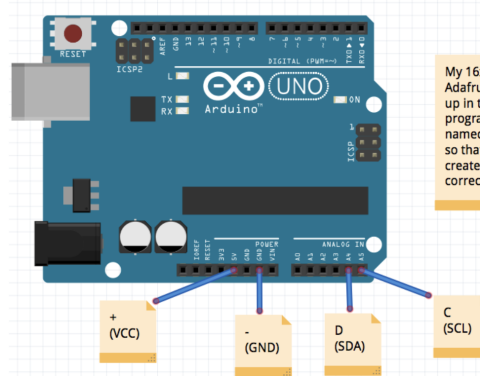
### Bouncing Ball

2016220205027 张梓豪

1. 让小球在16\*8矩阵屏幕中（实际屏幕排列方式为8\*16）无限反弹。
2. 部件连接图：



1. 使用元件：Arduino Uno开发板，16\*8 LED矩阵，以及数据线等



1. 搭建环境：mac操作系统，arduino IDE（官方），C语言
2. 库函数：

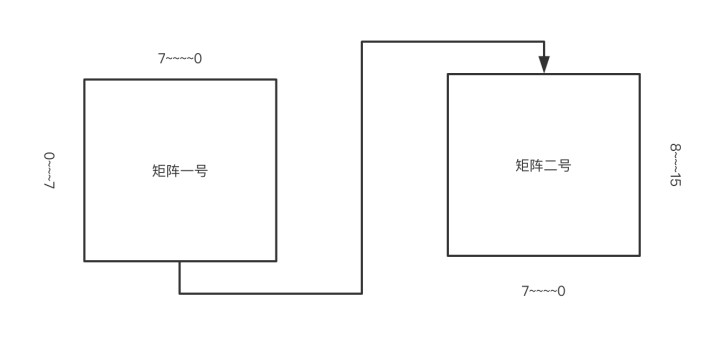
#include <Wire.h>

#include <Adafruit\_GFX.h> // Adafruit GFX Library

#include "Adafruit\_LEDBackpack.h" //Adafruit LED Backpack Library

1. 原理及其实现代码：

硬件注意点：LED矩阵排列模式为纵向，即16\*8。如下图所示：（数字为矩阵坐标）



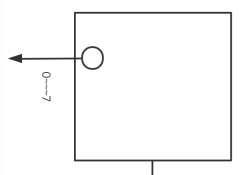
图一

即：两个矩阵排列并非左右连通（最终结果要求为左右连通），而是上下连通，但是元件物理结构为左右连通。

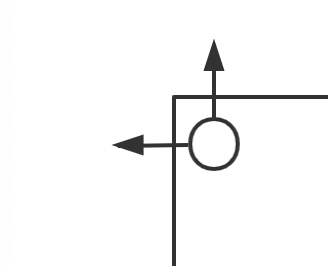
代码逻辑：tell：0表示方向向右，1表示左。till：0表示方向向上，1表示向下。

1. 首先考虑小球移动方向分解，分解为左右方向和上下方向。但是由于元件并非左右连通，如果想要实现8\*16模式，则需要在交接边修改坐标。
2. 综上所述，从每个矩阵各个边框开始考虑。

2.1矩阵一号（来自图一）左边框：在小球撞击边界点时，可以确定小球水平方向一定向左，竖直方向单独考虑。



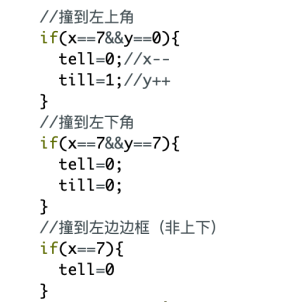
2.1.1左边框上边界点：可以确定这一点的竖直方向为上。那么撞击了边界后，水平和竖直方向均会改变。



