电子信息与电气工程学院

**《****单片机系统设计工程项目实践》**

**课程设计报告**

学 院:电子信息与电气工程学院

专业班级: 自动化1班

学生姓名:

学 号:

设计地点：成都大学十教10222

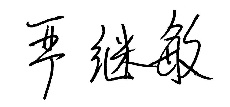
设计题目: 基于stm32的酒店管理系统

完成日期： 2024 年 1月 21 日

指导教师评语：

成绩（五级记分制）：

校内指导教师（签字）: 

校外指导教师（签字）: 

单片机系统设计工程项目实践成绩记录表

（专业认证用表）

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 学生姓名 | |  | 学院 | 电子信息与电气工程学院 | | | 专业 | 自动化 | 学号 |  |
| 设计题目 | |  | | | | | | | | |
| 评价项目 | 最  高  分 | 评 分 | | | | | 具体要求（A级标准参考） | | | |
| A | B | C | D | E |
| 设计目标和方案（20分） | 20 | 20 | 18 | 16 | 14 | ≤12 | 能独立查阅文献，理解选题要求，按照设计需求去理解相关原理中的知识点，寻求可以替代的解决方案。） | | | |
|  |  |  |  |  |
| 可行性论证（10分） | 10 | 10 | 9 | 8 | 7 | ≤6 | 能够基于科学原理并采用科学方法进行方案设计；设计方案可行，实现方法科学；技术路线科学合理。 | | | |
|  |  |  |  |  |  |
| 设计、仿真、实现及调试（10分） | 10 | 10 | 9 | 8 | 7 | ≤6 | 能够设计满足特定需求的系统，使用图纸、仿真结果等形式呈现设计成果。 | | | |
|  |  |  |  |  |
| 优选和创新设计意识（10） | 10 | 10 | 9 | 8 | 7 | ≤6 | 能分析系统的关键环节和参数，能判断参数或方案的优化值或优化途径。 | | | |
|  |  |  |  |  |
| 境保护和社会可持续发展（10） | 10 | 10 | 9 | 8 | 7 | ≤6 | 学生对环境保护和社会可持续发展的含义的理解，并证明设计方案的合理性。 | | | |
|  |  |  |  |  |
| 判断对环境造成的影响（10 | 10 | 10 | 9 | 8 | 7 | ≤6 | 通过查找资料获取相关数据，并说明自己采用的设计方案可能对环境造成的影响。 | | | |
|  |  |  |  |  |
| 正确、规范写技术文档（10） | 10 | 10 | 9 | 8 | 7 | ≤6 | 能够用图纸和设计报告等形式，呈现设计成果；设计报告结构严谨，逻辑性强；课程设计报告遵守撰写规范的具体规定，图表（图纸）、注释清晰明确。 | | | |
|  |  |  |  |  |
| 用英语进行口头、文字的表达和交流（10） | 10 | 10 | 9 | 8 | 7 | ≤6 | 概念明确，思路清晰，清楚地阐述设计的基本观点。回答问题有理有据，基本概念清楚；主要问题回答准确、思路清晰、逻辑严密。 | | | |
|  |  |  |  |  |
| 综合决策和组织管理能力 （10） | 10 | 10 | 9 | 8 | 7 | ≤6 | 积极参与课程设计，完成设计的态度端正，能与小组成员合作开展工作，能胜任团队成员的角色与责任。能综合团队成员意见进行决策。 | | | |
|  |  |  |  |  |
| 总分（百分制） | | |  | |  | | | | | |

电子信息与电气工程学院课程设计任务书

2023 -2024学年第 1 学期 21级自动化 专业 班级 1-4班

课程名称：单片机系统设计工程项目实践

设计题目： 《基于STM32的酒店管理系统》

完成期限：自 2024 年 1月 4 日至 2024 年 1 月 14 日共 2 周

|  |  |  |
| --- | --- | --- |
| 内  容  及  任  务 | 1. 设计的主要技术参数   系统主时钟：72MHz  PWM脉冲宽度：5000 -- 10000  串口波特率：115200  OLED屏通信方式: IIC  DTHT11检测范围：20－90％RH 0－50℃  二、设计任务   1. 实现温湿度数据采集 2. 实现光照数据采集 3. 利用OLED显示温湿度以及光照数据 4. 利用蜂鸣器进行报警 5. 原理图及PCB绘制 6. 利用阿里云平台进行远程控制和可视化显示   三、设计工作量  编写裸机代码初始化串口、DHT11、光敏传感器、OLED相应引脚；通过光敏传感器检测当前房间的光照强度，通过DHT11检测当前房间的温湿度；将检测的光照数据、温湿度数据 通过OLED进行显示，利用蜂鸣器进行报警，绘制原理图及pcb并连接阿里云平台进行远程控制和可视化显示。 | |
| 进  度  安  排 | 起止日期 | 工作内容 |
| 2022.11.19-2022.11.19 | C语言程序复习 |
| 2022.11.19-2022.11.20 | STM32简单GPIO编程，点灯实验 |
| 2022.11.20-2022.11.20 | STM32 DHT11温湿度模块讲解及数据采集 |
| 2022.11.26-2022.11.26 | STM32光照模块讲解及数据采集 |
| 2022.11.26-2022.11.26 | STM32 OLED模块显示温湿度以及光照及蜂鸣器报警 |
| 2022.11.27-2022.12.27 | 利用阿里云平台要进行远程控制和可视化显示 |
| 2022.12.3-2022.12.4 | 调试、答辩 |

|  |  |
| --- | --- |
| 主  要  参  考  资  料 | [1]章小宝.16路舵机控制板使用说明V2.0 [J]，2017.  [2]徐英慧，马忠梅等.ARM9嵌入式系统设计—基于S3C2410与Linux [M].3版.北京.北京航空航大大学出版社，2015.  [3]张仁彦，高正中等.单片机原理及应用 [M].北京:机械工业出版社，2016.  [4][美]史蒂芬.普拉达.C++ Primer Plus[M].6版.北京：人民邮电出版社，2020.  [5][美]布鲁姆，布雷斯纳汉.Linux命令行与shell脚本编程大全[M].北京：人民邮电出版社，2016.  [6]侯殿有.嵌入式系统开发基础—基于ARM9微处理器C语言程序设计[M].5版.北京：清华大学出版社，2019. |

指导教师（签字）：

            2024 年 1月28 日

系（教研室）主任（签字）：

         2024 年 1 月28 日

**基于STM32酒店管理系统**

摘要：本次课程设计的题目是《基于STM32的酒店管理系统》。本课程设计旨在开发一款基于STM32单片机的酒店管理系统，其中主要功能包括温湿度检测和光照强度检测，通过OLED显示屏实时展示监测结果。这一系统不仅提高了对嵌入式系统和STM32应用的理解，还满足了酒店管理中对环境监测的需求。

在项目中，我们选择STM32作为主控芯片，通过与温湿度传感器和光敏电阻等传感器的连接，实现了对酒店房间内环境的实时监测。温湿度传感器负责获取室内的温度和湿度数据，而光敏电阻则用于检测光照强度。这些数据经过STM32的处理后，通过OLED显示屏以直观的方式呈现给用户。

在软件设计方面，我们使用C语言编写了嵌入式程序，利用STM32的ADC（模数转换器）模块对模拟信号进行采样，并通过I2C通信协议与OLED显示屏进行数据传输。通过合理的算法和数据处理，确保了监测数据的准确性和实时性。

通过这个项目，我们不仅深入了解了STM32单片机的编程和硬件连接，还掌握了环境监测系统的设计与实现。这种系统不仅在酒店管理中具有实际应用价值，也为我们提供了一个实际项目中学习和应用嵌入式系统的机会。

该课程设计不仅仅是对硬件和软件知识的应用，更是对解决实际问题的创新思维和团队协作能力的培养。通过这样的实践，我们在嵌入式系统领域积累了宝贵的经验，为未来的职业发展奠定了坚实的基础。

关键词：酒店管理系统，光敏电阻，温湿度检测，OLED显示屏。

**目录**

[1 绪论 7](#_Toc18298)

[1.1课程背景 7](#_Toc19137)

[1.2选题的目的和意义 7](#_Toc30894)

[1.3国内外研究现状 7](#_Toc31758)

[1.4主要的研究内容 8](#_Toc27938)

[2 总体设计 9](#_Toc55)

[3 硬件设计 10-14](#_Toc19512)

[3.1整体设计原理图 10](#_Toc28648)

[3.2dht11 11](#_Toc28648)

[3.3光敏电阻模块 11](#_Toc28648)

[3.4OLED显示屏 12](#_Toc28648)

[3.5供电 13](#_Toc26481)

[3.6蜂鸣器 13](#_Toc11260)

[3.7ESP8266 14](#_Toc11260)

