



# e-Paper ESP32 Driver Board

## 用户手册

### 产品简介

这是一款电子墨水屏<sup>1</sup>无线网络驱动板，支持用户通过 wifi 或者蓝牙从 PC 机或者智能手机上获取图片信息，并将图片刷新到电子墨水屏上显示。

本驱动板集成了板载有 ESP32 模组，并在 PCB 两端引出 ESP32 的全部引脚，可以连接其他设备使用。提供有蓝牙以及 WIFI 示例程序，支持 Arduino 开发。

### 特点

- 板载 ESP32，支持 Arduino 开发
- 提供安卓手机 APP 程序，可通过蓝牙 EDR 更新显示内容，方便使用
- 提供 HTML 上位机程序，可通过网页远程更新显示内容，方便集成到各种网络应用中
- 支持 Floyd-Steinberg 抖动算法，以获得更多的颜色组合，对原始图片进行更好的阴影渲染
- 支持多种常用图片格式(BMP、JPEG、GIF 和 PNG 等)
- 出厂内置电子墨水屏驱动程序(开源)
- 提供完善的配套资料手册

### 参数

- WiFi 标准: 802.11b/g/n
- 蓝牙标准: 4.2，包含传统蓝牙(BR/EDR)和低功耗蓝牙(BLE)

---

<sup>1</sup> 仅支持微雪墨水屏

- 通信接口：3-wire SPI、4-wire SPI(默认)
- 工作电压：5V
- 工作电流：50mA ~ 150mA
- 外形尺寸：29.46mm x 48.25mm

## 引脚

硬件上，墨水屏占用连接的引脚入下图

```
26 /* SPI pin definition -----*/
27 #define PIN_SPI_SCK 13
28 #define PIN_SPI_DIN 14
29 #define PIN_SPI_CS 15
30 #define PIN_SPI_BUSY 25
31 #define PIN_SPI_RST 26
32 #define PIN_SPI_DC 27
```

## 支持的屏幕型号

- 1.54inch e-Paper, 1.54inch e-Paper (B), 1.54inch e-Paper (C)
- 2.13inch e-Paper, 2.13inc e-Paper (B), 2.13inch e-Paper (C), 2.13inch e-Paper (D)
- 2.7inch e-Paper, 2.7inc e-Paper (B)
- 2.9inch e-Paper, 2.9inc e-Paper (B), 2.9inch e-Paper (C)
- 4.2inch e-Paper, 4.2inc e-Paper (B), 4.2inch e-Paper (C)
- 5.83inch e-Paper, 5.83inch e-Paper (B), 5.83inch e-Paper (C)
- 7.5inch e-Paper, 7.5inc e-Paper (B), 7.5inch e-Paper (C)

## 应用

本产品配合墨水屏，适用于无线刷图的应用场景。

- 超市电子价签
- 电子名片
- 串口信息显示牌等



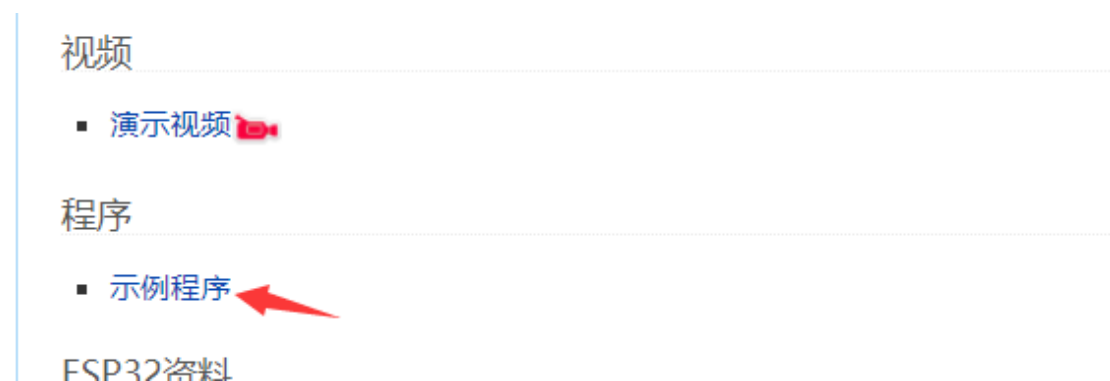
## 目录

产品简介 .....	1
特点 .....	1
参数 .....	1
引脚 .....	2
支持的屏幕型号 .....	2
应用 .....	2
使用说明 .....	5
下载例程和 APP .....	5
硬件连接 .....	6
Arduino IDE 安装和 EPS32 环境配置 .....	8
蓝牙 Demo 使用 .....	9
WiFi Demo 使用 .....	11
图像处理算法 .....	15
色阶法 .....	15
抖动法 .....	16
两种算法的处理效果图 .....	17
数据通信协议 .....	19
指令 .....	19
初始化 .....	19
图像数据格式 .....	21

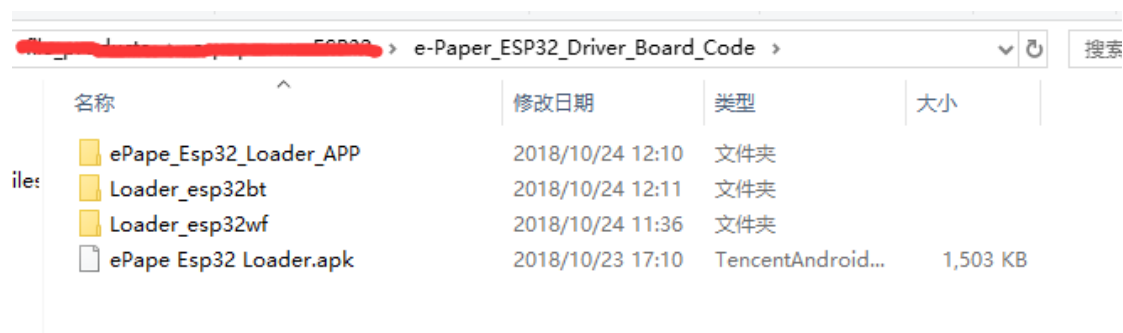
## 使用说明

### 下载例程和 APP

我们提供有蓝牙和 WiFi 两种例程。在微雪百科界面找到 [e-Paper ESP32 Driver Board](#) 资料界面，并下载示例程序。



将下载下来的压缩包解压出来，可以得到以下文件：



ePape\_Esp32\_loader\_APP: 蓝牙 APP 源码 (Android Studio)