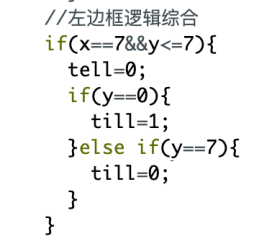
2.1.2左边框下边界点：同理与上边界点，水平和竖直方向均会改变。

2.1.3左边框常规点（非上下边界点）：那么只有撞击到边界的方向会改变，即水平方向改变。

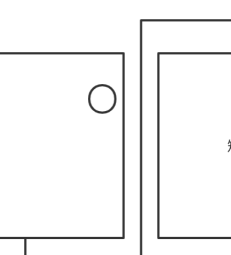
代码：



简化代码：

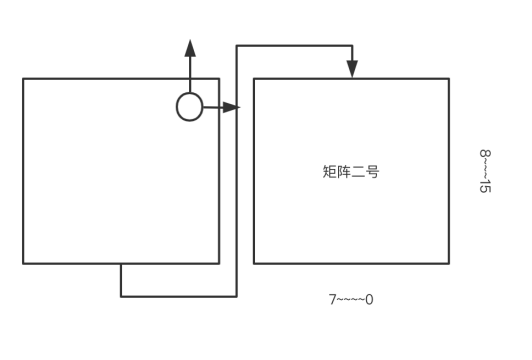
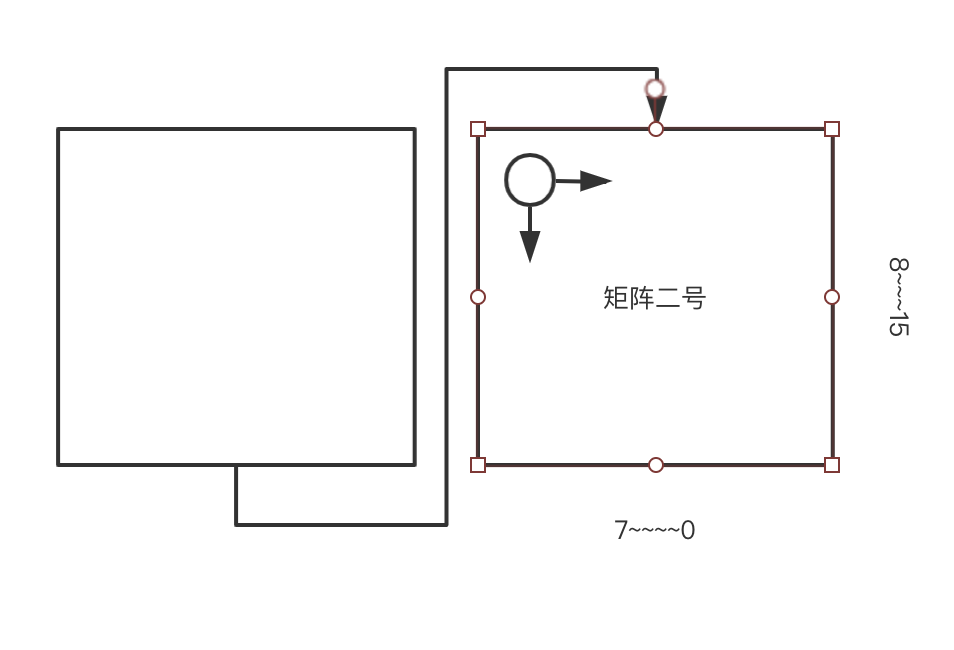


