

**本 科 生 毕 业 论 文(设计)**

**基于树莓派的智能家庭健康监测系统设计**

**学 院： 数学与计算机科学学院**

**专 业： 物联网工程**

**姓 名： 陈玄**

**学 号： 21213020490120**

**指导教师/职称： 王武/副教授**

**提 交 日 期：**

**教 务 处 制**

云南民族大学本科毕业论文（设计）原创性声明

本人郑重声明：所呈交的毕业论文(设计)，是本人在指导教师的指导下进行研究工作所取得的成果。除论文中已经注明引用的内容外，本论文没有抄袭、剽窃他人已经发表的研究成果。本声明的法律结果由本人承担。

论文(设计)作者签名：

日 期： 年 月 日

……………………………………………………………………………

云南民族大学毕业论文(设计)使用授权的说明

本人完全了解云南民族大学有关保留、使用毕业论文(设计)的规定，即：学校有权保留、送交论文的复印件，允许论文被查阅，学校可以公布论文(设计)的全部或部分内容，可以采用影印或其他复制手段保存论文(设计)。

（保密论文在解密后应遵守）

指导教师签名： 论文(设计)作者签名：

日 期： 年 月 日

**摘要**

近年来，随着物联网、人工智能和嵌入式技术的迅猛发展推动了智能硬件和智能传感器的广泛应用。本文介绍的基于树莓派的智能家庭健康监测系统，主要实现家庭健康数据采集与分析、智能的环境控制、信息交互和智能管理等功能，旨在为家庭中的成员提供一个智能化，个性化的健康监测和生活环境，为用户提供一种高效便捷的健康管理服务。首先简述该系统的研究背景和意义，随后深入探讨系统的设计方案，包括系统架构、硬件和软件设计，已经整个系统的核心技术与特色。硬件层面，以STM32F205RET6作为核心处理器负责数据采集与控制，整合多种传感器的数据以及控制执行设备，并利用树莓派5作为数据处理与显示平台，借助FreeRTOS实时操作系统增强系统的实时性和稳定性；软件层面，采用Qt框架开发用户界面程序，通过调用百度大模型提供的API实现语音识别和语音对话的功能，并且依托阿里云服务器搭建MySQL数据库，MQTT服务器，完成数据的存储，传输任务。最终通过功能测试与稳定性评估，验证系统的实用性与可靠性。该系统综合运用了传感器、微控制器、MQTT/HTTP通信、MySQL数据库、FreeRTOS、Qt开发以及百度API提供的语音识别和对话技术，针对家庭健康监测场景进行了专属定制开发。本文的研究成果为嵌入式智能家庭健康系统的设计与实现提供了重要参考，推动了智能家庭健康监测系统的广泛推广与深入应用。

**关键词：**智能家庭健康监测；树莓派；百度大模型；MQTT；STM32F205RET6

**Abstract**

In recent years, the rapid development of the Internet of Things, artificial intelligence and embedded technology has promoted the wide application of intelligent hardware and intelligent sensors. The intelligent family health monitoring system based on Raspberry PI introduced in this paper mainly realizes the functions of family health data collection and analysis, intelligent environmental control, information interaction and intelligent management, aiming to provide an intelligent and personalized health monitoring and living environment for family members, and provide users with an efficient and convenient health management service. Firstly, the research background and significance of the system are briefly described, and then the design scheme of the system is deeply discussed, including the system architecture, hardware and software design, as well as the core technology and characteristics of the whole system. At the hardware level, STM32F205RET6 is used as the core processor to collect and control data, integrate the data of various sensors and control execution equipment, and use Raspberry PI 5 as the data processing and display platform, with the help of FreeRTOS real-time operating system to enhance the real-time performance and stability of the system. At the software level, Qt framework is adopted to develop user interface programs, voice recognition and voice dialogue functions are realized by invoking API provided by Baidu Grand Model, and MySQL database and MQTT server are built by Alibaba Cloud server to complete data storage and transmission tasks. Finally, the practicability and reliability of the system are verified through functional testing and stability evaluation. The system integrated use of sensors, microcontrollers, MQTT/HTTP communication, MySQL database, FreeRTOS, Qt development and Baidu API provided voice recognition and dialogue technology, customized development for home health monitoring scenarios. The research results of this paper provide an important reference for the design and implementation of embedded intelligent home health system, and promote the wide popularization and deep application of intelligent home health monitoring system.

**Key words:** smart home health monitoring; Raspberry PI; Baidu Big model; MQTT; STM32F205RET6

目录

[1 绪论 1](#_Toc31713)

[1.1 研究背景和意义 1](#_Toc18570)

[1.1.1 研究背景 1](#_Toc32580)

[1.1.2 研究意义 2](#_Toc13737)

[1.2 国内外研究现状 3](#_Toc12464)

[1.3 研究内容和思路 4](#_Toc6948)

[1.3.1 研究内容 4](#_Toc26269)

[1.3.2 研究思路 4](#_Toc19297)

[1.4 论文组织结构安排 5](#_Toc20605)

[2 智能家庭健康监测系统概述 7](#_Toc19489)

[2.1 智能家庭健康监测系统的定义与组成 7](#_Toc2232)

[2.1.1 系统定义 7](#_Toc4888)

[2.1.2 系统组成 7](#_Toc1917)

[2.2 功能特点 8](#_Toc13301)

[2.3 智能家庭健康监测系统的发展现状与应用前景 8](#_Toc1290)

[2.3.1 发展现状 8](#_Toc29698)

[2.3.2 应用前景 9](#_Toc3939)

[3 智能家庭健康监测系统设计 10](#_Toc6031)

[3.1 智能家庭健康监测系统的功能需求分析 10](#_Toc30352)

[3.1.1 需求分析概述 10](#_Toc8114)

[3.1.2 外部接口需求 10](#_Toc3020)

[3.1.3 模块功能需求分析 11](#_Toc9008)

[3.2 系统结构分析 12](#_Toc21339)

[3.3 系统工作流程分析 13](#_Toc906)

[4 智能家庭健康监测系统开发 14](#_Toc1499)

[4.1 监测/采集终端开发 14](#_Toc1916)

[4.1.1 硬件设计 14](#_Toc25931)

[4.1.2 软件设计 20](#_Toc28278)

[4.1.3 监测/控制终端程序设计 23](#_Toc19232)

[4.2 控制/显示终端开发 31](#_Toc27858)

[4.2.1 软件移植和环境配置 31](#_Toc16411)

[4.2.2 Qt软件程序开发 33](#_Toc16090)

[4.3 智能家庭健康监测系统服务器搭建 42](#_Toc13247)

[4.3.1 MQTT服务器的搭建 42](#_Toc12505)

[5 智能家庭健康监测系统测试 44](#_Toc22685)

[5.1 硬件测试 44](#_Toc11464)

[5.2 软件编译 46](#_Toc5070)

[5.2.1 单片机程序编译 46](#_Toc8795)

[5.2.2 Qt程序的编译 46](#_Toc8519)

[5.3 系统功能测试 47](#_Toc1143)

[5.3.1 系统通信功能测试 47](#_Toc23550)

[5.3.2 功能测试总结 48](#_Toc12119)

[6 总结与展望 49](#_Toc20463)

[6.1 研究总结 49](#_Toc25840)

[6.2 研究展望 50](#_Toc11646)

[参考文献 52](#_Toc14673)

[致谢 53](#_Toc12499)

# 绪论

## 研究背景和意义

### 研究背景

据 2020 年第七次人口普查数据,我国60岁以上的老年人口总数约为2亿 6300万,占当年人口总数的18.7%｡其中,65岁及以上的人口比例为13.50%,约为3500万｡与前一个10年相比,60岁和65岁两个年龄段的增长率分别为5.44% 和 4.63%｡据预测,至2050 年,我国将迎来老龄化人口高峰,平均每 3 个人中就会有一位老人[1]｡在老龄化社会中,慢性病的高发使得日常健康监测显得尤为重要｡家庭成员通过在日常生活中对各项生理参数进行监测,能够在疾病早期预警身体异常和疾病,从而提高个家庭生活质量[2]｡传统的医疗体系主要依赖医院和和专业机构进行疾病诊断与治疗，然而这种模式在面对日益庞大的老年群体时显得捉襟见肘。不仅医疗资源分布不均，而且频繁就医对老年人和家庭来说都增加了经济与时间成本。因此，将健康管理的重心逐步转移到家庭环境中，通过日常监测实现疾病的早期预警，成为提示老年人生活质量、减轻社会医疗压力的重要途径。

随着人们对健康生活的日益关注,家庭健康监测设备需求不断增加｡无论是老年人还是年轻人，都希望通过科学的方式了解自身的健康状况，并且采取预防措施。然而现有设备功能单一,难以满足现代家庭对多参数监测与智能化操作的需求｡随着物联网(Internet of Things,IoT)技术的快速发展,使得家庭监测身体健康数据有了全新的方法[3]｡通过将传感器、无线通信和云服务等技术结合起来，物联网能够实现多种健康数据的采集、传输和存储。与此同时，人工智能技术的进步使得数据的分析和智能化反馈成为现实。AI通过算法模型分析采集到的数据，识别其中的异常，预测潜在的健康风险，并为用户提供科学的健康管理建议。基于此，结合物联网与人工智能技术开发一套多参数家庭健康监测系统，不仅能够满足现代家庭健康的实际需求，还能在老龄化社会中发挥重要作用。

在此背景下，本系统旨在设计并实现一套基于树莓派的智能家庭健康监测系统。该系统通过多种传感器采集心率、血氧饱和度、体温、噪声和空气质量等，并借助物联网和云服务器技术将这些进行传输和存储。百度大模型可以分析数据，提供个性化的建议，该系统还强调数据的长期存储和分析｡通过MySQL数据库,系统可以记录用户的健康数据,为用户提供直观的健康状况变化参考[4]｡帮助家庭成员及时掌握自身健康状况并预防疾病的发生，为家庭提供全面的健康保障。

### 研究意义

1. 现实意义

智能家庭健康监测系统为应对老龄化社会和慢性病高发的现实需求提供了解决方案｡通过全面的健康数据监测,家庭成员能够随时掌握自身及家人的健康状况,并在疾病早期发现异常,为后续治疗争取时间｡这种系统不仅能够改善家庭生活质量,还能够有效减少因健康问题导致的医疗费用支出｡

1. 社会意义

本设计也为医疗资源的优化利用提供了可能｡传统医疗资源分布不均,尤其在偏远地区,优质医疗资源难以触达｡而智能家庭健康监测系统通过提供的个人健康数据,能够实现初步的远程诊断支持,为基层医疗服务减轻压力[5]｡

从政策支持角度看,本系统的研究和开发也与国家“健康中国2030”规划目标相契合,为推动国家医疗和健康服务产业的数字化转型提供了实践依据[6]｡

1. 个人意义

对于我个人而言,本项目是一次理论结合实践的宝贵机会｡我将以“基于树莓派的智能家庭健康监测系统”为研究目标,通过该系统的设计与开发,将课堂学习的嵌入式技术､物联网通信协议､数据分析方法等知识应用到实际项目中｡

在项目中,我将完成数据采集､传输与分析模块的设计与实现,并深度研究如何利用物联网技术实现家庭健康参数的智能监测｡同时,通过开发云端数据存储与AI健康状态分析模型,我将进一步理解物联网与人工智能技术的集成应用[7]｡

