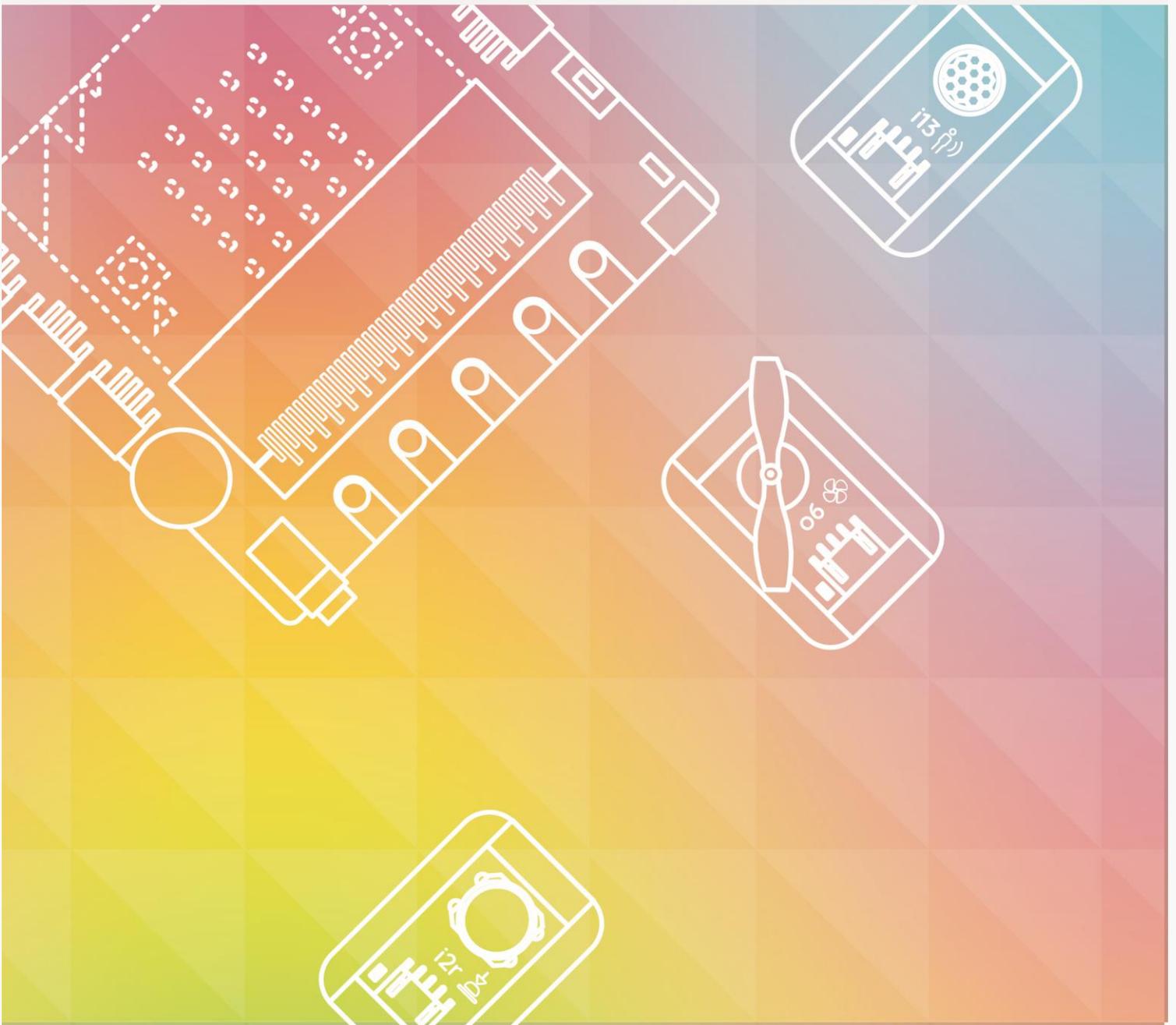




DFROBOT
DRIVE THE FUTURE

micro:bit

造物粒子 入门教程



目录

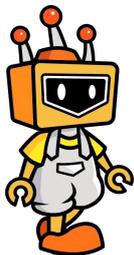
第一话：Mind+与 micro: bit.....	6
1. Mind+平台介绍.....	7
1.1 Mind+下载与安装.....	7
1.2 Mind+界面介绍.....	9
2. Micro: bit 简介.....	10
3. 初次使用 mind+和 micro: bit.....	11
第二话：小试牛刀.....	15
项目一：神秘的 micro: bit.....	15
1. 麦克的故事.....	15
2. 麦克的布兜.....	16
3. 小麦的愿望.....	16
4. 小克的黑板.....	16
5. 麦克小发明.....	19
项目二：闪烁的 LED.....	19
1. 麦克的故事.....	19
2. 麦克的布兜.....	20
3. 小麦的愿望：点亮 LED 灯.....	21
4. 小克的黑板.....	21
5. 小麦的愿望：闪烁 LED 灯.....	24
6. 小克的黑板.....	24
7. 麦克小发明.....	26
项目三：呼吸灯.....	27
1. 麦克的故事.....	27
2. 麦克的布兜.....	27
3. 小麦的愿望：按钮控制 LED 灯.....	29
4. 小克的黑板.....	29
5. 小麦的愿望：旋钮控制 LED 灯.....	31
6. 小克的黑板.....	32
7. 麦克小发明.....	33

项目四：变速风扇.....	34
1.麦克的故事.....	34
2.麦克的布兜.....	34
3.小麦的愿望：按钮风扇.....	35
4.小克的黑板.....	35
5.小麦的愿望：变速风扇.....	38
6.小克的黑板.....	38
7.麦克小发明.....	40
第三话：如虎添翼.....	41
项目一：电子蜡烛.....	41
1.麦克的故事.....	41
2.麦克的布兜.....	41
3.小麦的愿望：声音强度检测.....	42
4.小克的黑板.....	42
5.小麦的愿望：吹气“蜡烛”.....	44
6.小克的黑板.....	44
7.麦克创造屋.....	45
项目二：自动门.....	46
1.麦克的故事.....	46
2.麦克的布兜.....	47
3.小麦的愿望：按钮控制舵机.....	48
4.小克的黑板.....	48
5.小麦的愿望：运动传感器控制舵机.....	51
6.小克的黑板.....	51
7.麦克创造屋.....	52
项目三：音乐盒.....	53
1.麦克的故事.....	53
2.麦克的布兜.....	54
3.小麦的愿望：播放音乐.....	54
4.小克的黑板.....	55
5.小麦的愿望：身体感应音乐.....	56
6.小克的黑板.....	56
7.麦克创造屋.....	58

项目四：炫彩灯带.....	59
1. 麦克的故事.....	59
2. 麦克的布兜.....	60
3. 小麦的愿望：点亮彩虹灯带.....	60
4. 小克的黑板.....	61
5. 小麦的愿望：逐一点亮彩虹灯.....	64
6. 小克的黑板.....	64
7. 小麦的愿望：声音控制彩虹灯带.....	68
8. 小克的黑板.....	68
9. 麦克创造屋.....	70
第四话：大展拳脚.....	71
项目一：自平衡仪.....	71
1. 麦克的故事.....	71
2. 麦克的布兜.....	72
3. 小麦的愿望.....	72
4. 小克的黑板.....	74
5. 麦克发明室.....	77
项目二：DJ 演奏台.....	79
1. 麦克的故事.....	79
2. 麦克的布兜.....	80
3. 小麦的愿望.....	80
4. 小克的黑板.....	81
5. 麦克发明室.....	84
项目三：可移动门铃.....	85
1. 麦克的故事.....	85
2. 麦克的布兜.....	85
3. 小麦的愿望.....	86
4. 小克的黑板.....	87
5. 麦克发明室.....	91
项目四：拆弹游戏.....	92
1. 麦克的故事.....	92
2. 麦克的布兜.....	93

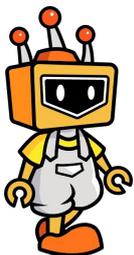
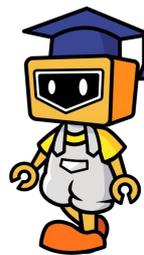
3.小麦的愿望.....	94
4.小克的黑板.....	94
5.麦克发明室.....	104

第一话：Mind+与 micro: bit



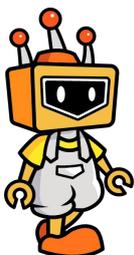
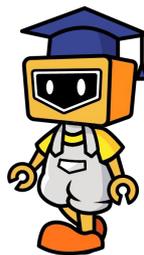
小克老师，我看到越来越多的人开始学习编程，他们设计的小发明都很有趣，我也想设计自己的东西~

哦？小麦同学，你想要设计什么呢？



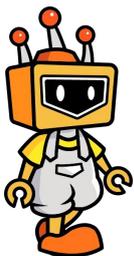
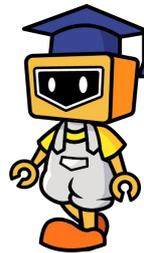
我想要给写字台上添加一个自动感应的台灯；
想要送朋友一个炫彩的音乐盒；
想要做一个遥控赛车送给同学；
想坐在沙发上就能打开家里的门；
甚至也想要制作一系列有趣的小发明！

既然你这么感兴趣，那小克老师就给你讲一讲神奇的图形化编程软件 **Mind+**，你还可以与集成好多功能的电子硬件、单片机 **micro: bit** 成为好朋友哦！



可是小克老师，我没有任何基础，好担心学不会呀~

这不用担心哦，你可以不用绞尽脑汁地去学习枯燥的代码就可以让生活中出现各种有趣的智能装置。



真的吗？小克老师，让我们开始学习编程神器 **Mind+** 以及神奇的 **micro: bit** 吧！

1. Mind+平台介绍

图形化编程软件有不少，硬件也多的让人眼花缭乱。可是有些编程软件只能在电脑屏幕上操作，或者只能支持某些特定的硬件和机器人。如果有一款软件不仅能图形化编程，还能兼容各种硬件，那就节省了不少的麻烦啦！

Mind+就是这样的软件！

Mind+不仅仅适用于中小学生的学习，还可以为想提高自身技能的“创客”提供 Arduino、python/c/c++等高级编程语言的学习环境。它支持 Arduino、micro: bit、掌控板等各种开源硬件，只需要拖动图形化程序块即可完成编程，让大家轻松体验创造的乐趣。

如果你什么都不懂，希望能一步一步跟着学，可以用 Mind+；如果你想学习代码式编程可以用 Mind+；如果你觉得下载程序太麻烦，想实时执行程序，可以用 Mind+；如果你想要物联网，可以用 Mind+；如果你想找社区，还可以用 Mind+！

1.1 Mind+下载与安装

STEP1: 下载 Mind+编程软件（本教程基于 V1.7.0 版本）

打开网页下载：<http://mindplus.cc>



图 1.1.1 Mind+下载界面

STEP2: 安装驱动

下载成功之后点击上面的“教程”，根据提示进行驱动安装。

(1) 点击上方“视频教程”按钮打开教程。



图 1.1.2 打开入门教程

(2) 点击教程，打开课程视频悬浮窗



图 1.1.3 打开课程悬浮窗

(3) 根据视频悬浮窗中的教程，一键安装驱动，这一步很重要！安装一次，一劳永逸！



图 1.1.4 安装驱动

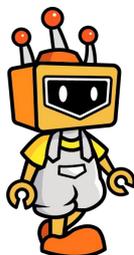
STEP3: 切换“上传模式”（本教程均为“上传模式”下操作）

(1) 点击右上角“上传模式”按钮，等待切换

(2) 切换“上传模式”模式成功

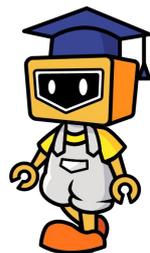


图 1.1.5 切换模式



小克老师，两种模式有什么区别呢？

小麦同学，实时模式是将脚本区可执行的程序在硬件和 Mind+ 舞台中实时执行；而上传模式是将程序上传到硬件设备之后执行哦。



1.2 Mind+界面介绍

下载安装成功之后让我们仔细看下 Mind+编程界面。如果把整个软件比作一个舞台的话，那么不同区域的功能是什么呢？



图 1.2.1 Mind+界面

菜单栏：用来设置软件的区域，这里就是整个“舞台”的幕后啦，没有菜单栏的帮助，连上台表演的机会都没有。“舞台”的幕后都有什么呢？

“项目”菜单可以新建项目、打开项目、保存项目等。

“教程”菜单可以查看大量教程，以及对新手友好的视频教程和示例程序。

“连接设备”菜单能检测到连接的设备，并且可以选择连接或是断开设备。

“上传模式/实时模式/Python模式”按钮切换程序执行的模式。

“（设置）”按钮用于设置软件主题、语言、学习基本案例，在线或加入交流群进行咨询。

指令区：这里是“舞台”的“道具”区，为了完成各种眼花缭乱的动作，需要很多不同的道具组合。在“扩展”里，可以选择更多额外的道具，支持各种硬件编程。

脚本区：这里就是“舞台表演”的核心啦，所有的“表演”都会按照“脚本区”的指令行动，这里是大家都能看得懂的图形化编程。拖拽指令区的指令就能在此编写程序。

代码查看区：如果想弄清楚“脚本区”图形化指令怎样用代码表述，这里是个好地方。

串口区：想知道“表演”的效果如何，那必须要和“观众”互动啦。这里能显示下载状况，比如可以看到程序有没有成功下载，哪里出错了；程序运行状况；还能显示串口通信数据，也就是说，如果你的 micro: bit 板外接了一个声音传感器，那么你就可以看到在这里显示的声音数值大小。这里还有：串口开关、滚屏开关、清除输出、波特率口、串口输入框、输出格式控制。

2. Micro: bit 简介

Micro: bit 是一款由英国广播公司 (BBC) 设计的 ARM 架构的单片机，内含板载蓝牙、加速度计、电子罗盘、三个按钮和 5×5LED 点阵。

使用 micro:bit 可以制作一些有趣和酷炫的小发明:打电话时它相当于一个微型手机，板载蓝牙可以和手机互联，实现通过手机发送指令控制单片机；无线连接功能可以让多块 micro:bit 板实现远程通信，完成一些需要远距离控制的任务；电子罗盘可以感知上下左右和东西南北，利用它可以制作不会迷路的装置；micor:bit 板中间的 5x5LED 点阵显示屏，可以创作有趣的表情包和小动画；点阵显示屏两边有两个可编程按钮，可以控制游戏操作或暂停/播放一首音乐.....

micro: bit 自带常见的感应装置，例如光线传感器，它在与光线有关的项目中可以派上大用处；温度传感器能感知环境的温度。新版micro: bit 还添加了内置的麦克风和扬声器等功能。因此 micro: bit 本身就可以实现很多生活中常见的智能化功能。

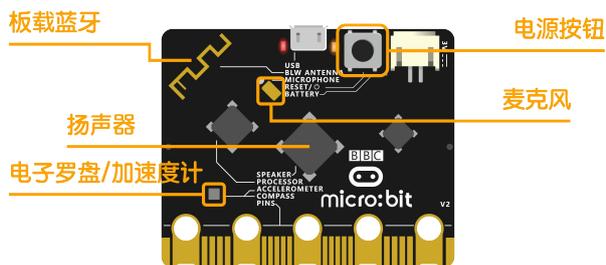


图 1.2.1 micro: bit V2 正面

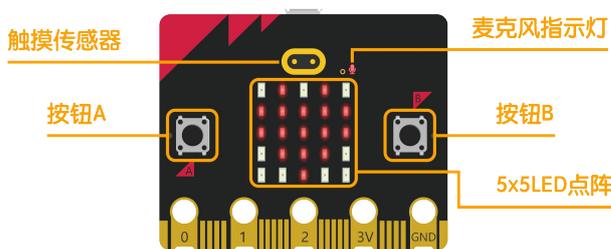


图 1.2.2 micro: bit V2 背面

micro: bit 与 Boson扩展板的结合

micro: bit 的使用限制

1.micro:bit 自身的驱动能力较差，无法直接驱动电机，舵机等大电流设备，因此需要通过金手指插槽来连接更多外部设备。

2.micro:bit 本身的供电和信号电压为 3.3V，无法直接和常用的 5V 电子元件一起使用。

Boson扩展板的好处

- 1.Boson扩展板能轻松将 micro: bit 额外的引脚引出，即插即用，简单快捷；
- 2.Boson扩展板含有扩展接口，连接稳定性更强；
- 3.外接的 USB 供电口 VIN 能为外部元件提供额外的供电，更好地支持灯带、舵机等大电流设备；（运行程序时，USB 连接线还是需要连接在 micro: bit 上哦）

4. Boson 扩展板同时兼容 micro: bit 和 掌控板 作为主控板。
 (本课程请切换至 micro: bit 模式)

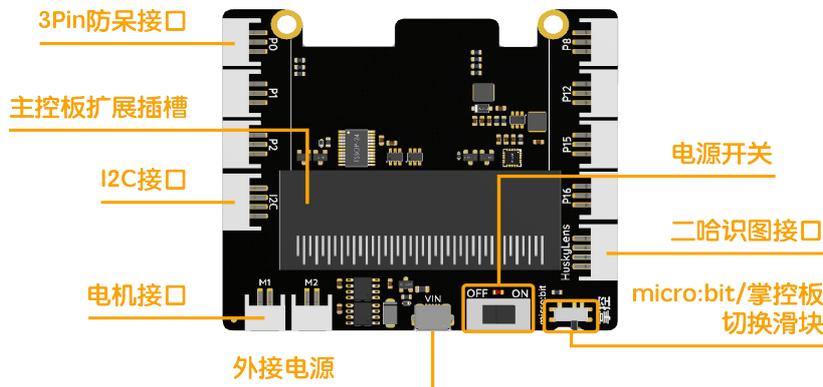


图 1.2.3 Boson 扩展板



micro: bit 扩展板最大额定电流为 500mA，使用前请确定元件的最大功率。勿将扩展板与大型舵机一起使用，否则可能引起板载元件损坏哦！

3. 初次使用 mind+和 micro: bit

本课程基于 Windows 操作系统开发，同时兼容 mac 系统。请准备电脑和如下物品：

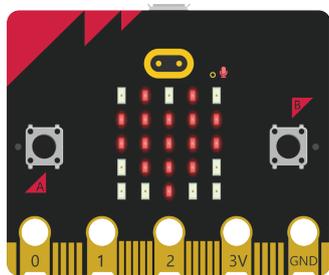


图 1.3.1 micro: bit 主板



图 1.3.2 micro USB 连接线

STEP1: 选择 micro: bit 模块

1. 点击左上角 “项目” 菜单中的 “新建项目”

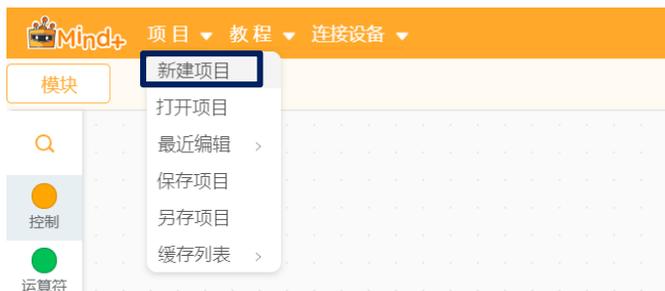


图 1.3.3 新建项目

2. (1) 选择界面左下角“扩展” (2) 选择“主控板” (3) 选中“micro: bit”



图 1.3.4 选中“扩展”

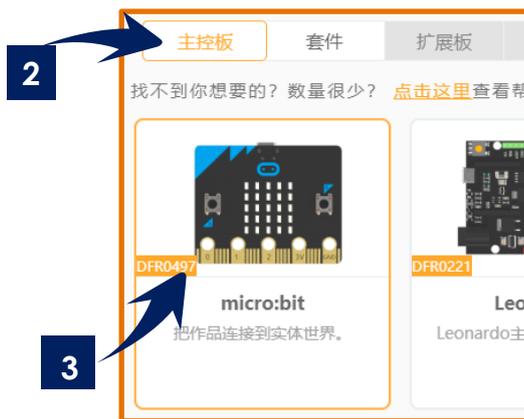


图 1.3.5 选中“micro: bit”

STEP2: 编辑一个简单的程序

编辑一个可以在 micro: bit 板的 LED 点阵屏上显示一个图案的程序:

1.将暂时不需要的指令，丢到指令区即可删除



图 1.3.6 删除指令

2.点击“micro: bit” 模块



图 1.3.7 点击“micro: bit” 模块

3.找到“显示图案或内置图案” 指令

4.将“显示图案或内置图案”放到“micro: bit 主程序开始”下面



图 1.3.8 编辑程序

STEP3: 上传程序到设备

1.将 micro: bit 单片机通过 USB 连接线连接到电脑，这时候 micro: bit 板上指示灯会亮起。

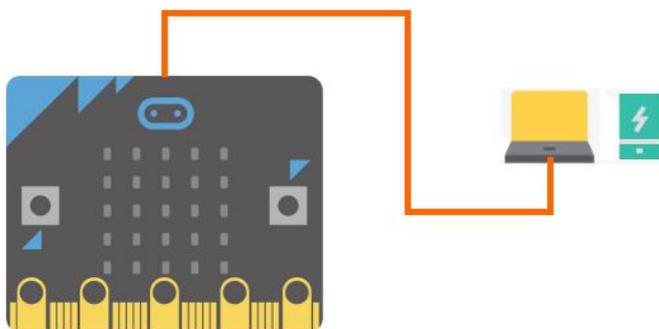


图 1.3.9 micro: bit 连接电脑

2.点击菜单栏上“连接设备”，找到并点击串口号"COM-Microbit V2"（Mac系统为"/dev/cu.usbmodem(Microbit V2)"），使 micro:bit单片机与电脑相连。

