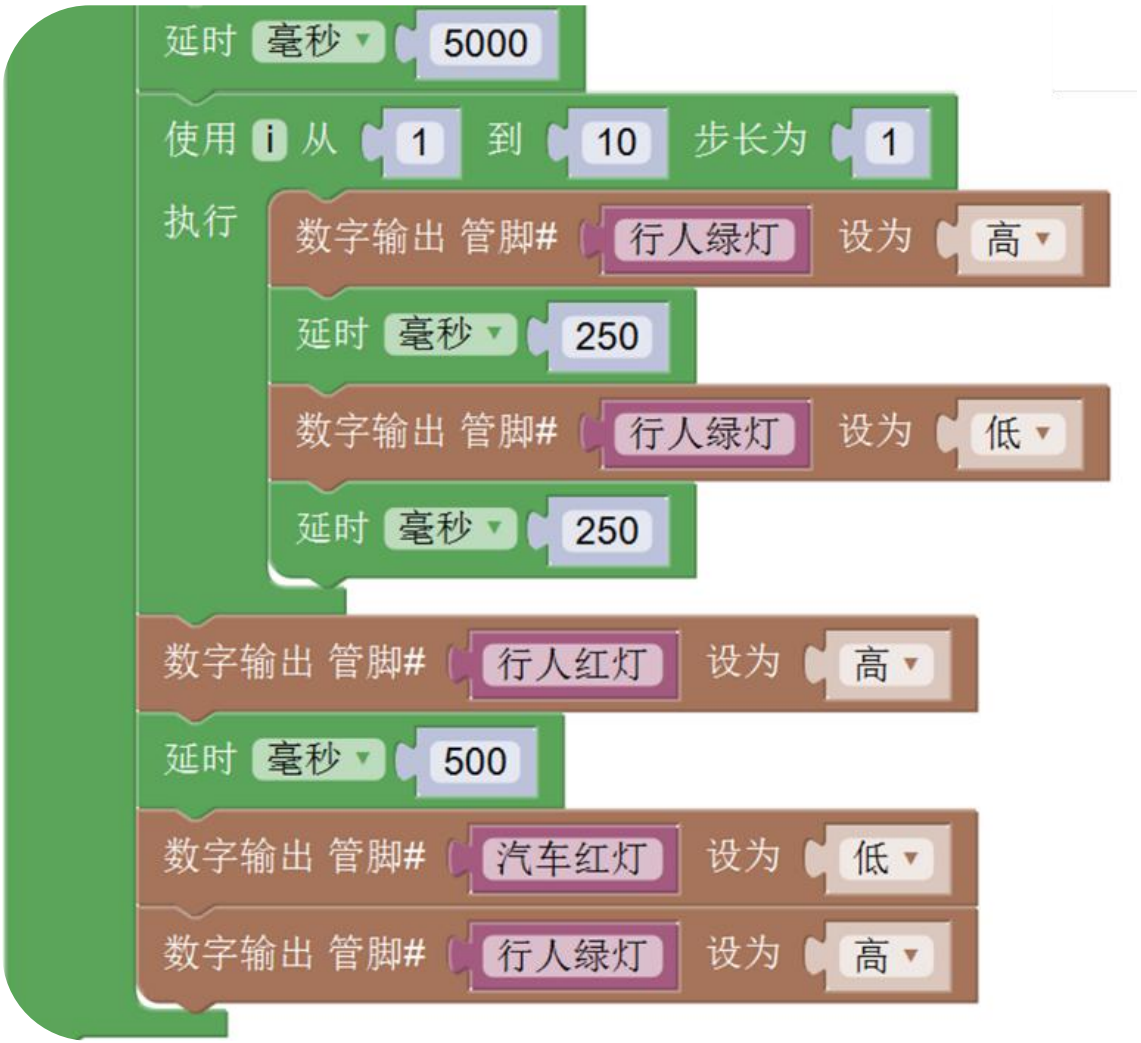
软件编写


如果
数字输入 管脚# 2
执行
数字输出 管脚# 汽车绿灯 设为 低
数字输出 管脚# 汽车黄灯 设为 高
延时 毫秒 2000
数字输出 管脚# 汽车黄灯 设为 低
数字输出 管脚# 汽车红灯 设为 高
延时 毫秒 1000
数字输出 管脚# 行人红灯 设为 低
数字输出 管脚# 行人绿灯 设为 高

3 流光沙漏

自主扩展任务 可控交通灯

软件编写



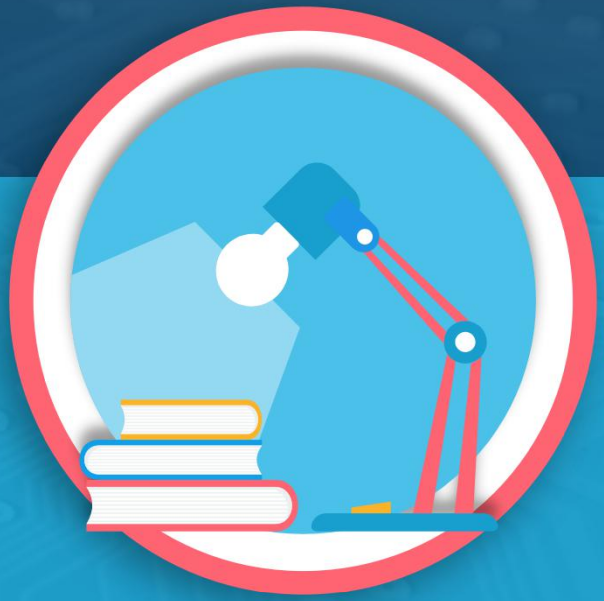
1. 请使用按钮控制灯的亮度：按下按钮时灯的亮度不断变化，松开按钮时灯将维持在此时的亮度状态。