Loder\_esp32bt: 蓝牙 demo 例程

Loader\_esp32wf: WiFi demo 例程

ePape Esp32 Loader.apk: 蓝牙 APP 安装包 (仅支持 Android 手机)

## 硬件连接

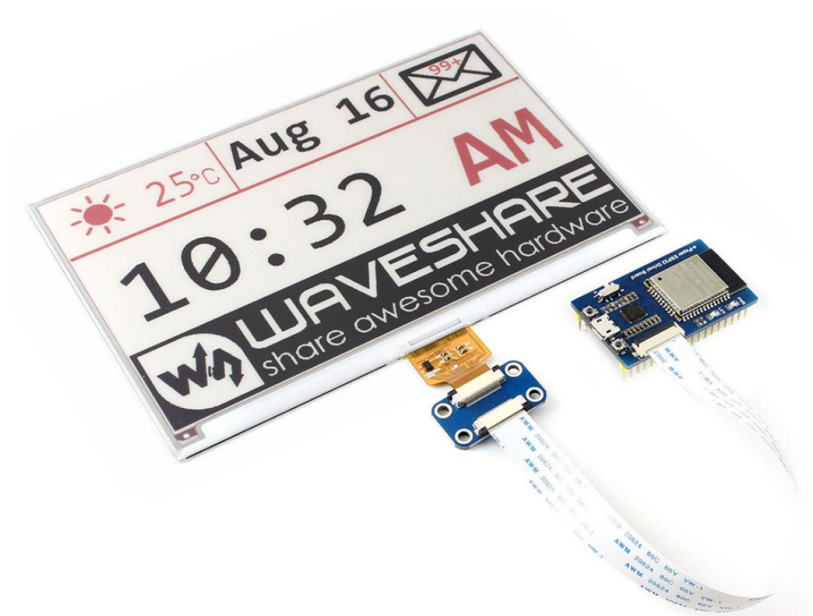
本产品出货的时候配有一个 ESP32 网络驱动板，一个转接板和 FFC 延长线。使用的时候你可以直接将屏幕接入到驱动板，或者是通过延长线和转接板接入

### 1 将屏幕接入 ESP32 驱动板：

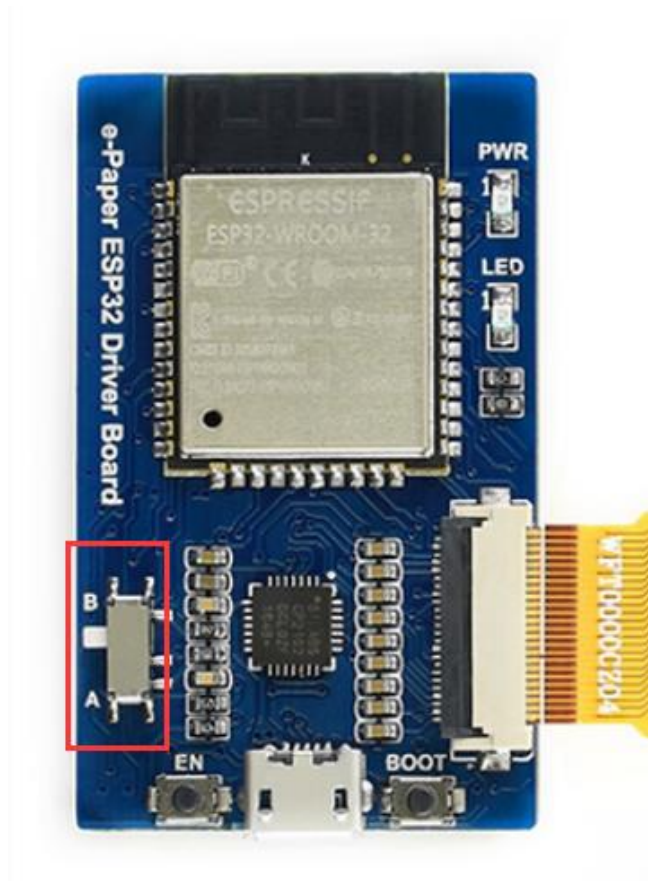
#### 1.1 直接接入驱动板：



#### 1.2 通过延长线和转接板接入：



- 2 设置型号开关，根据实际使用的墨水屏型号设置一下型号开关



- 3 使用一条 micro USB 线将 ESP32 驱动板接入到电脑或者 5V 电源。

**型号对应关系：**

A	B
1.54inch e-Paper	1.54inch e-Paper (B)
2.13inch e-Paper	1.54inch e-Paper (C)
2.13inch e-Paper (D)	2.13inch e-Paper (B)
2.9inch e-Paper	2.13inch e-Paper (C)
	2.7inch e-Paper (B)
	2.9inch e-Paper (B)
	2.9inch e-Paper (C)
	4.2inch e-Paper (B)
	4.2inch e-Paper (C)
	5.83inch e-Paper (B)
	5.83inch e-Paper (C)
	7.5inch e-Paper (B)
	7.5inch e-Paper (C)