3.设备连接成功后，编好程序，点击“上传到设备”（如果此时未连接设备，会提示“没有连接到设备”），上传过程中会显示上传进度，上传成功后，提示框自动消失。

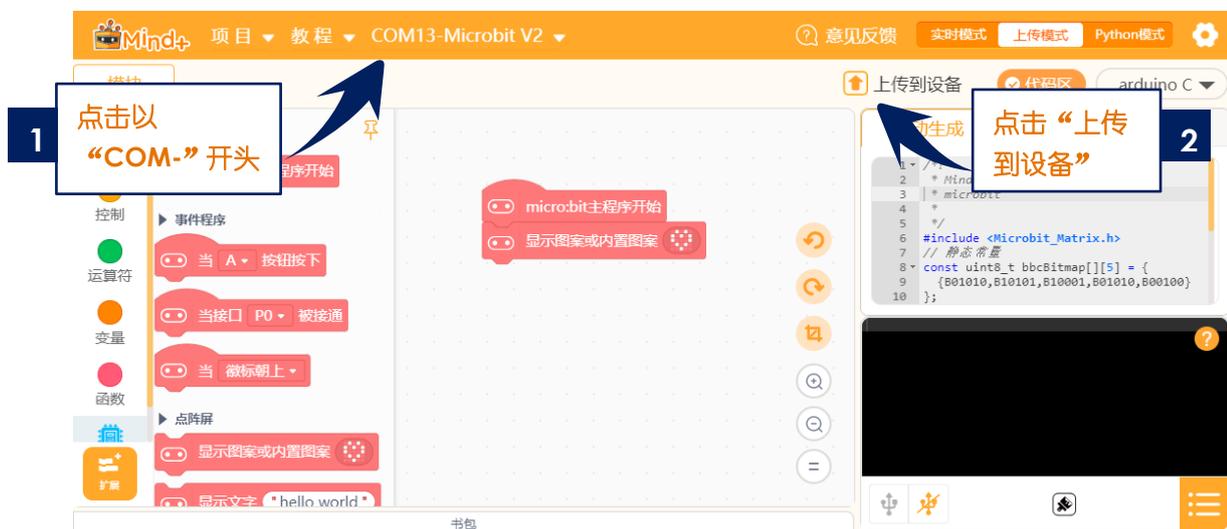


图 1.3.10 执行程序

只有在上传程序时 micro: bit 主板背面的电源信号灯会闪烁，上传完成后电源信号灯停止闪烁，保持长亮。

试试上传之后，你的 micro: bit 板上有没有出现一个小“心心”吧~

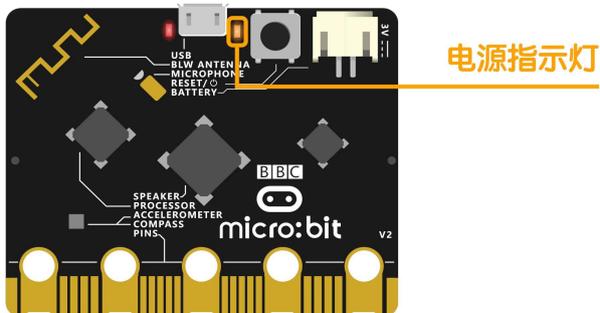
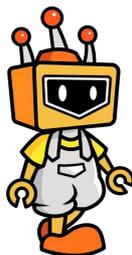


图 1.3.11 电源提示灯

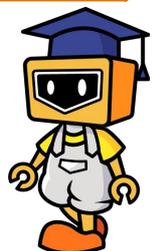


以上是使用 micro: bit 的基本方法。如果在使用中存在任何疑问或者建议，欢迎访问我们的论坛联系我们。论坛链接: <http://www.dfrobot.com.cn>

记得来逛 DFRobot 的社区看看更多的教程和精彩的项目哦。我们同样希望你能够把你自己的项目或者想法发在论坛上分享。欢迎成为我们的一员。

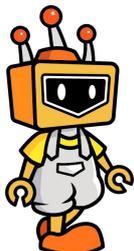
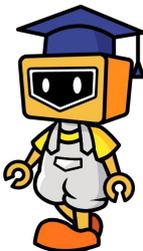


怎么样，小麦同学，学过这些之后觉得有什么进步吗？



当然啦！小克老师，这些讲解使我对 Mind+以及 micro: bit 有了初步的认识，对了，我还绘制出了小心心哦！

小麦同学你真棒！这才刚刚开始，我们以后会对 Mind+以及 micro: bit 有更熟悉的认识，也会使用他们做出你想要的智能装置哦！



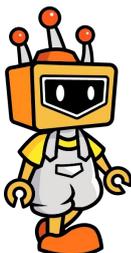
太好了小克老师，我一定会认真学习的，请您教我更多的知识吧~

第二话：小试牛刀

经过前面的知识介绍，你是不是跃跃欲试了呢？现在，我们就开始 micro: bit 的神奇之旅吧！

项目一：神秘的 micro: bit

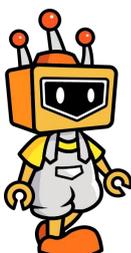
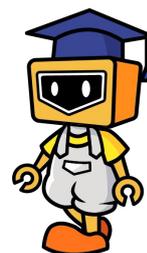
1. 麦克的故事



小克老师，我常常在手机上或者电脑上会见到一些萌萌的，很可爱的表情包，比如这样的：

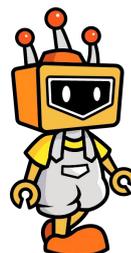
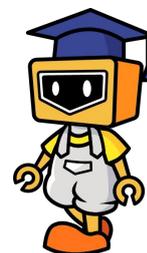


小麦同学，如果你喜欢，也可以制作属于自己的小表情哦~



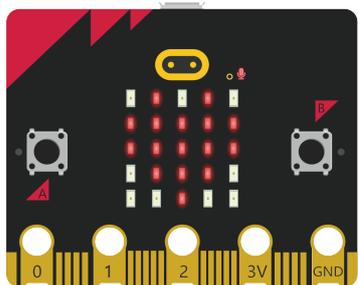
哇，真的吗？小克老师，快教我怎么做吧，我要带给小朋友一起看~

那就让我们继续使用新朋友 micro: bit，通过它可以做出属于你自己的表情包哦~



小克老师，我好期待呦！那请您教授我该怎么做吧~

2. 麦克的布兜



1 × micro: bit 主控板

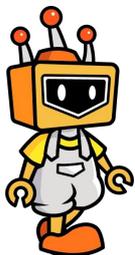


1 × micro USB 连接线



麦克的布兜里有这个项目需要的所有元件哦~

3. 小麦的愿望



看到板子上有 25 个小灯，我想要控制每个灯的亮灭，让它呈现出不同的好看的形状~

4. 小克的黑板

STEP1: 将 micro: bit 主控板用 USB 连接线连接在电脑上。

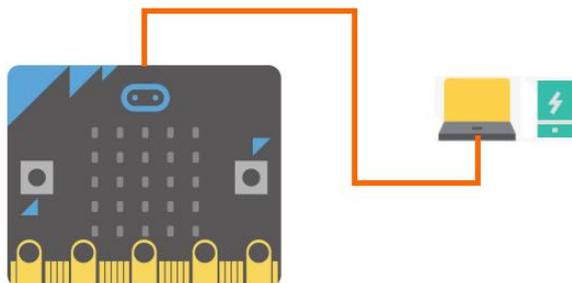


图 2.1.1 连接电脑

STEP2: 编写程序

①打开 Mind+编程界面，选择新建项目，并且从左下角“扩展”中选择“micro: bit”主控板。



不要忘记选择“micro: bit”主控板的步骤哦：（1）选择界面左下角“扩展”（2）选择“主控板”（3）选中“micro: bit”

②单击指令区最下面的“micro: bit”，在指令区的右侧模块中找到“micro: bit 主程序开始”模块，这个模块是必不可少的哦。

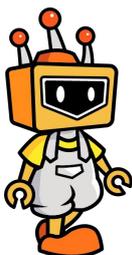


图 2.1.2 micro: bit 主程序开始

③将“循环执行”模块拖到左边区域，即可删除该模块。

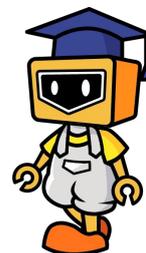


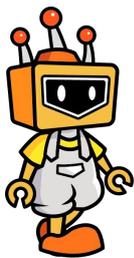
图 2.1.2 删除指令



咦？小克老师，我为什么要删除“循环执行”模块呢？什么时候需要它呀？

小麦同学，当程序需要重复执行的时候就需要“循环执行”啦，以后我们会用到的哦~





哦哦，是这样啊，原来每个模块都有自己的职责！

④点击 micro: bit 模块，在指令区找到“显示图案或内置图案”。将模块拖动至脚本区，并放入“micro: bit 主程序开始”模块下方。



图 2.1.3 拖动指令

⑤单击模块文字后的爱心，可以选择内置图案，也可以点击小方块设计自己的图案。首先，我们先来绘制一个萌萌的“心型”。



图 2.1.4 绘制心形

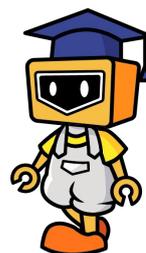
⑥单击“上传到设备”，即可完成上传程序并下载至 micro: bit 主板。这样，micro: bit LED 点阵屏上就会显示“心形”啦。



“上传到设备”之前要记得连接设备哦：点击菜单栏上“连接设备”，找到并点击串口号“COM-Micro bit V2”，使 micro: bit 单片机与电脑相连。

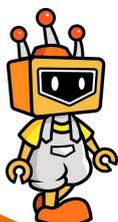
5. 麦克小发明

试着制作一个摆在桌子上的“电子小宠物”吧，它会长什么样子呢？



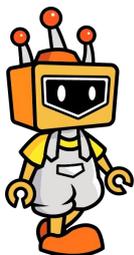
外观设计

程序设计



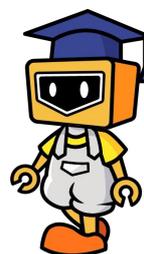
项目二：闪烁的 LED

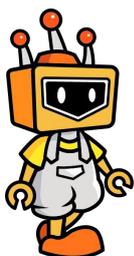
1. 麦克的故事



小克老师，上周用 Mind+和 micro: bit 制作的小心心真的太有趣了，我的同学非常喜欢~

小麦同学，除了小表情，它们还能呈现出很多好玩的东西哦~

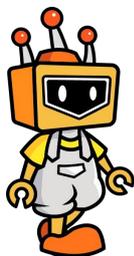
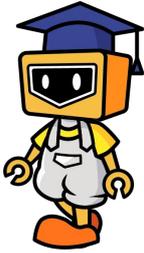




真的吗？小克老师，我特别喜欢五颜六色的小灯，能把它们带到家里吗？

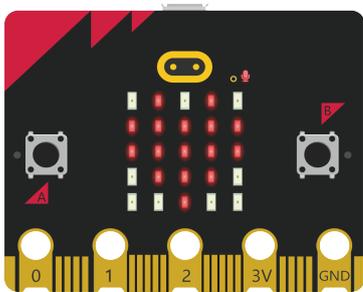


既然你这么喜欢，小克老师当然愿意教你啦，那今天我们就来认识认识新的朋友→LED 小灯吧！

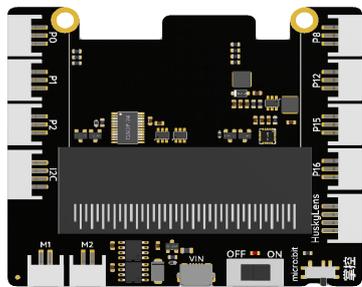


太好了！让我们快快开始吧！

2. 麦克的布兜



1 x micro: bit 主控板



1 x Boson 扩展板

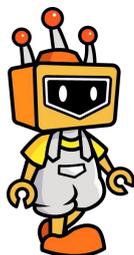


1 x LED 灯模块



1 x micro USB 连接线

3.小麦的愿望：点亮 LED 灯



这个小灯本来是不亮的, 我好想通过一些操作使它亮起来呀!

4.小克的黑板

STEP1: 将 micro: bit 主控板与 micro: bit 扩展板连接起来



图 2.2.1 连接方法

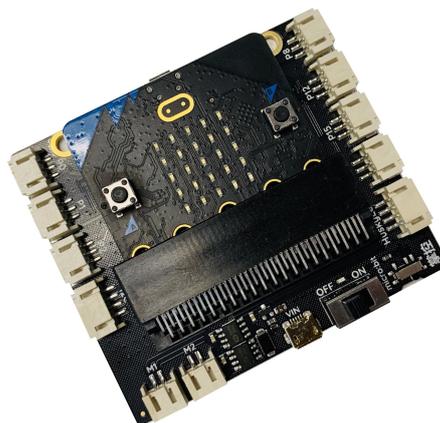
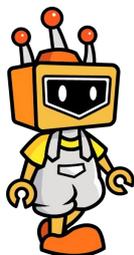


图 2.2.2 连接示意图

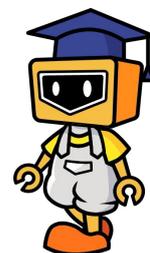


注意：接下来要学习的项目中都是将 **micro: bit** 主控板与扩展板结合在一起使用的，后续这一部分的连接将默认这种连接方式。



咦？小克老师，为什么要将他们两个一起使用呀？单独使用主控板不可以吗？

小麦同学，不要忘记以前学的知识哦~还记得扩展板的好处吗？快回到第一话中的 **Micro: bit** 简介复习一下吧~



STEP2: 将 LED 灯模块连接到扩展板的 P1 号接口

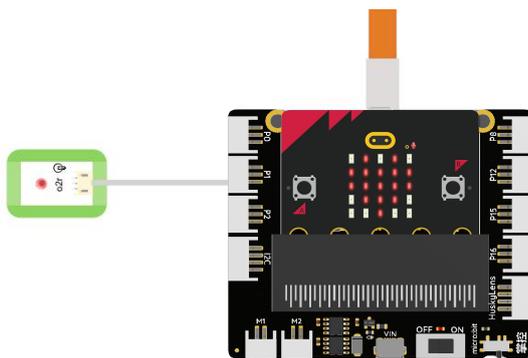


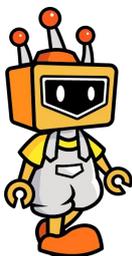
图 2.2.3 连接 LED 灯

STEP3: 编写程序

①打开 Mind+编程界面编写程序，新建项目，并从左下角扩展中选择 micro: bit 主控板。向下滑动“micro: bit”模块，会出现“设置数字引脚 P0 输出低电平”，这里我们需要将“P0”改成“P1”，将“低电平”改成“高电平”。

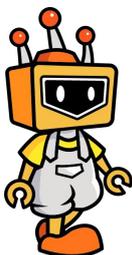


图 2.2.4 设置数字引脚

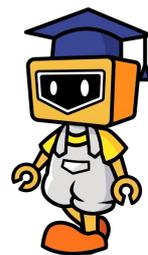


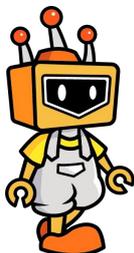
小克老师，这里我就看不懂了，有几个问题想问您~

小麦同学，那你一条一条的问吧，我会耐心的解答的！

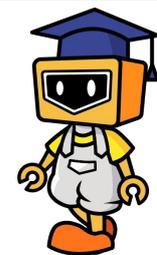


第一个问题：为什么要把 P0 改为 P1 呢？每一次都要改成 P1 吗？



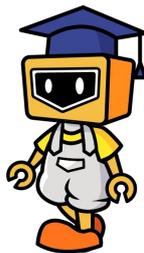


小麦同学，我们是通过控制引脚值来控制对应的模块，因为 LED 小灯是连接在 P1 引脚上，所以才改为 P1 哦~



哦哦，原来是这样，如果连接在 P2 引脚上，就要改为 P2 了~小克老师，第二个问题是：高低电平是怎么回事呢？我不懂。

小麦同学，当数字引脚输出低电平的时候，表示没有电压或电压很小，不能支持 LED 小灯亮起；但当数字引脚输出高电平的时候，相当于有支持小灯亮起的电压，因此小灯就会亮啦！

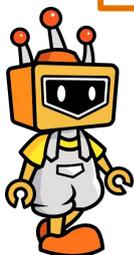


设置数字引脚 P1 输出 高电平

设置引脚为高电平，点亮 LED

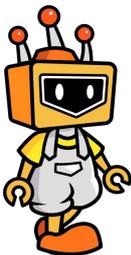
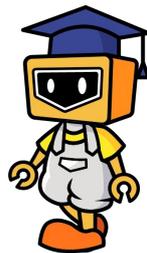
设置数字引脚 P1 输出 低电平

设置引脚为低电平，熄灭 LED



那也就是高电平对应灯亮，低电平对应灯灭！最后一个问题：小克老师，为什么我们选择数字引脚，不选择模拟引脚呢？

小麦同学，你是个善于观察的学生哦~
使用数字引脚说明我们控制的是数字信号，什么是数字信号呢？举个例子：比如这里的 LED 灯只有亮和灭两种模式，没有中间可以随意变化的亮度。那么小灯的亮度就是数字信号啦！
以后我们会学怎么用数字表示灯的亮灭以及什么时候使用模拟信号哦~



谢谢小克老师帮我解决这些困惑，我会自己再认真思考的！让我们继续学习吧！

②从“控制”模块拖出“循环执行”模块，将“设置数字引脚 P1 输出高电平”嵌套进入循环执行中。放在主程序下面。再将最终程序“上传至设备”。



图 2.2.5 执行界面

③实现效果：代码上传成功之后，LED 灯保持常亮。

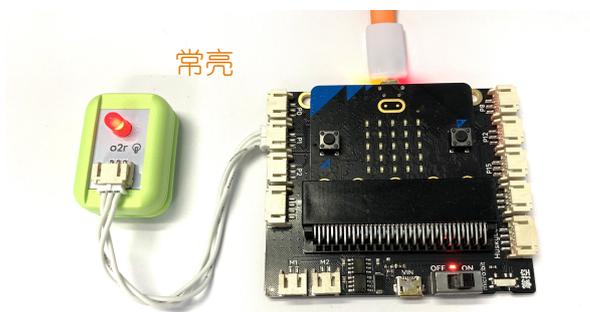
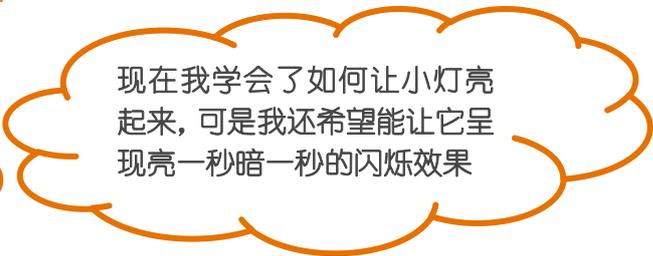
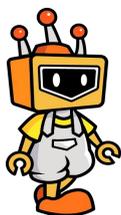


图 2.2.6 效果图

5.小麦的愿望：闪烁 LED 灯



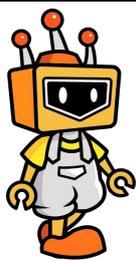
6.小克的黑板

①接线保持不变，新建项目——闪烁 LED 灯。

②怎么实现闪烁呢，需要让 LED 小灯保持 1 秒亮的状态之后再保持 1 秒熄灭的状态，并循环下去，所以需要在“控制”模块里找到“等待 1 秒”。

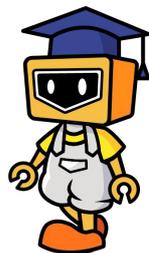


图 2.2.7 “等待 1 秒”界面



小克老师,我想起来了,如果想让程序一直执行,就需要用到“循环执行”模块了!

小麦同学的记忆力很好哦!没错,只有你想要它循环执行的时候才会使用“循环执行”模块哦~



③将“等待 1 秒”放到“设置数字引脚 P1 输出高电平”下方,这样就能让 LED 灯保持亮 1 秒再执行下一个程序。

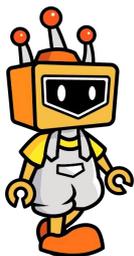


图 2.2.8 拖动“等待 1 秒”

④接下来需要做的是让小灯在熄灭状态下,保持 1 秒钟。鼠标右击“设置数字引脚 P1 输出高电平”,出现菜单选择“复制”。



图 2.2.9 “复制”模块



哇!原来可以右击鼠标选择复制呀!我要记住这个方式,会便捷很多呢!

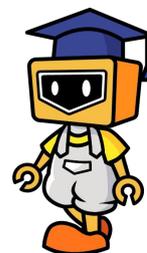
⑤将复制得到的模块放到“等待 1 秒”的下方，并且修改“高电平”为“低电平”，让 LED 灯在熄灭状态下，保持 1 秒钟。因为是循环执行，所以 LED 灯闪烁的程序会一直执行下去哦。



图 2.2.10 执行程序

实现效果：代码上传成功之后，LED 灯保持闪烁状态。

7. 麦克小发明



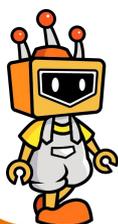
小麦同学，听说你要和朋友去探险，好担心你会迷路哦，不如使用学过的知识做一个可以发送求救灯信号的“求救装置”吧！

小克老师提醒一下哦：如何利用灯光发出 SOS 求救信号呢？只要我们按照“三短，三长，三短”的方式闪烁灯光，就等于发出 SOS 的求救信号啦。

用“亮——灭——亮——灭——亮——灭——亮——灭——”表示三长，“亮-灭-亮-灭-亮-灭-”表示三短，发现了吗？三长亮的时间是三短的三倍哦~快快做起来！

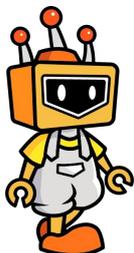
外观设计

程序设计



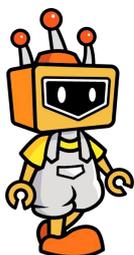
项目三：呼吸灯

1. 麦克的故事



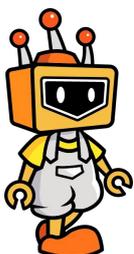
小克老师,这周我要和小伙伴去 DF 星球的星际飞船上玩,可是我好担心有危险!

小麦同学,你们要在飞船上度过好几个小时呢,看来要准备一些应急装置了!



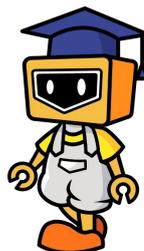
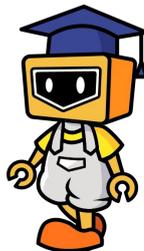
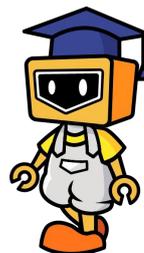
是的小克老师,如果飞船偏离了航向,在茫茫星海中我们迷了路该怎么办呀?

小麦同学,这种时候千万不要慌,你们要先想办法被别的飞船注意到,这样就容易获救了!

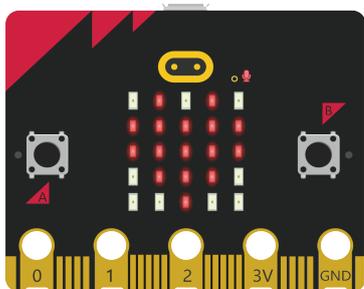


那我们可以做一个明暗交替的呼吸灯,有危险的时候就调到最亮,让别人注意到我们!小克老师你可以教我怎么做吗?

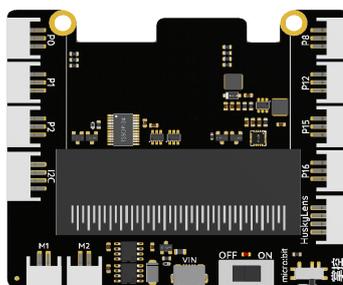
这个主意不错,那你要认真学习了哦!