2. 请在1的基础上加入一个按钮，用来控制灯的关闭。

第一单元 点亮创客之路

可调灯

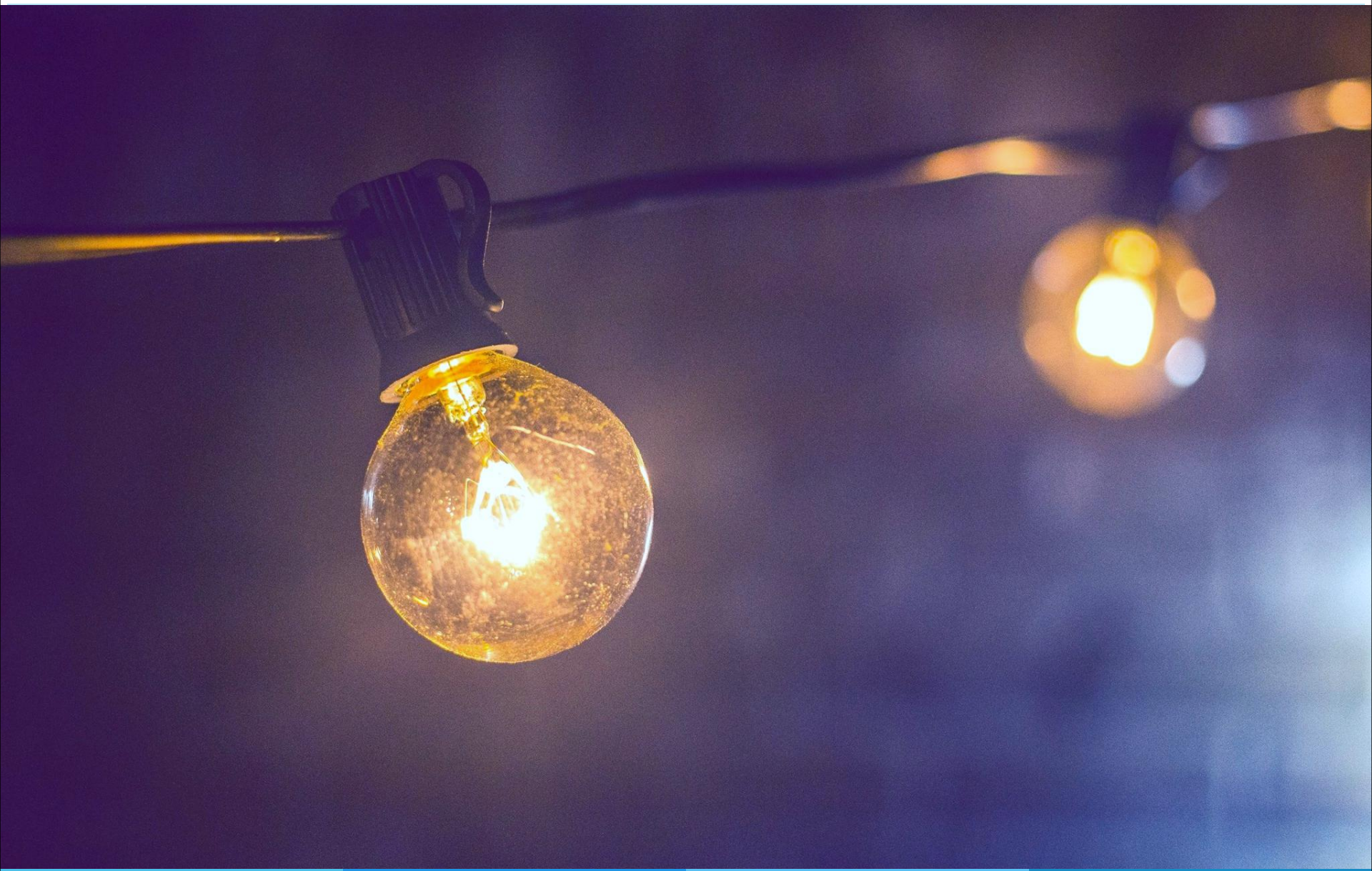
第4课



《创意电子入门》配套教学课件

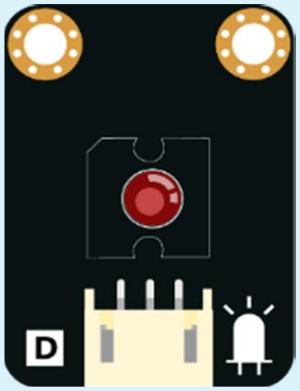
4 可调灯

情境引入

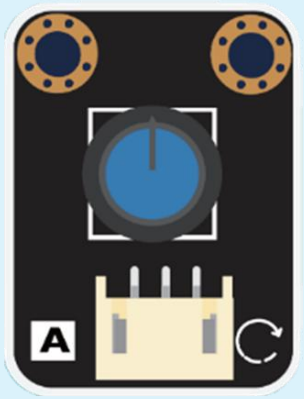


4 可调灯

模块清单



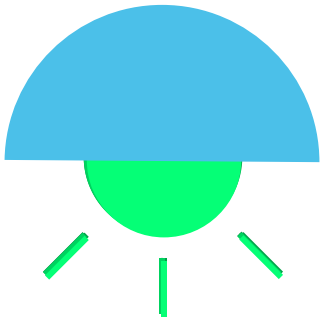
红色 LED 灯 ×1

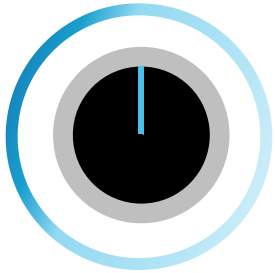


电位器 ×1

任务发布

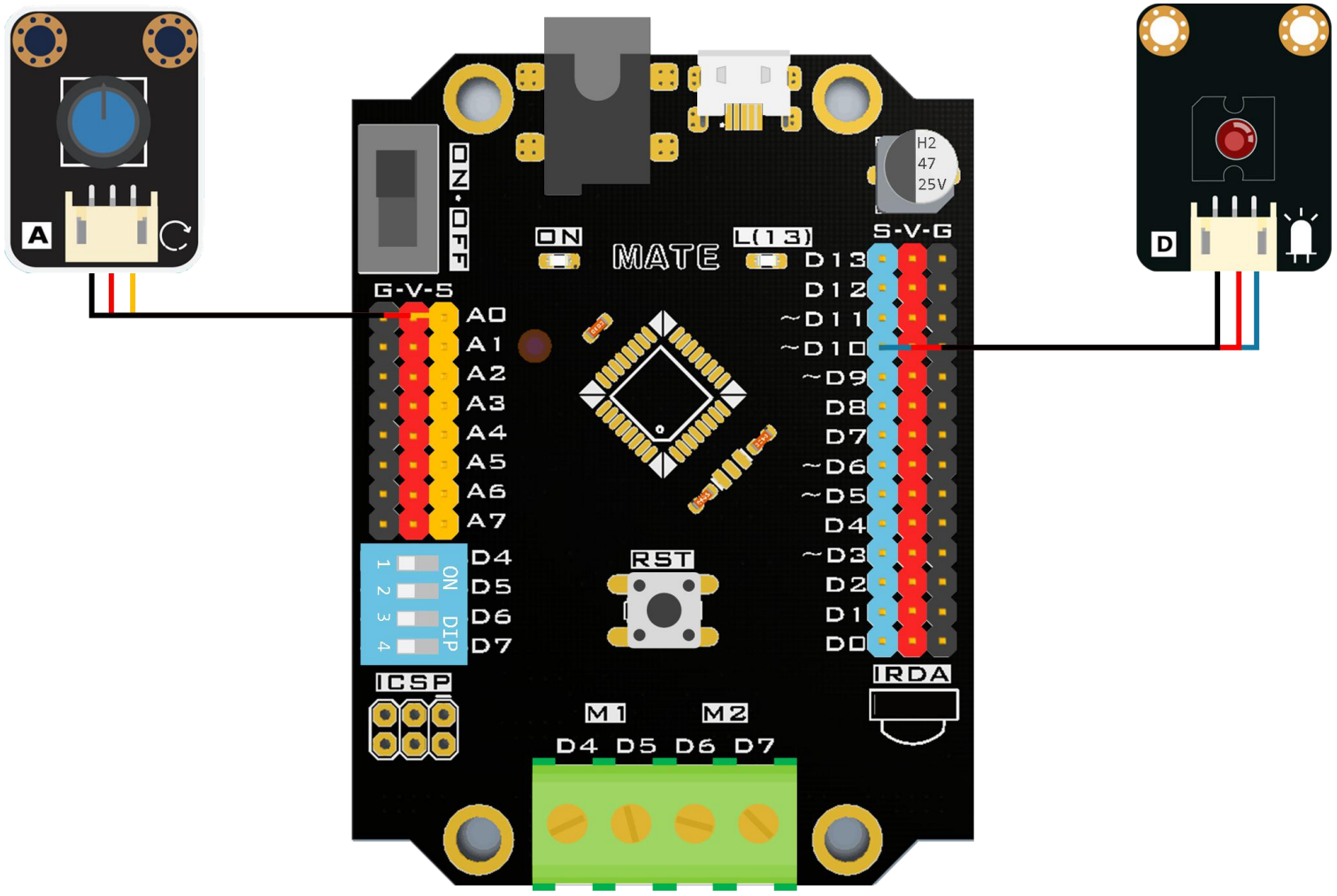
使用电位器和LED灯，制作一个可调灯，编写程序实现效果：程序上传后，通过旋转电位器的旋钮，改变LED灯的亮度。





简单任务 旋钮可调灯

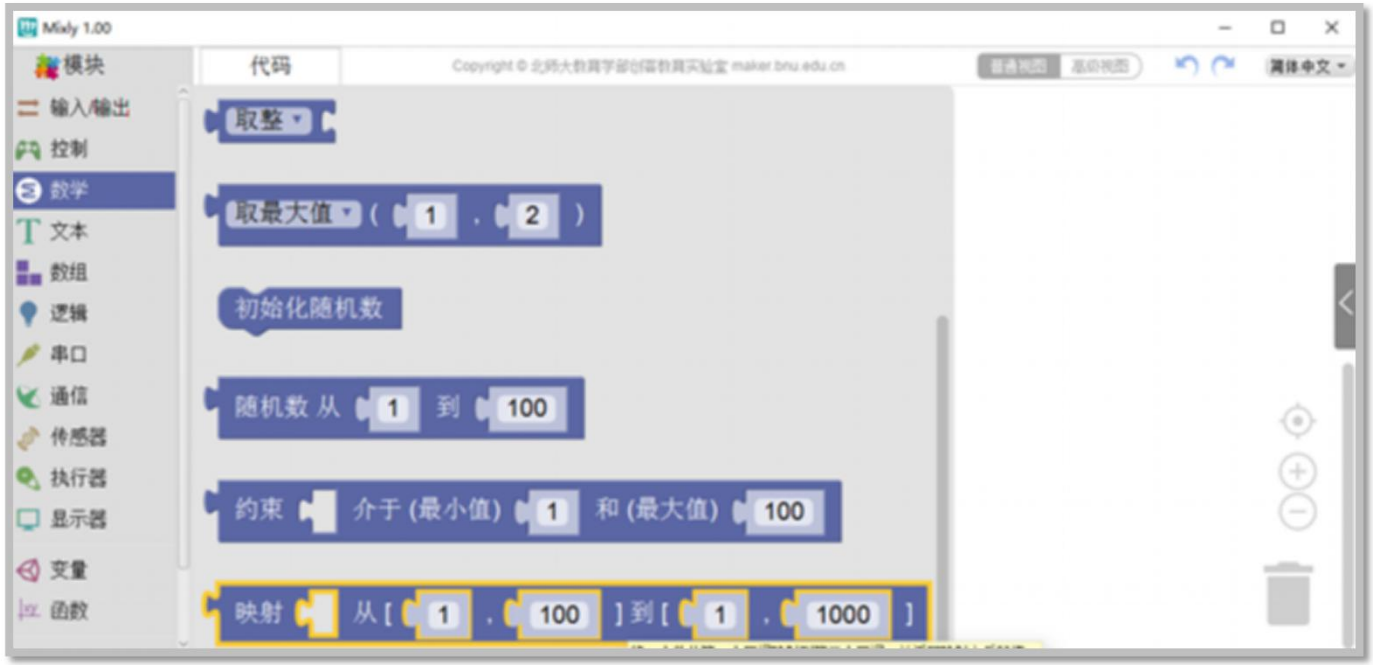
硬件连接



认识新代码块

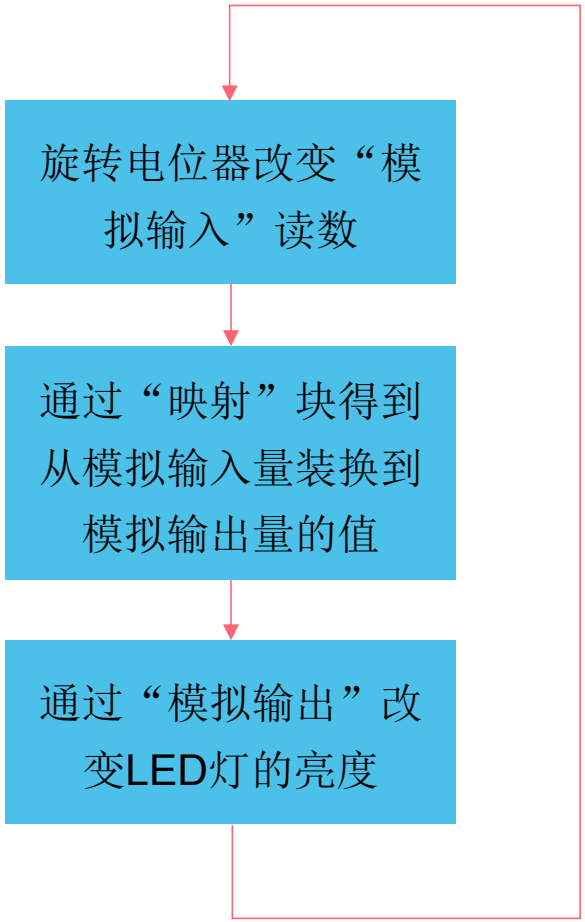
- “模拟输出”代码块位于“输入/输出”模块分类中
- “模拟输出的”管脚号在主控板上只能选择A0~A7，模拟输入的数值介于0~1023之间

认识新代码块



- “映射”代码块位于“数学”模块分类中
- 表示数字从一个范围到另一个范围之间的互相对应关系

编程思路

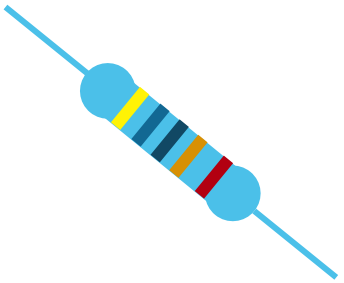

旋转电位器改变“模拟输入”读数
通过“映射”块得到从模拟输入量装换到模拟输出量的值
通过“模拟输出”改变LED灯的亮度

简单任务 旋钮可调灯

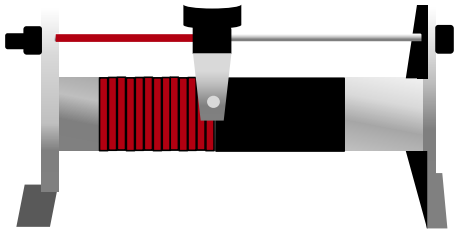
软件编写

知识点讲解 电阻

- 电阻表示导体对电流的阻碍作用，阻碍作用的大小成为电阻的阻值。
- 定值电阻一般是两个引脚，可以限制通过它的电流大小；常见的可变电阻包括滑动变阻器、电位器等。

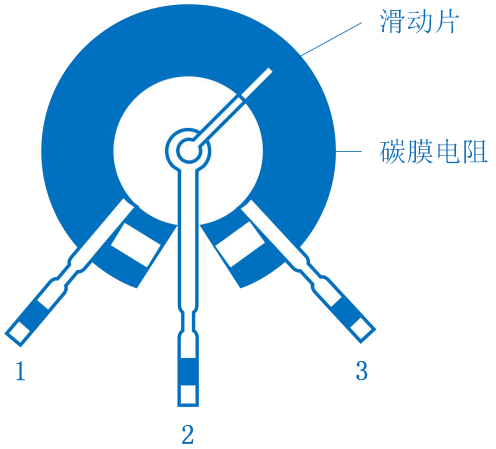


【定值电阻】

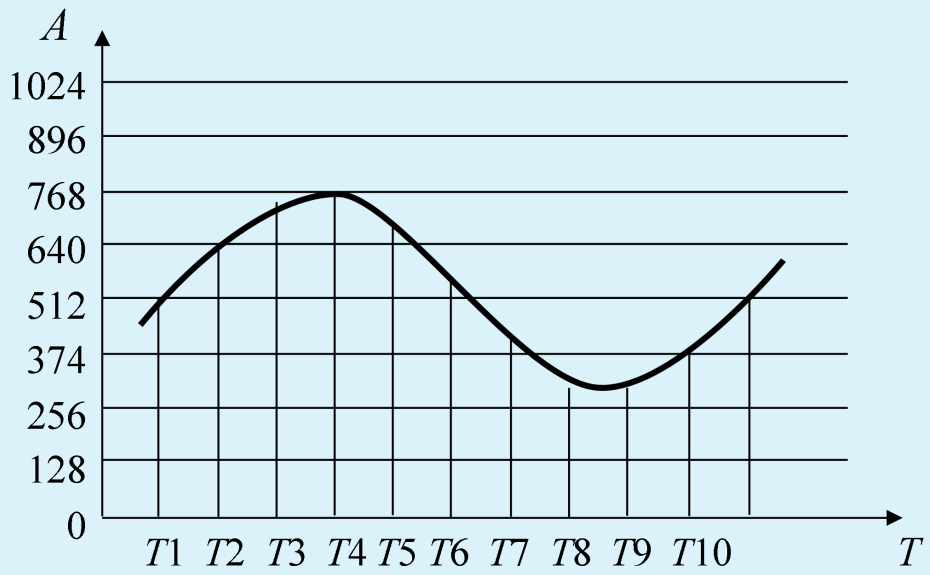


【滑动变阻器】

- 电位器是具有三个引出端、阻值可以按照某种变化规律调节的电阻元件。
- 电位器通常由碳膜电阻和可移动的滑动片组成，当滑动片沿碳膜电阻移动时，在输出端即可获得与位移量成一定关系的电阻值或电压。



