反客核心板 OpenMV 操作手册

版本：V1.1　　创建日期: 2023-12-12

本文档将介绍如何使用反客核心板进行 OpenMV 的相关操作，包括刷写固件、配套屏幕的扩展使用等。所有的一切是建立在您已经熟悉了如何下载程序到单片机的基础上。

反客科技

FANKE

版本历史

版本	日期	说明
V1.0	2023-10-25	初次发布
V1.1	2023-12-12	1. 增加使用自动对焦的说明 2. 修改使用 LCD 的说明 3. 增加脱机运行时的注意事项

目录

版本历史.....1

1.使用 keil 下载固件.....3
2.使用 STM32CubeProgrammer 下载固件.....5
3.将核心板刷回正常状态.....7
4.关于 OpenMV 的版本.....8
5.扩展屏幕的使用.....9
5.1 使用自定义分辨率.....10
5.2 屏幕缩放显示.....12
6.串口外设.....14
7.脱机使用 OpenMV.....16
8.启用自动对焦.....18

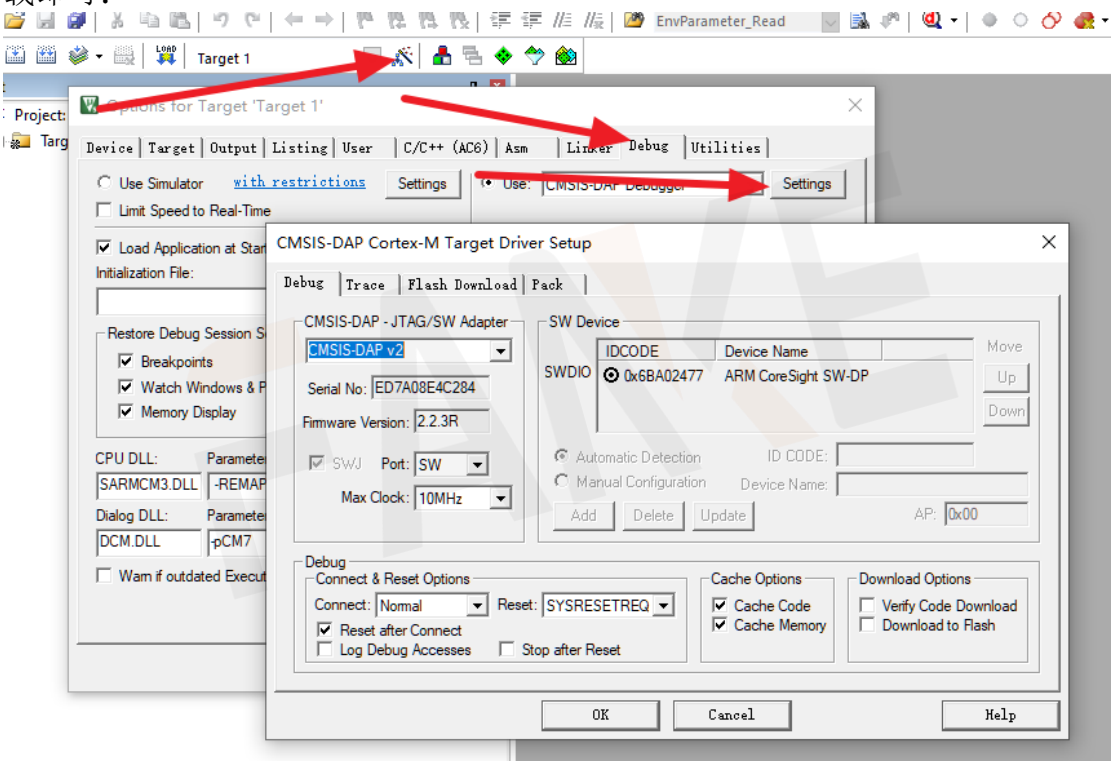
1.使用 keil 下载固件

所谓固件，其实就是一个.hex 或者.bin 程序文件，和点灯程序相比，就是程序文件大了一些，因此下载固件的方法很多，只要能把程序下载进单片机即可。

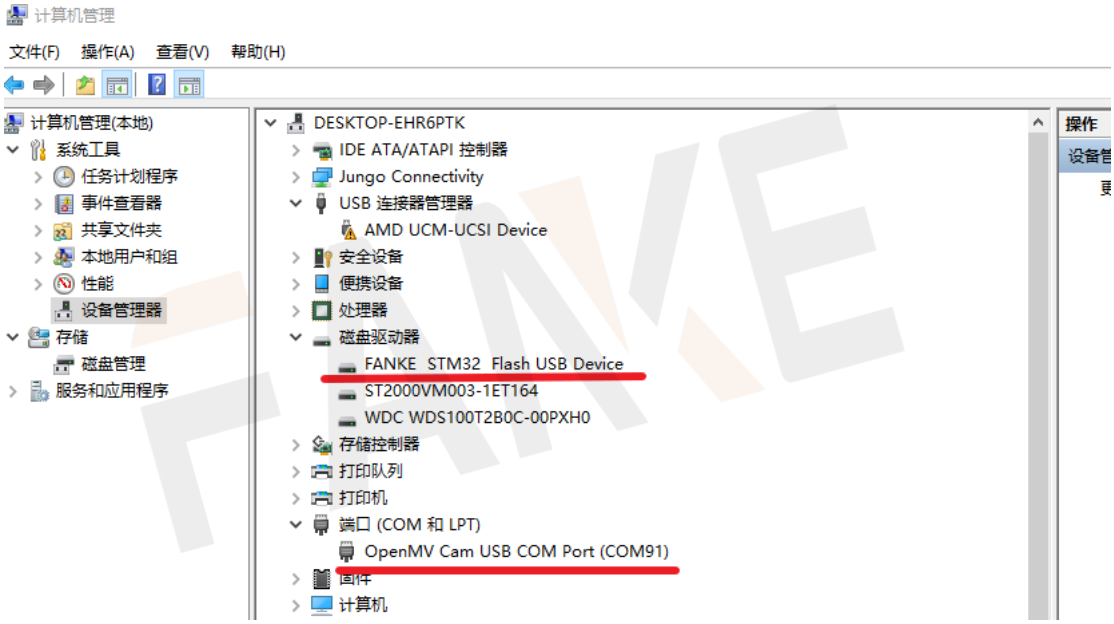
本章介绍的是使用 Keil 进行下载，**前提是您已经了掌握 Keil 以及相应下载器的使用方法**。