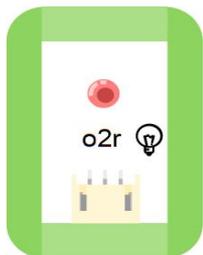
2. 麦克的布兜



1 x micro: bit 主控板



1 x Boson 扩展板



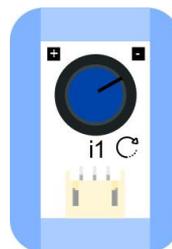
1 × LED 灯模块



1 × micro USB 连接线



1 × 按钮模块



1 × 旋钮模块



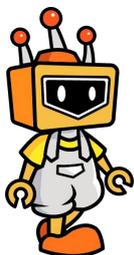
第一次看到按钮和旋钮模块，快来了解一下吧：

按钮模块和我们家用台灯的按钮一样，通过对 micro: bit 编程可以实现按钮被按下，小灯就会被点亮的效果哦；

使用旋钮模块的时候可以旋转不同的角度，控制小灯不同的亮度哦。

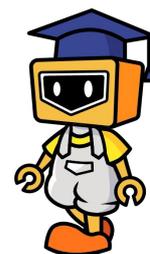
旋钮与按钮有什么异同呢？

两者都是输入模块哦~但按钮只有处于按下的状态时，电路才会接通；而旋钮只要转在一定范围内，电路都会处于接通状态哦。

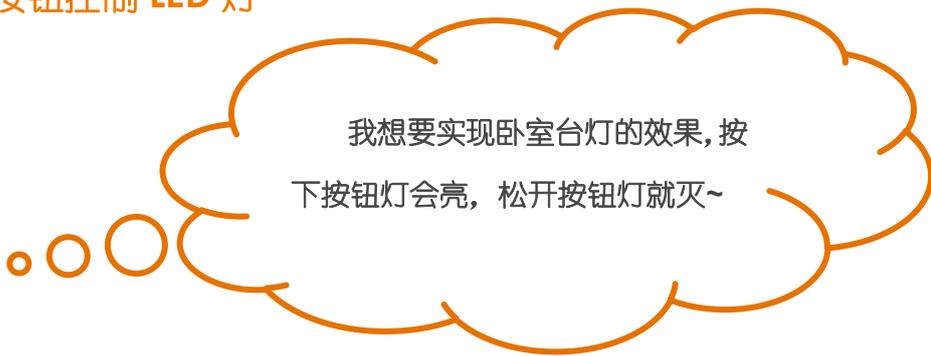
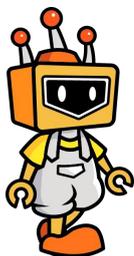


小克老师，输入模块是输入信号的吗？

小麦同学很聪明哦！你说的对，举个例子哦，比如对于按钮控制小灯项目来说，按钮是输入模块，小灯就是输出模块哦！



3.小麦的愿望：按钮控制 LED 灯



4.小克的黑板

STEP1: 首先要把按钮模块接到扩展板 P0 号接口上;把 LED 灯模块接到扩展板 P1 号接口上。

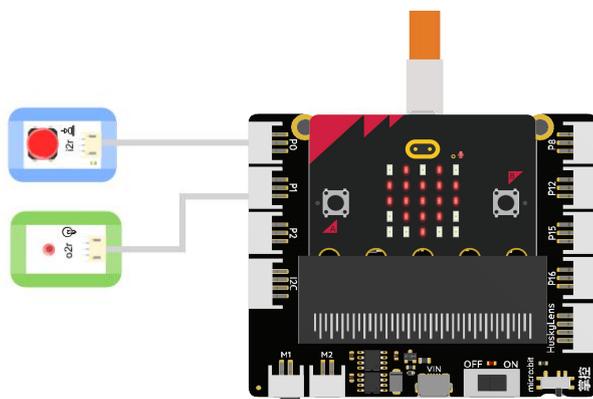


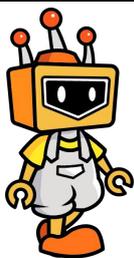
图 2.3.1 接线示意图

STEP2: 编写程序

①单击“micro:bit”模块,选择“读取数字引脚 P0”指令,并拖到脚本区。

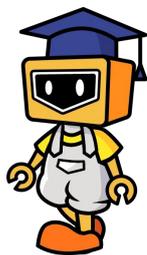


图 2.3.2 读取数字引脚 P0 指令



小克老师，这里为什么没有使用高低电平呢？

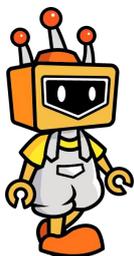
小麦同学，我们先要通过检测 P0 号引脚的值判断按钮是否被按下。当按钮被按下时，设置值为 1、引脚为高电平，LED 灯被点亮；当松开按钮时，设置值为 0、引脚为低电平，LED 灯熄灭。所以不是不用，是时候未到哦~



② “运算符” 模块中的 “=” 可以判断按钮是否被按下，如果按钮值为 “1” 成立，说明按钮被按下，否则，按钮没有被按下。将 “读取数字引脚 P0” 嵌套在等式中。

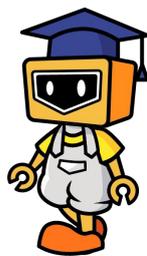


图 2.3.3 “=” 指令



小克老师，那我们在程序中怎么判断按钮是否被按下呢？

小麦同学这个问题问得好，我们可以通过条件判断语句中的 “如果那么执行，否则” 指令来判断按钮是否被按下哦~继续往下看吧~



③在 “控制” 模块中找到 “如果那么执行，否则” 指令。拖入脚本区的循环模块中。



2.3.4 “如果那么执行，否则” 指令

④将等式嵌套入“如果--那么执行”指令中，作为判断条件。



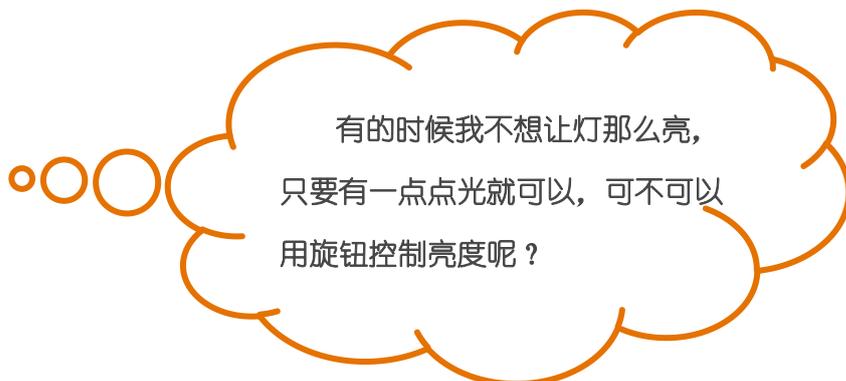
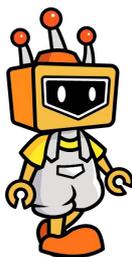
图 2.3.5 指令的嵌套

⑤在“micro: bit”模块中找到“设置数字引脚 P0 输出低电平”指令，将指令修改成“设置数字引脚 P1 输出高电平”，放在“那么执行”下面，当按钮按下之后，就会执行这个指令让小灯亮起来。再将“设置数字引脚 P1 输出低电平”放在“否则”下面，就能实现当松开按钮的时候，外接 LED 小灯熄灭啦。



图 2.3.6 执行程序

5.小麦的愿望：旋钮控制 LED 灯



有的时候我不想让灯那么亮，
只要有一点点光就可以，可不可以
用旋钮控制亮度呢？

6.小克的黑板

STEP1: 把按钮台灯中使用的按钮模块换为旋钮模块。

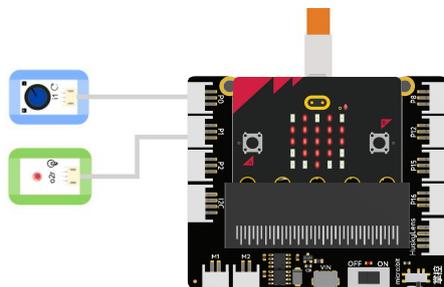


图 2.3.7 连线示意图

STEP2: 编写程序

①旋钮连接板子的 P0 号引脚，单击“micro: bit”模块，选择“读取模拟引脚 P0”。



图 2.3.8 读取模拟引脚



LED 灯的亮度由 P0 对应的旋钮值决定，将旋钮值传递给小灯，可以控制 LED 灯的亮度。micro: bit 板的读取模拟引脚（旋钮）的值范围是 0-1023，输出模拟（LED 灯）的范围也是 0-1023，所以此处直接读取模拟引脚 P0 的值就好啦！

②将“读取模拟引脚 P0”拖入脚本区。



图 2.3.9 “读取模拟引脚 P0” 指令

③将“micro: bit”模块中的“设置模拟引脚 P0 输出”拖入脚本区的“循环执行”中，将 P0 改为 P1。将“读取模拟引脚 P0”嵌套进“设置模拟引脚 P1 输出”指令。



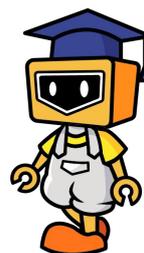
图 2.3.10 执行程序

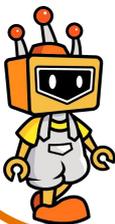


这段代码可以这样理解：获取旋钮当前旋转位置的值，用这个值控制LED小灯的亮度。

7. 麦克小发明

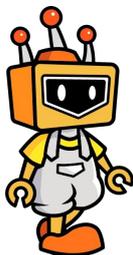
小麦同学，听说你的旧台灯不能调整亮度，也不能调整光线的角度，那现在你能不能设计一个好看又实用的旋钮台灯了呢？



外观设计	程序设计
	

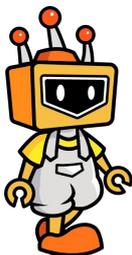
项目四：变速风扇

1. 麦克的故事



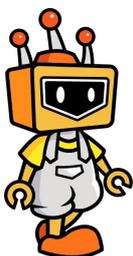
小克老师，DF 星球的星际飞船好好玩啊，我们在宇宙中随着惯性飘荡着，还可以看到舷窗外明暗交替的灯光。

小麦同学，那你们在飞船里还适应吗？



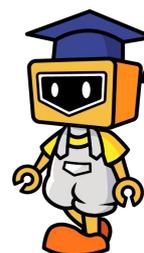
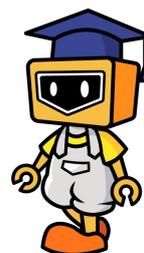
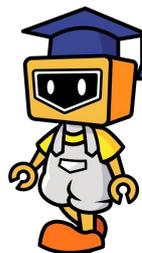
嗯，都还适应，除了...船舱内的温度有些高，我和小伙伴都冒了很多汗珠，当时想如果有个风扇就好了~

小麦同学，那你想自己做一个可以调节风速的风扇吗？

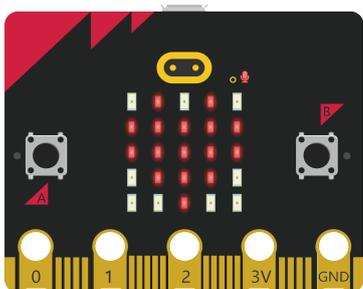


当然想呀！如果有一个可以调节风速的风扇，那我就可以随时吹风解热了！小克老师，您可以教我怎么做吗？

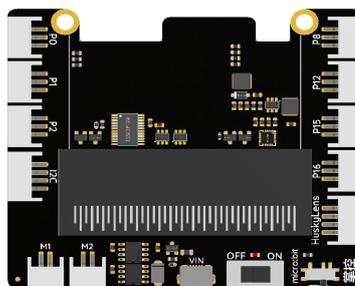
没问题，那你要仔细学习哦！



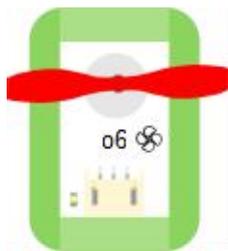
2. 麦克的布兜



1 × micro: bit 主控板



1 × Boson 扩展板



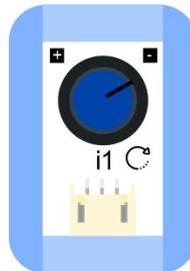
1 × 风扇模块



1 × micro USB 连接线



1 × 按钮模块

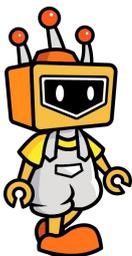


1 × 旋钮模块



风扇模块与家用风扇的原理相似，内部有马达，扇叶可以旋转，转速也可以调整。

3. 小麦的愿望：按钮风扇



我想要一个按钮风扇，这样就能实现坐在沙发上遥控风扇啦~

4. 小克的黑板

STEP1: 把按钮模块接到扩展板 P0 号接口上；把风扇模块接到扩展板 P1 号接口上。

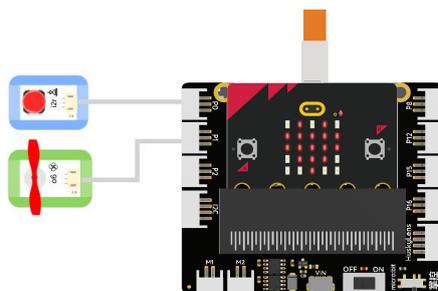
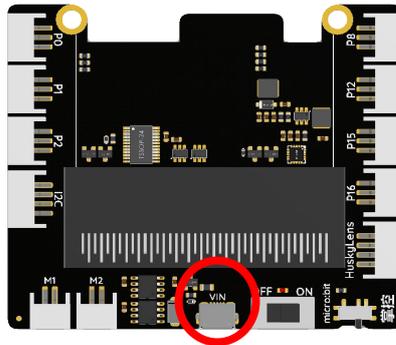


图 2.4.1 接线示意图



看看如何在扩展板上使用风扇吧：

硬件连接时，将 micro: bit 与电脑相连接，进行程序调试和下载；而在实验中，由于风扇或舵机等传感器的耗电量较大，所以，需要外接电源（或电脑 USB 口）独立供电。外接电源接在扩展板的外接电源接口 VIN。



通过检测引脚的值（0 和 1）判断按钮是否被按下。当按钮被按下时（值为 1），设置风扇连接的引脚为高电平，风扇开始旋转，当松开按钮时（值为 0），设置风扇所在引脚为低电平，风扇停止旋转。

STEP2: 编写程序

①单击“micro: bit”模块，选择“读取数字引脚 P0”指令，并拖到脚本区。



图 2.4.2 读取数字引脚

②“运算符”模块中的“=”可以判断按钮是否被按下，如果按钮值为“1”成立，说明按钮被按下，否则，按钮没有被按下。将“数字读取引脚 P0”嵌套在等式中。



图 2.4.3 “=” 指令

③在“控制”模块中找到“如果那么执行，否则”指令。拖入脚本区的循环模块中。



图 2.4.4 “如果那么执行，否则”指令

④将“=”左右成立的条件指令嵌套到条件判断指令“如果那么执行，否则”中。

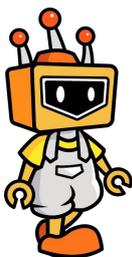


图 2.4.5 指令的嵌套

⑤“micro: bit”模块中找到“设置数字引脚 P0 输出低电平”指令，将指令修改成“设置数字引脚 P1 输出高电平”和“设置数字引脚 P1 输出低电平”，分别放在“那么执行”和“否则”下面，当按钮按下之后，就会执行这个指令让风扇转起来。松开按钮的时候，风扇停止转动。

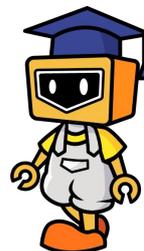


图 2.4.6 执行程序

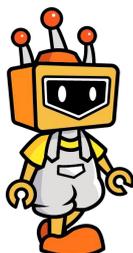


小克老师，这个程序和按钮控制小灯的程序好像呀！

小麦同学很善于观察哦~没错，他们的原理是一样的，以后你也要像现在这样多多思考哦！



5.小麦的愿望：变速风扇



按钮风扇转速太快了，我想要自己控制它的转速，可以做一个变速的风扇吗？

6.小克的黑板

STEP1：把按钮风扇中使用的按钮模块换为旋钮模块。

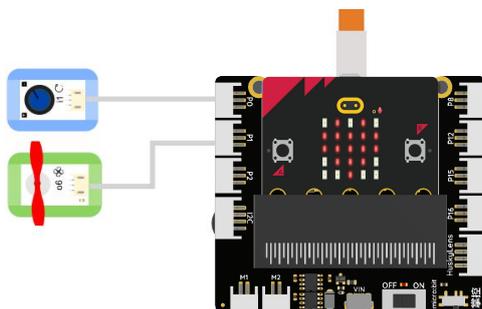


图 2.4.7 连线示意图

STEP2：编写程序

①单击“micro: bit”模块，选择“读取模拟引脚 P0”指令，并拖到脚本区。

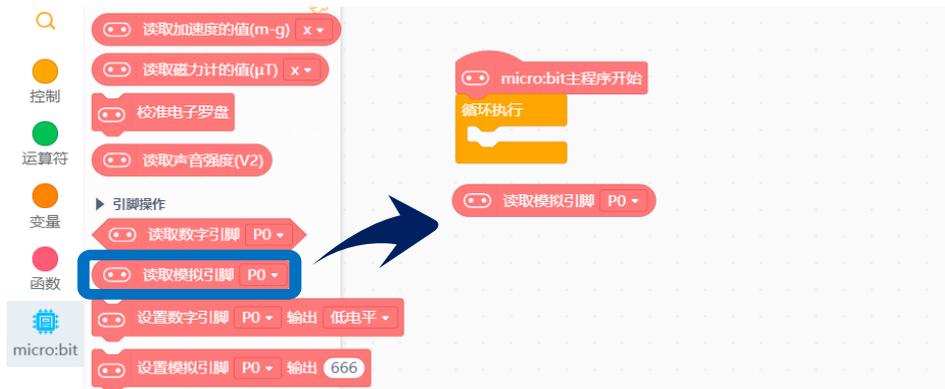
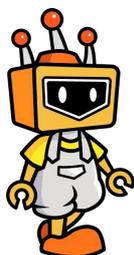


图 2.4.8 读取模拟引脚

②将 micro: bit 模块中的“设置模拟引脚 P0 输出”拖入脚本区的“循环执行”中，将P0 改为 P1。并与“读取模拟引脚 P0”指令嵌套在一起。

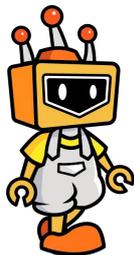
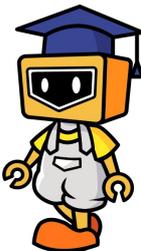


图 2.4.10 执行程序



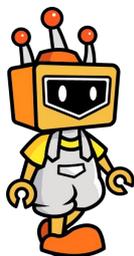
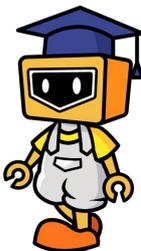
小克老师，为什么可以直接读取旋钮的值呢？

小麦同学，因为风扇的输出范围也是 0~1023，和旋钮的输入范围是一样的哦



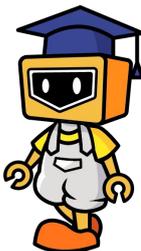
小克老师，如果范围不一样该怎么办呢？

如果范围不一样，我们就需要使用一个新的“映射”指令啦



就是把输入模块的范围映射至输出模块的范围，它就能自己对应啦？

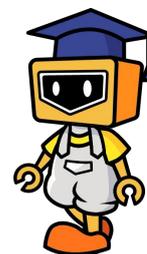
小麦同学，你说的对哦！



7. 麦克小发明

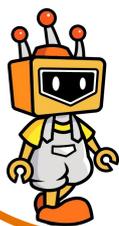
小麦同学，夏天就要来了，你是否想给自己的台灯加上一个小风扇呢，这样的话就不用把风扇搬来搬去了哦。

小提示：如何使用一个 **micro: bit** 板分别控制 **LED** 灯和风扇呢？可以使用其他引脚哦~



外观设计

程序设计

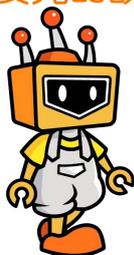


第三话：如虎添翼

除了前面学习的按钮模块、旋钮模块，还有很多常用的、有趣的传感器，使用他们可以制作出更加智能化、好玩和实用的作品，快来认识它们吧！

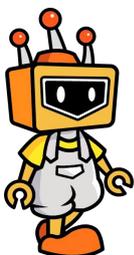
项目一：电子蜡烛

1. 麦克的故事



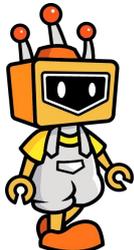
小克老师，好久不见了！我和小伙伴终于从 DF 飞船回来了~

小麦同学，你们玩的开心吗？怎么样，有什么感受吗？



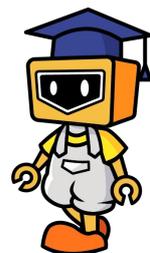
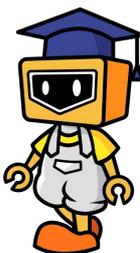
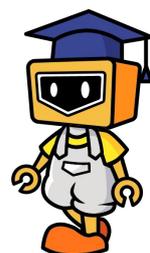
小克老师，我回来之后看到新闻，看到了温室效应呀~好恐怖啊~原来人类的任何活动都可能造成碳排放，比如烧火做饭之类的...

