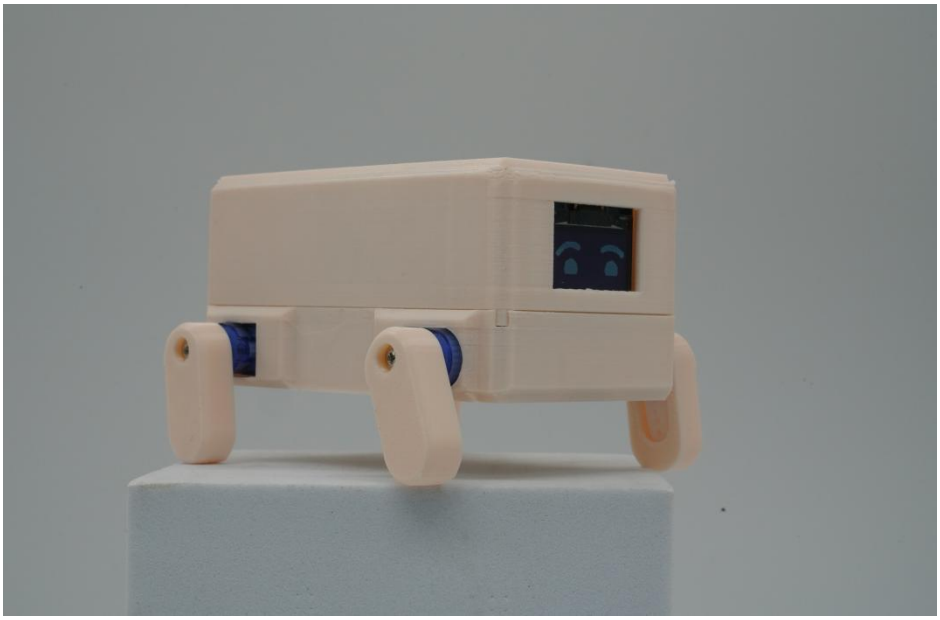
EDA-Robot项目开发文档



编写人	版本记录	修改信息
杨哲	V1.0	初版发布
杨哲	V1.1	添加电池电量检测功能
杨哲	V1.2	新增支持180度舵机
杨哲	V1.3	同步V1.3版型

1 项目介绍

1.1 项目简介

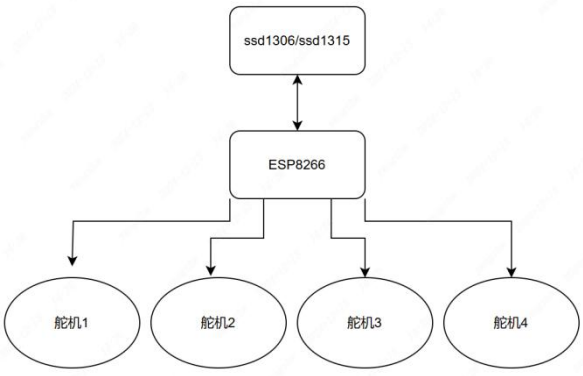
EDA-Robot项目是一个基于ESP8266开发的简单桌面机械宠物项目，能够实现手机遥控，天气显示、时间显示、表情显示、随机运动、坐下、趴下等功能，该项目大部分器件使用易焊接的插件封装，部分使用大尺寸的贴片封装，非常适合新手复刻及学习。在保证易焊接的前提下尽可能的缩小体积，使其既可以作为桌面宠物，也可以充当桌面信息显示器。该项目能帮助使用者深入理解桌面小电视等相关硬件的设计思路和电路原理。

1.2 性能指标

- 采用ESP8266主控，支持2.4G信号WIFI热点控制。
- 采用4个360度舵机，使用PWM实现多角度自定义控制。
- 支持SSD1306,SSD1315驱动屏幕，实现表情及信息显示。
- 硬件采用ESP8266主控，支持2.4G信号WIFI热点控制。
- 支持双14500电池，使用2个LDO独立降压，支持外部充电模块连接。

2 设计

2.1 系统方案



硬件方向的系统方案较为简洁，外设一共就1个屏幕和4个电机

2.2 电路分析

该项目在硬件设计中涉及的主要电路模块包括：

(1) 主控电路：作为项目的核心控制单元，负责处理按键输入、驱动显示屏，并通过内置Wi-Fi模块实现网络连接，用于获取实时信息和远程更新数据。

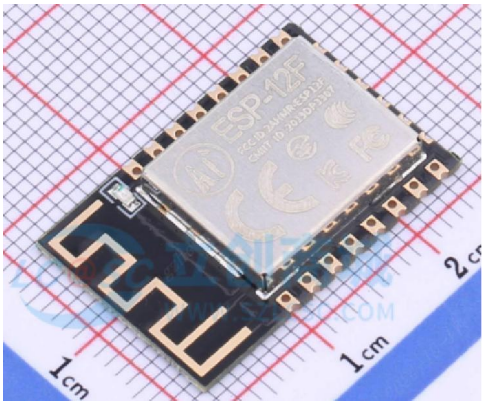
(2) 舵机电路：作为项目的运动部分，负责控制舵机，确保电机正常驱动。

(3) 电源电路：为整个电路系统提供稳定的电源输出，确保所有模块在合适的电压下稳定运行。

总体而言，该项目的硬件部分设计较为简单，主要通过软硬件结合实现信息显示与运动交互控制。

2.2.1 主控电路

（1）ESP8266模组外部电路



项目采用了乐鑫科技的ESP8266芯片模组作为主控。ESP8266芯片以其高集成度、优异的RF性能和低功耗著称，集成了完整的Wi-Fi功能，非常适合用于物联网应用的开发。其内置的Tensilica L106 32位RISC处理器，最高工作频率可达160 MHz，支持实时操作系统（RTOS）和Wi-Fi协议栈，使其在处理数据传输及任务管理时具备极强的灵活性和高效性。

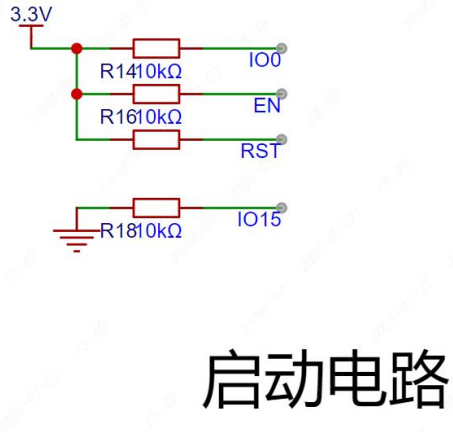
注意：

（1）、模组外围电路，GPIO0 必须上拉到 VCC，GPIO15 必须下拉到 GND。

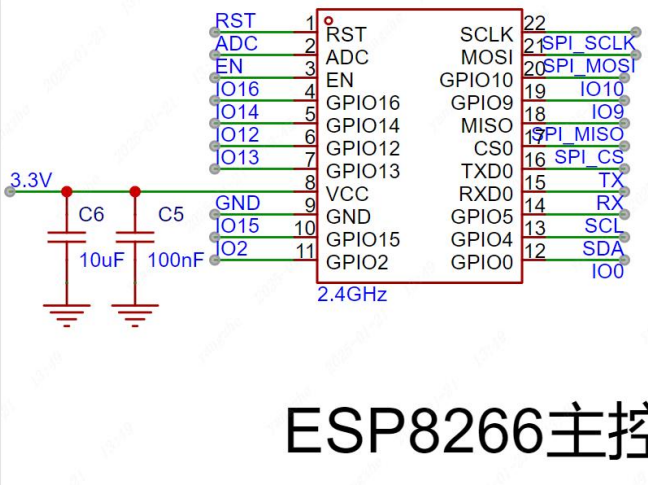
（2）、EN 脚和 RST 脚必须上拉到 VCC。

（3）、模组的 pin9-pin14 不可用。

该模组的电路设计遵循官方推荐的设计规范。在GPIO0、GPIO2、RST、EN等引脚上拉10K电阻，以确保这些引脚在芯片启动时保持高电平，从而确保芯片能够正常工作。CS片选引脚则通过10K下拉电阻保持低电平，确保模块通信的稳定性。通过参考ESP8266的数据手册和官方的应用设计图，在嘉立创EDA中我们设计的电路最终如下图所示：



启动电路



ESP8266主控

（2）硬件IIC通信

Arduino / variants / nodemcu / pins_arduino.h

Code Blame 54 lines (42 loc) · 1.61 KB

```
23      $Id: wiring.h 249 2007-02-03 16:52:51Z mellis $
24    */
25
26    #ifndef Pins_Arduino_h
27    #define Pins_Arduino_h
28
29    #define PIN_WIRE_SDA (4)
30    #define PIN_WIRE_SCL (5)
31
32    static const uint8_t SDA = PIN_WIRE_SDA;
33    static const uint8_t SCL = PIN_WIRE_SCL;
34
```

另外,在ESP8266的库函数定义中我们可以知道该芯片的硬件IIC引脚分别为GPIO4-SDA、GPIO5-SCL为了让屏幕刷新更快，显示效果更佳，我们选用硬件IIC为屏幕通信。

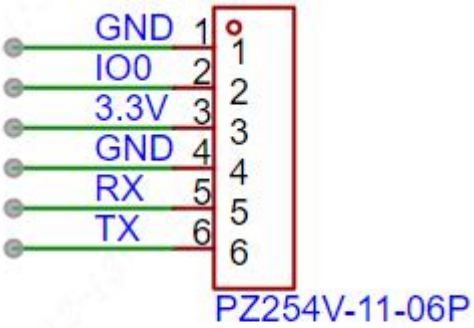
（3）模式切换电路

表 模组启动模式说明

模式	CH_PD(EN)	RST	GPIO15	GPIO0	GPIO2	TXD0
下载模式	高	高	低	低	高	高
运行模式	高	高	低	高	高	高

注意：部分引脚已经内部上拉，请参考原理图

ESP8266的启动模式包括运行模式和下载模式，为确保能够顺利烧录固件并调试程序，电路设计中引出了TX、RX、GND和3.3V引脚作为串口通信接口，同时增加了IO0和GND引脚用于进入下载模式的跳线设计。用户可以通过简单的跳线操作快速切换工作模式，从而实现程序烧录与正常运行。在嘉立创EDA中我们设计的电路最终如下图所示：



烧录接口

至此，我们的主控电路就基本设计完成

2.2.2 交互电路

（1）IIC屏幕显示电路

该项目的显示屏采用了OLED屏幕模组，该模组本身已经集成了驱动电路，因此不需要额外设计复杂的外围电路。

1.5Pin Definition

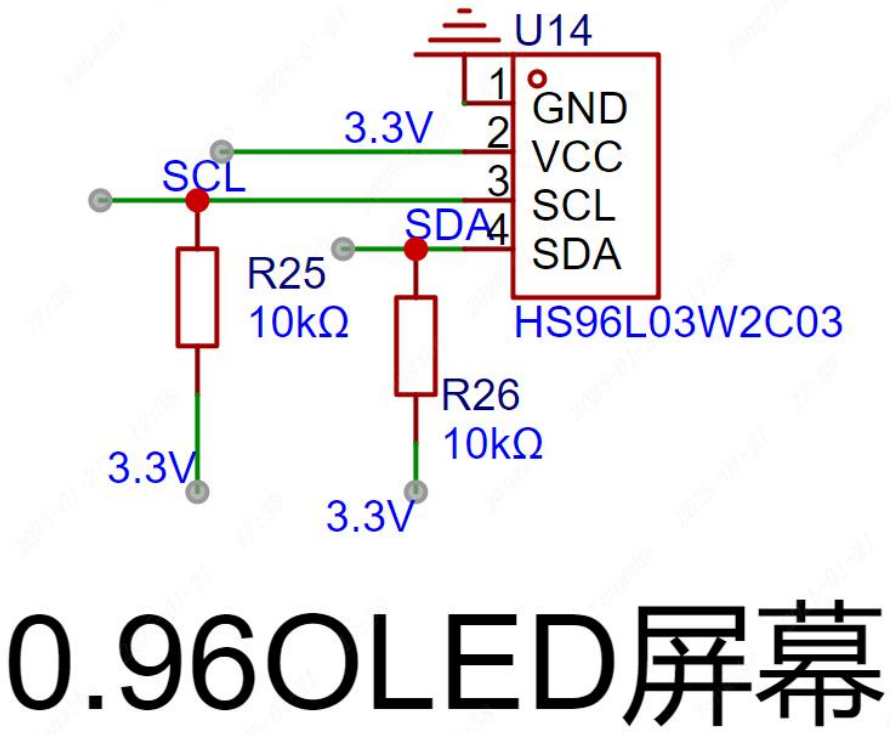
PIN No.	Symbol	Description
1	GND	Power Supply for OLED Ground of Logic Circui This is a ground pin. It must be connected to Ground
2	VCC	Power Supply for OLED This is a voltage supply pin. It must be connected to Source
3	SCL	The serial clock input SCL
4	SDA	The serial data input SDA

2. Absolute Maximum Ratings

Parameter	Symbol	Min	Max	Unit	Notes
Supply Voltage for Display	VCC	3	5	V	1, 2
Supply Voltage for Logic	SCL/SDA	1.65	3.3	V	
Operating Temperature	T_{OP}	-40	80	°C	
Storage Temperature	T_{STG}	-40	85	°C	3
Life Time (120 cd/m²)		10,000	-	hour	4
Life Time (80 cd/m²)		30,000	-	hour	4
Life Time (60 cd/m²)		50,000	-	hour	4

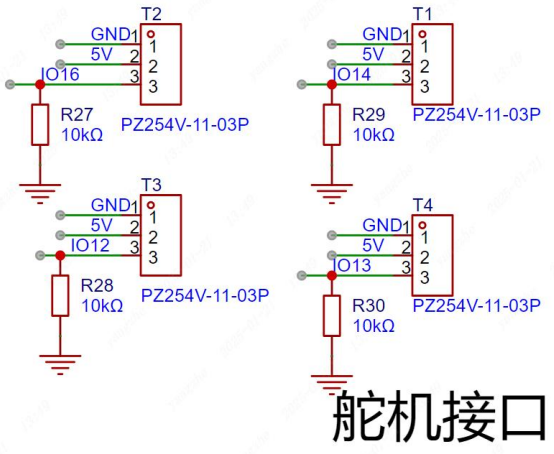
Note 1: All the above voltages are on the basis of "GND = 0V".
Note 2: When this module is used beyond the above absolute maximum ratings, permanent breakage of the module may occur. Also, for normal operations, it is desirable to use this module under the conditions according to Section 3. "Optics & Electrical Characteristics". If this module is used beyond these conditions, malfunctioning of the module can occur and the reliability of the module may deteriorate.
Note 3: The defined temperature ranges do not include the polarizer. The maximum withstood temperature of the polarizer should be 80°C.
Note 4: VCC = 3.3 V, T_a = 25°C, 50% Checkerboard.
Software configuration follows Section 4.4 Initialization.
End of lifetime is specified as 50% of initial brightness reached. The average operating lifetime at room temperature is estimated by the accelerated operation at high temperature conditions.

通过直接将屏幕模组与主控模块的SPI接口相连，即可完成图像和信息的显示。我们通过查阅屏幕模组的技术规格，确认了引脚排列、接口类型以及屏幕的功耗需求，确保其与主控电路的匹配性。在电路设计中，屏幕模组的连接端口采用标准插座接口，简化了电路设计并提高了可维护性和模块化程度。在嘉立创EDA中我们设计的电路最终如下图所示：



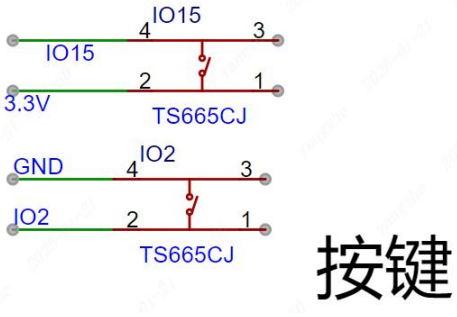
（2）舵机驱动电路

在本项目中为了使用易焊接的插件器件并确保体积较小，所以没有在舵机的PWM信号引脚接下拉电阻等信号优化措施，建议可以增加下拉电阻，确保电机停转和起始状态稳定，在后续的软件调试中也会更方便。



（3）按键电路

按键部分因为接口有限选择的是GPIO15和GPIO2接口，由于ESP12F在开机时GPIO15必须下拉，并接有下拉电阻，所以这里开关选择接入3.3V，当GPIO15输入高电平时触发开关。GPIO2为正常接入GND。



（4）ADC电量检测电路

修改分压器适配 8.4V 到 1V
现在需要适配新的输入电压范围（最大 8.4V）到 ESP8266 的 1.0V ADC 输入。分压比计算如下：

$$分压比 = \frac{1V}{8.4V} = \frac{1}{8.4} \approx 0.119$$

根据分压公式：

$$\frac{R_2}{R_1 + R_2} = 0.119$$

假设保持100k ，计算 ：

$$\frac{100k}{R_1 + 100k} = 0.119$$

$$R_1 + 100k = \frac{100k}{0.119} \approx 840k$$

$$R_1 \approx 740k\Omega$$

检验新设计
对于 ，输出电压：

$$V_{out} = 8.4 \times \frac{100k}{740k + 100k} = 8.4 \times \frac{100}{840} \approx 1.0V$$

对于电压较低时（如 4.2V），输出电压为：

$$V_{out} = 4.2 \times \frac{100k}{740k + 100k} = 4.2 \times \frac{100}{840} \approx 0.5V$$

分压电路成功将 8.4V 的输入电压压缩到 0-1V 范围内。

2.2.3 电源电路

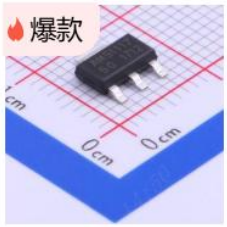
（1）舵机电源供电电路

依照舵机的数据手册，我们可以知道SG90舵机的驱动电压是4.8-6V，当然，在测试中其实3.3V也能驱动舵机，但是低电压驱动会导致舵机扭矩不足，动力较差，甚至用手轻挡就会让电机停止转动，由于我们的EDA-Robot是需要背负2节14500电池，需要承载相应的重量，所以这里我们选择使用5V驱动

Specifications

- Weight: 9 g
- Dimension: 22.2 x 11.8 x 31 mm approx.
- Stall torque: 1.8 kgf·cm
- Operating speed: 0.12 s/60 degree
- Operating voltage: 3.3 V (~6V)
- Dead band width: 1 μs
- Temperature range: 0 °C – 55 °C

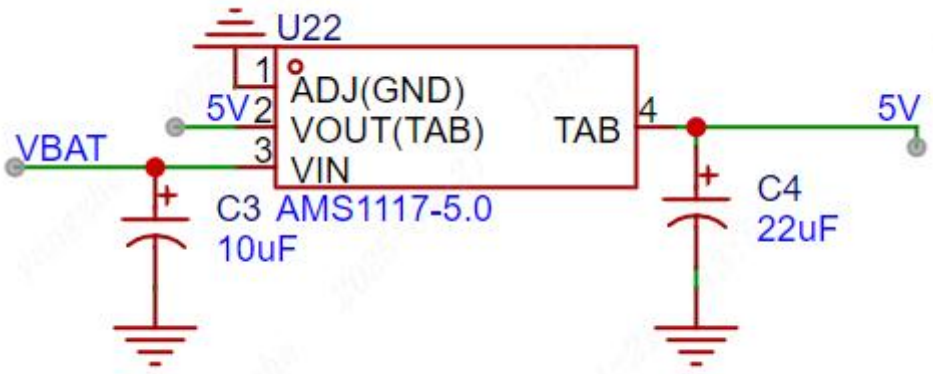
既然选择5V驱动，所以我们选择了AMS1117-5V固定输出LDO，我们只要确保输入LDO的电压在5-18V即可



AMS1117-5.0
品牌: UMW(友台半导体)
封装: SOT-223
输出类型: 固定
工作电压: 18V
输出电压: 5V
输出电流: 1A
领券 最高减150
嘉立创贴片惊喜价格(库存93K+)

（2）电池选择

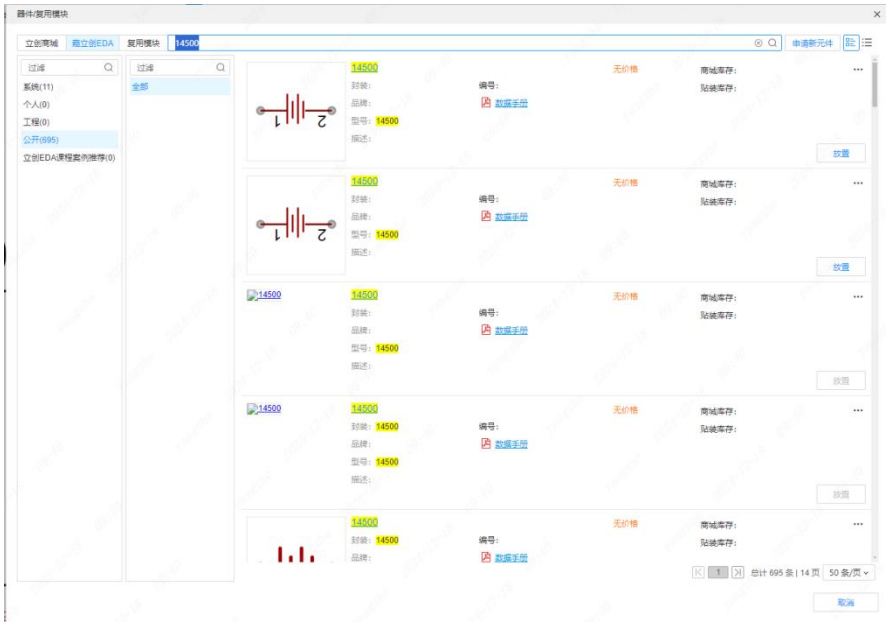
前面我们选择了AMS1117-5V版本，所以我们要确保输入电压5-18V，为了简化电路，缩小体积，提升安全性，这里我们选取14500钢壳锂电池。单节电池范围在4.2-2.75V，那么由串联分压可知我们可以串联2节14500电池，这样电池范围就在5.5-8.4V，在AMS1117-5V输入范围内，这样既能得到高容量，又能省去升压电路，极为方便。



4. 产品规格 Product Specifications

项目 Item	规格 Specification
4.1 标称容量 4.1 Nominal Capacity	500mAh (0.5C 放电容量, 4.20~2.75V) 500mAh (0.5C discharge, 4.20~2.75V)
4.2 最小容量 4.2 Minimum Capacity	450mAh (0.5C 放电容量, 4.20~2.75V) 450mAh (0.5C discharge, 4.20~2.75V)
4.3 标称电压 4.3 Nominal Voltage	3.7 V
4.5 最大充电电流 4.5 Max. Charge Current	500mA (Ambient temperature 25℃)
4.6 最大放电电流 4.6 Max. Discharge Current	1000mA (Ambient temperature 25℃)

由于商城没有14500电池，所以这里我们使用在线公开库的14500电池，当然你也可以自己设计一个符号和封装。



（3）主控电源供电电路

由esp8266的数据手册可知，主控工作电压在3.3V，那么我们选择AMS1117-3.3V固定输出LDO就可以了

电气特性

参数		条件	最小值	典型值	最大值	单位
供电电压		VDD	3.0	3.3	3.6	V
I/O	V_{IL}/V_{IH}	–	-0.3/0.75VIO	–	0.25VIO/3.6	V
	V_{OL}/V_{OH}	–	N/0.8VIO	–	0.1VIO/N	V
	I_{MAX}	–	–	–	12	mA

4、供电

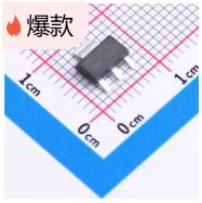
（1）、推荐 3.3V 电压，峰值 500mA 以上电流

（2）、建议使用 LDO 供电；如使用 DC-DC 建议纹波控制在 30mV 以内。

（3）、DC-DC 供电电路建议预留动态响应电容的位置，可以在负载变化较大时，优化输出纹波。

（4）、3.3V 电源接口建议增加 ESD 器件。

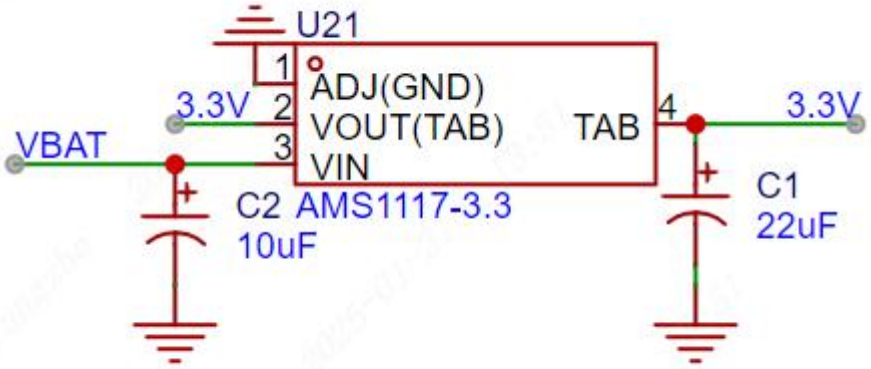
这里我们选择了AMS1117-3.3V，支持3.3V-18V输出，在ESP8266的数据手册中要求供电电流要高于500mA,为了确保电流足够，我们直接从14500取电，确保舵机和主控的供电互不干涉。



AMS1117-3.3
品牌: UMW(友台半导体)
封装: SOT-223
输出类型: 固定
工作电压: 18V
输出电压: 3.3V
输出电流: 1A
类目: 线性稳压器(LDO)
编号: C347222
详细: 数据手册