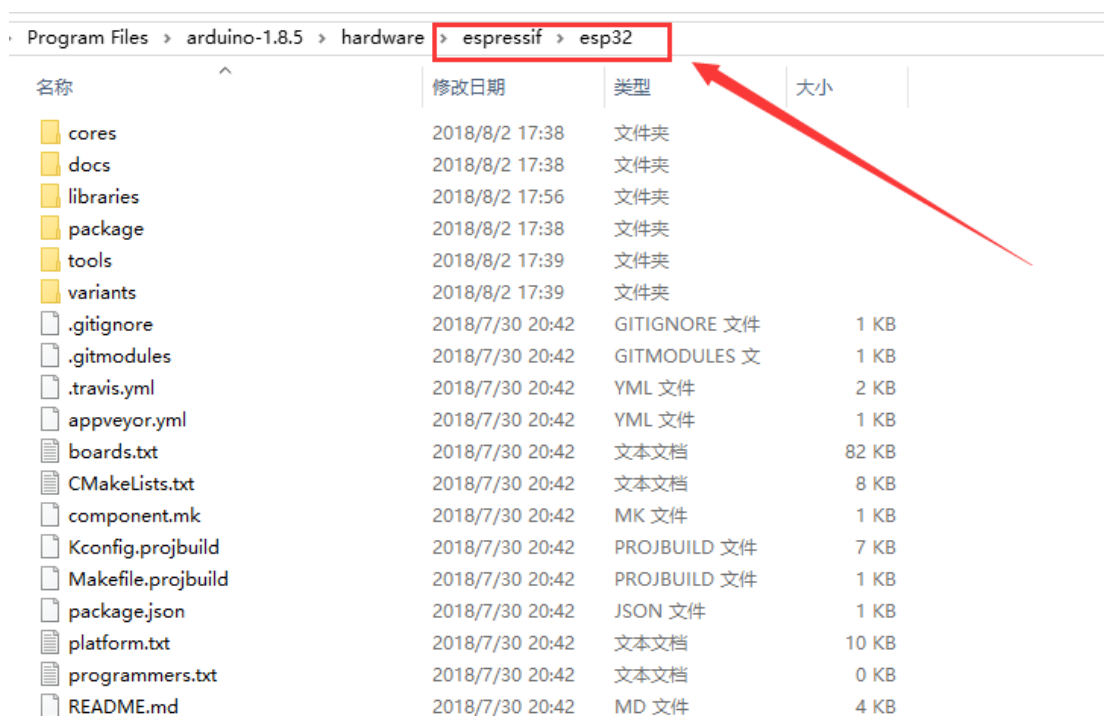
## ARDUINO IDE 安装和 EPS32 环境配置

1. 如果您电脑之前并没有安装有 Arduino IDE，或者 IDE 的版本比较老。建议到 Arduino 官方网站根据自己的系统型号下载最新的 IDE 并安装。

-官网链接: <https://www.arduino.cc/en/Main/Software>

2. 下载 Arduino-ESP32 支持包: <https://codecadload.github.com/espressif/arduino-esp32/zip/master> . 并将压缩包里面的文件解压到 Arduino IDE 安装目录下的

Hardware->espressif->esp32 路径。（注意，如果在安装目录下没有相应文件夹的话，可以手动创建一下）。



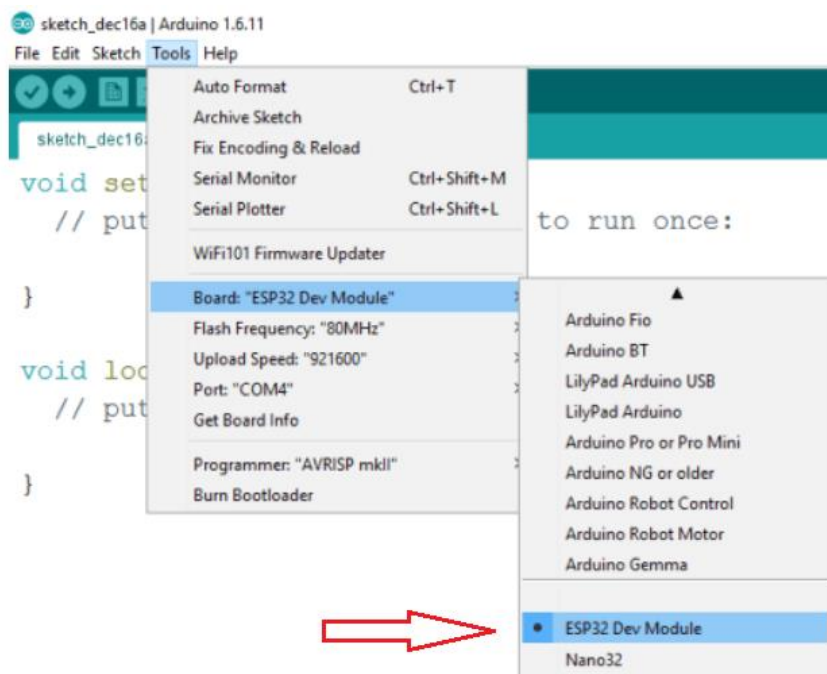
名称	修改日期	类型	大小
cores	2018/8/2 17:38	文件夹	
docs	2018/8/2 17:38	文件夹	
libraries	2018/8/2 17:56	文件夹	
package	2018/8/2 17:38	文件夹	
tools	2018/8/2 17:39	文件夹	
variants	2018/8/2 17:39	文件夹	
.gitignore	2018/7/30 20:42	GITIGNORE 文件	1 KB
.gitmodules	2018/7/30 20:42	GITMODULES 文	1 KB
.travis.yml	2018/7/30 20:42	YML 文件	2 KB
appveyor.yml	2018/7/30 20:42	YML 文件	1 KB
boards.txt	2018/7/30 20:42	文本文档	82 KB
CMakeLists.txt	2018/7/30 20:42	文本文档	8 KB
component.mk	2018/7/30 20:42	MK 文件	1 KB
Kconfig.projbuild	2018/7/30 20:42	PROJBUILD 文件	7 KB
Makefile.projbuild	2018/7/30 20:42	PROJBUILD 文件	1 KB
package.json	2018/7/30 20:42	JSON 文件	1 KB
platform.txt	2018/7/30 20:42	文本文档	10 KB
programmers.txt	2018/7/30 20:42	文本文档	0 KB
README.md	2018/7/30 20:42	MD 文件	4 KB