[4 软件设计 15-29](#_Toc9235)

[4.1dht11 15](#_Toc21783)

[4.2光敏电阻模块 16](#_Toc30435)

[4.3OLED显示屏 16-23](#_Toc12522)

[4.4蜂鸣器 23-24](#_Toc12522)

[4.5ESP01S 24-26](#_Toc12522)

[4.6主函数 27-28](#_Toc12522)

[4.7可视化界面和APP 28-29](#_Toc12522)

[5 项目实现与调试 30-33](#_Toc10947)

[5.1程序框图 30](#_Toc12522)

[5.2设备调试 30](#_Toc12522)

[5.3云端调试 31-33](#_Toc12522)

[结语 34](#_Toc29409)

[致谢 35](#_Toc12599)

[参考文献 36](#_Toc19218)

**1 绪论**

**1.1课程背景**

单片机是一种集成了处理器核、存储器、输入/输出设备和定时器等功能于一体的微型计算机系统。学生已经学习并了解了相关的计算机体系结构，编程语言，数字电路和逻辑设计，电子工程基础以及嵌入式系统设计还有通信协议等理论知识，但是缺乏相关方面的实训经验。单片机系统设计工程项目实践便由此诞生，这门课程便旨在于提高学生的相关实践经验，为以后的发展奠定下基础

**1.2选题的目的和意义**

通过该课程设计项目，学生可以将所学的嵌入式系统设计理论付诸实践。这有助于加深对STM32等单片机的理解和应用能力。项目中涉及温湿度传感器和光敏电阻等传感器的应用，学生可以学到如何选择、连接和使用各类传感器。本次课程设计需要在软硬件协同的环境下完成项目，这培养了他们的团队协作能力和解决问题的技能。并且能够积累实际项目经验，加深对嵌入式系统开发的了解，提高自己的综合应用能力，为未来的职业发展奠定基础。

综合而言，本次单片机课程设计为我们提供一个具体的、实践性强的项目，帮助他们更好地理解和应用所学的课程知识，同时培养解决问题、团队合作和沟通能力。

**1.3国内外研究现状**

国内研究现状：

嵌入式系统在酒店管理的应用： 在国内，一些酒店业界和科研机构可能已经开始采用嵌入式系统技术，包括STM32等单片机，用于实现酒店的各项管理功能。这可能包括房间状态监测、智能门锁、温湿度控制等。

物联网技术在酒店领域的研究： 中国的一些物联网研究机构可能在酒店领域进行了一些相关研究，探讨如何通过物联网技术实现酒店管理的智能化和自动化。

国外研究现状：

酒店物联网系统的发展： 在国外，一些发达国家的酒店业可能更早尝试采用物联网技术，包括嵌入式系统，以提高酒店的运营效率和服务质量。这可能包括智能客房控制、能源管理系统等。

单片机在酒店设备中的应用： 在一些先进的国家，可能有关于单片机应用于酒店设备的研究，以实现设备的智能化和互联。

**1.4主要的研究内容**

1. 本次课程设计中我们学习STM32单片机的基本架构、指令集和编程方法，并了解到了STM32在嵌入式系统中的广泛应用，并理解其优势和适用场景。
2. Keil5软件、Proteus仿真软件和stm32cubemx软件的相关运用，以及完成一个简单工程，到熟练掌握并最终完成本次课程设计。
3. Dht11数字温湿度传感器、光敏电阻模块、OLED显示屏以及蜂鸣器。
4. 原理图绘制以及PCB绘制，进行元器件焊接。
5. 代码程序编程，并进行调试。
6. 利用阿里云平台进行远程控制及可视化显示。

**2 总体设计**

根据要用到的模块设计电源电路、引脚接线，然后匹配对应的引脚，如ESP01用到了UART，使用PA9 P10引脚与单片机连接，利用LCEDA完成然后完成电路原理图的设计、器件封装管理、PCB的绘制，发给嘉立创公司打印，打印后焊接。

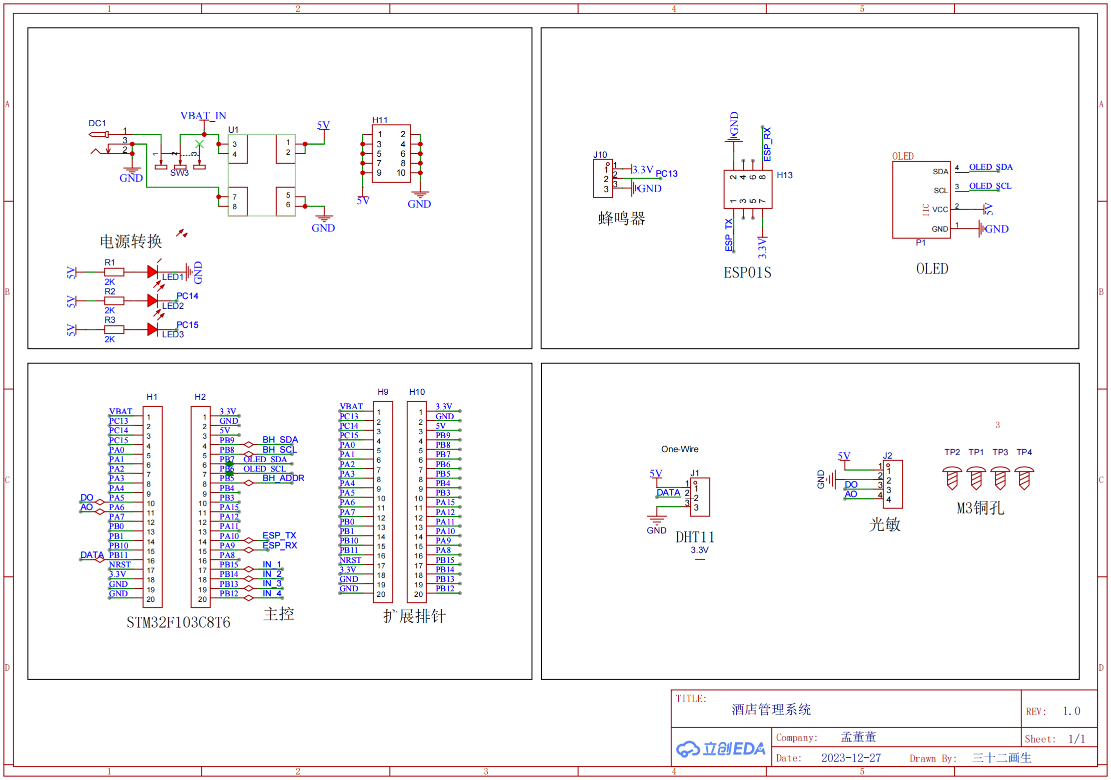
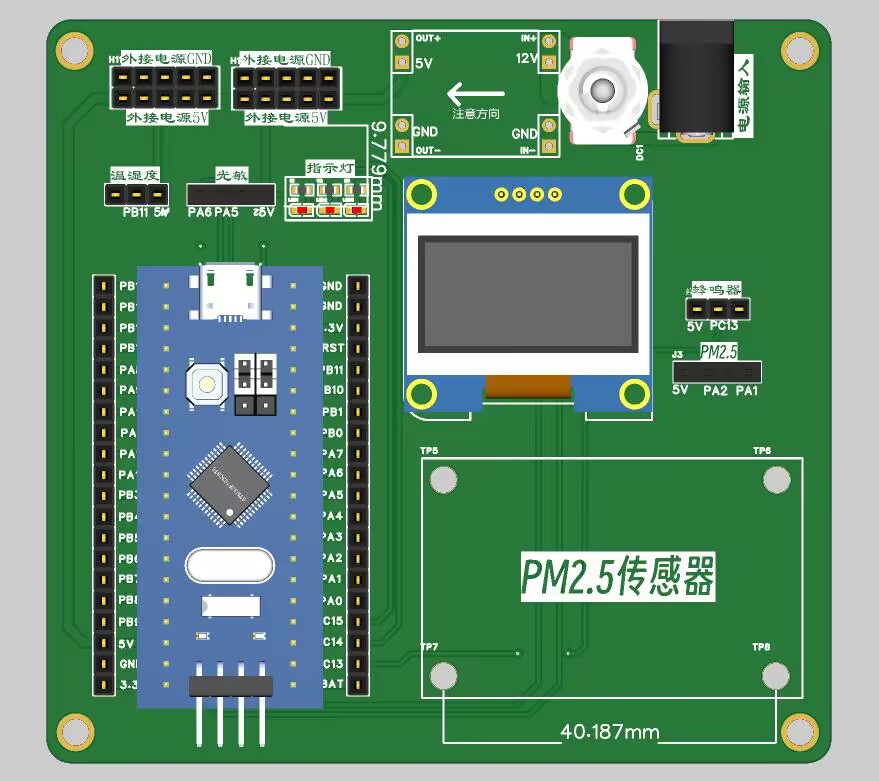
将各个模块焊接后，寻找驱动函数，如ADC的初始化、UART初始化、OLED初始化、OLED如何显示汉字等等。将各个模块的函数库和头文件加入工程，之后再主函数中进行初始化Init调用，调用后进行读取、写入、OLED显示，如温湿度的读取显示。