领券 最高减150 嘉立创贴片惊喜价格(库存629K+) SMT扩展库

查看更多渠



3 PCB设计

3.1 原理图设计

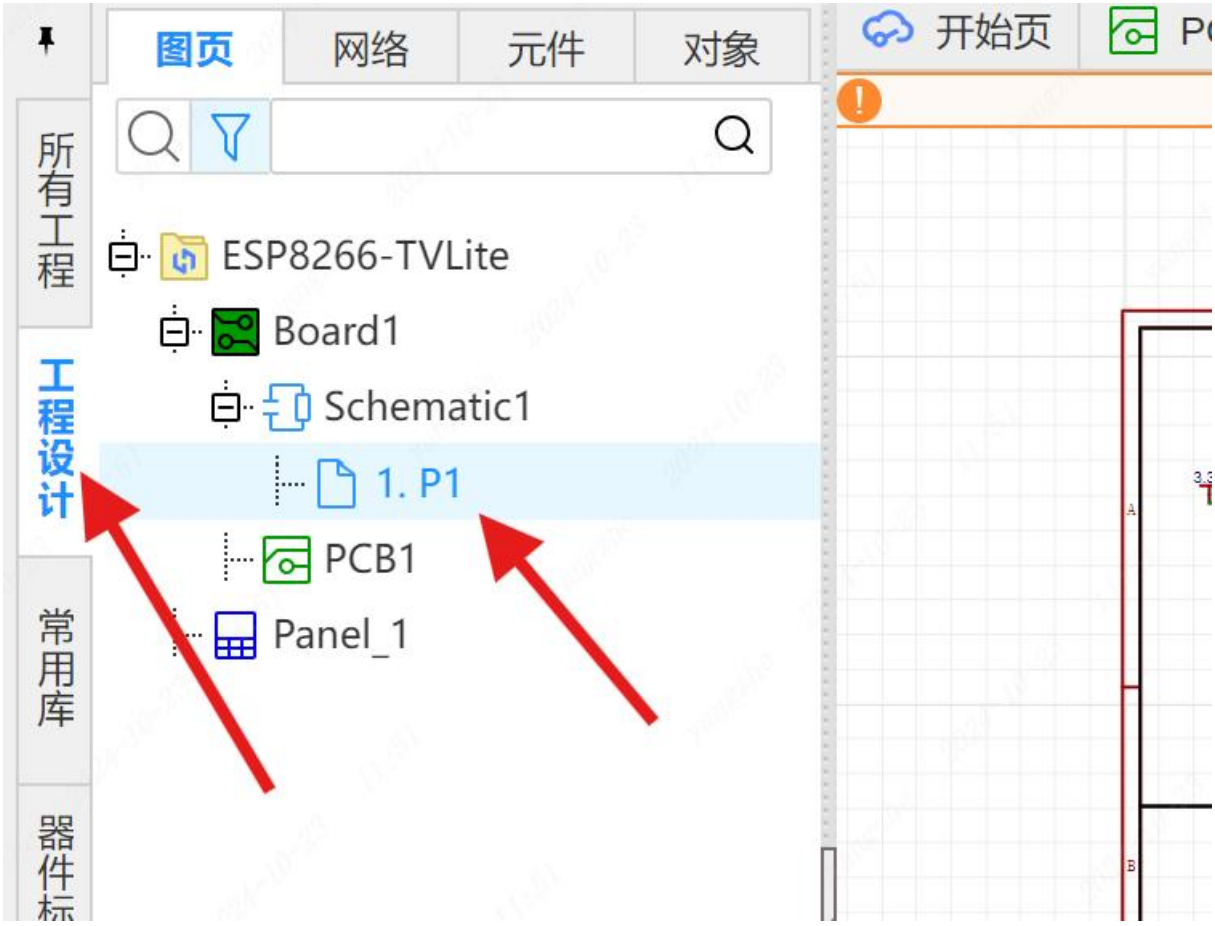
在了解完示波器硬件电路原理后接下来进行原理图与PCB设计环节，原理图设计部分包含了元器件选型、元器件搜索以及原理图整理的内容；PCB设计部分包含边框设计、元器件分类布局、PCB走线与设计检查、PCB生产与打样等内容。

3.1.1 原理图设计

（1）工程创建

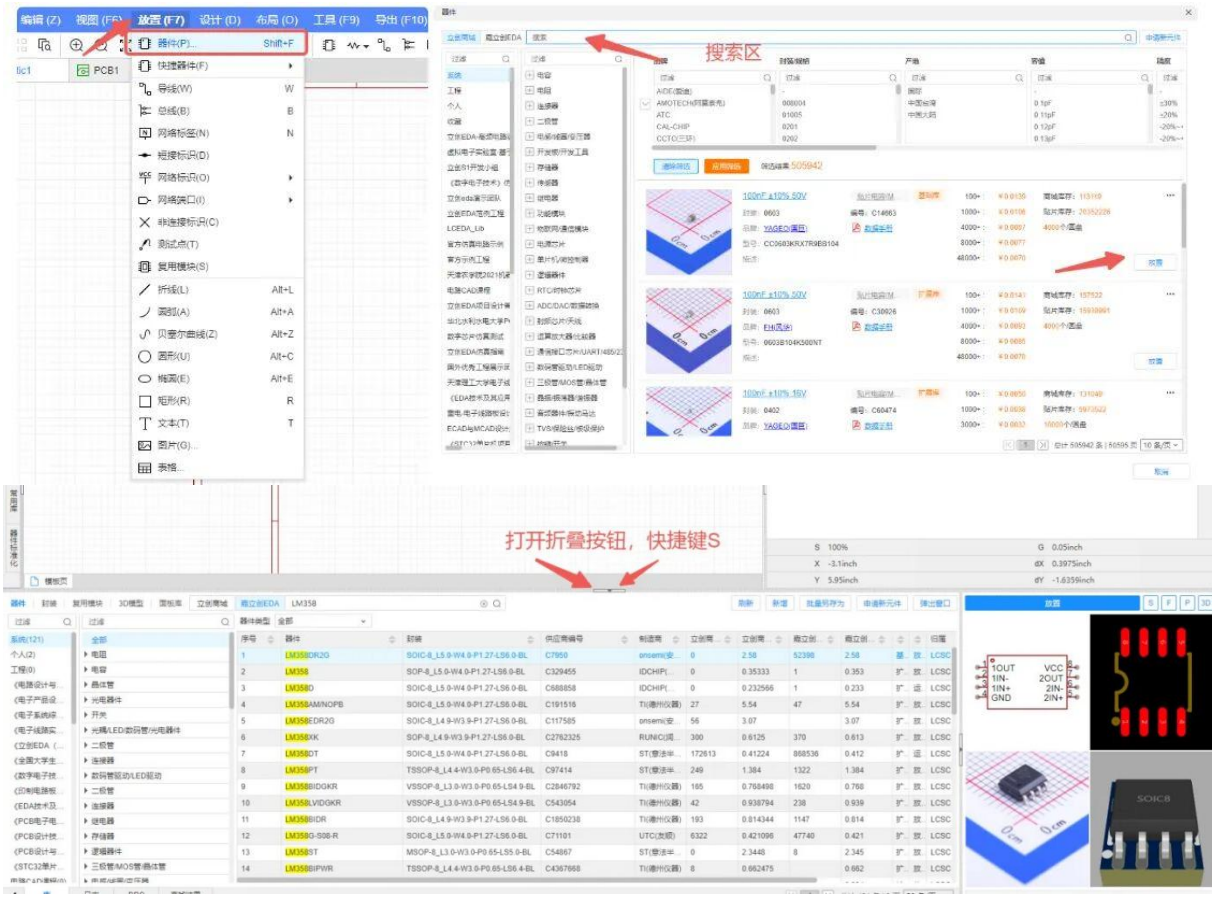
打开嘉立创EDA专业版软件（https://pro.lceda.cn/editor），登录账号后选择创建工程， 输入工程名称：【仪器仪表】基于GD32的简易示波器设计，系统会自动创建一个工程项目，接下来就在该项目中完成示波器的原理图与PCB、3D外壳与面板设计的内容。

创建好工程后打开左侧工程列表中的Schematic1图页，在右侧画布区域放置元器件进行连接，底下的PCB1是用来绘制PCB图的页面，设计流程是先完成原理图的设计后再到PCB设计。



（2）元器件搜索

元器件搜索的方式有三种途径，第一种可以在左侧的常用库中找到官方提供的参考库进行放置，优点是比较方便，缺点是库种类较少；第二种方式是通过顶部菜单栏中的放置按钮选项，可以看到器件实物图与参考价格、数据手册等信息；第三种方式是在软件底部面板中搜索放置器件，这种方式的好处是可以看到所选器件的符号、封装与3D模型效果图。

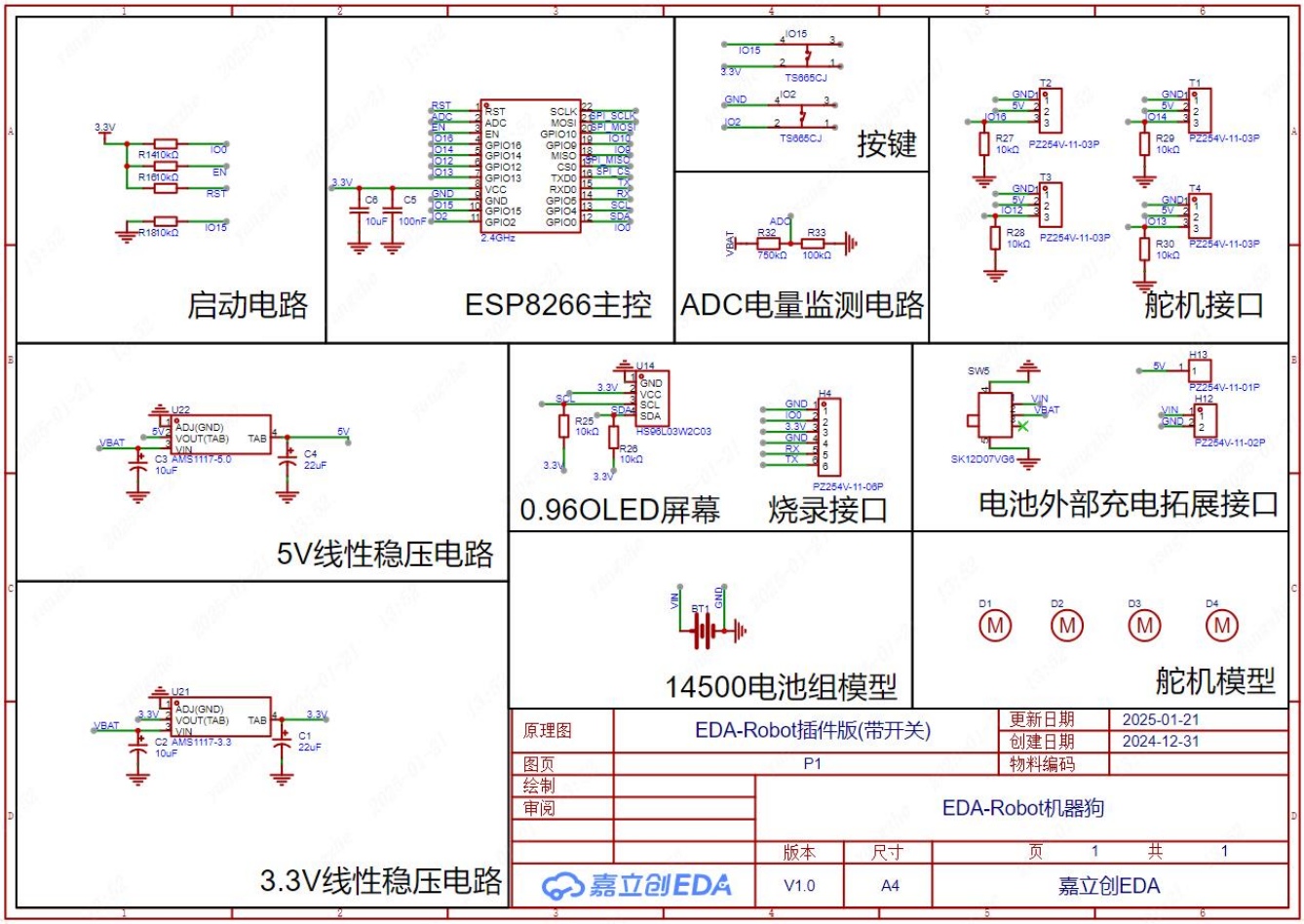


（3）元器件放置

为了方便初学者学习，该项目提供了完成电路图中所需的元器件清单，可以直接根据器件清单中提供的器件编号及备注信息进行元器件的搜索和放置。以底部面板搜索器件为例，在原理图工作区打开底部库面板，选择器件类目，在搜索栏中输入器件编号，进行搜索找到对应元器件点击放置在画布中即可。里面的M3铜柱在EDA左侧常用库的最后一项可以找到并放置。

（4）原理图整理

完成元器件放置后接下来进行电路图的连接与整理工作，参照以下电路图，完成元器件间的连接，使用网络标签可以替代导线连接，两个相同网络的位置放置相同网络标签即可。接下来按功能模块划分各个电路，使用矩形边框包围住各个电路模块，并用文本加上电路模块标识说明。最好使用设计菜单栏下的检查DRC功能查看电路连接是否有误。



完整原理图如上图所示，需要注意的是模型部分的原理图符号无实际意义，仅用于生成3D模型，方便外壳绘制。

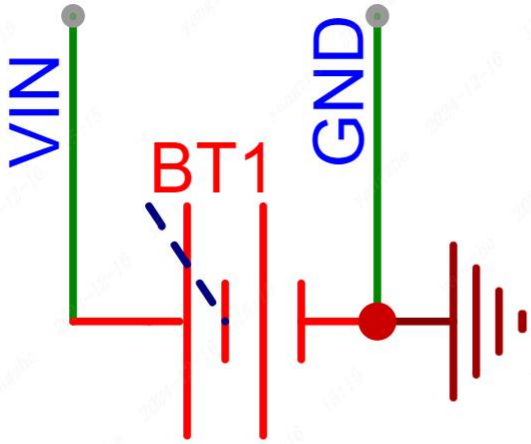
EDA-Robot项目物料清单

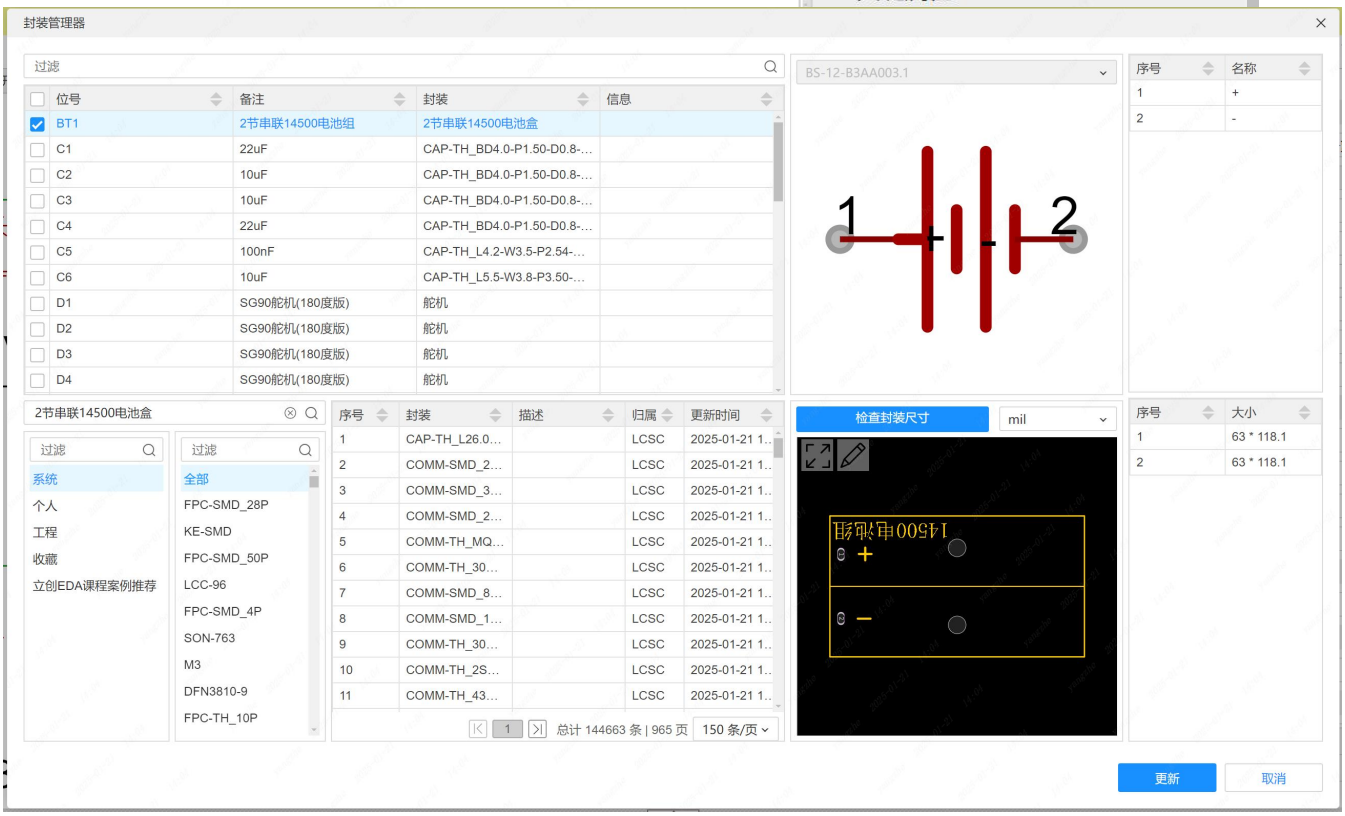
序号	名称	参数	器件位号	数量	封装	商品编号
1	电池盒	2节串联14500电池组	BT1	1	2节串联14500电池盒	
2	电容	10uF	C1,C2,C3	3	CAP-TH_BD4.0-P1.50-D0.8-FD	C43351
3	电容	22uF	C4	1	CAP-TH_BD4.0-P1.50-D0.8-FD	C43840
4	电容	100nF	C5	1	CAP-TH_L4.2-W3.5-P2.54-D0.5	C5632430
5	电容	10uF	C6	1	CAP-TH_L5.5-W3.8-P3.50-D0.6	C2761733
6	舵机	SG90舵机(180度版)	D1,D2,D3,D4	4	舵机	
7	烧录接口	PZ254V-11-06P	H4	1	HDR-TH_6P-P2.54-V-M	C492405
8	充电接口	PZ254V-11-02P	H12	1	HDR-TH_2P-P2.54-V-M	C492401
9	5V拓展口	PZ254V-11-01P	H13	1	HDR-TH_1P-P2.54-V-M	C492400
10	按键	TS665CJ	IO2,IO15	2	SW-TH_4P-L6.0-W6.0-P4.50-LS6.5	C393938
11	电阻	10kΩ	R14,R16,R17,R18,R25,R26,R27,R28,R29,R30	10	RES-TH_BD2.4-L6.3-P10.30-D0.6	C410695
12	电阻	750kΩ	R32	1	RES-TH_BD2.4-L6.3-P10.30-D0.6	C173043
13	电阻	100kΩ	R33	1	RES-TH_BD2.4-L6.3-P10.30-D0.6	C173137
14	开关	SK12D07VG6	SW5	1	SW-TH_SK12D07VG6	C2681575
15	舵机排针	PZ254V-11-03P	T1,T2,T3,T4	4	HDR-TH_3P-P2.54-V-M	C2937625
16	ESP12F	2.4GHz	U10	1	WIFIM-SMD_ESP-12F-ESP8266MOD	C82891
17	0.96屏幕	HS96L03W2C03	U14	1	OLED-TH_L27.8-W27.2-P2.54_C9900033791	C5248080
18	LDO-3.3V	AMS1117-3.3	U21	1	SOT-223-4_L6.5-W3.5-P2.30-LS7.0-BR	C347222
19	LDO-5V	AMS1117-5.0	U22	1	SOT-223_L6.7-W3.5-P2.30-BR	C347223

3.2 PCB设计

3.2.1封装管理

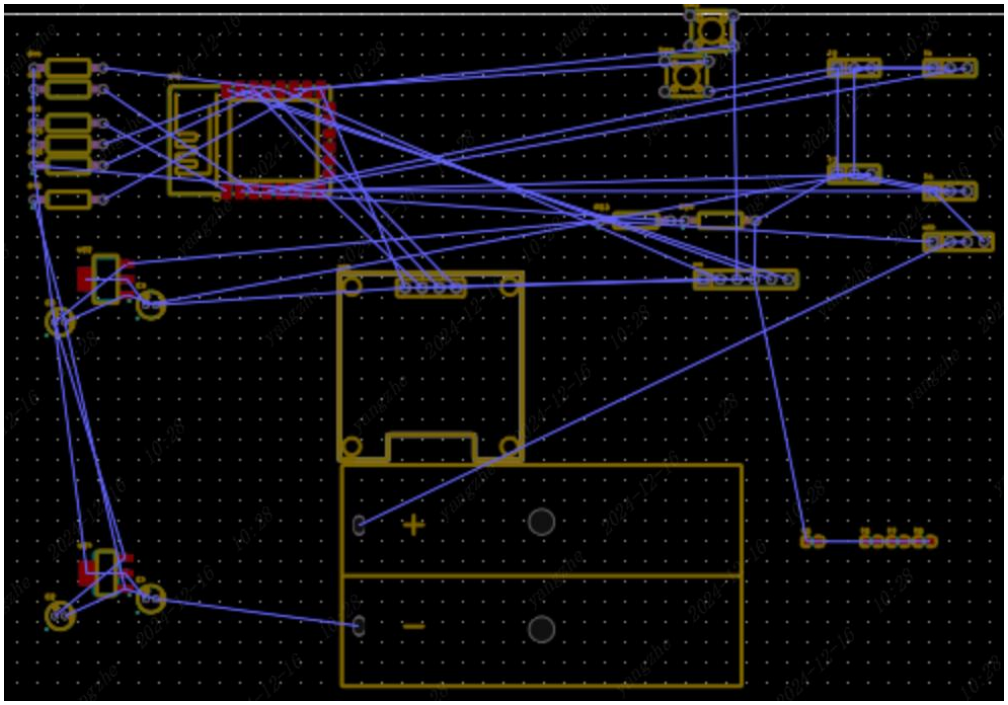
在开始PCB设计之前，将14500电池的第三方符号封装替换成14500电池盒封装，选中LED，在右侧点击封装选项。搜索14500电池盒*2，选择这种只有2P接口的串联14500电池盒即可。





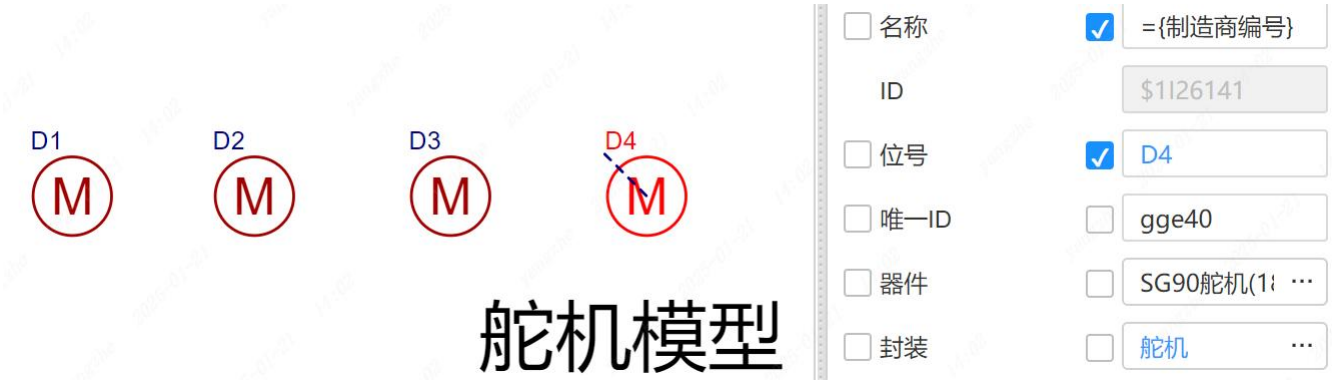
3.2.2电路模块分类

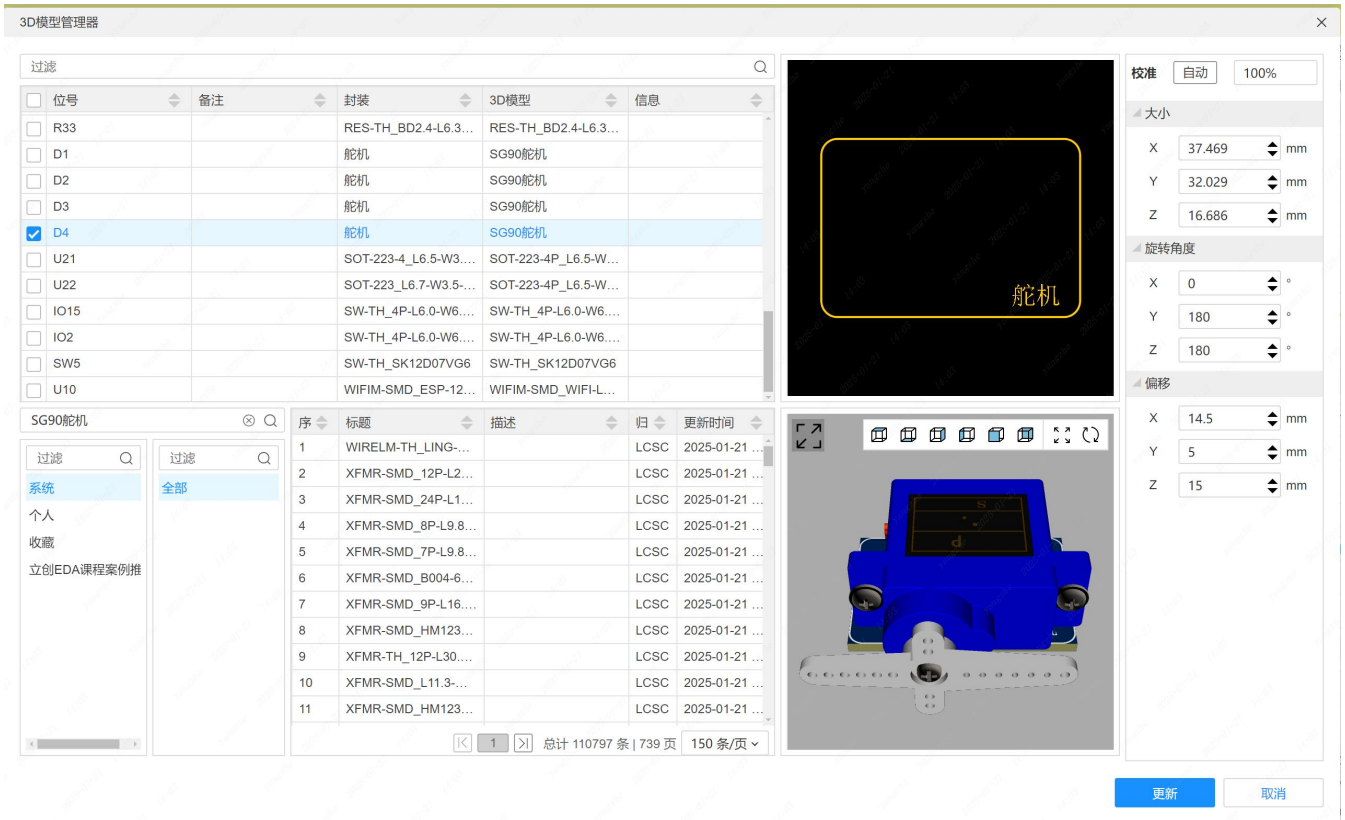
将原理图生成PCB后接下来进行元器件的布局与走线，刚转到PCB画布时元器件摆布是比较杂乱的，首先要做的是将元器件按电路功能进行分类，分类的方式是先在原理图页面对各个电路模块进行单独框选，然后选择“设计”菜单栏下的“布局传递”功能，传送到PCB将对应的元器件提取出来重新摆放，这一步是分类的关键。