打开 OpenMV 的固件工程，路径如下（**此处以我们 743 的板子举例，实际选择您对应核心板的资料即可，路径存放位置一样**）：

直接用 keil 打开，配置一下您所用的下载器，其它的什么都不要改，直接下载即可：



断电重启核心板，插上 USB，等待几秒钟，就可以在设备管理器看到相应的设备，此时就可以直接打开 OpenMV IDE 使用了。

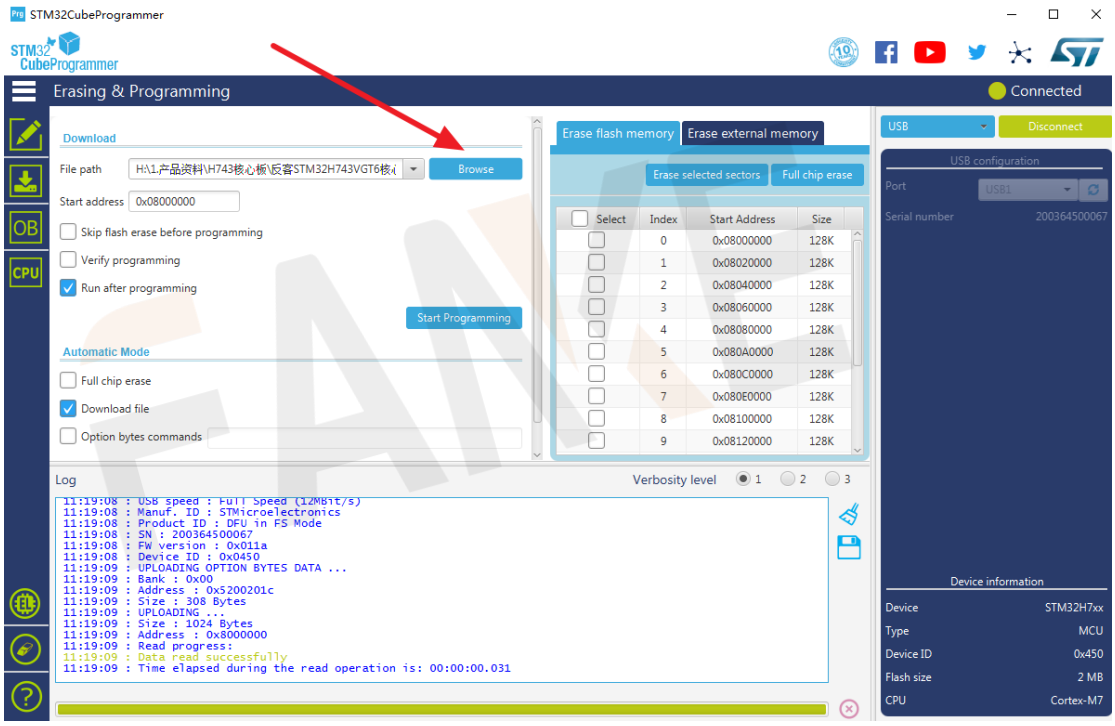


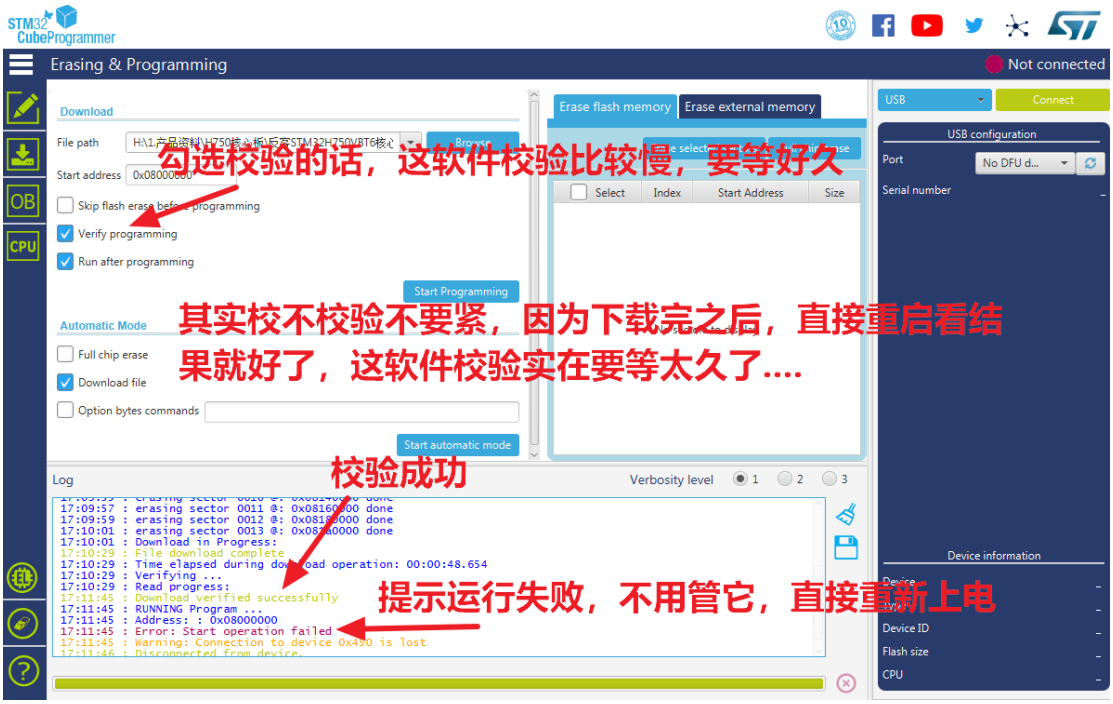
2.使用 STM32CubeProgrammer 下载固件

所谓固件，其实就是一个.hex 或者.bin 程序文件，和点灯程序相比，就是程序文件大了一些，因此下载固件的方法很多，只要能把程序下载进单片机即可。本章介绍的是使用 STM32CubeProgrammer 进行下载，**前提是您已经学会如何使 USB DFU 下载**，对应方法可以参考我们这个文档：

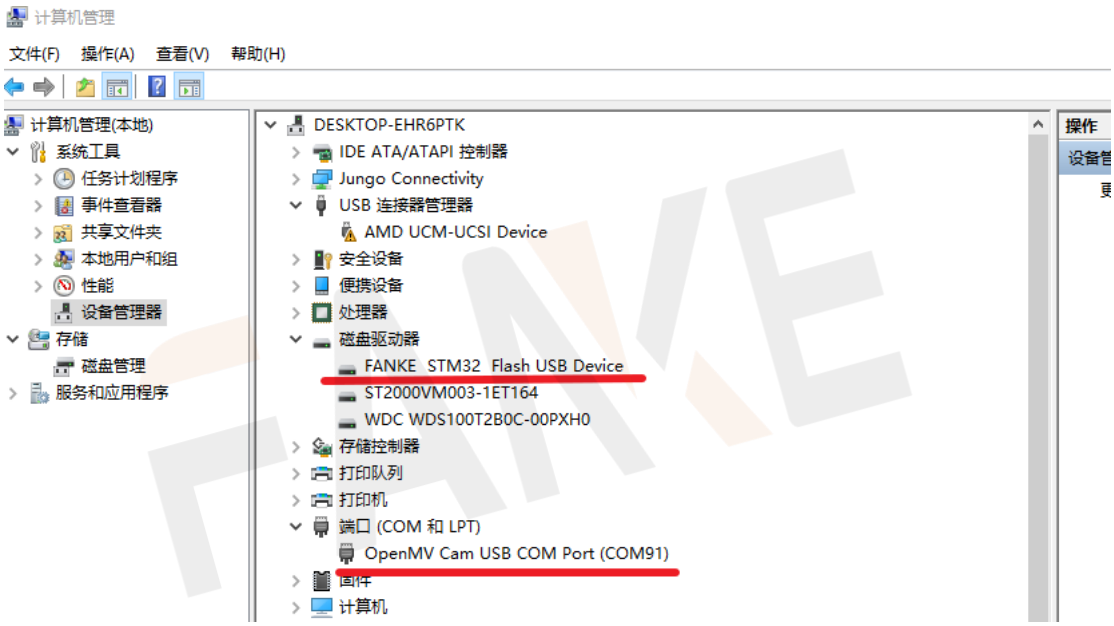
找到 OpenMV 的固件，路径如下（**此处以我们 743 的板子举例，实际选择您对应的核心板的资料即可，路径存放位置一样**）：

