江南大学

嵌入式系统原理课程设计

题目： 基于 STM32F103C8T6 和 OneNet 云平台的物联网智能家居系统

物联网工程 学 院 物联网 2301

学 号 1034230105

学生姓名 梁开妍

指导教师 孙顺远

2025 年 7 月

目录

一、 系统原理图 3
二、单片机端系统工作原理 7
2.1 单片机设备端系统框架 7
2.2 系统初始化 7
2.3 数据采集和上传云端 8
2.4 对云端指令的响应和执行 8
2.5 异常处理与恢复 8
2.6 系统运行周期 9
2.7 主循环流程图 10
三、小程序前端设计 11
3.1 小程序首页-监测控制 11
3.2 小程序页面二-智能对话交互 12
3.3 小程序页面三-设备状态展示 15
四、开发环境介绍 17
4.1 嵌入式硬件设计工具选择 17
4.2 嵌入式软件开发工具选择 17
4.3 云端平台选择 17
4.4 前端开发技术选型 18
4.5 前端集成开发环境选择 19
4.6 前端调试环境选择 19
五、自我评价与未来展望 20
5.1 自我评价 20
5.2 未来展望 20

一、 系统原理图

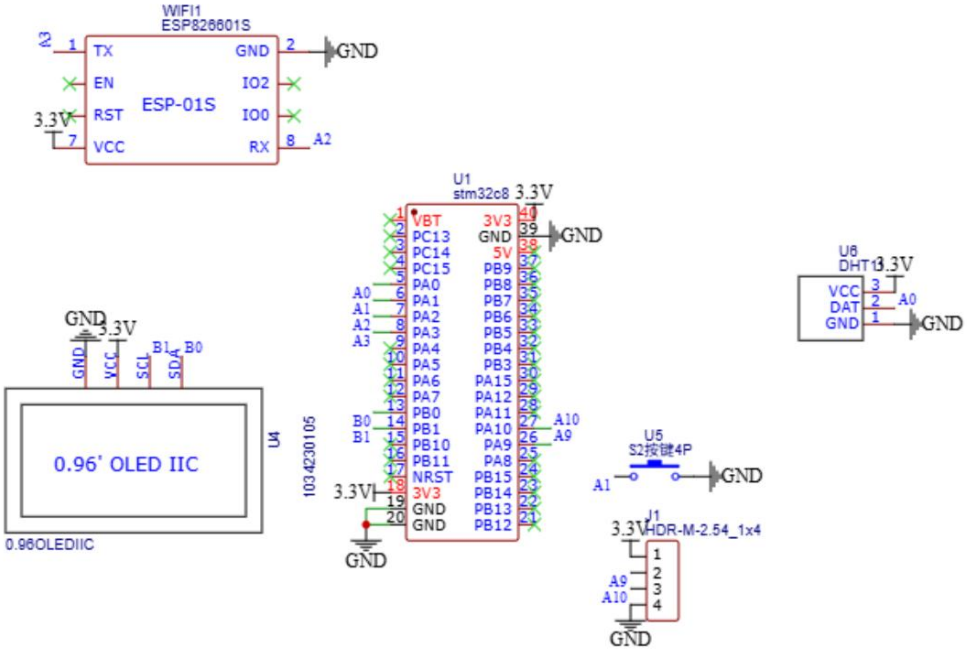

图 1 系统原理图

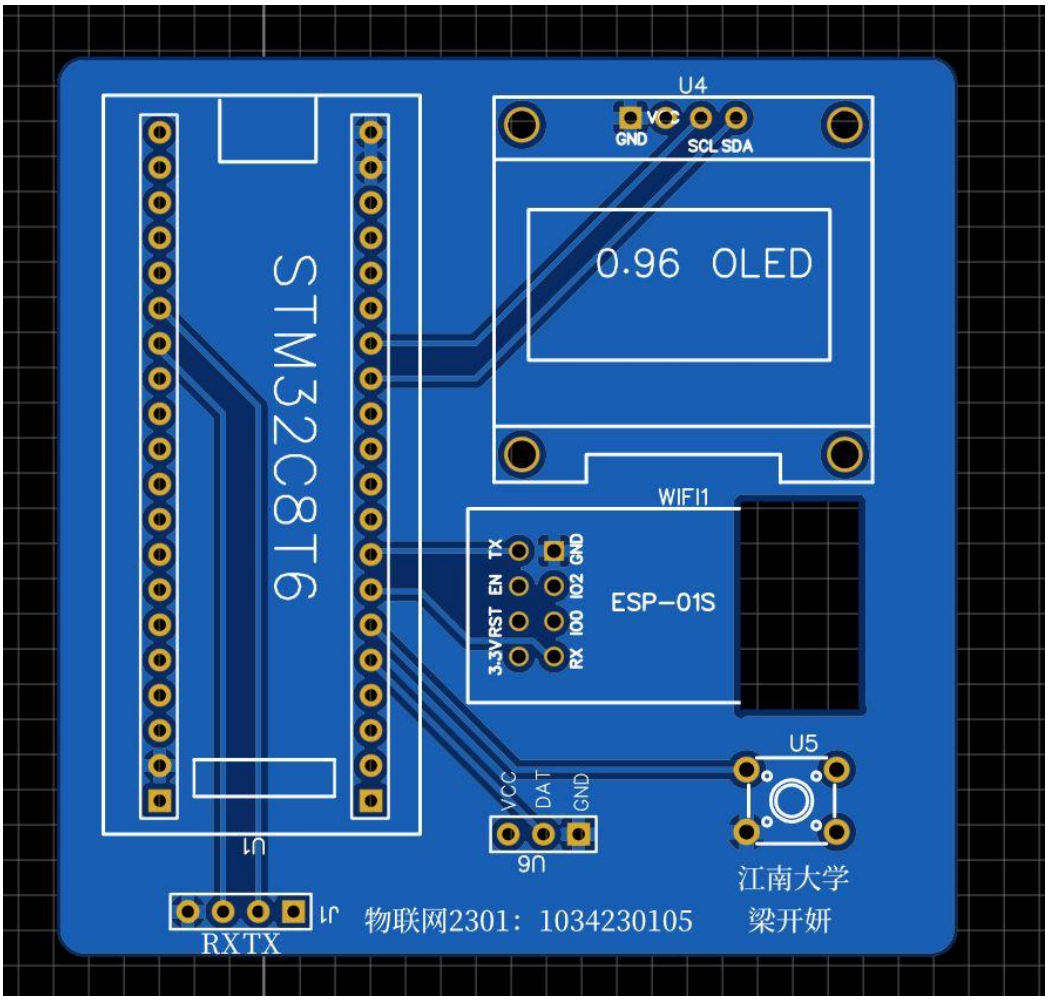
U4
GND
VCC
SCL SDA
0.96 OLED
STM32C8T6
WIFI1
3.3V RST EN TX
RX IO0 IO2 GND
ESP-01S
U5
VCC DAT GND
U6
U1
J1
RXTX
物联网2301：1034230105
江南大学
梁开妍