你说的没错哦，所以我们要尽量减少污染哦~

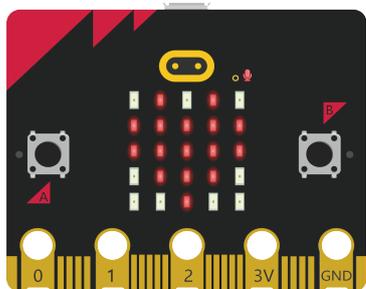


小克老师，我的好朋友快过生日了，为了不污染环境，我想给她制作一个电子蜡烛，你可以教我怎么做吗？

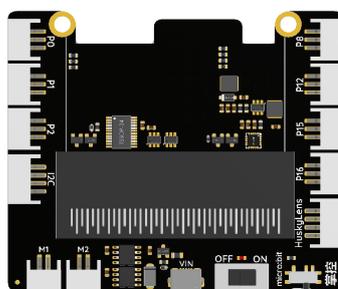
小麦同学，这个想法不错哦，既然你这么想学，那我们就试试吧！



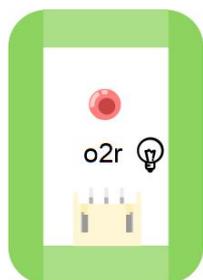
2. 麦克的布兜



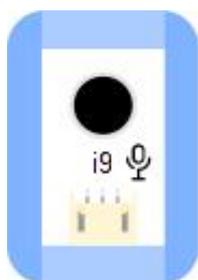
1 x micro: bit 主控板



1 x Boson 扩展板



1 × LED 灯模块



1 × 声音传感器模块



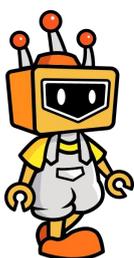
1 × micro USB 连接线



第一次看到声音传感器模块，快来了解一下吧：

声音传感器模块能够感知外界声音的强弱，经过编写代码后可以实现声音强弱控制 LED 小灯的亮灭哦。

3.小麦的愿望：声音强度检测



我想要知道自己的声音多大，比如如果板载LED点阵屏上面写 1，就表示声音小；写 2，就表示声音大~



如何通过学过的知识进行设计呢？

我们可以设置当声音强度模拟值小于 80 的时候，板载LED点阵屏显示数字“1”，表示声音较小，强度为 1 级；

否则，声音强度模拟值大于 80 的时候，板载LED点阵屏显示数字“2”，表示声音较大，强度为 2 级。

4.小克的黑板

STEP1: 首先要把声音传感器模块接到扩展板 P0 号接口上；把 LED 灯模块接到扩展板 P1 号接口上。

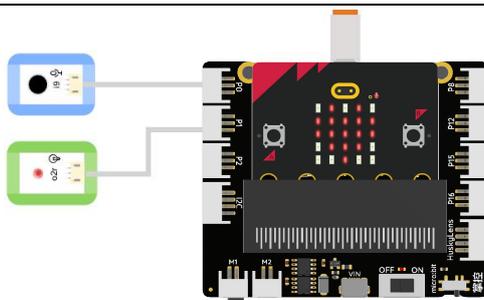


图 3.1.1 接线示意图

STEP2: 编写程序

①根据上一个小贴士的提示，我们可以进行初步的设计啦：单击指令区最下面的“micro: bit”选择“读取模拟引脚 P0”拖动至脚本区；将“控制”代码模块中的条件语句“如果--否则”拖动至脚本区；将“运算符”中的“<”拖动至脚本区。



我们顺便来复习一下“如果--否则”语句吧：

条件语句代码“如果--否则”的功能：如果满足右边逻辑判断语句。则执行在“那么执行”下方的代码语句，反之，如果不满足逻辑判断语句，则执行的是在“否则”下方的代码。



图 3.1.2 “如果--否则”指令



图 3.1.3 “<”指令

②通过“读取模拟引脚 P0”连接的声音传感器的模拟信号数值（再次强调，模拟数值范围在 0—1023 之间哦）来检测声音的强度。

③设定“声音强度探测”的实现程序。逻辑指令中的“<”可以判断声音强度的模拟值是否达到设定的标准。

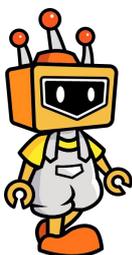
如图 3.1.4，读取模拟引脚 P0 的数值“<80”成立的时候，也就是声音强度为 1 级的时候，板载小灯显示数字“1”；

否则，模拟读取引脚 P0 的数值 “>80” 时，也就是声音强度为 2 级，板载小灯显示数字 “2”。在 “否则” 代码之后，可以增加 “等待 1 秒”，防止数字闪烁。

```

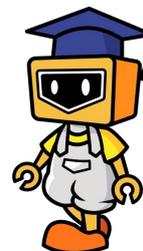
micro:bit主程序开始
循环执行
  如果 读取模拟引脚 P0 < 80 那么执行
    显示文字 "1"
  否则
    显示文字 "2"
    等待 1 秒
    
```

图 3.1.4 执行程序

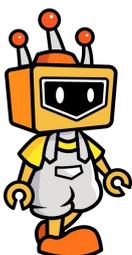


小克老师, 按照程序我真的可以检测出自己的声音强度呢!

小麦同学, 你做的不错哦, 要是你的同学已经很用力说话或吹气了却还是不能显示相应的数字, 可以试试调节判断数值哦!



5.小麦的愿望：吹气“蜡烛”

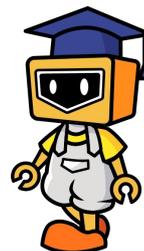


我想要不吹气的时候小灯一直亮, 当我用力吹小灯, 它就会灭, 就像蜡烛一样

6.小克的黑板

编程过程中，通过设置连接声音传感器的引脚值来控制小灯的亮暗哦。

小克老师再来解释一下：
 没有吹气时，也就是声音没有达到设定的强度，小灯呈现亮的状态，小灯对应的引脚应该为高电平；
 吹气时，如果声音达到了设定的强度，小灯呈现灭的状态，小灯对应的引脚应该为低电平。



①使用“设置数字引脚 P0 输出”这一模块，由于当前小灯连接的是 P1 引脚，将 P0 修改为 P1，拖至脚本区。



图 3.1.5 “设置数字引脚”指令

②设定“电子蜡烛”的实现程序。

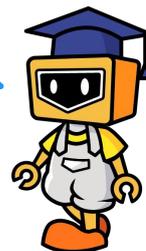
当声音传感器模拟值小于设定值 80 的时候，小灯保持亮起的状态，反之，轻轻一吹，声音强度大于 80 时，小灯将保持 2 秒的熄灭状态。

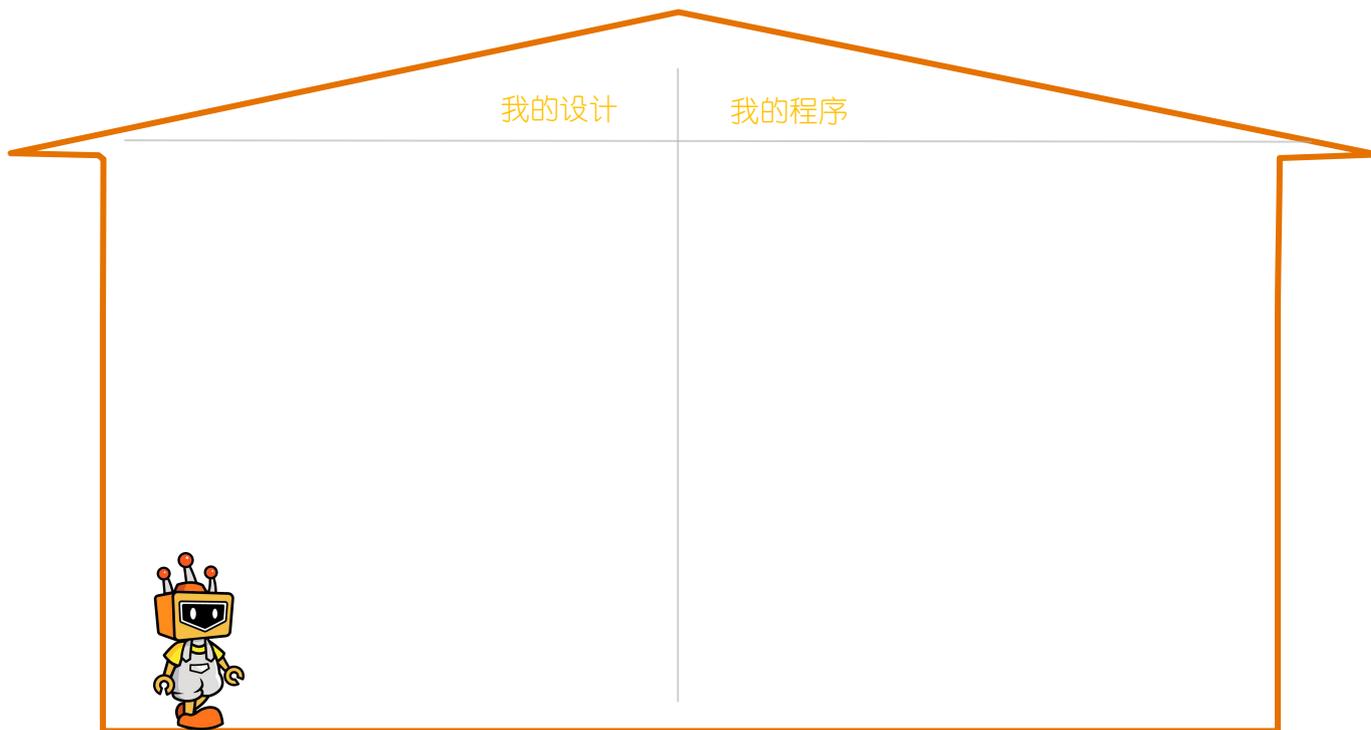


图 3.1.6 执行程序

7. 麦克创造屋

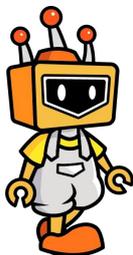
小麦同学，以后你也可以提倡其他同学制作这样既环保又有趣的生日蜡烛哦~除了做生日蜡烛，你还能用这节课布兜里面的东西做出其他好玩的东西吗？快把你的好想法设计出来吧~





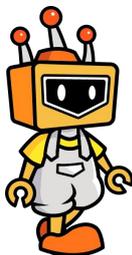
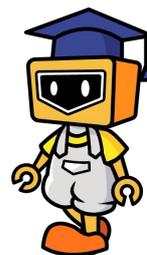
项目二：自动门

1. 麦克的故事



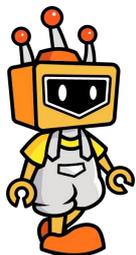
小克老师，上周做的蜡烛简直太有趣了，我发现咱们的生活中还有好多有趣的发明呀！

哦？小麦同学，你说说看都发现什么好玩的了？



小克老师，上次我跟同学去图书馆借学习机器人的书，走到门口的时候发现不用自己开门，门自动开了。

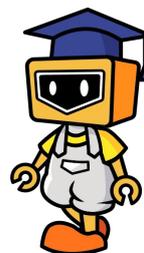
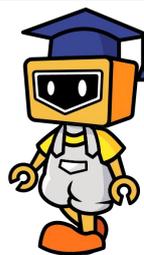




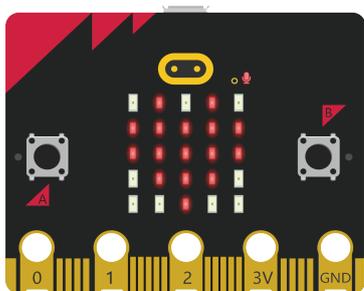
小麦同学，这种仔细观察周围的行为值得表扬哦，那你猜猜门为什么会开呢？

小克老师，难道是因为有人知道我们要过去，所以在后面偷偷给我们开门吗？

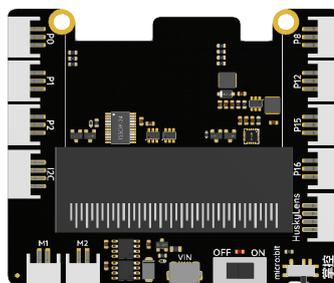
哈哈，小麦同学不是的哦，是因为有一个很神奇的装置，当它知道有人经过就会自动开门~



2. 麦克的布兜



1 × micro: bit 主控板



1 × Boson 扩展板



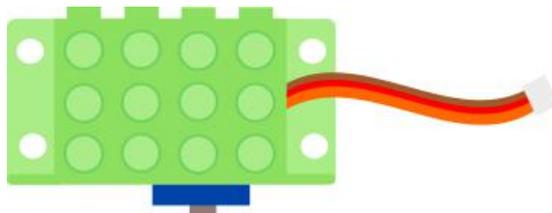
1 × 按钮模块



1 × micro USB 连接线



1 × 运动传感器模块



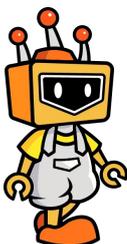
1 × 舵机 (伺服机构)



第一次看到运动传感器模块和舵机模块，快来认识一下吧：

舵机：它能旋转特定角度，根据型号不同，旋转范围有 0 ~ 180 度或 0 ~ 360 度。它的特点是转速相对电机小，力气更大。一般用于机械臂完成更灵活更高难度的动作，也可以让机械装置更加精确地运动哦~（使用时尽可能避开 0 度和 180 度临界值，避免舵机抖动）

3.小麦的愿望：按钮控制舵机



我想要用按钮控制舵机，如果按下按钮，舵机转动到 100°，不按按钮，舵机转到 10°。

4.小克的黑板

STEP1: 把按钮模块接到扩展板 P0 号接口上；把舵机接到扩展板 P1 号接口上。

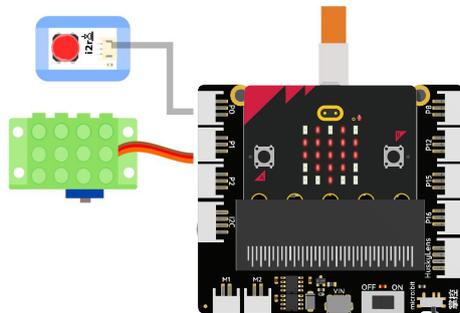


图 3.2.1 接线示意图



在编写程序之前需要增加控制器模块哦：

如果我们想要控制舵机的运行，那就需要增加一个控制舵机的指令模块，怎么调用呢？

如图 3.2.2 和 3.2.3，在左下角的“扩展”中找到“执行器”，选择“舵机模块”，点击舵机模块后再返回到编程界面即可。



图 3.2.2 “扩展” 模块



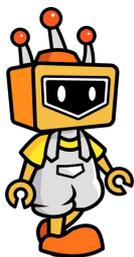
图 3.2.3 执行器-舵机模块

STEP2: 编写程序

①点开执行器的模块可以控制舵机，通过设置 P1号引脚可以控制舵机的值。

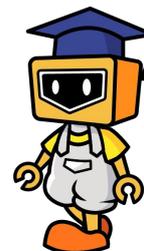


图 3.2.4 设置舵机值



小克老师，控制舵机值的意思是控制它的角度吗？

是的，小麦同学，通过输入舵机的值来实现你想要的转动角度~



②设置如果按钮按下，舵机转动到 100°，LED 小灯显示“o”；否则舵机转动到 10°，LED 小灯显示“x”。需要用到的指令为：“如果--否则”。



图 3.2.5 “如果-否则” 指令



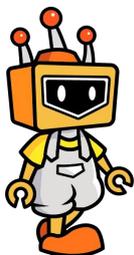
注意：由于舵机（伺服机构）耗电量大，需要外接电源哦！！
 如果你忘了怎么外接电源，翻回至“按钮风扇”复习一下吧！

③最终程序：

```

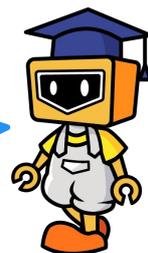
micro:bit主程序开始
循环执行
  如果 读取数字引脚 P0 = 1 那么执行
    设置 P1 引脚伺服舵机为 100 度
    显示图案或内置图案
  否则
    设置 P1 引脚伺服舵机为 10 度
    显示图案或内置图案
  
```

图 3.2.6 执行程序



小克老师，这就是我想要的效果！不过刚开始我没有外接电源，还以为舵机坏了呢！

所以小麦同学，一定要耐心和细心哦！



实验效果：

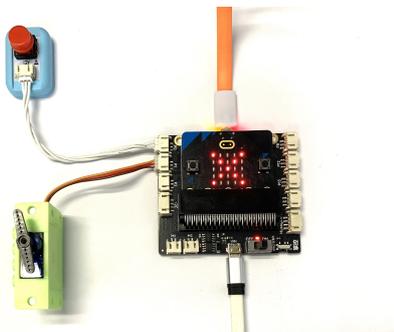


图 3.2.7 实验效果-未按按钮

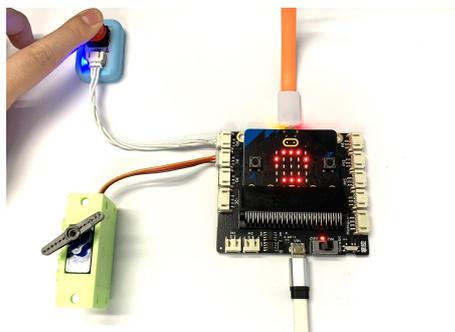
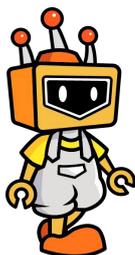


图 3.2.8 实验效果-按下按钮

5.小麦的愿望：运动传感器控制舵机



我想要自动门的效果，只要有人经过，它（舵机）就转到 100°，否则就转到 10°。

6.小克的黑板

STEP1: 将舵机连接至 P1 引脚；将运动传感器连接至 P0 引脚。

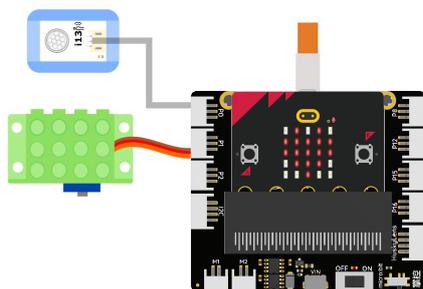


图 3.2.9 连线示意图

STEP2: 编写程序

① 通过读取数字引脚 P0 的值来判断运动传感器的数字信号数值。



图 3.2.10 “读取数字引脚”指令

②当运动传感器检测到有人经过时，实现舵机转动到 100°；没有人经过时，舵机旋转到 10°。需要用到的指令为：“如果-否则”。



图 3.2.11 “如果-否则”指令

③最终程序：

```
micro:bit主程序开始
循环执行
  如果 读取数字引脚 P0 = 1 那么执行
    设置 P1 引脚伺服舵机为 100 度
    显示图案或内置图案
  否则
    设置 P1 引脚伺服舵机为 10 度
    显示图案或内置图案
```

图 3.2.12 执行程序

实验效果：

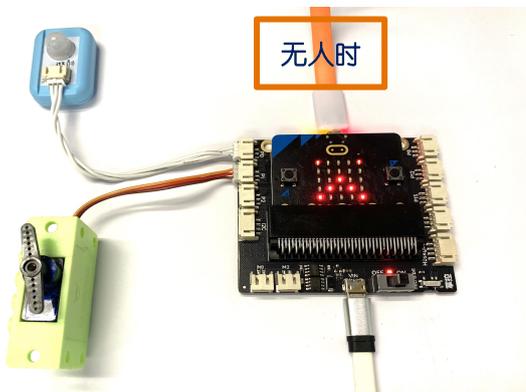


图 3.2.13 实验效果-无人经过

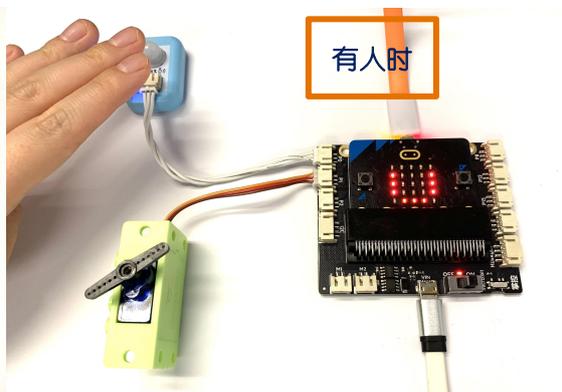
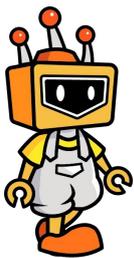


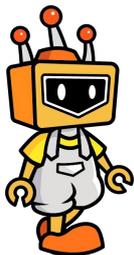
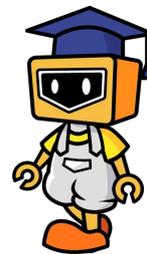
图 3.2.14 实验效果-有人经过

7. 麦克创造屋

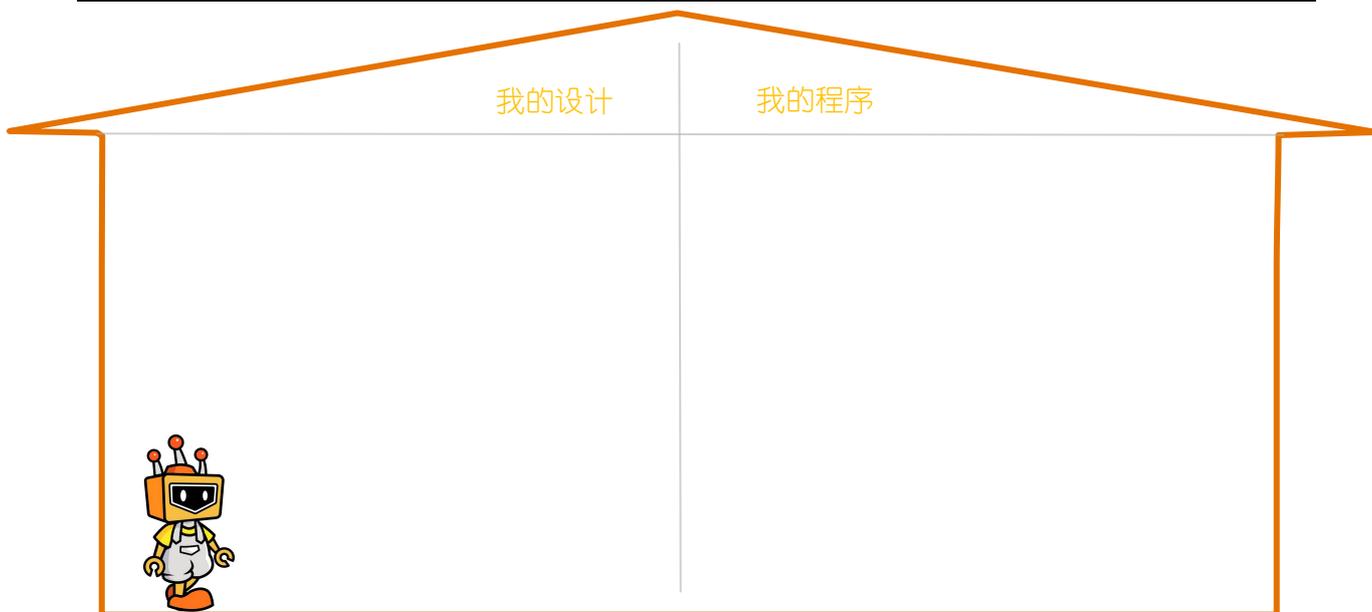


小克老师，这简直太神奇了！和我见到的自动门一模一样！

是的，小麦同学，还记得我说过什么吗？你可以用这些有趣的模块做很多好玩的东西哦！

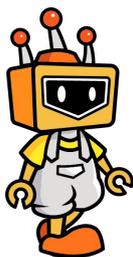


小克老师，我要再想一想，除了自动门，还可以用布兜的东西做出什么好玩的东西！



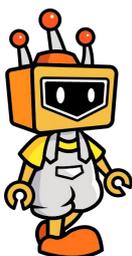
项目三：音乐盒

1. 麦克的故事



小克老师，你知道我除了机器人课之外，最喜欢什么课吗？

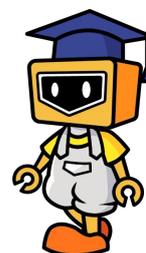
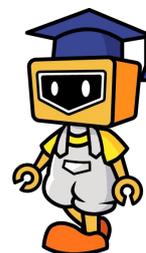
让我猜一猜，应该是音乐课吧？美妙的音符和动听的旋律肯定很吸引人~