## 国内外研究现状

智能家庭健康监测系统的研究在国内外都呈现蓬勃发展的态势，国内的研究近年来尤为活跃，主要得益于物联网，人工智能等技术的飞速进步和人们对健康管理需求的日益增长。学者们往往比较关注如何将这些技术应用到实际中，比如通过可穿戴设备和智能家居传感器采集心率、血氧等数据，再结合云技术进行传输、存储、分析等，这种实践导向的研究在国内占据了主导地位[8]。很多团队致于开发低成本、高精度的设备，希望让系统进入更多的普通家庭，同时学者们也在研究如何应对慢性病管理、老年人监护等多样化的场景。此外，国家政策的支持，比如“健康中国2030”战略[6]，也为这项研究提供了不少助力，但要把技术真正普及到千家万户，标准化的制定和用户的接受度还需要时间去磨合。

反观国外，这方面的研究起步更早，整体显得更加成熟，理论和实践的结合也更加的紧密。且国外已从技术设备开发与系统集成应用转向探讨老年人对技术的接受与采用;国际研究的知识基础可分为老年技术采纳评估、健康监测技术创新和智能健康管理应用三类。 后续应加强工程学和医学多领域的跨学科交叉研究,在健康监测技术与医疗产品融合、用户体验优化及长期效用验证等方面进行深入探讨[9]。

总的来说，国内外的研究目标其实是一致的，都是想通过技术提升健康管理效率和生活质量，用的技术也大同小异，比如无线传感器、物联网、云技术、AI、边缘计算等技术。只是侧重点不一样：国内更倾向于把系统做实惠、做普及，解决基础的需求；国外则在技术的前沿性和专业化上更有优势，同时理论研究和商业化也更成熟。未来随着技术的更新迭代，像6G、区块链和人工智能技术的发展可能会带来更多的可能性，国内外的研究者都在努力探索如何用好这些工具，设计出既智能又安全的系统，让家庭的健康监测真正的融入家庭生活中。

## 研究内容和思路

### 研究内容

本文的研究目标是设计并实现一个基于树莓派的智能家庭健康监测系统，旨在通过现代技术的融合，实现对人体健康指标和室内环境的实时监测、智能反馈和便携管理。以STM32微控制器作为数据采集与控制核心，运行FreeRTOS操作系统，负责多传感器数据采集与传输;以树莓派5作为主控终端，运行Linux系统，开发Qt界面的用户程序，实现数据的接收、分析、显示与控制功能。系统不仅能监测心率、血氧、体温等数据，还能感知空气质量、温湿度、噪声等环境参数，并通过自动控制设备如风扇、雾化器优化家庭环境，同时提供健康记录与AI分析建议，满足家庭健康管理的需求。

本文的研究内容主要包括：

1. 智能家庭健康监测系统设计。首先分析系统的功能需求，包括健康数据采集、环境监测、自动反馈和用户交互等，然后选择合适的硬件设备，如STM32F205RET6芯片，树莓派5，以及MAX30102、AHT20等传感器模块。确定好软件开发工具，如KEIL用于单片机的编程，Qt和MySQL用于树莓派界面与数据管理；设计系统结构，采用MQTT协议实现STM32与树莓派之间的通信，并制定传感器数据上传与控制命令的协议。
2. 智能家庭健康监测系统开发。首先搭建开发环境，在STM32端开发裸机驱动程序验证各个传感器的功能，并移植FreeRTOS实现多任务管理，通过WIFI模块将数据上传至MQTT Broker;在树莓派端基于Qt开发登录、注册、Home主界面、健康记录与健康报告等程序，集成MySQL存储数据，并通过HTTP协议调用百度AI大模型进行分析，同时实现风扇、雾化器等设备的控制。
3. 智能家庭健康监测系统的性能评估与优化。通过实验验证系统的各项功能，如数据的采集的精度、传输的实时性以及控制响应的可靠性，根据测试过程中暴露的问题进行改进和优化，最终构建一个高效、可靠的智能家庭健康监测方案。

### 研究思路

1. 查阅相关文献。通过查阅国内外关于智能家庭健康监测系统的相关文献，了解当前的研究现状和技术趋势。结合自己的设计想法，评估使用树莓派和STM32协同工作的可行性，提出文本的主要研究内容，即开发一个集健康监测、环境感知和自动反馈于一体的智能系统。
2. 硬件选型。选用树莓派5作为主控核心，负责数据处理和用户交互,STM32F205RET6微控制器作为数据采集与控制设备的核心处理器。根据功能需求，选择MAX30102模块监测心率和血氧，GY906探头测量体温，Y01传感器和DC01红外模块分别检测空气质量和PM2.5,AHT20采集环境的温湿度，LM2904感知噪声；同时使用直流电机、雾化器、舵机和蜂鸣器作为反馈设备，通过使用嘉立创工具绘制PCB,然后焊接组装形成完整的硬件系统。
3. 架构选择和软件选型。系统采用C/S架构，树莓派作为服务端处理数据并提供操作界面，STM32作为客户端采集并上传数据。软件开发选用跨平台的Qt作为树莓派端的主要技术，用于界面设计，同时集成MySQL数据进行数据存储和管理；STM32端使用KEIL开发单片机程序，并移植FreeRTOS操作系统。
4. 通信功能设计。设备间通过WIFI实现网络通信，选用MQTT协议传输数据，确保采集数据的实时性和可靠性。设计JSON数据格式，用于树莓派与STM32之间的高效通信，如传感器数据上传和设备控制指令的下发。
5. 系统测试。通过实际测试验证系统的核心功能，包括数据采集的准确性、传输的及时性以及控制的稳定性。

## 论文组织结构安排论文组织结构图

图 1.1论文组织结构图

第一章：绪论。围绕智能家庭健康监测的现实需求与技术趋势，阐明选题意义，分析国内外研究现状，确定基于树莓派和STM32智能家庭健康监测系统的研究内容与思路。

第二章：智能家庭健康监测系统概述。通过文献调研定义系统并分析其组成，介绍本文设计的健康监测与环境控制的功能特点，探讨其在家庭和医疗领域的应用潜力。

第三章：智能家庭健康监测系统设计。分析系统的功能需求，设计硬件与软件结构，明确数据采集、传输、控制及用户交互的功能模块。

第四章：智能家庭健康监测系统开发。介绍硬件选型与组装、STM32数据采集终端、树莓派控制显示终端及服务器搭建的开发流程与功能实现。

第五章：测试硬件采集精度与控制响应、软件通信与界面稳定性及整体功能表现，优化系统性能。

第六章：总结与展望。总结基于树莓派的智能家庭健康监测系统的设计与实现成果，分析存在的问题，展望未来技术改进与应用扩展方向。

# 智能家庭健康监测系统概述

## 智能家庭健康监测系统的定义与组成

### 系统定义

智能家庭健康监测系统是一种基于物联网技术、传感器和无线通信构建的综合性系统，旨在实时采集并监测老年人和家庭成员的人体健康指标以及家庭环境参数。赵有明[10]认为，智能家庭健康监测系统主要针对慢性病患者和独居老人，强调通过低成本、实时监测实现家庭健康管理，减轻医疗负担，同时通过服务器端与客户端通信传输数据，确保医生和家属及时了解健康状况。凌信航[11]补充，该系统通常以微控制器为核心，通过WIFI、MQTT或NB-INT等技术将数据传输至云平台端或移动端应用，实现远程监控、异常报警和智能控制等功能，强调系统整合可穿戴设备与室内环境监测，满足后疫情时代人们对家庭健康和环境健康的需求。总体而言，智能家庭健康监测系统以低成本，高精度、实时性和用户友好性为特点，适用于家庭健康管理、远程监护和医疗辅助场景。

### 系统组成

本文基于树莓派的智能家庭健康监测系统的结构主要体现为不同功能模块的划分，具体包括以下几个方面：

1. 健康监测模块。包括心率、血氧、体温等传感器。通过STM32微控制器采集人体健康数据，并将数据传输至树莓派进行处理，辅助用户监测自身体健康状态。
2. 环境调节模块。包括空气质量监测系统、温湿度调节系统以及噪声监测系统，通过风扇、雾化器、舵机、继电器等设备实现对室内环境的智能监测和调节，营造健康、安全的居家环境。
3. 智能控制模块。利用无线通信技术WIFI和MQTT协议，实现对家庭设备的远程控制管理，例如通过树莓派Qt界面远程开关风扇、雾化器或调节舵机开窗等功能。
4. 人工智能交互模块。通过HTTP协议调用百度AI大模型，实现健康数据的分析与对话功能，生成个性化健康建议，并为用户提供咨询服务，增强系统的智能化和用户体验。

## 功能特点

本文介绍的智能家庭健康监测系统，主要具备以下几个核心功能特色：

1. 健康数据与环境监测。通过心率血氧传感器、体温探头、空气质量传感器、PM2.5传感器、噪声传感器及温湿度传感器，实现人体健康数据和室内环境数据的采集。
2. 智能数据传输和云端互联。基于STM32的多种通信接口采集传感器数据，并通过WIFI和MQTT协议上传至云端，支持远程访问与数据管理。
3. AI分析与个性化建议。将采集的数据上传至百度AI模型进行分析，生成健康建议，同时支持AI对话功能，增强交互性。
4. 用户友好交互设计。提供美观的登录注册界面、多功能数据显示界面以及软键盘支持，适配家庭多用户管理，操作便捷直观。

## 智能家庭健康监测系统的发展现状与应用前景

### 发展现状

随着物联网、人工智能、传感器和云计算等技术的快速发展，智能家庭健康监测系统正逐步成为智能家居邻域的重要组成部分。该系统依托微控制器、嵌入式平台以及多样化的传感器模块，实现对人体健康指标和室内环境的实时监测与智能控制。当前，市场上已出现多种类似的产品，例如智能手环、空气净化设备等，但集健康监测、环境监测、环境调节和AI分析于一体的综合性家庭健康解决方案仍处于快速发展的阶段。本系统通过低成本、高度集成的设计，展现了在家庭场景中的广泛适用性。而技术的飞速进步为系统的功能拓展提供了可能性。例如，MQTT协议使得数据传输更加高效,而AI技术的引入则提升了数据分析的智能化水平。

### 应用前景

在当前老龄化趋势加剧与健康意识日益增强的社会背景下，智能家庭健康监测系统在养老服务、健康保险以及智慧社区建设中展现出广阔的市场潜力。企业可以通过功能定制满足多样化的用户需求，该系统凭借其多功能性、成本低及高可扩展性，在家庭健康管理、环境优化以及智能化生活领域展现出显著的发展潜力。随着技术的持续进步与用户需求的日益多样化，该系统有望成为智能家居生态中的核心构成，为提升生活质量以及社会健康水平作出贡献。

# 智能家庭健康监测系统设计

## 智能家庭健康监测系统的功能需求分析

### 需求分析概述

智能家庭健康监测系统由三个核心部分组成：

1. 数据采集与控制终端：基于自制的PCB板，集成STM32F205RET6及多种传感器，负责采集健康数据和环境数据、处理数据并执行反馈控制。
2. 数据处理与显示终端：采用树莓派5运行嵌入式Linux操作系统，通过Qt框架实现数据的接收、存储、分析以及可视化展示。
3. 云服务器：基于阿里云服务器，运行MQTT Broker和MySQL数据库，支持数据上传和存储，同时结合百度AI模型进行分析。
4. 通过这三部分的协同工作，系统能够实现家庭成员健康监测、环境优化以及智能化管理。