图 2 2D PCB 图（铺铜）

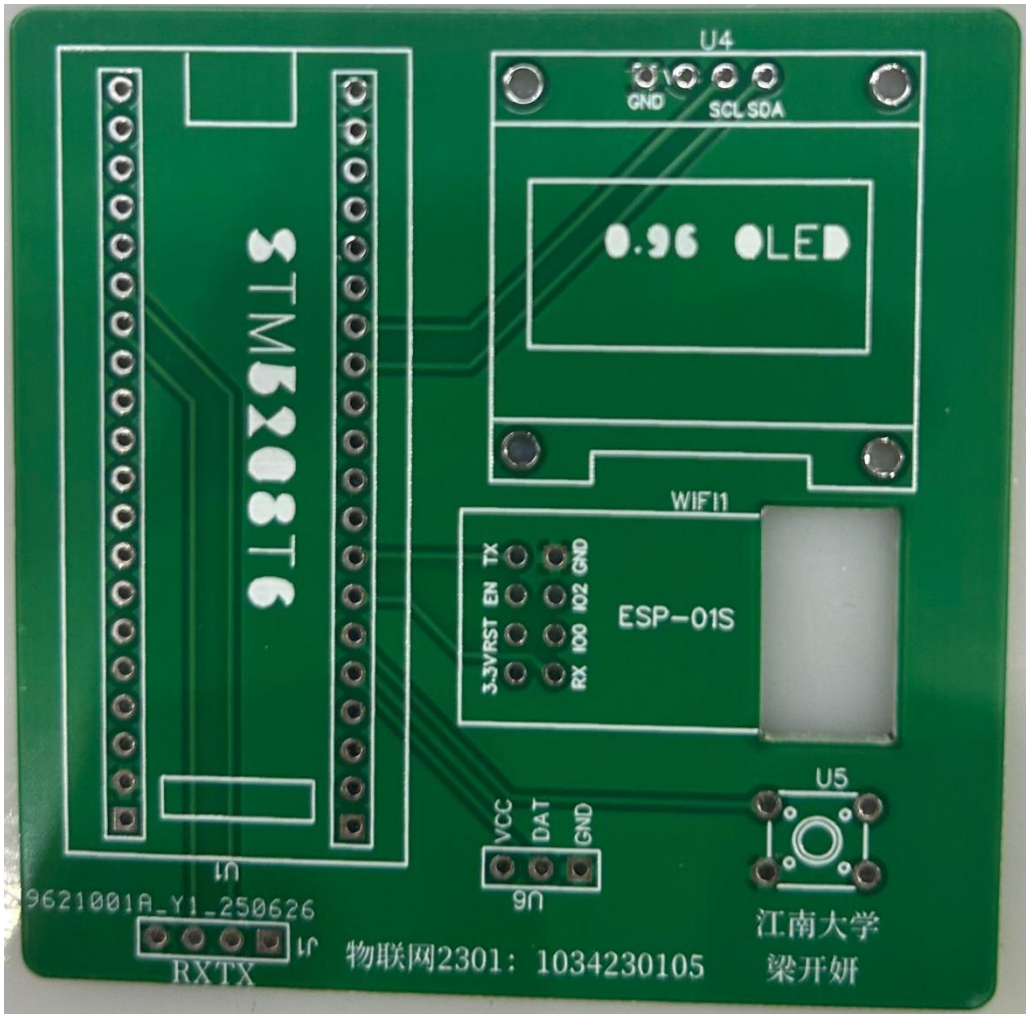
U4
GND
SCL SDA
0.96 OLED
WIFI1
3.3V RST EN TX
RX IO0 IO2 GND
ESP-01S
STM32F103C8T6
U1
U5
VCC
DAT
GND
U6
9621001A_Y1_250626
J1
RXTX
物联网2301：1034230105
江南大学
梁开妍