直接使用 STM32CubeProgrammer 打开固件，下载即可：



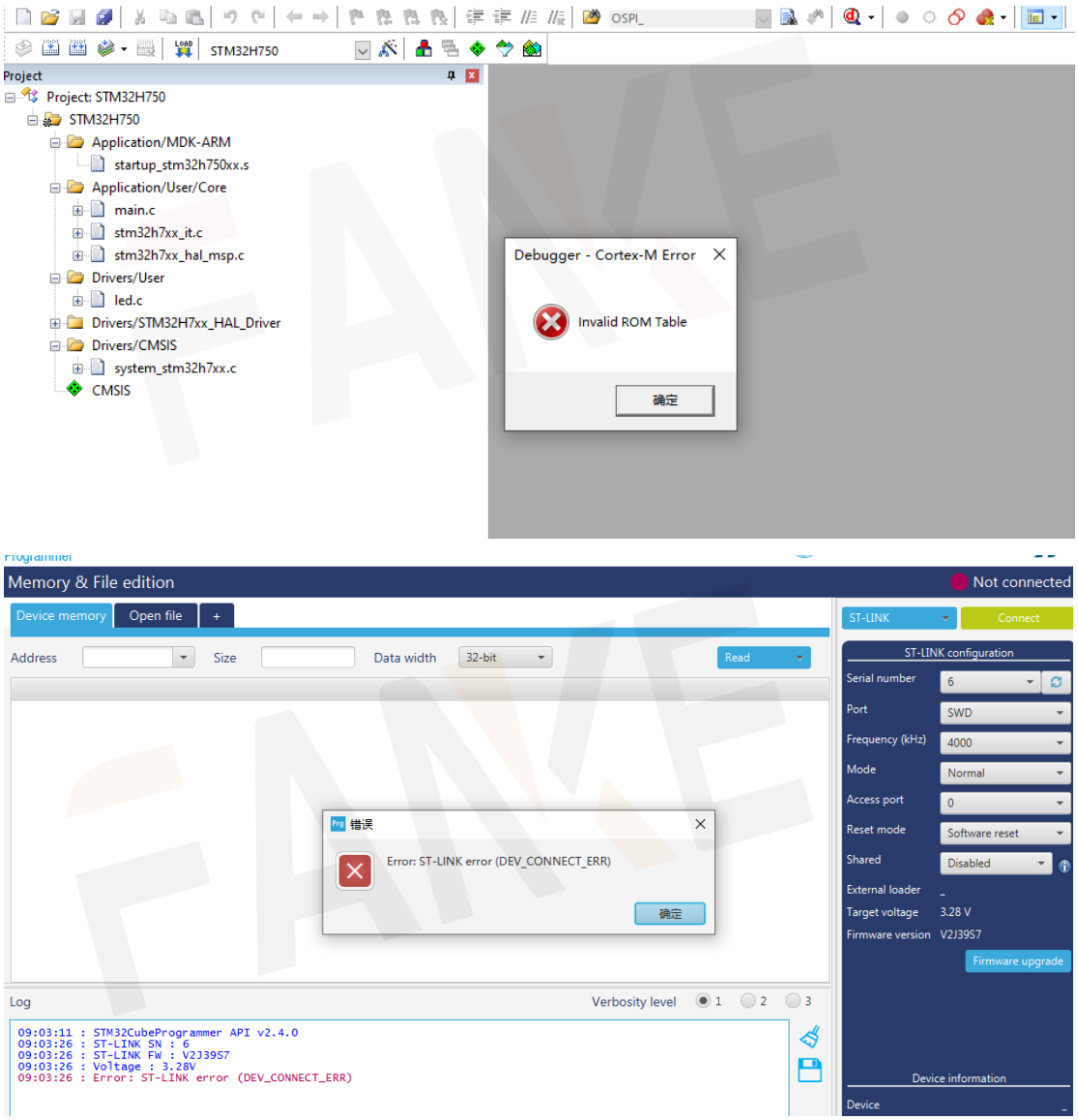


程序下载完之后，重启单片机，插上 USB，等待几秒钟，就可以在设备管理器看到相应的设备，此时就可以直接打开 OpenMV IDE 使用了。



3.将核心板刷回正常状态

如果用户想把核心板从 OpenMV 刷回正常的 STM32 开发模式，如果直接使用 SWD 进行连接，会提示连接下载失败：



出现这个情况，这时候按一下核心板的 BOOT 按键，然后会有一个黄色 LED 亮起，接着松开按键，再进行连接就好了，或者直接用 USB DFU 随便刷一个点灯的程序进去也可以。

4.关于 OpenMV 的版本

我们会在资料这里说明当前的 OpenMV 版本：

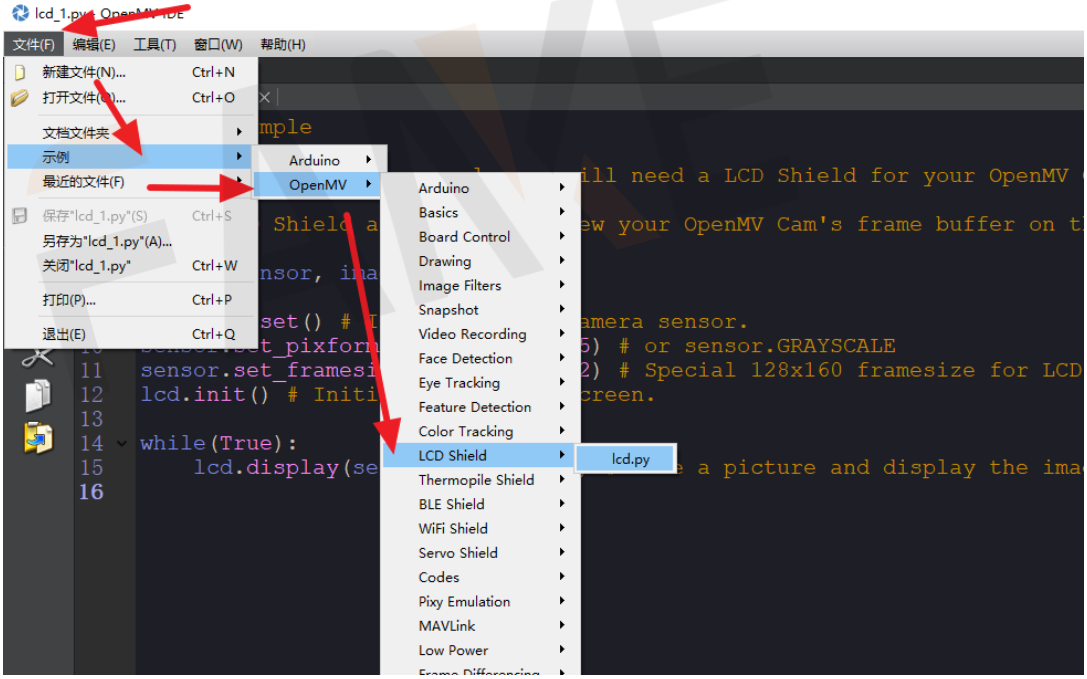
关于 OpenMV 每个版本的发行说明，大家可以到 github 搜 OpenMV 查看，其实很多时候并没有什么变动，有时候仅仅是针对一些板卡的调整和修复（OpenMV 不仅仅只有 STM32 的板子，还有很多其它的），所以我们不会实时跟着官方发布新固件，只有在隔一个大版本或者有大变动的时候进行更新：