利用ESP01完成云平台通信和显示功能。首先使用乐鑫公司提供的收录软件，烧录MQTT固件到ESP01模块中，烧录后，在阿里云新建MQTT协议的项目和设备，获得MQTT连接的相关参数，如ClientID，Passward、Url、Username，利用Printf重定向功能向ESP01发送AT指令，连接云平台。云平台设置相关的属性，设备定订阅对应的消息，然后利用Printf完成向云端数据的传输，利用USART接受云端数据，然后进行处理，获取需要的数据，进而控制设备，如打开灯光。

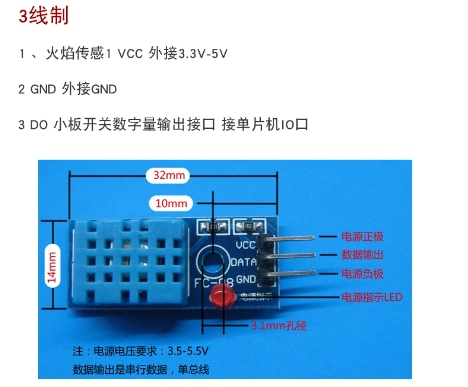
利用云平台提供的Iot-Studio工具完成Web端可视化操作，加入系统提供的组件，绑定数据源，调节大小，放置图标工具，配置数据源，完成显示和控制。同时利用云平台提供的工具，完成小程序、APP端的可视化界面，创建用户，设置账号和密码，使得用户能够登录APP端或者Web端远程和控制设备

**3.硬件设计**

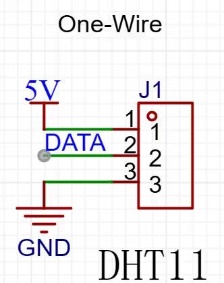
**3.1 整体设计原理图**

****

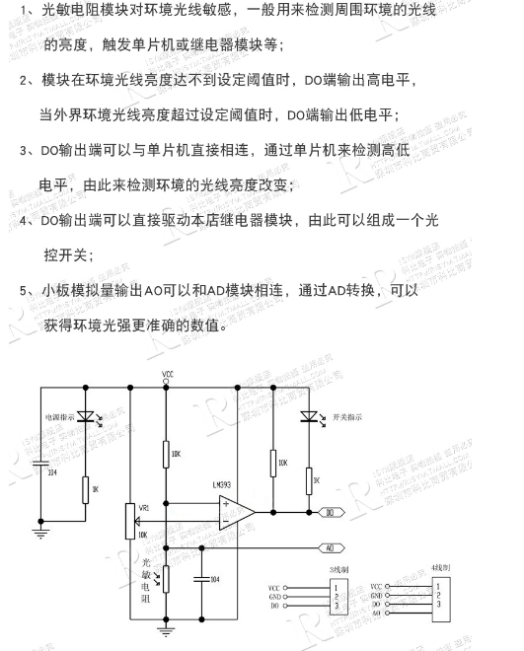
**3.2 dht11**



从左到右引脚是GND DATA VCC 5V供电

DATA PB11 one-wire通信

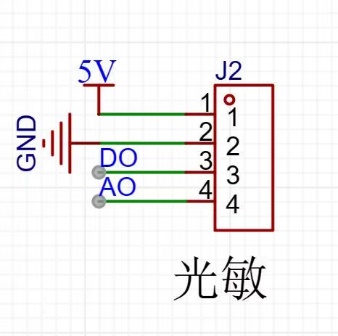
**3.3 光敏电阻模块**



从左到右的引脚为AO DO GND VCC 5V供电

AO PA6 模拟输出口 ADC\_Channel\_6

DO PA5 数字输出口 光照强度超过设定时，输出高电平，反之，输出低电平

ADC\_RegularChannelConfig(ADC1, ADC\_Channel, 1, ADC\_SampleTime\_55Cycles5);

**3.4 OLED显示屏**

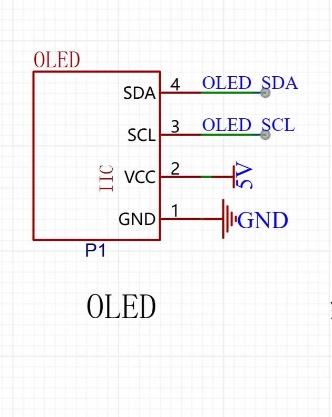


使用从做往右一次是 GND VCC SCL SDA引脚配置的0.96寸OLED

5V供电

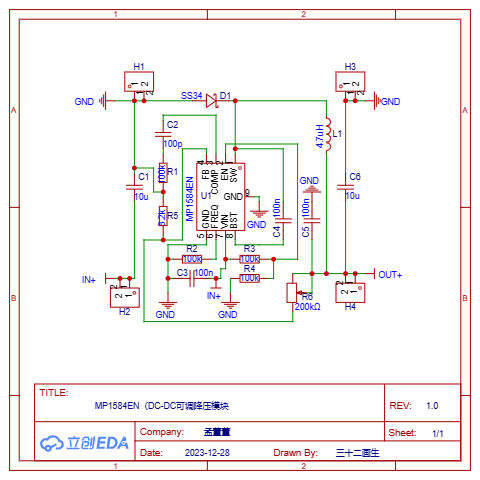
OLED\_SCL -PB6

OLED\_SDA-PB7



### 3.5 供电

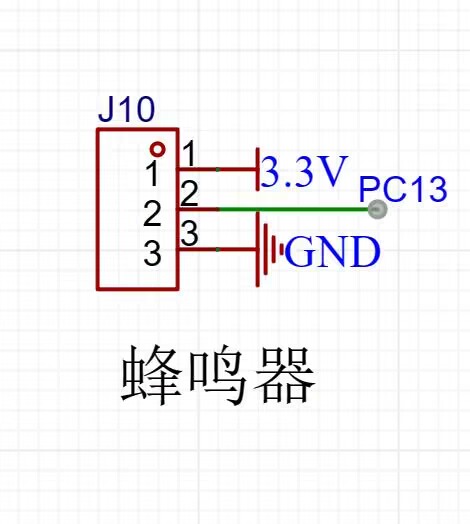
12.6V电池，MP1584EN降压到5V给外设和芯片供电



### 3.6 蜂鸣器

3.3V PC15 GND 不要使用5V供电，否则一上电就会报警





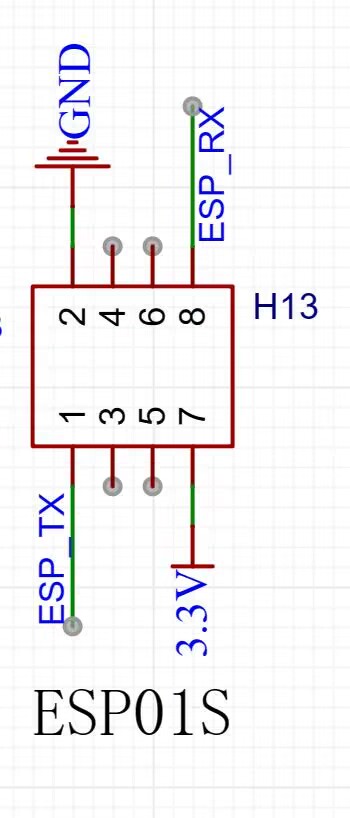
## 3.7 ESP8266



VCC GND RXD TXD //串口要交叉链接，TX接RX

3.3V GND PA9 PA10 //IO0工作时不用接，下载固件时，必须接地。





**4 软件设计**

**4.1dht11**

使用的是PB11引脚来传输数据

//从DHT11读取一个位

//返回值：1/0

u8 DHT11\_Read\_Bit(void)

{

u8 retry=0;

while(DHT11\_DQ\_IN&&retry<100)//等待变为低电平

{

retry++;

Delay\_us(1);

}

retry=0;

while(!DHT11\_DQ\_IN&&retry<100)//等待变高电平

{

retry++;

Delay\_us(1);

}

Delay\_us(40);//等待40us

if(DHT11\_DQ\_IN)return 1;

else return 0;

}

//从DHT11读取一个字节

//返回值：读到的数据

u8 DHT11\_Read\_Byte(void)

{

u8 i,dat;

dat=0;

for (i=0;i<8;i++)

{

dat<<=1;

dat|=DHT11\_Read\_Bit();

}

return dat;