2.2矩阵一号右边框：矩阵一号的右边框并非整体的右边框，而是两个矩阵的交界处，



2.2.1小球到达右边框水平方向为右：

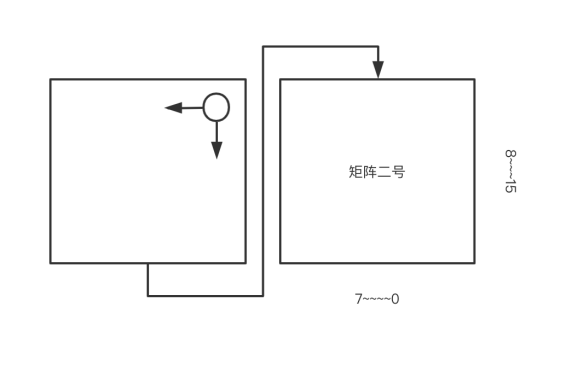
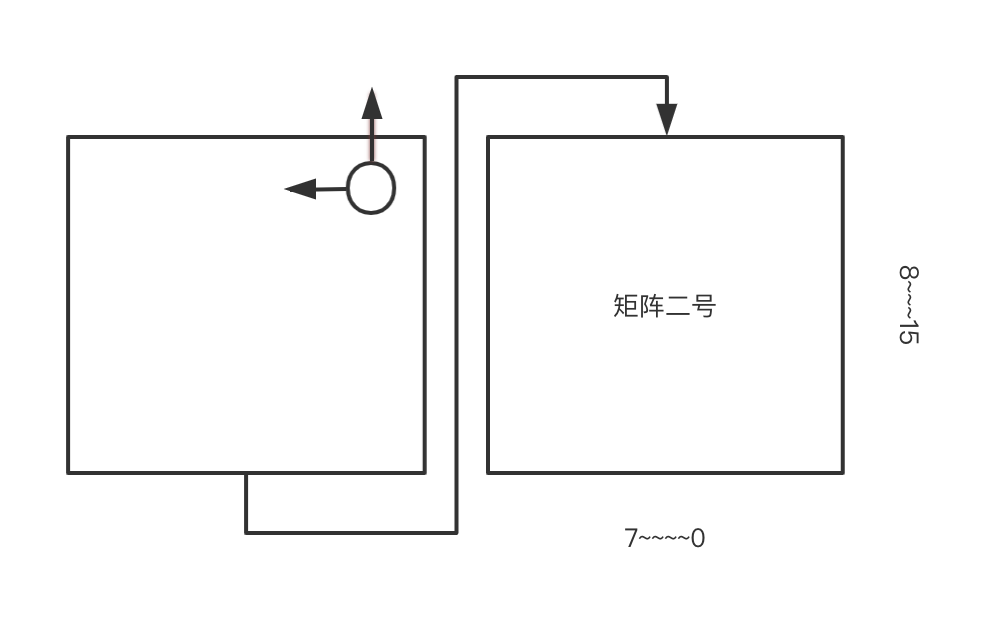
2.2.1.1上下边界点：更改上下方向，水平方向不变。纵坐标+8（移动到下一个矩阵坐标系中），横坐标修改为最大横坐标。下图为上边界点两个状态的转换。

2.2.1.2.右边框非边界点：修改坐标即可（因为没有碰触逻辑边界不用修改方向）

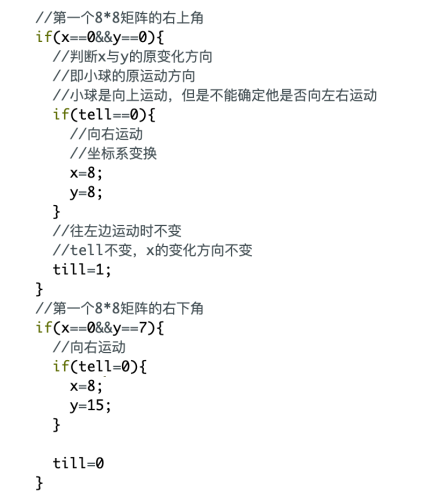
2.2.2小球到达右边框方向为左：

2.2.2.1上下边界点：修改水平方向，竖直方向不变。以下从左至右为移动方向。

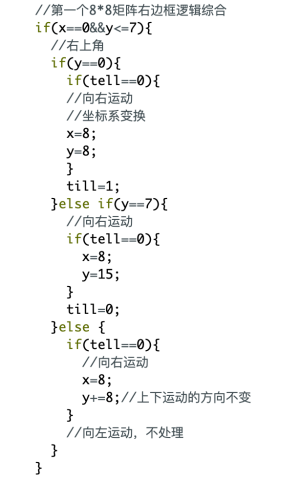


2.2.2.2非边界点：无需修改。

代码逻辑：



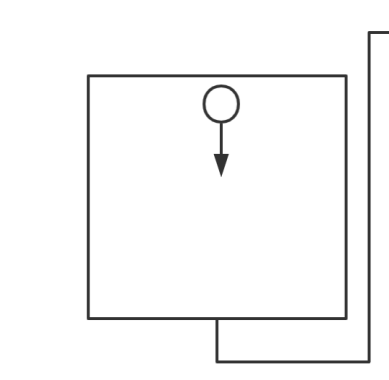
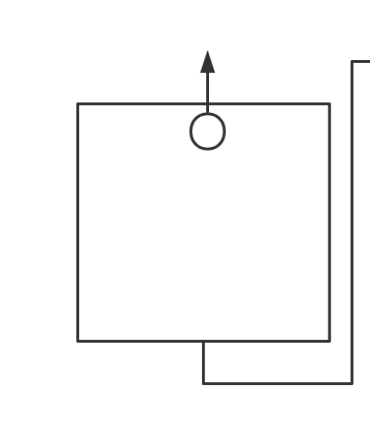
简化：