如果 OpenMV IDE 这里提示您的固件过期时，可以不必理会，不会影响您的正常使用。而且因为我们的板子和官方的板子硬件不一样，不能直接用 OpenMV IDE 进行固件的更新。

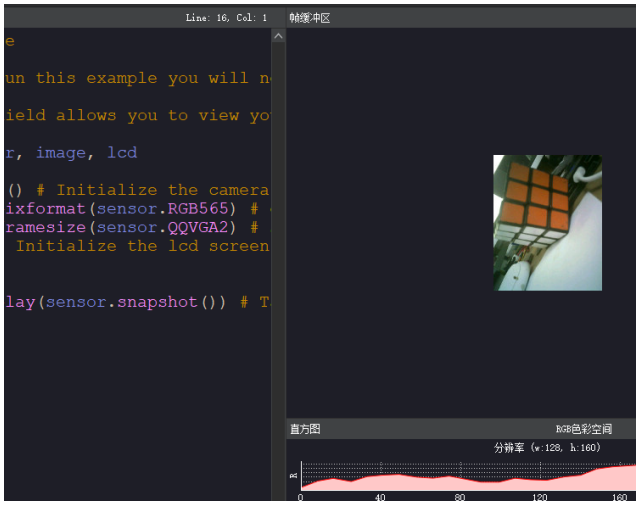
5.扩展屏幕的使用

我们为核心板搭配了好几个屏幕模块，用户可以搭配屏幕进行脱机使用，这样就可以不依赖电脑，将画面实时显示到屏幕上。

使用方法也很简单，直接打开 OpenMV 里的 LCD 示例：

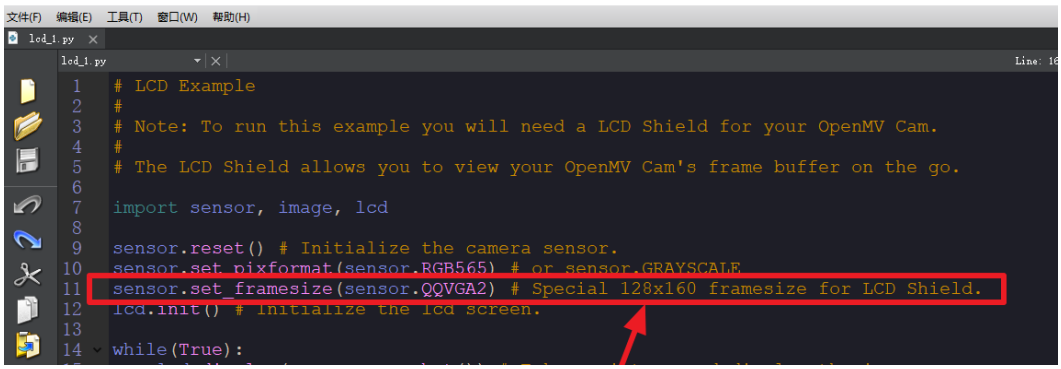


画面会同步到 LCD 模块显示：



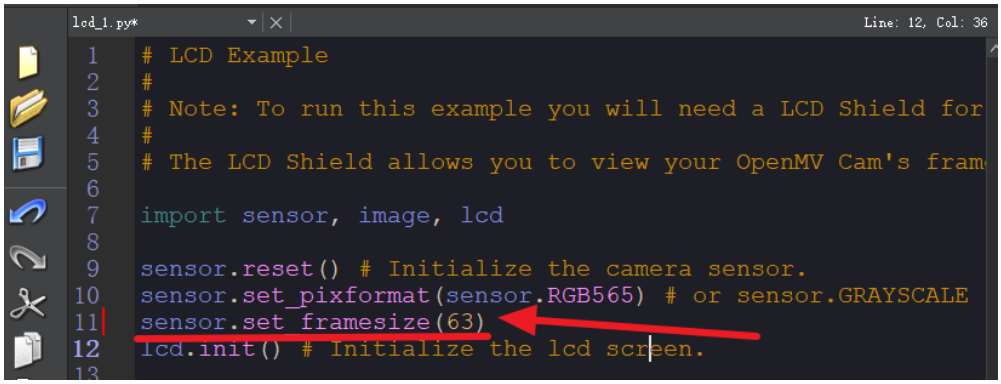
5.1 使用自定义分辨率

在上面的演示中，虽然屏幕能正常使用，但是画面很小且无法铺满屏幕，这是因为该示例中 OpenMV 配置的画幅只有 128x160，而实际我们使用的屏幕都是 240x240 或者更高。

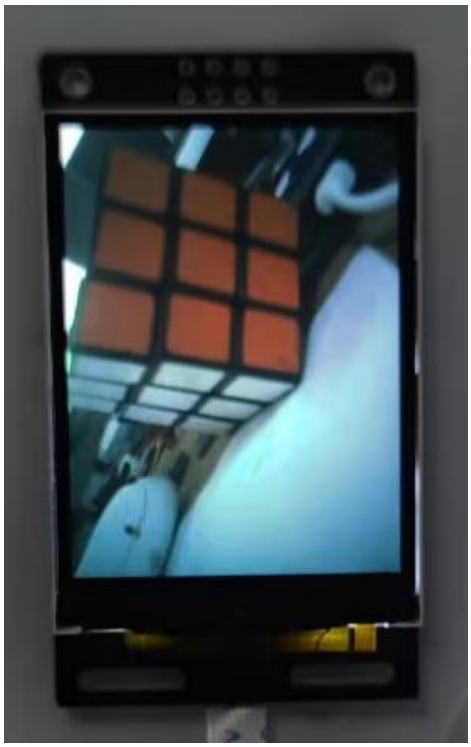


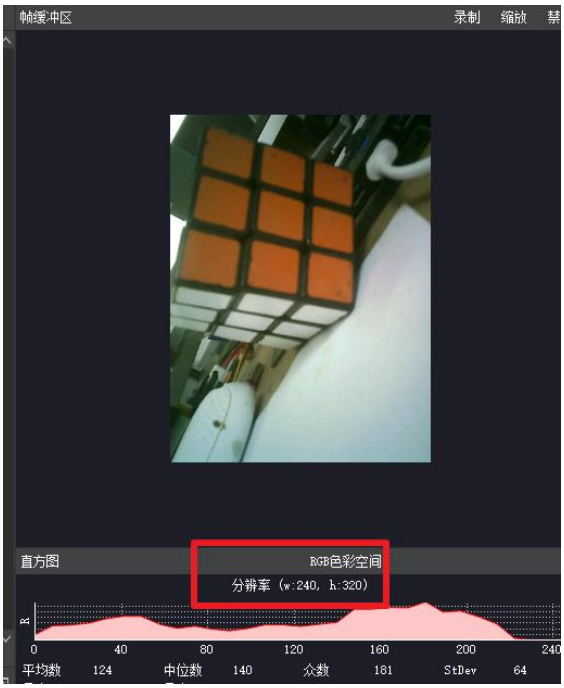
因此，我们在编译固件时，专门增加了几组自定义的分辨率用以适配我们的 LCD 模块。