}

u8 DHT11\_Init(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB, ENABLE); //使能PB端口时钟

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_11; //PB11端口配置

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP; //推挽输出

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOB, &GPIO\_InitStructure); //初始化IO口

GPIO\_SetBits(GPIOB,GPIO\_Pin\_11); //PB11 输出高

DHT11\_Rst(); //复位DHT11

return DHT11\_Check();//等待DHT11的回应

}

**4.2光敏电阻模块**

使用GPIOA5 GPIOA6 使用ADC\_CHANNEL6读取光照值，然后转换成0-100的数值，显示出来光照强度

void Lsens\_Init(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA,ENABLE);//使能GPIO时钟

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_6;//PA6 Anolog输入

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AIN; //模拟输入引脚

GPIO\_Init(GPIOA, &GPIO\_InitStructure);

}

//读取Light Sens的值

//0~100:0,最暗;100,最亮

u8 Lsens\_Get\_Val(void)

{

u32 temp\_val=0;

u8 t;

for(t=0;t<LSENS\_READ\_TIMES;t++) //取平均值 10次

{

temp\_val+=AD\_GetValue(LSENS\_ADC\_CHX); //读取ADC值 最大为4096

Delay\_ms(5);

}

temp\_val/=LSENS\_READ\_TIMES; //得到平均值

if(temp\_val>4000)temp\_val=4000; //限幅

return (u8)(100-(temp\_val/40);

}

**4.3OLED显示屏**

使用的是PB5--SCL PB6--SDA两个引脚

/\*引脚配置\*/

#define OLED\_W\_SCL(x) GPIO\_WriteBit(GPIOB, GPIO\_Pin\_6, (BitAction)(x))

#define OLED\_W\_SDA(x) GPIO\_WriteBit(GPIOB, GPIO\_Pin\_7, (BitAction)(x))

u8 OLED\_GRAM[128][8];

/\*引脚初始化\*/

void OLED\_I2C\_Init(void)

{

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOB, ENABLE);

GPIO\_InitTypeDef GPIO\_InitStructure;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_OD;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_6;

GPIO\_Init(GPIOB, &GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_7;

GPIO\_Init(GPIOB, &GPIO\_InitStructure);

OLED\_W\_SCL(1);

OLED\_W\_SDA(1);

}

/\*\*

\* @brief I2C开始

\* @param 无

\* @retval 无

\*/

void OLED\_I2C\_Start(void)

{

OLED\_W\_SDA(1);

OLED\_W\_SCL(1);

OLED\_W\_SDA(0);

OLED\_W\_SCL(0);

}

/\*\*

\* @brief I2C停止

\* @param 无

\* @retval 无

\*/

void OLED\_I2C\_Stop(void)

{

OLED\_W\_SDA(0);

OLED\_W\_SCL(1);

OLED\_W\_SDA(1);

}

/\*\*

\* @brief I2C发送一个字节

\* @param Byte 要发送的一个字节

\* @retval 无

\*/

void OLED\_I2C\_SendByte(uint8\_t Byte)

{

uint8\_t i;

for (i = 0; i < 8; i++)

{

OLED\_W\_SDA(Byte & (0x80 >> i));

OLED\_W\_SCL(1);

OLED\_W\_SCL(0);

}

OLED\_W\_SCL(1); //额外的一个时钟，不处理应答信号

OLED\_W\_SCL(0);

}

/\*\*

\* @brief OLED写命令

\* @param Command 要写入的命令

\* @retval 无

\*/

void OLED\_WriteCommand(uint8\_t Command)

{

OLED\_I2C\_Start();

OLED\_I2C\_SendByte(0x78); //从机地址

OLED\_I2C\_SendByte(0x00); //写命令

OLED\_I2C\_SendByte(Command);

OLED\_I2C\_Stop();

}

/\*\*

\* @brief OLED写数据

\* @param Data 要写入的数据

\* @retval 无

\*/

void OLED\_WriteData(uint8\_t Data)

{

OLED\_I2C\_Start();

OLED\_I2C\_SendByte(0x78); //从机地址

OLED\_I2C\_SendByte(0x40); //写数据

OLED\_I2C\_SendByte(Data);

OLED\_I2C\_Stop();

}

/\*\*

\* @brief OLED设置光标位置

/\*\*

\* @brief OLED清屏

\* @param 无

\* @retval 无

\*/

void OLED\_Clear(void)

{

uint8\_t i, j;

for (j = 0; j < 8; j++)

{

OLED\_SetCursor(j, 0);

for(i = 0; i < 128; i++)

{

OLED\_WriteData(0x00);

}

}

}

/\*\*

\* @brief OLED显示一个字符

\* @param Line 行位置，范围：1~4

\* @param Column 列位置，范围：1~16

\* @param Char 要显示的一个字符，范围：ASCII可见字符

\* @retval 无

\*/

void OLED\_ShowChar(uint8\_t Line, uint8\_t Column, char Char)

{

uint8\_t i;

OLED\_SetCursor((Line - 1) \* 2, (Column - 1) \* 8); //设置光标位置在上半部分

for (i = 0; i < 8; i++)

{

OLED\_WriteData(OLED\_F8x16[Char - ' '][i]); //显示上半部分内容

}

OLED\_SetCursor((Line - 1) \* 2 + 1, (Column - 1) \* 8); //设置光标位置在下半部分

for (i = 0; i < 8; i++)

{

OLED\_WriteData(OLED\_F8x16[Char - ' '][i + 8]); //显示下半部分内容

}

}

/\*\*

\* @brief OLED显示字符串

\* @param Line 起始行位置，范围：1~4

\* @param Column 起始列位置，范围：1~16

\* @param String 要显示的字符串，范围：ASCII可见字符

\* @retval 无

\*/

void OLED\_ShowString(uint8\_t Line, uint8\_t Column, char \*String)

{

uint8\_t i;

for (i = 0; String[i] != '\0'; i++)

{

OLED\_ShowChar(Line, Column + i, String[i]);

}

}

/\*\*

\* @brief OLED显示数字（十进制，正数）

\* @param Line 起始行位置，范围：1~4

\* @param Column 起始列位置，范围：1~16

\* @param Number 要显示的数字，范围：0~4294967295

\* @param Length 要显示数字的长度，范围：1~10

\* @retval 无

\*/

void OLED\_ShowNum(uint8\_t Line, uint8\_t Column, uint32\_t Number, uint8\_t Length)

{

uint8\_t i;

for (i = 0; i < Length; i++)

{

OLED\_ShowChar(Line, Column + i, Number / OLED\_Pow(10, Length - i - 1) % 10 + '0');

}

}

/\*\*

\* @brief OLED显示数字（十进制，带符号数）

\* @param Line 起始行位置，范围：1~4

\* @param Column 起始列位置，范围：1~16

\* @param Number 要显示的数字，范围：-2147483648~2147483647

\* @param Length 要显示数字的长度，范围：1~10

\* @retval 无

\*/

void OLED\_ShowSignedNum(uint8\_t Line, uint8\_t Column, int32\_t Number, uint8\_t Length)

{

uint8\_t i;

uint32\_t Number1;

if (Number >= 0)

{

OLED\_ShowChar(Line, Column, '+');

Number1 = Number;

}

else

{

OLED\_ShowChar(Line, Column, '-');

Number1 = -Number;

}

for (i = 0; i < Length; i++)

{

OLED\_ShowChar(Line, Column + i + 1, Number1 / OLED\_Pow(10, Length - i - 1) % 10 + '0');

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* 函数名 OLED\_ShowChinese

\* 功能 显示汉字

\* 参数Line 设置显示字符所在的行 1~4

\* 参数Column 设置显示字符所在的列 1~8

\* 参数num 设置显示字符在点阵中的序号

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void OLED\_ShowChinese(uint8\_t Line,uint8\_t Column,uint8\_t num)

{

uint8\_t i;

OLED\_SetCursor((Line - 1) \* 2, (Column - 1) \* 16);

for(i = 0;i<16;i++)

{

OLED\_WriteData(Font\_Chinese[2\*num][i]);

}

OLED\_SetCursor((Line - 1) \* 2 + 1, (Column - 1) \* 16);

for(i = 0;i<16;i++)

{

OLED\_WriteData(Font\_Chinese[2\*num+1][i]);

}

}

void OLED\_ShowApple(uint8\_t Line,uint8\_t Column,uint8\_t num)

{

uint8\_t i;

OLED\_SetCursor((Line - 1) \* 2, (Column - 1) \* 16);

for(i = 0;i<16;i++)

{

OLED\_WriteData(Font\_Apple[2\*num][i]);

}

OLED\_SetCursor((Line - 1) \* 2 + 1, (Column - 1) \* 16);

for(i = 0;i<16;i++)