图 3 PCB 实物图

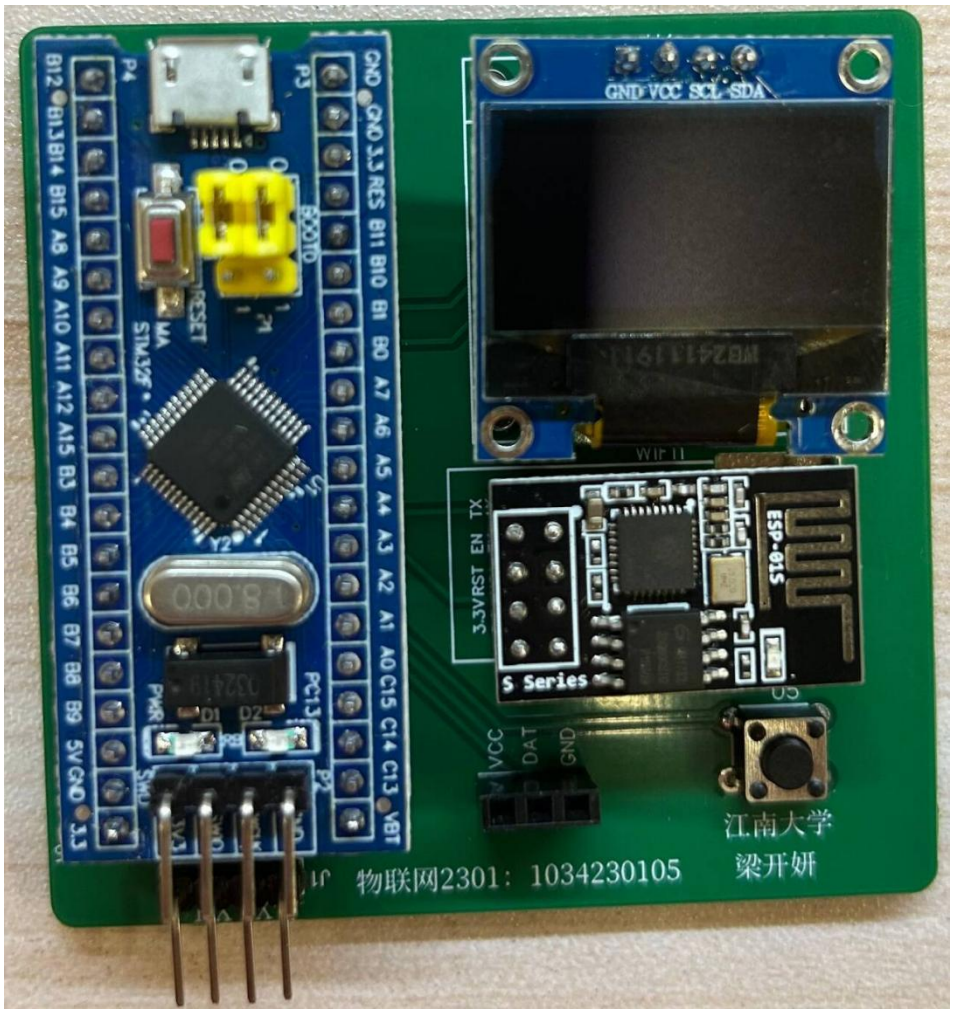
GND VCC SCL SDA
3.3VRST EN TX
ESP-01S
S Series
VCC
DAT
GND
江南大学
梁开妍
物联网2301：1034230105