修改也很简单，直接将 sensor.set_framesize() 里的参数改成对应的分辨率编号，假设屏幕分辨率为 240x320，则直接输入 63。



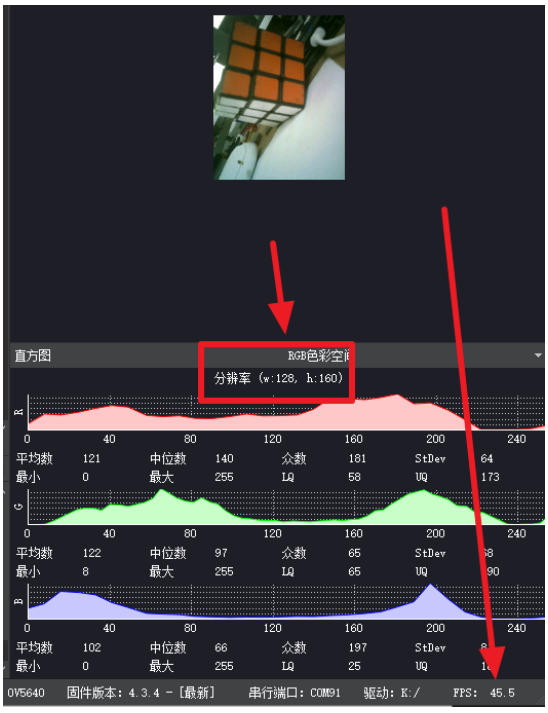
修改成功之后，可以在右边的信息中看到当前设置的分辨率，并且屏幕的显示也会跟着变化。

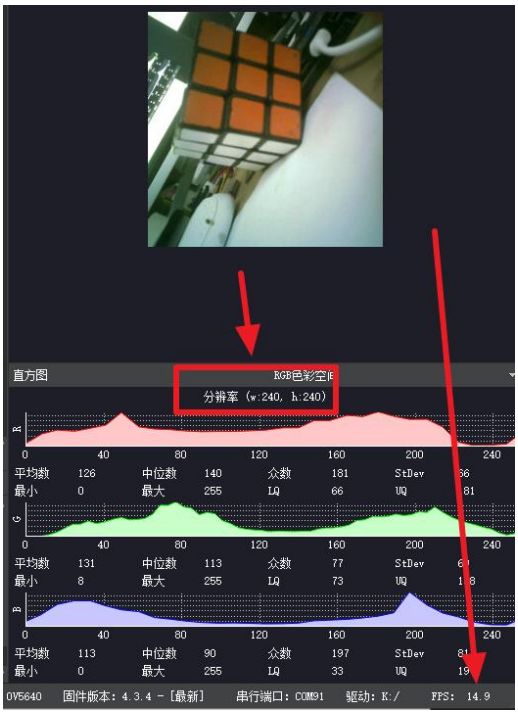




5.2 屏幕缩放显示

使用高分辨率虽然画面清晰且画幅大易观察，但是相应的也会降低画面捕捉的帧率，以下为不同分辨率下帧率的对比：

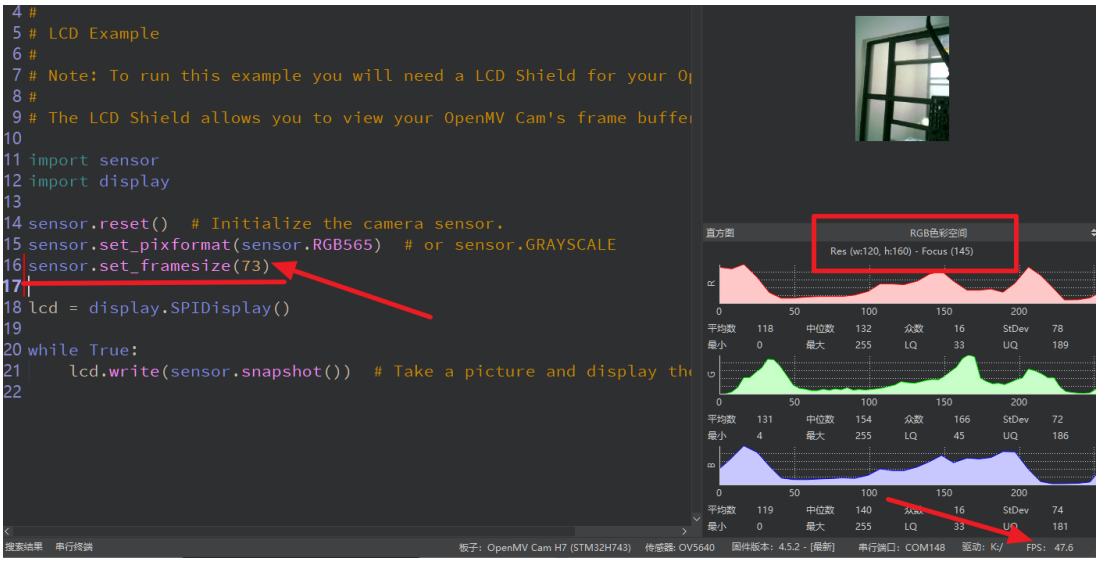




OpenMV 提供了缩放显示的功能，在使用小分辨获得高帧率的同时，能将画面铺满屏幕以便观察，为此我们在固件定义了半分辨率的配置：

```
{ 80, 120},  /*  70*/     // 定义一半的分辨率
{120, 120},  /*  71*/
{120, 240},  /*  72*/
{120, 160},  /*  73*/
```

假设屏幕分辨率为 240x320，直接将 sensor.set_framesize() 里的参数改成对应的半分辨率编号，这里输入 73。可以看到画幅已经变成了 120x160，也就是屏幕分辨率的一半，帧率会很高。



接着添加显示缩放参数：

lcd.write(sensor.snapshot(),x=0,y=0,x_scale=2.0,y_scale=2.0)