{

OLED\_WriteData(Font\_Apple[2\*num+1][i]);

}

}

/\*\*

\* @brief OLED初始化

\* @param 无

\* @retval 无

\*/

void OLED\_Init(void)

{

uint32\_t i, j;

for (i = 0; i < 1000; i++) //上电延时

{

for (j = 0; j < 1000; j++);

}

OLED\_I2C\_Init(); //端口初始化

OLED\_WriteCommand(0xAE); //关闭显示

OLED\_WriteCommand(0xD5); //设置显示时钟分频比/振荡器频率

OLED\_WriteCommand(0x80);

OLED\_WriteCommand(0xA8); //设置多路复用率

OLED\_WriteCommand(0x3F);

OLED\_WriteCommand(0xD3); //设置显示偏移

OLED\_WriteCommand(0x00);

OLED\_WriteCommand(0x40); //设置显示开始行

OLED\_WriteCommand(0xA1); //设置左右方向，0xA1正常 0xA0左右反置

OLED\_WriteCommand(0xC8); //设置上下方向，0xC8正常 0xC0上下反置

OLED\_WriteCommand(0xDA); //设置COM引脚硬件配置

OLED\_WriteCommand(0x12);

OLED\_WriteCommand(0x81); //设置对比度控制

OLED\_WriteCommand(0xCF);

OLED\_WriteCommand(0xD9); //设置预充电周期

OLED\_WriteCommand(0xF1);

OLED\_WriteCommand(0xDB); //设置VCOMH取消选择级别

OLED\_WriteCommand(0x30);

OLED\_WriteCommand(0xA4); //设置整个显示打开/关闭

OLED\_WriteCommand(0xA6); //设置正常/倒转显示

OLED\_WriteCommand(0x8D); //设置充电泵

OLED\_WriteCommand(0x14);

OLED\_WriteCommand(0xAF); //开启显示

OLED\_Clear(); //OLED清屏

}

void OLED\_MyShowStart(void)

{

//OLED\_ShowString(2,1,"\*\*");

OLED\_ShowChinese(2,2,21);//酒

OLED\_ShowChinese(2,3,22);//店

OLED\_ShowChinese(2,4,23);//管

OLED\_ShowChinese(2,5,24);//理

OLED\_ShowChinese(2,6,14);//系

OLED\_ShowChinese(2,7,15);//统

}

void OLED\_Apple(void)

{

int i=2;

OLED\_ShowApple(1,1+i,1);//

OLED\_ShowApple(1,2+i,2);//

OLED\_ShowApple(1,3+i,3);//

OLED\_ShowApple(1,4+i,4);//

OLED\_ShowApple(2,1+i,5);//

OLED\_ShowApple(2,2+i,6);//

OLED\_ShowApple(2,3+i,7);//

OLED\_ShowApple(2,4+i,8);//

OLED\_ShowApple(3,1+i,9);//

OLED\_ShowApple(3,2+i,10);//

OLED\_ShowApple(3,3+i,11);//

OLED\_ShowApple(3,4+i,12);//

OLED\_ShowApple(4,1+i,13);//

OLED\_ShowApple(4,2+i,14);//

OLED\_ShowApple(4,3+i,15);//

OLED\_ShowApple(4,4+i,16);//

}

**4.4蜂鸣器**

RCC\_APB2PeriphClockCmd( RCC\_APB2Periph\_GPIOC , ENABLE); //GPIO C

//蜂鸣器 PC13

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_13;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP; //推挽输出

GPIO\_Init(GPIOC,&GPIO\_InitStructure);

GPIO\_SetBits(GPIOC,GPIO\_Pin\_13);

报警功能实现

void Alarm(void)

{

if(Light>80)

{

AlarmSwitch=1;

GPIO\_ResetBits(GPIOC,GPIO\_Pin\_13);

}

else

{

AlarmSwitch=0;

GPIO\_SetBits(GPIOC,GPIO\_Pin\_13);

}

}

**4.5 ESP01S**

char RECS[250];

unsigned char i;

void MyUSART\_Init(void)

{

NVIC\_PriorityGroupConfig(NVIC\_PriorityGroup\_2);

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_USART1 ,ENABLE);//1.开启时钟

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_AFIO ,ENABLE);

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA ,ENABLE);

GPIO\_InitTypeDef GPIO\_InitStructure; //初始化GIPIO

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_9;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOA,&GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_IPU;

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_10;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_Init(GPIOA,&GPIO\_InitStructure);

USART\_InitTypeDef USART\_InitStructure; //初始化串口

USART\_InitStructure.USART\_BaudRate = 115200;

USART\_InitStructure.USART\_HardwareFlowControl =USART\_HardwareFlowControl\_None;

USART\_InitStructure.USART\_Mode = USART\_Mode\_Rx | USART\_Mode\_Tx;

USART\_InitStructure.USART\_Parity =USART\_Parity\_No ;

USART\_InitStructure.USART\_StopBits = USART\_StopBits\_1;

USART\_InitStructure.USART\_WordLength = USART\_WordLength\_8b;

USART\_Init (USART1,&USART\_InitStructure);

NVIC\_InitTypeDef NVIC\_InitStructure;

NVIC\_InitStructure.NVIC\_IRQChannel =USART1\_IRQn;

NVIC\_InitStructure.NVIC\_IRQChannelCmd = ENABLE;

NVIC\_InitStructure.NVIC\_IRQChannelPreemptionPriority = 1;

NVIC\_InitStructure.NVIC\_IRQChannelSubPriority = 0;

NVIC\_Init(&NVIC\_InitStructure);

USART\_ITConfig(USART1,USART\_IT\_RXNE,ENABLE);

USART\_Cmd(USART1,ENABLE);

}

char\* MyUSART\_GetString(void)

{

return RECS;

}

void MyUSART\_SendString(char\* str)

{

uint8\_t stri=0;

while(str[stri] != '\0')

USART\_SendData (USART1,str[stri++]);

}

void USART1\_IRQHandler()

{

if(USART\_GetITStatus(USART1,USART\_IT\_RXNE))

{

RECS[i++]=USART\_ReceiveData(USART1);

if((RECS[i-2]=='\r')|(RECS[i-1]=='\n'))

{

RECS[i-2]='\0';

i = 0;

CommandAnalyse();

}

}

}

char esp\_Init(void)

{

memset(RECS,0,sizeof(RECS));

printf("AT+RST\r\n"); //重启

Delay\_ms(2000);

memset(RECS,0,sizeof(RECS));

printf("ATE0\r\n"); //关闭回显

Delay\_ms(10);

if(strcmp(RECS,"OK"))

return 1;

printf("AT+CWMODE=1\r\n"); //Station模式

Delay\_ms(1000);

if(strcmp(RECS,"OK"))

return 2;

memset(RECS,0,sizeof(RECS));

printf("AT+CWJAP=\"%s\",\"%s\"\r\n",WIFI,WIFIASSWORD); //连接热点

Delay\_ms(2000);

if(strcmp(RECS,"OK"))

return 3;

memset(RECS,0,sizeof(RECS));

printf("AT+MQTTUSERCFG=0,1,\"%s\",\"%s\",\"%s\",0,0,\"\"\r\n",ClintID,username,passwd);//用户信息配置

Delay\_ms(10);

if(strcmp(RECS,"OK"))

return 4;

memset(RECS,0,sizeof(RECS));

printf("AT+MQTTCONN=0,\"%s\",1883,1\r\n",Url); //连接服务器

Delay\_ms(1000);

if(strcmp(RECS,"OK"))

return 5;

printf("AT+MQTTSUB=0,\"%s\",1\r\n",subtopic); //订阅消息

Delay\_ms(500);

if(strcmp(RECS,"OK"))

return 5;

memset(RECS,0,sizeof(RECS));

return 0;

}

**4.6 主函数**

使用了阿里云的云平台，这里有一个宏定义，进行选择

YUN\_CONNECT 1 //1表示运行云链接，0表示不运行云链接

#include "led.h"

#include "esp.h"

#include "OLED.H"

#include "light.h"

#include "Delay.h"

#include <stdio.h>

#include "dht11.H"

#include "MyUSART.H"

#include "stm32f10x.h"

#include "AD.h"

#define YUN\_CONNECT 1 //1表示运行云链接，0表示不运行云链接

u8 Temperature;

u8 Shidu;

u8 Light;

u8 LED1;

u8 LED2;

u8 AlarmSwitch;

u8 Light;

u8 Judge=0;

u8 Light;

void Yun\_LED(void);

extern char RECS[200];

void Alarm(void);

void Init(void)

{

OLED\_Init();

AD\_Init();

Lsens\_Init();

LED\_GPIO\_Config();

OLED\_Apple();

Delay\_ms(1000);

OLED\_Clear();