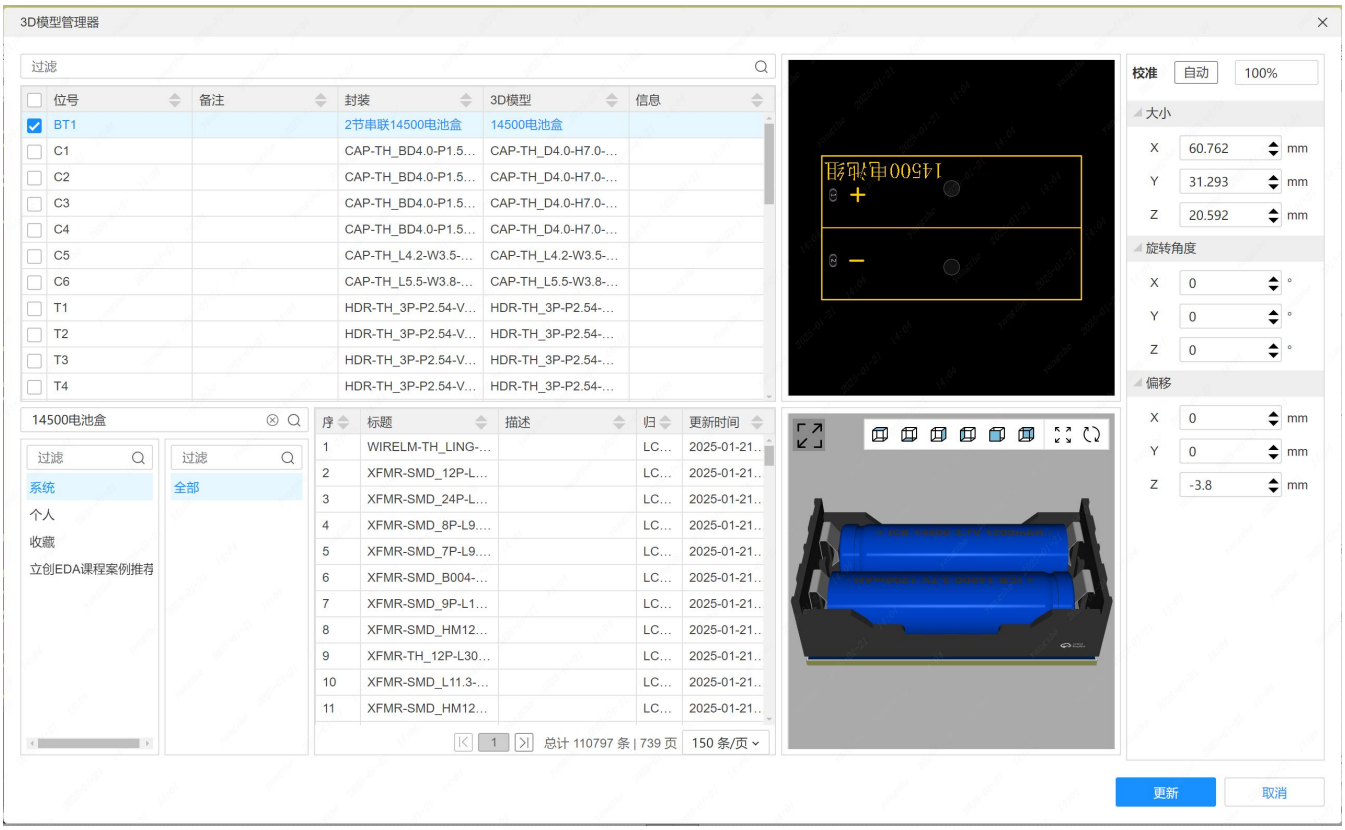


3.2.3封装3D模型设置

在本项目中，绘制了几个图标和封装充当舵机模型，用于方便预览。模型库选择的SG90，即可找到舵机模型，选择并调整模型位置参数即可。

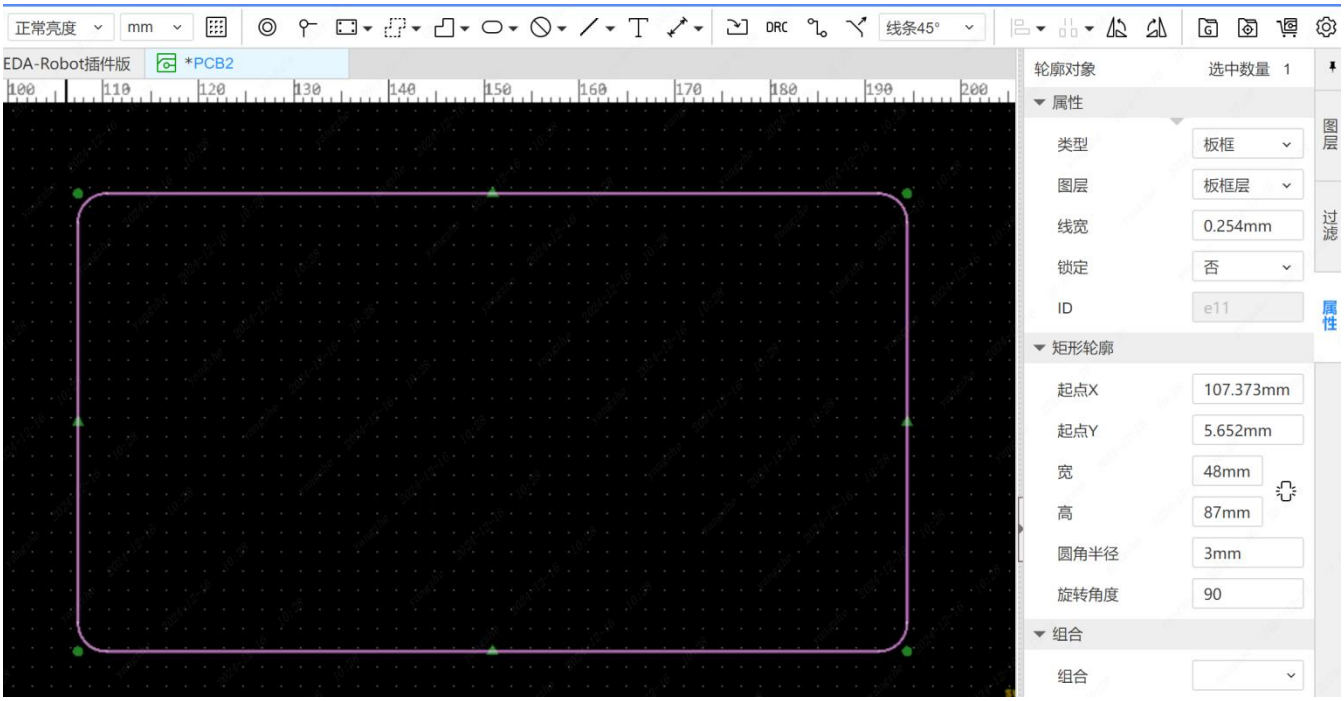






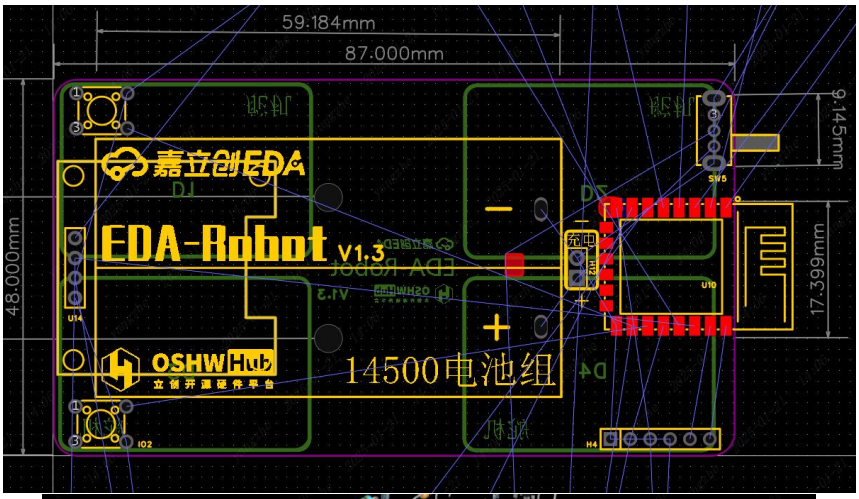
3.2.4边框设置

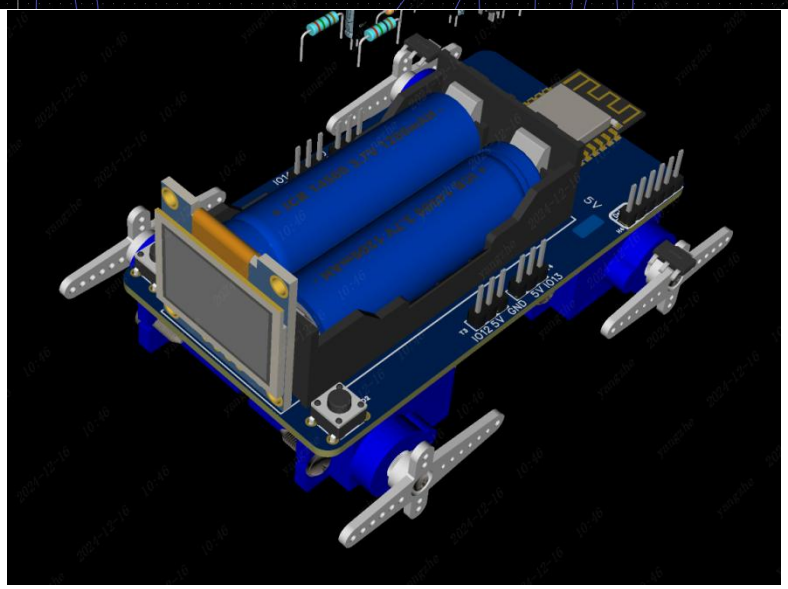
嘉立创可供免费PCB打样的尺寸是10cm*10cm，结合该项目情况我们设为了48mm x 87mm，在放置菜单栏中选择放置-板框，在PCB画布中任意放置一个矩形，点击矩形框，在右侧属性栏中将尺寸改为48*87mm，圆角尺寸设为3mm。



3.2.5PCB布局

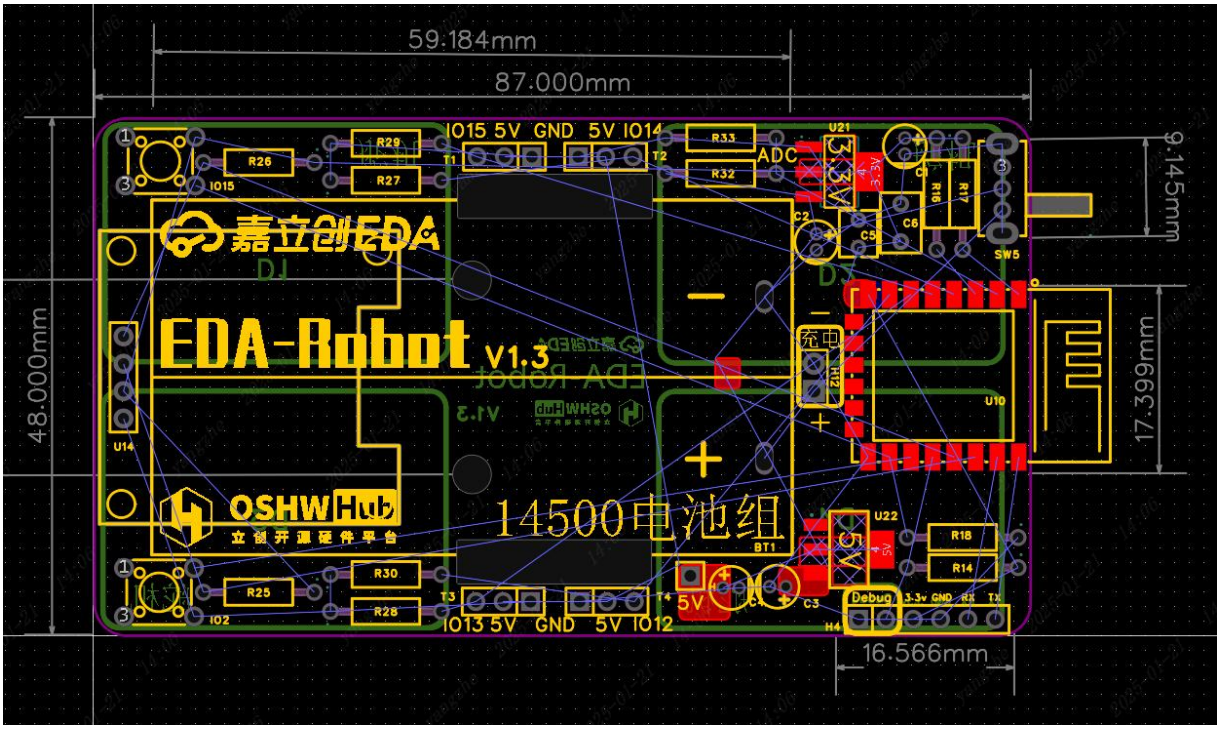
边框放置好后可以将四个螺丝孔分别放置在板子四周，布局时先将大尺寸器件放置在板子内部，进行初步布局，使整个板面电路模块清晰，布局合理，使用方便。布局时使用3D预览功能实时查看布局效果是否合适。

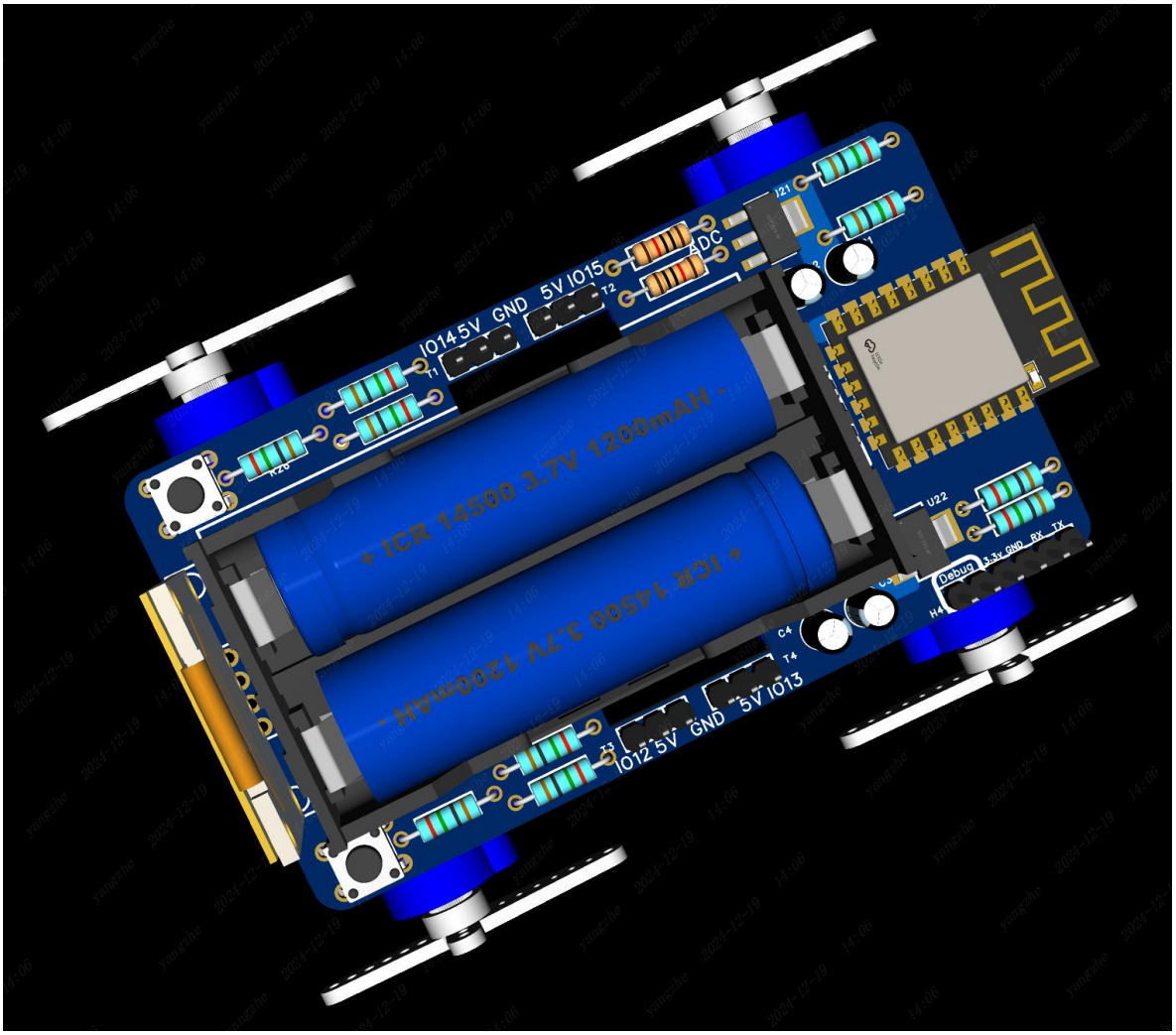




布局时元器件相互连接处有一根淡蓝色的线条，这根线叫做飞线，它起的作用是告诉我们那两个焊盘是相同网络，需要使用导线连接，所以飞线也叫做指引线。但是页面中飞线太多影响布局摆放，在布局走线时可以将GND网络的飞线隐藏，使页面更简洁。隐藏方式是：在左侧“工程设计”列表中选择网络，在搜索栏中搜索GND，在飞线列表中将AGND和GND前的眼睛关闭即可。走线完成后别忘了重新打开哦~

接下来布局时把相关模块电路放到一起，按照飞线的指引摆放，尽可能使飞线水平，走线时减少拐弯，开关接口靠边方便操作，另外将ESP8266的天线部分移至板框外，这样既可以使得WIFI信号较好，又可以充当机械狗的尾巴，最终布局效果如下所示：



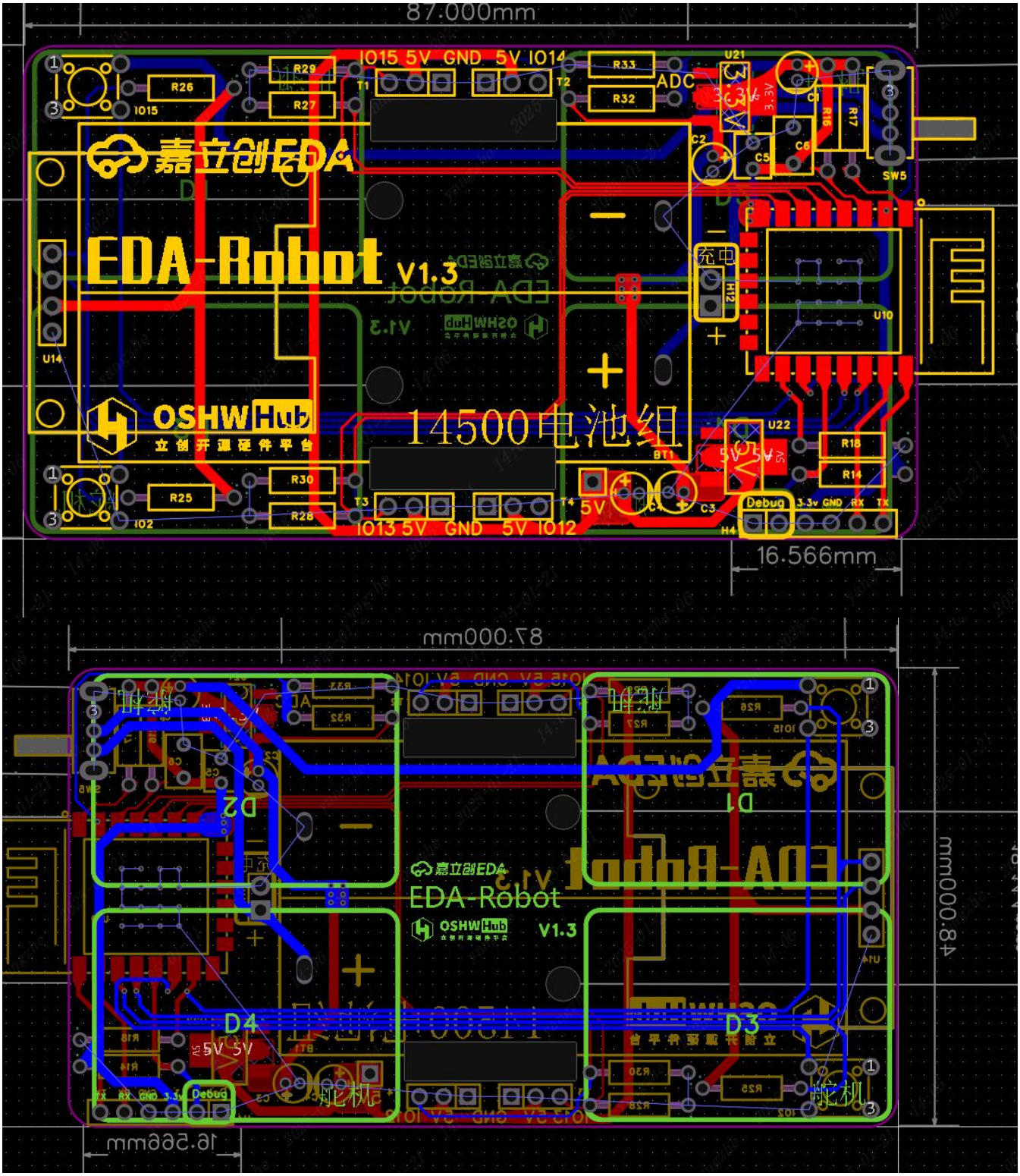


3.2.6PCB走线

好的布局已经成功了一半，接下来只需掌握以下几点走线基本要求即可：

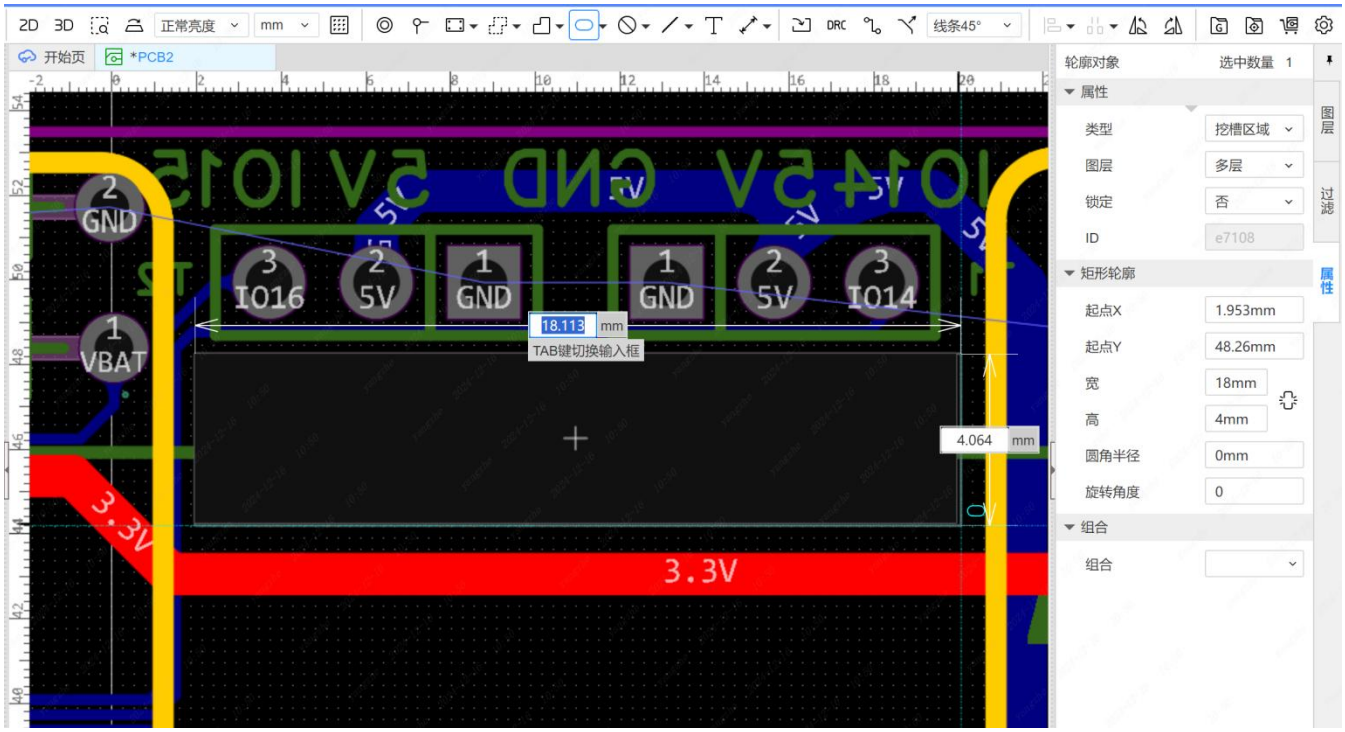
- 走线以直线为主，如需拐弯时拐角以135°钝角或圆角优先，减少直角的使用；

- 走线线宽电源线宽大于信号线，该项目中信号线走线宽度为15mil，电源走线为20mil，GND和AGND网络使用铺铜的方式连接；
- 建议优先使用顶层走线，走不通的地方使用过孔建立顶层和底层的连接后转到底层继续走，底层走不通同样可以放置过孔换到顶层连线；
- 对AGND和GND需要以0欧姆电阻处为分界单独覆铜，这里需结合PCB布局情况来调整覆铜范围；
- 覆铜完成后如果还存在飞线，可通过在存在飞线的位置放置对应网络的过孔或者是调整走线位置使网络能够连接，也可以采用手动接线的方式消除飞线；
- 走线完成后可在“工具”菜单栏选择泪滴添加，加强焊盘与走线的连接，最后再进行覆铜操作，如果对走线有移动调整也应使用快捷键Shift+B进行重建覆铜。