图 4 单片机端实物图

二、单片机端系统工作原理

2.1 单片机设备端系统框架

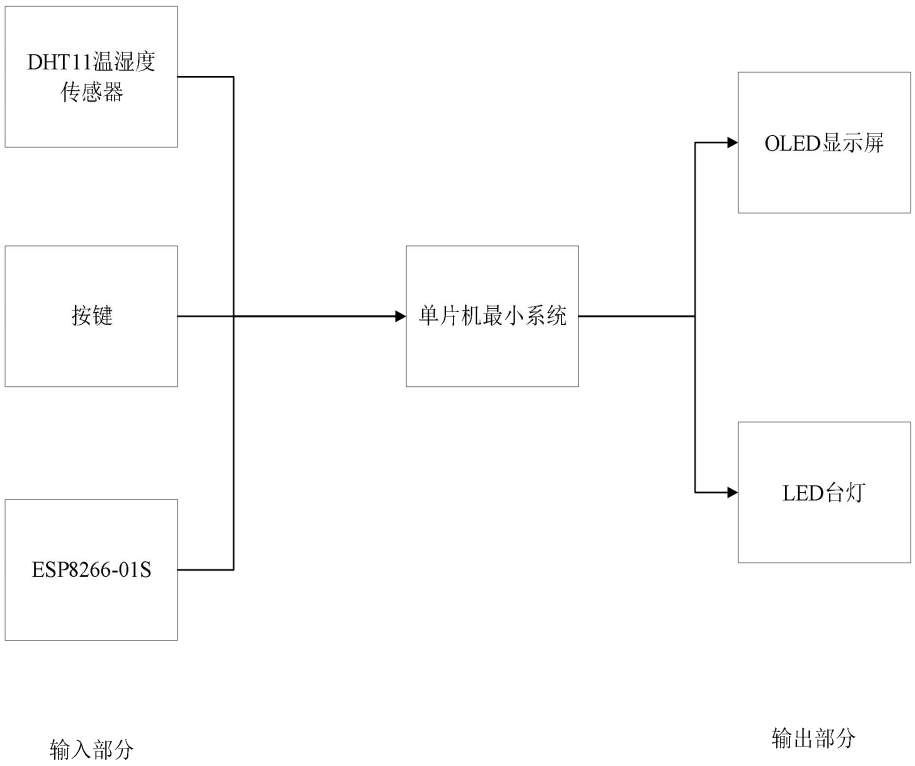

图 5 单片机端系统框架图

整个系统围绕 STM32F103 主控单元构建，形成清晰的输入、处理和输出架构。在输入部分，DHT11 温湿度传感器负责环境数据采集，ESP8266-01S 模块接收来自云端的指令，物理按键则提供用户本地控制能力，这三者均将信息输入到 STM32 主控。处理核心即为 STM32F103C8T6 主控芯片，负责协调整个系统的运行。在输出部分，系统将处理结果通过 OLED 显示屏进行数据显示，通过按键或者云端/小程序端操作进行 LED 状态控制，并利用 ESP8266-01S 模块将数据上报至云端。

2.2 系统初始化

系统启动后进行硬件初始化。STM32 主控初始化 GPIO、USART、I²C 外设。随后初始化 OLED 显示屏，包括加载字库、清屏并显示启动界面。同时 DHT11 温湿度传感器初始化，系统检测工作状态是否正常。再进行网络连接初始化。ESP8266 模块通过发送 AT 指令。连接指定的 WiFi 网络。连接成功后，再通过指令 AT+CIPSTART="TCP","mqtts.heclouds.com",1883 与 OneNET 平台

的 MQTT 服务器建立 TCP 连接。系统使用 HMAC-SHA1 结合 Base64 的加密算法生成设备令牌完成 OneNet 云平台的设备认证。最后建立 MQTT 协议连接订阅云端指令通道，为后续通信做准备。

2.3 数据采集和上传云端

主循环中持续进行环境数据采集处理。核心任务是通调用函数 DHT11_Read_Data(&temp, &humi)来读取 DHT11 传感器的温湿度数据。采集到的数据用 OLED_ShowString(54, 0, temp_str, 16)和 OLED_ShowString(54, 3, humi_str, 16)实时更新到本地 OLED 显示屏上，并同时根据 LED 的当前亮或灭状态显示出来。本地处理完成后，系统将数据上报至 OneNet 云端服务器。首先构造包含温度值、湿度值和 LED 状态值的 JSON 格式数据包，然后通过 MQTT_PacketPublish(MQTT_PUBLISH_ID, "$sys/CRU6Hr7I1v/iot1/thing/property/post", json_data, data_len)函数调用即使用 MQTT 协议将此数据包发布到$sys/CRU6Hr7I1v/iot1/thing/property/post 主题上供云端接收。

2.4 对云端指令的响应和执行

系统持续监听来自云端的指令。它通过 ESP8266 的数据缓冲区接收数据，并识别具有特定特征，本系统为$sys/CRU6Hr7I1v/iot1/thing/property/set 的 MQTT 指令。一旦识别到有效指令，系统对其进行解析，解析出 JSON 格式指令如{"params":{"led":true}}，表示一个开灯指令。解析后，系统执行对应的操作。例如如果解析到 led 字段值为 true，则调用 Led_Set(LED_ON)函数开启 LED。执行动作后，系统会更新 OLED 显示屏上 LED 的状态显示。最后会用 MQTT_PacketCmdResp(cmdid, response_data, &mqttPacket)构造一个响应消息，通过 MQTT 协议返回给云端，确认指令已接收并执行。

2.5 异常处理与恢复

系统设计了机制应对运行中可能出现的异常。对于网络异常，系统会持续检测 TCP 连接状态，并在检测到断线时触发自动重连机制，表现为如果 ESP8266_SendCmd("AT+CIPSTATUS", "CONNECT")返回非零值表示未连接，则重新执行 ESP8266_Init()初始化 WiFi 模块，并调用 OneNet_DevLink()重新连接 OneNet 云平台。在数据处理层面实施了多重校验机制，表现为对 DHT11 读取的数据进行 CRC 校验以确保准确性，对构造和接收的 JSON 数据进行格式合法性检查，对 MQTT 报文进行完整性验证。

2.6 系统运行周期

数据采集和状态检测任务以 200ms 为周期循环执行，按键输入处理和网络数据接收/解析以 10ms 为周期执行。多速率循环结构确保实时完成数据采集到本地显示、云端同步、指令响应的完整流程。