OLED\_LiuShowStart();//酒店管理系统

Delay\_ms(1000);

OLED\_Clear();

#if YUN\_CONNECT

OLED\_ShowString(1,1," Linking...");

MyUSART\_Init(); //初始化串口

LED\_GPIO\_Config();

do

{

Judge = esp\_Init();

OLED\_ShowString(1,1,"error code: ");

OLED\_ShowNum(2,1,Judge,1);

}while(Judge); //连接阿里云直到成功

OLED\_Clear();

OLED\_ShowString(3,1," YUN IS OK");

Delay\_ms(500);

#endif

OLED\_Clear();

}

int main(void)

{

Init();

while(1)

{

cnt++;

DHT11\_Read\_Data( &Temperature, &Shidu); //读取温湿度值

Delay\_ms(1000);

Light=Lsens\_Get\_Val();

pm2();

Alarm();

//显示温度----------------------------------------------------

OLED\_ShowChinese(1,1,0);//温

OLED\_ShowChinese(1,2,1);//度

OLED\_ShowString(1,5,":");

OLED\_ShowNum(1,6,Temperature, 2);

OLED\_ShowChinese(1,5,6);//℃

//显示湿度----------------------------------------------------

OLED\_ShowChinese(2,1,2);//湿

OLED\_ShowChinese(2,2,1);//度

OLED\_ShowString(2,5,":");

OLED\_ShowNum(2, 6, Shidu, 2);

OLED\_ShowString(2,8,"%RH");

//显示光照----------------------------------------------------

OLED\_ShowChinese(3,1,3);//光

OLED\_ShowChinese(3,2,4);//照

OLED\_ShowString(3, 5, ":");

OLED\_ShowNum(3,6,Light,2);

OLED\_ShowString(3, 8, "%");

}

}

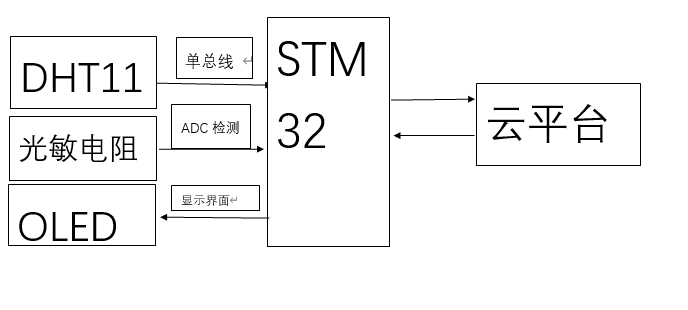
**4.7可视化界面和APP**

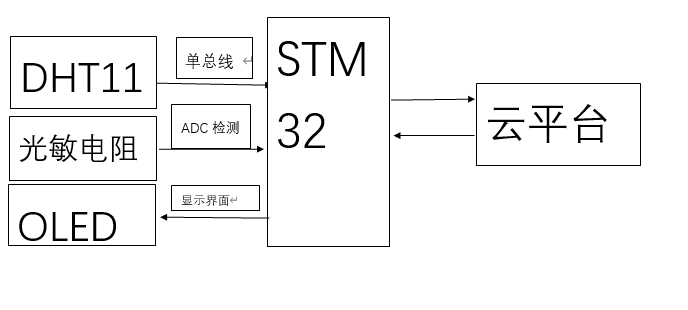
我利用网络资源自行制作了网页可视化界面可以实时反馈和读取数据，对设备进行远程控制。