- 使用传感器采集的物理量需要使用“模拟输入”功能。
- 计算机只能处理数字信号，需要进行模拟信号到数字信号的转换(A/D转换)。



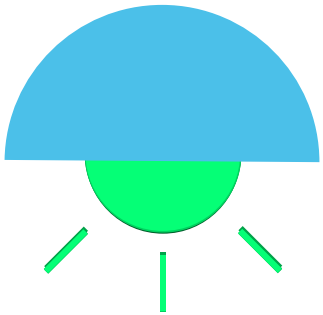
- Mixly的映射块用来建立一种数学关系

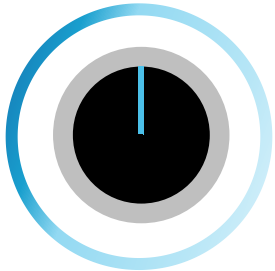
读作：“把一个从1到1000的数通过一定运算映射到一个从1到100的数”

- “映射”运算将通过简单的运算，将模拟输入管脚采集到的介于0~1023的数字转化为0~255之间的数字

任务发布

请使用电位器和LED灯，制作一个变速呼吸灯，编写程序，使程序上传后，当旋转电位器的旋钮时，LED灯的呼吸速度随之改变。

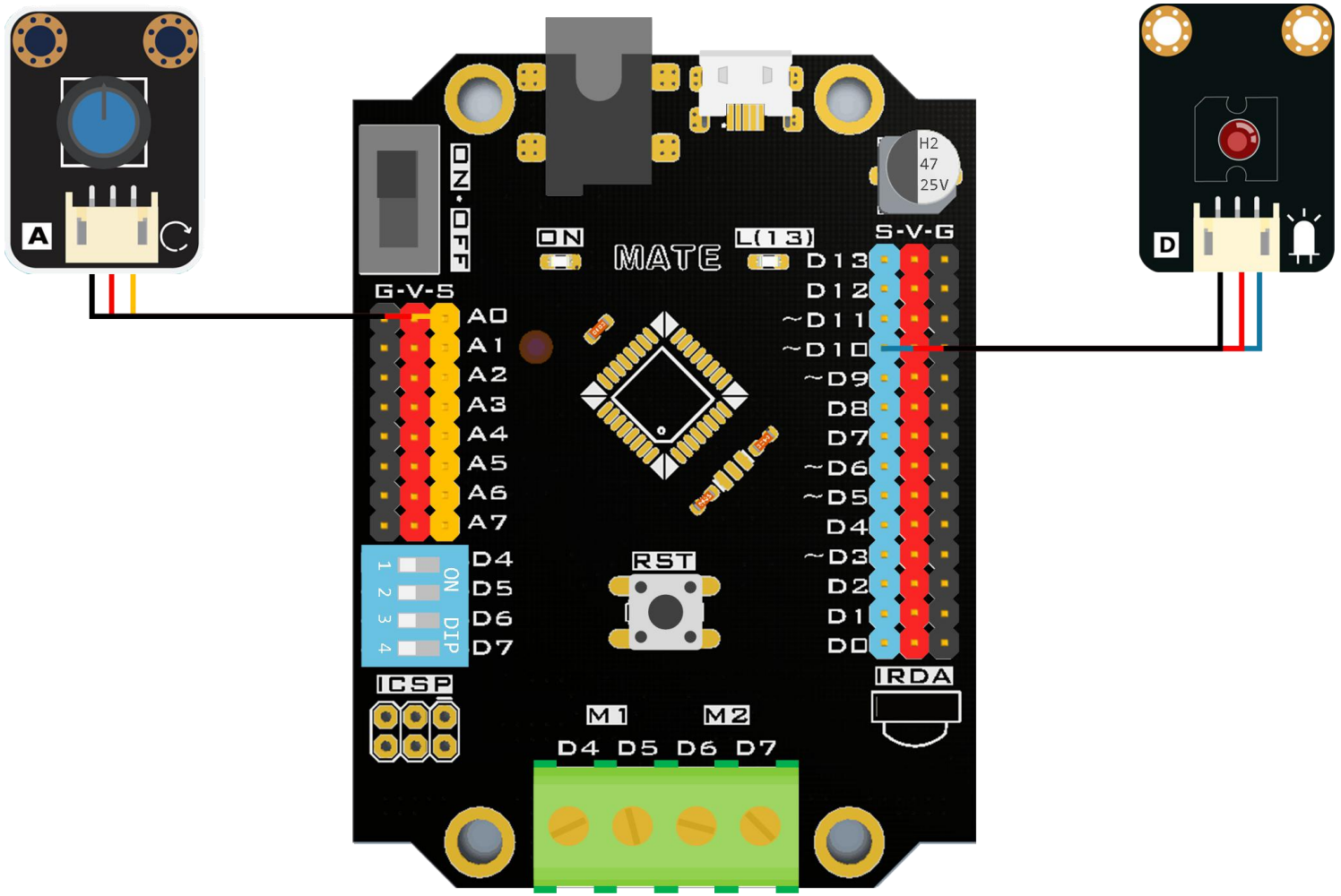




4 可调灯

可扩展任务 变速呼吸灯

硬件连接



编程思路

旋转电位器改变
“模拟输入”读数

↓

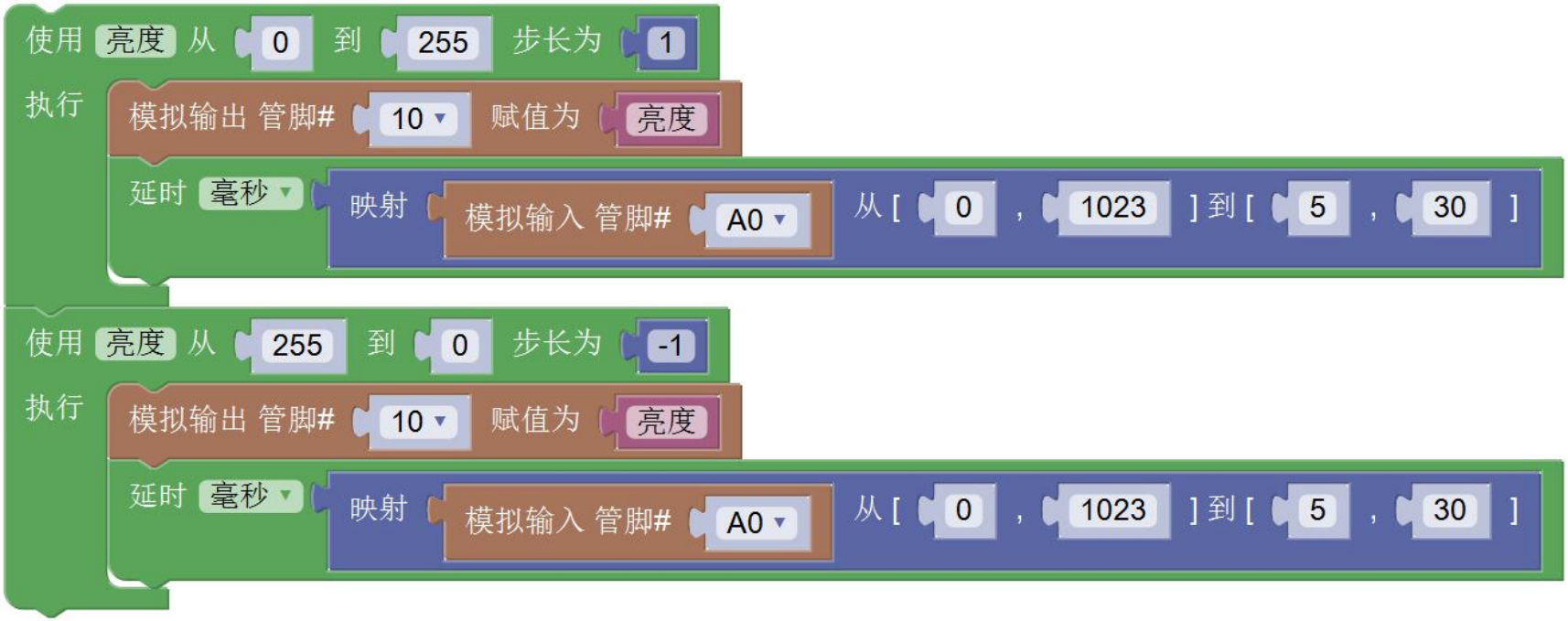
通过“映射”块
得到从模拟输入量转换到
5~30之间的“时间”值

↓

通过“时间”
改变LED灯的亮度

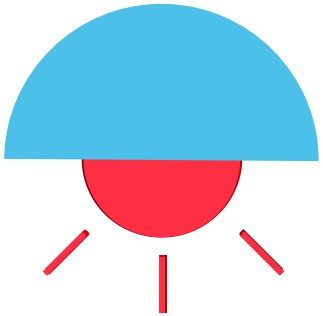
可扩展任务 变速呼吸灯

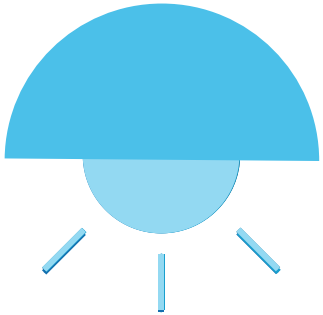
软件编写



任务发布

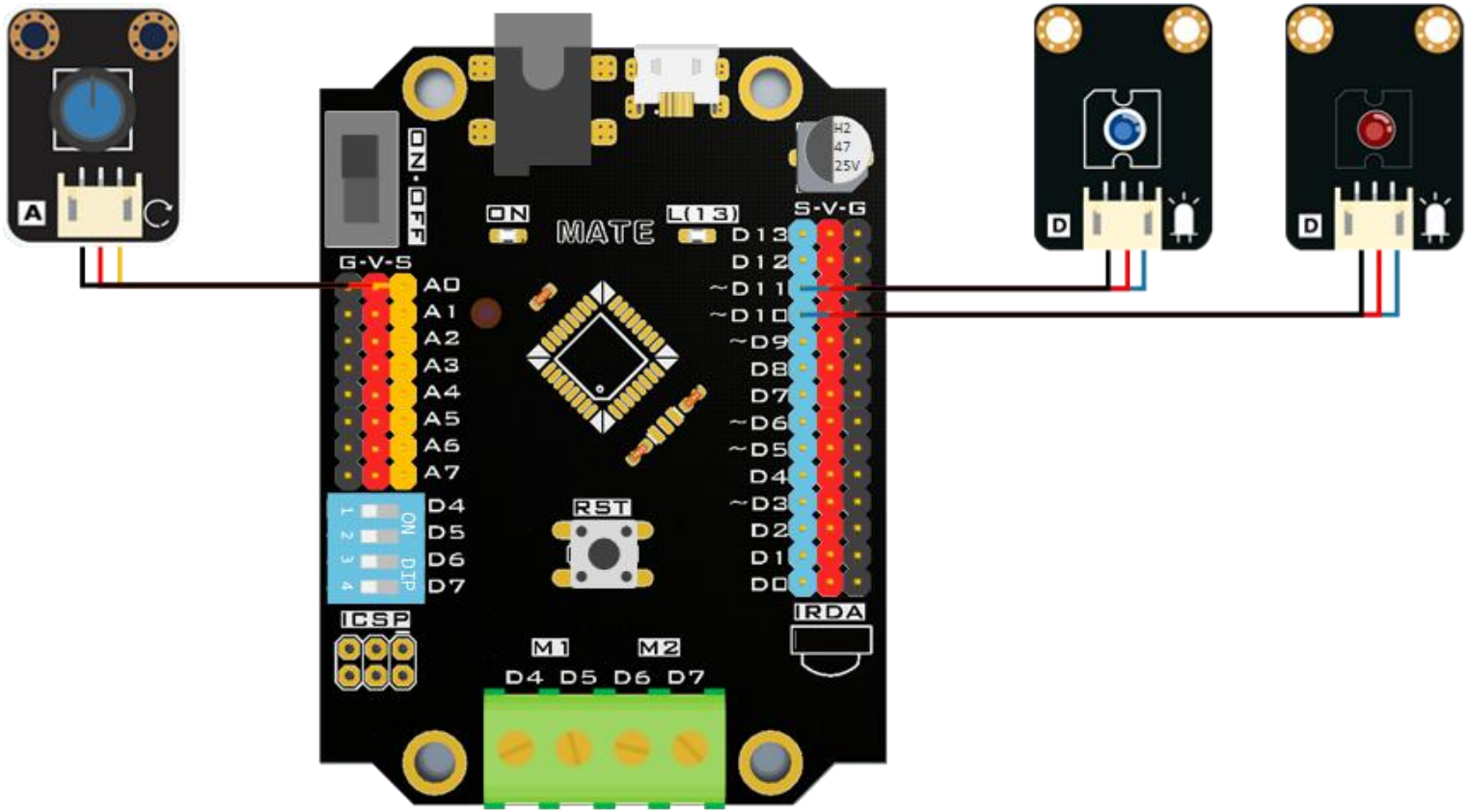
请使用电位器、红色LED灯和蓝色LED灯，编写程序，通过旋转电位器的旋钮，切换两种不同的报警效果。