2.7 主循环流程图

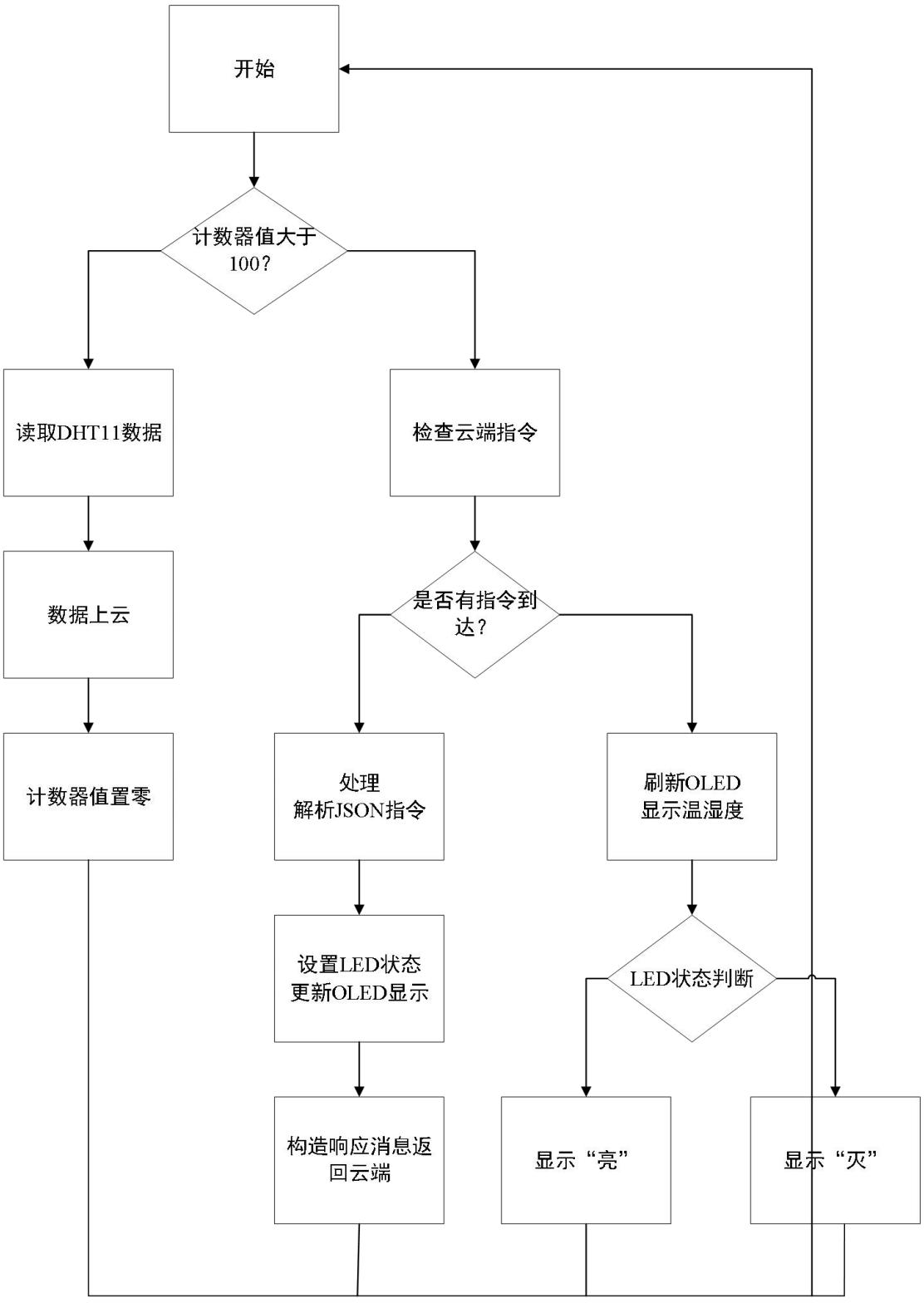

图 6 主循环流程图

三、小程序前端设计

本研究设计了一款基于物联网的智能家居远程控制应用，通过与 OneNet 云平台的数据交互实现设备环境实时监测与控制。系统前端主界面动态展示温湿度传感数据，并提供台灯远程开关功能。用户通过操作界面中的按钮切换台灯状态，云端控制指令经由 HTTP 请求完成设备属性设置。

3.1 小程序首页-监测控制

为构建多维环境监测系统，界面集成双数据源显示功能。最上方区域展示江苏无锡宏观气象信息，包含环境温度、晴好天气状况、云量覆盖及风速数据。再往下侧区域独立显示 DHT11 传感器采集的微环境数据，形成宏观与微观环境对比视图。该功能通过双 API 协同实现：调用高德地图气象接口获取区域气候数据（待申请之后实现），同时访问 OneNet 平台设备数据点接口读取传感器实时数值。

智能设备控制中枢采用可视化交互设计。台灯控制模块通过色彩编码直观呈现状态：绿色按钮表示开启状态，白色按钮表示关闭状态。控制协议采用 HTTP-POST 方法下发指令，当用户触发开关时，前端构造包含设备 ID 与目标状态的 JSON 数据包发送至云端控制接口。