我使用了两个云平台，一个是阿里云平台，一个是ThingsCloud，效果非常好看，而且控制迅速

**5 项目实现与调试**

**5.1程序框图**

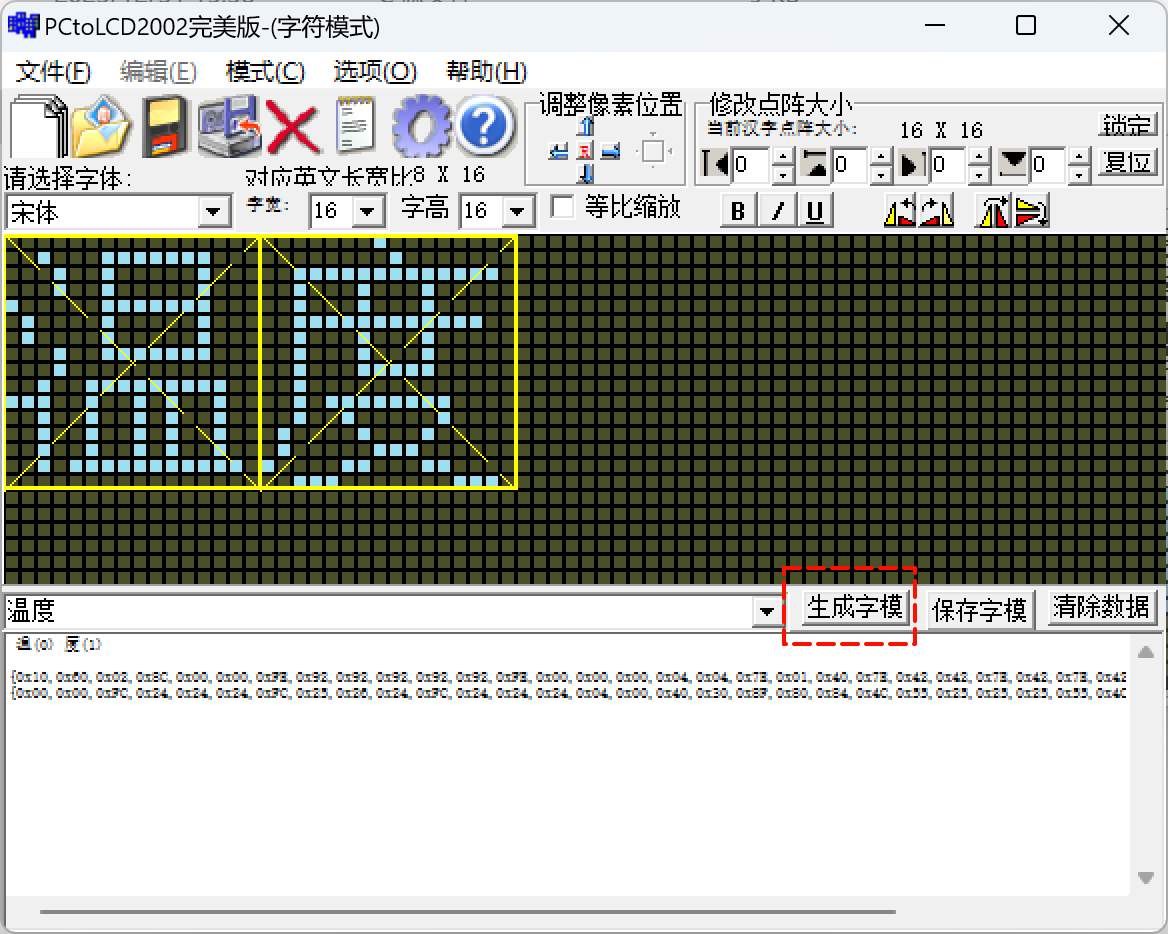




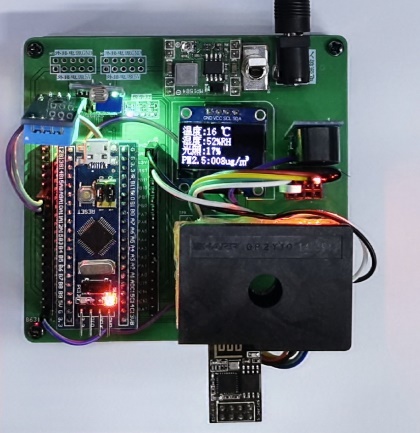
**5.2 设备调试**

首先，我想显示汉字，于是我利用取模软件取模汉字显示，在OLED屏幕上显示出来。步骤如下

按照此图片设置，切记选择C51格式，生成字模后，复制一个汉字，是含有32个元素的数组



取到字模后，写代码调试功能，显示在OLED屏幕上，到这里，设备端的调试就结束了



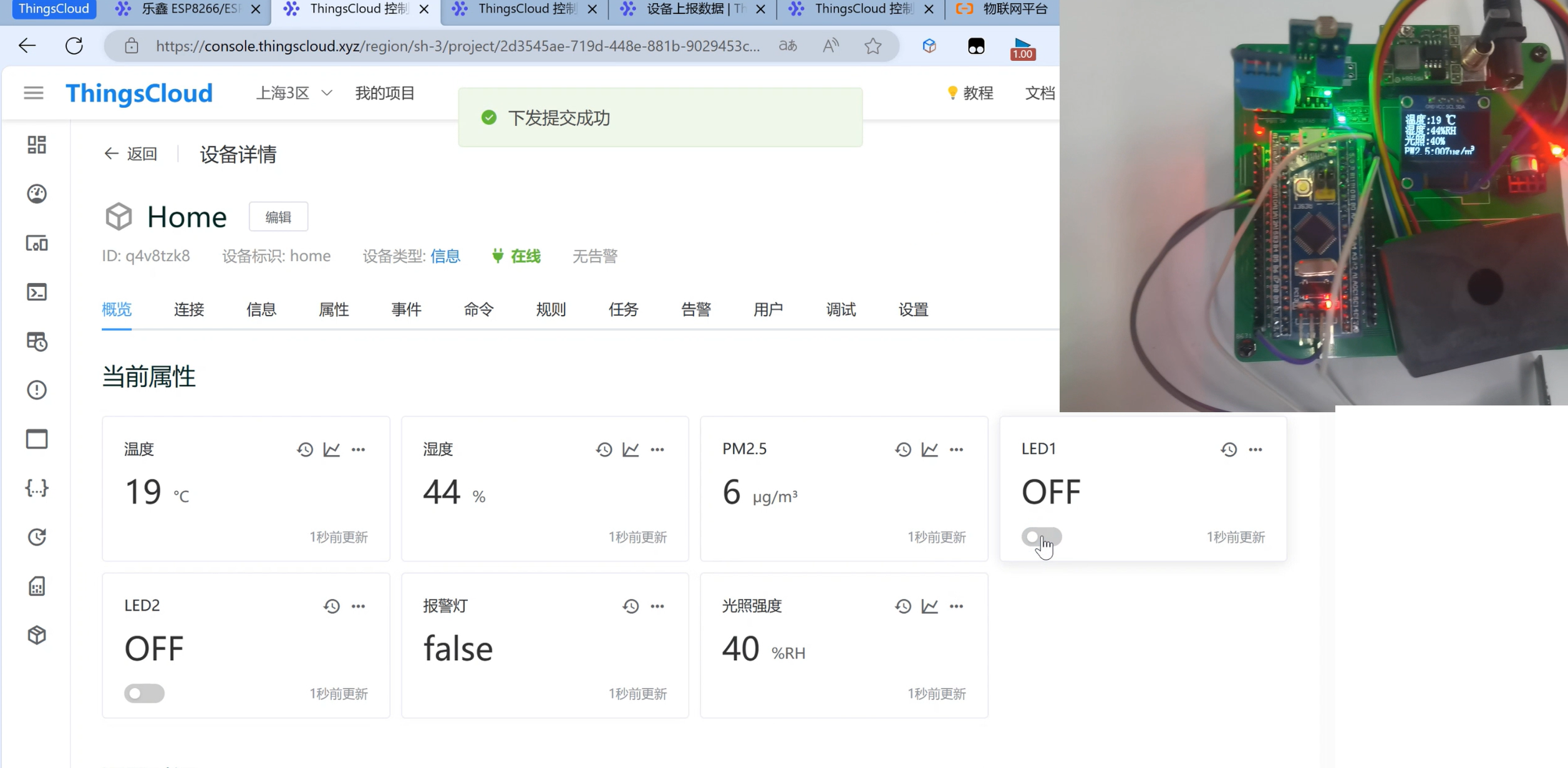
**5.3 云端调试**

首先对设备新建属性，用来接收设备上传的数据，如下图。其中温度、湿度、PM2.5、光照强度全部是Number类型，LED1 LED2 报警灯全部是布尔类型



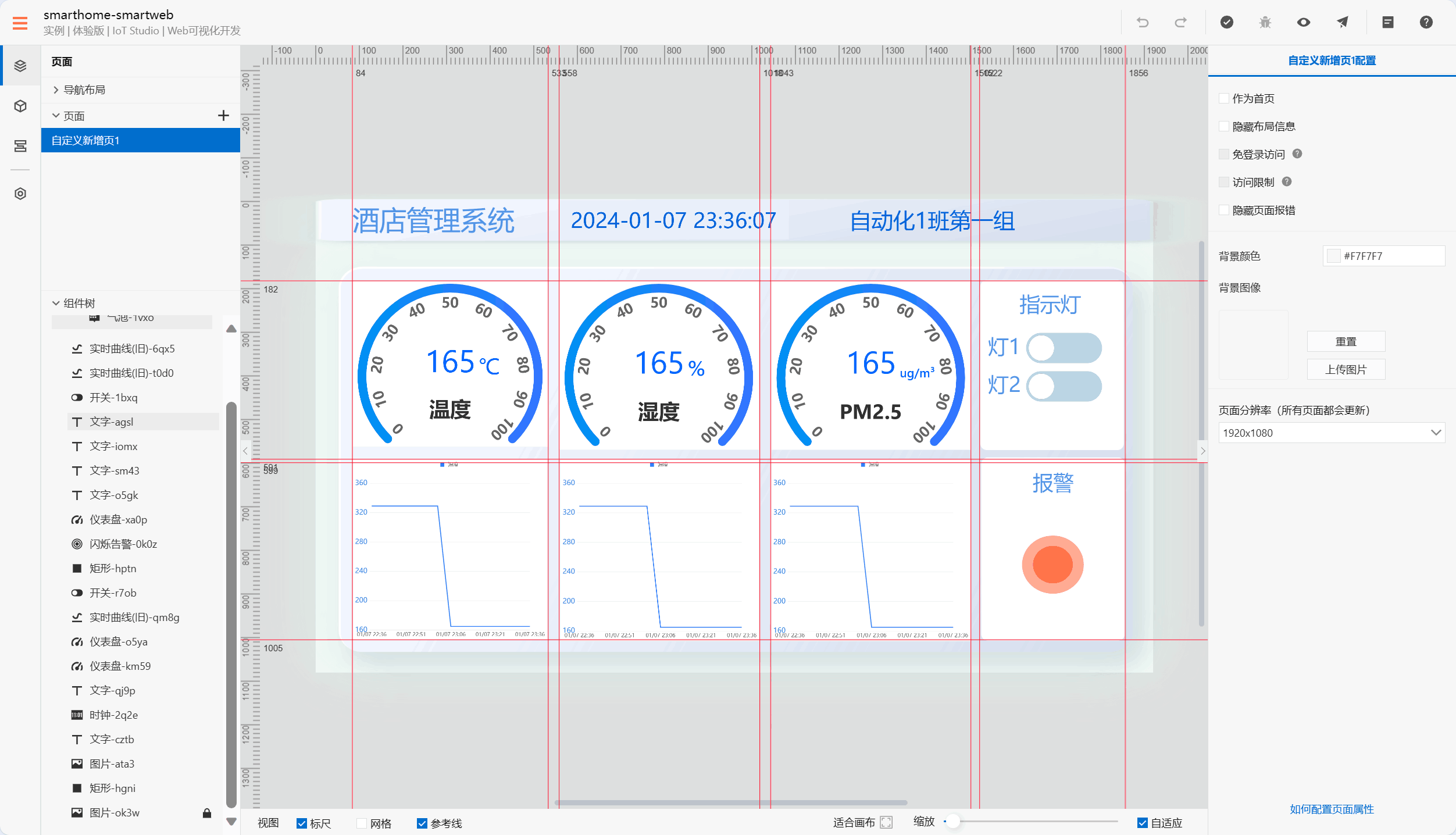
建立完属性后，上传到云端数据，对数据进行显示和通信。下发数据和上传数据如图所示。通过串口解析此字符串进行设备控制，如下是控制远程开关灯和数据读取