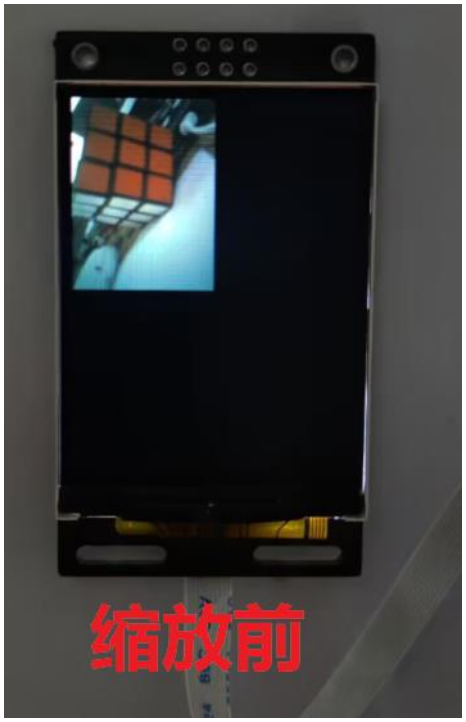
120x160 刚好是屏幕分辨率 240x320 的一半，因此 X 和 Y 轴缩放 2 倍刚好可以铺满屏幕。

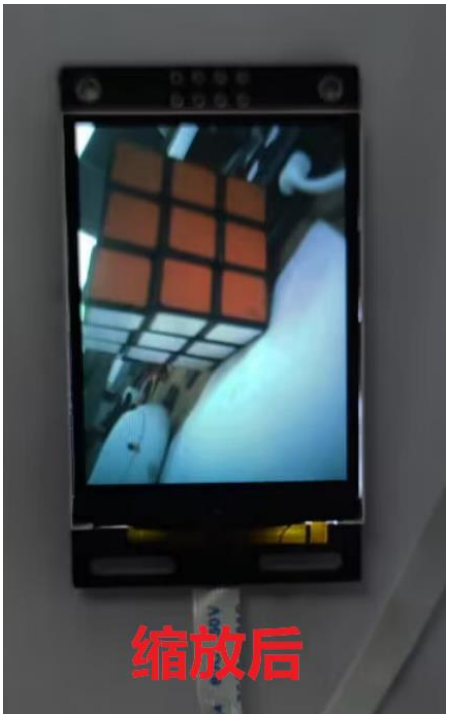
```
11 import sensor
12 import display
13
14 sensor.reset()  # Initialize the camera sensor.
15 sensor.set_pixformat(sensor.RGB565)  # or sensor.GRAYSCALE
16 sensor.set_framesize(73)
17
18 lcd = display.SPIDisplay()
19
20 while True:
21     lcd.write(sensor.snapshot(),x=0,y=0,x_scale=2.0,y_scale=2.0)
```

起始坐标

缩放倍数

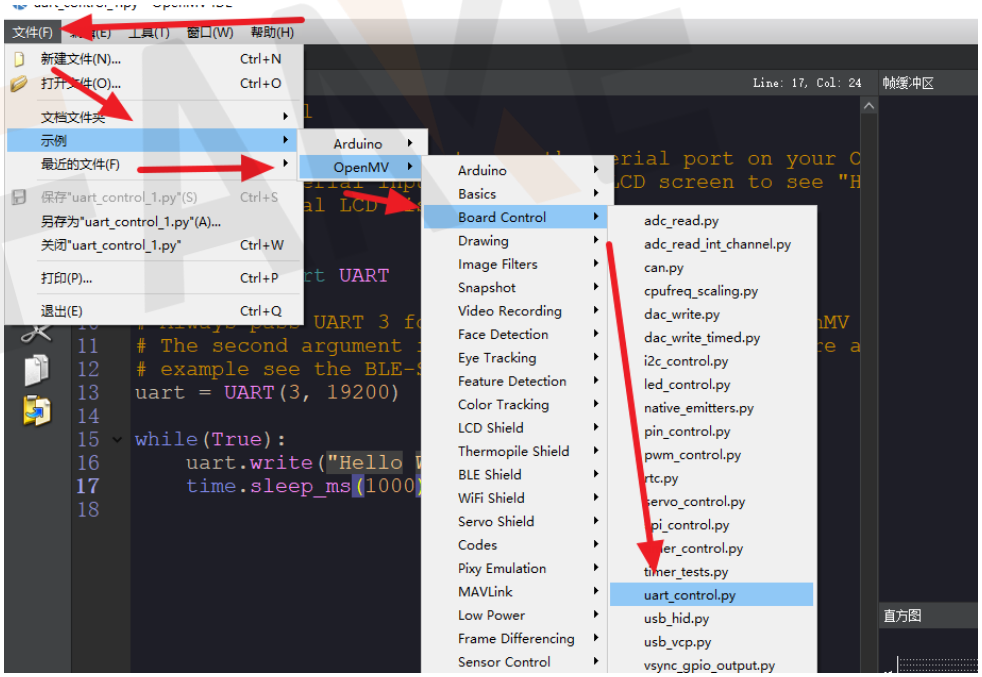
最终效果如下，用户可以根据实际情况，选择合适的分辨率以及是否使用缩放显示。



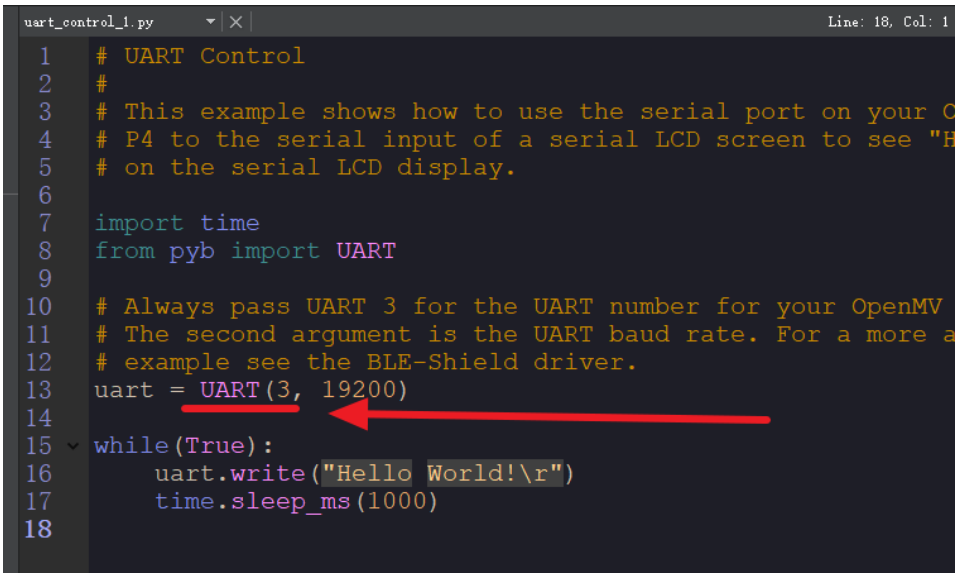


6.串口外设

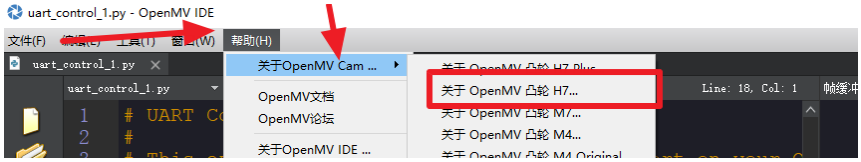
通常我们需要使用串口，将 OpenMV 识别到的数据发到别的单片机进行处理，本章节介绍的是如何使用核心板对应 OpenMV 的引脚。先打开 IDE 里的串口示例：