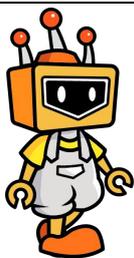


小克老师您猜对了！前几天音乐老师给我们演示了一个音乐盒，非常的漂亮，而且里面的旋律很好听，我也好想有一个自己的音乐盒呀~



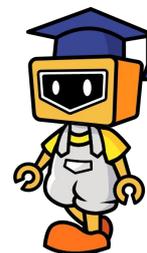
小麦同学，那你为何不自己做一个呢？



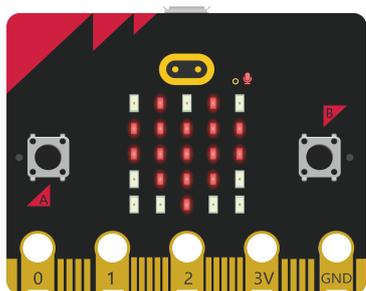


小克老师，我记得您说过，Micro: bit 可以做任何想做的东西，它真的可以做音乐盒吗？

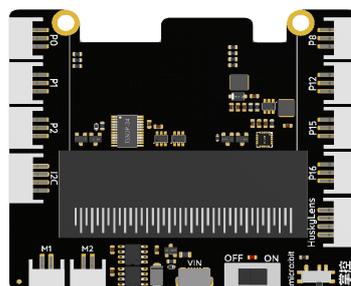
我们试一试就知道啦！



2. 麦克的布兜



1 x micro: bit 主控板



1 x Boson 扩展板

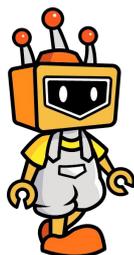


1 x 运动传感器模块



1 x micro USB 连接线

3. 小麦的愿望：播放音乐



我的愿望很简单，想要它播放出美妙的音乐！

4.小克的黑板

STEP1: 用数据线将 micro: bit 主板与电脑连接起来，将运动传感器模块接在扩展板的 P0 号接口。



注意: 本课程基于micro: bit V2。新版本的micro: bit 添加了内置扬声器，我们只需要通过micro: bit 就能够播放声音了。

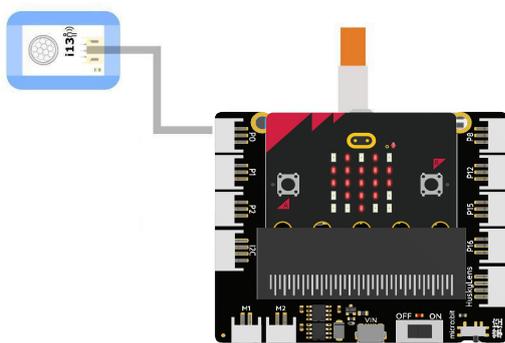


图 3.3.1 接线示意图

STEP2: 编写程序

①将“micro: bit”模块中的音乐模块“接口 P0 播放声音--直到结束”，拖动到脚本区。点击接口，切换为“蜂鸣器(V2)”。播放声音模块中已经有内置的很多旋律可供选择，我们在这里选择的是“DADADADUM”。



图 3.3.2 “播放声音”直到结束指令

②最终程序。

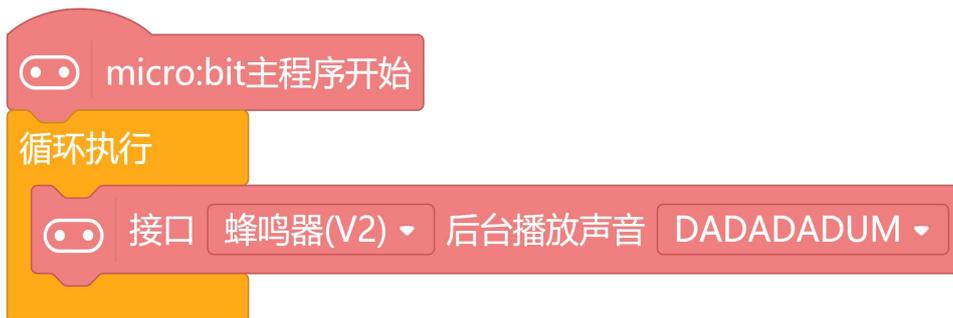
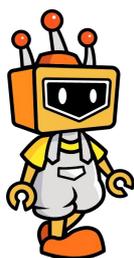
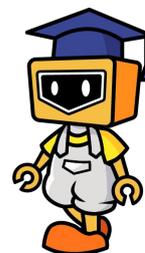


图 3.3.3 执行程序

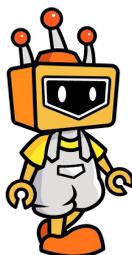


小克老师, 好神奇呀! 小小的板子还有播放音乐的功能!

是的, 不要小瞧它哦~以后还会发现更多有趣的功能呢!



5. 小麦的愿望: 身体感应音乐

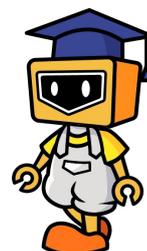


我想让它播放自己编的音乐! 要是当我靠近时播放, 没有人靠近时不播放就更好了!

6. 小克的黑板

- ① 编写一段歌曲《小星星》的音乐。

小麦同学, 那你就快快去小贴士学习一下吧!





先看看《小星星》的简谱是怎样的吧！音调不同，音符不同，需要的发声频率也不同哦！

小星星

调号 1=C 1 1 5 5 6 6 5 — 4 4 3 3 2 2 1 音符

一闪一闪亮晶晶，满天都是小星星，

怎么用 Mind+把简谱呈现出来以便播放好听的音乐呢？
首先要学会把简谱中的音符和字母对应起来哦，见下表：

音符	1	2	3	4	5	6	7
	(do)	(re)	(mi)	(fa)	(sol)	(la)	(si)
字母	C	D	E	F	G	A	B

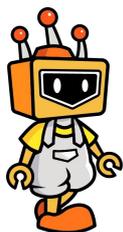
② 将“micro: bit”模块集中的播放旋律模块“接口 P0 播放音符 1 低 C/C3 1 拍”拖至脚本区，这条指令可以通过调节参数“C3”演奏不同的音符，可以通过调节参数“1”调整节拍。（点击音符可以看到音符在钢琴上的对应关系）参照简谱编写音符指令。



图 3.3.5 “播放音符”指令



图 3.3.6 编写音符指令



小麦同学，编写好小星星的曲子之后我们的 micro: bit 已经学会唱歌啦~

小克老师，如何让它实现有人靠近就自动播放，不靠近就不播放呢？

小麦同学，还记得自动门吗？和它的原理一样哦，所以只要增加一个运动传感器就可以实现啦！



③ 运动传感器连接的是 P0 引脚，如果有人经过时 P0 引脚的值为 1，没有人则为 0，因此添加一个已经学过的条件判断语句“如果-那么执行”就可以实现智能音乐盒啦！

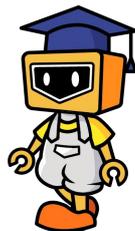
```

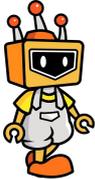
micro:bit主程序开始
循环执行
  如果 读取数字引脚 P0 = 1 那么执行
    接口 蜂鸣器(V2) 播放音符 1 中 C/C4 1/2 拍
    接口 蜂鸣器(V2) 播放音符 1 中 C/C4 1/2 拍
    接口 蜂鸣器(V2) 播放音符 5 中 G/G4 1/2 拍
    接口 蜂鸣器(V2) 播放音符 5 中 G/G4 1/2 拍
    接口 蜂鸣器(V2) 播放音符 6 中 A/A4 1/2 拍
    接口 蜂鸣器(V2) 播放音符 6 中 A/A4 1/2 拍
    接口 蜂鸣器(V2) 播放音符 5 中 G/G4 1/2 拍
  
```

图 3.3.7 执行程序

7. 麦克创造屋

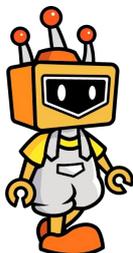
小麦同学，你想设计一个属于自己的音乐盒吗？开动脑筋创造起来吧~



我的设计	我的程序
	

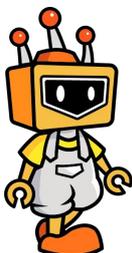
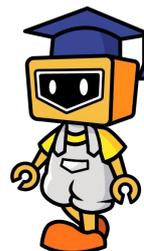
项目四：炫彩灯带

1. 麦克的故事



小克老师，上节课的音乐盒简直太厉害了！不过...

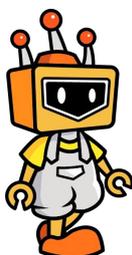
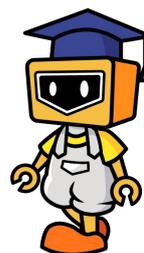
哦？小麦同学，你有什么疑问吗？



小克老师，我想给音乐盒做一个好看的外壳，想要它有炫彩的灯光，这样看起来就更梦幻了！

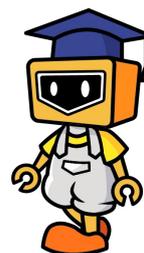


小麦同学，这个想法不错哦，那你想自己做一个吗？

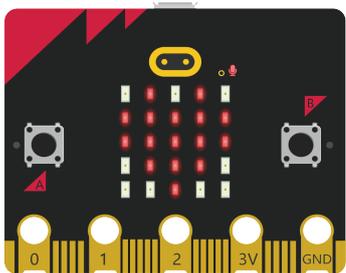


哇！看来 **micro: bit** 真的无所不能呀！小克老师您快教我怎么做吧！

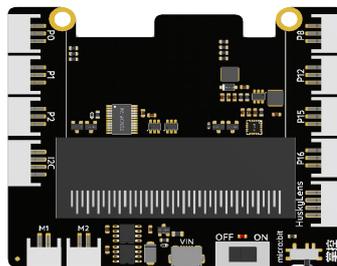
那就让我们开始吧~



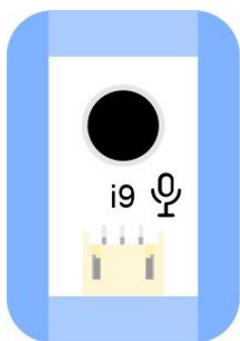
2. 麦克的布兜



1 x micro:bit 主控板



1 x Boson 扩展板



1 x 声音传感器



1 x 彩虹灯带



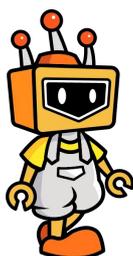
1 x micro USB 连接线



第一次看到彩虹灯带，快来认识一下吧：

彩虹灯带可以通过调节色彩参数调出不同颜色的灯效哦！

3. 小麦的愿望：点亮彩虹灯带



我想点亮全部的彩虹灯，
让它发出七彩的光！

4.小克的黑板

STEP1: 将彩虹灯带接在扩展板的 P0 号接口。

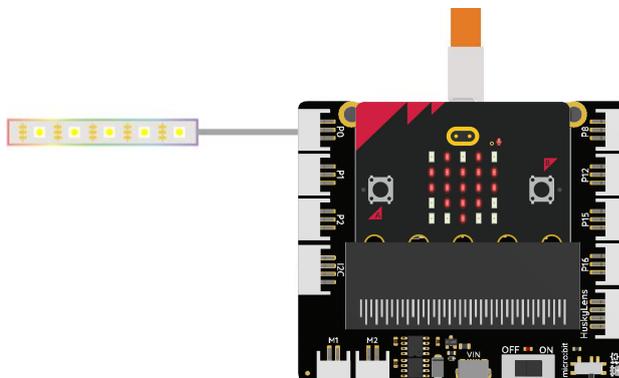


图 3.4.1 接线示意图

STEP2: 增加扩展灯带的指令库。使用灯带需要调用“扩展”功能中的“显示器”模块，选中“Ws2812 RGB 灯”。



图 3.4.2 “扩展”指令



图 3.4.3 添加“Ws2812 RGB 灯”显示器

STEP3: 编写程序

①灯带常用指令：在指令区左侧点击“显示器”模块，所有灯带指令就会显示出来啦。

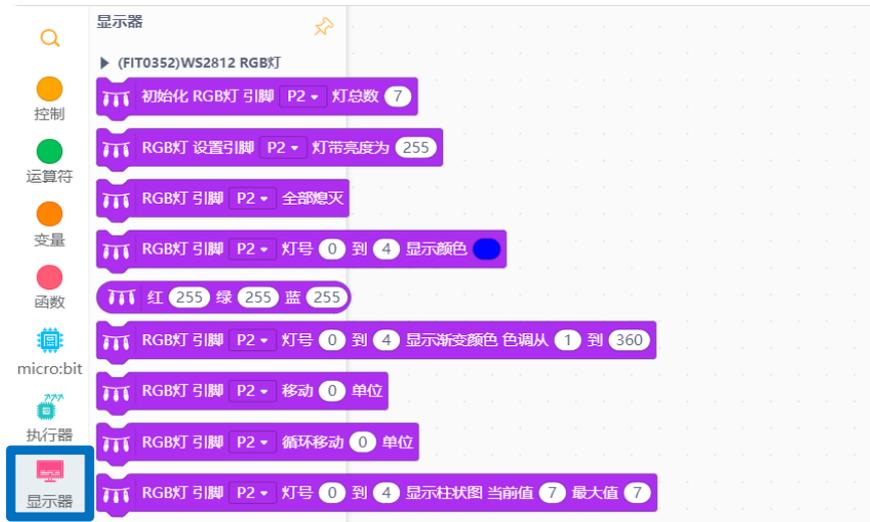
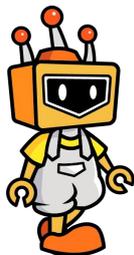
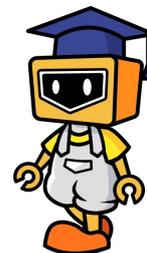


图 3.4.4 “扩展”指令



哇！小克老师，怎么有这么多的指令呀？
我都看晕了！

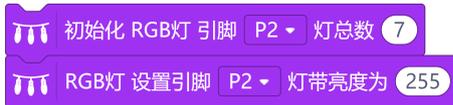
哈哈，小麦同学不用着急哦~
快看看小贴士学习一下常用指令吧！



如何使用常用的彩虹灯带指令呢？

在使用彩虹灯带的过程中，首先需要选定灯带连接 micro: bit 的引脚、灯带 LED 灯的数目及颜色模式，需要调用的指令如下：

①如果你要确定 LED 灯使用个数和亮度，需要调用的指令是：

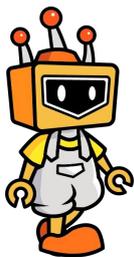


(其中灯总数是实际灯带的灯珠总个数哦，亮度最高是 255)

②如果你想确定彩虹灯的色调范围，需要调用的指令是：

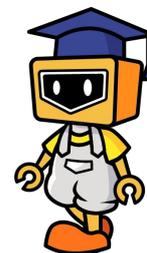


(灯号 0 代表第一个灯，色调表示 360 种不同的颜色)



小克老师，为什么是 0 代表第一个灯，怎么不是 1 代表第一个灯呢？

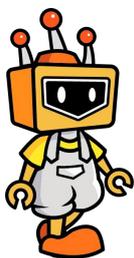
小麦同学，这个问题问的好，你可以这样理解哦：阿拉伯数字是从 0 开始的，所以 0 代表第一个灯~



② 灯带连接引脚为 P0，包含 7 个 LED 小灯，设置彩虹灯效色彩范围 1-360（红色--绿色--蓝色），彩灯就会呈现七彩效果啦。

```
micro:bit 主程序开始
初始化 RGB 灯 引脚 P0 灯总数 7
RGB 灯 设置引脚 P0 灯带亮度为 255
RGB 灯 引脚 P0 灯号 0 到 6 显示渐变颜色 色调从 1 到 360
```

图 3.4.5 执行程序



让我想一想，灯带包含 7 个小灯，从 0 开始到 6 刚好是 7 个数字，所以灯号才是 0~6 呀！

实验效果：

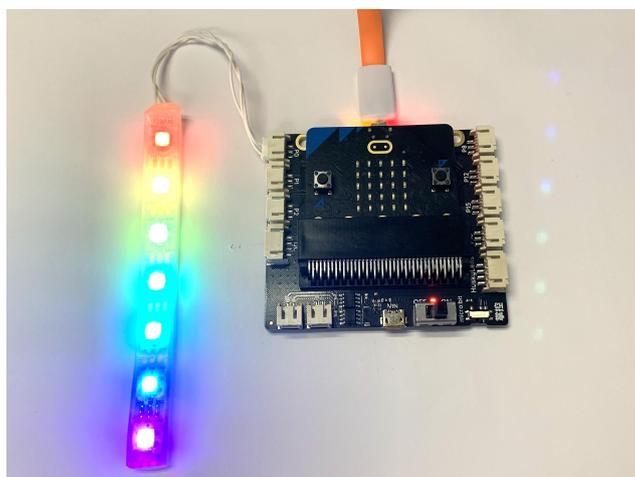
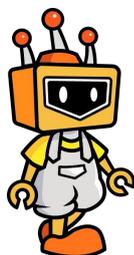


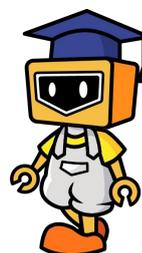
图 3.4.6 实验效果

5.小麦的愿望：逐一点亮彩虹灯

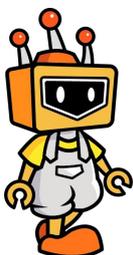


6.小克的黑板

小麦同学，如果你想实现这个效果需要认真的思考，仔细做好笔记哦！

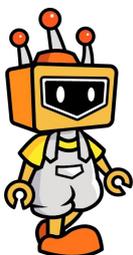
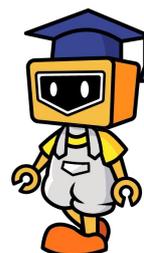


① 如果想要控制彩虹灯带的 LED 小灯每隔一秒点亮一盏，那么就需要运用一个新的东西：**变量**。



小克老师，为什么要使用变量呢？

小麦同学，如果你想要小灯一盏一盏闪动的效果，那么就需要判断亮的灯的数量，如果超过七盏了，就要再重新循环闪烁啦！



哦~原来是这样呀！

需要调用指令的流程为：“**变量**----**新建数字类型变量**”，任意设定变量名（如：亮灯数量），再单击确定。



图 3.4.7 新建变量图



图 3.4.8 设置变量名称



初次新建变量，来认识认识吧？

新建变量之后，在变量指令区会出现常用的“变量+变量名称的模块”和“设置变量的值”的模块：



点击“设置变量的值”的下拉列表可以选择创建过的变量进行赋值。

当你需要使用该变量的时候，直接拖动模块到脚本区就可以进行编辑啦！

②要对小灯的引脚和总灯数进行初始化，并设置亮度。“亮灯数量”（也可以简称为 α ）最初设定为 0（因为最初没有小灯被点亮哦）。



图 3.4.9 初始化引脚和亮灯数

③当“亮灯数量” (α) 小于等于 6 的时候，需要点亮 α 盏灯，等待 1 秒之后让 α 加 1，如果 α 超过 6 的话就要将灯设置为全部熄灭，再重新开始，逻辑图如图 3.4.10。

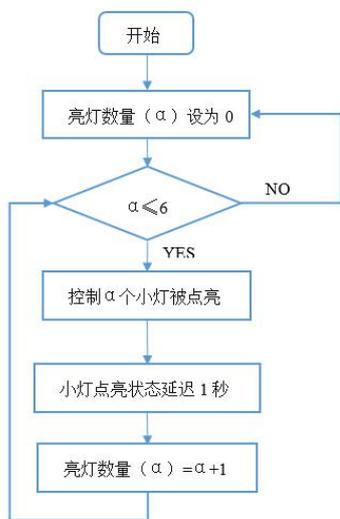
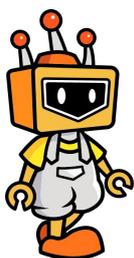


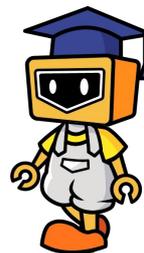
图 3.4.10 逻辑图



小麦同学，流程图要好好思考一下哦~

小克老师，通过流程图分析步骤好清晰呀！

没错哦小麦同学，以后你也可以尝试绘制流程图哦！



④按照流程进行程序的编写，首先加入“如果那么执行-否则”指令和“循环执行”指令。其次向里面添加条件：如果：“亮灯数量” (α) 小于等于 6。



图 3.4.11 部分指令

⑤当“亮灯数量” (α) 小于等于 6 为真，那么执行：点亮 α 盏灯，等待 1 秒之后让 α 加 1，也就是再多亮起一盏灯。

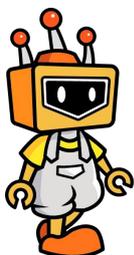
```
micro:bit主程序开始
初始化 RGB灯 引脚 P0 灯总数 7
RGB灯 设置引脚 P0 灯带亮度为 255
设置 亮灯数量 的值为 0
循环执行
  如果 变量 亮灯数量 <= 6 那么执行
    RGB灯 引脚 P0 灯号 0 到 变量 亮灯数量 显示渐变颜色 色调从 1 到 360
    等待 1 秒
    设置 亮灯数量 的值为 变量 亮灯数量 + 1
```

图 3.4.12 部分指令

⑥直到亮灯数量大于 6，在“否则”指令下，将变量“亮灯数量”的值设置成 0，并且让所有小灯熄灭。

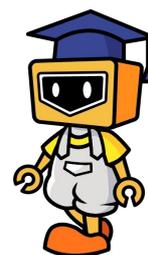
```
micro:bit主程序开始
初始化 RGB灯 引脚 P0 灯总数 7
RGB灯 设置引脚 P0 灯带亮度为 255
设置 亮灯数量 的值为 0
循环执行
  如果 变量 亮灯数量 <= 6 那么执行
    RGB灯 引脚 P0 灯号 0 到 变量 亮灯数量 显示渐变颜色 色调从 1 到 360
    等待 1 秒
    设置 亮灯数量 的值为 变量 亮灯数量 + 1
  否则
    设置 亮灯数量 的值为 0
    RGB灯 引脚 P0 全部熄灭
```