### 外部接口需求

1. 用户接口需求

树莓派上的Qt客户端，提供图形化的界面，支持用户触摸操作。

1. 硬件接口需求
2. 自制的STM32开发板：

* IIC接口：MAX30102心率血氧传感器、GY906体温探头、AHT20温湿度、OLED显示屏
* USART接口：空气质量传感器、PM2.5传感器、ESP8266-01s
* ADC接口：LM2904噪声传感器
* PWM接口：直流电机风扇、舵机
* GPIO接口：LED、蜂鸣器、继电器、雾化器模块

1. 树莓派5：

* USB接口：麦克风、喇叭
* HDMI接口：7英寸触摸显示屏

1. 软件接口需求
   1. 嵌入式Linux操作系统：树莓派5运行ubuntu24.04
   2. Qt库：用于界面开发
   3. MySQL数据库：连接阿里云服务器的MySQL数据库
   4. MQTT客户端：连接阿里云服务器部署的Mosquitto
   5. C/C++语言：STM32和树莓派程序开发
   6. FreeRTOS支持库：STM32实时任务调度
2. 通信接口需求
3. IIC、USART、ADC、GPIO、PWM：传感器与STM32开发板进行通信
4. WIFI:STM32与阿里云服务器中的MQTT Broker进行通信
5. USB:树莓派与外设通信
6. MQTT协议：STM32与树莓派通过阿里云MQTT Broker通信
7. HTTP协议：百度AI模型交互

### 模块功能需求分析

1. 输入/输出数据
   1. 输入数据（STM32）：

* 健康数据：心率（次/分钟）、血氧饱和度（%）、人体温度（℃）
* 环境数据：TVOC浓度（ppm）、CO2浓度（ppm）、甲醛浓度（dug/m³）、PM2.5浓度（dug/m³）、噪声水平（db）、环境温度（℃）、环境湿度（%RH）
  1. 输出数据（树莓派）：
* 健康数据展示：心率、血氧、体温数值
* 环境数据展示：VOC、CO2、甲醛、PM2.5、噪声、温湿度数值
* 设备状态：风扇、雾化器、舵机、蜂鸣器
* AI分析结果：健康建议、对话

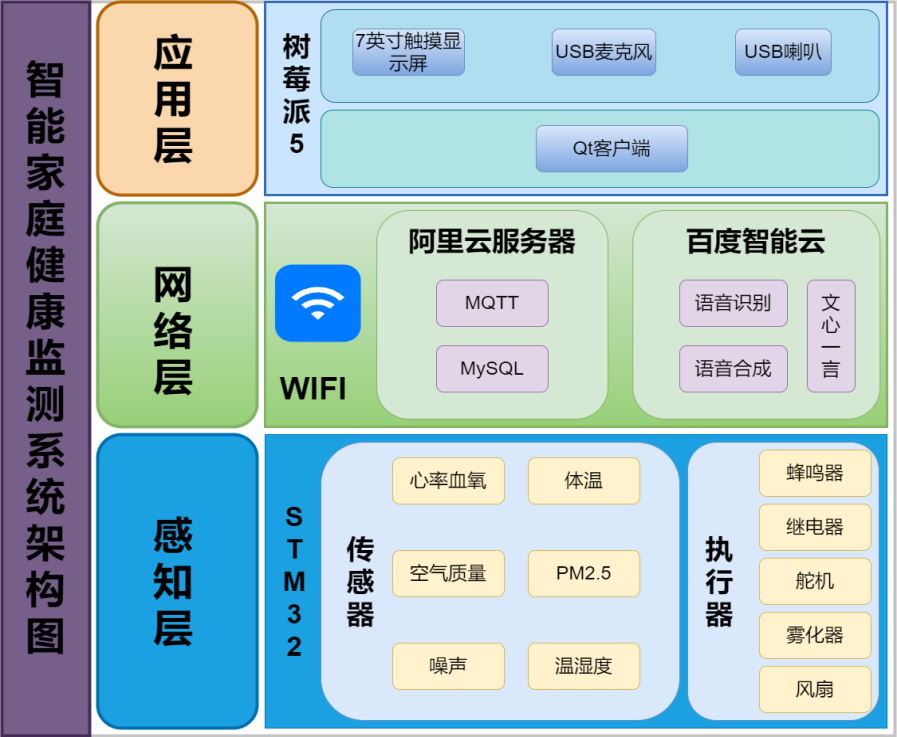
1. 功能描述

本系统的功能涵盖健康监测、环境控制和数据管理，具体如下：

1. 健康监测：通过MAX30102采集心率和血氧数据，GY906测量体温，实时显示于Qt界面。
2. 环境监测与自动调节：采集VOC、CO2、甲醛、PM2.5、噪声、温湿度数据，根据阈值自动控制。
3. 设备管理：通过Qt界面手动控制风扇，加湿器、舵机等的开关状态
4. 数据存储与查询：数据存储在MySQL数据库，支持历史记录查询并生成趋势图。
5. AI健康分析：通过HTTP协议上传至百度AI模型，生成健康建议并支持AI对话。
6. 多用户管理：支持多用户注册与登录，记录家庭成员信息如：姓名、年龄、性别、身高、体重，界面支持用户的切换。

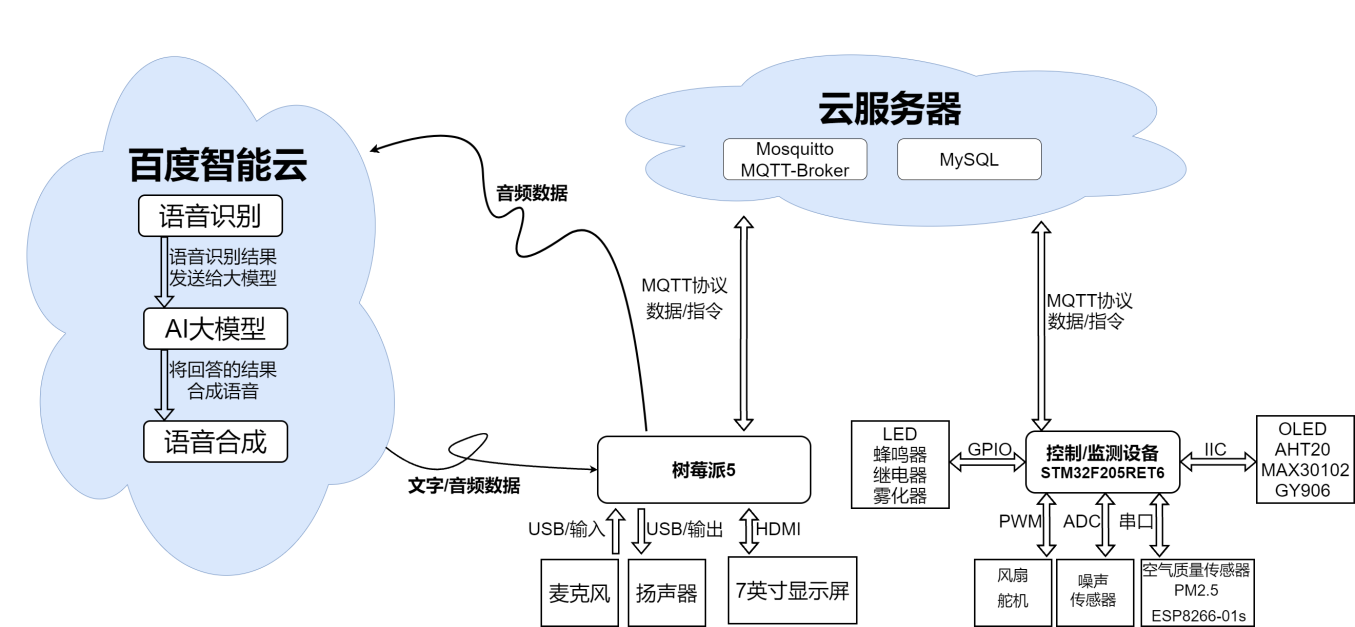
## 系统结构分析

本系统基于物联网技术，整合了感知层、网络层和应用层，以实现智能家庭健康监测系统的全面功能。感知层通过集成多种传感器，实时采集人体健康数据和环境参数，并接收来自应用层的控制指令，从而确保系统对物理环境的精准感知与动态反馈。网络层作为感知层与应用层之间的通信枢纽，利用WIFI技术进行数据传输，并依托TCP/IP协议栈实现高效且可靠的网络通信。应用层以树莓派作为硬件平台，运行基于Qt框架的用户界面软件，负责健康数据的可视化展示、用户交互操作以及智能控制指令的生成与下发。架构如图 3.1所示：

图 3.1智能家庭健康系统架构图

## 系统工作流程分析

智能家庭健康监测系统由控制/监测终端、客户端程序和云服务器三部分构成。控制/监测终端通过串口、IIC和GPIO等通信接口与各类传感器和控制器实现数据交互。客户端程序运行于树莓派，借助USB接口连接麦克风和喇叭以实现音频输入与输出，并通过WIFI或有线网络与云服务器及百度AI大模型建立连接，采用MQTT协议进行数据通信。云服务器基于阿里云平台，部署了MQTT服务和MySQL数据库；控制/监测终端通过WIFI模块，利用MQTT协议与云服务器进行高效通信。各个部分的详细通信结构如图 3.2所示：

图 3.2系统通信流程图

# 智能家庭健康监测系统开发

## 监测/采集终端开发

### 硬件设计

监测/控制终端是智能家庭健康监测系统的核心硬件部分，负责健康数据和环境数据的采集、初步处理以及设备控制。本硬件设计基于自制的PCB板，选用STM32F205RET6微控制器作为主控芯片，集成了多种传感器和执行器，通过合理的电路设计和接口配置，确保数据采集的准确性和控制的可靠性。

1. 电源部分

电源主要通过USB接口输入5V，经AMS1117-3.3线性稳压器输出3.3V，为STM32提供3.3V电压，为风扇，舵机等器件提供5V电压。加入10uF与100nF去耦电容滤除电源的噪声。通过CH340X芯片将USB信号转为串口信号方便调试。具体原理图如图 4.1和图 4.2所示：