自主扩展任务 可调报警灯

硬件连接



软件编写


初始化
声明 时间间隔 为 整数 并赋值 0
使用 亮度 从 0 到 255 步长为 1
执行
如果 模拟输入 管脚# A0 > 500
执行 时间间隔 赋值为 10
否则 时间间隔 赋值为 2
模拟输出 管脚# 10 赋值为 亮度
模拟输出 管脚# 11 赋值为 255 - 亮度
延时 毫秒 时间间隔

软件编写


模拟输出 管脚# 11 赋值为 255 - 亮度
延时 毫秒 时间间隔
使用 亮度 从 255 到 0 步长为 -1
执行 如果 模拟输入 管脚# A0 > 500
执行 时间间隔 赋值为 10
否则 时间间隔 赋值为 2
模拟输出 管脚# 10 赋值为 亮度
模拟输出 管脚# 11 赋值为 255 - 亮度
延时 毫秒 时间间隔

1. 判断下列说法是否正确：

a) 只有数字管脚0~13可以用于数字输出和数字输入

（ ）

b) 只有模拟管脚A0~A7可以用于模拟输入

（ ）

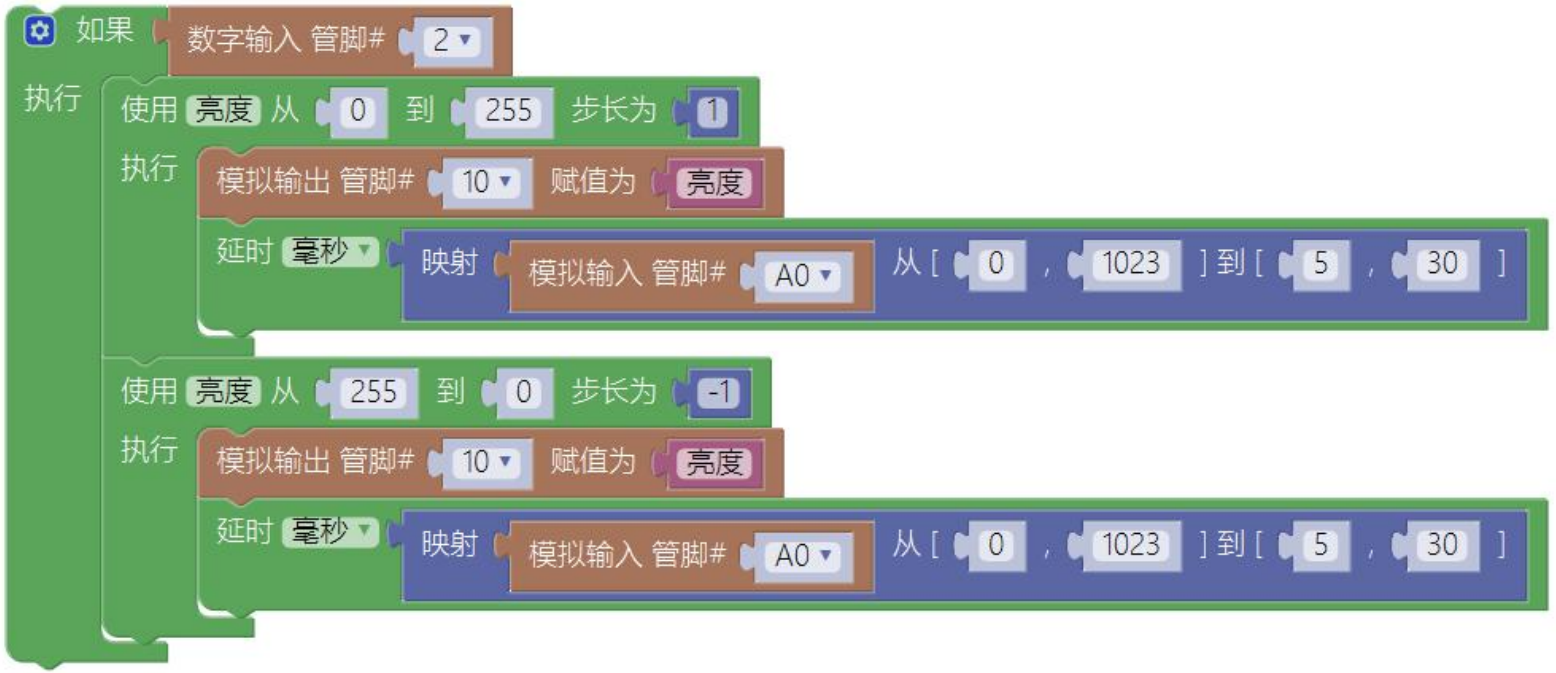
c) 模拟管脚A0~A7可以用于模拟输出

（ ）

d) 部分数字管脚可以用于模拟输出

（ ）

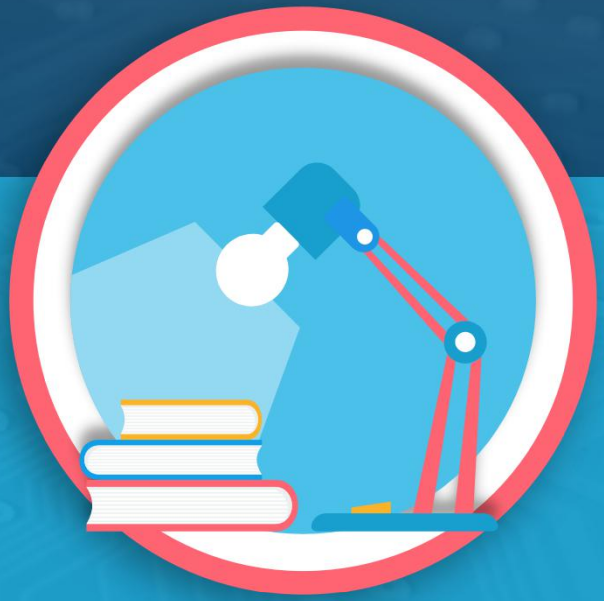
2. 在本节可扩展任务的基础上添加按钮，用来随时控制呼吸灯的开关。某同学给出了以下的程序。这个程序可以实现预期的功能吗？如果可以，请画出程序执行的流程图；如果不行，请加以改正。