3. 打开 tools，并以管理员身份运行一下 get.exe 文件
4. 等待安装完成后，你可以在 IDE 的 Tools->Boards 里面找到 ESP32 Dev Module 的型号选项即可。



## 蓝牙 DEMO 使用

- 1 打开 Loader\_esp32bt 目录，双击 Loader\_esp32bt.ino 文件打开 Arduino 工程
- 2 选择 Tools->Boards->ESP32 Dev Module，并且选择好对应的串口（Tools->Port->\_



- 3 然后点击上传，把程序编译并下载到 ESP32 驱动板上
- 4 手机安装 APP
  - 4.1 将之前下载的 apk 文件传到安卓手机中进行安装
  - 4.2 安装完成后打开蓝牙 APP，在主页面有五个选项分别是：

**连接蓝牙：**用来连接 ESP32 设备

**打开图像文件：**点击可以选择手机里面的一张图片打  
开

**选择墨水屏型号：**选择你接入到驱动板的墨水屏型号

**选择图片处理算法：**由于手机中的图片并不一定符合



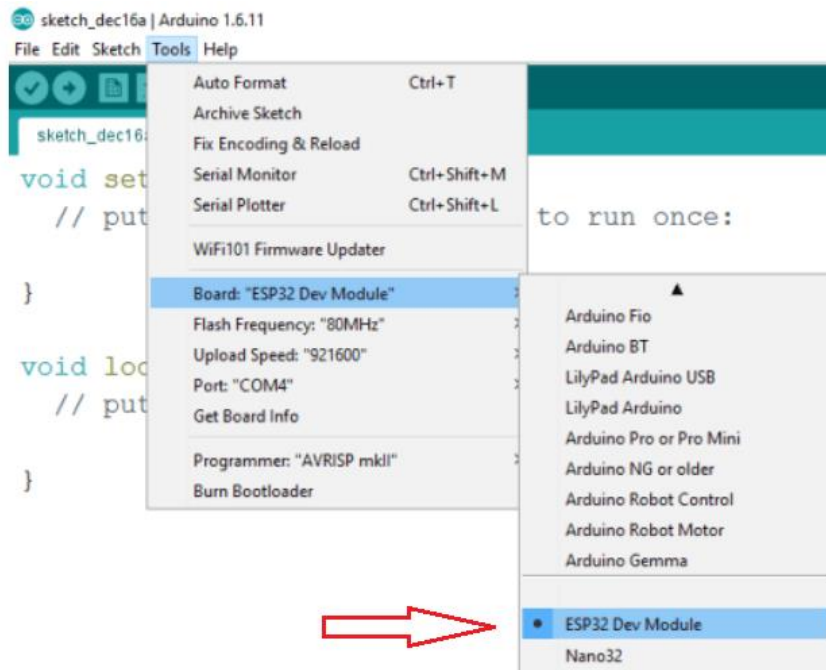
墨水屏型号的需求，所以要先处理一下图片

**上传图片：**将处理之后的图片上传到墨水屏，并刷新到屏幕上去

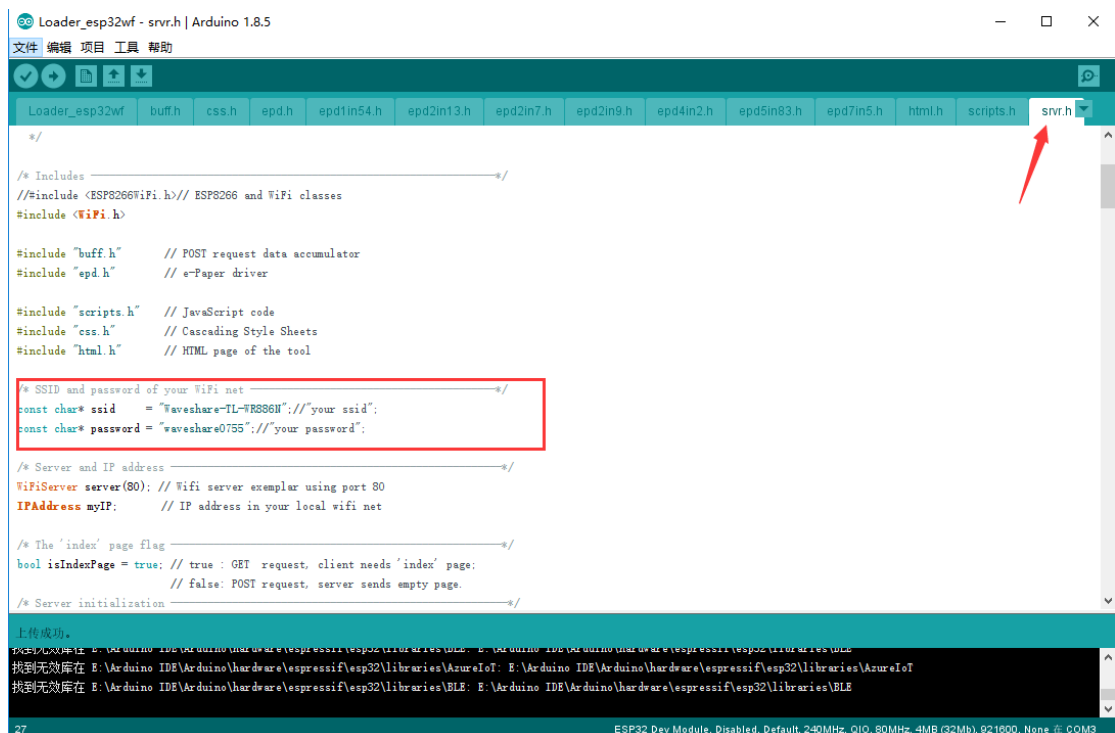
- 5 首先确保你已经打开手机蓝牙。点击 “连接蓝牙” -> 点击右上角的 “SCAN” 进行蓝牙设备扫描。
- 6 找到 ESP32 设备，点击进行连接。如果你是第一次连接这个设备，会弹出配对信息，点击确认完成配对。（注意：如果设备没有进行配对，将无法正常工作，并且可能出现 APP 闪退的问题）
- 7 点击 “打开图像文件” 选择一张图片
- 8 点击 “选择墨水屏型号” 选择你连接的墨水屏对应型号
- 9 点击 “选择图片处理算法” 选择对应的处理算法，并确认
  - 黑白色阶算法（将图片处理成黑白两色，并根据墨水屏分辨率切割图片大小）
  - 三色色阶算法（将图片处理成三色，并根据墨水屏分辨率切割图片大小，只适用于三色墨水屏）
  - 黑白抖动算法（将图片处理成黑白两色，并根据墨水屏分辨率切割图片大小）
  - 三色抖动算法（将图片处理成三色，并根据墨水屏分辨率切割图片大小，只适用于三色墨水屏）
- 10 点击 “上传图像”，把处理过后的图像上传到墨水屏中显示