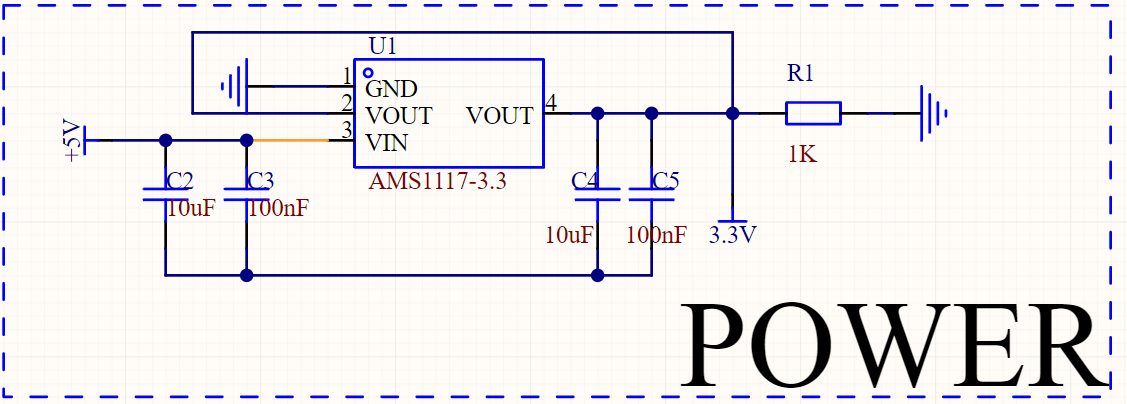
图 4.1电源模块

图 4.2 USB模块

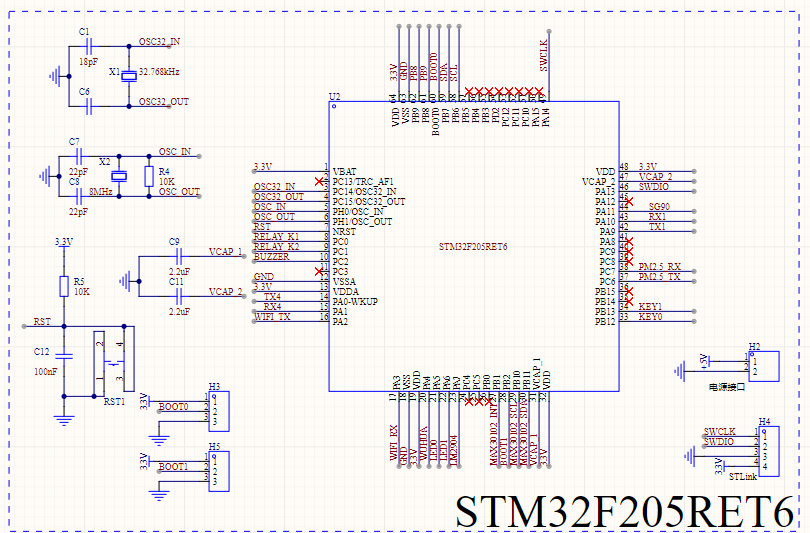
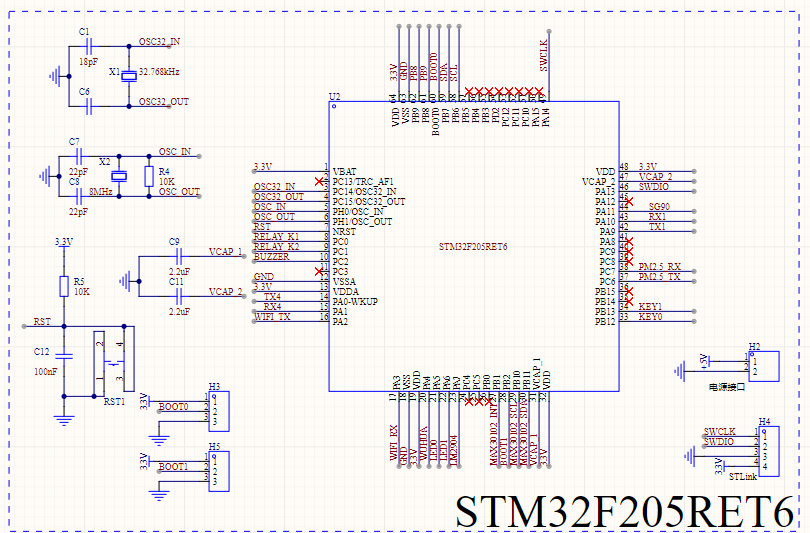
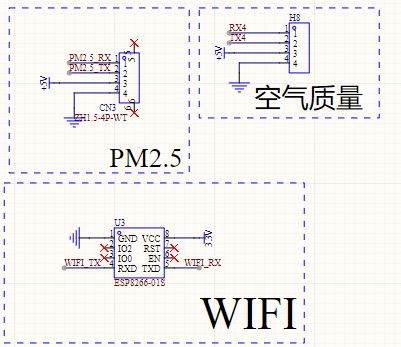
1. STM32F205RET6外围电路  
    这是一个基于STM32F205RET6微控制器的电路设计，STM32F205RET6是一款32位Cortex-M3微控制器，具有丰富的GPIO引脚、多通信接口和内部外设。原理图包括微控制器核心部件以及与其连接的外围电路，如电源管理、时钟电路、复位电路，调试下载电路等。如图 4.3所示：

图 4.3 STM32F205RET6外围电路

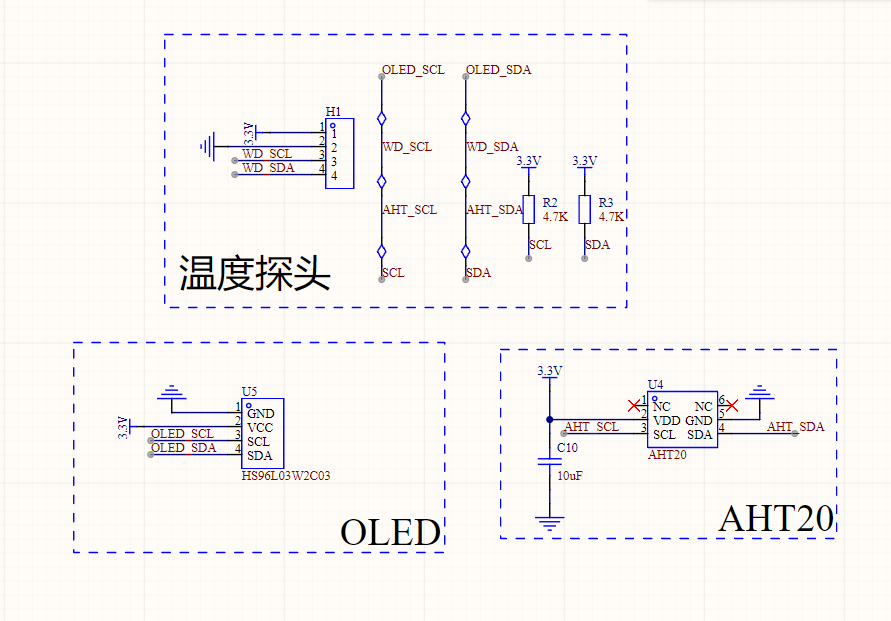
1. 串口电路

本设计中的串口电路主要用于实现微控制器与外部模块之间的数据通信。电路采用串行通信协议，通过UART接口完成数据的收发，图 4.4展示了PM2.5传感器、空气质量传感器与微控制器的串口连接，以及与WIFI模块与微控制器的串口连接，体现多设备间的串口扩展和数据交互功能。

图 4.4串口电路

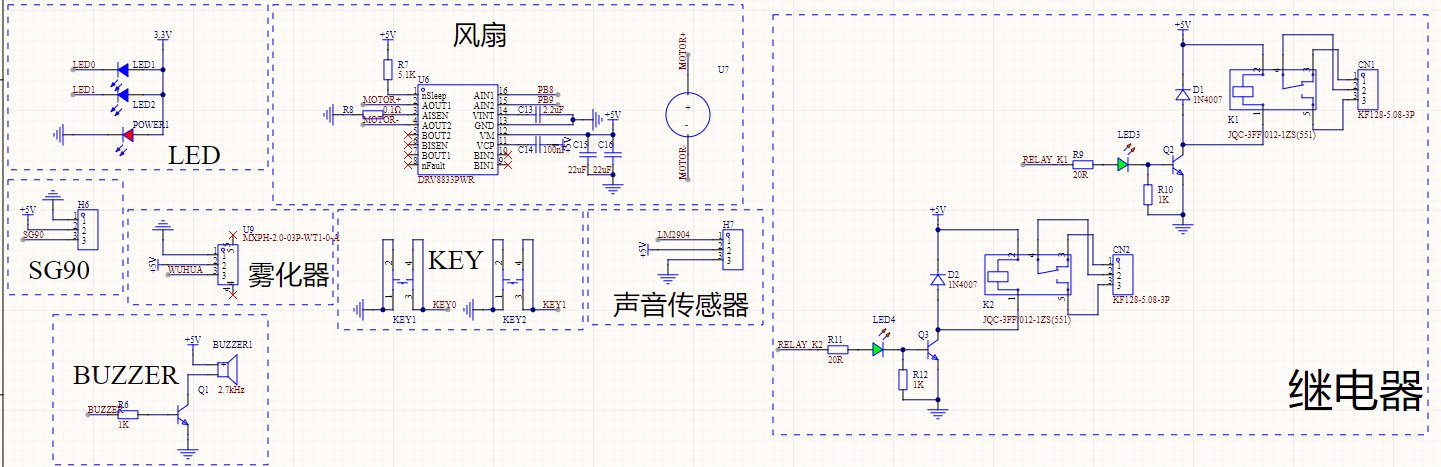
1. IIC电路

在本设计中，OLED 屏幕、AHT20 温湿度传感器以及体温传感器通过 IIC总线实现高效通信。设计充分利用了 IIC 总线的多设备扩展能力、简化的硬件连接以及低功耗特性，从而提升了系统的集成效率。电路中通过选用 4.7kΩ 上拉电阻并结合滤波元件，有效保证了通信的稳定性和可靠性。STM32 微控制器作为总线的主设备，而 OLED 屏幕、AHT20 温湿度传感器及体温传感器作为从设备，仅需利用 SDA（数据线）和 SCL（时钟线）两条信号线，便可实现对各传感器的灵活控制。具体连接如图 4.5所示：

图 4.5 IIC电路

1. GPIO口电路

在本设计中，按键、LED 指示灯以及蜂鸣器通过 STM32 微控制器的 GPIO 引脚实现高效控制。设计充分利用了 GPIO 接口的多功能性、简化的硬件连接以及低功耗特性，从而提升了系统的交互效率。电路中通过选用上拉电阻并结合限流电阻和三极管驱动，有效保证了信号的稳定性和驱动可靠性。STM32 微控制器作为控制的主设备，而按键、LED 指示灯及蜂鸣器作为从属模块，仅需利用 GPIO 引脚，便可实现对各输入输出设备的灵活控制。具体原理图如图 4.6所示：

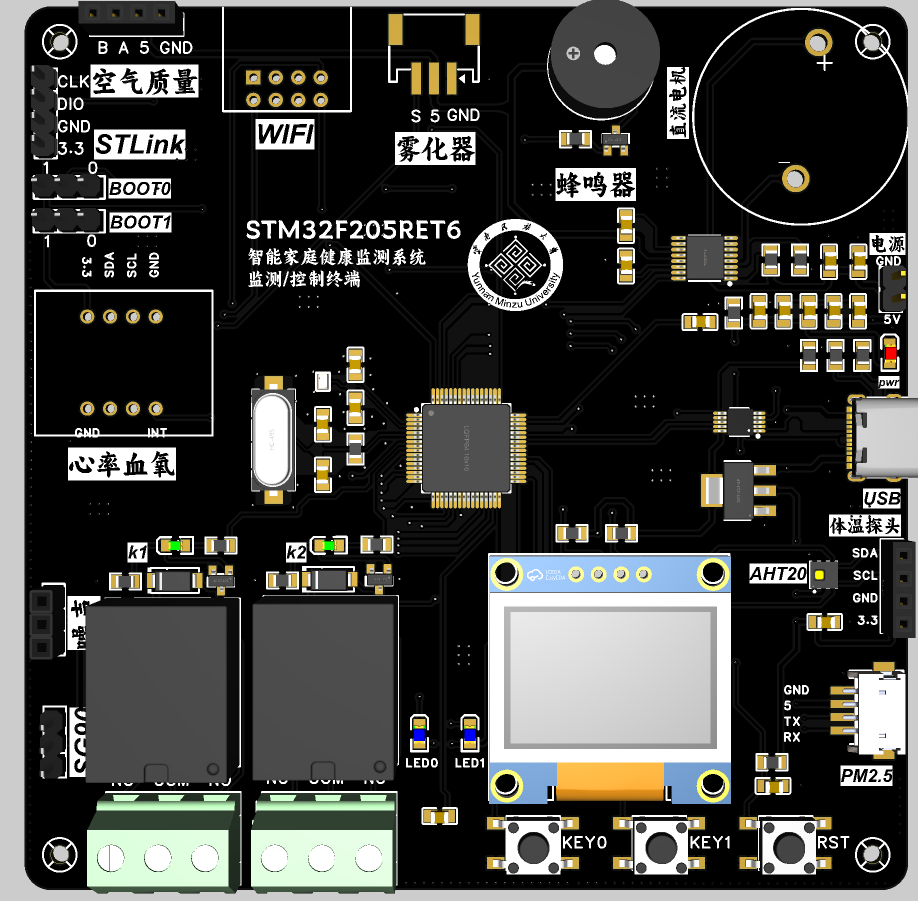
图 4.6 GPIO口电路

1. 传感器与执行器功能

在本系统中，PM2.5 传感器用于监测室内空气质量，AHT20 温湿度传感器负责采集环境温湿度数据，体温传感器则用于监测用户的体温状态，心率血氧传感器用于采集人体的心率血氧数据，共同构建了全面的健康与环境监测网络。采集到的数据通过 STM32 微控制器进行初步处理，并通过 OLED 屏幕实时显示，方便用户直观了解当前环境和健康状况。同时，系统通过蜂鸣器实现异常报警，例如 PM2.5 浓度超标或体温异常时发出提示音，通过继电器控制风扇或舵机调节室内环境，如启动通风或调整空气流向，从而实现智能化、自动化的家庭健康管理。