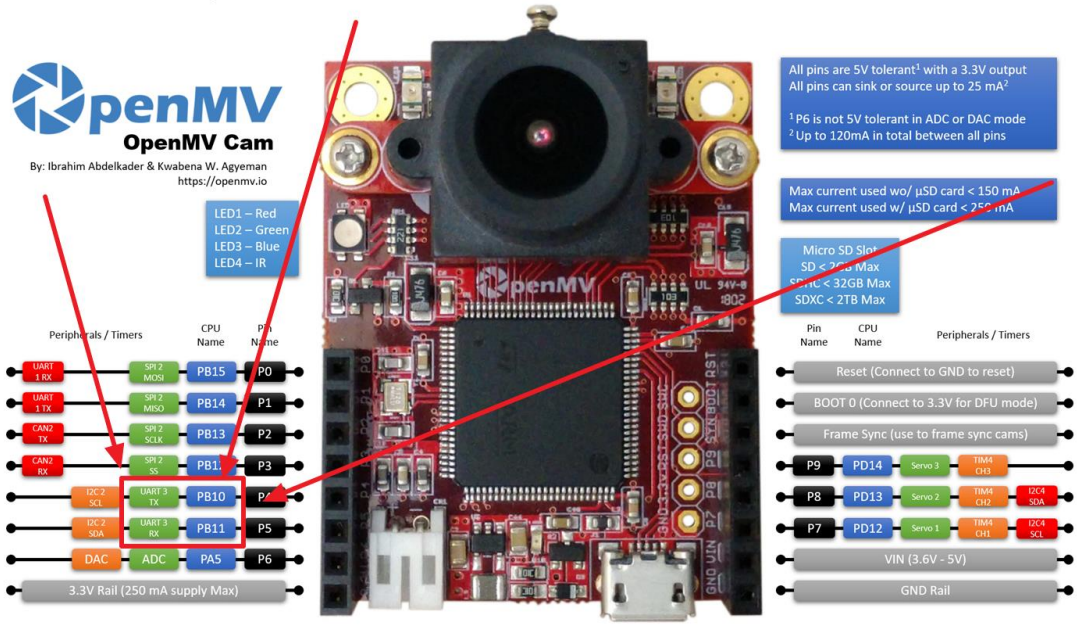
该示例用的是串口 3，波特率 19200：



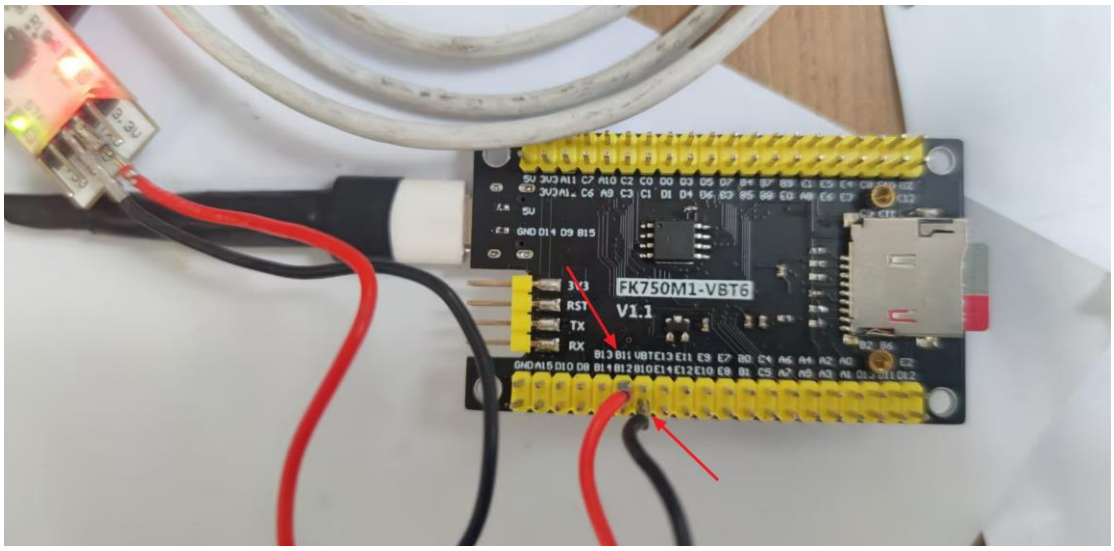
接着打开帮助文档，查看引脚说明：



在弹出的引脚图里，可以看到串口 3 引脚用的是 PB10/PB11：



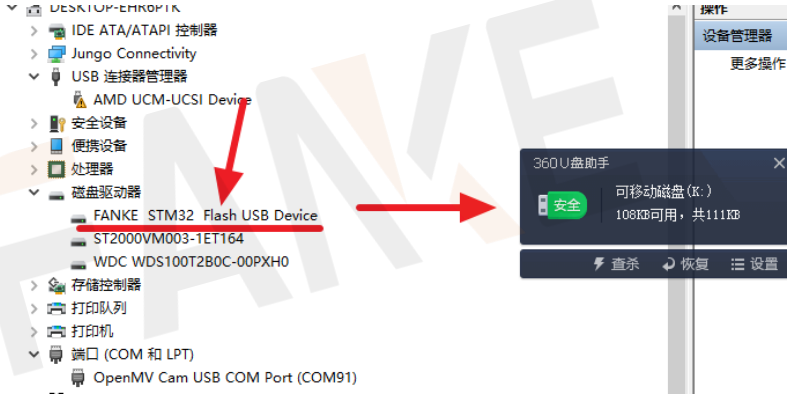
最后找到核心板的 PB10/PB11 引脚即可，这里以 FK750M1-VBT6 核心板为例，其它的 H750 核心板引脚也一样，找到 PB10/PB11 就好了。