## WIFI DEMO 使用

- 1 进入 Loader\_esp32wf 文件夹，双击 Loader\_esp32wf.ino 文件打开工程。
- 2 选择 Tools->Boards->ESP32 Dev Module，并且选择好对应的串口 (Tools->Port->\_

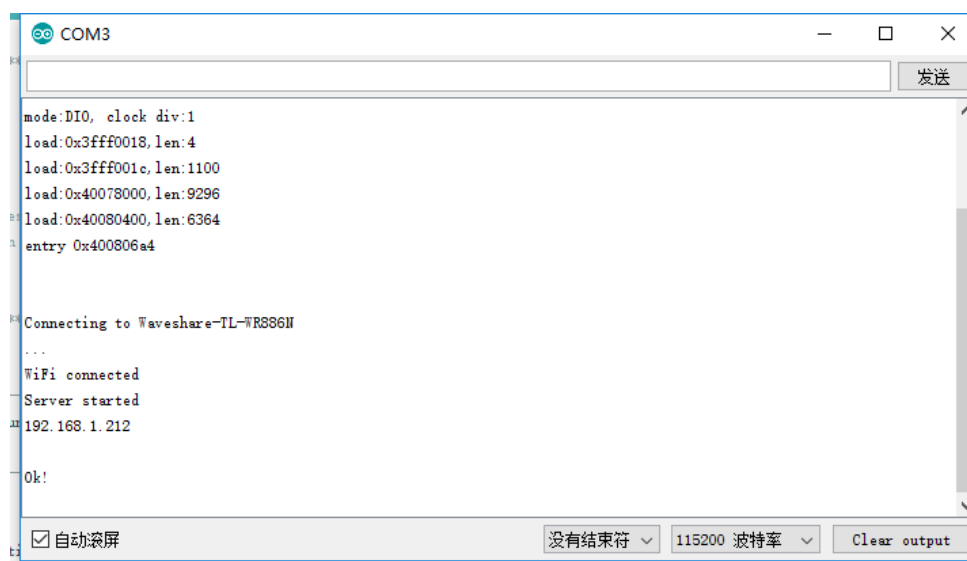


- 3 打开 srvr.h 文件，将 ssid 和 password 改成实际使用的 WiFi 用户名和密码



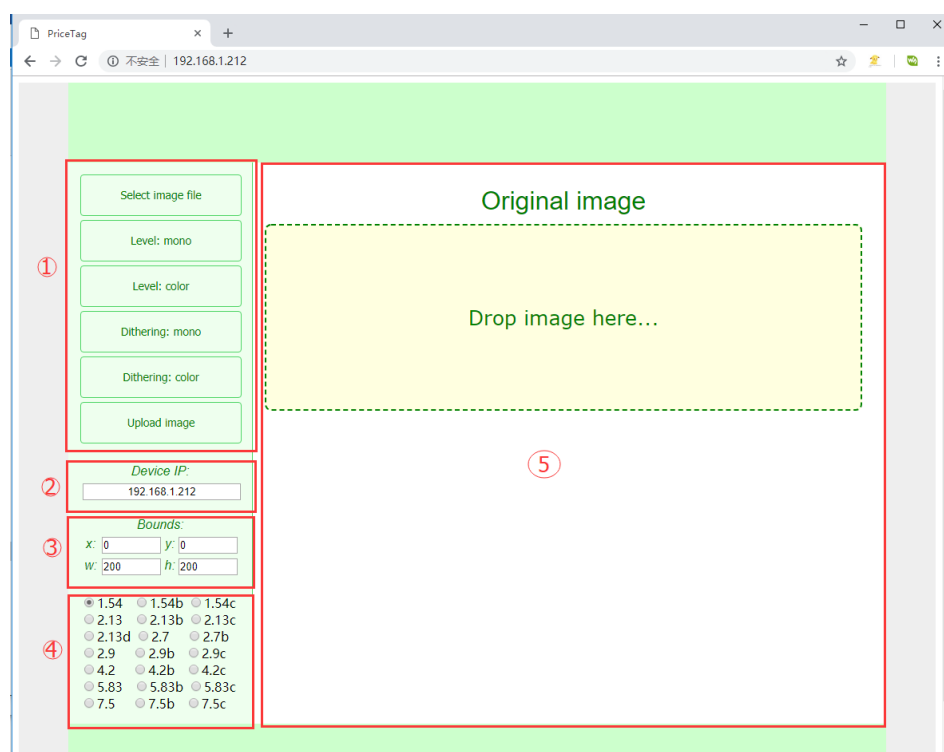
- 4 然后点击上传，把程序编译并下载到 ESP32 驱动板上
- 5 打开串口监视器，设置波特率为 115200，按下驱动板上面的 EN 按键，可以看到串口将

ESP32 驱动板的 IP 地址打印出来：



- 6 电脑或者手机（注意电脑/手机接入的网络需要时跟 ESP32 接入的 wifi 是同一个网段的才可以）打开浏览器，在网址输入栏输入 ESP32 的 IP 地址并打开，可以看到操作界面如