1. 硬件系统整体设计总结

本智能家庭健康监测系统的硬件设计以 STM32F205RET6 微控制器为核心，通过集成多种传感器，包括 PM2.5 传感器、AHT20 温湿度传感器、体温传感器，以及执行器，如风扇、舵机、蜂鸣器等，实现了健康数据和环境数据的实时采集、处理以及设备控制功能。设计通过合理的电路模块划分，例如电源、通信、控制模块，并结合 UART、IIC、GPIO 等接口配置，确保了数据采集的精确性和设备控制的可靠性。该硬件系统适用于智能家居、健康监测等场景，为家庭用户提供便捷的环境管理和健康监测服务，具有较高的工程应用价值和推广前景。本系统的PCB3D模型图和实物图如图 4.7和图 4.8所示：

图 4.7 PCB3D模型图

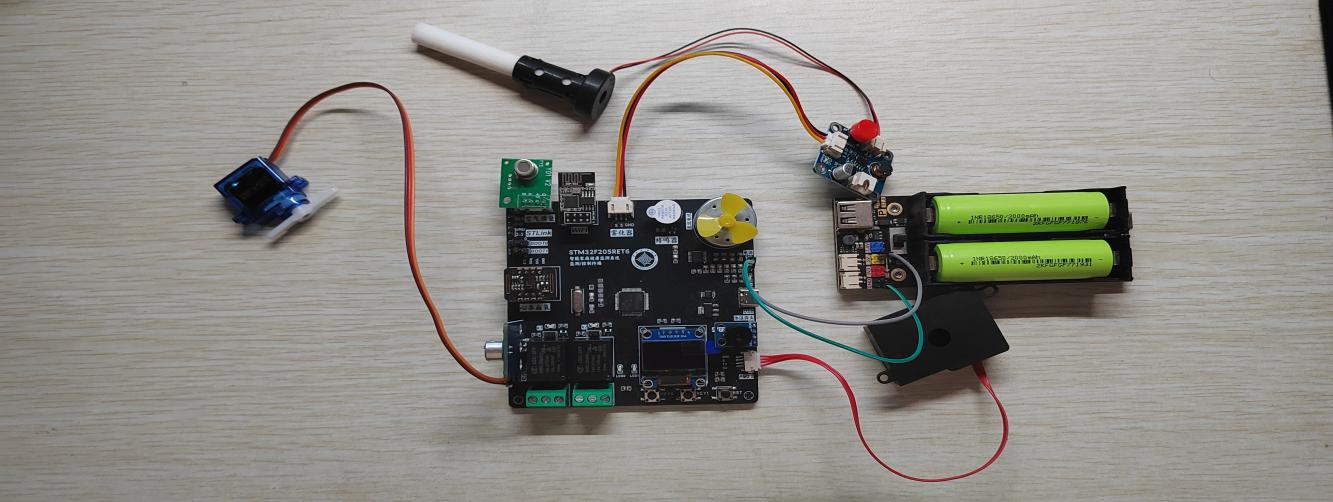


图 4.8实物图

### 软件设计

1. FreeRTOS实时操作系统移植

在单片机程序开发领域，开发方式主要分为裸机开发和基于实时操作系统的开发，两种方法各具优势与劣势。其核心区别在于，裸机开发过程较为直观，开发者直接通过操作寄存器进行编程，操作简便，但实时性较弱；相比之下，实时操作系统开发过程更为复杂，需管理任务调度和资源分配，然而其显著提升了系统的实时性能。裸机程序采用顺序执行的方式，而实时操作系统则通过任务调度实现并发执行。本设计中涉及多种传感器用于设备控制，且需实时接收和处理信号，为确保系统响应及时并提升用户体验，选用实时操作系统开发模式。

目前，单片机领域中流行的实时操作系统种类繁多，本文选用 FreeRTOS 作为开发平台。FreeRTOS 以其开源免费、开发流程简便等优势，受到广泛认可。然而，开发过程中无法直接使用官方提供的内核文件，需从官方网站下载相关文件，并对其进行移植与适配，以匹配本项目的开发板硬件环境。

FreeRTOS操作系统的移植步骤如下：

1. 从FreeRTOS官网下载最新的内核源码。
2. 在KEIL5种创建一个包含STM32F205RET6的基本配置的工程。
3. 将FreeRTOS源码中的内核文件task.c、queue.c等、内存管理文件heap\_4.c以及文件port.c和portmacro.h添加到工程中。
4. 修改FreeRTOSConfig.h文件，根据项目需求配置系统功能，例如设置任务调度方式、滴答定时器的频率、以及任务优先级的数量等。
5. 修改stm32f2xx\_it.c文件，确保SysTick中断和PendSV中断处理函数适配FreeRTOS的调度需求。
6. 在main.c中编写测试任务，调用vTaskStartScheduler启动调度器，验证移植是否成功。测试成功后，FreeRTOS移植完成，可直接使用系统提供的API函数如任务创建、信号量管理等进行后续软件的开发。
7. 软件系统架构分析

本系统的软件架构采用分层设计，分为APP层、操作系统层和驱动层，运行于STM32监测/控制终端，确保数据采集、处理、传输和设备控制的高效实现。以下结合图 4.8系统软件架构图进行分析各层功能及其交互。

1. APP层

APP层指基于FreeRTOS是实现的任务管理与功能调度，主要包括传感器数据采集任务、数据发送任务、数据接收任务和设备控制任务：

* 数据采集任务：通过IIC和UART接口，分别从MAX30102模块采集心率和血氧数据，从GY906探头采集体温数据，从AHT20传感器采集环境温湿度数据，从空气质量传感器采集TVOC、CO2、甲醛数据，从DC01传感器采集PM2.5数据，以及通过LM2904传感器采集噪声数据。
* 数据发送任务：利用MQTT协议，通过WIFI模块将采集到的数据发布到MQTT Broker,实现数据上云。
* 数据接收任务：通过MQTT协议订阅树莓派下发的控制指令，例如设备开关命令，完成远程控制。
* 设备控制任务：根据传感器数据分析结果和收到的指令，控制执行器动作。

1. 操作系统层

操作系统层基于FreeRTOS操作系统，负责多任务调度和资源管理：

* 任务调度：FreeRTOS通过抢占式调度机制管理多个任务，确保数据采集、处理和设备控制的实时性。例如，数据发送任务优先级较高，确保传感器数据集及时进行传输。
* 资源管理：通信信号量、消息队列等机制实现任务间的同步和通信，例如要使用消息队列将传感器采集到的数据发送给数据上传任务。

1. 驱动层

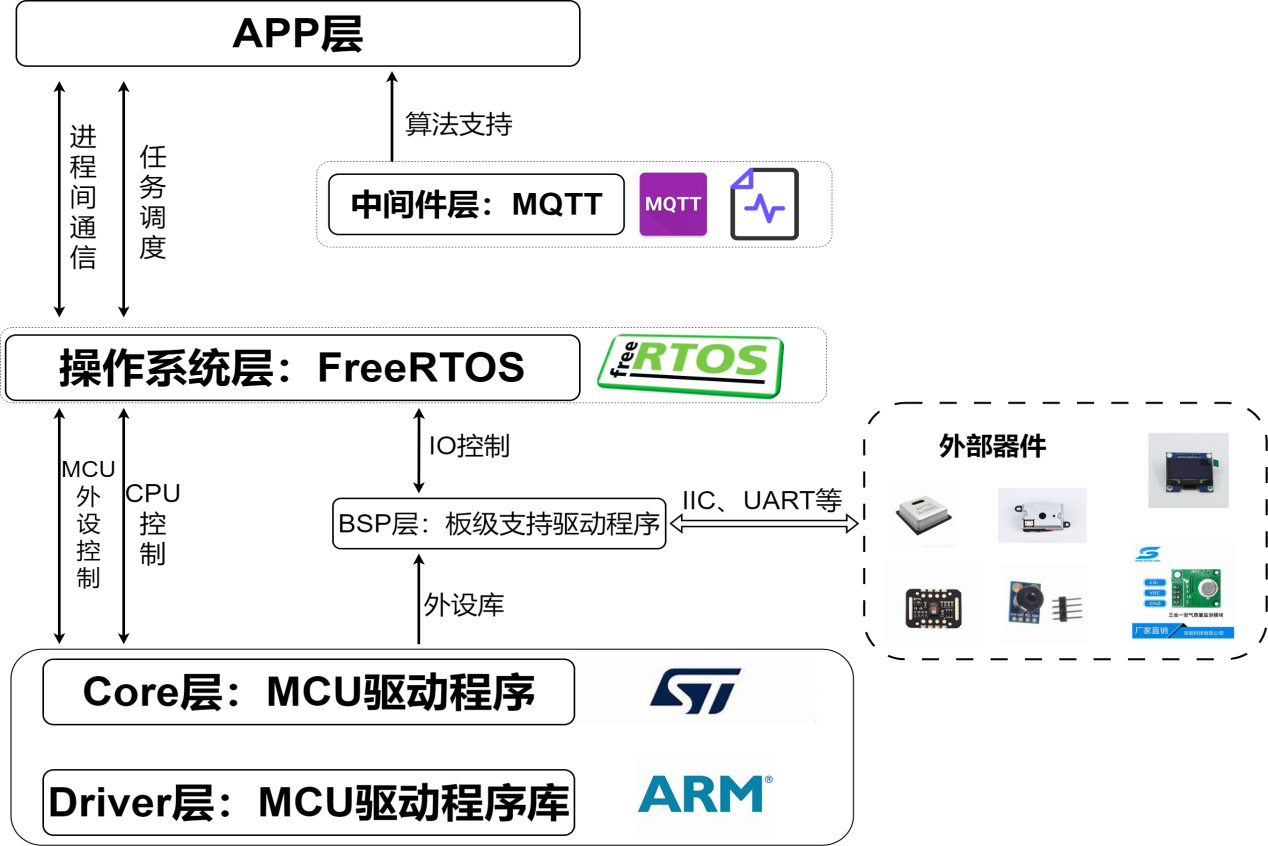
* 硬件驱动：包括IIC、UART、ADC、PWM、TIM、GPIO等驱动。
* 板级支持包BSP:基于STM32HAL库进行开发，HAL库提供了STM32 的外设初始化和操作函数，例如时钟配置、GPIO 配置、NVIC 中断管理等，为上层任务提供统一的硬件访问接口。