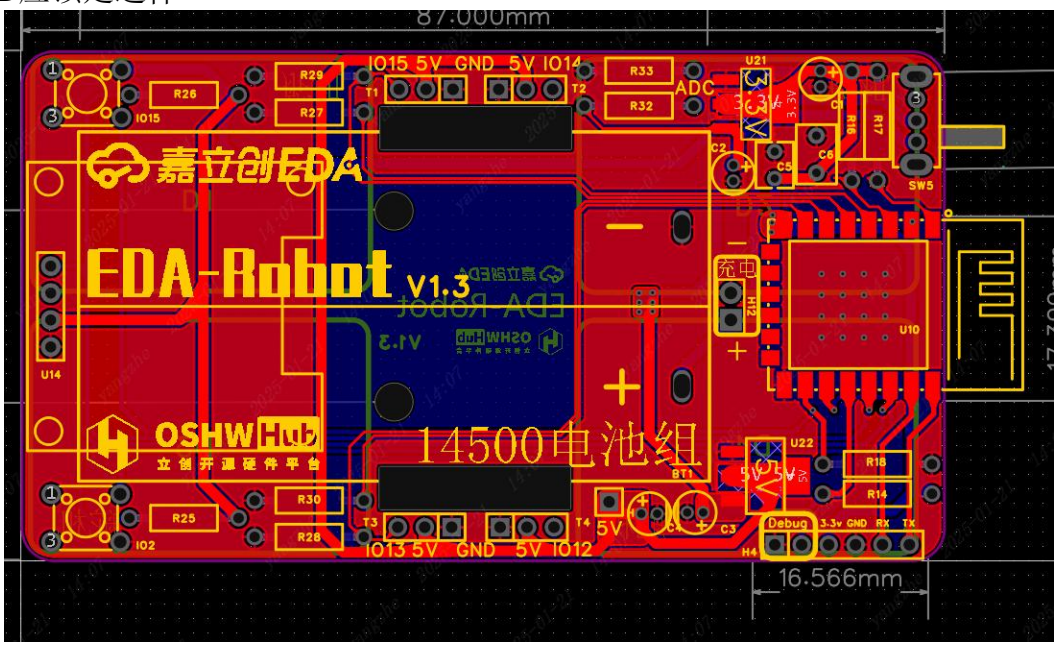


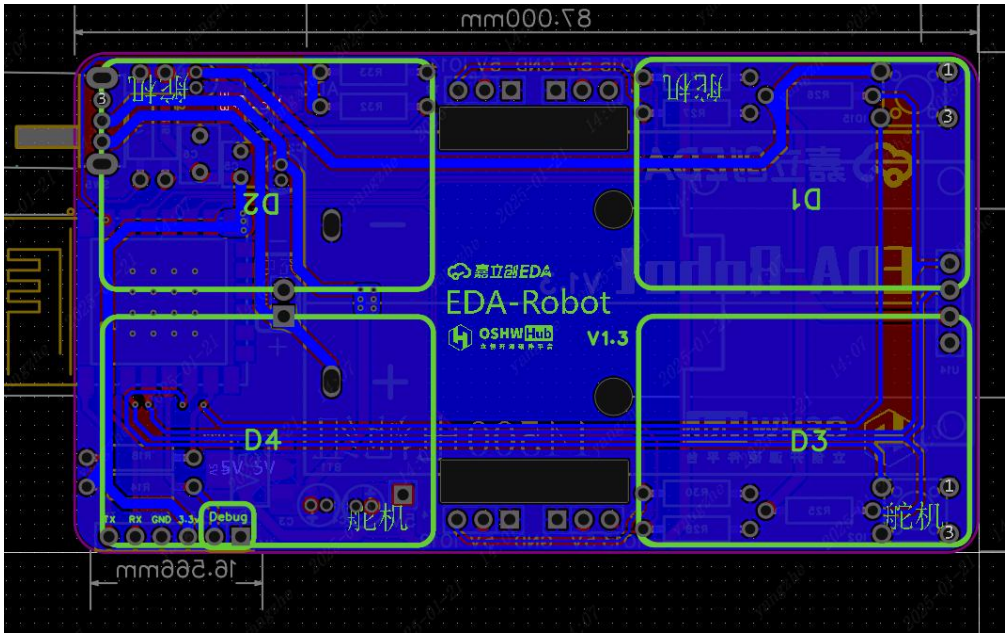
保留GND网络暂时先不管，其他网络布线完成后最后铺上GND铜层,最后再为没连上的GND层打上过孔就好了,对于舵机部分，为了方便走线，我们选择在PCB开槽，允许舵机导线从PCB穿过到达接口。

点击顶部，框选舵机接口下面区域即可。



完整的PCD应该是这样





3.3 免费PCB打样

嘉立创自2019年起给全国电子工程师及爱好者普及PCB免费打样服务至今，伴随着电子工程师的成长，解决学习过程中打样贵、打样慢的问题。在完成PCB设计后我们可以大胆地将文件提交到嘉立创平台进行PCB的打样，每个月可以支持免费打样两次，每次可打一款板子，实际生产5片，重点是还全国包邮！接下来介绍如何在嘉立创平台进行免费的PCB制板服务：

3.3.1 优惠券领取

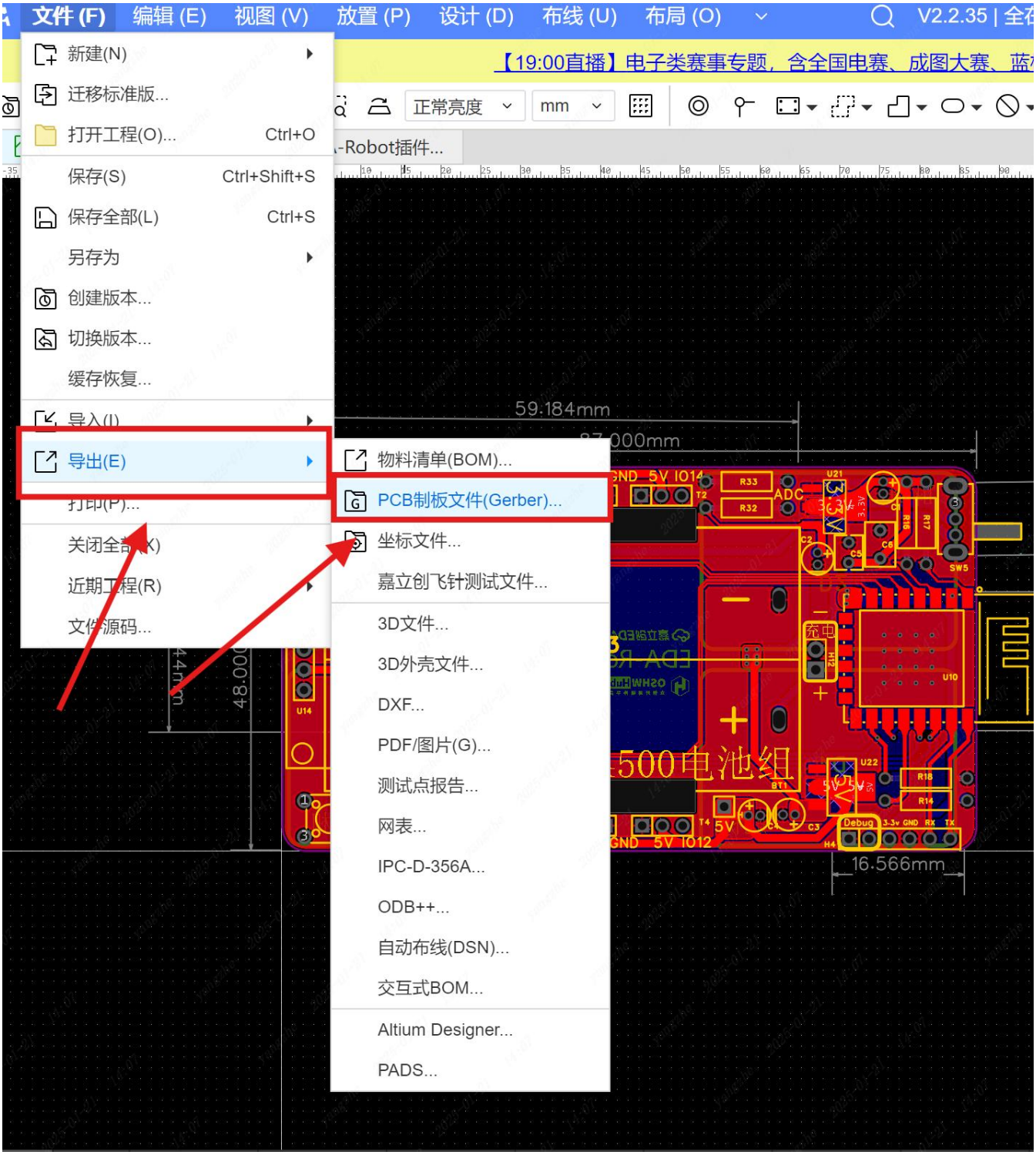
在进行PCB下单前需要先领取一张免费打样券，打开嘉立创下单平台的优惠券领券专区页面，嘉立创下单的账号与嘉立创EDA的账号通用，登录账号后即可领取优惠券：

深圳嘉立创科技集团股份有限公司

进入领券专区后，选择PCB+SMT喷锡免费券进行领取，每个月可领取两张，领取后30天内有效。这里的PCB+SMT的意思是可用于PCB和SMT而不是一定要用SMT才能用券，系统会随机抽取幸运儿，有极小概率抽中的话还可以免费体验工厂帮忙焊接元器件（SMT）的服务，如果能抽中那也是极好的！

3.3.2 生成制造文件

领取优惠券后回到PCB设计页面，点击顶部菜单栏中的“文件”，选择“导出”选项，在里面可以选择导出物料清单（BOM）、PCB制板文件（Gerber）以及坐标文件，这三个文件是实际PCB设计生产中常用到的文件。其中物料清单是用于元器件采购以及SMT贴片时物料选择、坐标文件是用于SMT贴片时器件位置定位，而里面的PCB制板文件（Gerber）文件就是PCB生产用的文件。



点击导出PCB制板文件（gerber），在选项中选择“导出Gerber”，这时系统会提醒先检查DRC再进行导出，这里需要点击“是，检查DRC”，当检查无误后才会导出Gerber文件下载到本地。然后再将Gerber文件上传到嘉立创下单系统即可完成下单。

在导出Gerber文件的窗口还有一个“PCB下单”的选型，直接点击PCB下单的话可以不需要下载文件直接跳转到嘉立创平台上进行PCB的打样操作，这里给PCB下单提供了更多的便利。点击自动下单后 同样需要对DRC进行检查，检查无误后系统会讲下单数据生产，点击“确认”按钮即可跳转到嘉立创下单平台页面进行下单。

3.3.3 下单页面介绍

进入下单平台后，需要填写板子生产工艺参数，具体说明如下：

（1）基本信息

●**板材类别：**选择FR-4，另外FPC板材为柔性PCB、铝基板常用于做灯板、铜基板散热性较好、高频板用于设计制作多阻抗和信号要求较高的板子；

●**板子尺寸：**默认会自动识别出来的，没有识别的话也可手动填写；

●**板子数量：**免费打样数量为5片，如果多打需要自费；

●**板子层数：**嘉立创现在支持1-4层的免费打样，板子设计是两层板，这里选择“2”；

●**产品类型：**选择工业/消费/其他类电子产品，航空和医疗板精度设计要求较高；

●**确认生产稿：**如果是批量生产那必须要确认生产稿，避免生产文件有误影响板子功能，免费打样则选择不需要生产稿即可。

（2）PCB工艺

PCB工艺选项里面内容较多，仅需关注下图中框选出来的几个选项：

●**拼板款数：**在进行批量打样时常将多个PCB拼在一个板子上生产，这样成本更低。由于目前仅做免费打样，拼板数量应为1，出货方式为单片；

●**板子厚度：**默认板子厚度为1.6mm，无特殊要求建议是默认1.6mm ，选择其他厚度和颜色匹配时容易选到冷门工艺会额外添加工艺费；

●**阻焊颜色：**即板子的颜色，嘉立创支持七种不同的阻焊颜色，其中绿色的生产周期最快，最快48小时内发货，其他颜色最快是72小时发货，可结合自身喜好和板子的着急程度选择合适的阻焊颜色。

（3）个性化服务

个性化服务没有特殊需求选默认即可。

（4）交期

交期与所选颜色有关，嘉立创最快支持12小时加急，但需额外支付加急费。若无特殊需求，选用默认交期即可。

（5）SMT贴片与激光钢网

如需选择工厂代焊接元器件则在此选择需要SMT贴片，工厂生产PCB后会将元器件一同焊接好寄出，SMT属于收费服务，若不需要则选择不需要即可。钢网是用于SMT贴片刷锡膏用的，如果自己有贴片机，可在生产PCB时选择开钢网回来自己进行贴片焊接。

（6）开票与支付

嘉立创是支持开票的，下单前需填写开票税号，免费打样无需开票，开票信息填写个人即可。在确认订单方式推荐选择“系统自动扣款并确认”选项，避免个人疏忽忘记确认订单影响生产。

（7）发货与快递

在发货页面建议选择电子收据，发货方式可选不同交期是否一起发后，填写收货地址，下单联系人和技术联系人都可以填自己的联系方式，快递根据地区不同会由不同的快递显示，选择里面一个免费的快递即可。

（8）使用优惠券下单

填写完下单数据后，在右侧结算页面选择使用优惠券下单即可获得减免，这里注意下优惠前面额应该是20元，如果超过20那可能是选了某个特殊工艺，应修改回默认工艺后再使用优惠券下单。

提交订单后还需确认订单，系统可能会自动审核通过，也有可能会有人工审核参与，这时需等待审核通过后进行确认订单才可以进入生产环节，如果前面选了“系统自动扣款并确认选项”在有余额的前提下系统就自动确认并进入生产啦~

生产中的板子可在嘉立创下单平台的PCB订单中查看进度以及快递情况，稍等几天就能拿到5块新鲜的电路板啦~

3.4 PCB焊接练习

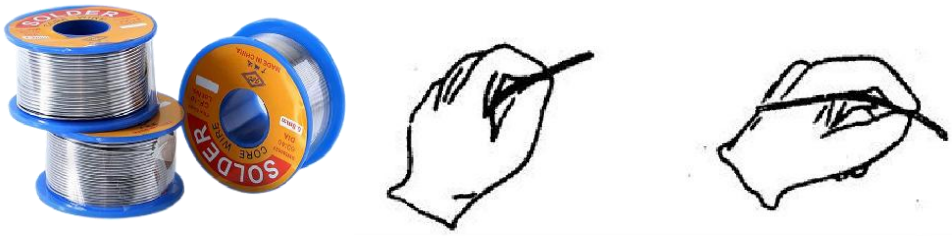
准备好元器件与PCB板后下一步进入焊接调试焊接，需要掌握常用焊接工具的使用、插件元器件的焊接与拆卸方法。

3.4.1 焊接工具

需要用到的焊接工具有：电烙铁、焊锡丝、高温海绵、斜口钳、吸锡器、松香、洗板水等，其中电烙铁、焊锡丝、高温海绵、斜口钳这四项为必备工具，其余几个有最好，没有也可以。

电烙铁选择常规烙铁就行，有条件的可以选择焊台，升温快且稳定。烙铁加热时请勿触摸金属位置，避免烫伤，头发绑好避免烫到，桌面保持整洁。

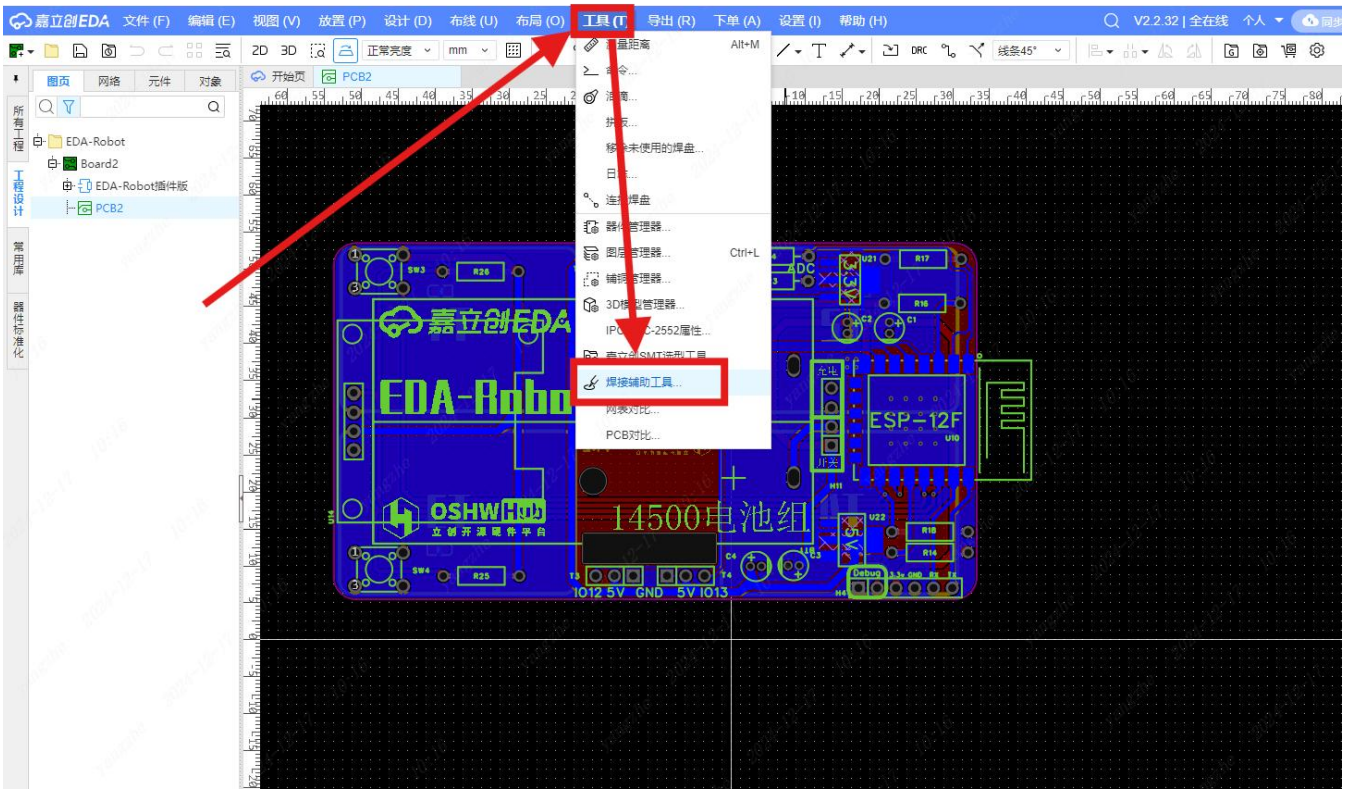
焊锡建议是选用无铅焊锡，焊锡时较好比较好上锡。一般左手拿焊锡，右手拿烙铁，如果惯用左手的话反之。



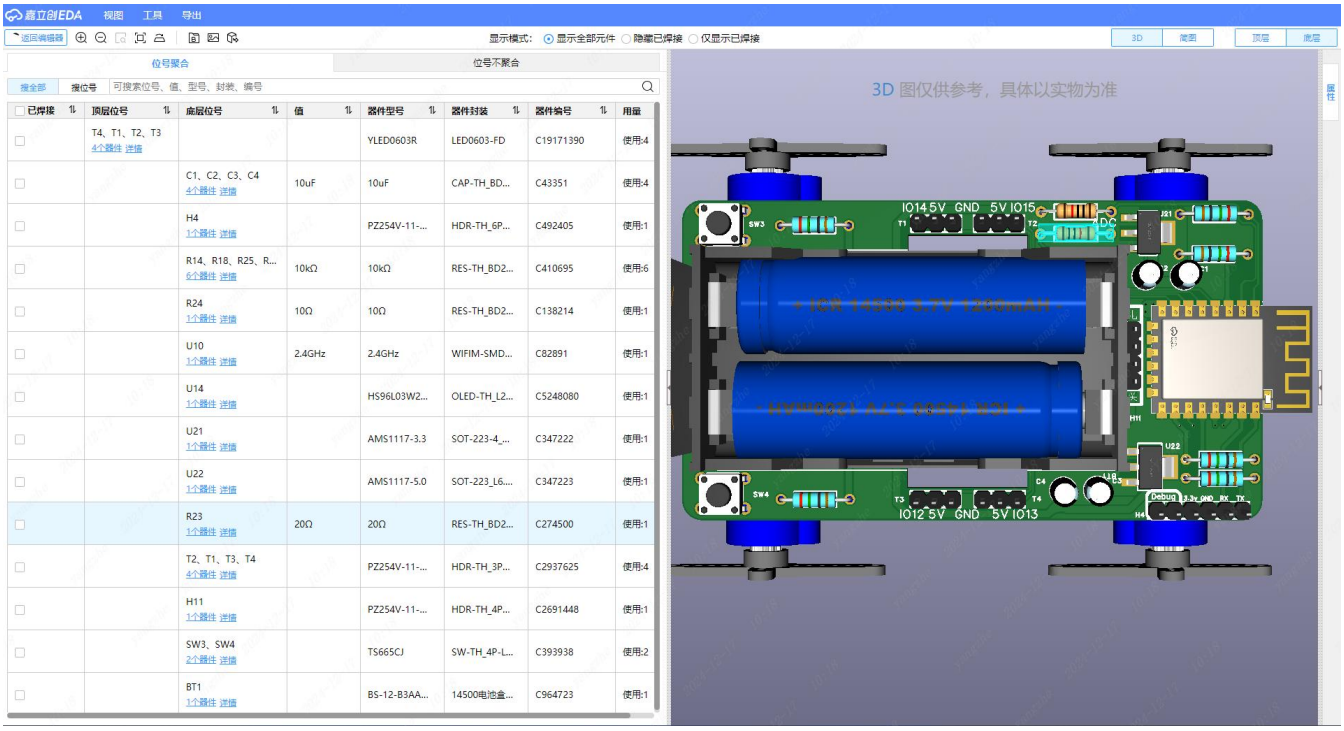
高温海绵在使用前先用水浸湿，然后拧干后使用，海绵上不能沾太多水分，避免烙铁头高温沾水后急速降温损坏烙铁头；斜口钳使用时用手捏住多余引脚，然后再剪，免得引脚导出弹飞。

3.4.2 辅助焊接工具

在嘉立创EDA专业版中提供了一个焊接辅助工具，焊接时可在线打开该工具对照元器件的参数及位置，实时进行查看焊接情况。打开方式是该项目的PCB设计页面，点击“工具”菜单栏，选择“焊接辅助工具”；也可以直接点击下载以下离线文件，并在浏览器中打开。



进入焊接辅助工具后，在显示模式中选择仅显示已焊接选项，左侧选择焊接时勾选对应器件，右侧的3D预览效果图显示当前器件焊接位置及情况，避免焊接错误。焊接的原则是从矮到高以此焊接。焊接时建议根据下表中的焊接顺序进行焊接。



已焊接	顶层位号	底层位号	值	器件型号	器件封装	器件编号	用量
	T4、T1、T2、T3 4个器件 详情			YLED0603R	LED0603-FD	C19171390	使用:4
	C1、C2、C3、C4 4个器件 详情		10uF	10uF	CAP-TH_BD...	C43351	使用:4
	H4 1个器件 详情			PZ254V-11-...	HDR-TH_6P...	C492405	使用:1
	R14、R18、R25、R... 6个器件 详情		10kΩ	10kΩ	RES-TH_BD2...	C410695	使用:6
	R24 1个器件 详情		10Ω	10Ω	RES-TH_BD2...	C138214	使用:1
	U10 1个器件 详情		2.4GHz	2.4GHz	WIFIM-SMD...	C82891	使用:1
	U14 1个器件 详情			HS96L03W2...	OLED-TH_L2...	C5248080	使用:1
	U21 1个器件 详情			AMS1117-3.3	SOT-223-4_...	C347222	使用:1
	U22 1个器件 详情			AMS1117-5.0	SOT-223_L6....	C347223	使用:1
	R23 1个器件 详情		20Ω	20Ω	RES-TH_BD2...	C274500	使用:1
	T2、T1、T3、T4 4个器件 详情			PZ254V-11-...	HDR-TH_3P...	C2937625	使用:4
	H11 1个器件 详情			PZ254V-11-...	HDR-TH_4P...	C2691448	使用:1
	SW3、SW4 2个器件 详情			TS665CJ	SW-TH_4P-L...	C393938	使用:2
	BT1 1个器件 详情			BS-12-B3AA...	14500电池盒...	C964723	使用:1

4 软件开发

4.1 开发环境介绍

4.1.1 VSCode

（1）介绍

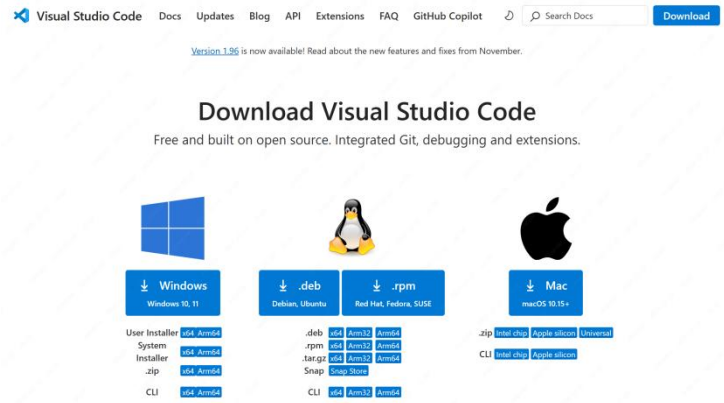
官网：https://code.visualstudio.com/

VSCode是由微软开发的免费开源代码编辑器，是大多数开发者的首选编辑器，拥有卓越的性能和极其丰富的拓展插件，使用VSCode不仅可以编写嵌入式代码，还可编写软件，前后端及Linux服务端程序等等，功能强大。