下：



### 6.1 图像操作区域:

- Select Image file: 点击在电脑或者手机里面选择一张图片
- Level: mono: 黑白色阶图像处理算法
- Level: color: 三色色阶图像处理算法 (只对三色屏幕生效)
- Dithering: mono: 黑色抖动图像处理算法
- Dithering: color: 三色抖动图像处理算法 (只对三色屏幕生效)
- Update image: 上传图像

### 6.2 IP 信息显示区域: 这里显示的是你当前连接的模块的 IP 地址信息

### 6.3 图像大小设置区域: 这里 x 和 y 可以设置你要显示的起始位置, 这个设置是相对于你选择的图片文件的, 比如选择一张 800x480 的图片, 但是连接的墨水屏是 2.9 寸的, 这时候墨水屏并无法显示整张图片的信息, 所以在选择图像处理算法的时候, 算法会自动从左上角开始截取一部分图片传到墨水屏显示, 这里设置 x 和 y 可以自定义截取的起始位置。w 和 h 是当前墨水屏的分辨率大小。

-注意: 如果修改了 x 和 y 的指的话, 需要重新点击一下处理算法生成新的图像

### 6.4 型号选择区域: 这里可以选择你接入的墨水屏型号

### 6.5 图像显示区域: 这里会显示你选择的图片以及处理之后的图像

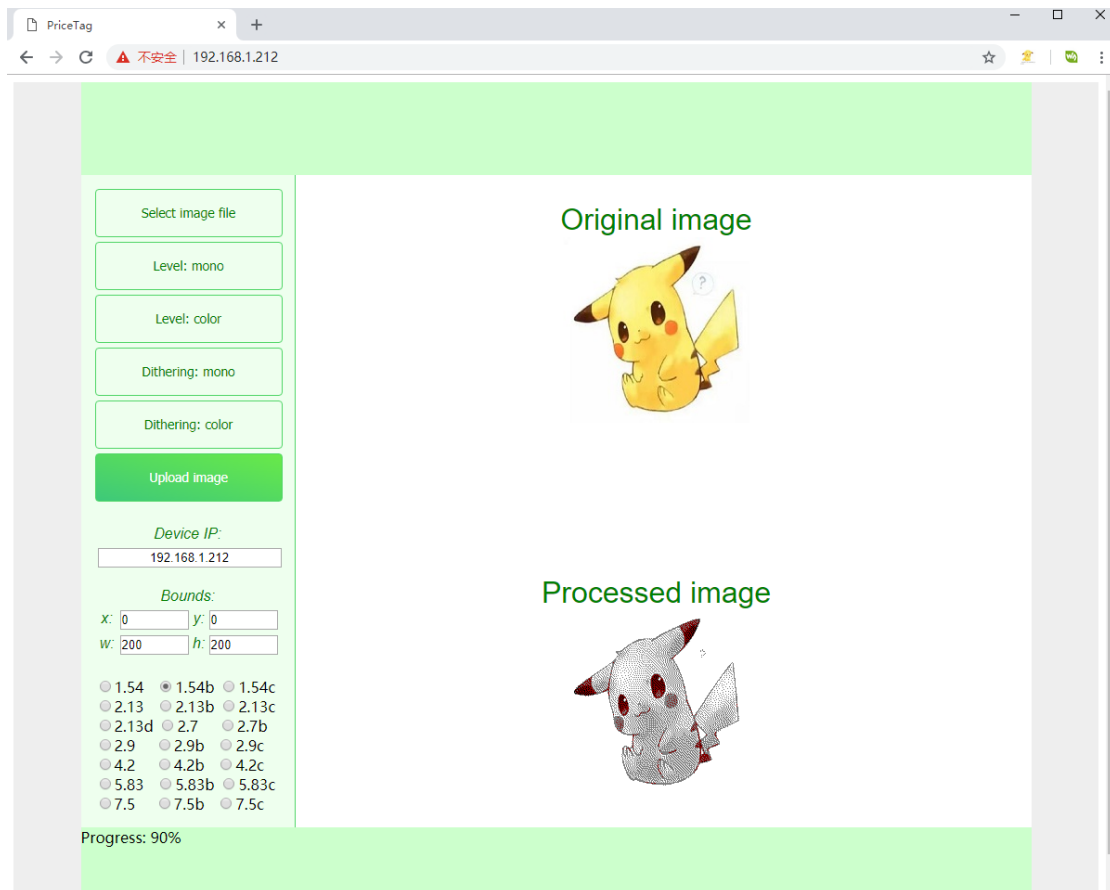
### 6.6 在上传图像的时候, 底部会显示上传的数据进度

7 ①点击 Select image file 选择一张图片, 或者直接将图片拖拽至 Original image 的区域内

8 ④选择对应的墨水屏型号, 例如: 1.54b

9 ①点击一种图像处理算法, 例如: Dithering: color

10 ①点击 Upload image 将图片上传到墨水屏显示。



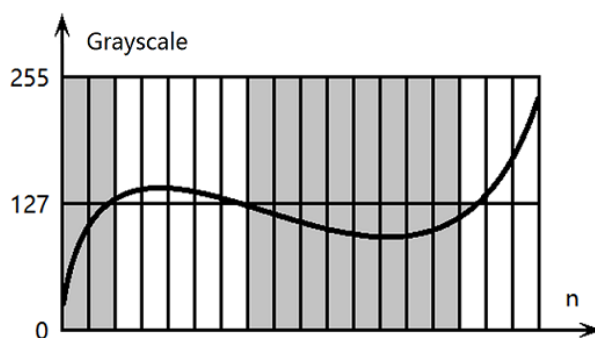
## 图像处理算法

在两个提供的 demo 中，提供了两种图像处理算法，分别是 Level（色阶法）以及 Dithering（抖动法）

### 色阶法

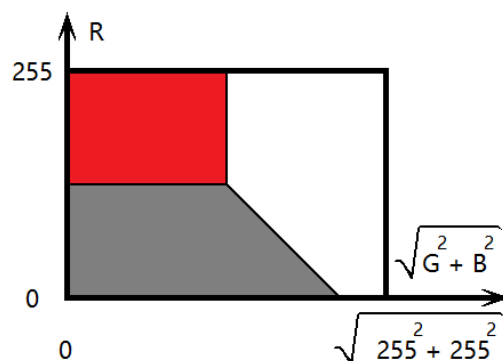
一张图像，我们可以把它划分为几个大的颜色域，图像上的每个像素点根据颜色跟这几个色域的趋近程度，被划分到这些颜色域中去。这种方法比较使用于颜色不多的图像，例如亮色或者三色的形状或者文字图像。以黑白红三色墨水屏为例，处理图像的时候我们希望把他处理成黑白红三色，因此对于一张图像来说，我们可以把图像的所有颜色划分三个大的颜色区域，黑色区域，白色区域，红色区域。

比如根据下图，如果灰度图中的某个像素点的值等于或者小于 127 的话，我们把这个像素点视为黑色像素，否则，就是白色。



灰度图的灰度值和灰度的关系曲线

对于彩色图像来说，我们都知道有 RGB 三色通道，相对于红色通道来说，我们可以把蓝色和绿色统称为 蓝-绿通道的，或者是非红通道。根据下面的图，彩色图像上的某个像素点，如果它红色通道的值很高，但是蓝-绿通道的值很低的话，我们将他归为红色像素，如果说它红色通道和蓝-绿通道的值都很低的话，我们将它归为黑色像素，红色和蓝-绿通道值都很高的话我们把它归为白色。



算法中，对于颜色的定义是根据 RGB 值以及预期颜色值的平方和的差值计算的。其中预期颜色值是指的像素点最趋近的那个颜色值，这些值被保存在 curPal 数组中。

```
// Returns the discrepancy between given (r, g, b)