图 4.9系统软件架构图

1. 软件系统架构总结

系统的数据流从底层到上层依次传递：驱动层通过 IIC、UART 等接口采集传感器数据，传递至 FreeRTOS 层进行处理；FreeRTOS 层通过任务调度完成数据传输和任务间的同步；应用层根据采集数据或接收到的指令，调用驱动层接口控制风扇、舵机等设备，实现闭环控制并通过 MQTT 协议将数据发布至云端。本监测/控制终端的软件架构以 FreeRTOS 为核心，结合 STM32 硬件平台，实现了多传感器数据采集、设备控制和数据传输的完整功能。分层设计和任务调度机制确保了系统的实时性和稳定性，为智能家庭健康监测提供了可靠的软件支持。

### 监测/控制终端程序设计

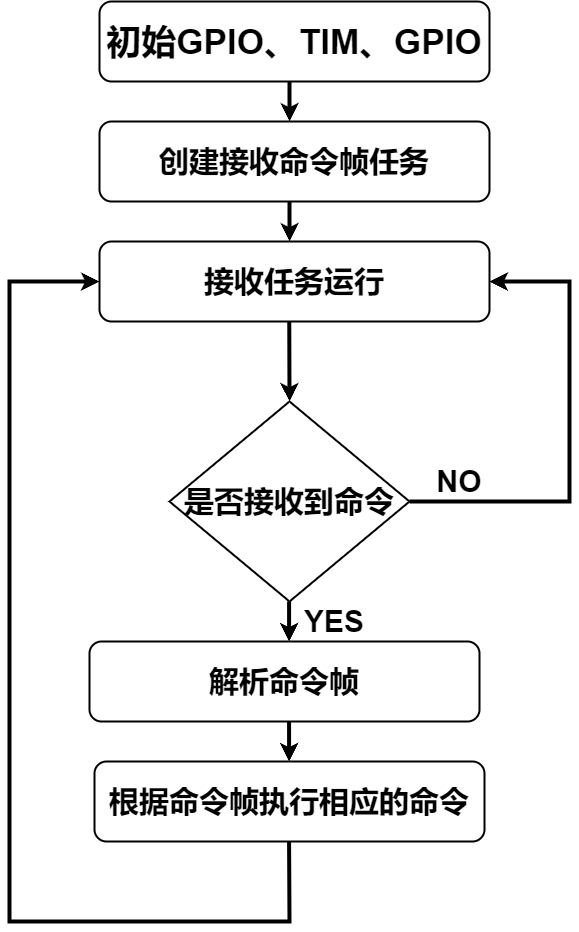
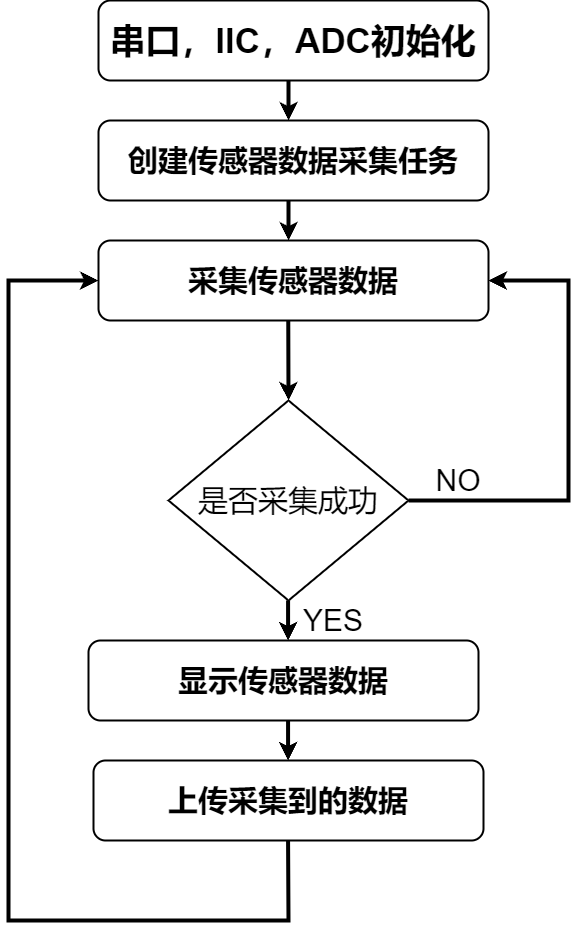
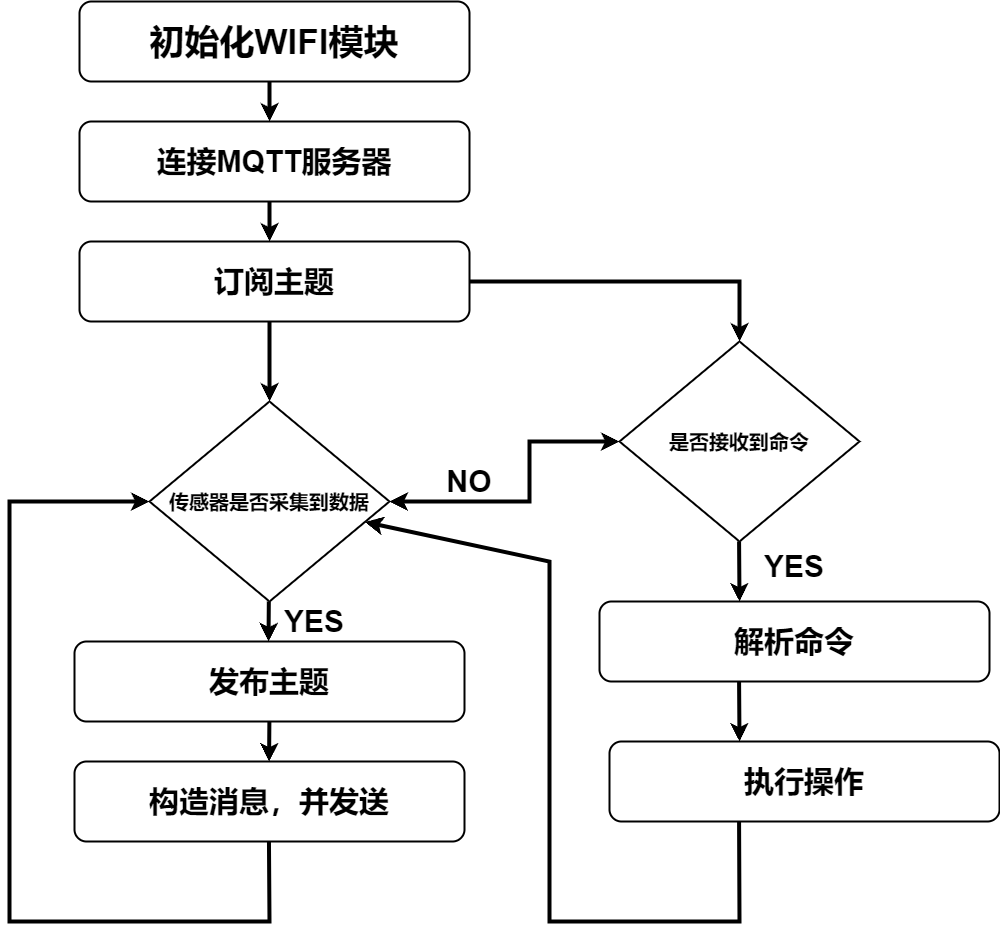
1. 程序设计流程图
2. 传感器数据采集流程如图 4.10所示。

图 4.10传感器数据采集流程图

1. 设备控制实现流程如图 4.11所示。

图 4.11设备控制流程图

1. MQTT通信流程如图 4.12所示。

图 4.12 MQTT通信流程图

1. 部分编码实现

在智能家庭健康监测系统的开发过程中，我采用了基于FreeRTOS实时操作系统的开发方式，通过创建多个任务并进行合理调度，以实现系统的核心功能。系统的设计目标是通过多任务并发执行，高效完成传感器数据采集、数据传输、设备控制以及状态提示等功能。为了满足实时性与资源管理的需要，我为每个任务分配了不同的堆栈大小和优先级，并利用信号量和消息队列实现任务间的同步与通信。以下是实现的详细说明：

1. 传感器数据采集任务：包括温湿度监测任务、空气质量监测、噪声监测、体温监测、心率血氧监测，负责从各自传感器采集数据。
2. 数据通信任务：包括数据发送任务和数据接收任务，负责通过MQTT协议与服务器进行数据交互。
3. 启动任务：作为系统的初始化任务，负责创建其它任务并在初始化完成后自我删除。

为了确保系统的高效运行，我根据任务的实时性要求和资源占用情况，设置了不同优先级的堆栈深度。例如，数据发送任务、数据接收任务和心率血氧监测任务被赋予较高的优先级，以保证数据的及时传输和关键健康数据的实时采集；而启动任务优先级较低，仅用于初始化。下面是部分代码展示：

/\* Includes ------------------------------------------------------------------\*/

#include "FreeRTOS.h"

#include "task.h"

#include "main.h"

#include "cmsis\_os.h"

/\* Private includes ----------------------------------------------------------\*/

/\* USER CODE BEGIN Includes \*/

#include "semphr.h"

#include "queue.h"

#include "list.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Task\_Include\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "app\_mqttSend\_task.h"

#include "app\_aht20\_task.h"

#include "app\_air\_task.h"

#include "app\_lm2904\_task.h"

#include "app\_gy906\_task.h"

#include "app\_mqttRev\_task.h"

#include "app\_max30102\_task.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*BSP\_Include\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "bsp\_usart\_driver.h"

#include "bsp\_aht20\_driver.h"

#include "bsp\_lm2904\_driver.h"

#include "bsp\_airquality\_driver.h"

#include "bsp\_oled\_driver.h"

#include "bsp\_led\_driver.h"

#include "bsp\_pm2.5\_driver.h"

#include "bsp\_sg90\_driver.h"

#include "bsp\_led\_driver.h"

#include "bsp\_relay\_driver.h"

#include "bsp\_buzzer\_driver.h"

#include "bsp\_wuhua\_driver.h"

#include "bsp\_esp8266\_driver.h"

#include "bsp\_drv8833\_driver.h"

#include "bsp\_max30102\_driver.h"

/\* USER CODE END Includes \*/

/\* Private typedef -----------------------------------------------------------\*/

/\* USER CODE BEGIN PTD \*/

/\* USER CODE END PTD \*/

/\* Private define ------------------------------------------------------------\*/

/\* USER CODE BEGIN PD \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Task\_Init\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*start\_task\*/

#define START\_TASK\_DEPTH 256

#define START\_TASK\_PRIORITY 1

TaskHandle\_t start\_task\_handle = NULL;

/\*mqtt\_send\_task\*/

#define MQTT\_SEND\_TASK\_DEPTH 1024

#define MQTT\_SEND\_TASK\_PRIORITY 27

TaskHandle\_t mqtt\_send\_handle = NULL;

/\*aht20\_task\*/

#define AHT20\_TASK\_DEPTH 256

#define AHT20\_TASK\_PRIORITY 24

TaskHandle\_t aht20\_task\_handle = NULL;

/\*air\_task\*/

#define AIR\_TASK\_DEPTH 256

#define AIR\_TASK\_PRIORITY 24

TaskHandle\_t air\_task\_handle = NULL;

/\*lm2904 task\*/

#define LM2904\_TASK\_DEPTH 256

#define LM2904\_TASK\_PRIORITY 24

TaskHandle\_t lm2904\_task\_handle = NULL;