（2）下载

访问https://code.visualstudio.com/Download

选择你使用的平台，下载对应程序即可。



4.1.2 PlatformIO

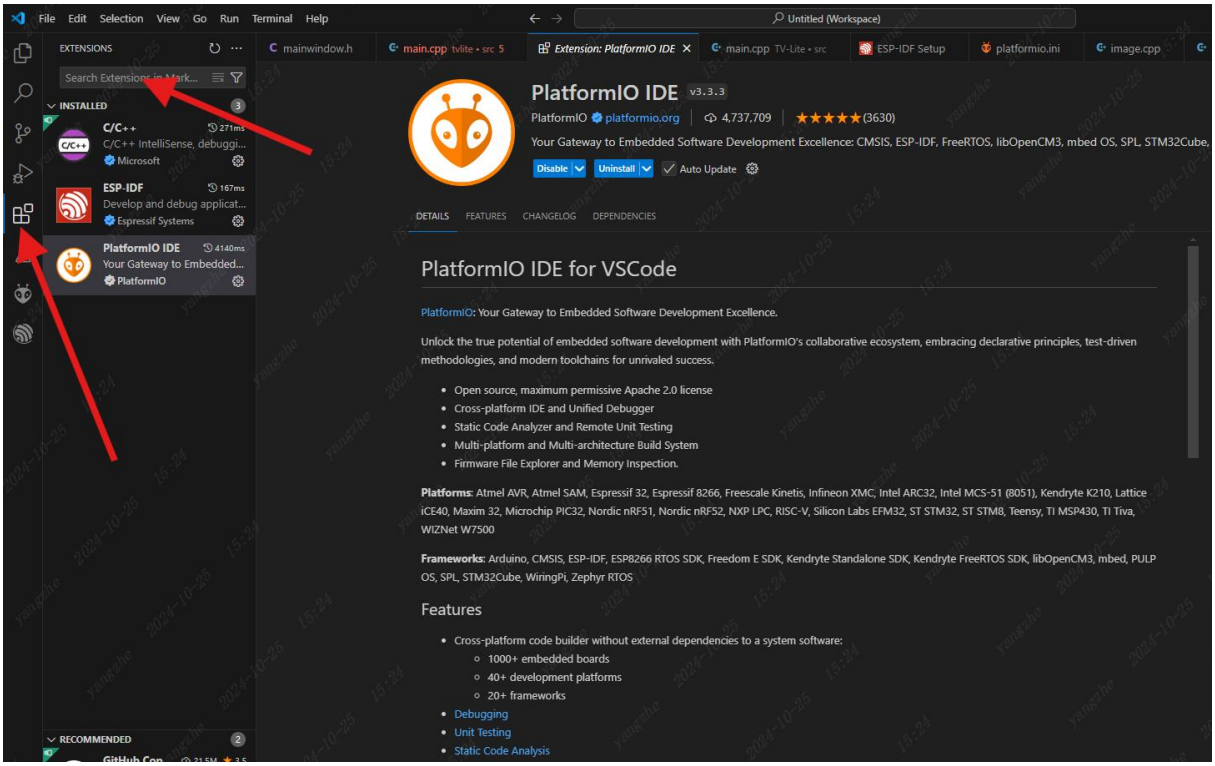
（1）介绍

PlatformIO是一个强大的嵌入式开发框架，你可以通过自由选择MCU平台，以及MCU固件包框架，例如ESP系列可以自由选择使用ESP-IDF还是Arduino再或者是第三方开源框架,非常方便。而且PlatformIO将MCU配置集成在platformIO.ini文件内，你可以通过该文件为MCU分区，或者超频，拥有较强的自定义功能。

这里你可以自由选择PlatformIO、ESPIDF、Arduino来编写，都是通用的

（3）安装

在VsCode点击左侧插件图标，搜索安装PlatfromIO框架插件即可



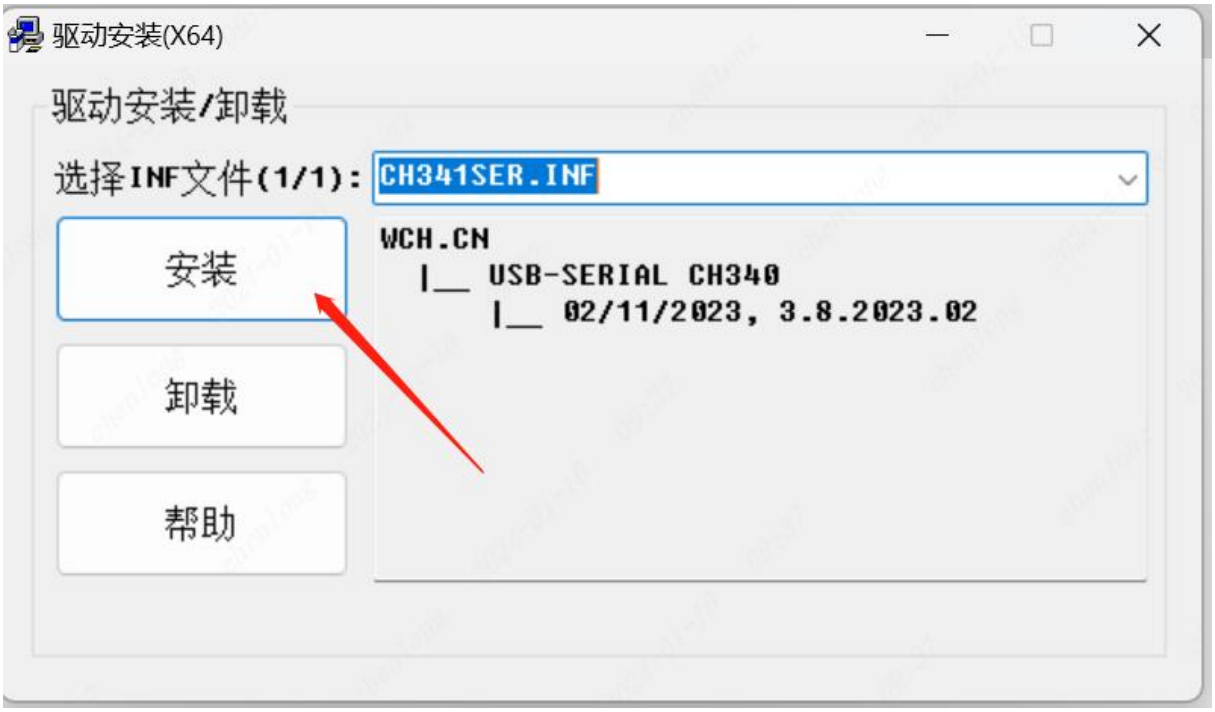
4.1.3 程序下载说明

ESP8266模组主要使用TTL串口下载方式，准备一个USB转TTL或者DAP-Link即可

（1）安装CH340驱动

鼠标右键以管理员身份进行安装；

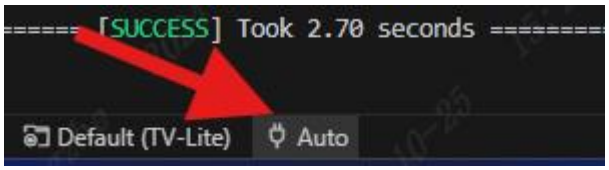
如果出现安装失败，那可能是你电脑已经有了该驱动，你可以尝试先点击卸载，再次安装即可。



（2）硬件配置

在硬件设计中，我们特意为主板设计了下载模式跳线接口，使用跳帽或镊子短接后再上电，即可进入下载模式进行程序下载。然后再将主板上的TX连接到烧录器的RX，主板上的RX连接到烧录器的TX（RX、TX交叉连接）

如果成功进入下载模式，此时在VsCode左下角应该有端口选择按钮，点击选择能看到端口

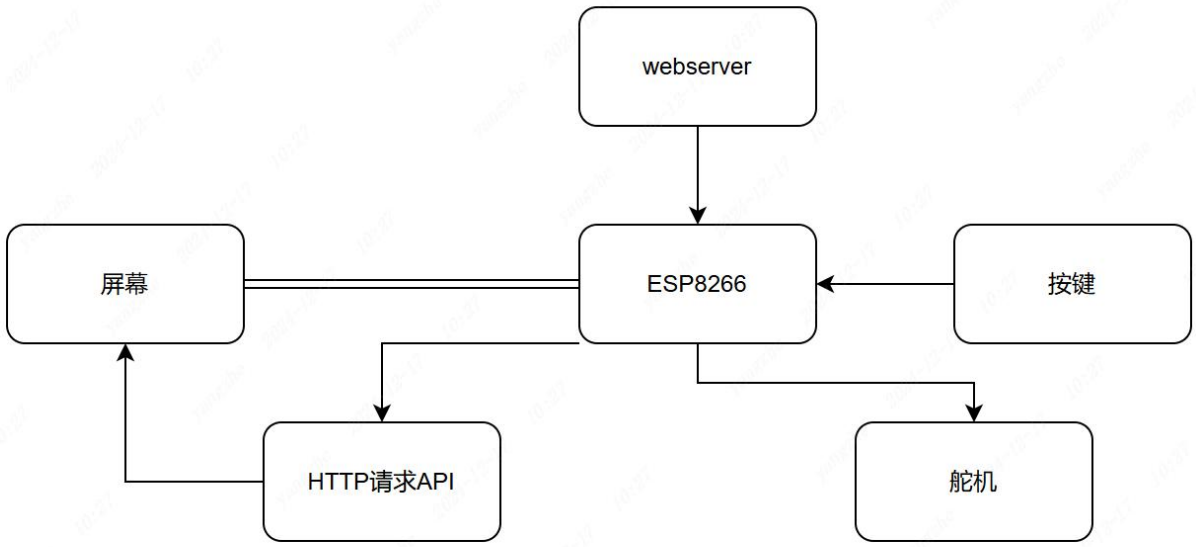


4.2 开发流程

在开发之前，我们先要对整个程序的流程进行设计，确保知道我们的项目需要做什么

4.2.2 整体程序分析

对于本项目来说，我们应该实现表情显示与切换、时钟显示、天气信息显示和舵机控制并通过网页完成交互，大致的程序流程如下

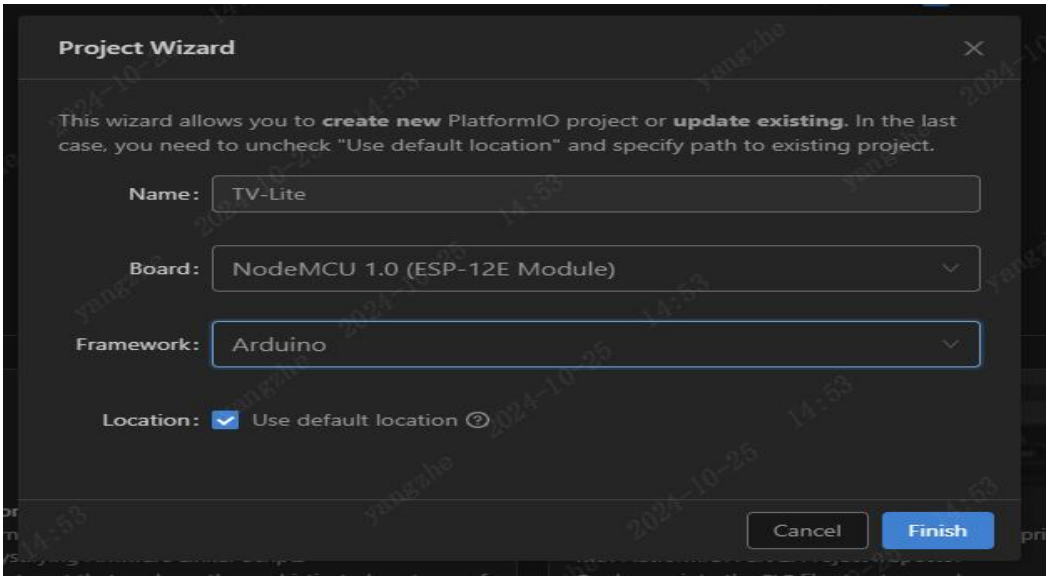


当然，这仅局限于本教程，你也可以为你的小电视开发更多好玩有意思的功能

4.3 创建项目

4.3.1 构建项目框架

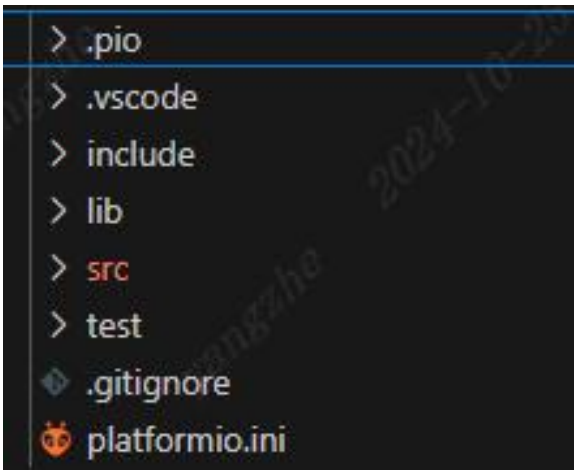
首先，我们新建一个PlatformIO项目，修改好项目名称，版型选择NodeMCU中的NodeMCU 1.0,使用Arduino固件然后点击创建。（版型的选择是随意的，主要是选对正确的芯片类型，因为ESP8266有多种款式，我们使用的类型是ESP8266,80MHZ,4M）



4.3.2 项目框架结构介绍

（1）项目文件夹

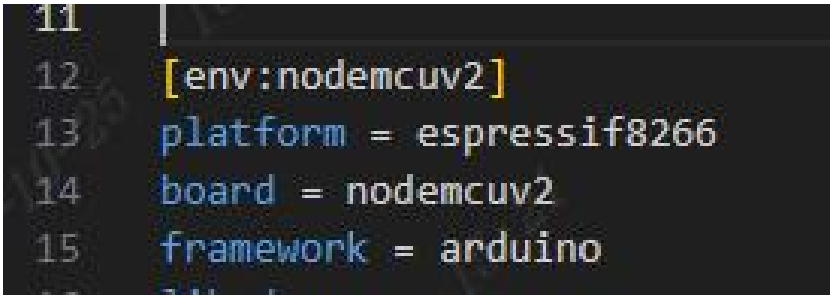
在项目文件夹下共有5个文件夹，这些文件夹我们在后面介绍。我们先来讲解该目录下最为重要的platformio.ini文件，这个文件是用于存放我们需要调用的第三方库和配置板级配置的文件，在这个文件中可以配置分区、配置处理器频率、固件类型等等



platformio.ini

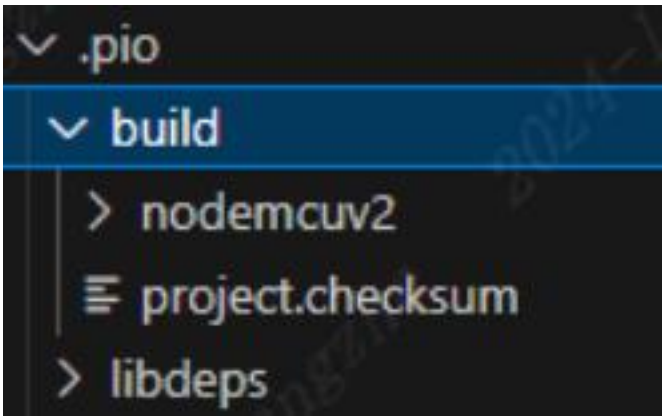
默认的文件内容如下，platform指的是我们使用的平台，board指的是我们使用的版型，framework指的是我们使用的固件类型，完整的文件配置信息可以查看官方的说明文档

“platformio.ini” (Project Configuration File) — PlatformIO latest documentation



（2）.pio文件夹

.pio文件夹是用于存放我们依赖的第三方库文件以及构建后的项目固件及构建文件的地方，默认文件如下



build

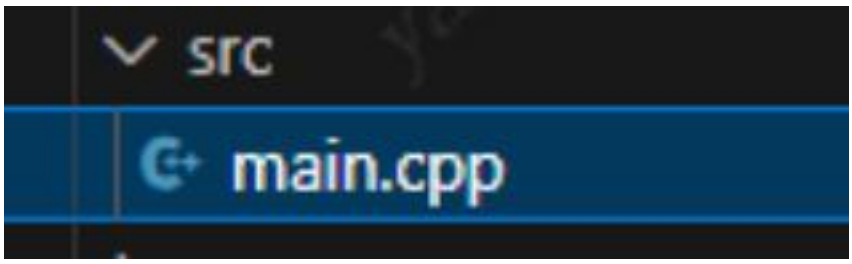
用于存放构建文件，生成的固件也会在这个文件夹中

libdeps

当引入第三方库时才会生成该目录，在该目录下可以修改第三方库的源代码

（3）src文件夹

其他文件夹就不再讲解了，主要是用于分类和自行导入第三方库用的。
src文件夹是生成初始程序的文件夹，在新建的项目中会在这里生成main.cpp,这个就是我们的主程序了



4.3.3 程序结构介绍

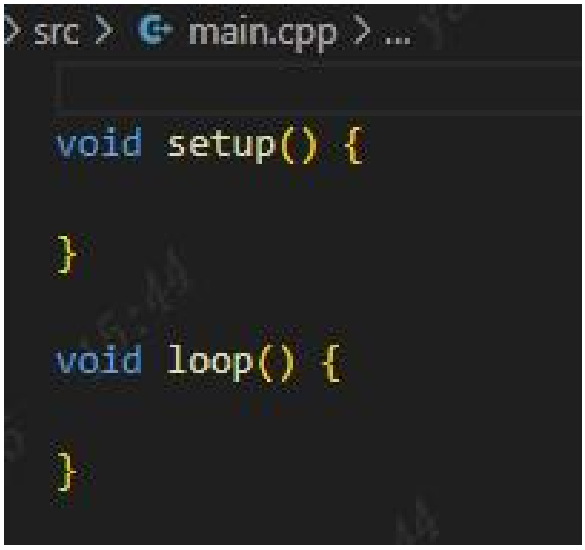
这就是一个基本的Arduino程序结构，在程序最上面导入头文件和全局变量，但是与Arduino不同的是这是一个完全遵循C++标准的后缀为.cpp的程序源文件。

setup()

在这个方法中我们主要存放需要初始化的代码，在该方法中的所有程序仅在开机时执行一次

loop()

在这个方法在我们主要存放主代码，在该方法中的所有程序会进入死循环，无限次重复执行



4.4 驱动库

4.4.1 OLED屏幕

（1）库介绍

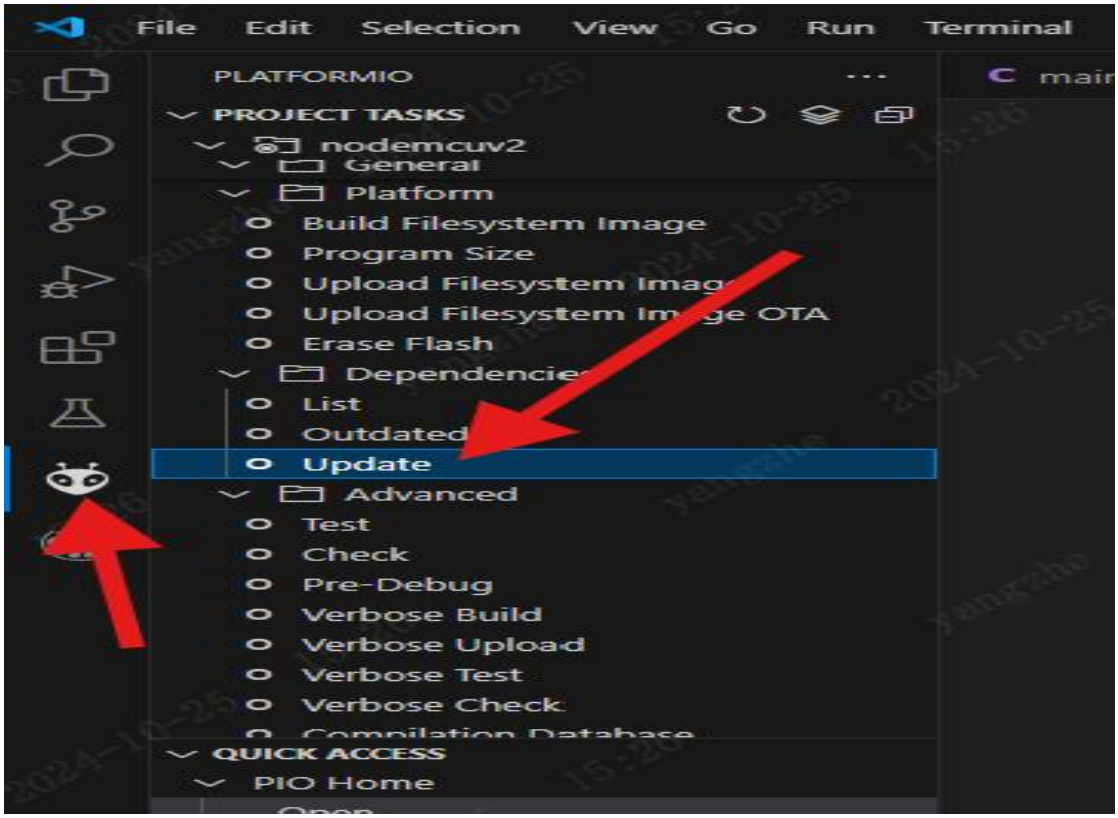
对于OLED屏幕，通常可以使用U8g2驱动库驱动，本项目使用的ssd1315驱动和ssd1306驱动是大致一样，通用的，所以这里选择U8g2的ssd1306驱动进行驱动。

（2）库导入

在platformio.ini中最下面添加

```
lib_deps =
    olikraus/U8g2 @ ^2.36.2
```

然后我们点击左侧PlatformIO的图标，选择Dependencies，点击Update更新依赖即可



现在platformio会自动拉取第三方库的git仓库到项目

（3）屏幕代码初始化

在main.cpp中最顶部添加

```
#include <U8g2lib.h>
#include <Wire.h>
```

在setup()中添加

```
u8g2.begin();
u8g2.setDisplayRotation(U8G2_R2);
```

4.4.2 NTP

NTPClient 是一个用于在 Arduino 和其他嵌入式设备上获取网络时间的库。它通过网络时间协议（NTP）从互联网的 NTP 服务器获取当前时间，适用于需要精确时间戳的项目，比如时钟、日志记录或定时器等。

主要特点

1. **简单易用：**
   提供直观的 API，使得获取和管理时间变得容易。
2. **支持时区：**
   可以设置时区偏移量，以获取本地时间。
3. **定时更新：**
   支持定期更新时间，确保时钟保持准确。

（1）NTPClient库导入

```
lib_deps =
arduino-libraries/NTPClient @ ^3.2.1
```

然后我们重新加载

（2）NTP初始化

NTPClient 库是一个简单的 NTP 客户端库，可以从 NTP 服务器同步时间。在这个项目中，ESP8266 使用 NTPClient 获取当前时间，用于显示或其他时间相关的功能。

在main.cpp中最顶部添加

```
#include <NTPClient.h>
WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "ntp1.aliyun.com", 8 * 3600, 60000);
```

4.4.3 Json格式提取

ArduinoJson 是一个用于在 Arduino 和其他嵌入式平台上处理 JSON 数据的轻量级库。它允许开发者轻松地解析、生成和操作 JSON 数据，非常适合用于处理来自网络请求的响应、配置文件或其他结构化数据。

主要特点

1. **轻量级：**
   设计针对内存受限的环境，内存占用小。
2. **易于使用：**
   提供直观的 API，简单的语法使得 JSON 的创建和解析变得方便。
3. **支持动态和静态内存分配：**
   可以根据需求选择动态分配内存或使用静态数组，适应不同的使用场景。
4. **文档生成：**
   支持生成和解析 JSON 文档，便于处理复杂的数据结构。

Json格式的读取与写入我们使用ArduinoJson库完成

（1）ArduinoJson库导入

```
lib_deps =
bblanchon/ArduinoJson @ ^7.2.0
```

然后我们重新加载

（2）ArduinoJson初始化

在main.cpp中最顶部添加

```
#include <ArduinoJson.h>
```

4.4.4 ESPWeb本地服务器

ESPAsyncWebServer 是一个为 ESP8266 和 ESP32 平台设计的异步 Web 服务器库。与传统的 Web 服务器库相比，它允许在处理 HTTP 请求时非阻塞地执行其他操作，适合于需要同时处理多个连接的应用。

主要特点

1. **异步处理：**
   允许同时处理多个客户端连接，而不会阻塞主程序流。
2. **简单的 API：**
   提供易于使用的 API，可以快速设置路由和处理 HTTP 请求。
3. **支持多种 HTTP 方法：**
   支持 GET、POST、PUT、DELETE 等多种 HTTP 方法。
4. **处理静态文件：**
   能够直接提供存储在 SPIFFS 或 LittleFS 文件系统中的静态文件（如HTML、CSS、JavaScript 文件等）。
5. **WebSocket 支持：**
   支持 WebSocket，允许与客户端进行实时双向通信。

为了实现本地网页配网，我们需要一个Web本地服务器

（1）ESPAsyncWebServer库导入

```
lib_deps =
esphome/ESPAsyncWebServer-esphome @ ^3.2.2
```

然后我们重新加载

（2）ESPAsyncWebServer初始化

在main.cpp中最顶部添加

```
#include <ESP8266HTTPClient.h>
#include <ESPAsyncWebServer.h>
AsyncWebServer server(80);
```

4.4.5 FS文件系统

FS（文件系统）在微控制器和嵌入式开发中用于管理和存储文件。特别是在Arduino、ESP8266和ESP32等平台上，文件系统可以用来读取和写入数据，便于保存配置、网页文件和其他信息。

常见的文件系统

1. SPIFFS（SPI Flash File System）：
   设计用于闪存设备，适合于较小的存储需求。
   支持文件读取、写入、删除等基本操作。
   在ESP8266和ESP32上常用。
2. LittleFS：
   提供更好的耐用性和性能，相比SPIFFS在特定场景下更为高效。
   支持原子性写入和可变大小的文件，适合更复杂的应用。