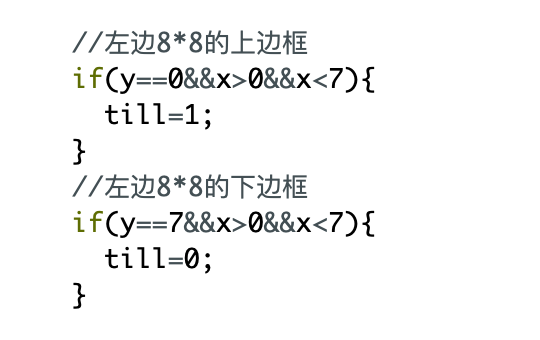
2.3矩阵一号上下边框：

2.3.1边界点无需考虑，左右边框处理已经完成边界点。

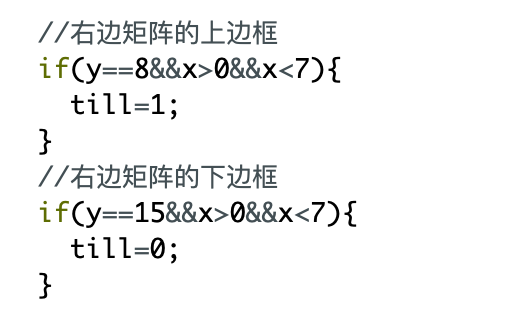
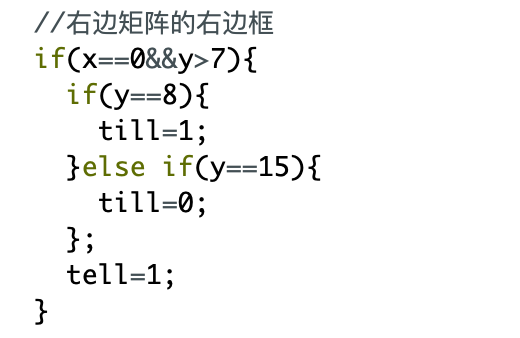
2.3.2非边界点：修改竖直方向，水平方向不变。

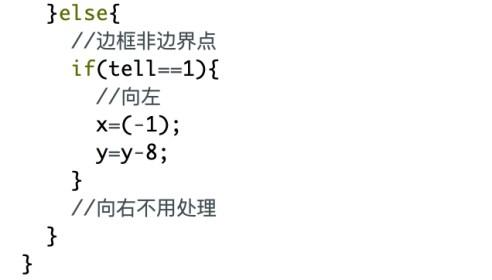
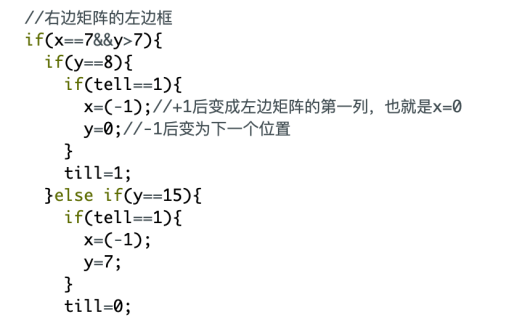


代码：



2.4矩阵二号处理模式与矩阵一号同理。原理不过多解释，以下为部分关键代码





1. 运行效果照片：



1. 参考文献：

8.1Arduino Uno and a 8x8 LED Matrix HT16K33 backpack---<https://create.arduino.cc/projecthub/wrightmac/arduino-uno-and-a-8x8-led-matrix-ht16k33-backpack-43e052>

8.2<https://create.arduino.cc/projecthub/Kian_Calnan/bouncing-ball-7ff589?ref=platform&ref_id=424_trending_part_beginner_&offset=21>

8.3Adafruit GFX图形库的使用 <http://www.elecfans.com/d/1123443.html>

8.4Arduino Adafruit GFX图形库介绍 <https://blog.csdn.net/weixin_44481398/article/details/99662120>

1. github地址：https://github.com/Sorrowa/Bouncing-Ball