// and available colors
function getErr(r, g, b, avlCol)
{
    r -= avlCol[0];
    g -= avlCol[1];
    b -= avlCol[2];
    return r*r + g*g + b*b;
}

// Returns the index of available color
// which has minimal discrepancy with the given one
function getNear(r,g,b)
{
    var ind=0;
    var err=getErr(r,g,b,curPal[0]);

    for (var i=1;i<curPal.length;i++)
    {
        var cur=getErr(r,g,b,curPal[i]);
        if (cur<err){err=cur;ind=i;}
    }

    return ind;
}
```

## 抖动法

对于那些颜色比较多，或者渐变区域比较多的图像，上面的色阶法并不太适合，很多时候图像里面的渐变区域的像素可能跟所有颜色域都很接近，如果用色阶法的话就会让图像丢失很多图像细节。很多摄像头拍摄的图片，通过混合颜色的方法来绘画阴影和过度区域，这些图像中，渐变区域占了大部分。

对于人眼来说，很容易把特别小的颜色混淆了，比如两种颜色红和蓝并列，如果把它缩小到足够小的手，在人眼看来会变成一种由红和蓝混合而成的颜色。人眼的缺陷意味着我们可以



通过欺骗人眼，利用“混合”的方法来获取更多可以表现的颜色，抖动算法就是采用了这一种现象。

提供的 demo 中我们使用了 Floyd-Steinberg 抖动算法-基于错误扩散（由 Robert Floy 和 Louis Steinberg 在 1976 年发表）。公式是根据下面的图像的方式进行错误扩散

$$\begin{array}{ccc} & X & 7 \\ 3 & 5 & 1 \\ (1/16) \end{array}$$

X 就是错误（原始颜色和灰度值（颜色值）之间的一个标量（矢量）差值），这个错误会向右边，右下，下边，和左下四个方向扩散，分别以  $7/16$ ， $1/16$ ， $5/16$  和  $3/16$  的权重添加到这四个像素点的值中去。

#### 两种算法的处理效果图



原图



“黑白色阶处理” 和 “三色色阶处理”。



“黑白抖动处理” and “三色抖动处理” .

## 数据通信协议

程序处理数据的时候，是将图像数据分包，然后再分发送给 ESP32 模块。如果你对于 WiFi 的相关协议不是很理解或者不想深入了解，只是想简单的在 HTML 页面上操作，通过 wifi 将图像数据传送到墨水屏上刷新的话，可以直接使用我们的例程，并且参照上面的[使用说明](#)。

我们提供的 wifi 例程是基于 POST 应答数据传输协议的，你可以简单的修改他，添加你自己的指令或者函数。

### 指令

在例程中，主要有 4 个指令：

- EPDn - 初始化 n 型号的屏幕
- LOAD - 装载图像数据（黑色或者红色）
- NEXT - 从红色通道切换到黑色通道
- DONE - 刷新屏幕并设置屏幕为睡眠模式

### 初始化

根据下面的流程图中，“Upload image” 会创建一个对象，发送指令并且监听服务器应答信息，然后发送 EPDn 指令。驱动板在接收到 EPDn 指令之后，会初始化墨水屏，并且是能单色或者黑色通道等待数据写入。