数据可视化增强模块引入动态折线图组件。该组件实时渲染湿度变化趋势，持续存储并展示最近七次采集的湿度数据点，形成直观的环境变化曲线。

系统设计凸显两大核心亮点：双温度对比机制并置显示设备温度与环境温度数据，揭示局部微环境与区域气候的差异；设备位置精确标注功能明确标识"蠡湖大道 1800 号江南大学 DHT11"安装点位，增强系统可信度。如图所示的主界面采用分层布局设计，顶部气象信息区、中部设备控制区、底部趋势图表区与导航栏共同构成完整控制闭环。

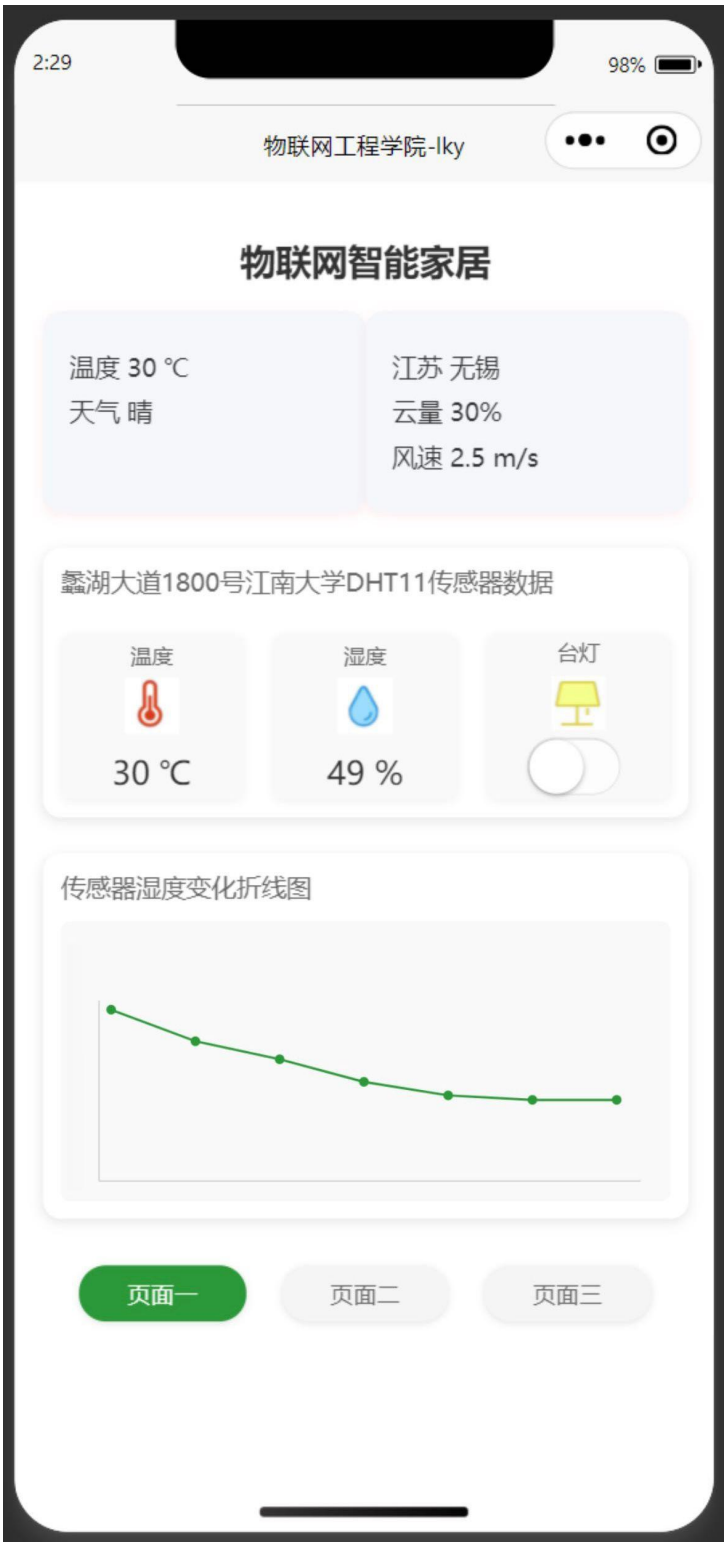

图 7 微信小程序主页面

3.2 小程序页面二-智能对话交互

微信小程序页面二设计了一款基于 DeepSeek 的智能对话交互界面，作为实现设备无关智能服务的核心模块。该页面支持多领域自然语言问答功能，用户可

自由输入天气查询、知识咨询、设备操作指导等多样化请求。

实时对话界面采用动态消息流设计，自动保持最新对话内容可见。智能上下文处理机制持续追踪对话逻辑链条，确保多轮交互的语义连贯性。视觉设计遵循简洁高效原则，通过清晰的信息分区与响应式布局优化人机交互体验。

该模块特别适用于解决硬件控制之外的综合性咨询需求，当遇到超出系统实时处理能力的请求时,如动态天气数据获取，主动提供可操作的解决方案框架。

图 8 微信小程序 DeepSeek 交互页面

图 9 微信小程序 DeepSeek 交互页面

3.3 小程序页面三-设备状态展示

本设计构建了智能家居系统的设备状态监控中心，实现三大核心功能协同运作。人员状态追踪模块动态显示开发者实时状态如"School"并附带状态说明，设备运行监控单元实时反馈 STM32 设备供电状态"电源供电中/没在使用"，灯光控制统计模块记录当日台灯开关次数并提示控制入口。当用户访问页面时，系统自