/\*gy906 task\*/

#define GY906\_TASK\_DEPTH 256

#define GY906\_TASK\_PRIORITY 24

TaskHandle\_t gy906\_task\_handle = NULL;

/\*MQTT REV task\*/

#define MQTT\_REV\_TASK\_DEPTH 512

#define MQTT\_REV\_TASK\_PRIORITY 24

TaskHandle\_t mqtt\_rev\_task\_handle = NULL;

/\*max30102 task\*/

#define MAX30102\_TASK\_DEPTH 512

#define MAX30102\_TASK\_PRIORITY 26

TaskHandle\_t max30102\_task\_handle = NULL;

\* Private macro -------------------------------------------------------------\*/

/\* USER CODE BEGIN PM \*/

SemaphoreHandle\_t xAirQualitySemaphore;

SemaphoreHandle\_t xPM25Semaphore;

QueueHandle\_t xSensorDataQueue = NULL;

/\* USER CODE END PM \*/

/\* Private variables ---------------------------------------------------------\*/

/\* USER CODE BEGIN Variables \*/

/\* USER CODE END Variables \*/

/\* Definitions for defaultTask \*/

osThreadId\_t defaultTaskHandle;

const osThreadAttr\_t defaultTask\_attributes = {

.name = "defaultTask",

.stack\_size = 128 \* 4,

.priority = (osPriority\_t) osPriorityNormal,

};

void StartDefaultTask(void \*argument);

void MX\_FREERTOS\_Init(void); /\* (MISRA C 2004 rule 8.1) \*/

/\* Hook prototypes \*/

void vApplicationStackOverflowHook(xTaskHandle xTask, signed char \*pcTaskName);

/\* USER CODE BEGIN 4 \*/

void vApplicationStackOverflowHook(xTaskHandle xTask, signed char \*pcTaskName)

{

/\* Run time stack overflow checking is performed if

configCHECK\_FOR\_STACK\_OVERFLOW is defined to 1 or 2. This hook function is

called if a stack overflow is detected. \*/

printf("Stack overflow in task: %s\n", pcTaskName);

while (1);

}

void MX\_FREERTOS\_Init(void) {

defaultTaskHandle = osThreadNew(StartDefaultTask, NULL, &defaultTask\_attributes);

}

void StartDefaultTask(void \*argument)

{

/\* USER CODE BEGIN StartDefaultTask \*/

/\* Infinite loop \*/

/\*创建数据发送的消息队列 \*/

xSensorDataQueue = xQueueCreate(20, sizeof(SensorData\_t));

/\*创建信号量\*/

xAirQualitySemaphore = xSemaphoreCreateBinary();

xPM25Semaphore = xSemaphoreCreateBinary();

/\*创建任务\*/

/\*1. mqttsend\*/

xTaskCreate(

(TaskFunction\_t)mqtt\_send\_task,

(char \*)"mqtt\_send\_connect",

(configSTACK\_DEPTH\_TYPE)MQTT\_SEND\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)MQTT\_SEND\_TASK\_PRIORITY,

(TaskHandle\_t \*)&mqtt\_send\_handle);

/\*2. aht20 task\*/

xTaskCreate(

(TaskFunction\_t)aht20\_task,

(char \*)"aht20\_task",

(configSTACK\_DEPTH\_TYPE)AHT20\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)AHT20\_TASK\_PRIORITY,

(TaskHandle\_t \*)&aht20\_task\_handle);

/\*3. air task\*/

xTaskCreate(

(TaskFunction\_t)air\_task,

(char \*)"air\_task",

(configSTACK\_DEPTH\_TYPE)AIR\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)AIR\_TASK\_PRIORITY,

(TaskHandle\_t \*)&air\_task\_handle);

/\*4. lm2904 task\*/

xTaskCreate(

(TaskFunction\_t)lm2904\_task,

(char \*)"lm2904\_task",

(configSTACK\_DEPTH\_TYPE)LM2904\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)LM2904\_TASK\_PRIORITY,

(TaskHandle\_t \*)lm2904\_task\_handle);

/\*5. gy906 task\*/

xTaskCreate(

(TaskFunction\_t)gy906\_task,

(char \*)"gy906\_task",

(configSTACK\_DEPTH\_TYPE)GY906\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)GY906\_TASK\_PRIORITY,

(TaskHandle\_t \*)gy906\_task\_handle);

/\*6. revData task\*/

xTaskCreate(

(TaskFunction\_t)mqtt\_rev\_task,

(char \*)"revData\_task",

(configSTACK\_DEPTH\_TYPE)MQTT\_REV\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)MQTT\_REV\_TASK\_PRIORITY,

(TaskHandle\_t \*)mqtt\_rev\_task\_handle);

/\*7. max30102 task\*/

xTaskCreate(

(TaskFunction\_t)max30102\_task,

(char \*)"max30102\_task",

(configSTACK\_DEPTH\_TYPE)MAX30102\_TASK\_DEPTH,

(void \*)NULL,

(UBaseType\_t)MAX30102\_TASK\_PRIORITY,

(TaskHandle\_t \*)max30102\_task\_handle);

vTaskDelay(pdMS\_TO\_TICKS(100));

vTaskDelete(NULL);

}

## 控制/显示终端开发

智能家庭健康监测系统的控制与显示终端基于树莓派5开发，作为整个系统的核心管理与交互平台，集成了数据显示、设备控制以及用户交互等功能。在开发过程中，为了提高系统的灵活性和用户体验，通过区分登录账号来进行家庭成员的管理。

### 软件移植和环境配置

因为要在树莓派5中运行Qt图型界面程序，我们需要进行相关程序的移植，才能保证图形界面的正常运行。

1. Ubuntu24.10系统

树莓派中可以运行多种操作系统，如树莓派原生的操作系统RaspberryPiOS、Apertis、AlpineLinux、Ubuntu等操作系统，在本设计中，我选择了Ubuntu 24.10作为树莓派5的操作系统。之所以选择Ubuntu，是因为它拥有丰富的软件生态和庞大的社区支持，通过APT包管理器可以轻松安装Qt开发框架、MySQL客户端等工具，简化开发环境搭建，同时提供完善的桌面环境以支持Qt图形界面的流畅运行，且对ARM架构优化良好，兼容性与稳定性兼备，而其活跃的社区和技术支持也便于调试和优化。具体选用Ubuntu 24.10，则是因为它作为2024年10月发布的最新版本，集成了最新的Linux内核和软件包，能充分利用树莓派5的硬件性能，支持最新Qt版本，并通过Wayland显示服务器提升图形渲染效率，同时提供最新的安全补丁和驱动支持，确保系统在联网环境下的安全性，这不仅满足了智能家庭健康监测系统对实时性、可视化和交互性的高要求，也让我在开发过程中能够体验前沿技术并为未来优化奠定基础。移植步骤如下：

1. 准备一张32G的内存卡以及读卡器，下载树莓派镜像烧录器。
2. 打开镜像烧录器，选择树莓派5，选择ubuntu24.10操作系统下载至内存卡中。
3. 等待下载完成，将内存卡插入树莓派5中。
4. 启动树莓派，只要能进行ubuntu的桌面则移植成功。
5. Qt2.15.13版本移植

移植步骤如下：

1. sudo apt-get install qtbase5-dev qtbase5-dev-tools qtchooser

安装默认的Qt库。

1. sudo apt-get install qtcreator安装QtCreator。
2. sudo apt-get install build-essential sudo apt-get install libgl1-mesa-dev sudo apt install libfmt-dev sudo apt install liblog4cplus-dev sudo apt install libqt5sql5 libqt5sql5-psql libqt5sql5-mysql 安装Qt程序运行的依赖。
3. 测试Qt程序，只要能运行程序就代表Qt移植完成。
4. MQTT的Qt版本移植

由于程序的开发需要支持MQTT协议但是Qt的官方库中并没有MQTT库，所以需要自己进行移植。

1. 进入GitHub下载对应版本的MQTT库的源码。
2. 用Qt打开然后编译即可。
3. 将编译好的库文件移植到Qt的相关目录下。

到此为止，针对树莓派5的所有移植工作均已顺利完成，为后续的Qt应用程序开发奠定了坚实的基础。在嵌入式开发中，通常需要在性能更强的PC（X86架构）上使用交叉编译工具，将代码编译为目标设备（ARM架构）的可执行文件，这是因为许多嵌入式设备的计算资源有限，内存和处理器性能不足以支持直接运行复杂的开发环境和编译过程。然而，得益于树莓派5配备的大容量内存和卓越的硬件性能，我们能够直接在树莓派上安装功能强大的Qt Creator集成开发环境，并利用其原生编译功能直接完成程序的开发与构建，而无需依赖繁琐的交叉编译工具来进行交叉编译操作。这种方式不仅简化了开发流程，提高了工作效率，还充分发挥了树莓派5的高性能优势，确保了开发环境的稳定性和程序运行的流畅性，为智能家庭健康监测系统的图形界面开发提供了极大的便利。

### Qt软件程序开发

1. 软件功能分析

确定控制/显示终端负责的功能，进行Qt程序的设计。目前本Qt程序拟实现传感器数据的显示、设备的远程控制、健康分析、语音助手和用户管理等功能。

传感器数据显示和设备远程控制的实现，终端连接到MQTT服务器，订阅STM32发布的主题，进行传感器数据的解析，并在Qt界面上进行实时显示。设备的远程控制通过Qt发布的主题，STM32进行订阅，就可以实现对设备的远程控制。

健康数据分析通过 HTTP 请求上传至百度 AI，AI 返回分析结果后，终端解析并显示在 Qt 界面上，提供健康评估报告和优化建议。

语音识别功能实现人机交互。终端通过麦克风录音，并向百度语音识别 API 发送 HTTP 请求，获取识别结果。

用户管理功能支持账号注册、登录及健康数据存储，所有用户信息和健康数据存储于 MySQL 数据库，支持多用户管理。

确定各项功能之后，设计主程序 UI 界面，Home 界面用于实时数据显示和设备控制，Report 界面用于 AI 健康分析和语音交互，User 界面用于用户管理和历史数据查询。最后，按照以上功能流程进行编码和测试，完成软件开发。