图 3.4.13 执行程序

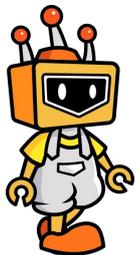


小克老师，这个程序操作起来好像有点复杂呢！

是的哦，小麦同学，慢慢你会发现程序会越来越复杂，但是做出来的东西也会越来越好玩的！所以快快动用你的小脑瓜熟练操作起来吧！



7.小麦的愿望：声音控制彩虹灯带



哎？可不可以每次发出声音都能点亮一盏灯，直到七盏小灯全部点亮之后再重新开始玩呢？

8.小克的黑板

STEP1: 彩灯连接引脚保持不变，将声音传感器连接到 P1 引脚。

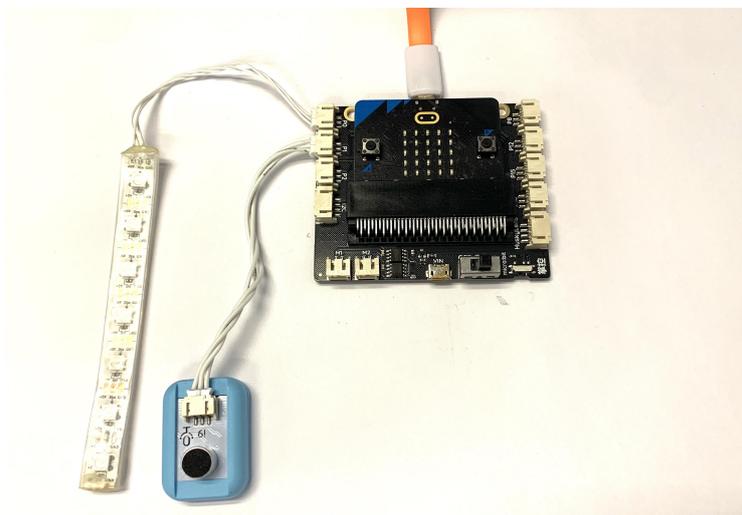
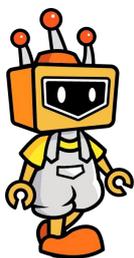
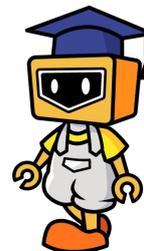


图 3.4.14 连线示意图



小克老师，声音传感器在做电子蜡烛的时候用过欸！

小麦同学很聪明哦，没错，这里同样要使用前面讲过的声音传感器模块，也就是使用“micro: bit”里面的“模拟读取引脚”哦~



STEP2: 单击“micro: bit”模块，选择“读取模拟引脚 P0”指令，拖动到脚本区并将 P0 改为 P1。



图 3.4.15 “读取模拟引脚”指令

STEP3: 当声音传感器检测到声音>40 且“亮灯数量 α ”不超过 6 时，控制灯带点亮 α 盏 LED 灯，之后持续不断检测声音是否超过 40。

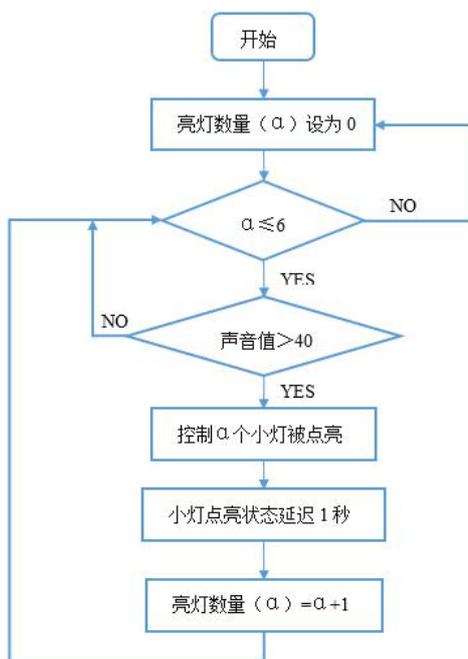
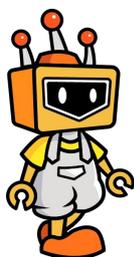
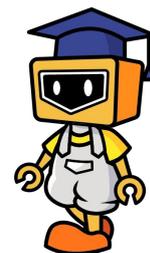


图 3.4.16 逻辑图



小克老师，我知道了，和我的上一个愿望相比，这个愿望就是多了一个声音控制的条件~

小麦同学很聪明哦~没错就是在上一个的基础上增加声音的条件就可以了哦！清晰的画出流程图是成功的关键步骤哦！



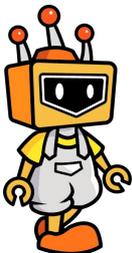
STEP4: 编写程序

在上一个愿望的执行程序的基础上，只要再增加一个声音强度的判断条件就可以了哦，也就是说，在“亮灯数量 α ” ≤ 6 时，如果模拟引脚 P1 的值 > 40 ，那么执行点亮 α 盏灯，等待 1 秒之后让 α 加 1，并持续不断检测声音是否超过 40。直到“亮灯数量 α ” > 6 ，灯全部熄灭，重新开始。

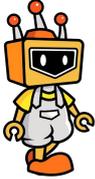
```
micro:bit主程序开始
初始化 RGB灯 引脚 P0 灯总数 7
RGB灯 设置引脚 P0 灯带亮度为 255
设置 亮灯数量 的值为 0
循环执行
  如果 变量 亮灯数量 <= 6 那么执行
    如果 读取模拟引脚 P1 >= 40 那么执行
      RGB灯 引脚 P0 灯号 0 到 变量 亮灯数量 显示渐变颜色 色调从 1 到 360
      等待 1 秒
      设置 亮灯数量 的值为 变量 亮灯数量 + 1
    否则
      设置 亮灯数量 的值为 0
      RGB灯 引脚 P0 全部熄灭
      等待 1 秒
```

图 3.4.17 执行程序

9. 麦克创造屋



哇！这节课的程序好难呀！看来我要好好的练习一下了，不如回去做一个好玩的声控灯吧！谁和我一起设计一下呢？

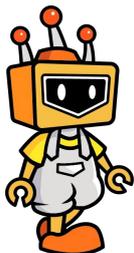
我的设计	我的程序
	

第四话：大展拳脚

前面这么多愿望都实现了，是不是已经和 micro: bit 成为好朋友了呢？现在到大展拳脚的时候啦！快使用 micro: bit 主控板和各种传感器，制作出炫酷好玩的作品吧！

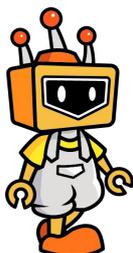
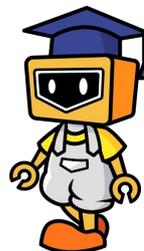
项目一：自平衡仪

1. 麦克的故事



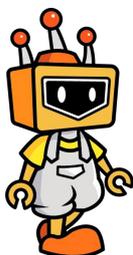
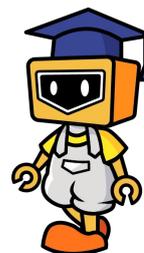
小克老师，这次出去玩我发现一个特别神奇的现象！

小麦同学，什么现象呢？说来听听？



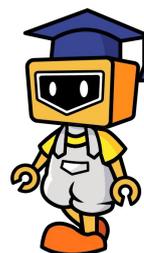
小克老师，我本来把手机横放在手里，但是当我手向左转的时候，屏幕里的页面就神奇的向右转动，这是为什么呢？

小麦同学很善于观察哦~这是因为手机里有自平衡装置，为了方便观看，屏幕里的页面要始终保持水平向上，所以根据开发者的设置，它会感应手机的方向，如果你把手机向左旋转，屏幕里的页面就向右转啦！

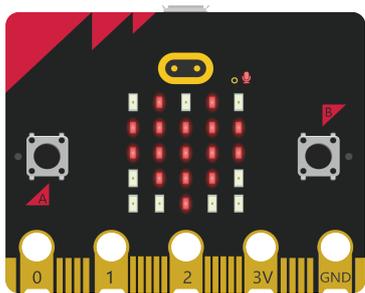


哦~原来是这么神奇呀！小克老师，我们可以制作一个这样的自平衡装置吗？

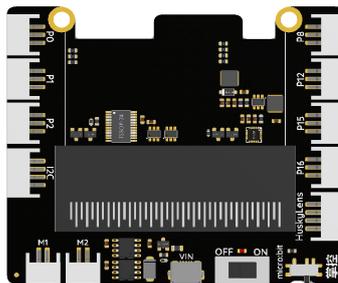
可以哦！来一起探索 micro: bit 的这个神奇功能吧！



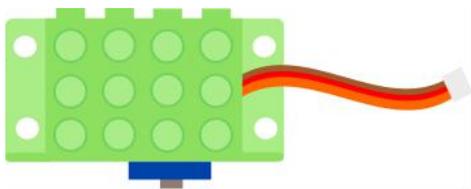
2. 麦克的布兜



1 x micro: bit 主控板



1 x Boson 扩展板

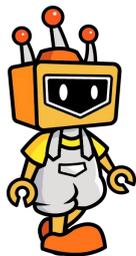


1 x 舵机 (伺服机构)



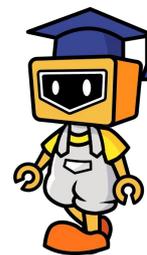
1 x micro USB 连接线

3. 小麦的愿望

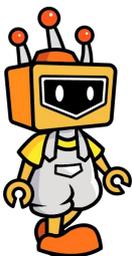


我想要无论怎么旋转, 小舵机上的指针永远垂直于地面指向上方~

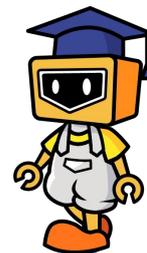
小麦同学, 你知道重力加速度吗?



小克老师, 我听说过重力加速度, 制作自平衡仪和重力加速度有什么联系吗?



理解重力加速度有助于理解自平衡仪的原理哦~ 快去小贴士学习一下吧~





什么是重力呢？

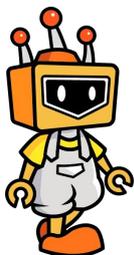
我们都生活在地球上，而地球是有吸引力的~由于它的吸引而使物体受到的力就是重力。

什么是重力加速度呢？

由重力产生的加速度就叫做重力加速度，也叫自由落体加速度，用 g 表示。

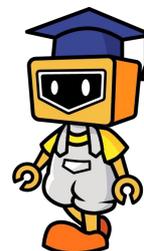
使用重力加速度可以做些什么呢？

micro: bit 自带加速度计，它能实时检测 micro: bit 的姿态，通过重力加速度的感应和程序设计就可以控制舵机指针一直指向正上方啦！



小克老师，这么看来，我们是要将舵机和板子连在一起，然后通过板子的重力加速度的感应来控制舵机的旋转方向喽？

小麦同学理解的很对哦！不过重力加速度还有不同的方向呢！去下面的小贴士里学学我们今天要用哪个方向的重力加速度吧！



Micro: bit 加速度计：

Micro: bit 可以检测 X,Y,Z 三个方向的重力加速度，其中 X 沿着左右方向，Y 沿着前后方向，Z 垂直于板，沿着上下方向，如图 4.1.1。

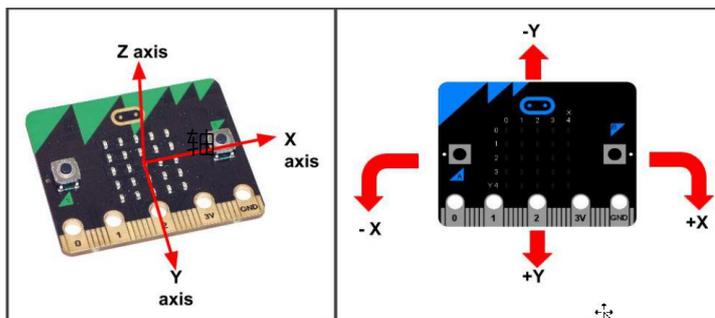


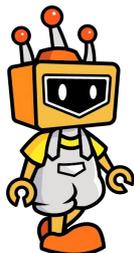
图 4.1.1 三个方向

图 4.1.2 数值变化



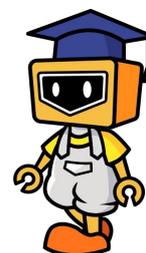
当主板平放在桌面上时，micro: bit 只受到竖直方向上的重力（Z 轴），所以 X,Y 都接近于 0。

而当板子产生左右倾斜时，X 轴的数值会发生变化。向左侧倾斜则 X 为负值，反之为正值，如图 4.1.2。



小克老师，所以如果想满足我的愿望，就要计算 X 轴的加速度是吗？

小麦同学，你说的对哦~



4.小克的黑板

STEP1: 将舵机与扩展板的 P0 号接口相连接。

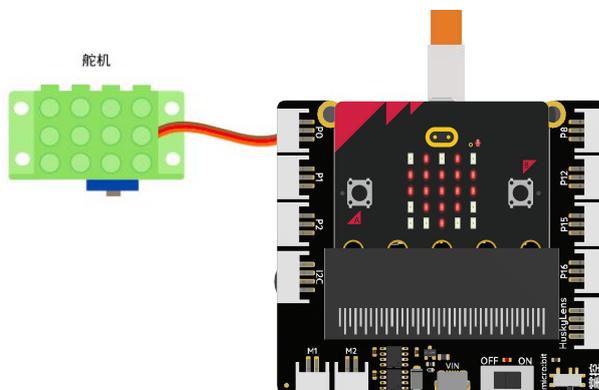


图 4.1.3 连接示意图

STEP2: 编写程序

①通过“读取加速度的值”模块实现获取 X,Y,Z 轴的加速度数值。



可以通过下拉列表选择 X,Y,Z 轴哦~



图 4.1.4 读取加速度的值

②加速度传感器的数值范围在-1023~1023 之间变化，可以通过“显示”模块来查询变化范围哦。



图 4.1.5 显示加速度的值

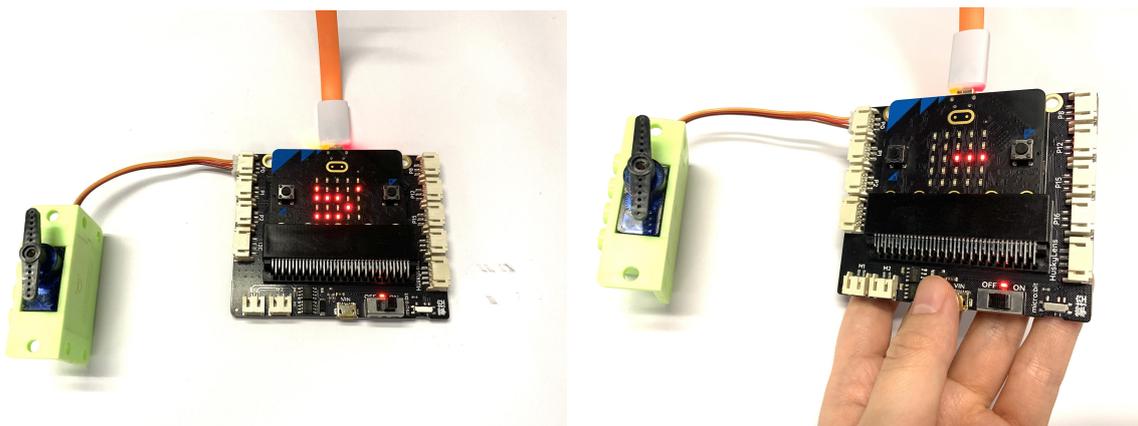
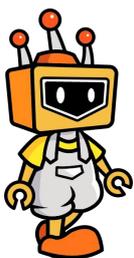


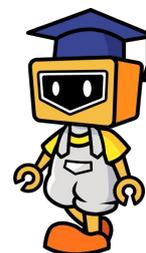
图 4.1.6 实现效果

③当 micro: bit 板子向左倾斜时，X 方向会产生一个负值，这个时候就需要舵机向右转动来抵消这个角度啦！



小克老师，那 micro: bit 板子的加速度与舵机旋转的角度是怎么对应的呢？

小麦同学，这个问题问得好，继续往下看吧！



这里需要用到“映射模块”，将加速度 X 轴的值 (-1023~1023)，映射到舵机的转动角度 (0~180)。映射之后，当 micro: bit 板子放在水平桌面时，舵机的角度就为90度啦，对应关系如表 5.1.1。

表 5.1.1

舵机角度	0 度	90 度	180 度
micro: bit 加速度	-1023	0	1023

在“运算符”中找到“映射模块”。

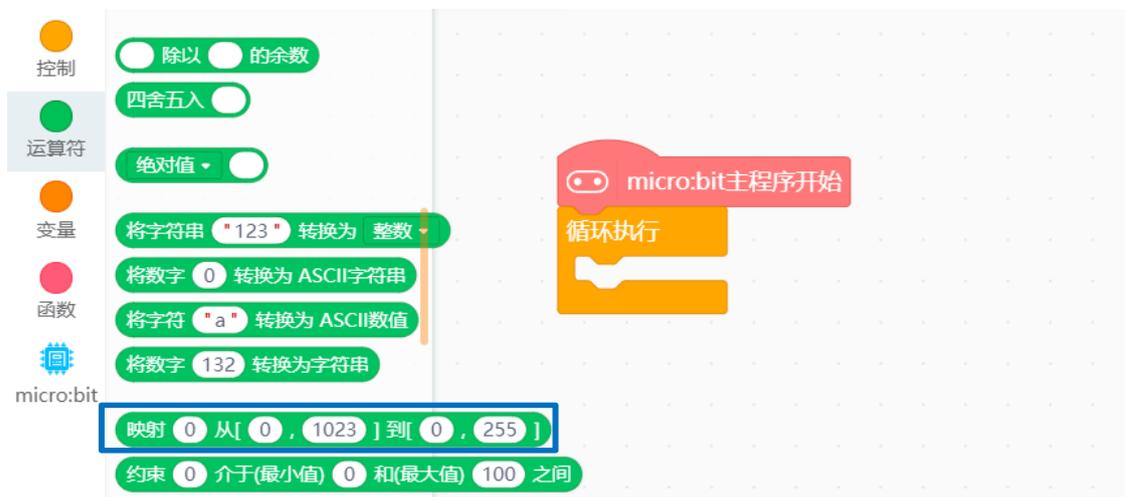


图 4.1.7 映射模块

④在“扩展”中找到“执行器”模块，再找到“舵机模块”。在左侧指令区点击“舵机模块”，可以看到如下指令。



图 4.1.8 舵机模块

⑤选中“设置舵机角度”放入“循环执行”中，将角度设定为“加速度”经过“映射模块”后的值：



图 4.1.9 执行程序

⑥最后进行结构制作，用双面胶将舵机和扩展板固定起来，使两者保持相对静止。

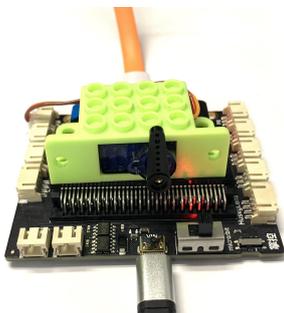


图 4.1.10 结构固定

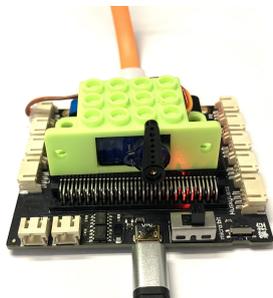


图 4.1.11 水平静止

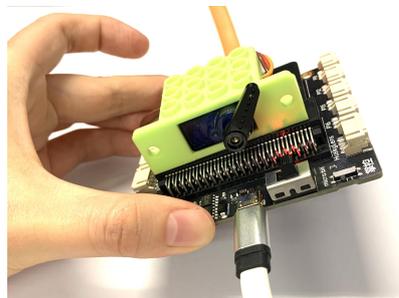
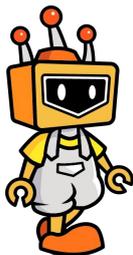


图 4.1.12 向左旋转

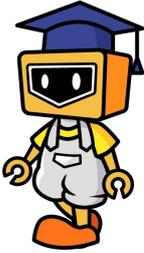
5. 麦克发明室



小克老师，自平衡仪简直太神奇了！我看到 Mind+里面有“指南针模块”，难道 micro: bit 还能实现指南针功能吗？



没错，小麦同学，这一次你自己尝试使用 micro: bit 自带的电子罗盘功能制作一个指南针吧！



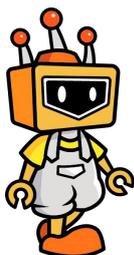
由于 micro: bit 自带的舵机模块的旋转范围是 $0^{\circ}\sim 180^{\circ}$ ，而指南针需要 360° 旋转，因此你可以不使用舵机，而采用在 micro: bit LED 点阵上“显示”的“指南针朝向度数”来判断方位哦！仔细阅读 LED 点阵上的提示操作，你可以做到的哦！

我的设计	我的程序

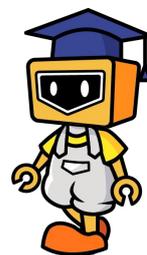


项目二：DJ 演奏台

1. 麦克的故事



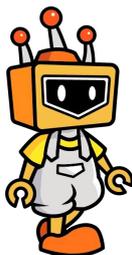
小麦同学，圣诞节快到了，你准备表演什么节目呢？



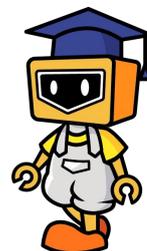
小克老师，今年我想搞点新花样，我想要像 DJ 一样呈现出炫彩的灯光，演奏出动感的音符。



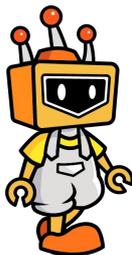
小麦同学，这个想法不错哦，那你想自己做一个吗？



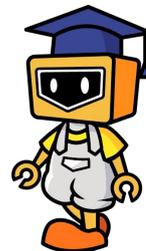
嗯！我要用 `micro: bit` 做一个像模像样的 DJ 演奏台，小克老师你愿意帮助我吗？



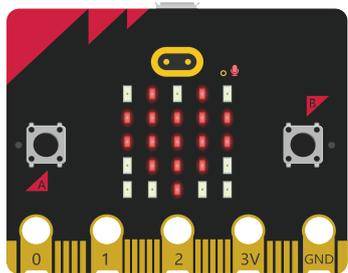
好的，小克老师愿意助你一臂之力！



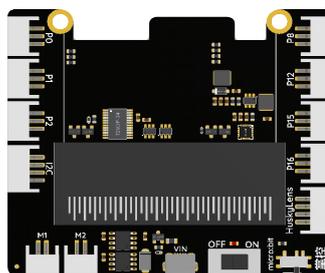
太好了小克老师，那我们就开始吧~



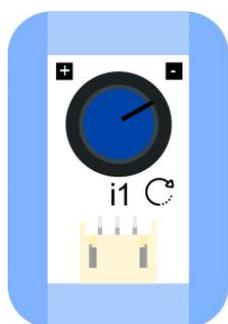
2. 麦克的布兜



1 × micro:bit 主控板



1 × Boson 扩展板



1 × 旋钮模块



1 × 彩虹灯带



1 × micro USB 连接线

3. 小麦的愿望



我想要通过调整旋钮的转动角度，实现演奏不同的音乐，同时灯带闪烁不一样的色彩！

当用手拨动旋钮时，旋钮对应的引脚读出的模拟值在 0~1023 之间变化，我们可以将它分隔成 8 个小范围，对应音乐中的一个八度“Do~Do'”，并且对应小灯的变化，对应图见表 5.2.1。