动触发状态检测机制：在线状态下显示实时数据，离线时展示最后更新时间，同步更新设备状态并保持缓存数据可用性。

数据管理采用双重机制保障时效性。在页面显示周期内，通过全局状态同步获取开关计数数据，同时启动 3 秒轮询机制持续获取设备数据。智能状态判定算法计算当前时间与最后更新时间差值，超过 30 分钟即判定设备为"没在使用"状态，确保状态反馈准确性。

图 10 微信小程序设备状态页面

四、开发环境介绍

4.1 嵌入式硬件设计工具选择

嘉立创 EDA 为整个项目的电路实现提供了专业的设计平台。在 STM32 主控板与外设模块，包括 ESP8266 WiFi 模块、OLED 显示屏、DHT11 传感器以及 LED 控制电路的设计过程中，我用嘉立创实现了从概念到成品的完整设计。其集成的原理图绘制功能准确表达了各模块的电气连接关系，特别是 STM32F103C8T6 与 ESP8266 之间的 UART 通信接口、DHT11 数据采集电路等关键连接。设计规则检查 DRC 功能可以拦截了潜在短路风险。最终通过软件直接生成的 Gerber 文件与钻孔数据，无缝衔接嘉立创的 PCB 制造服务，实现了设计到生产的闭环。

4.2 嵌入式软件开发工具选择

Keil MDK 5.36 作为 ARM 架构的专属开发环境，承担了系统固件的开发责任。该环境为 STM32F103C8T6 提供了完整的工具链支持，包括 C/C++compiler5.06（在 MDK5.37 版本之后不再自动安装）、汇编器、链接器以及强大的调试器。在程序烧录阶段，通过 ST-Link V2 调试器实现 SWD 接口通信，将编译生成的 HEX 文件直接写入 Flash 存储器，完成固件部署。

4.3 云端平台选择

本系统选择 OneNet 作为云端支撑，源于多方面的技术优势与生态适配性。在协议兼容性与开放性方面，平台原生支持 MQTT/HTTP 等主流物联网协议，提供完整的 RESTful API 文档和 SDK，如本设计直接调用的 onenet.c 中的 API 函数。这种开放性架构有效避免了私有协议导致的生态封闭问题，为设备接入提供了技术自由度。平台通过设备动态注册机制实现自动化管理，该机制支持设备首次连接时自动创建云平台实体，实时监控设备在线/离线状态，并动态分配唯一设备密钥，实现全生命周期的设备管理。

在通信可靠性方面，平台提供运营商级的基础设施保障，消息传输延迟小于 200ms，保证了控制的实时响应；数据默认保存 30 天，具有不错的历史数据回溯能力。

安全体系设计采用四重防护机制：设备级 API Key 加密、动态 Token 时效控制、传输层报文 ID 防重放保护，以及 HTTPS API 全加密（本设计因 STM32 资源限制未启用）。动态 Token 生成算法具体实现为：

//(onenet.c)

```
OneNET_Authorization() {
    string = et + method + res + ver //  拼接签名要素
    sign = Base64(HMAC-SHA1(api_key, string)) //  加密核心
    Token = "version=...&sign=" + URLEncode(sign)
}
```

平台提供完善的开发工具链支持，包括虚拟设备 mqtt.fx 调试环境，无需硬件即可测试数据流、拖拽式数据可视化引擎、支持短信/邮件通知的阈值触发告警规则引擎，以及 PHP/Python 等后端集成方案。

对比其他主流平台，阿里云 IoT 功能强大但配置复杂学习曲线陡峭，腾讯云 IoT 在微信生态整合好但 MQTT 文档不完善，AWS IoT 全球服务但国内延迟高且免费额度有限。OneNET 在部署成本上具有显著优势，其设备连接费、消息传输费、API 调用费、数据存储费全部为零。

其教育免费策略实现课程设计零成本，千万级消息免费额度满足原型开发需求，且无强制绑定的增值服务。本地化服务优势包括全国部署的 8 大低延迟接入点（本设计使用 mqtts.heclouds.com 节点）、全中文技术文档，以及活跃的开发者社区。

综合而言，OneNET 在协议开放性、安全机制、教育支持及成本控制上形成独特优势，其设备管理 API 与 MQTT 实现深度契合 STM32+ESP8266 架构需求，成为本项目物联网设备快速接入的理想选择。

4.4 前端开发技术选型

本系统采用 Uni-app 框架作为跨平台前端开发的核心技术方案。Uni-app 作为跨平台开发框架，在打包至微信小程序时展现出显著的技术优势。其跨平台开发效率体现在采用 Vue.js 统一语法体系，通过条件编译指令如#ifdef MP-WEIXIN 实现代码自动适配微信小程序平台，免除开发者学习 WXML/WXSS 等平台特有语法的负担。例如 uni.navigateTo API 函数在 Web/H5/小程序端保持完全一致行为，大幅降低开发成本约 70%。当项目需扩展 H5 版本时，Uni-app 仅需增加 20%工作量，而原生开发则需巨大重写成本。