第一单元 点亮创客之路

调光台灯

第5课



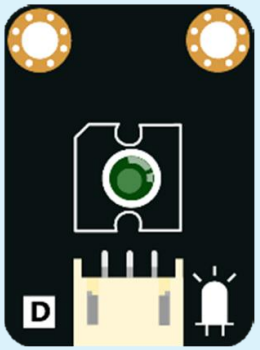
《创意电子入门》配套教学课件

5 调光台灯

情境引入

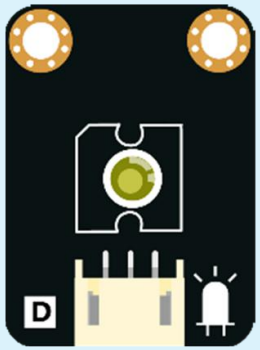
5 调光台灯

模块清单



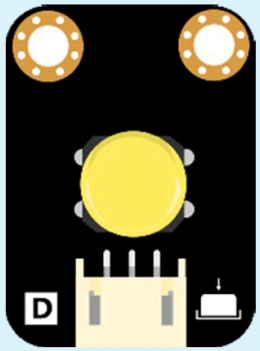
绿色 LED 灯

×1



黄色 LED 灯

×1

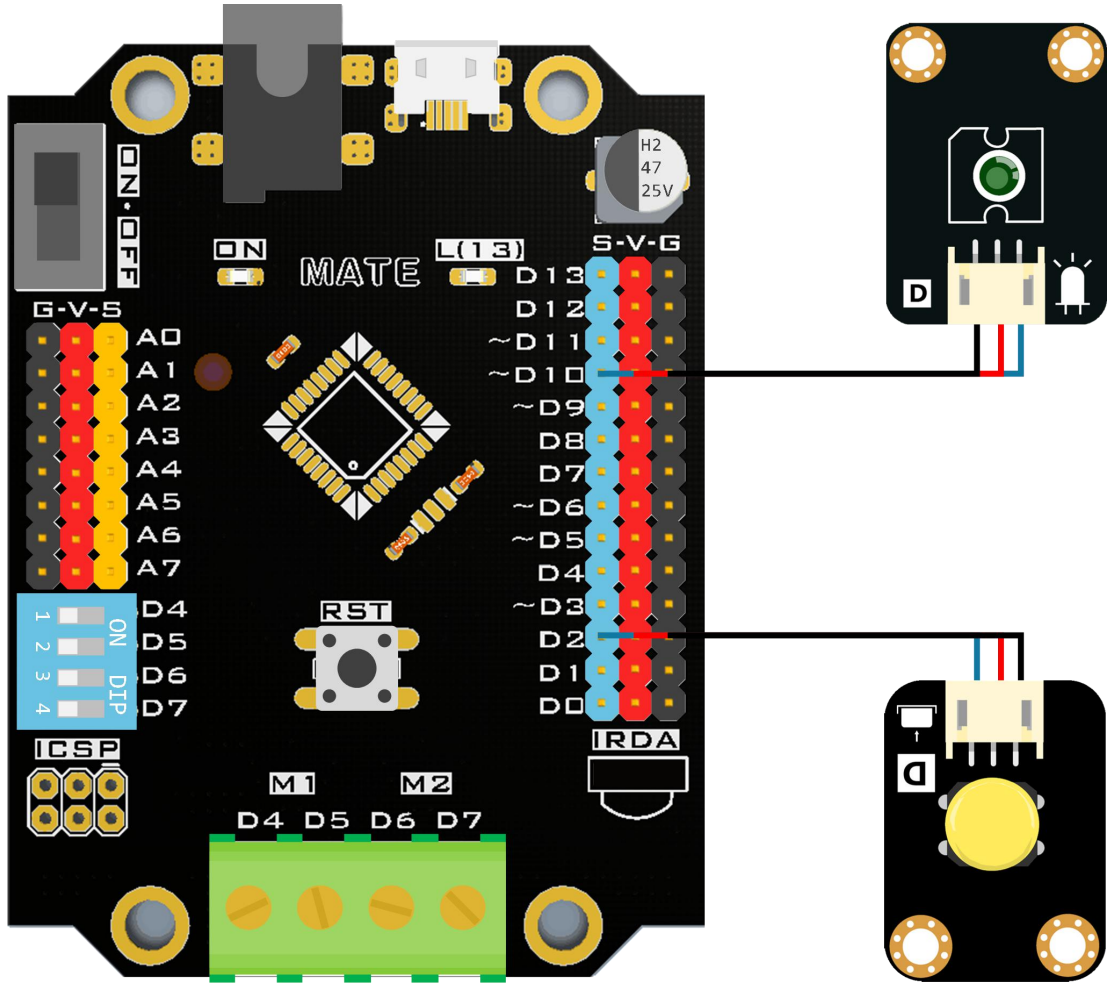


黄色按钮×1

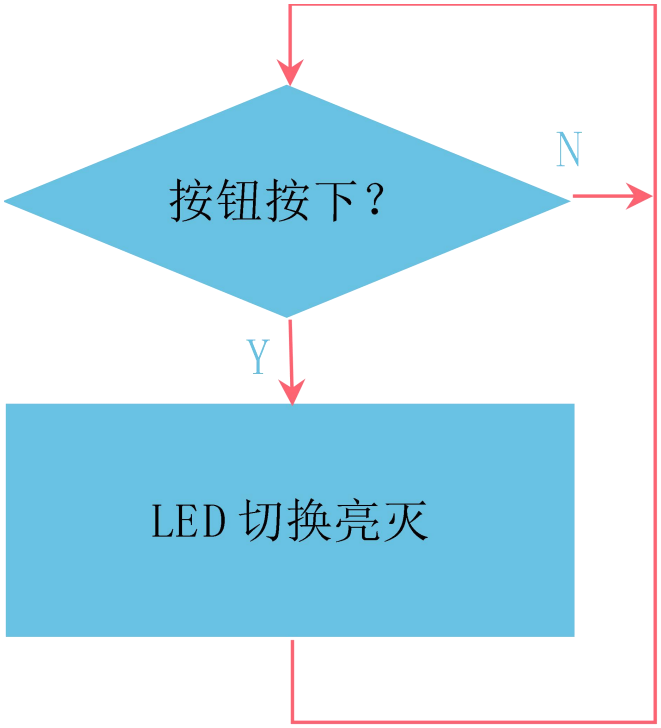
任务发布

请使用一个LED灯和一个按钮实现以下效果：每按一次按钮，LED灯切换亮灭。

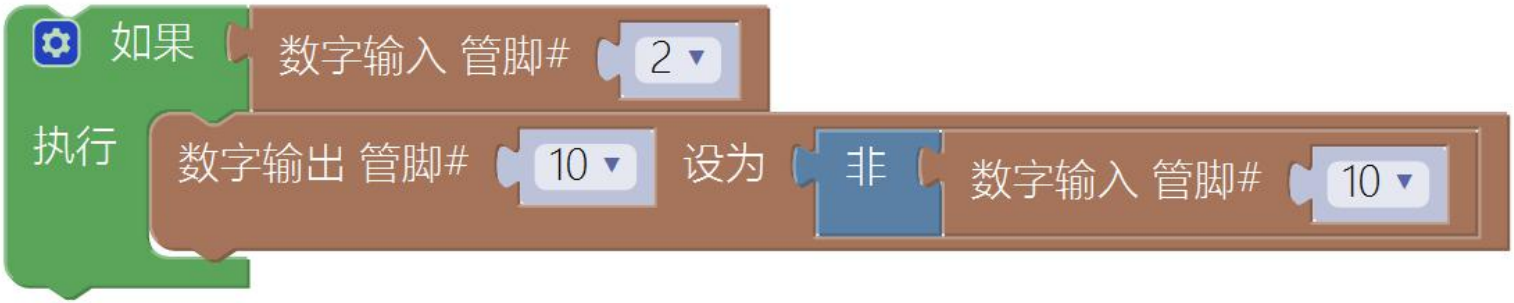
硬件连接


ON•OFF
ON
MATE
L(13)
S-V-G
G-V-5
A0
A1
A2
A3
A4
A5
A6
A7
D4
D5
D6
D7
RST
ICSP
M1
M2
D4 D5 D6 D7
D13
D12
~D11
~D10
~D9
D8
D7
~D6
~D5
D4
~D3
D2
D1
D0
IRDA
H2
47
25V
D

编程思路



软件编写



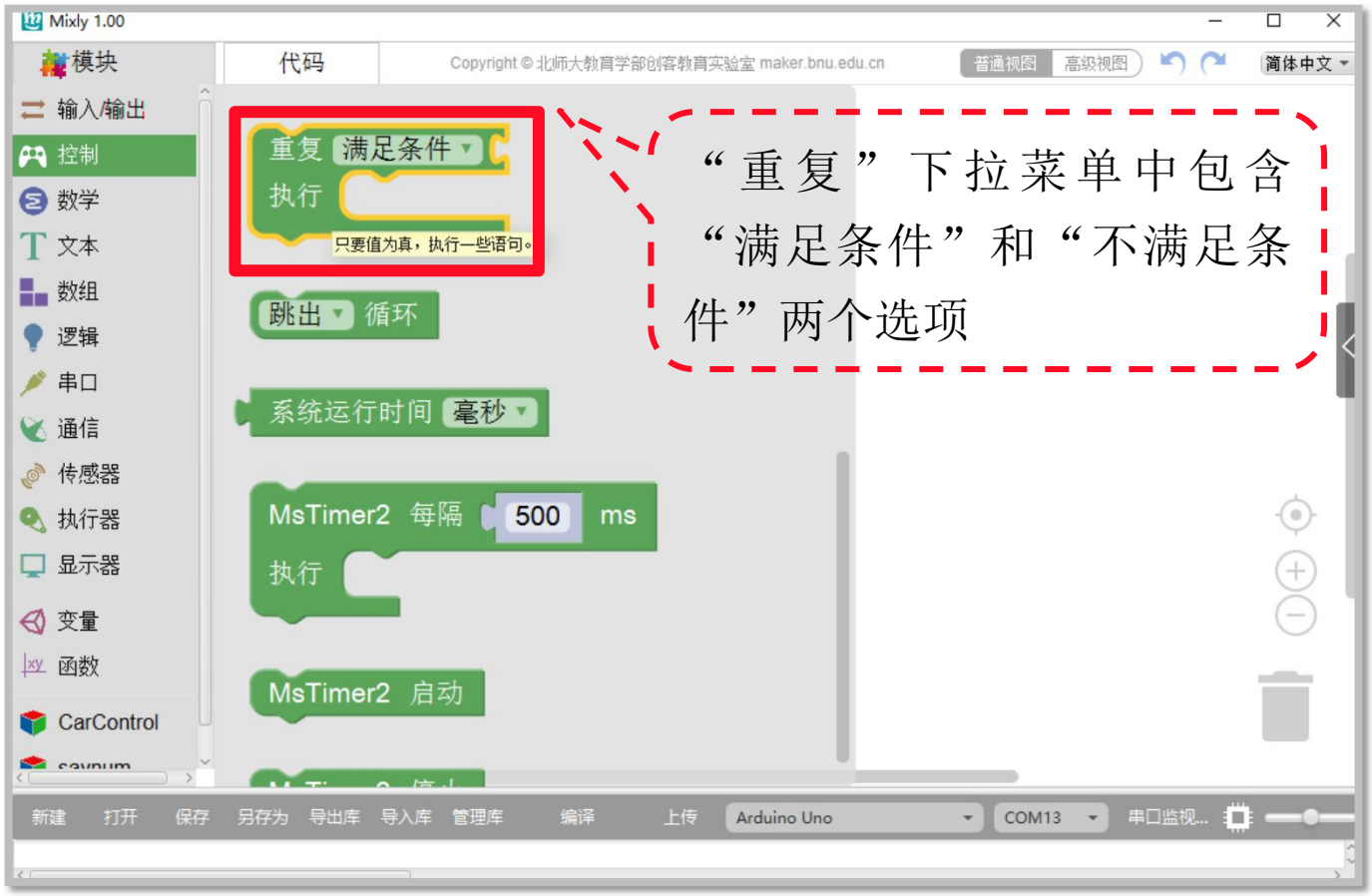
代码排错

上述代码运行后存在什么问题？

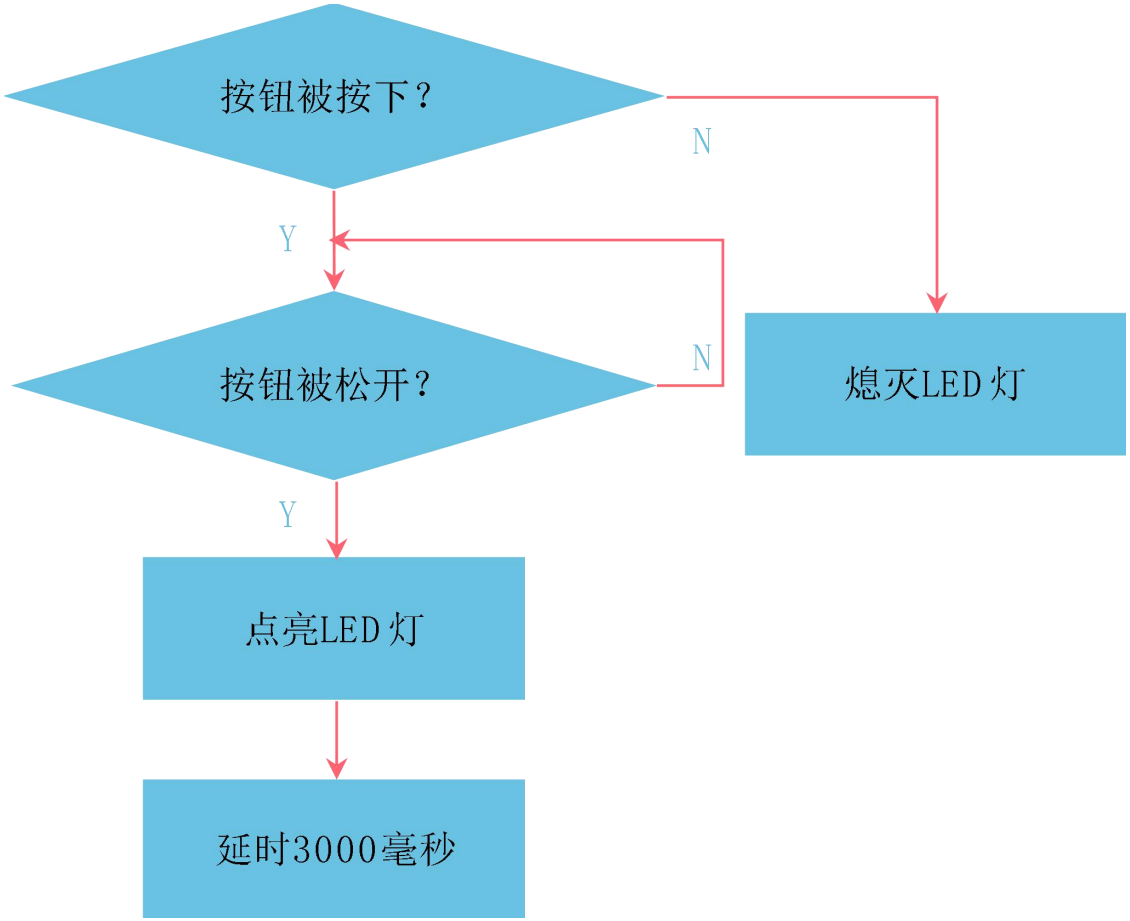
认识新代码块

- “非”代码块位于Mixly软件的“逻辑”模块下，其作用是取相反的逻辑

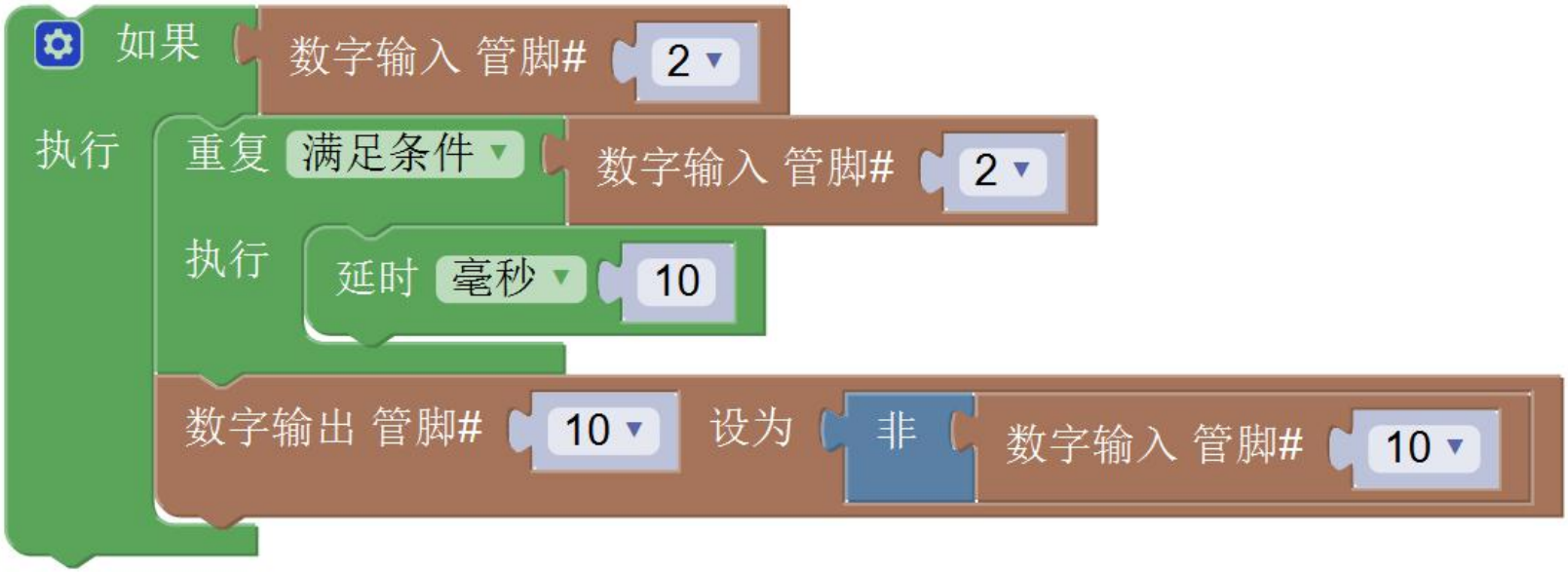
认识新代码块


Mixly 1.00
模块
代码
Copyright © 北师大教育学部创客教育实验室 maker.bnu.edu.cn
普通视图
高级视图
简体中文
输入/输出
控制
数学
文本
数组
逻辑
串口
通信
传感器
执行器
显示器
变量
函数
CarControl
重复 满足条件
执行
只要值为真，执行一些语句。
跳出 循环
系统运行时间 毫秒
MsTimer2 每隔 500 ms
执行
MsTimer2 启动
“重复”下拉菜单中包含“满足条件”和“不满足条件”两个选项
新建
打开
保存
另存为
导出库
导入库
管理库
编译
上传
Arduino Uno
COM13
串口监视...