表 5.2.1

旋钮 (P1) 的输入值	声音	灯带
0~128	Do	1 颗灯亮
128~256	Re	2 颗灯亮
256~384	Mi	3 颗灯亮
384~512	Fa	4 颗灯亮
512~640	So	5 颗灯亮
640~768	La	6 颗灯亮
768~996	Xi	7 颗灯亮
996~1023	Do'	7 颗灯亮, 均为蓝色

4.小克的黑板

STEP1: 将旋钮模块接在扩展板的 P1 号接口；将彩虹灯带接在扩展板的 P2 号接口。

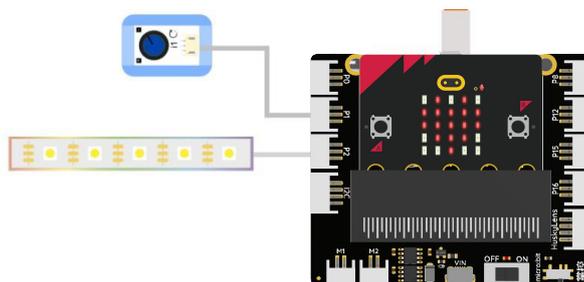


图 4.2.1 连线示意图

STEP2: 想要完成愿望，首先要绘制出 DJ 演奏台的实现逻辑图哦。

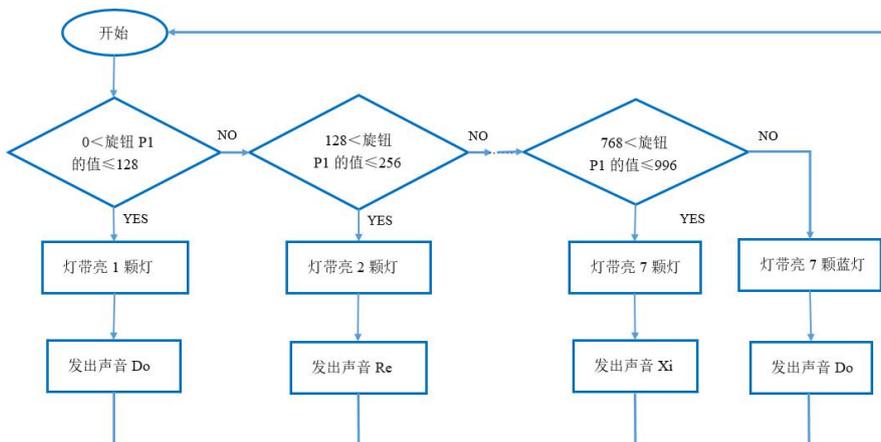


图 4.2.2 逻辑图

STEP3: 编写程序

①以“ $0 < \text{旋钮 P1} < 128$ ”为例，通过“运算符”模块中的“与”模块设置旋钮 P1 值的控制范围，将范围放入“如果...那么执行”的条件中，在“那么执行”下嵌入需要的灯带效果与音效，并加入“等待...秒”。下图是第 1 个音符的控制程序，后 6 个控制程序以此类推。

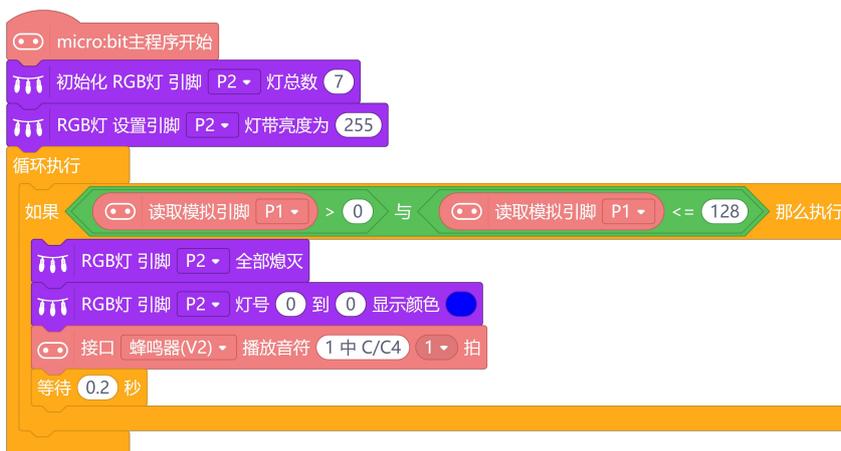
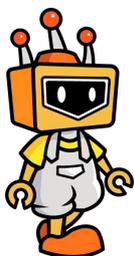


图 4.2.3 程序示例 1

②由于只有 7 盏灯，在第 8 个音符中，可以通过不同的灯带表现形式来表达“Do”，程序如下图。

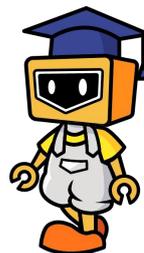


图 4.2.4 程序示例 2



小克老师，我发现这个程序其实并不复杂，只是有些繁琐~

没错的小麦同学，编写程序的时候可以巧妙运用复制功能哦~



```
micro:bit主程序开始
初始化 RGB灯 引脚 P2 灯总数 7
RGB灯 设置引脚 P2 灯带亮度为 255

循环执行
  如果 读取模拟引脚 P1 > 0 与 读取模拟引脚 P1 <= 128 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 0 显示颜色 蓝色
    接口 蜂鸣器(V2) 播放音符 1 中 C/C4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 128 与 读取模拟引脚 P1 <= 256 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 1 显示颜色 红色
    接口 蜂鸣器(V2) 播放音符 2 中 D/D4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 256 与 读取模拟引脚 P1 <= 384 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 2 显示颜色 橙色
    接口 蜂鸣器(V2) 播放音符 3 中 E/E4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 384 与 读取模拟引脚 P1 <= 512 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 3 显示颜色 黄色
    接口 蜂鸣器(V2) 播放音符 4 中 F/F4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 512 与 读取模拟引脚 P1 <= 640 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 4 显示颜色 绿色
    接口 蜂鸣器(V2) 播放音符 5 中 G/G4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 640 与 读取模拟引脚 P1 <= 768 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 5 显示颜色 青色
    接口 蜂鸣器(V2) 播放音符 6 中 A/A4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 768 与 读取模拟引脚 P1 <= 886 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 6 显示颜色 紫色
    接口 蜂鸣器(V2) 播放音符 7 中 B/B4 1 拍
    等待 0.2 秒

  如果 读取模拟引脚 P1 > 886 与 读取模拟引脚 P1 <= 1023 那么执行
    RGB灯 引脚 P2 全部熄灭
    RGB灯 引脚 P2 灯号 0 到 6 显示颜色 深蓝色
    接口 蜂鸣器(V2) 播放音符 1 高 C/C5 1 拍
    等待 0.2 秒
```

图 4.2.5 执行程序

③为了可以方便进行演奏，需要给 DJ 演奏台做一个便于操作的外观哦。给旋钮做个手柄，方便进行操作，将旋钮固定好之后，将每个音的范围标注出来吧！

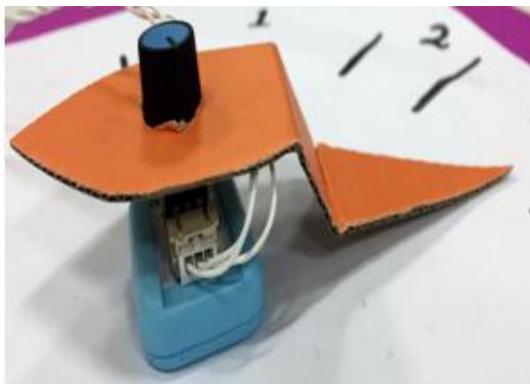


图 4.2.7 结构改变 1

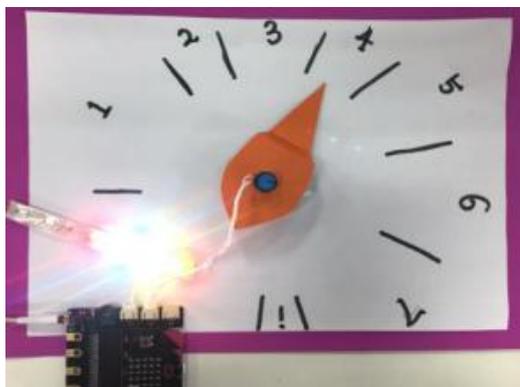
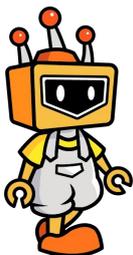


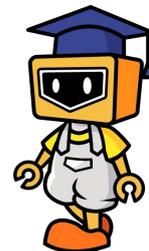
图 4.2.8 结构改变 2

5. 麦克发明室



小克老师，我真的好开心，能做一个自己的 DJ 演奏台~

小麦同学，想不想让你的演奏台更完善呢？把 **micro: bit** 主板的 **5×5LED** 点阵也利用起来，试试能不能当演奏出不同的音节时，点阵对应显示该音节？快尝试制作一下吧！



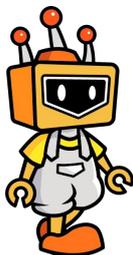
我的设计

我的程序



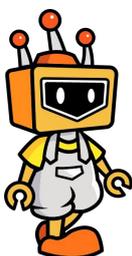
项目三：可移动门铃

1. 麦克的故事



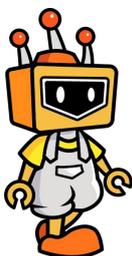
小克老师，我需要你的帮助，昨天有朋友来家里找我玩，但是我在卧室听音乐，没有听到他的敲门声，所以他生气的走了~

小麦同学，看来你需要好好和同学解释一下了！



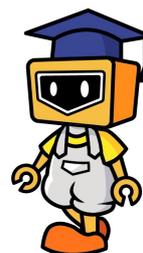
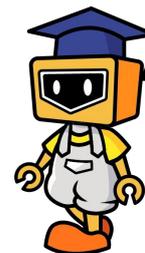
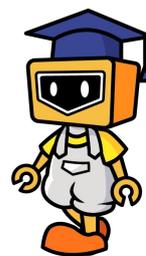
是的小克老师，可是我担心下次还会出现这种情况，该怎么解决呢？

小麦同学，你想在卧室就听到是否有人在门外敲门？可以制作一个可以远程遥控的门铃哦~

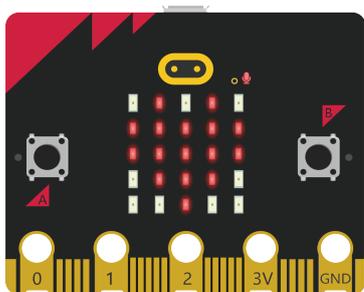


真的吗小克老师，如果可以制作一个远程门铃就太好了！这样同学来找我的时候，我就不用担心听不到门铃声音啦！

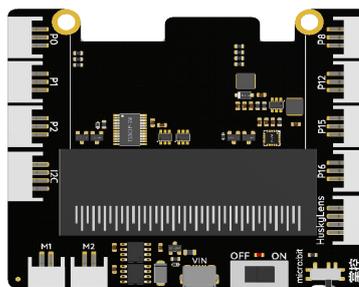
好，那我们开始制作吧！



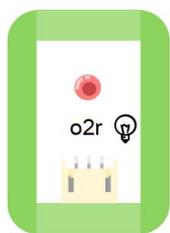
2. 麦克的布兜



2x micro:bit 主控板



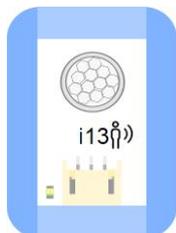
2x Boson 扩展板



1 × LED 灯模块



1 × 按钮模块

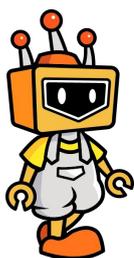


1 × 运动传感器模块

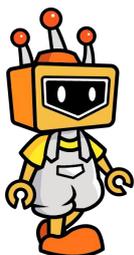


2 × micro USB 连接线

3.小麦的愿望

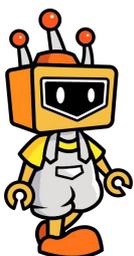
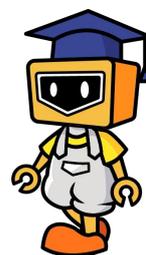


我想在卧室就知道门外是否有人按门铃~



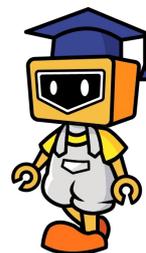
小克老师，为什么这次要使用两块板呢？

小麦同学，我们要使用两块板来模拟远程门铃哦~其中一块作为门外的发射端，也就相当于门铃装置；另一块放在房间里，作为接收端，这样你就能在卧室知道门外的情况啦！



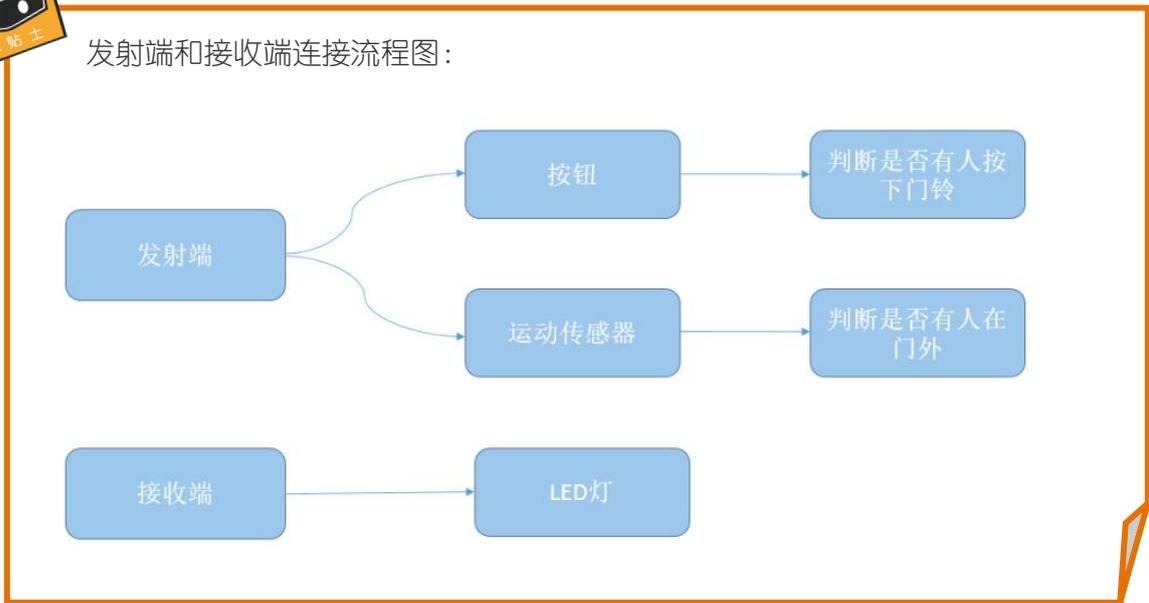
小克老师，那发射端和接收端要连接什么呢？

小麦同学，这一次我们要运用闪烁的 LED、呼吸灯和自动门项目中学习到的知识，快去小贴士看看怎么发射端和接收端需要连些什么吧！





发射端和接收端连接流程图：



4.小克的黑板

STEP1: 发射端硬件连接，按钮连接P0端口，运动传感器连接P1端口：

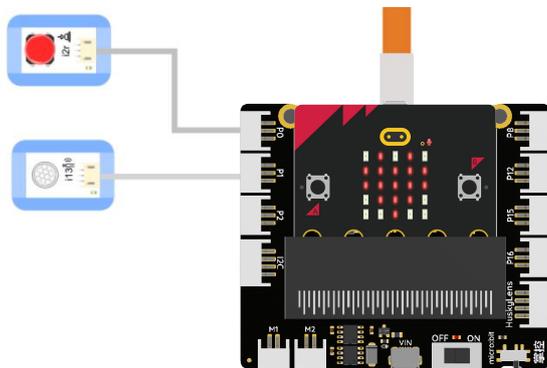


图 4.3.1 发射端连接示意图

接收端硬件连接，LED灯连接P1端口：

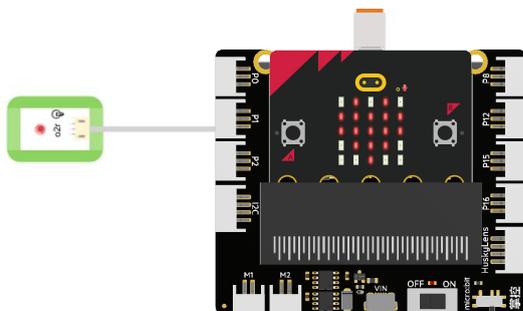
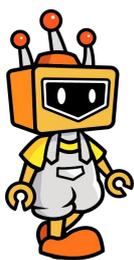


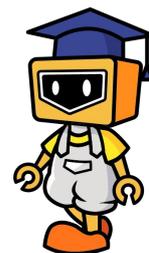
图 4.3.2 接收端连接示意图

STEP2: 了解无线传输模块



小克老师, 那怎么将门外是否有人的消息远程发射出去呢?

小麦同学, 这就需要学习一个新的知识:
无线传输模块啦! micro:bit之间可以进行**无线通信**。当发射端接收到信号之后, 通过**无线传输模块**将信号传到接收端, 所以需要**分别对发射端和接收端进行程序编写**哦!



常用的无线传输模块:

- 设定信号组:
- 无线发射信号:
- 无线接收信号:

注意: 接收端需要识别接收到的信号, 从而做出对应的动作。这里发射的信号可以是数字或者文字, 但最终都会自动转化为字符串的格式发送。

STEP3: 设置发射端程序

①首先对发射端进行程序编辑, 编写逻辑图。

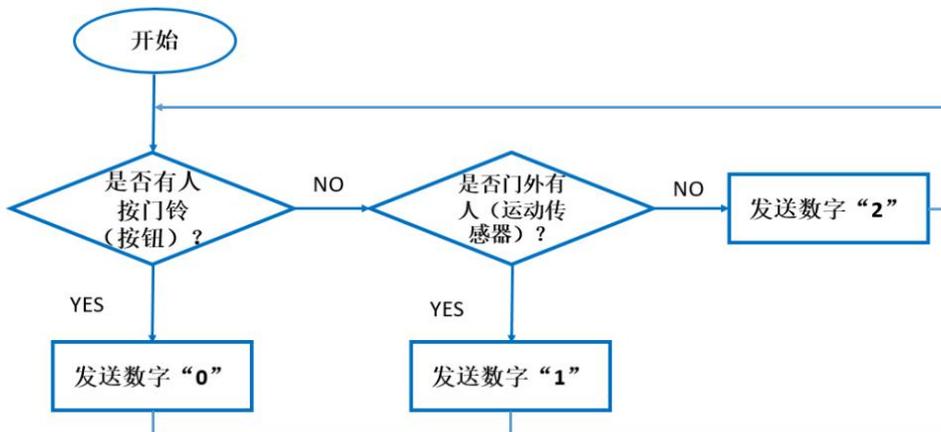
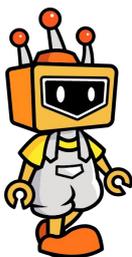
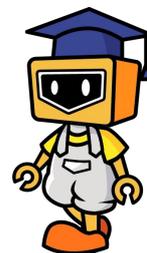


图 4.3.3 发射端逻辑图



小克老师，这里怎么设置无线连接呢？

小麦同学问得好，这里需要注意：在编写程序之前记得“打开无线通讯”模块，并且要“设置无线频道”哦~



②根据逻辑图编写**发射端**完整程序。

```
micro:bit主程序开始
打开无线通信
设置无线频道为 7
循环执行
  如果 读取数字引脚 P0 = 1 那么执行
    通过无线发送 "0"
  如果 读取数字引脚 P0 = 0 与 读取数字引脚 P1 = 1 那么执行
    通过无线发送 "1"
  如果 读取数字引脚 P0 = 0 与 读取数字引脚 P1 = 0 那么执行
    通过无线发送 "2"
```

图 4.3.4 发射端执行程序

STEP4: 设置接收端程序

①对接收端端进行程序编辑，编程逻辑图。

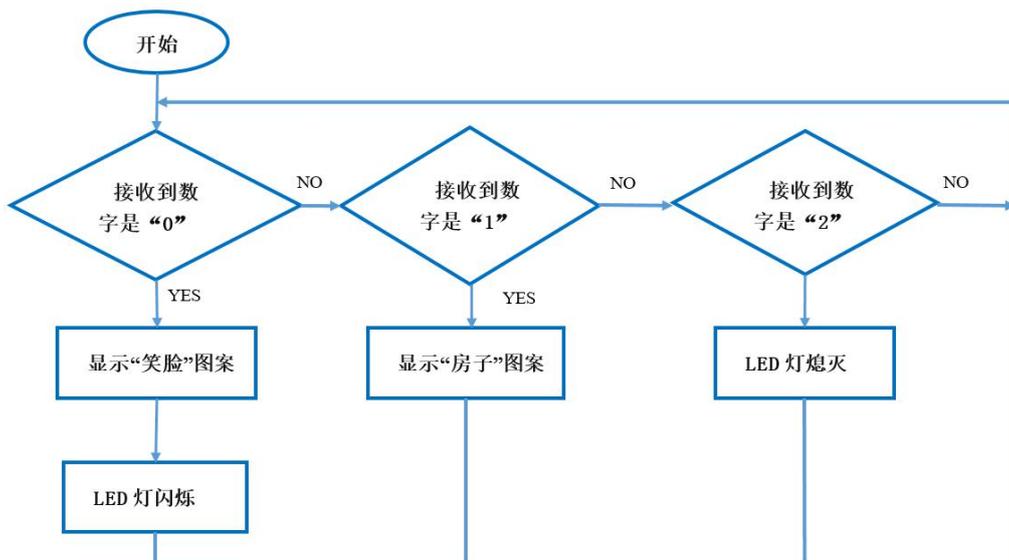
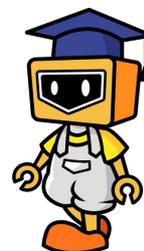