（1）FS库导入

在main.cpp中最顶部添加

```
#include <FS.h>
```

4.4.6 WIFI网络

ESP8266WiFi 是一个专为 ESP8266 微控制器设计的库，用于处理 WiFi 连接。它提供了简单易用的 API，使得开发者可以方便地连接到 WiFi 网络、管理连接、发送和接收数据等。

（1）ESP8266WiFi库导入

在main.cpp中最顶部添加

```
#include <ESP8266WiFi.h>
```

4.4.7 舵机控制

Servo是一个Arduino官方设计的舵机控制库，用于处理PWM发送。它提供了简单易用的 API，使得开发者可以方便地连接控制电机，当然你也可以手动发送PWM脉冲控制电机。

（1）Servo库导入

在main.cpp中最顶部添加

```
#include <Servo.h>
```

4.5 图像库

4.5.1 分辨率

图像中我们尽可能统一像素

图标像素：64x64

表情像素：128x64

4.5.2 取模

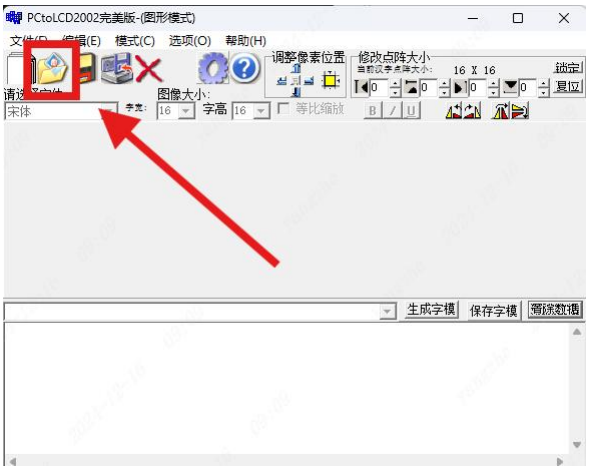
取模的软件有很多，本文档中我们主要使用PCtoLCD2002来完成图像和字库

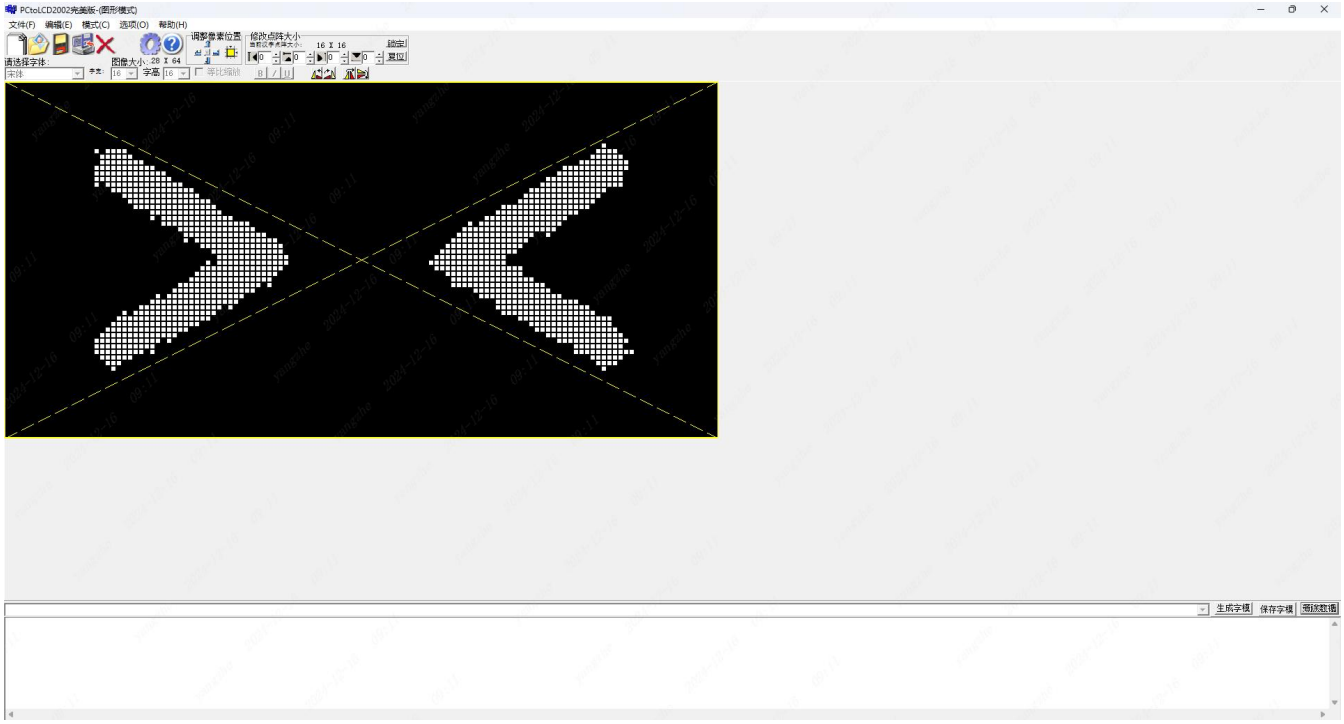
（1）项目创建

在开始之前，先确保图片的像素已经调整到正确的大小，并且由于是单色屏，还需要将图片进行位图处理，得到BMP格式的位图。

（2）图像导入

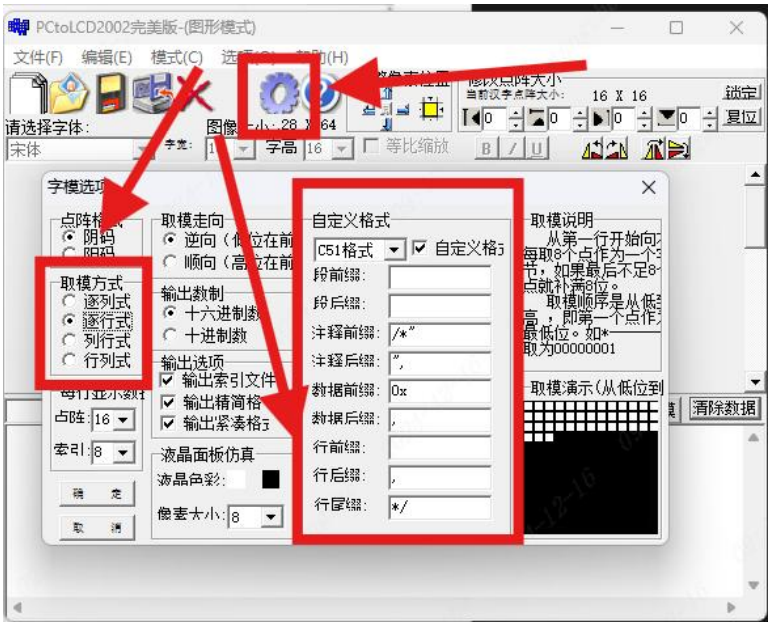
左上角文件夹图标





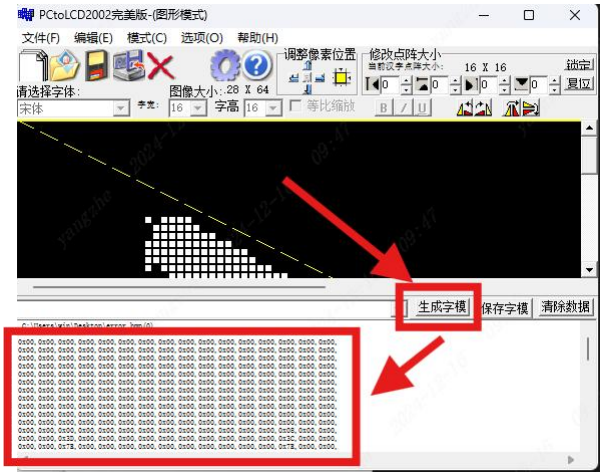
（3）取模设置

点击设置图标确保



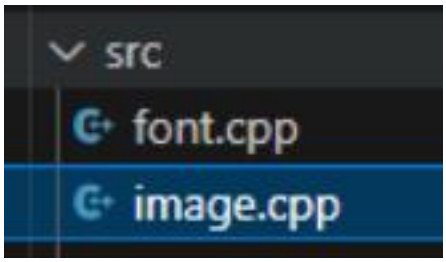
（4）生成

点击生成字模即可获取到对应图片的数组



4.5.2 存储

现在我们在src中新建一个image.cpp




```
const  uint8_t 在此定义名称[] PROGMEM= {
//在此处粘贴取模后的数值
};
```

4.5.3 导入

```
#include "image.cpp"
```

4.6 程序开发

现在，你已经完成了所有准备工作，开始编写程序吧。

为方便理解，本项目教程的程序并没有进行性能和安全优化，如果有需求可以自行修改。

本项目使用C++语言编写，由VSCode+PlatfromIO框架环境完成。

本项目一共创建有15个函数接口

```
void handleWiFiConfig()//创建Web路由
void loadWiFiConfig()//读取保存在FS文件系统的WIFI信息
void fetchWeather()//获取天气信息
void front()//前进
void back()//后退
void left()//左转
void right()//右转
void sitdown()//坐下
void lie()//躺下
void toplefthand()//举起左手
void toprighthand()//举起右手
void left90()//舵机集体左转90度
void right90()//舵机集体右转990度
```

```
void setup()//初始化
void loop()//运行
```

4.6.1 库的导入

这个项目旨在使用 ESP8266 芯片，通过 U8g2库在屏幕上显示图像以及其他信息，同时可以联网并使用 NTP 客户端进行时间同步，以及控制舵机，HTTPClient 用于访问网络数据，ArduinoJson 用于解析 JSON 数据，ESPAsyncWebServer 用于创建 Web 服务器。

```
#include <Arduino.h>//Arduino库文件
#include <Servo.h>//舵机库文件
#include <U8g2lib.h>//U8g2显示驱动文件
#include <Wire.h>//IIC库文件
#include <FS.h>//spiffs文件系统库
#include <ESP8266WiFi.h>//WIFI库
#include <ESPAsyncWebServer.h>//WebServer异步库
#include <ESP8266HTTPClient.h>//HTTP请求库
#include <NTPClient.h>//NTP同步库
#include <ArduinoJson.h>//Json文件处理库
#include <WiFiUdp.h>//UDP库
#include "image.cpp"//表情库
```

4.6.2 全局变量与定义

（1）舵机定义

```
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
```

在本项目程序中，我们需要使用这些全局变量，用于舵机驱动

（2）屏幕定义

```
U8G2_SSD1306_128X64_NONAME_1_HW_I2C u8g2(U8G2_R0, /* reset=*/U8X8_PIN_NONE,
/* clock=*/5, /* data=*/4);
```

这里定义U8g2使用SSD1306，硬件IIC驱动屏幕，使用硬件IIC可以确保屏幕响应高速率，高反应。

（3）NTP 客户端和时间同步

```
WiFiUDP ntpUDP; // 创建 WiFi UDP 对象
NTPClient timeClient(ntpUDP, "ntp1.aliyun.com", 8 * 3600, 60000); // 创建 NTP 客户端
```

ntpUDP 用于网络时间协议（NTP）同步时创建 UDP 通信。timeClient 是 NTP 客户端对象，连接阿里云的 NTP 服务器（"ntp1.aliyun.com"）同步时间，时区设置为 UTC+8。

（4）创建异步 Web 服务器

```
AsyncWebServer server(80); // 创建异步 Web 服务器，监听端口 80
```

server 是一个异步 Web 服务器对象，监听端口号为 80，允许客户端访问 ESP8266 提供的 Web 页面来配置或查看信息。

（5）定义热点名称

```
const char *ssid = "EDA-Robot";
const char *password = ""; // 无密码
```

ssid 是 ESP8266 创建的 WiFi 热点的名称，设备可以通过连接该热点进行配置。

（6）天气API URL

```
const char* weatherAPI = "http://api.seniverse.com/v3/weather/daily.json?key="; // 天气 API URL 前缀
```

weatherAPI 是天气 API 的 URL 前缀，可以根据位置等参数进行扩展。

（7）天气信息变量

```
static bool initweather = false; //天气初始化定义
String temperature = ""; // 温度
String humidity = ""; // 湿度
String weather = ""; // 天气状况
String useruid = ""; // 用户 ID
String cityname = ""; // 城市名称
String weatherapi = ""; // 天气 API 密钥
```

这些字符串变量用于存储天气和 B 站信息，例如温度、湿度、天气、粉丝数量等，便于在屏幕或 Web 页面上显示。

（8）WiFi 信息文件路径

```
const char* ssidFile = "/ssid.json"; // 保存 WiFi 信息的文件路径
```

ssidFile 用于存储 WiFi 配置信息（如 SSID 和密码），保存到 ESP8266 文件系统中，以便下次启动时读取。

（9）舵机控制定义

***180舵机删除了补偿定义，因为180电机自带限位器，无需校准补偿**

```
int engine1 = 14;                    // 舵机引脚
int engine1offsetleftpwm = -5;       // 舵机左转补偿
int engine1offsetrightpwm = -20;     // 舵机左转补偿-40
int engine2 = 16;                    // 舵机引脚
int engine2offsetleftpwm = -63;      // 舵机左转补偿
int engine2offsetrightpwm = -55;     // 舵机左转补偿-60
int engine3 = 12;                    // 舵机引脚
int engine3offsetleftpwm = 16;       // 舵机左转补偿
int engine3offsetrightpwm = -11;     // 舵机左转补偿
int engine4 = 13;                    // 舵机引脚
int engine4offsetleftpwm = -56;      // 舵机左转补偿
int engine4offsetrightpwm = -60;     // 舵机左转补偿-71
int runtime = 100;                   // 运动延时**预留变量，用于控制动作连贯性**
```

（10） 按键定义

```
#define BUTTON_PIN 2 // GPIO2 引脚 (D4)
#define BUTTON_PIN2 15
volatile bool buttonPressed = false;        // 按键标志
volatile bool buttonPressed2 = false;       // 按键标志
unsigned long lastPressTime = 0;            // 上次按键时间
const unsigned long debounceDelay = 50;     // 消抖时间 (ms)
unsigned long lastPressTime2 = 0;           // 上次按键时间
const unsigned long debounceDelay2 = 50; // 消抖时间 (ms)
```

（11） ADC电量检测定义

```
// 分压器比例（输入电压到 ADC 电压的比例）
const float voltageDividerRatio = 8.4; // 分压比（8.4倍缩小）
// 电压范围（电池电压）
const float minVoltage = 6.4; // 电压为0%时
const float maxVoltage = 8.4; // 电压为100%时
// 采样次数
const int numSamples = 10;
float batteryVoltage = 0; // 计算电池电压
int batteryPercentage=0;
```

（12）其他定义

```
bool freestate = false;// 定义一个标志来检查是否需要执行动作
int prevEmojiState = -1; // 用于跟踪之前的 emojiState
int actionstate = 0;//用于跟踪活动状态
int emojiState = 0; // 表情状态
```

4.6.3 初始化

首先，我们先完成一些初始化设定

```
void setup() {
//在此写入初始化逻辑
}
```

（1）初始化串口通讯

启动串口通信并设置波特率为 9600，这使得可以通过串口监视器查看调试信息。

```
Serial.begin(9600);
```

（2）初始化舵机

```
    servo1.attach(engine1);
    servo2.attach(engine2);
    servo3.attach(engine3);
    servo4.attach(engine4);
```

（3）初始化SPIFFS文件系统

```
SPIFFS.begin();
```

（4）初始化OLED显示

```
u8g2.begin();
```

（5）初始化WIFI

```
WiFi.softAP(ssid, password);
```

（6）启动WebServer服务

```
loadWiFiConfig();
```

这个方法还没定义，会在后面进行定义。

（7）按键初始化

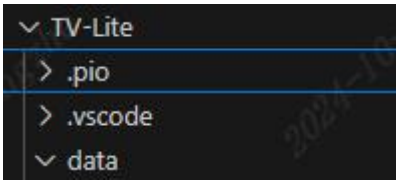
```
pinMode(BUTTON_PIN, INPUT_PULLUP); // GPIO2 设置为输入并启用内部上拉电阻
    attachInterrupt(digitalPinToInterrupt(BUTTON_PIN), handleButtonPress, FALLING);
    pinMode(BUTTON_PIN2, INPUT); // GPIO15 设置为输入（无需内部上拉）

    // 监听上升沿触发中断
    attachInterrupt(digitalPinToInterrupt(BUTTON_PIN2), handleButtonPress, RISING); // 设置下降沿中断
```

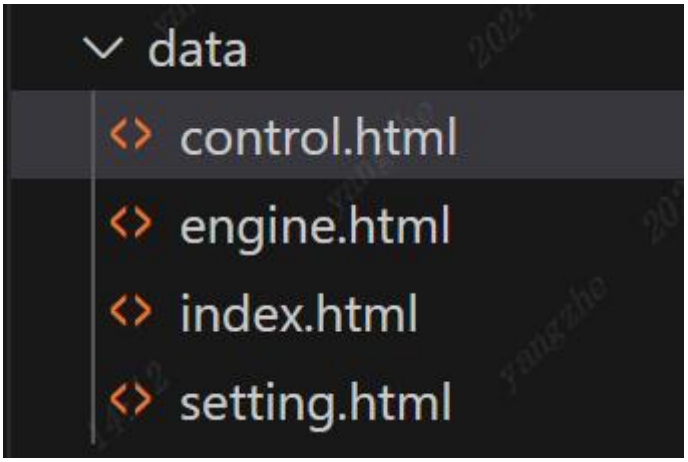
4.6.4 页面开发

（1）配置框架

1. 在项目目录下我们新建一个data文件夹，用于存放要烧录到FS文件系统的文件



2. 在data文件夹下新建一个html文件，用于存放我们的配网页面



（2）index.html索引页

为了布局美化，我们简单设计了几个style，用于美化UI


```
<!DOCTYPE html>
<html lang='en'>
<head>
    <meta charset='UTF-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <title>EDA-Robot</title>
    <style>

        body {
```

```
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
        }

        .container {
            max-width: 800px;
            margin: 0 auto;
            padding: 20px;
            text-align: center;
        }

        h1 {
            text-align: center;
        }

        .button {
            display: inline-block;
            height: 30px;
            width: 300px;
            margin-top: 20px;

            padding: 10px 20px;
            background-color: deepskyblue;
            color: #fff;
            border: none;
            border-radius: 20px; /* 添加圆角 */
            text-decoration: none;
            line-height: 2; /* 通过调整line-height的值来调整文字的垂直位置 */
            text-align: center; /* 文字居中 */
            box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.2); /* 添加立体感 */
            transition: all 0.3s ease; /* 添加过渡效果 */
        }

        .button:hover {
            background-color: skyblue; /* 鼠标悬停时的背景颜色 */
            transform: translateY(2px); /* 点击效果 */
            box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.3); /* 添加更多立体感 */
        }

    </style>

</head>
<body>

<div class='container'>
    <h1>EDA-Robot控制中心</h1>
    <p>本项目基于ESP8266主控开发</p>
```

```
    <input type='button'    style="height: 50px;width: 320px"    class='button'    value="遥控
器"onclick="window.location.href='./control.html'">
    <input type='button'    style="height: 50px;width: 320px"    class='button'    value="配置
"onclick="window.location.href='./setting.html'">
    <input type='button'    style="height: 50px; width: 320px"    class='button'    value="电机
校准"onclick="window.location.href='./engine.html'">
    <input type='button'    style="height: 50px; width: 320px"    class='button'    value="嘉立
创EDA">
    <a href="https://lceda.cn/">
        <table class="container button" style="height: 200px">
            <tr>
                <th>设备名称:</th>
                <td>EDA-Robot</td>
            </tr>
            <tr>
                <th>内存大小:</th>
                <td>4MB</td>
            </tr>
            <tr>
                <th>控制台版本:</th>
                <td>V1.0</td>
            </tr>
            <tr>
                <th>官网:</th>
                <td> <a href="https://lceda.cn/">嘉立创EDA</a></td>
            </tr>

        </table>
    </a>
</div>

</body>
</html>
```

（3）setting.html控制页

这里我们使用from表单的形式提交设置数据


```
<!DOCTYPE html>
<html lang='en'>
<head>
    <meta charset='UTF-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <title>EDA-Robot</title>
    <style>

        body {
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
        }

        .container {
            max-width: 800px;
            margin: 0 auto;
            padding: 20px;
            text-align: center;
        }

        h1 {
            text-align: center;
        }

        .button {
            display: inline-block;
            height: 30px;
            width: 300px;
            margin-top: 20px;
```

```
            padding: 10px 20px;
            background-color: deepskyblue;
            color: #fff;
            border: none;
            border-radius: 20px; /* 添加圆角 */
            text-decoration: none;
            line-height: 2; /* 通过调整line-height的值来调整文字的垂直位置 */
            text-align: center; /* 文字居中 */
            box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.2); /* 添加立体感 */
            transition: all 0.3s ease; /* 添加过渡效果 */
        }

        .button:hover {
            background-color: skyblue; /* 鼠标悬停时的背景颜色 */
            transform: translateY(2px); /* 点击效果 */
            box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.3); /* 添加更多立体感 */
        }
        .search-box {
            margin-top: 20px;
            display: inline-block;
            height: 30px;
            width: 300px;
            padding: 5px 10px;
            background-color: #f0f0f0;
            border: 1px solid #ccc;
            border-radius: 20px;
            text-align: center; /* 文字居中 */
        }
        .hidden {
            display: none; /* 初始隐藏 */
        }
    </style>

</head>
<body>
<form action='/connect' method='POST'>
    <div class='container'>
        <h1>EDA-Robot设备配置页</h1>
        <p>本项目基于ESP8266主控开发</p>
        <input type='text' name='ssid' placeholder='输入WIFI名称' class='search-box'>
        <input type='password' name='pass' placeholder='输入WIFI密码' class='search-box'>
        <input type='uid' name='uid' placeholder='输入bilibili用户ID' class='search-box'>
        <input type='api' name='api' placeholder='输入心知天气API密钥' class='search-
box'>
        <input type='city' name='city' placeholder='输入城市拼音小写' class='search-box'>
        <input type='submit'    style="height: 50px;width: 320px"    class='button'    value="
保存">
```

```
            <a href="https://lceda.cn/">
                <table class="container button" style="height: 200px">
                    <tr>
                        <th>设备名称:</th>
                        <td>EDA-Robot</td>
                    </tr>
                    <tr>
                        <th>内存大小:</th>
                        <td>4MB</td>
                    </tr>
                    <tr>
                        <th>控制台版本:</th>
                        <td>V1.0</td>
                    </tr>
                    <tr>
                        <th>官网:</th>
                        <td> <a href="https://lceda.cn/">嘉立创EDA</a></td>
                    </tr>

                </table>
            </a>
        </div>
    </form>
    </body>
    </html>
```

（4）engine.html舵机校准页

***180舵机版本的程序中删除了该页面，因为180舵机自带限位器，无需校准**

这里我们简单配置了几个script，用于触发路由，确保点击按钮后既不跳转又能触发事件

```
<!DOCTYPE html>
<html lang='en'>
```

```
<head>
    <meta charset='UTF-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <title>EDA-Robot</title>
    <style>

        body {
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
        }

        .container {
            max-width: 800px;
            margin: 0 auto;
            padding: 20px;
            text-align: center;
        }

        h1 {
            text-align: center;
        }

        button {
            display: inline-block;
            height: auto;
            width: auto;
            margin-top: 20px;

            padding: 10px 20px;
            background-color: deepskyblue;
            color: #fff;
            border: none;
            border-radius: 20px; /* 添加圆角 */
            text-decoration: none;
            line-height: 2; /* 通过调整line-height的值来调整文字的垂直位置 */
            text-align: center; /* 文字居中 */
            box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.2); /* 添加立体感 */
            transition: all 0.3s ease; /* 添加过渡效果 */
        }

        button:hover {
            background-color: skyblue; /* 鼠标悬停时的背景颜色 */
            transform: translateY(2px); /* 点击效果 */
            box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.3); /* 添加更多立体感 */
        }
        .button-grid2 {
            display: grid;
```

```
            grid-template-columns: repeat(2, 1fr);
            gap: 10px;
            justify-content: center;
            align-content: center;
            text-align: center;
            margin: 20px;
        } .button-grid1 {
                display: grid;
                border-radius: 20px; /* 添加圆角 */
                grid-template-columns: repeat(1, 1fr);
                justify-content: center;
                align-content: center;
                text-align: center;
                margin: 10px;

          }

    </style>
    <script>
        // 简化 AJAX 请求函数
        function sendCommand(action) {
            fetch(`/${action}`)
                .then(response => response.text())
                .catch(() => alert('发送失败，请检查设备连接'));
        }

        function refreshState(url, displayElementId) {
            fetch(url)
                .then(response => response.text())
                .then(data => {
                    document.getElementById(displayElementId).innerText = data;
                });
        }

        function setRefreshInterval(url, displayElementId) {
            setInterval(() => refreshState(url, displayElementId), 1000);
        }

        const states = [
            { url: '/engine1offsetleftpwm', displayId: 'engine1offsetleftpwmDisplay' },
            { url: '/engine1offsetrightpwm', displayId: 'engine1offsetrightpwmDisplay' },
            { url: '/engine2offsetleftpwm', displayId: 'engine2offsetleftpwmDisplay' },
            { url: '/engine2offsetrightpwm', displayId: 'engine2offsetrightpwmDisplay' },
            { url: '/engine3offsetleftpwm', displayId: 'engine3offsetleftpwmDisplay' },
            { url: '/engine3offsetrightpwm', displayId: 'engine3offsetrightpwmDisplay' },
            { url: '/engine4offsetleftpwm', displayId: 'engine4offsetleftpwmDisplay' },
            { url: '/engine4offsetrightpwm', displayId: 'engine4offsetrightpwmDisplay' },
            { url: '/speed', displayId: 'speedDisplay' }
        ];
```

```
            states.forEach(state => setRefreshInterval(state.url, state.displayId));

        </script>
</head>
<body>
<div class='container'>
        <h1>EDA-Robot校准中心</h1>
        <p>电机校准页</p >
        <div class="button-grid1" style="background-color: rgb(0, 140, 255)">
            <h3>校准</h3>
            <div class="button-grid1">
            <div>
                <p>电机转速: <span id="speedDisplay">0</span></p>
                <button onclick="sendCommand('speeddown')">减少</button>
                <button onclick="sendCommand('speedup')">增加</button>
            </div>
            </div>
            <div class="button-grid2">
                <div>
                    <p>电机1左转补偿: <span
id="engine1offsetleftpwmDisplay">0</span></p>
                    <button onclick="sendCommand('engine1offsetleftpwmdown')">减少
</button>
                    <button onclick="sendCommand('engine1offsetleftpwmup')">增加
</button>
                </div>              <div>
                <p>电机1右转补偿: <span id="engine1offsetrightpwmDisplay">0</span></p>
                <button onclick="sendCommand('engine1offsetrightpwmdown')">减少
</button>
                <button onclick="sendCommand('engine1offsetrightpwmup')">增加</button>
            </div>              <div>
                <p>电机2左转补偿: <span id="engine2offsetleftpwmDisplay">0</span></p>
                <button onclick="sendCommand('engine2offsetleftpwmdown')">减少</button>
                <button onclick="sendCommand('engine2offsetleftpwmup')">增加</button>
            </div>              <div>
                <p>电机2右转补偿: <span id="engine2offsetrightpwmDisplay">0</span></p>
                <button onclick="sendCommand('engine2offsetrightpwmdown')">减少
</button>
                <button onclick="sendCommand('engine2offsetrightpwmup')">增加</button>
            </div>              <div>
                <p>电机3左转补偿: <span id="engine3offsetleftpwmDisplay">0</span></p>
                <button onclick="sendCommand('engine3offsetleftpwmdown')">减少</button>
                <button onclick="sendCommand('engine3offsetleftpwmup')">增加</button>
            </div>              <div>
                <p>电机3右转补偿: <span id="engine3offsetrightpwmDisplay">0</span></p>
```

```
                <button onclick="sendCommand('engine3offsetrightpwmdown')">减少
</button>
                <button onclick="sendCommand('engine3offsetrightpwmup')">增加</button>
            </div>              <div>
                <p>电机4左转补偿: <span id="engine4offsetleftpwmDisplay">0</span></p>
                <button onclick="sendCommand('engine4offsetleftpwmdown')">减少</button>
                <button onclick="sendCommand('engine4offsetleftpwmup')">增加</button>
            </div>              <div>
                <p>电机4右转补偿: <span id="engine4offsetrightpwmDisplay">0</span></p>
                <button onclick="sendCommand('engine4offsetrightpwmdown')">减少
</button>
                <button onclick="sendCommand('engine4offsetrightpwmup')">增加</button>
            </div>
                <button onclick="sendCommand('left90')">电机左转90度</button>
                <button onclick="sendCommand('right90')">电机右转90度</button>
            </div>
        </div>
</div>
</body>
</html>
```