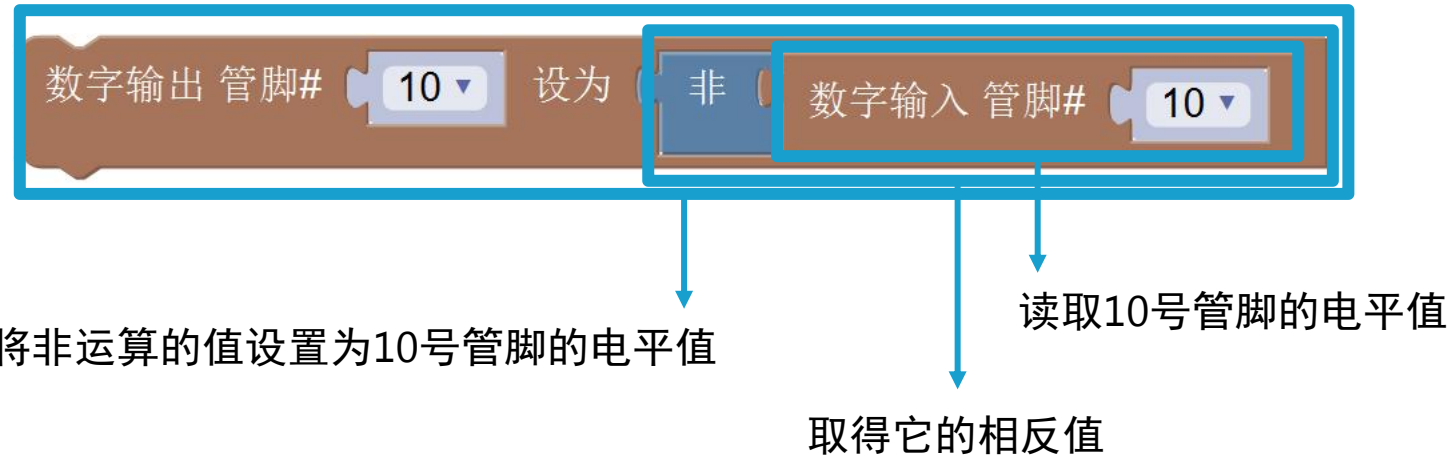
编程思路

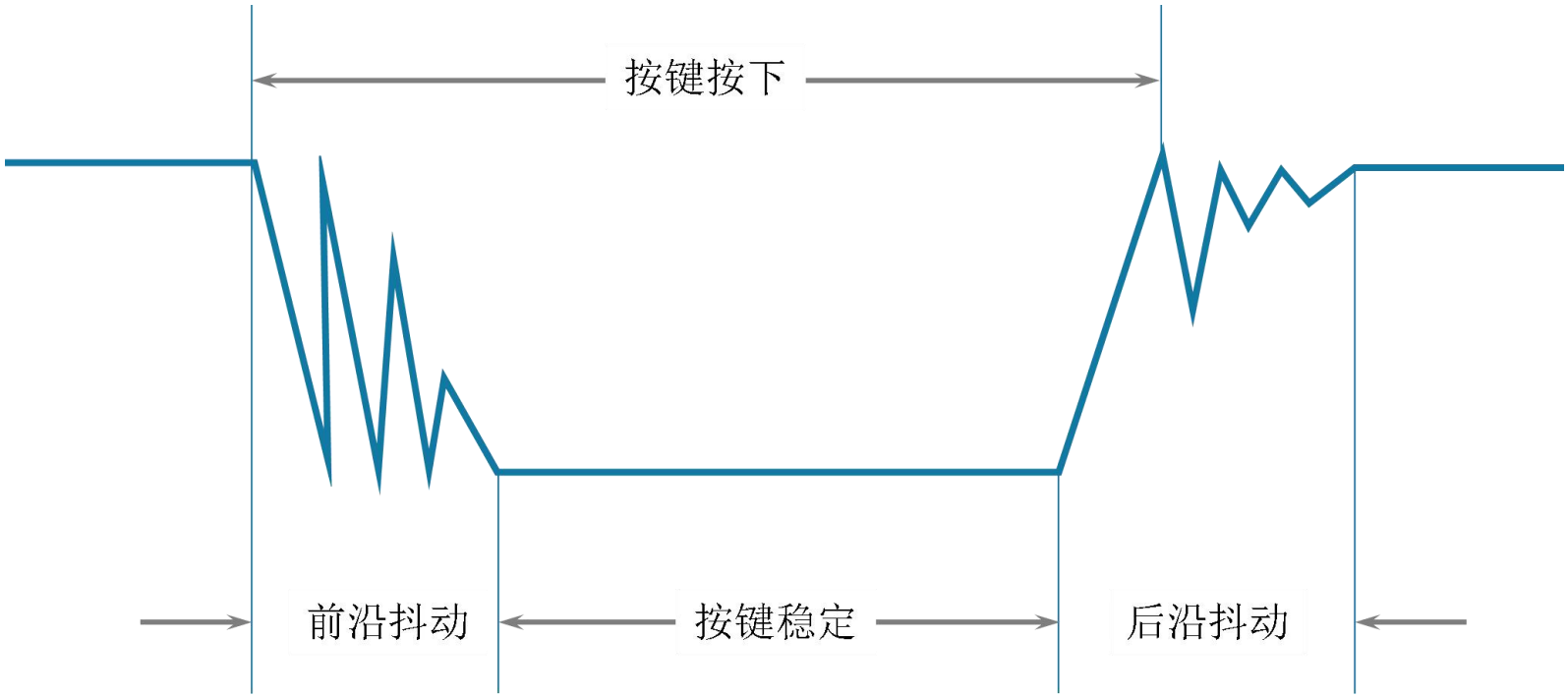


程序改写


如果
数字输入 管脚# 2
执行
重复 满足条件
数字输入 管脚# 2
执行
延时 毫秒 10
数字输出 管脚# 10 设为 非
数字输入 管脚# 10

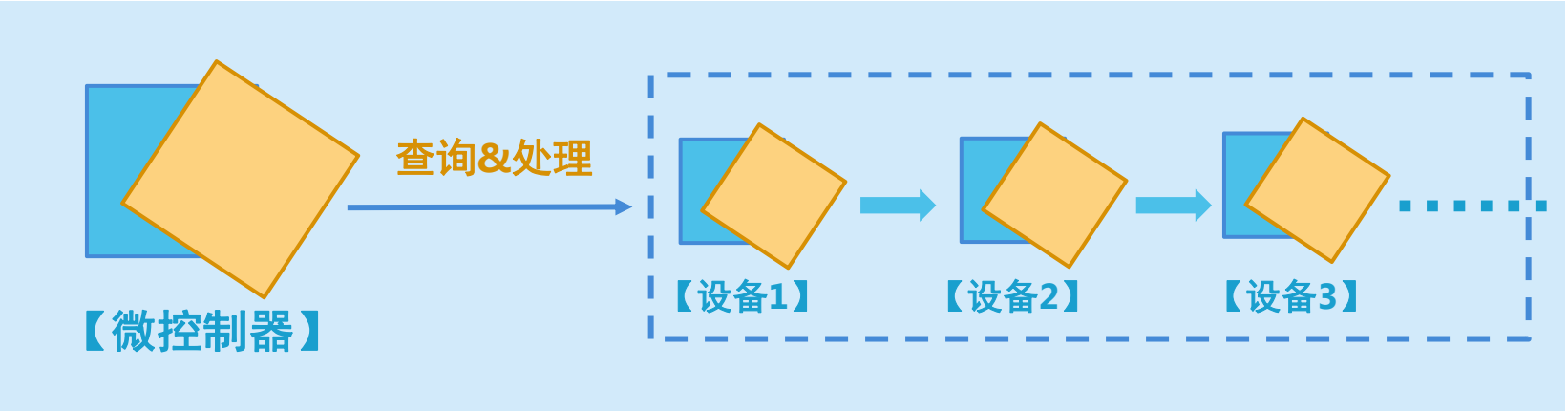
- Arduino的所有管脚都既可以输出也可以输入。可同时指定其为输入模式和输出模式。
- 由于程序中的赋值语句从右向左执行





- 检测到按键输入为1之后，延时5ms~10ms，再次检测，如果按键还为1，那么就认为有按键输入。

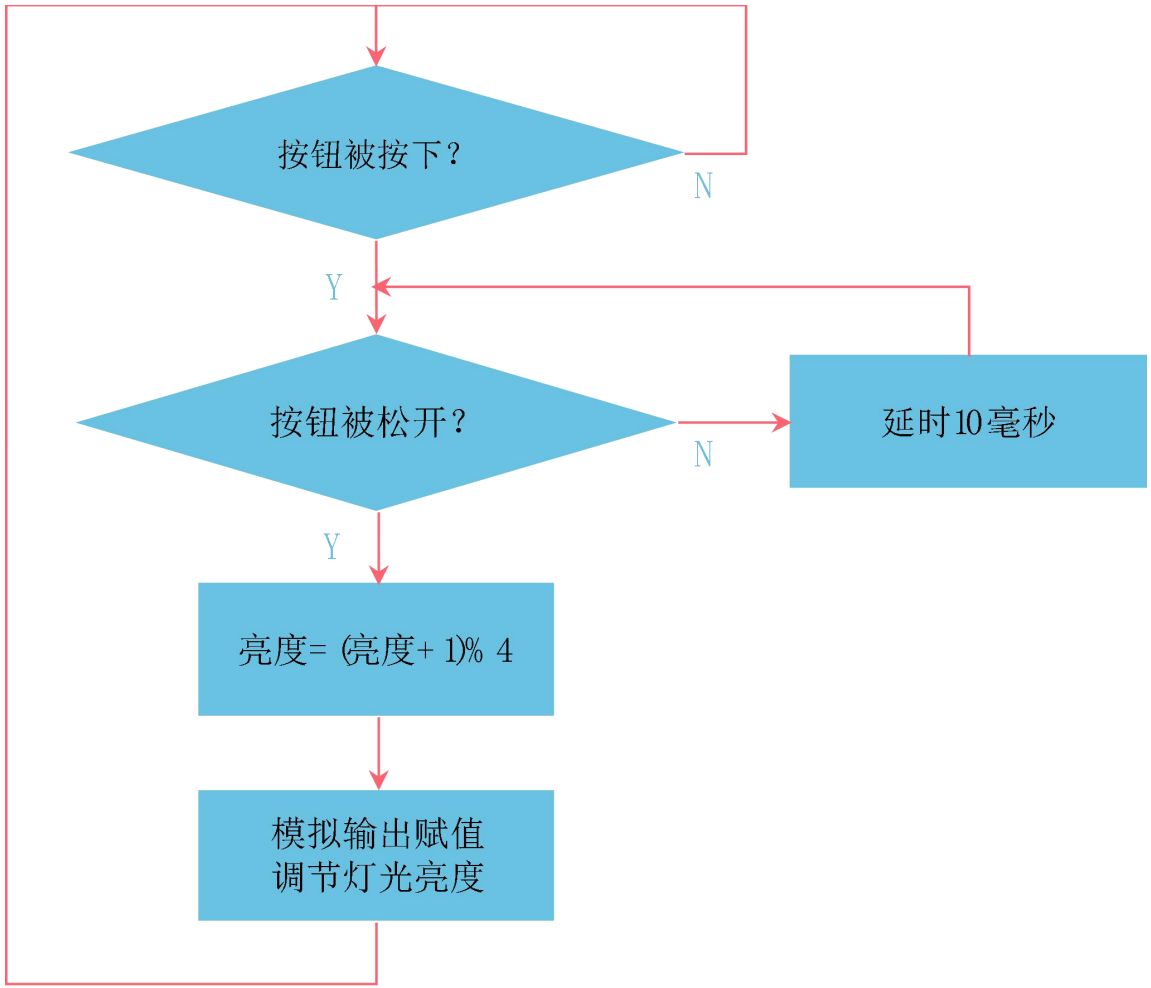
- 轮询I/O方式或程序控制I/O方式，是让微控制器以一定的周期按次序查询每一个传感器，看它是否有数据输入或输出的要求。若有，则进行相应的输入/输出服务；若无，或I/O处理完毕，微控制器就接着查询下一个设备。