图 4.3.5 接收端逻辑图

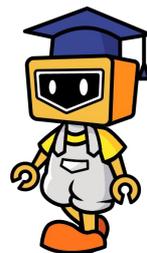
小麦同学，这里同样需要注意：在编写程序之前记得“打开无线通讯”模块并“设置无线频道”，并在“当无线接收到数据”时执行程序哦！



②根据逻辑图编写接收端完整程序。

图 4.3.6 接收端执行程序

小麦同学，我们读取信号的时候要记得输加上英文输入模式下的"0"，"1"，"2"哦~（不要忘记因为是读取字符串，所以要加双引号哟）



实验效果：

(1) 当门外有人经过但是没有按门铃时：

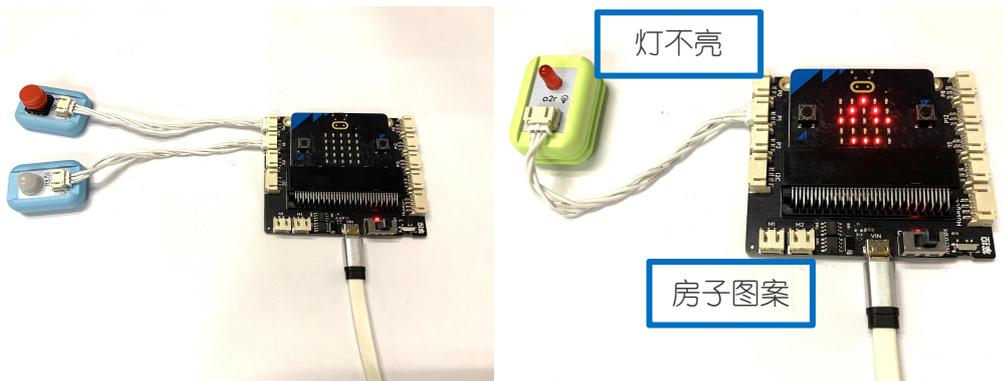


图 4.3.7 实验效果 1

(2) 当门外有人按门铃时

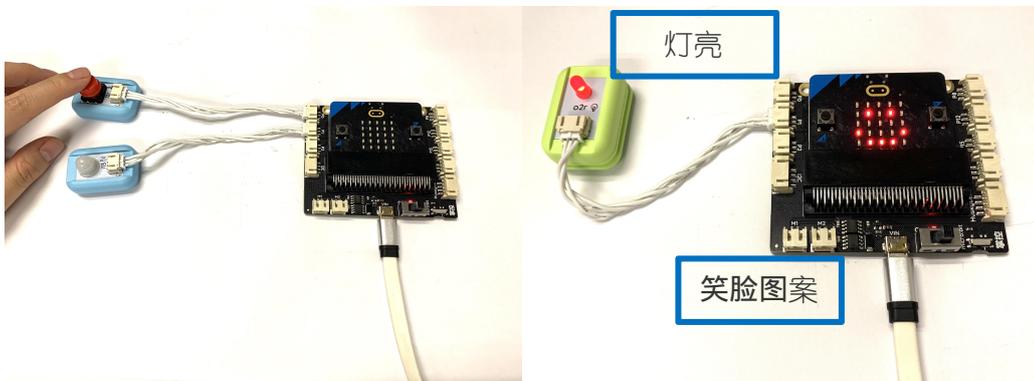
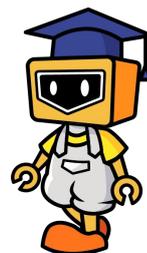


图 4.3.8 实验效果 2

5. 麦克发明室

小麦同学，在门外按门铃的人一定很焦急，
请为门外的门铃设定专属铃声吧~

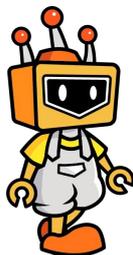


我的设计	我的程序



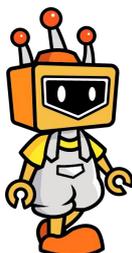
项目四：拆弹游戏

1. 麦克的故事

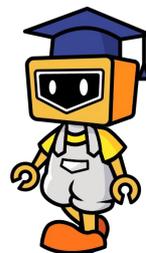


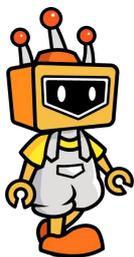
小克老师，我的好朋友这周要来家里做客，我想和他们一起玩好玩的游戏~

哦？那你要和他们玩什么游戏呢？

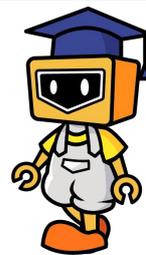


我们可以一起玩遥控汽车、遥控飞机，可是我想玩一些新鲜的、有趣的、酷炫的玩具~





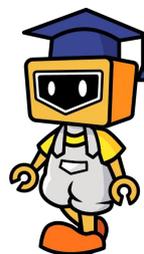
小麦同学，你是想自己制作一个有趣的玩具吗？



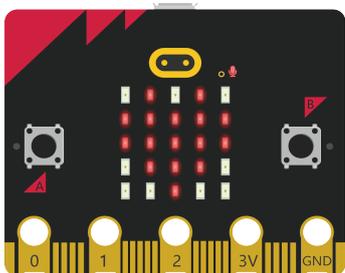
是的，我想用布兜里的工具制作一个考验反应能力的拆弹游戏！小克老师你愿意帮助我吗？



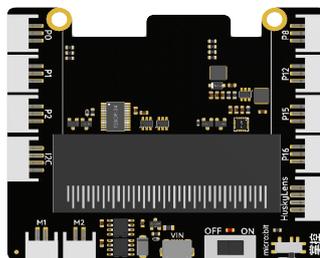
好的小麦同学，那我们研究研究如何制作吧！



2. 麦克的布兜



1 × micro: bit 主控板



1 × Boson 扩展板

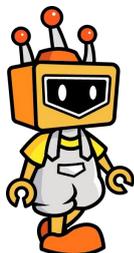


1 × 彩虹灯带



1 × micro USB 连接线

3.小麦的愿望



我想要做一个炸弹游戏，启动游戏之后，灯带开始进入倒计时模式，灯全部熄灭也就是倒计时结束，炸弹会被引爆~

如果想在炸弹引爆之前解锁，需要把 **micro: bit** 向 LED 点阵提示的方向倾斜~

如果在 灯带灯熄灭之前完成了 20 个方向的倾斜，那么就是拆弹成功啦！



这样的拆弹装置由以下三个部分组成：

1. 动作识别：用来识别 **micro: bit** 板的倾斜方向是否与 LED 点阵指示的方向一致。
2. 彩灯倒计时：为 灯带设置倒计时的效果，每隔 10s 亮灯数量减 1。
3. 计分系统：在 灯带全部熄灭之前完成 20 个动作则拆弹成功；若 灯带全部熄灭仍然没有完成 20 个动作，则游戏结束。

4.小克的黑板

STEP1：将彩虹灯带接在扩展板的 P0 号接口。

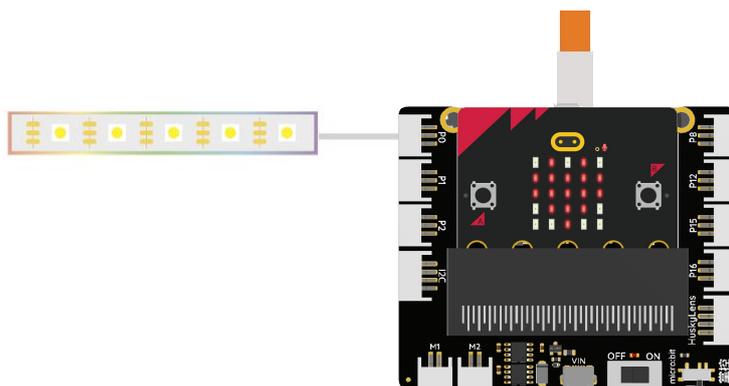


图 4.4.1 连线示意图

STEP2：动作识别部分

1.相关指令学习

首先我们需要让 micro: bit 板随机产生并显示一个方向的箭头，如果我们的动作与该方向一致，那么再随机产生一个方向箭头；如果我们的动作与该方向不一致，那么一直保持当前的箭头方向，程序的流程图大致如下。

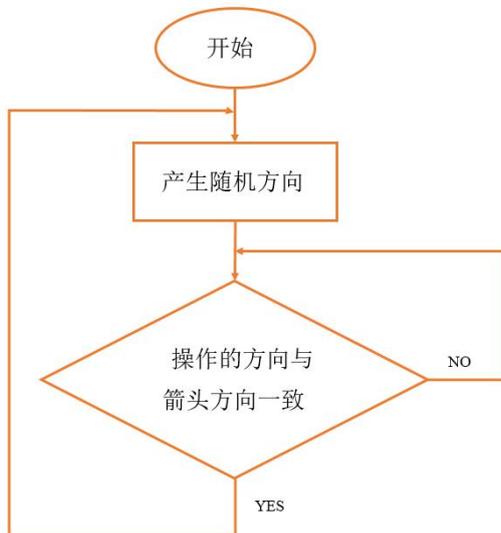
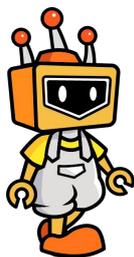
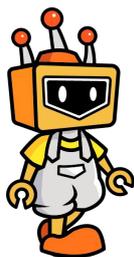
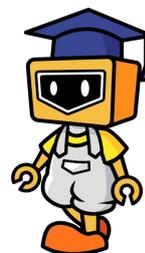


图 4.4.2 动作识别部分逻辑图



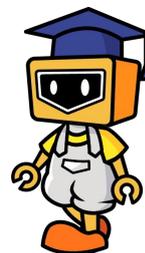
小克老师，如何产生随机的方向呢？

这里需要使用“选取随机数”模块哦



小克老师，那我们怎么把随机数和方向对应起来呢？

小麦同学提的问题很好哦~这里还需要一个新知识，就是“构建函数”，快去小贴士学习一下吧~





选取随机数模块：

通过运算符中的“选取随机数”模块来随机选取指定范围中的数字，比如上下左右有 4 个方向，我们就可以指定随机数从 0 到 3 之间随机选取啦！

在 0 和 3 之间取随机数

构建函数模块：

通过函数中的“自定义模块”可以构建函数，构建函数之后，如果需要使用到该函数，直接将指令拖到对应位置就好啦！

比如我们现在需要将“0, 1, 2, 3”四个数字和“上下左右”四个方向对应起来，那么就可以构建“产生随机方向”的函数，在构建之后就会出现“产生随机方向”指令啦！



获取当前姿态模块：

可以通过“当前姿态为”指令获取当前 micro: bit 板的姿态：



2.程序编写

①设置变量“目标方向”，它表示 LED 灯显示的方向，需要将“目标方向”的值设置为随机数 0~3。



图 4.4.3 设置目标方向的值



如果你忘记了如何设置变量，去“项目四-点亮彩虹灯带”复习一下

②建立一个函数“产生随机方向”，使“0123”4个数字分别对应 LED 点阵上不同方向的箭头。



图 4.4.4 “产生随机方向”函数

③设置变量“操作方向”，它表示 micro: bit 板的当前的倾斜方向，并通过“当前姿态为”指令将“操作方向”与“0123”4个数字对应起来。



图 4.4.5 获取当前姿态指令

④设置变量“积分”，通过“如果-那么执行”指令进行条件判断，如果操作方向与目标方向一致，那么继续产生随机方向，同时积分+1。



图 4.4.6 条件判断程序

⑤这个部分的完整程序为：

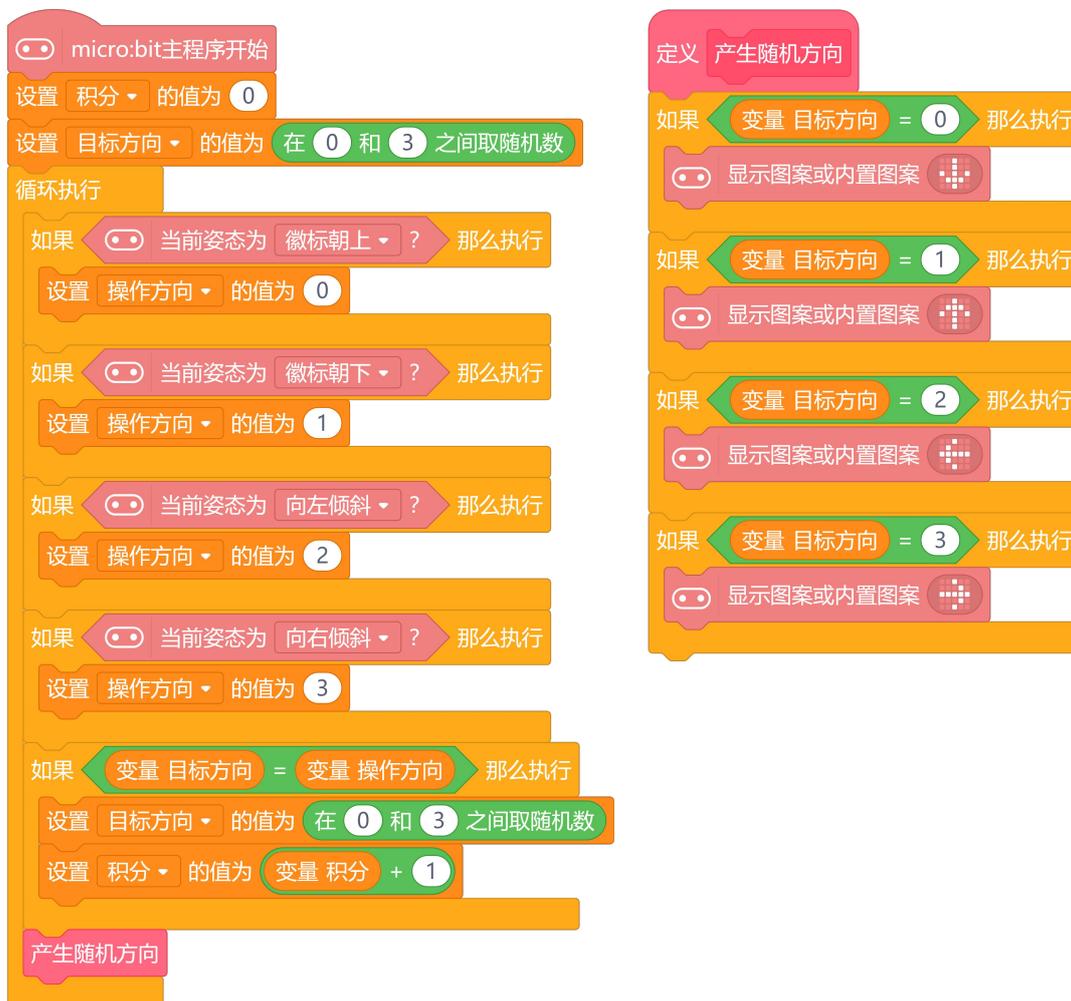


图 4.4.7 动作识别部分完整程序

STEP3：彩灯倒计时部分

1.逻辑图绘制

随着时间的推移，led 灯珠将会以稳定的速度相继熄灭。所以反应速度越快，小灯剩余数量越多。

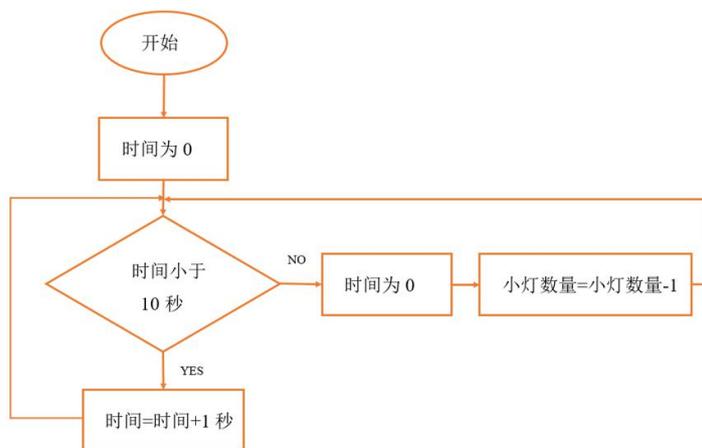


图 4.4.8 彩灯倒计时部分逻辑图

2.程序编写

①构建变量“时间”和“小灯数量”，并将“小灯数量”设置为 7（也就是 7 盏灯都亮起），将“时间”设置为 0，通过“如果-否则”实现逻辑图的要求。



注意：因为 micro: bit 在处理数据时会消耗掉一定的时间，所以倒计时模块实际运行时走得慢一些哦~



图 4.4.9 “如果-那么执行”指令

②在倒计时的同时，不要忘记设置小灯的亮灯程序哦：



图 4.4.10 LED 灯显示指令

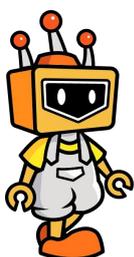
③前两部分的完整程序为：

```
micro:bit 主程序开始
设置 积分 的值为 0
设置 目标方向 的值为 在 0 和 3 之间取随机数
设置 小灯数量 的值为 7
设置 时间 的值为 0

循环执行
  如果 当前姿态为 微标朝上 ? 那么执行
    设置 操作方向 的值为 0
  如果 当前姿态为 微标朝下 ? 那么执行
    设置 操作方向 的值为 1
  如果 当前姿态为 向左倾斜 ? 那么执行
    设置 操作方向 的值为 2
  如果 当前姿态为 向右倾斜 ? 那么执行
    设置 操作方向 的值为 3
  如果 变量 目标方向 = 变量 操作方向 那么执行
    设置 目标方向 的值为 在 0 和 3 之间取随机数
    设置 积分 的值为 变量 积分 + 1
  产生随机方向
  如果 变量 时间 < 10 那么执行
    设置 时间 的值为 变量 时间 + 1
    等待 1 秒
  否则
    设置 时间 的值为 0
    设置 小灯数量 的值为 变量 小灯数量 - 1

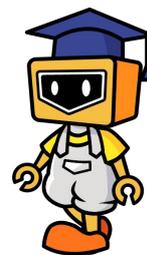
初始化 RGB 灯 引脚 P0 灯总数 7
RGB 灯 设置引脚 P0 灯带亮度为 255
RGB 灯 引脚 P0 全部熄灭
RGB 灯 引脚 P0 灯号 0 到 变量 小灯数量 显示颜色
```

图 4.4.11 前两个部分完整程序



小克老师，10 秒的时间有点长，我们想加快时间和节奏！

没问题的小麦同学，如果想倒计时的时间短，那就试着将 10 秒缩小到 5 秒或者 1 秒吧~



STEP4: 计分部分

1.逻辑图绘制

如果每次操作方向和目标方向一致，则积一分，如果积满 20 分，亮灯的数量还大于 0，则挑战成功，即拆弹成功，最终得分为 LED 灯的剩余个数；否则，拆弹失败。

流程图大致如下：

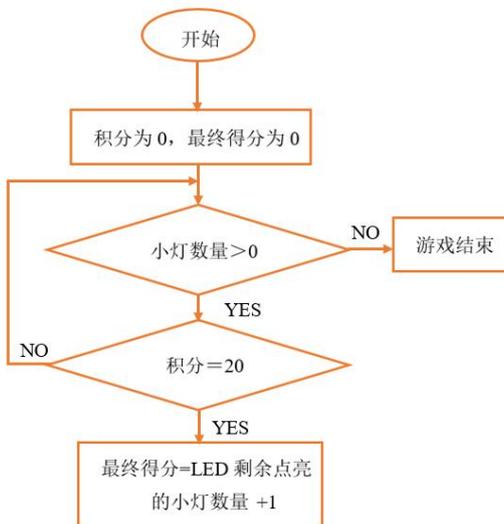


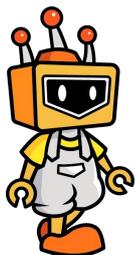
图 4.4.12 计分部分逻辑图

2.程序编写

①构建变量“最终得分”，将“最终得分”设置为 0，通过“如果-那么执行”实现逻辑图的要求。

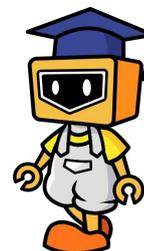
```
if (变量 小灯数量 > 0) then execute
if (变量 积分 = 20) then execute
set (最终得分) to (变量 小灯数量 + 1)
say (显示文字 变量 最终得分) for 5 seconds
wait 5 seconds
```

图 4.4.13 部分程序



小克老师，为什么最终得分要等于小灯数量+1呢？

小麦同学，还记得吗？小灯是从 0 开始计算的，所以实际亮灯的数量比程序中小灯的数量值大 1 哦~

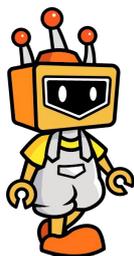


②炸弹游戏完成程序如下：



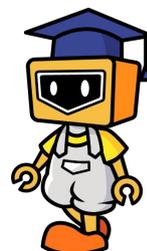
图 4.4.14 完整执行程序

5. 麦克发明室



小克老师，拆弹游戏简直太有趣了！我和小伙伴们玩的不亦乐乎！

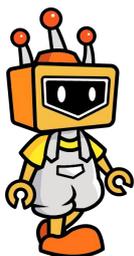
小麦同学，如果你想要它更完整更有趣，可以加一个游戏结束的标志哦，快去和小伙伴探索一下如何实施吧！



我的设计

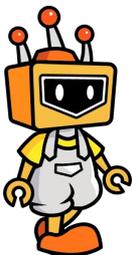
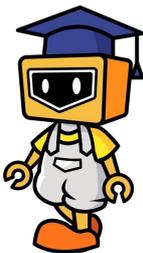
我的程序





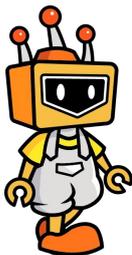
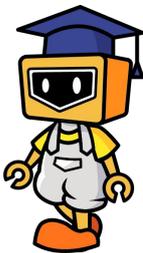
小克老师，拆弹游戏简直太有趣了！micro: bit 的功能真强大！

小麦同学，其实 micro: bit 和 Mind+还有很多很多功能，就等着你慢慢去挖掘啦！
小克老师希望你的 micro: bit 之旅不会止步，用你的奇思妙想，玩出更多新颖有创意的作品。



小克老师，感谢您一直以来的帮助，我会一直保持着创造的热情继续探索好玩的游戏的！
可是如果以后我有关于 micro: bit 和 Mind+ 的问题要去哪里找到您呢？

小麦同学，如果以后你有什么疑问或者又有好的点子愿意和小克老师分享的话，可以去 **DF 创客社区**：
www.dfrobot.com.cn 里找到我哦！那里还有很多和你兴趣相投的小伙伴呢！希望可以在论坛里看到有关你成长的点点滴滴！



哇！太棒了小克老师，希望我能碰到更多有趣的同学，我们论坛再见！