（5）control.html控制页

控制页面是控制机械狗的主要页面，同样使用几个script来简化请求，避免网页代码量过多，大小过大，导致FS文件系统读取较慢，无法响应等问题。


```
<!DOCTYPE html>
<html lang='en'>
<head>
    <meta charset='UTF-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <title>EDA-Robot</title>
    <style>

        body {
            margin: 0;
            padding: 0;
            font-family: Arial, sans-serif;
        }

        .container {
            max-width: 800px;
            margin: 0 auto;
            padding: 20px;
            text-align: center;
        }
```

```
h1 {
    text-align: center;
}

button {
    display: inline-block;
    height: auto;
    width: auto;
    margin-top: 20px;

    padding: 10px 20px;
    background-color: deepskyblue;
    color: #fff;
    border: none;
    border-radius: 20px; /* 添加圆角 */
    text-decoration: none;
    line-height: 2; /* 通过调整line-height的值来调整文字的垂直位置 */
    text-align: center; /* 文字居中 */
    box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.2); /* 添加立体感 */
    transition: all 0.3s ease; /* 添加过渡效果 */
}

button:hover {
    background-color: skyblue; /* 鼠标悬停时的背景颜色 */
    transform: translateY(2px); /* 点击效果 */
    box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.3); /* 添加更多立体感 */
}
.button-grid3 {
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    gap: 10px;
    justify-content: center;
    align-content: center;
    text-align: center;
    margin: 20px;
}
.button-grid2 {
    display: grid;
    grid-template-columns: repeat(2, 1fr);
    gap: 10px;
    justify-content: center;
    align-content: center;
    text-align: center;
    margin: 20px;
} .button-grid1 {
        display: grid;
        border-radius: 20px; /* 添加圆角 */
```

```
                grid-template-columns: repeat(1, 1fr);
                justify-content: center;
                align-content: center;
                text-align: center;
                margin: 10px;

            }

    </style>
    <script>
        // 简化 AJAX 请求函数
        function sendCommand(action) {
            fetch(`/${action}`)
                .then(response => response.text())
                .catch(() => alert('发送失败，请检查设备连接'));
        }

        function refreshState(url, displayElementId) {
            fetch(url)
                .then(response => response.text())
                .then(data => {
                    document.getElementById(displayElementId).innerText = data;
                });
        }

        function setRefreshInterval(url, displayElementId) {
            setInterval(() => refreshState(url, displayElementId), 1000);
        }

        const states = [
            { url: '/batteryVoltage', displayId: 'batteryVoltageDisplay' },
            { url: '/batteryPercentage', displayId: 'batteryPercentageDisplay' },
        ];
        states.forEach(state => setRefreshInterval(state.url, state.displayId));

    </script>
</head>
<body>
<div class='container'>
    <h1>EDA-Robot遥控台</h1>
    <p>本项目基于ESP8266主控开发</p >
  <div class="button-grid2" style="display: flex; justify-content: center;">
            <button style="margin-right: 70px"><p>电压：<span
id="batteryVoltageDisplay">0</span></p></button>
            <button style="margin-left: 70px"><p>电量：<span
id="batteryPercentageDisplay">0</span></p></button>
        </div>
    <!-- 运动控制 -->
```

```
    <div class="button-grid1" style="background-color: papayawhip">
        <h3>运动控制</h3>
        <div class="button-grid1"style="display: flex; justify-content: center;">
            <button    onclick="sendCommand('front')">↑</button>
        </div>
        <div class="button-grid2" style="display: flex; justify-content: center;">
            <button    onclick="sendCommand('left')" style="margin-right:
70px">←</button>
            <button    onclick="sendCommand('right')" style="margin-left:
70px">→</button>
        </div>
        <div class="button-grid1" style="display: flex; justify-content: center;">
            <button    onclick="sendCommand('back')">↓</button>
        </div>
        <div    class="button-grid3">
            <button onclick="sendCommand('toplefthand')">抬左手</button>
            <button onclick="sendCommand('toprighthand')">抬右手</button>
            <button onclick="sendCommand('sitdown')">坐下</button>
            <button onclick="sendCommand('lie')">趴下</button>

            <button onclick="sendCommand('free')">自由模式开</button>
            <button onclick="sendCommand('offfree')">自由模式关</button>
        </div>
    </div>

    <!-- 表情控制 -->
    <div class="button-grid1" style="background-color: limegreen">
        <h3>表情控制</h3>
        <div class="button-grid3" >
            <button onclick="sendCommand('histate')">开心</button>
            <button onclick="sendCommand('angrystate')">生气</button>
            <button onclick="sendCommand('sickstate')">难受</button>
            <button onclick="sendCommand('dowhatstate')">好奇</button>
            <button onclick="sendCommand('lovestate')">喜欢</button>
            <button onclick="sendCommand('errorstate')">错误</button>
            <button onclick="sendCommand('yunstate')">晕</button>
        </div>
    </div>

    <!-- 联网功能 -->
    <div class="button-grid1" style="background-color: orange">
        <h3>联网功能</h3>
        <div class="button-grid2">
            <button onclick="sendCommand('time')">时间</button>
            <button onclick="sendCommand('weather')">天气</button>


        </div>
```

```
    </div>

</div>
</body>
</html>
```

4.6.5 路由配置逻辑

前面我们已经完成了页面开发部分，并在html内定义了一些路由节点，为了确保这些路由能正确触发并被我们监听到触发事件，我们还需要在cpp程序内定义路由监听事件，确保硬件舵机能正确执行命令。

```
void handleWiFiConfig()
{
    // 启动服务器
    server.on("/left90", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      actionstate = 10;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/right90", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      actionstate = 11;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/front", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      actionstate = 1;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/back", HTTP_GET, [](AsyncWebServerRequest *request)
              {
     actionstate = 4;    // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/left", HTTP_GET, [](AsyncWebServerRequest *request)
              {
     actionstate = 2;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/right", HTTP_GET, [](AsyncWebServerRequest *request)
              {
     actionstate = 3;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/toplefthand", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      actionstate = 5;    // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/toprighthand", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      actionstate = 6;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
    server.on("/sitdown", HTTP_GET, [](AsyncWebServerRequest *request)
              {
```

```
      actionstate = 8;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
server.on("/lie", HTTP_GET, [](AsyncWebServerRequest *request)
              {
   actionstate = 7;
      request->send(200, "text/plain", "Front function started"); });
// server.on("/dance", HTTP_GET, [](AsyncWebServerRequest *request)
//                 {
//        actionstate = 7;   // 设置标志，执行舵机动作
//        request->send(200, "text/plain", "Front function started"); });
server.on("/free", HTTP_GET, [](AsyncWebServerRequest *request)
              {
   freestate=true;
      request->send(200, "text/plain", "Front function started"); });
server.on("/offfree", HTTP_GET, [](AsyncWebServerRequest *request)
              {
   freestate=false;
      request->send(200, "text/plain", "Front function started"); });
server.on("/histate", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      emojiState = 0;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
server.on("/angrystate", HTTP_GET, [](AsyncWebServerRequest *request)
              {
      emojiState = 1;    // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });

server.on("/errorstate", HTTP_GET, [](AsyncWebServerRequest *request)
              {
    emojiState = 2;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
server.on("/engine1offsetleftpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine1offsetleftpwm)); });
server.on("/engine2offsetleftpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine2offsetleftpwm)); });
server.on("/engine3offsetleftpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine3offsetleftpwm)); });
server.on("/engine4offsetleftpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine4offsetleftpwm)); });
server.on("/engine1offsetrightpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine1offsetrightpwm)); });
server.on("/engine2offsetrightpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine2offsetrightpwm)); });
server.on("/engine3offsetrightpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine3offsetrightpwm)); });
server.on("/engine4offsetrightpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine4offsetrightpwm)); });
server.on("/engine4offsetrightpwm", HTTP_GET, [](AsyncWebServerRequest *request)
              { request->send(200, "text/plain", String(engine4offsetrightpwm)); });
```

```
    server.on("/batteryVoltage", HTTP_GET, [](AsyncWebServerRequest *request)
                { request->send(200, "text/plain", String(batteryVoltage)); });
    server.on("/batteryPercentage", HTTP_GET, [](AsyncWebServerRequest *request)
                { request->send(200, "text/plain", String(batteryPercentage)); });
  server.on("/speed", HTTP_GET, [](AsyncWebServerRequest *request)
                { request->send(200, "text/plain", String(speed)); });
    server.on("/speedup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        speed++;   // 设置标志，执行舵机动作
         request->send(200, "text/plain", "Front function started"); });
    server.on("/speeddown", HTTP_GET, [](AsyncWebServerRequest *request)
                {
    speed--;
         request->send(200, "text/plain", "Front function started"); });
          server.on("/engine1offsetrightpwmup", HTTP_GET, [](AsyncWebServerRequest
*request)
                {
        engine1offsetrightpwm++;   // 设置标志，执行舵机动作
         request->send(200, "text/plain", "Front function started"); });
    server.on("/engine1offsetrightpwmdown", HTTP_GET, [](AsyncWebServerRequest
*request)
                {
        engine1offsetrightpwm--;
         request->send(200, "text/plain", "Front function started"); });

    server.on("/engine1offsetleftpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine1offsetleftpwm++;   // 设置标志，执行舵机动作
         request->send(200, "text/plain", "Front function started"); });
    server.on("/engine1offsetleftpwmdown", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine1offsetleftpwm--;
         request->send(200, "text/plain", "Front function started"); });

    server.on("/engine2offsetrightpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine2offsetrightpwm++;   // 设置标志，执行舵机动作
         request->send(200, "text/plain", "Front function started"); });
    server.on("/engine2offsetrightpwmdown", HTTP_GET, [](AsyncWebServerRequest
*request)
                {
        engine2offsetrightpwm--;
         request->send(200, "text/plain", "Front function started"); });
    server.on("/engine2offsetleftpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine2offsetleftpwm++;   // 设置标志，执行舵机动作
         request->send(200, "text/plain", "Front function started"); });
    server.on("/engine2offsetleftpwmdown", HTTP_GET, [](AsyncWebServerRequest *request)
                {
```

```
        engine2offsetleftpwm--;
          request->send(200, "text/plain", "Front function started"); });

    server.on("/engine3offsetrightpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine3offsetrightpwm++;   // 设置标志，执行舵机动作
          request->send(200, "text/plain", "Front function started"); });
    server.on("/engine3offsetrightpwmdown", HTTP_GET, [](AsyncWebServerRequest
*request)
                {
        engine3offsetrightpwm--;
          request->send(200, "text/plain", "Front function started"); });
    server.on("/engine3offsetleftpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine3offsetleftpwm++;   // 设置标志，执行舵机动作
          request->send(200, "text/plain", "Front function started"); });
    server.on("/engine3offsetleftpwmdown", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine3offsetleftpwm--;
          request->send(200, "text/plain", "Front function started"); });

    server.on("/engine4offsetrightpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine4offsetrightpwm++;   // 设置标志，执行舵机动作
          request->send(200, "text/plain", "Front function started"); });
    server.on("/engine4offsetrightpwmdown", HTTP_GET, [](AsyncWebServerRequest
*request)
                {
        engine4offsetrightpwm--;
          request->send(200, "text/plain", "Front function started"); });
    server.on("/engine4offsetleftpwmup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine4offsetleftpwm++;   // 设置标志，执行舵机动作
          request->send(200, "text/plain", "Front function started"); });
    server.on("/engine4offsetleftpwmdown", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        engine4offsetleftpwm--;
          request->send(200, "text/plain", "Front function started"); });
    server.on("/speedup", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        speed++;   // 设置标志，执行舵机动作
          request->send(200, "text/plain", "Front function started"); });
    server.on("/speeddown", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        speed--;
          request->send(200, "text/plain", "Front function started"); });
    server.on("/dowhatstate", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        emojiState = 3;   // 设置标志，执行舵机动作
```

```
      request->send(200, "text/plain", "Front function started"); });
server.on("/lovestate", HTTP_GET, [](AsyncWebServerRequest *request)
              {
    emojiState = 4;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
server.on("/sickstate", HTTP_GET, [](AsyncWebServerRequest *request)
              {
    emojiState = 5;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });
server.on("/yunstate", HTTP_GET, [](AsyncWebServerRequest *request)
              {
    emojiState = 6;
      request->send(200, "text/plain", "Front function started"); });
server.on("/time", HTTP_GET, [](AsyncWebServerRequest *request)
              {
    emojiState = 8;
      request->send(200, "text/plain", "Front function started"); });
server.on("/weather", HTTP_GET, [](AsyncWebServerRequest *request)
              {
    emojiState = 7;   // 设置标志，执行舵机动作
      request->send(200, "text/plain", "Front function started"); });

server.on("/connect", HTTP_POST, [](AsyncWebServerRequest *request)
              {
      // 获取POST参数：ssid、pass、uid、city、api
      String ssid = request->getParam("ssid", true)->value();
      String pass = request->getParam("pass", true)->value();
      String uid = request->getParam("uid", true)->value();
      String city = request->getParam("city", true)->value();
      String api = request->getParam("api", true)->value();

      // 打印接收到的参数
      Serial.println(ssid);
      Serial.println(pass);

      // 保存WiFi信息到JSON文件
      DynamicJsonDocument doc(1024);
      doc["ssid"] = ssid;
      doc["pass"] = pass;
      doc["uid"] = uid;
      doc["city"] = city;
      doc["api"] = api;
      fs::File file = SPIFFS.open(ssidFile, "w");   // 打开文件进行写入
      if (file) {
            serializeJson(doc, file);   // 将JSON内容写入文件
            file.close();   // 关闭文件
      }
```

```
        // 更新全局变量
        useruid = uid;
        cityname = city;
        weatherapi = api;

        // 开始连接WiFi
        WiFi.begin(ssid.c_str(), pass.c_str());
        // 发送HTML响应，告知用户正在连接
        request->send(200, "text/html", "<h1>Connecting...</h1>"); });

    server.on("/", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        // 检查SPIFFS文件系统中是否存在index.html文件
        if (SPIFFS.exists("/index.html")) {
            fs::File file = SPIFFS.open("/index.html", "r");    // 打开index.html文件
            if (file) {
                size_t fileSize = file.size();    // 获取文件大小
                String fileContent;

                // 逐字节读取文件内容
                while (file.available()) {
                    fileContent += (char)file.read();
                }
                file.close();    // 关闭文件

                // 返回HTML内容
                request->send(200, "text/html", fileContent);
                return;
            }
        }
        // 如果文件不存在，返回404错误
        request->send(404, "text/plain", "File Not Found"); });
    server.on("/control.html", HTTP_GET, [](AsyncWebServerRequest *request)
                {
        // 检查SPIFFS文件系统中是否存在index.html文件
        if (SPIFFS.exists("/control.html")) {
            fs::File file = SPIFFS.open("/control.html", "r");    // 打开index.html文件
            if (file) {
                size_t fileSize = file.size();    // 获取文件大小
                String fileContent;

                // 逐字节读取文件内容
                while (file.available()) {
                    fileContent += (char)file.read();
                }
                file.close();    // 关闭文件

                // 返回HTML内容
```

```
            request->send(200, "text/html", fileContent);
            return;
        }
    }
    // 如果文件不存在，返回404错误
    request->send(404, "text/plain", "File Not Found"); });
server.on("/engine.html", HTTP_GET, [](AsyncWebServerRequest *request)
          {
    // 检查SPIFFS文件系统中是否存在index.html文件
    if (SPIFFS.exists("/engine.html")) {
        fs::File file = SPIFFS.open("/engine.html", "r");  // 打开index.html文件
        if (file) {
            size_t fileSize = file.size();  // 获取文件大小
            String fileContent;

            // 逐字节读取文件内容
            while (file.available()) {
                fileContent += (char)file.read();
            }
            file.close();  // 关闭文件

            // 返回HTML内容
            request->send(200, "text/html", fileContent);
            return;
        }
    }
    // 如果文件不存在，返回404错误
    request->send(404, "text/plain", "File Not Found"); });
server.on("/setting.html", HTTP_GET, [](AsyncWebServerRequest *request)
          {
    // 检查SPIFFS文件系统中是否存在index.html文件
    if (SPIFFS.exists("/setting.html")) {
        fs::File file = SPIFFS.open("/setting.html", "r");  // 打开index.html文件
        if (file) {
            size_t fileSize = file.size();  // 获取文件大小
            String fileContent;

            // 逐字节读取文件内容
            while (file.available()) {
                fileContent += (char)file.read();
            }
            file.close();  // 关闭文件

            // 返回HTML内容
            request->send(200, "text/html", fileContent);
            return;
        }
    }
```

```
        // 如果文件不存在，返回404错误
        request->send(404, "text/plain", "File Not Found"); });
    // 启动服务器
    server.begin();
};
```

handleWiFiConfig()方法就是我们定义的路由，这里是所有web路由监听事件，在配置路由时务必确保html的触发事件和cpp程序的监听事件路由一致，否则可能出现触发不生效问题。在路由中我们通过emojiState记录表情事件，通过actionstate记录运动事件，因为如果直接在路由内编写运动事件可能会导致html无响应，触发看门狗死机重启。因此我们通过记录事件然后再到loop中执行，确保异步操作正常。

4.6.6 WIFI配置加载

在前面的handleWiFiConfig()中，我们定义的/connect路由中，我们将设置信息保存在FS文件系统中，为了方便用户在断电重启后避免二次配置，所以还需要写一个将FS保存的WIFI配置同步到全局定义上。

```
void loadWiFiConfig()
{
    if (SPIFFS.begin())
    {
        fs::File file = SPIFFS.open(ssidFile, "r");
        if (file)
        {
            DynamicJsonDocument doc(1024);
            DeserializationError error = deserializeJson(doc, file);
            if (!error)
            {
                String ssid = doc["ssid"];
                String pass = doc["pass"];
                String uid = doc["uid"];
                String city = doc["city"];
                String api = doc["api"];
                useruid = uid;
                cityname = city;
                weatherapi = api;
                WiFi.begin(ssid.c_str(), pass.c_str());
                // 尝试连接WiFi，最多等待10秒
                unsigned long startAttemptTime = millis();
                while (WiFi.status() != WL_CONNECTED && millis() - startAttemptTime <
5000)
                {
                    delay(500);
                }
                // 如果连接失败，打印状态
                if (WiFi.status() != WL_CONNECTED)
                {
                    Serial.println("WiFi connection failed, starting captive portal...");
                    handleWiFiConfig(); // 启动强制门户
```

```
                    }
                    else
                    {
                        Serial.println("WiFi connected");
                        timeClient.begin();
                    }
                }
                file.close();
            }
        }
    }
```

4.6.7 天气

用于从网络 API 获取天气数据，并根据当前的天气状况在屏幕上显示相应的图标和数据（温度、湿度等）。

天气的程序和Bilibili的逻辑大致相似，都是先GET请求，再截取返回的Json值，然后做个简单的判断再显示即可

（1）创建方法

```
void fetchWeather() {
//在此写入逻辑代码
}
```

（2）检查天气数据是否已初始化

```
if(initweather == false) {
mylcd.fillScreen(TFT_BLACK);
```

通过检查 initweather 是否为 false 来确定是否需要获取天气数据。如果未初始化（即 initweather == false），则清空屏幕为黑色背景，准备显示天气数据。

（3）检查 WiFi 连接状态

```
if (WiFi.status() == WL_CONNECTED) {
WiFiClient client;
HTTPClient http;
```

确认设备已连接 WiFi，才能发送请求获取天气数据。创建 WiFi 客户端 client 和 HTTP 客户端 http，以用于发送和接收数据。

（4）构建并发送 HTTP 请求

```
if (http.begin(client, weatherAPI + weatherapi + "&location=" + cityname +
"&language=zh-Hans&unit=c&start=0&days=1")) {
int httpCode = http.GET();
```

使用 http.begin() 函数创建 HTTP GET 请求，将 API 密钥、城市名称、语言设置等拼接到 URL 中。http.GET() 发送 GET 请求，返回 HTTP 响应代码 httpCode 以确认请求成功与否。

（5）解析服务器响应并提取数据

```
if (httpCode > 0) {
String payload = http.getString();
Serial.println("JSON Response:");
Serial.println(payload);
DynamicJsonDocument doc(1024);
deserializeJson(doc, payload);
```

如果 httpCode > 0 表示请求成功，将服务器响应内容保存到 payload。

（6）从 JSON 中获取温度、湿度和天气状况

```
String temperature2 = doc["results"][0]["daily"][0]["high"];
String humidity2 = doc["results"][0]["daily"][0]["humidity"];
String weathe2r = doc["results"][0]["daily"][0]["text_day"];
```

从 JSON 数据中提取每日最高温度、湿度和白天天气状况。

（7）保存数据并更新初始化状态

```
temperature = temperature2;
humidity = humidity2;
weather = weathe2r;
initweather = true;
```

将数据保存到全局变量 temperature、humidity 和 weather 中，并将 initweather 标记为 true，表示天气数据已获取成功。

（8）数据打印与错误处理

```
Serial.print("Data received: "); // 打印数据
```

```
Serial.println(temperature);
Serial.println( humidity);
Serial.println( weather);
} else { // 如果请求失败
Serial.printf("HTTP GET request failed, error: %s\n",
http.errorToString(httpCode).c_str());
}
http.end(); // 结束HTTP客户端
} else {
Serial.println("Failed to connect to server");
}
}
}
```

（9）显示对应天气的图标

```
if (weather == "阴" || weather == "多云")
    {
        do
        {
            u8g2.setFont(u8g2_font_ncenB08_tr);
            u8g2.drawXBMP(0, 0, 64, 64, cloud);
            u8g2.drawStr(64, 20, "Temp");
            String temperatureString = String(temperature) + " C";
            u8g2.drawStr(64, 30, temperatureString.c_str());
            u8g2.drawStr(64, 50, "Humidity");
            String humidityString = String(humidity) + " %";
            u8g2.drawStr(64, 60, humidityString.c_str());
        } while (u8g2.nextPage());
        // mylcd.pushImage(mylcd.width() / 2-(99/2), 10, 99, 99,cloud);
    }
    else if (weather == "小雨" || weather == "大雨" || weather == "暴雨" ||
weather == "雨")
    {
        do
        {
            u8g2.setFont(u8g2_font_ncenB08_tr);
            u8g2.drawXBMP(0, 0, 64, 64, rain);
            u8g2.drawStr(64, 20, "Temp");
            String temperatureString = String(temperature) + " %";
            u8g2.drawStr(64, 30, temperatureString.c_str());
            u8g2.drawStr(64, 50, "Humidity");
            String humidityString = String(humidity) + " %";
            u8g2.drawStr(64, 60, humidityString.c_str());
```

```
        } while (u8g2.nextPage());
    }
    else if (weather == "晴")
    {
        do
        {
            u8g2.setFont(u8g2_font_ncenB08_tr);
            u8g2.drawStr(64, 20, "Temp");
            u8g2.drawXBMP(0, 0, 64, 64, sun);
            String temperatureString = String(temperature) + " %";
            u8g2.drawStr(64, 30, temperatureString.c_str());
            u8g2.drawStr(64, 50, "Humidity");
            String humidityString = String(humidity) + " %";
            u8g2.drawStr(64, 60, humidityString.c_str());
        } while (u8g2.nextPage());
    }
```