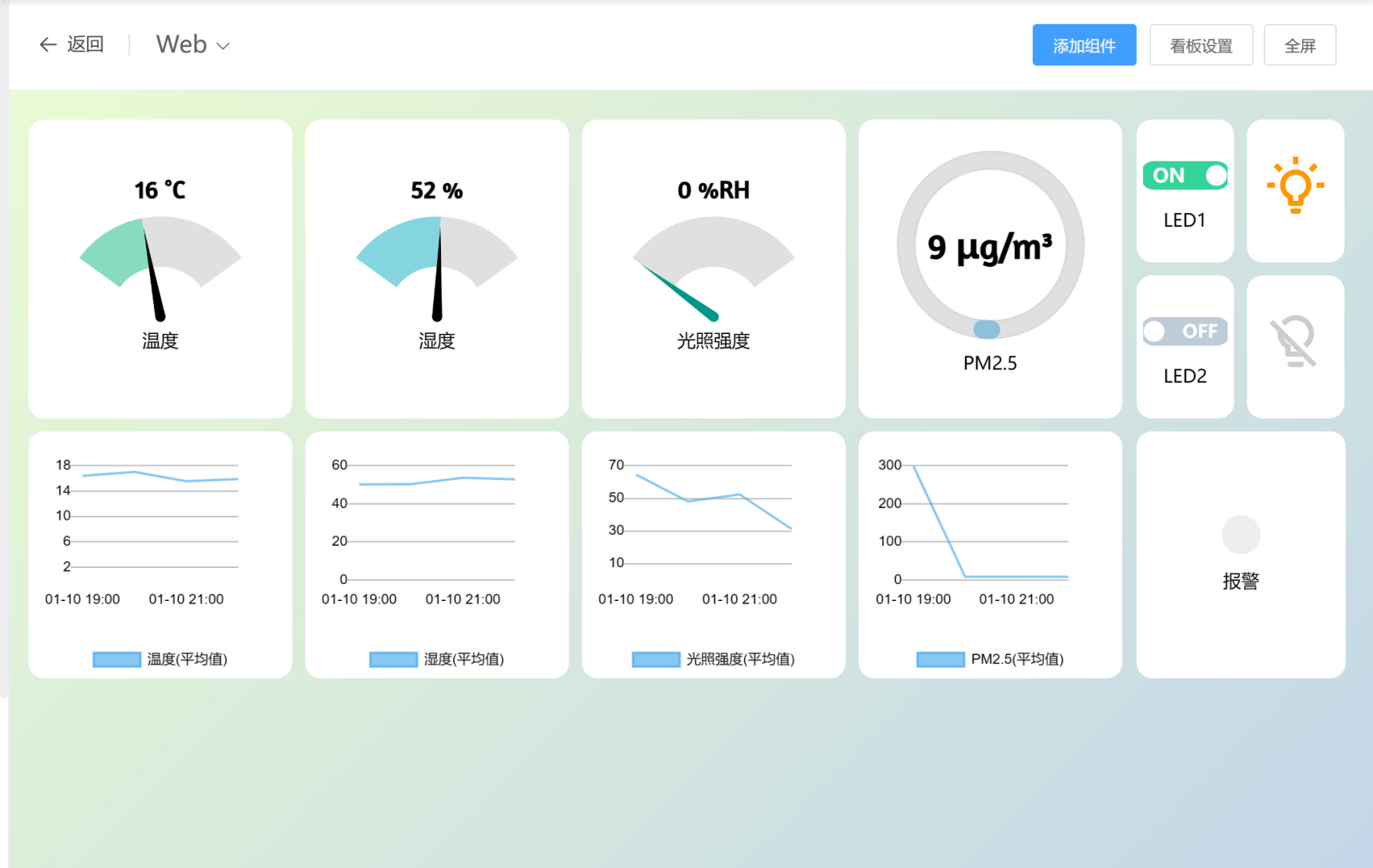




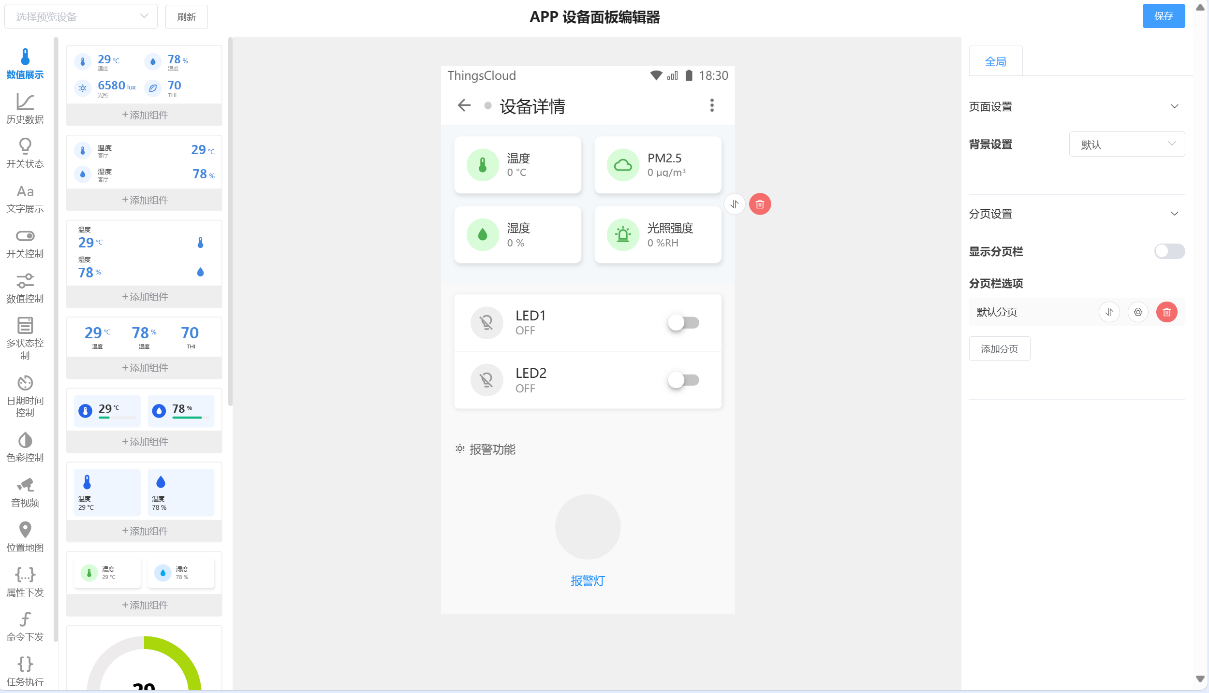
下面是可视化设计

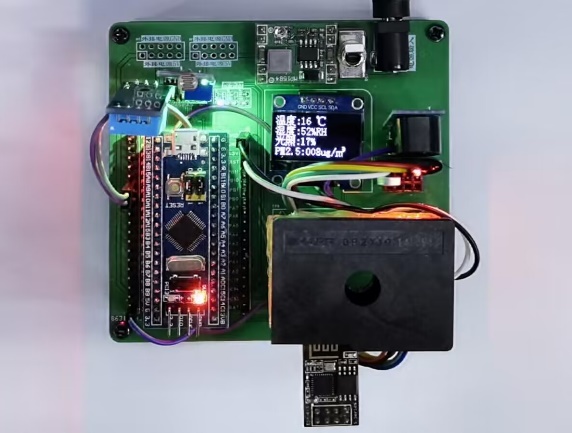
每个属性都可以设置对应的数据面板显示，具体显示步骤如下





最后是APP端显示与实物演示





**结语**

两周单片机课程设计帮助我们建立了嵌入式系统和单片机编程的基础知识。我们学会了如何理解和使用不同类型的单片机，以及它们的架构和工作原理。通过实际的项目和实践，我们提高了单片机编程的技能。我们学到了如何使用C语言等编程语言来编写有效且可靠的嵌入式代码。这包括处理输入和输出、中断服务例程、时钟管理等方面的技能。单片机课程设计让我们对硬件设计和连接有了更深入的理解。我们学到了如何选择适当的传感器、执行器和其他外设，并将它们与单片机连接以构建完整的系统。在实际项目中，我们经常面临各种问题，如错误的连接、编程错误等。通过不断的调试和解决问题，我们学到了如何有效地诊断和修复单片机系统中的错误。在某些项目中，我们有机会与同学合作，这加强了我们的团队合作和沟通能力。共同努力解决问题和完成项目让我们认识到团队协作在嵌入式系统开发中的重要性。单片机课程设计不仅仅是理论知识的学习，更是通过实际项目获得的宝贵经验。这种实践经验对于将来从事嵌入式系统开发或相关领域的工作非常有帮助。单片机领域的技术不断更新，因此我们培养了持续学习的意识。我们意识到需要不断关注新技术和发展动态，以保持自己在这个领域的竞争力。

总体而言，单片机课程设计是一次非常有益的学习经历。通过这门课程，我们不仅掌握了单片机编程的技能，还培养了解决问题和团队合作的能力，为我们的职业发展奠定了坚实的基础。

**致谢**

感谢技术指导老师严继敏老师与唐琳老师的悉心指导教学。

感谢华清远见老师李旖澜、谢开元、魏玉芳各位老师关于职业生涯的指导。

**参考文献**

[1]章小宝.16路舵机控制板使用说明V2.0 [J]，2017.

[2]徐英慧，马忠梅等.ARM9嵌入式系统设计—基于S3C2410与Linux [M].3版.北京.北京航空航大大学出版社，2015.

[3]张仁彦，高正中等.单片机原理及应用 [M].北京:机械工业出版社，2016.

[4][美]史蒂芬.普拉达.C++ Primer Plus[M].6版.北京：人民邮电出版社，2020.

[5][美]布鲁姆，布雷斯纳汉.Linux命令行与shell脚本编程大全[M].北京：人民邮电出版社，2016.

[6]侯殿有.嵌入式系统开发基础—基于ARM9微处理器C语言程序设计[M].5版.北京：清华大学出版社，2019.