性能优化机制通过双重技术路径实现：首先将 Uni-app 组件编译为原生小程序组件，其次采用虚拟 DOM 技术减少 30%的 setData 通信量。构建时自动处理样式单位转换（如 300rpx 适配不同屏幕）和资源压缩，确保运行效率。工程化优势表现为统一构建流程，支持模块化开发=。

与微信生态的深度集成通过 uni 命名空间实现，直接调用微信原生能力。在 pages.json 中自动配置小程序分包规则，优化加载性能。多端一致性保障采用 Flex 布局结合 rpx 单位方案，关键组件在小程序端使用原生实现，并通过条件编译隔

离平台差异逻辑：

调试部署流程高度简化，支持 HBuilderX 一键扫码真机调试，云端打包服务免除证书配置环节，CI/CD 集成实现自动化提交微信审核。性能边界方面，复杂 SVG 图表需控制数据点数，高频更新的传感器数据建议采用 WebSocket 替代定时请求。

Uni-app 为微信小程序开发展现出工程化、跨端适配与性能优化的三位一体优势，其技术特性完美契合物联网应用快速迭代需求，当前项目结构正是这一价值主张的实证。

4.5 前端集成开发环境选择

HBuilderX 作为全栈式前端集成开发环境，专为高效开发跨平台应用而设计，尤其适合本系统这种物联网监控与聊天机器人小程序的复合型项目。其核心架构由三大功能模块构成：编辑器引擎提供智能代码提示与语法高亮，运行调试系统支持实时预览与真机调试，发布管理系统实现云端一键打包。这些模块协同工作形成完整的项目开发流程，涵盖从项目创建到最终发布的各个环节。

该开发环境具备四大核心优势显著提升物联网开发效率。内置设备模拟器支持多屏幕尺寸适配，实时日志流功能可直接显示 MQTT 连接状态与网络请求详情。跨平台热更新系统实现高效开发迭代，当开发者修改 Vue 组件时，500 毫秒内即可在微信模拟器看到更新效果。可视化云端发布流程实现代码编写到微信后台上传的自动化流水线。

项目开发工作流采用优化策略。界面构建使用 view 组件配合 v-for 指令生成动态列表，如聊天消息展示。API 调试通过 uni.request 连接物联网平台与 AI 服务，右键浏览器运行功能实现 CSS 实时调试。性能优化方面启用发行模式压缩代码，聊天模块实施分包加载减少首屏资源。

4.6 前端调试环境选择

微信开发者工具为微信官方提供的一站式开发环境。其环境架构由四大核心模块构成：模拟器实现多设备预览，调试器提供代码诊断能力，性能分析工具优化运行效率，云开发控制台集成云端服务。这些模块协同支持完整项目工作流，涵盖从代码导入、实时预览到真机调试和版本发布的各个环节。

该工具与 Uni-app 框架实现深度集成，当 HBuilderX 编译生成 dist 目录后，开发者工具自动导入并实施语法转换，将 Uni-app 组件映射为小程序原生组件。实时热重载机制确保代码变更立即同步至预览界面，错误定位系统精准映射小程序报错到 uni-app 源码位置。在物联网开发场景中，设备控制 API 调试界面可实时显示 uni.request 发往 iot-api 的请求与返回数据，支持无硬件条件下的传感器数据模

拟，并精确测量设备接口响应时间。发布流程遵循代码提交、体验版生成、性能优化、正式发布的标准化路径。

安全合规保障体系自动校验 IoT API 域名白名单，引导位置权限申请配置，监控敏感 API 调用如 crypto.createHmac 并提示功能声明要求。

五、自我评价与未来展望

5.1 自我评价

总评：优。

嵌入式硬件设计部分：良。原因：原理图和 PCB 设计清晰合理，成功实现主控电路、WiFi 模块、按键电路和 OLED 显示电路，PCB 上包含学号姓名信息；缺点：外设太少，仅仅包含一个温湿度传感器。

嵌入式软件设计部分：良。原因：完整实现 MQTT 协议栈、数据采集、指令响应和异常恢复，代码模块化设计清晰；缺点：功耗较高，未优化休眠机制。

前端开发部分（包括与云端交互）：优。原因：Uni-app 框架实现微信小程序界面，实时显示温湿度，并将湿度变化可视化折线图展示，支持远程控制，具有与 DeepSeek 聊天功能，具有数据统计功能。

5.2 未来展望

对于微信小程序前端，在数据可视化层面，可以在现有数值显示基础上扩展动态趋势分析功能。通过集成微信小程序原生 canvas 组件或 F2 图表库，调用 OneNet 历史数据 API 获取 30 天存储的温湿度记录。该改造即可让用户直观查看 24 小时温度变化规律（当前只能显示最近七个获取到的湿度数值），为空调启停等决策提供数据支撑。

未来新增设置页面允许设定温度上限，通过微信模板消息 API 推送报警通知。硬件联动需单片机端配合扩展执行器，小程序下发阈值指令至设备后，当温度超标自动开启风扇。该改进使系统从被动监控转向主动预警，大幅提升实用性。

为了进一步交互体验优化还可以集成微信语音识别 API，实现语音控制指令解析（如“开灯”触发设备操作）。同时引入 uni-gesture 手势库支持滑动调光，需硬件端升级 STM32 的 LED 控制模块以支持 PWM 调光协议。