对于黑白屏幕，EPDn 和 NEXT 指令写入的内存为：

***EPD\_SendCommand(0x24); //WRITE\_RAM***

对于黑白红屏幕，写入的内存为（分为黑色和红色通道）

***EPD\_SendCommand(0x10); //DATA\_START\_TRANSMISSION\_1,***

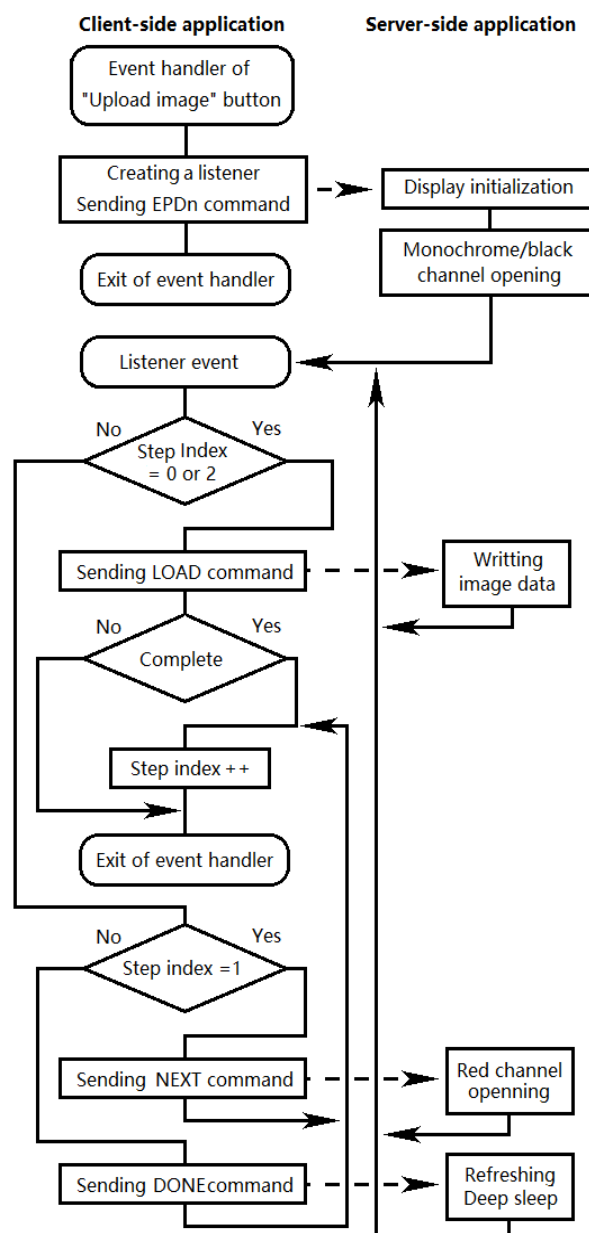
***EPD\_SendCommand(0x13); //DATA\_START\_TRANSMISSION\_2,***

其中有一些特殊情况：

- 2.7inch 和 4.2inch 的黑白屏幕，写入的是红色通道 (0x13)
- 7.5inch 三色屏幕是同时写入黑色和红色数据的 (其他是先黑色，后红色)

【注意】在移植程序的时候需要注意不同型号写入的内存地址不同。并且在向墨水屏写入图像数据前，都要先运行 WRITE\_RAM, DATA\_START\_TRANSMISSION\_1 或者

DATA\_START\_TRANSMISSION\_2 指令。但是，2.13inch 屏幕可以一行一行的写入数据， 这也就是说，在每次写入行数据前，都要执行依次 WRITE\_RAM 指令。



## 图像数据格式

一般图片的数据格式是 24bpp, 经过算法处理之后会将数据格式降低为:

1bpp (黑白屏幕):

0 – 白色;

1 – 黑色;

2bpp (三色屏幕):

0 – 白色;

1 – 黑色;

2 – 灰色 (只有 1.54inch e-Paper B 支持)

3 – 红色

由于在传输数据的时候, 为了方便转换和传输, 会简化图像数据格式, 跟屏幕内部寄存器所支持的格式会不想动, 所以在发送给墨水屏显示之前, 要再一次对图像数据进行调整扩展

屏幕型号 (通道)	传输 (p – pixel, b - bits)	墨水屏
1.54 (黑色)	p: 01234567 - b: 76543210	p:01234567 - b:76543210
1.54b (红色)	0 – 黑色 or 红色; 1 – 白色.	0 – 黑色 or 红色; 1 – 白色.
2.13 (黑色)		备注: 传输格式和墨水屏内存支持格式相同
2.13b		
2.7b		
2.9		

2.9b  4.2 (黑色)  4.2b		
2.7	<p>p: 01234567 - b:76543210</p> <p>0 – 黑色;</p> <p>1 – 白色.</p>	<p>p:01234567 - b:76543210</p> <p>0 – 白色;</p> <p>1 – 黑色 .</p> <p><b>备注:</b> 颜色翻转了</p>
7.5	<p>p:01234567 - b:76543210</p> <p>0 – 黑色 or 红色;</p> <p>1 – 白色.</p>	<p>p:0 - b:7,6,5,4 (7 is high, 4 is low);</p> <p>p:1 – b:3,2,1,0;</p> <p>p:2 – b:15,14,13,12;</p> <p>p:3 – b:11,10,9,8;</p> <p>0000 (0) – 黑色;</p> <p>0011 (3) – 白色.</p>
1.54b (黑色)	<p>p:0 – b:1,0;</p> <p>p:1 – b:3,2;</p> <p>p:2 – b:5,4;</p> <p>p:3 – b:7,6;</p> <p>00 (0) – 黑色;</p> <p>01 (1) – 白色;</p> <p>10 (2) – 灰色;</p>	<p>p:0 – b:7,6;</p> <p>p:1 – b:5,4;</p> <p>p:2 – b:3,2;</p> <p>p:3 – b:1,0;</p> <p>00 (0) – 黑色;</p> <p>01 (1) – 灰色;</p> <p>11 (3) – 白色;</p>

	11 (3) – 红色, is read as 10 (2).	
7.5b	<p>p:0 – b:1,0;</p> <p>p:1 – b:3,2;</p> <p>p:2 – b:5,4;</p> <p>p:3 – b:7,6;</p> <p>00 (0) – 黑色;</p> <p>01 (1) – 白色;</p> <p>10 (2) – 灰色;</p> <p>11 (3) – 红色.</p>	<p>p:0 - b:7,6,5,4 (7 is high, 4 is low);</p> <p>p:1 – b:3,2,1,0;</p> <p>p:2 – b:15,14,13,12;</p> <p>p:3 – b:11,10,9,8;</p> <p>0000 (0) – 黑色;</p> <p>0011 (3) – 白色;</p> <p>0100 (4) – 红色.</p>