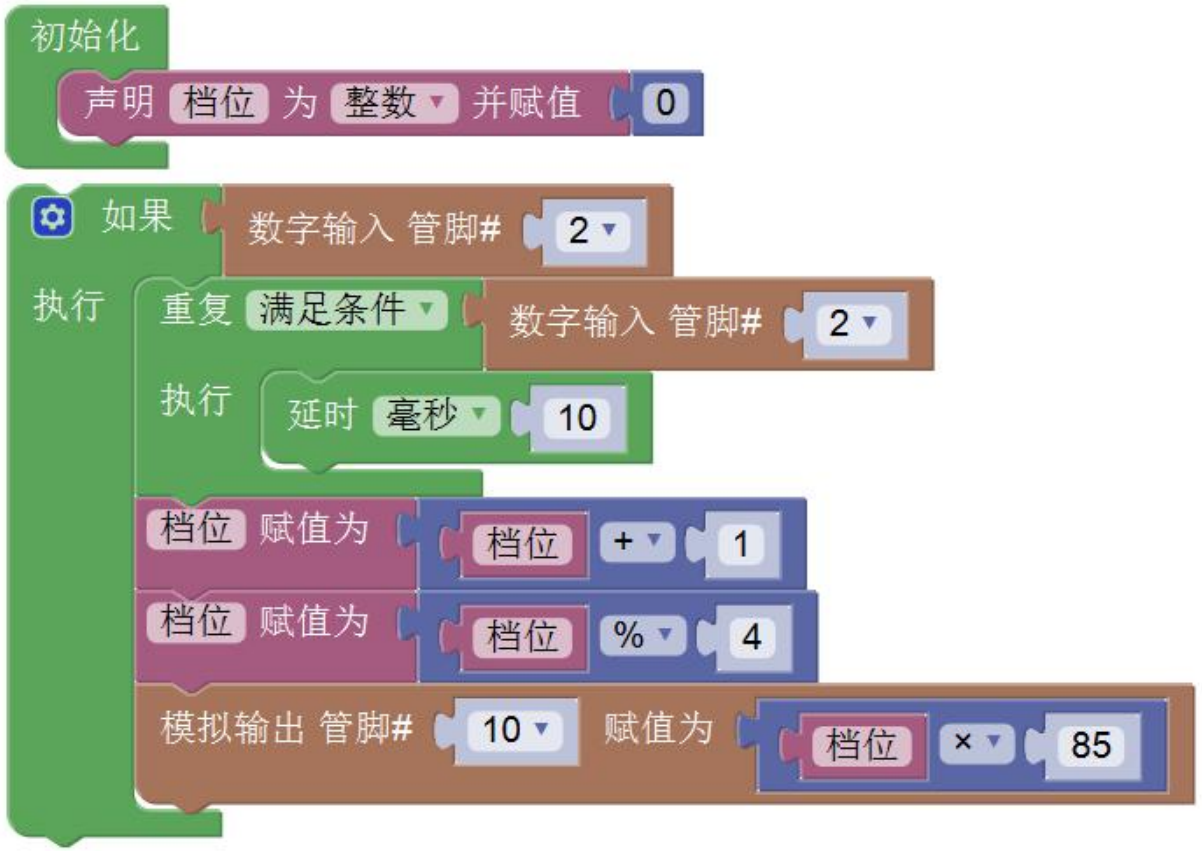
任务发布

请制作一个使用按钮控制的四档调光台灯，每按下一次按键，灯的亮度相应发生变化。

编程思路



软件编写

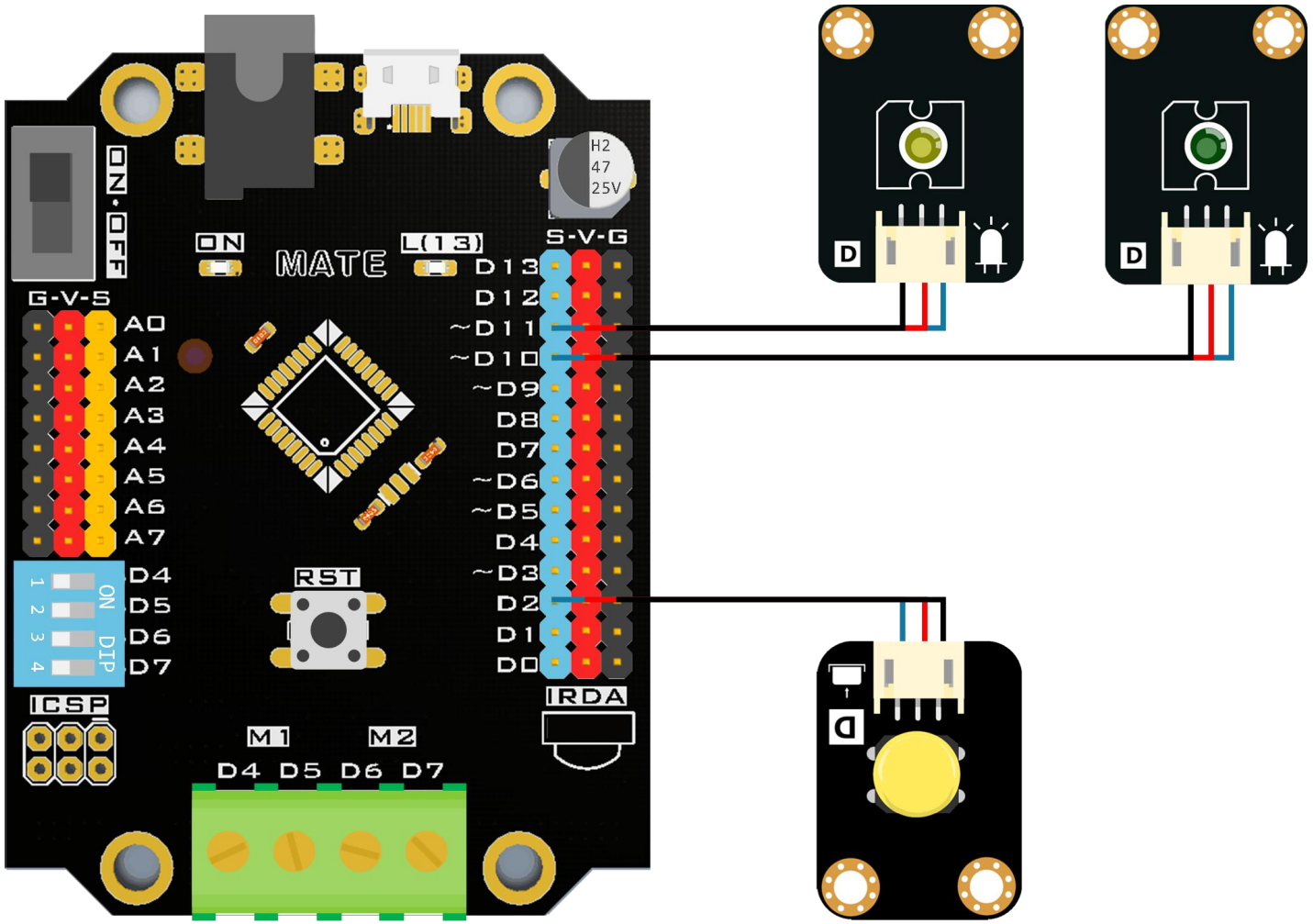

初始化
声明 档位 为 整数 并赋值 0
如果
数字输入 管脚# 2
执行
重复 满足条件
数字输入 管脚# 2
执行
延时 毫秒 10
档位 赋值为 档位 + 1
档位 赋值为 档位 % 4
模拟输出 管脚# 10 赋值为 档位 × 85

任务发布

使用一个按钮和两个LED灯来模拟双闪灯的控制。

自主扩展任务 双闪灯

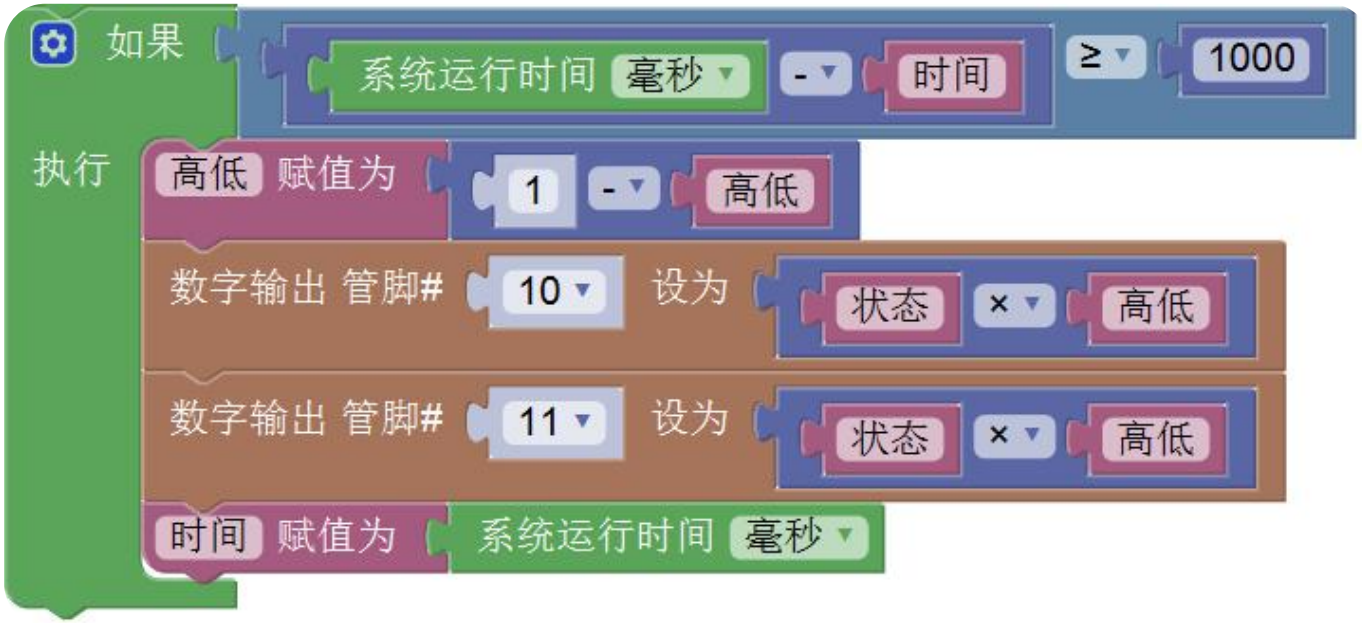
硬件连接



软件编写


初始化
声明 状态 为 整数 并赋值 0
声明 高低 为 整数 并赋值 0
声明 时间 为 整数 并赋值 0
如果 数字输入 管脚# 2
执行
重复 满足条件 数字输入 管脚# 2
执行
延时 毫秒 10
状态 赋值为 1 - 状态

软件编写


如果
系统运行时间 毫秒 - 时间 ≥ 1000
执行
高低 赋值为 1 - 高低
数字输出 管脚# 10 设为 状态 × 高低
数字输出 管脚# 11 设为 状态 × 高低
时间 赋值为 系统运行时间 毫秒