4.6.8 运动控制

机械狗的运动控制不像汽车轮子那样简单，我们需要观察四足动物行走，进行仿生模拟。

（1）前进

前进的逻辑如下，先将机械狗的2和3号腿向前迈出，再将1和4腿向后蹬出，然后再将2和3腿归位，最后将3和4腿归位即可。这里的speed就是速度，1500是中位值，越远离1500则舵机转速越快，越靠近1500则舵机速度越慢。而且由于SG90舵机是普通电机驱动的，没有磁编码器，所以没有精准的电机控制功能，不能向PID那样精确，每个舵机都会受到摩擦力，导线长短，电流稳定性或各种各样的外部因素影响，所以，为了确保4个舵机能同步，还加入了engineXoffsetXpwm，用于控制各个电机的补偿，这就需要我们耐心去找到适宜的补偿值。4个电机的转动角度一致。

```
void front()
{
    //+30C 2/3
    servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
    servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    delay(500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    //-30C 1/4
    servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
    delay(500);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    // 0C 2/3
    servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
    servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
```

```
    delay(500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    // 0C 1/4
    servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
    delay(500);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    //+30C 1/4
    servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
    delay(500);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    //-30C 2/3
    servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
    servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
    delay(500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    // 0C 1/4
    servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
    delay(500);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    // 0C 2/3
    servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
    servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    delay(500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
}
```

（2）后退

后退的逻辑和前进基本相同，只需要把前进的逻辑反过来就行了。

```
void back()
{

    //+30C 2/3
    servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
    servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
    delay(500-runtime);
```

```
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    //-30C 1/4
    servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
    delay(500-runtime);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    // 0C 2/3
    servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
    servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    delay(500-runtime);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    // 0C 1/4
    servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
    delay(500-runtime);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);

    //+30C 1/4
    servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
    delay(500-runtime);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    //-30C 2/3
    servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
    servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    delay(500-runtime);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    // 0C 1/4
    servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
    delay(500-runtime);
    servo1.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    // 0C 2/3
    servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
    servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
    delay(500-runtime);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
}
```

（3）左转

左转的逻辑略有不同，先要让1和4腿前进30度，再等待2S让2和3腿前进30度，然后再使其归位，重复操作4次，确保转动角度明显。

```
void left()
{
	int num = 0;
	while (num < 4)
	{
		//   +30
		servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
		servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
		delay(200);
		servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
		servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
		delay(300);
		servo1.writeMicroseconds(1500);
		servo4.writeMicroseconds(1500);
		delay(200);
		servo2.writeMicroseconds(1500);
		servo3.writeMicroseconds(1500);
		delay(100);
		//-30
		servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
		servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
		delay(200);
		servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
		servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
		delay(300);
		servo1.writeMicroseconds(1500);
		servo4.writeMicroseconds(1500);
		delay(200);
		servo2.writeMicroseconds(1500);
		servo3.writeMicroseconds(1500);
		num++;
	}
}
```

（4）右转

右转逻辑与左转相同，只需要把左转的反过来即可

```
void right()
{

	int num = 0;
```

```
    while (num < 4)
    {
        //   +30
        servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
        servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
        delay(200);
        servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
        servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
        delay(300);
        servo1.writeMicroseconds(1500);
        servo4.writeMicroseconds(1500);
        delay(200);
        servo2.writeMicroseconds(1500);
        servo3.writeMicroseconds(1500);
        delay(100);
        //-30
        servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
        servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
        delay(200);
        servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
        servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
        delay(300);
        servo1.writeMicroseconds(1500);
        servo4.writeMicroseconds(1500);
        delay(200);
        servo2.writeMicroseconds(1500);
        servo3.writeMicroseconds(1500);
        num++;
    }
}
```

（5）坐下

坐下的逻辑就比较简单了，直接让2和4腿向前旋转90度坐下即可。

```
void sitdown()
{

    servo2.writeMicroseconds(1500 + speed + engine3offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    delay(1000);
    servo2.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    delay(3000);
    servo2.writeMicroseconds(1500 - speed - engine3offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
```

```
    delay(1000);
    servo2.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
}
```

（6）趴下

趴下逻辑较为简单，1和3腿左转90，2和4腿右转90，使机械狗水平趴下。

```
void lie()
{

    servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
    servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
    delay(1000);
    servo1.writeMicroseconds(1500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
    delay(3000);
    servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
    servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
    servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
    delay(1000);
    servo1.writeMicroseconds(1500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
}
```

（7）举起左手

举起左手的逻辑也很简单，使3腿旋转90度停顿再归位，往返3次

```
void toplefthand()
{
    int num = 0;
    while (num < 3)
    {
        servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
        delay(1000);
        servo3.writeMicroseconds(1500);
```

```
            delay(500);
            servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
            delay(1000);
            servo3.writeMicroseconds(1500);
            num++;
        }
    }
```

（8）举起右手

举起右手和左手一致，参考左手逻辑修改电机编号即可

```
void toprighthand()
{

    int num = 0;
    while (num < 3)
    {
        servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
        delay(1000);
        servo1.writeMicroseconds(1500);
        delay(500);
        servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
        delay(1000);
        servo1.writeMicroseconds(1500);
        num++;
    }
}
```

（9）舵机同步左转

舵机同步左转需要考虑舵机摆放位置，因为1和2电机与3和4电机摆放相反，所以要留意加减法方向

```
void left90()
{
    servo1.writeMicroseconds(1500 + speed + engine1offsetleftpwm);
    servo2.writeMicroseconds(1500 + speed + engine2offsetleftpwm);
    servo3.writeMicroseconds(1500 - speed - engine3offsetleftpwm);
    servo4.writeMicroseconds(1500 - speed - engine4offsetleftpwm);
    delay(500);
    servo1.writeMicroseconds(1500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
```

```
}
```

（10）舵机同步右转

右转逻辑参考左转，基本一致，改变加减法符号即可

```
void right90()
{

    servo1.writeMicroseconds(1500 - speed - engine1offsetrightpwm);
    servo2.writeMicroseconds(1500 - speed - engine2offsetrightpwm);
    servo3.writeMicroseconds(1500 + speed + engine3offsetrightpwm);
    servo4.writeMicroseconds(1500 + speed + engine4offsetrightpwm);
    delay(500);
    servo1.writeMicroseconds(1500);
    servo2.writeMicroseconds(1500);
    servo3.writeMicroseconds(1500);
    servo4.writeMicroseconds(1500);
}
```

4.6.9 ADC电量检测

（1）电压采样平均

```
float getAverageAdcVoltage() {
  long totalAdcValue = 0;

  // 多次采样
  for (int i = 0; i < numSamples; i++) {
    totalAdcValue += analogRead(A0); // 读取 ADC 数据
    delay(10); // 每次采样间隔 10ms
  }

  // 计算平均 ADC 值
  float averageAdcValue = totalAdcValue / (float)numSamples;

  // 将 ADC 值转换为电压
  return (averageAdcValue / 1023.0) * 1.0; // ESP8266 的参考电压为 1.0V
}
```

（2）电量转换计算

```
// 计算电池电量百分比的函数
```

```
int mapBatteryPercentage(float voltage) {
   if (voltage <= minVoltage) return 0;     // 小于等于最小电压时，电量为 0%
   if (voltage >= maxVoltage) return 100; // 大于等于最大电压时，电量为 100%

   // 根据线性比例计算电量百分比
   return (int)((voltage - minVoltage) / (maxVoltage - minVoltage) * 100);
}
```

4.6.10 运行程序逻辑

Loop程序是一个无限循环的程序，它会在setup初始程序执行后进入，在loop内程序会无限循环，我们通常在这里编写主要的逻辑。

（1）屏幕刷新逻辑

为了确保每次执行表情前都能刷新一次屏幕，但又不能在循环中无限刷新，所有必须设置一个逻辑，确保屏幕切换时只执行一次清屏，如果是循环清屏会导致屏幕出现闪烁，影响显示效果。

```
if (emojiState != prevEmojiState)
      {
            u8g2.clearDisplay();              // 状态变化时清屏
            prevEmojiState = emojiState; // 更新状态
      }
```

（2）actionstate状态逻辑

在前面，为了确保网页异步，防止长时间未响应触发看门狗，于是我们设置了actionstate用于记录运动状态，使其在loop中执行。所以在这里我们需要为actionstate指定需要执行的运动，要确保actionstate和前面在网页配置的按钮名字相同，不要点击前进按钮但机械狗实际向后退的问题。

```
      switch (actionstate)
      {
      case 0 /* constant-expression */:
            /* code */
            break;
      case 1:
            front(); // 执行一次舵机动作
            actionstate = 0;
            break;
      case 2:
            left(); // 执行一次舵机动作
            actionstate = 0;
            break;
      case 3:
            right(); // 执行一次舵机动作
            actionstate = 0;
```

```
        break;
    case 4:
        back(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 5:
        toplefthand(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 6:
        toprighthand(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 10:
        left90(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 11:
        right90(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 7:
        lie(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 8:
        sitdown(); // 执行一次舵机动作
        actionstate = 0;
        break;
    case 9:
        emojiState = random(0, 7); // 执行一次舵机动作
        actionstate = 0;
        break;
    default:
        break;
    }
```

（3）emojiState表情逻辑

在前面的网页监听事件中，表情控制使用的是emojiState，所以我们还要指定emojiState的对应表情

```
switch (emojiState)
    {
    case 0: // 首页
        u8g2.setFont(u8g2_font_ncenB14_tr);
        do
```

```
        {

            u8g2.drawXBMP(0, 0, 128, 64, hi);
        } while (u8g2.nextPage());

        break;
    case 1: //  第二页

        u8g2.setFont(u8g2_font_ncenB14_tr);
        do
        {

            u8g2.drawXBMP(0, 0, 128, 64, angry);
        } while (u8g2.nextPage());

        break;
    case 2: //  第三页
        do
        {
            u8g2.setFont(u8g2_font_ncenB14_tr);
            u8g2.drawXBMP(0, 0, 128, 64, error);
        } while (u8g2.nextPage());

        break;
    case 3: //  第四页
        do
        {
            u8g2.setFont(u8g2_font_ncenB14_tr);
            u8g2.drawXBMP(0, 0, 128, 64, dowhat);
        } while (u8g2.nextPage());

        break;
    case 4: //  第四页

        do
        {
            u8g2.setFont(u8g2_font_ncenB14_tr);
            u8g2.drawXBMP(0, 0, 128, 64, love);
        } while (u8g2.nextPage());
        break;
    case 5: //  第四页

        do
        {
            u8g2.setFont(u8g2_font_ncenB14_tr);
            u8g2.drawXBMP(0, 0, 128, 64, sick);
        } while (u8g2.nextPage());
        break;
    case 6: //  第四页
```

```
        do
        {
            u8g2.setFont(u8g2_font_ncenB14_tr);
            u8g2.drawXBMP(0, 0, 128, 64, yun);
        } while (u8g2.nextPage());

        break;
    case 7: // 第四页
        if (WiFi.status() != WL_CONNECTED)
        {
            do
            {
                u8g2.setFont(u8g2_font_ncenB08_tr);
                u8g2.drawXBMP(0, 0, 64, 64, wifi);
                u8g2.drawStr(64, 20, "IP:");
                u8g2.drawStr(64, 40, "192.168.4.1");
                u8g2.drawStr(64, 60, "Need NET");
            } while (u8g2.nextPage());
        }
        else
        {
            fetchWeather();
        }
        break;

        break;
    case 8: // 第四页
        if (WiFi.status() != WL_CONNECTED)
        {
            do
            {
                u8g2.setFont(u8g2_font_ncenB08_tr);
                u8g2.drawXBMP(0, 0, 64, 64, wifi);
                u8g2.drawStr(64, 20, "IP:");
                u8g2.drawStr(64, 40, "192.168.4.1");
                u8g2.drawStr(64, 60, "Need NET");
            } while (u8g2.nextPage());
        }
        else
        {
            do
            {
                timeClient.update(); // 更新时间
                u8g2.setFont(u8g2_font_ncenB14_tr);
                timeClient.update();
                u8g2.drawXBMP(0, 0, 64, 64, timeimage);
                // 获取当前时间
                // 显示时间到 OLED
                int currentHour = timeClient.getHours();
```

```
                int currentMinute = timeClient.getMinutes();
                String timeToDisplay = String(currentHour) + ":" + String(currentMinute);
                u8g2.drawStr(64, 30, "TIME");
                u8g2.setCursor(64, 50);
                u8g2.print(timeToDisplay);

            } while (u8g2.nextPage());
        }
        break;
    default:
        // 添加默认 case 来处理其他情况
        break;
    }
```

（4）自由模式运动逻辑

因为没有为机械狗增加避障等模块，所以自由模式实际上是随机数控制的，在自由模式下机械狗会每隔3S执行actionstate=0到actionstate=9之间的随机actionstate动作。

```
    if (freestate)
     {
        delay(3000);
        actionstate = random(0, 10);
     }
```

（5）按键控制逻辑

```
if (buttonPressed)
    {
        buttonPressed = false; // 清除按键标志
        front();
    }
    if (buttonPressed2)
    {
        buttonPressed2 = false; // 清除按键标志
        back();
    }
```

（6）ADC电量检测

```
    // 对 ADC 数据多次采样并求平均
  float adcVoltage = getAverageAdcVoltage();
```

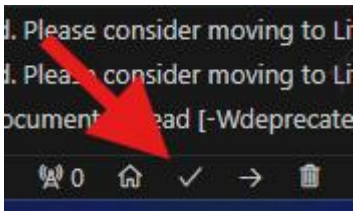
```
    // 将采样的 ADC 电压转换为实际电池电压
batteryVoltage = adcVoltage * voltageDividerRatio; // 计算电池电压

    // 根据电池电压计算电量百分比
batteryPercentage = mapBatteryPercentage(batteryVoltage);
```

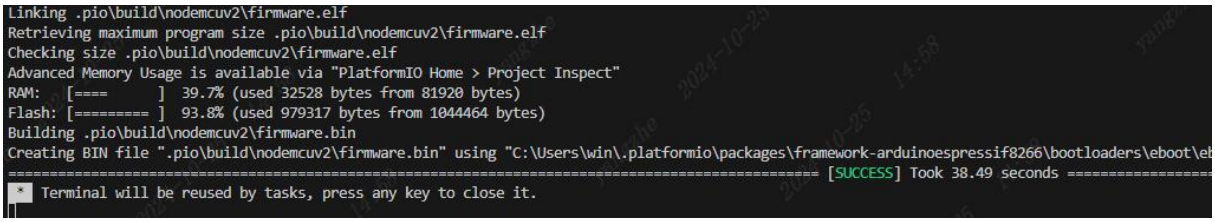
4.7 编译

4.7.1 编译主程序

点击底下一排中的勾，开始构建

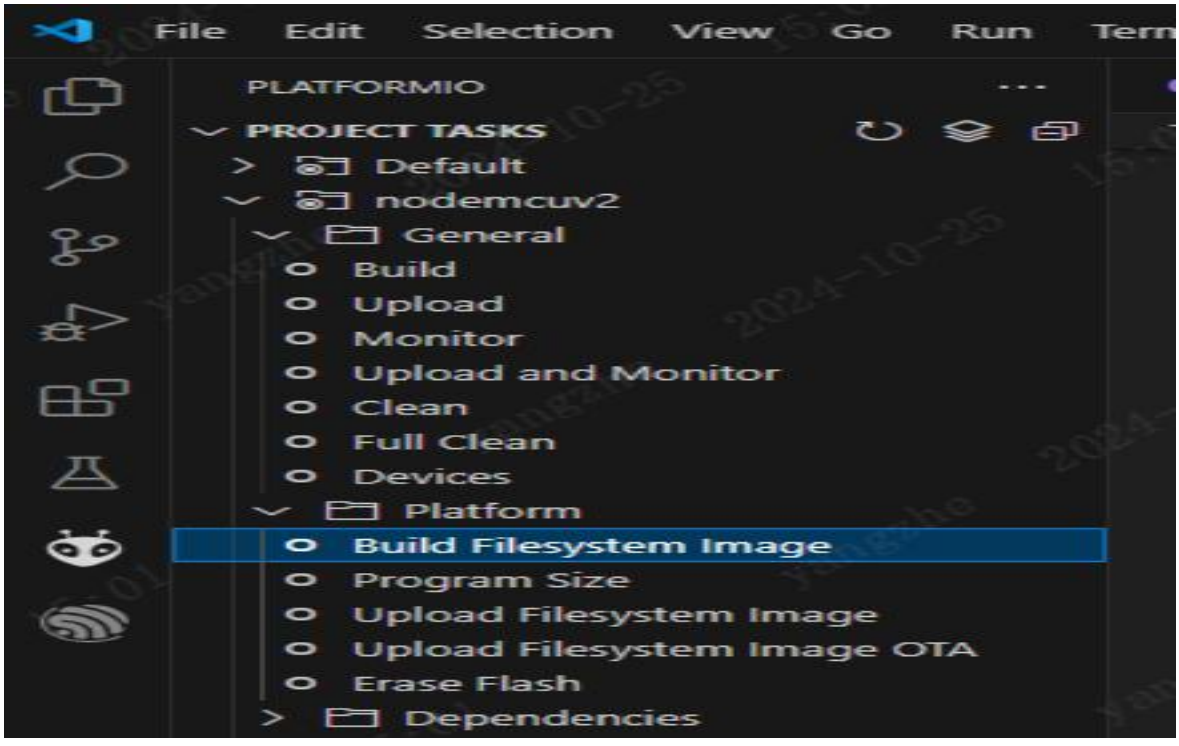


如果显示SUCCESS则表示构建成功

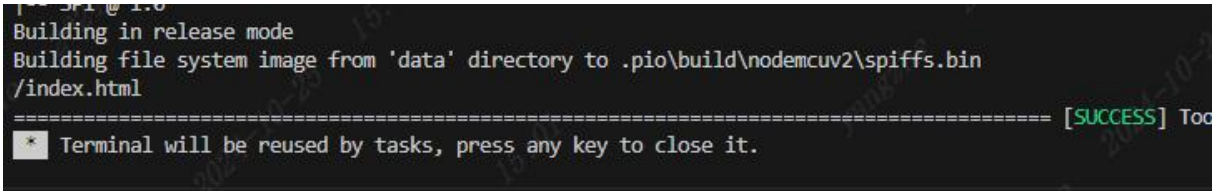


4.7.2 编译FS文件系统

点击左侧PlatformIO的图标，选择Build Filesystem Image



一样，出现SUCCESS则表示构建成功

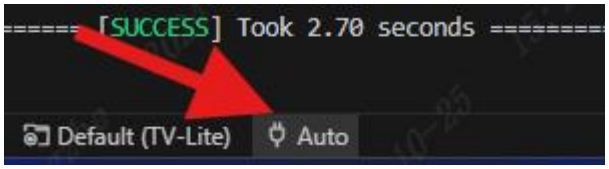


4.8 烧录

4.8.1 接线

在硬件设计中，我们特意为主板设计了下载模式跳线接口，使用跳帽或镊子短接后再上电，即可进入下载模式进行程序下载。然后再将主板上的TX连接到烧录器的RX，主板上的RX连接到烧录器的TX（RX、TX交叉连接）

如果成功进入下载模式，此时在VsCode左下角应该有端口选择按钮，点击选择或使用默认的自动识别都可以。



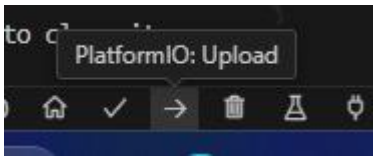
4.8.2 烧录主程序

方法1：使用官方程序

构建成功后，在.pio/build/nodemcuv2文件夹下，应该会有bin文件和elf文件，这都是固件，只是类型不同，您可以通过这些固件使用官方烧录器烧录

方法2：通过编译器

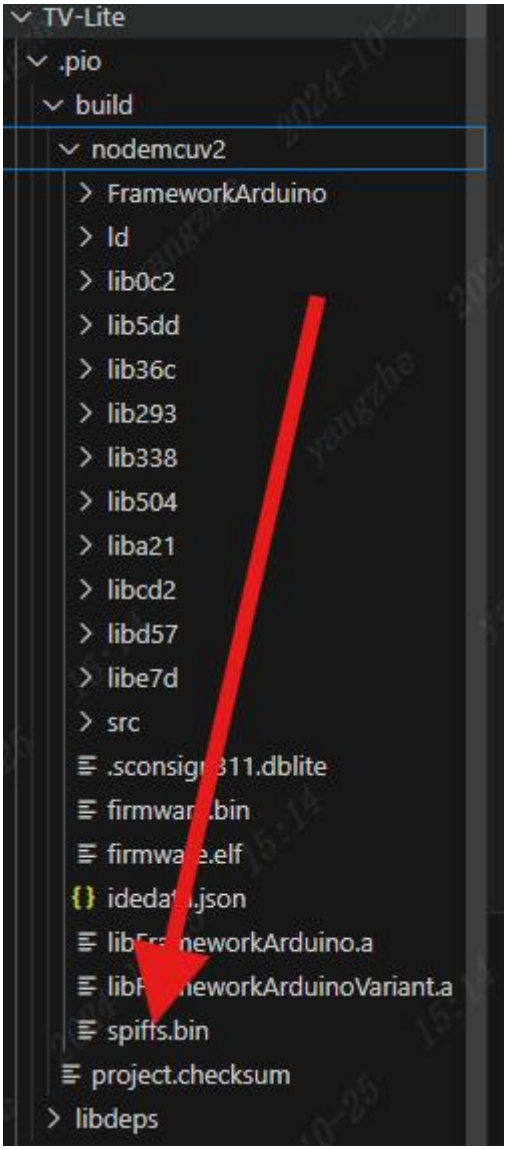
点击底部箭头，等待烧录完成即可



4.8.3 烧录文件系统

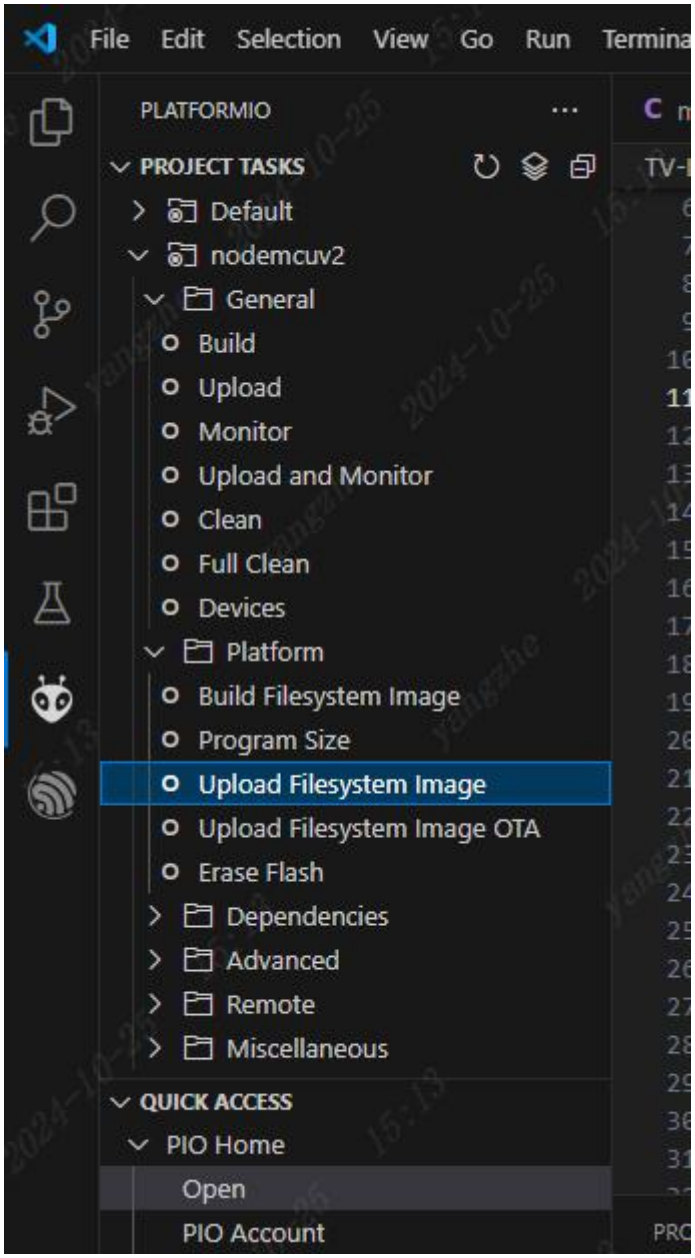
方法1：使用官方程序

文件系统的固件同样位于.pio/build/nodemcuv2文件夹下，先烧录完主程序后再烧录文件系统



方法2：通过编译器

点击左侧的Upload Filesystem Image，等待烧录完成即可


File
Edit
Selection
View
Go
Run
PLATFORMIO
PROJECT TASKS
Default
nodemcuv2
General
Build
Upload
Monitor
Upload and Monitor
Clean
Full Clean
Devices
Platform
Build Filesystem Image
Program Size
Upload Filesystem Image
Upload Filesystem Image OTA
Erase Flash
Dependencies
Advanced
Remote
Miscellaneous
QUICK ACCESS
PIO Home
Open
PIO Account

4.9 舵机校准

刷入的程序是没有进行电机校准的，这会导致电机在运动时出现转动角度不一致等问题。

1. 首先先启动机械狗并连接EDA-Robot热点。
2. 进入浏览器，输入192.168.4.1
3. 进入电机校准页

4.电机底部向左或向右旋转按钮，通过减少和增加电机左右转补偿按钮校准电机

5.记录下认为合理的各个电机补偿值，修改程序的补偿定义，重新刷入程序，当然，不重新输入也可以，这个值是立即生效的。但是不会保存到FS文件系统。

```
35  int engine1 = 14;                    // 舵机引脚
36  int engine1offsetleftpwm = -5;       // 舵机左转补偿
37  int engine1offsetrightpwm = -20;     // 舵机左转补偿-40
38  int engine2 = 16;                    // 舵机引脚
39  int engine2offsetleftpwm = -63;      // 舵机左转补偿
40  int engine2offsetrightpwm = -55;     // 舵机左转补偿-60
41  int engine3 = 12;                    // 舵机引脚
42  int engine3offsetleftpwm = 16;       // 舵机左转补偿
43  int engine3offsetrightpwm = -11;     // 舵机左转补偿
44  int engine4 = 13;                    // 舵机引脚
45  int engine4offsetleftpwm = -56;      // 舵机左转补偿
46  int engine4offsetrightpwm = -60;     // 舵机左转补偿-71
47  int speed = 100; // 舵机转速
```

5 项目资料

5.1 项目资源包

EDA-Robot资源文件

5.2 3D外壳工程

嘉立创云CAD_EDA-Robot外壳工程