外接 USB 转 TTL 串口模块，再用串口助手验证一下，看看是否通信正常。如果用户需要使用其它外设引脚，方法也和上面一样，对照一下两边的引脚即可。

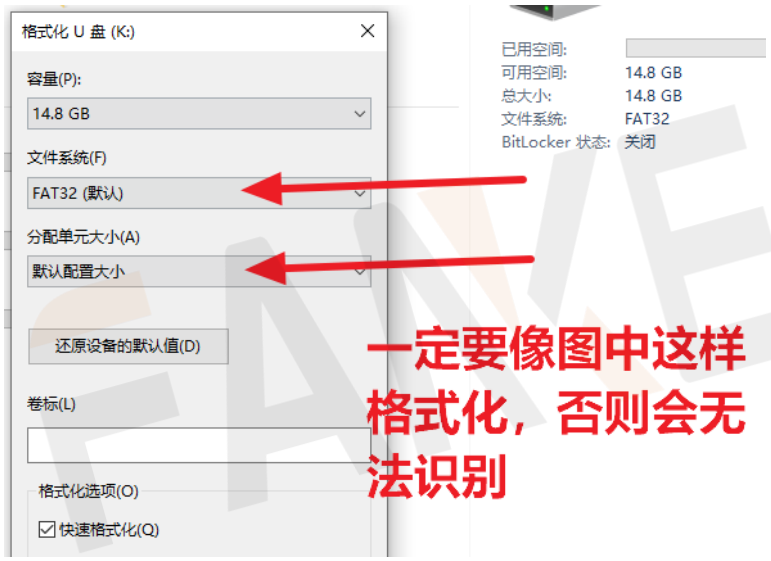
7.脱机使用 OpenMV

脱机使用需要将程序文件存储到 OpenMV 对应的磁盘里，如果没有插入 SD 卡，设备管理器的名称如下，并且磁盘大小只有 100 多 KB。

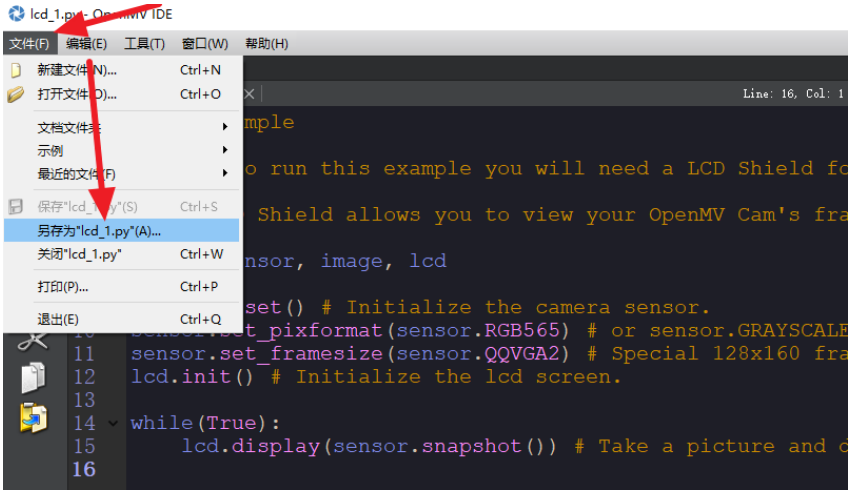


如果默认的磁盘不够使用，可以插入 SD 卡（TF 规格）到核心板背面的卡座，然后重新连接电脑，电脑会将 SD 卡识别成 U 盘，且会在设备管理器看到对应的设备。

如果 SD 卡插进去没有被 OpenMV 识别到，可以先找个读卡器，重新对 SD 卡进行格式化，并且一定要按图片那样配置。

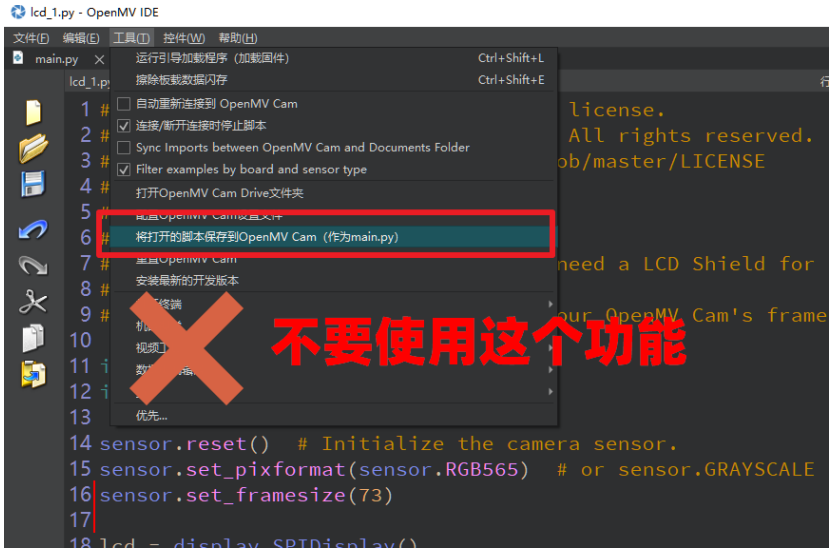


在电脑端将程序调试好之后，直接点击另存为，找到 OpenMV 的磁盘，将文件命名为 main.py



然后重新上电，核心板会自动运行磁盘里的 main.py 文件，所以一定要记得修改文件名，不然是不会运行的。

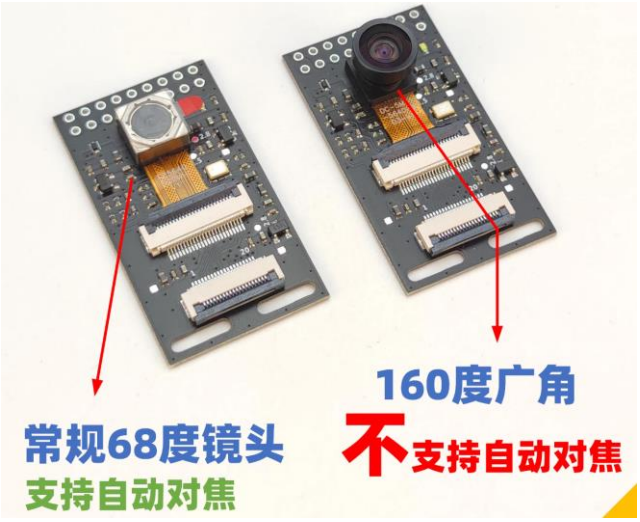
一定要按照上面的步骤来，不要直接使用 IDE 自带的这个功能，不然容易掉固件：



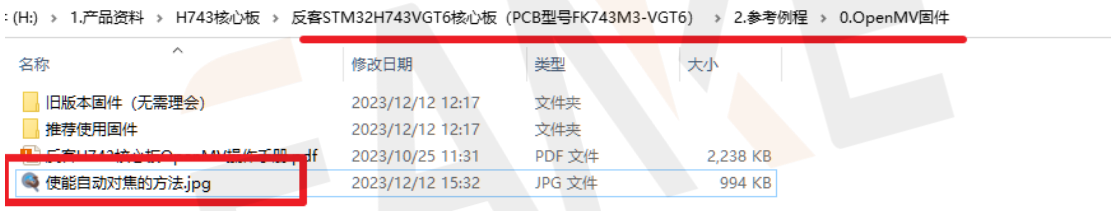
8.启用自动对焦

因为自动对焦功能会经常被误触发，所以默认状态下，是不启用的，也就是默认对焦远处，但是那样拍近物就会比较模糊。

不是所有摄像头都支持自动对焦的，您先核对购买信息，看看您的摄像头是否支持自动对焦的功能，例如 5640：



启用自动对焦的方法很简单，短接两个引脚即可，具体您可以在附带的资料链接里找到说明，每个板子的使能引脚不一样，一定要对好型号：



因为不是微距镜头，如果摄像头离物品太近，是无法对焦的。如果发现无法触发对焦，可以尝试把摄像头对准远处，对焦远处之后，再拉回，一般都可以触发对焦。