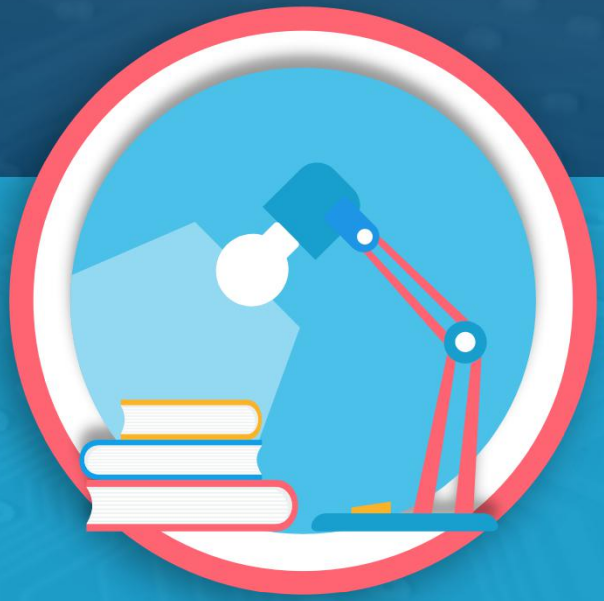
1. 请尝试在第2课延时灯的基础上添加功能：延时的3秒中内再按一次按钮，LED灯立刻熄灭。

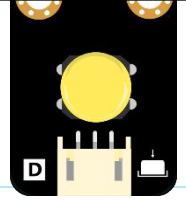
2. 请使用两个按钮和两个LED灯制作一个抢答器。当其中一个按钮按下时对应的LED灯点亮，此时另一个按钮按下时将不起作用。提示：每次抢答后可以通过复位键进行初始化。

第一单元 点亮创客之路

反应测试

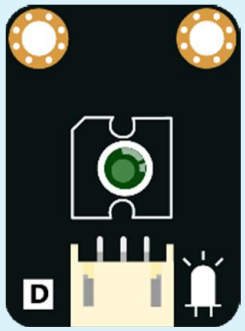
单元任务





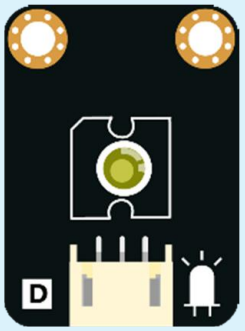
反应测试

模块清单



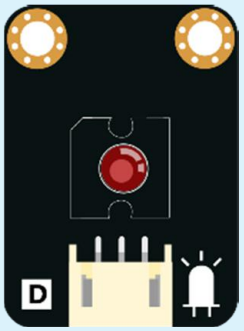
绿色 LED 灯

×1



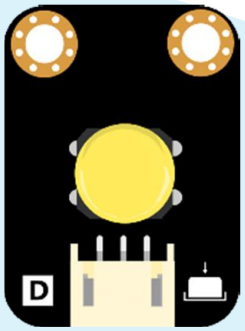
黄色 LED 灯

×1



红色 LED 灯

×1

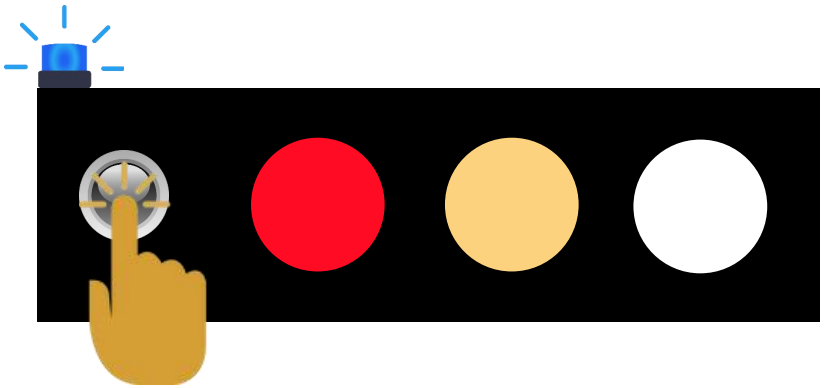


黄色按钮

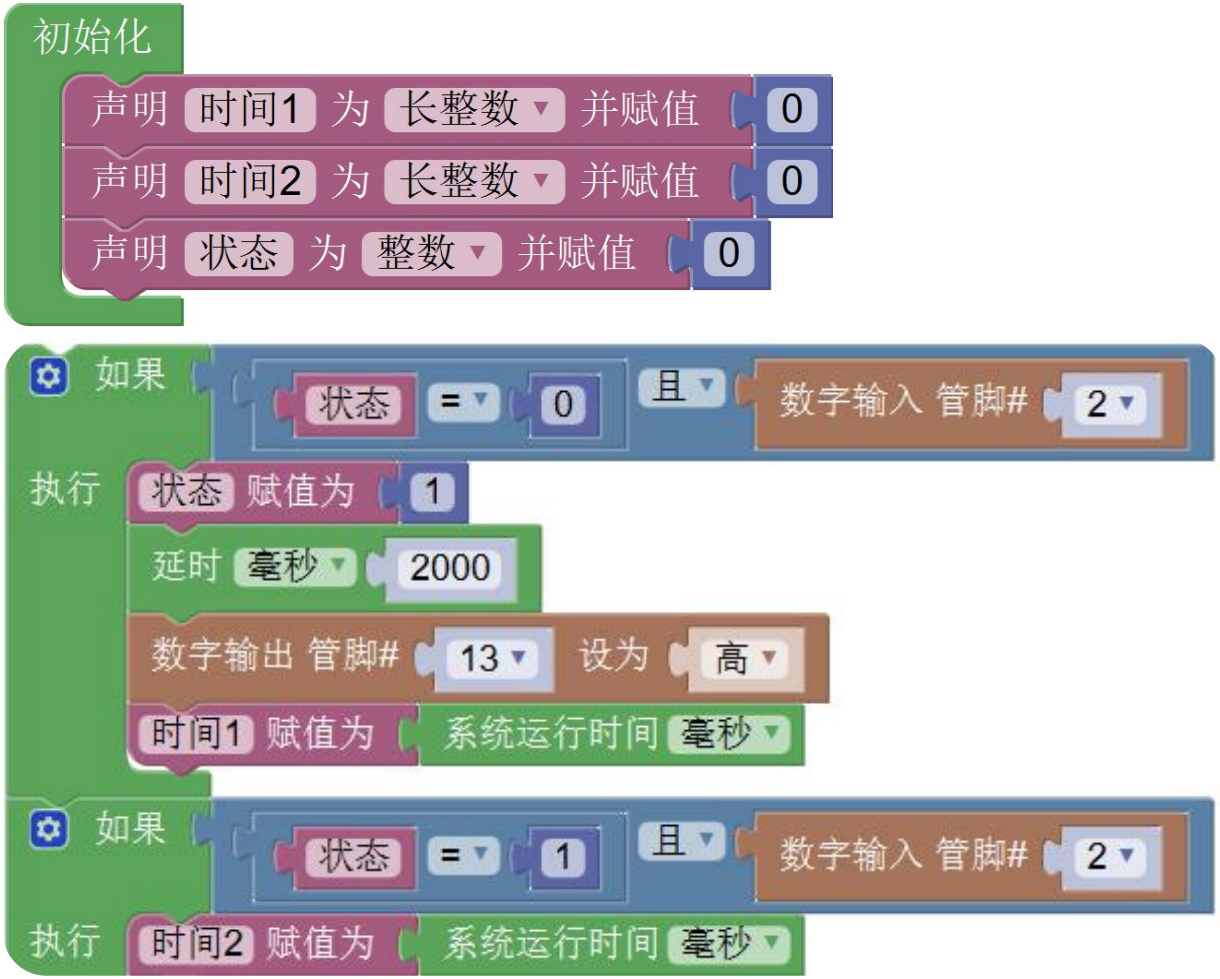
×1

任务发布

请同学们使用一个按钮和三个LED灯完成反应测试游戏的设计：当板载灯亮时，立即按下按钮，按钮被按下的时间与灯亮的时间间隔越短，闪烁的LED灯就越多。



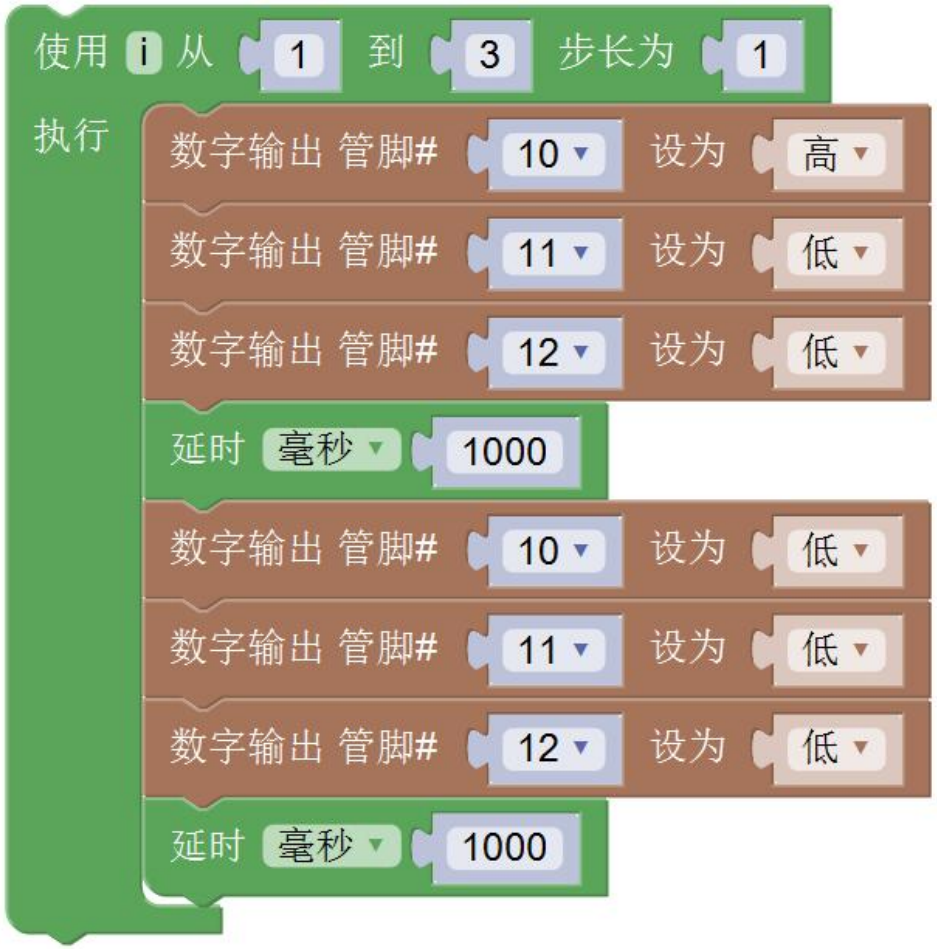
软件编写


初始化
声明 时间1 为 长整数 并赋值 0
声明 时间2 为 长整数 并赋值 0
声明 状态 为 整数 并赋值 0
如果
状态 = 0 且 数字输入 管脚# 2
执行
状态 赋值为 1
延时 毫秒 2000
数字输出 管脚# 13 设为 高
时间1 赋值为 系统运行时间 毫秒
如果
状态 = 1 且 数字输入 管脚# 2
执行
时间2 赋值为 系统运行时间 毫秒

软件编写


如果
状态 = 1
且
数字输入 管脚# 2
执行
时间2 赋值为 系统运行时间 毫秒
数字输出 管脚# 13 设为 低
状态 赋值为 0
如果
时间2 - 时间1 < 100
执行
使用 i 从 1 到 3 步长为 1 执行 数字输出 ...
否则如果
时间2 - 时间1 < 300
执行
使用 i 从 1 到 3 步长为 1 执行 数字输出 ...
否则如果
时间2 - 时间1 < 500
执行
使用 i 从 1 到 3 步长为 1 执行 数字输出 ...

软件编写


使用 i 从 1 到 3 步长为 1
执行
数字输出 管脚# 10 设为 高
数字输出 管脚# 11 设为 低
数字输出 管脚# 12 设为 低
延时 毫秒 1000
数字输出 管脚# 10 设为 低
数字输出 管脚# 11 设为 低
数字输出 管脚# 12 设为 低
延时 毫秒 1000