1. 程序设计流程图
2. 登录/注册界面功能实现流程如图 4.13所示。



图 4.13登录/注册界面功能流程图

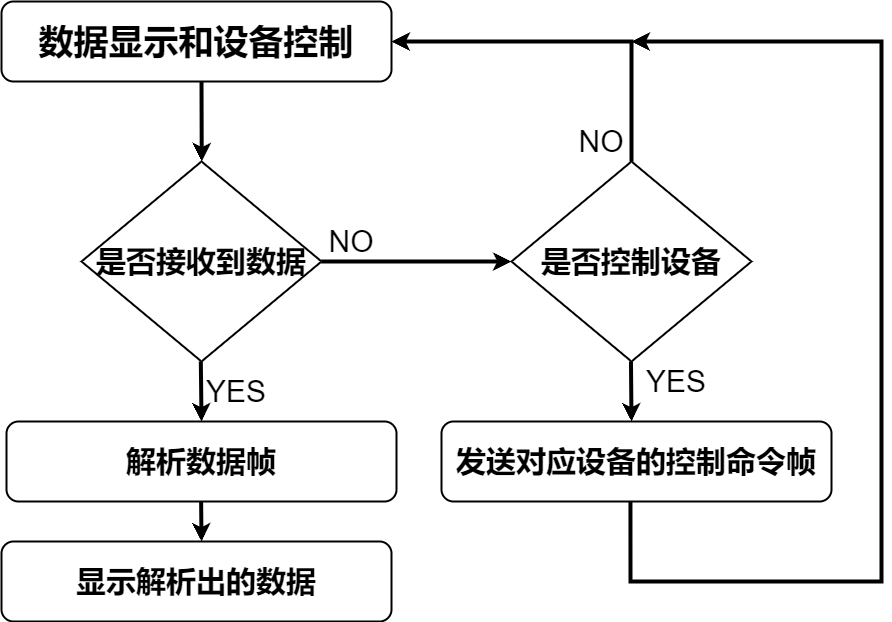
1. 传感器数据显示和设备控制功能的功能流程如图 4.14所示。

图 4.14传感器数据显示和控制流程图

1. 语音对话和建议功能流程如图 4.15所示。

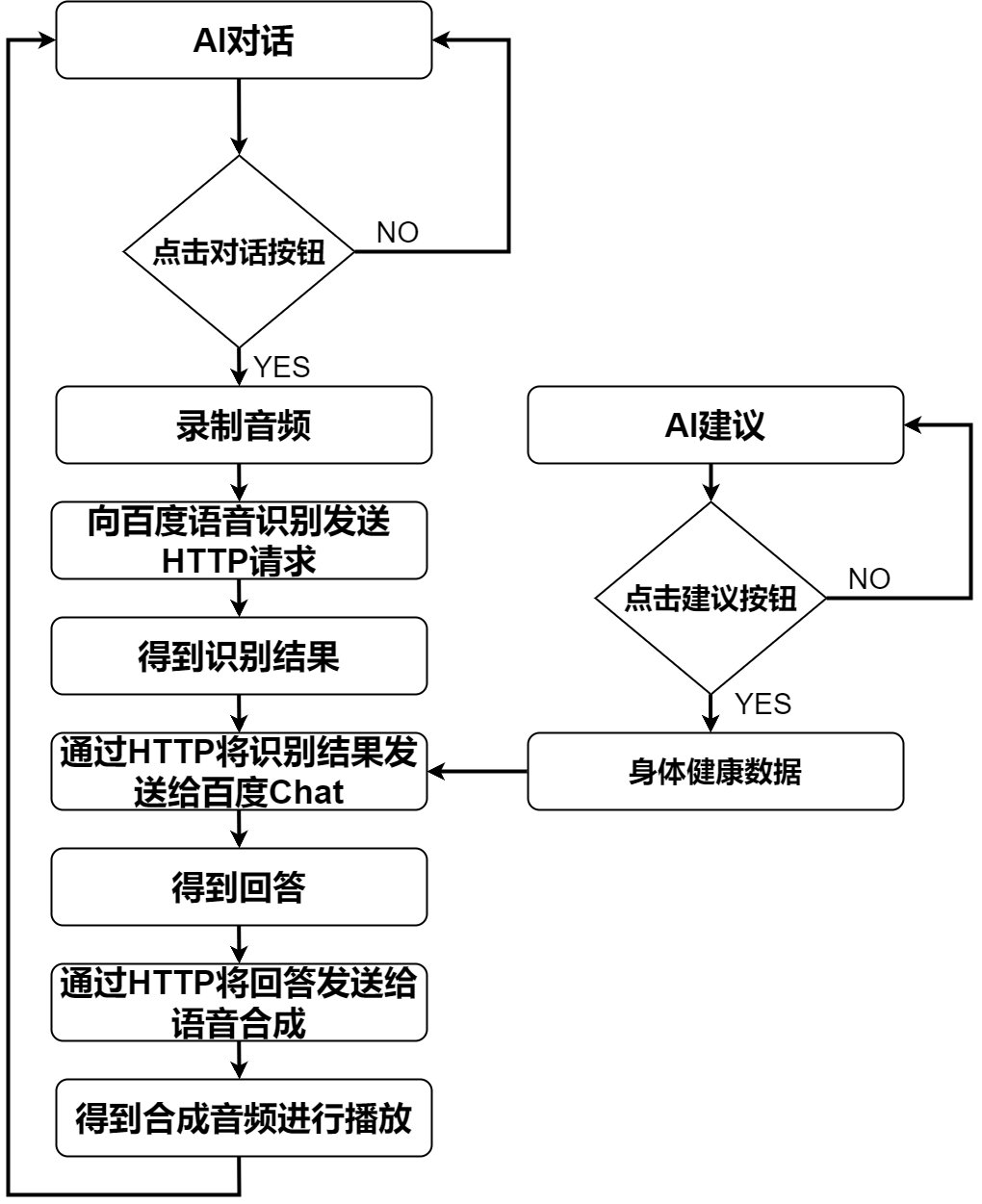


图 4.15语音对话和建议功能流程图

1. 程序UI界面设计

Qt界面程序由自主设计实现，具体功能展示如下所示。



图 4.16登录界面



图 4.17注册界面



图 4.18Home界面



图 4.19报告界面



图 4.20User界面

1. 部分编码实现

控制/显示终端的核心代码为MQTT通信，MYSQL数据库的插入、查询，百度AI的HTTP请求等,编程语言使用C++进行编程。主要展示该部分：

class mqtt : public QWidget

{

Q\_OBJECT

enum MessageQoS{

m\_QoS0,

m\_QoS1,

m\_Qos2

};

public:

explicit mqtt(QWidget \*parent = nullptr);

~mqtt();

// 连接mqtt服务器

void mqtt\_connect(void);

void Subscribe(const QString &topic,int qos);//订阅消息

int Publish(const QString &topic,const QString &message,int qos);//发布消息

QString get\_mqttValue(QJsonDocument JsonDocument, QString Key);/\*分离json数据\*/

private slots:

/\*mqtt连接成功\*/

void mqtt\_connect\_success(void);

public:

QString Sub\_Topic = "smart\_home/publish"; //订阅

QString Pub\_Topic = "smart\_home/subscribe";//发布

QEventLoop loop;

//设消息等级

enum MessageQoS m\_MessageQos = m\_QoS1;

signals:

};

namespace Ui {

class health\_report;

}

class health\_report : public QWidget

{

Q\_OBJECT

public:

explicit health\_report(QWidget \*parent = nullptr);

~health\_report();

/\*获取百度chat 和 语音识别的 token\*/

void baidu\_http\_get\_token(void);

public slots:

/\*请求结束 返回结果解析\*/

void http\_finished(QNetworkReply \*Reply);

/\*将音频上传到百度云识别\*/

void baidu\_Audio\_Send(void);

/\*将语音识别完成的文字发送给Chat\*/

void baidu\_Chat\_Send(QString result);

/\*将百度Chat的回答上传 进行语音合成\*/

void baidu\_AudioOut\_Send(QString text);

/\*暂停录音\*/

void on\_autoStopAudio(void);

private slots:

/\*开始对话按钮\*/

void on\_pushButton\_AiChat\_clicked();

/\*停止对话按钮\*/

void on\_pushButton\_stopChat\_clicked();

/\*健康建议按钮\*/

void on\_pushButton\_AiReport\_clicked();

private:

Ui::health\_report \*ui;

// get请求返回的token

QString Audio\_access\_token = "";

QString Chat\_access\_token = "";

QNetworkAccessManager\* baidu\_Audio = nullptr;

QNetworkAccessManager\* baidu\_Chat = nullptr;

// 音频格式

QAudioFormat Audio\_Format;

QAudioInput \*Audio\_in = nullptr;

// 音频文件处理

QFile destinationFile;

QFile \*sendFile;

// 结束聊天的标志变量

bool shouldStopChat = false;

};

public:

explicit health\_home(QWidget \*parent = nullptr);

~health\_home();

//查询sql数据库中的信息

void sql\_body\_information(void);

//计算BMI值

double computeBMI(double height,double weight,QString\* BMI\_R);

void getSysTime(void);//获取系统时间

private slots:

void time\_reflash(void);//时间刷新

/\*接收消息\*/

void MQTT\_RevData\_Success(const QByteArray &message);

/\*雾化器按钮\*/

void on\_checkBox\_wuhua\_toggled(bool checked);

/\*继电器按钮\*/

void on\_checkBox\_door\_toggled(bool checked);

/\*舵机按钮\*/

void on\_checkBox\_window\_toggled(bool checked);

/\*风扇按钮\*/

void on\_checkBox\_fenshang\_toggled(bool checked);

private:

void setupChart(); // 初始化折线图

private:

Ui::health\_home \*ui;

/\*创建mqtt类\*/

mqtt my\_mqtt;

// 获取时间

QString myData;

QString myTime;

QChart \*chart; // 合并后的折线图

QLineSeries \*hrSeries;

QLineSeries \*spo2Series;

const int maxPoints = 60; // 最大显示60个数据点（例如60秒的数据）

};

## 智能家庭健康监测系统服务器搭建

本文所使用的服务器是由作者购买的阿里云服务器，并在该服务器上部署了MQTT服务和MySQL服务。下面将对各个部分进行详细解析。

### MQTT服务器的搭建

1. MQTT简介

MQTT（Message Queuing Telemetry Transport，消息队列遥测传输协议）是一种采用发布/订阅（publish/subscribe）模式的轻量级通信协议，基于 TCP/IP 协议构建。它的主要优势在于，只需少量的代码和较小的带宽，就能为远程设备提供实时、稳定的消息传输服务。作为一种低资源消耗、低带宽需求的即时通信协议，MQTT 在物联网、小型设备和移动应用等领域得到了广泛应用。其简单、轻便、开放和易于部署的特点，使其适用范围极为广泛。

MQTT 服务器通常被称为 MQTT Broker，目前常见的版本包括 EMQ、HiveMQ、VerneMQ、ActiveMQ 和 Mosquitto 等。前三个版本既有开源版本也有商业版本，适合我们选用开源版进行学习；而后两者为完全开源。本文选择使用 Mosquitto 版本，由 Eclipse 开源社区开发，其功能足以应对大多数使用场景。接下来，我们将在服务器上部署一个属于自己的 MQTT 服务器。

1. 部署步骤

步骤一：登录阿里云服务器后，输入指令安装Mosquitto MQTT服务器。

步骤二：安装完成后，启动 Mosquitto 服务，启动后系统将运行 Mosquitto 的 MQTT Broker。

步骤三：启动命令执行后，如果成功，会显示 started successfully! 的输出信息，这表明 Mosquitto 服务已正常运行。

1. MySQL数据库部署

对于 MySQL 服务的安装，可以直接在服务器上通过 Linux 命令进行部署。  
安装步骤如下：

步骤一：运行 sudo apt-get update 命令更新系统软件包列表。  
 步骤二：通过 sudo apt-get install mysql-server 命令安装 MySQL 服务。  
 步骤三：执行 sudo mysql\_secure\_installation 命令进行初始安全配置。  
 步骤四：输入 systemctl status mysql.service 命令，查看 MySQL 服务是否正常运行。  
 步骤五：定位到 MySQL 安装目录下的配置文件 mysqld.cnf，调整相关参数以支持远程连接访问。  
 步骤六：从远程主机尝试连接，验证连接是否成功。  
 如果远程主机能够顺利访问服务器上的数据库，则说明配置已完成。因为本系统的数据表比较简单，只有用户表存储家庭成员的信息，所以就不进行赘述。

# 智能家庭健康监测系统测试

## 硬件测试

在硬件测试部分，需要验证供电情况是否正常，例如检查是否存在稳定的 5V 电压和 3.3V 电压输出；同时需确认晶振是否能够正常启动并振荡；此外，还要检查电路板上是否存在虚焊等焊接质量问题，以确保硬件的可靠性和稳定性。实物电路板如图 5.1所示。

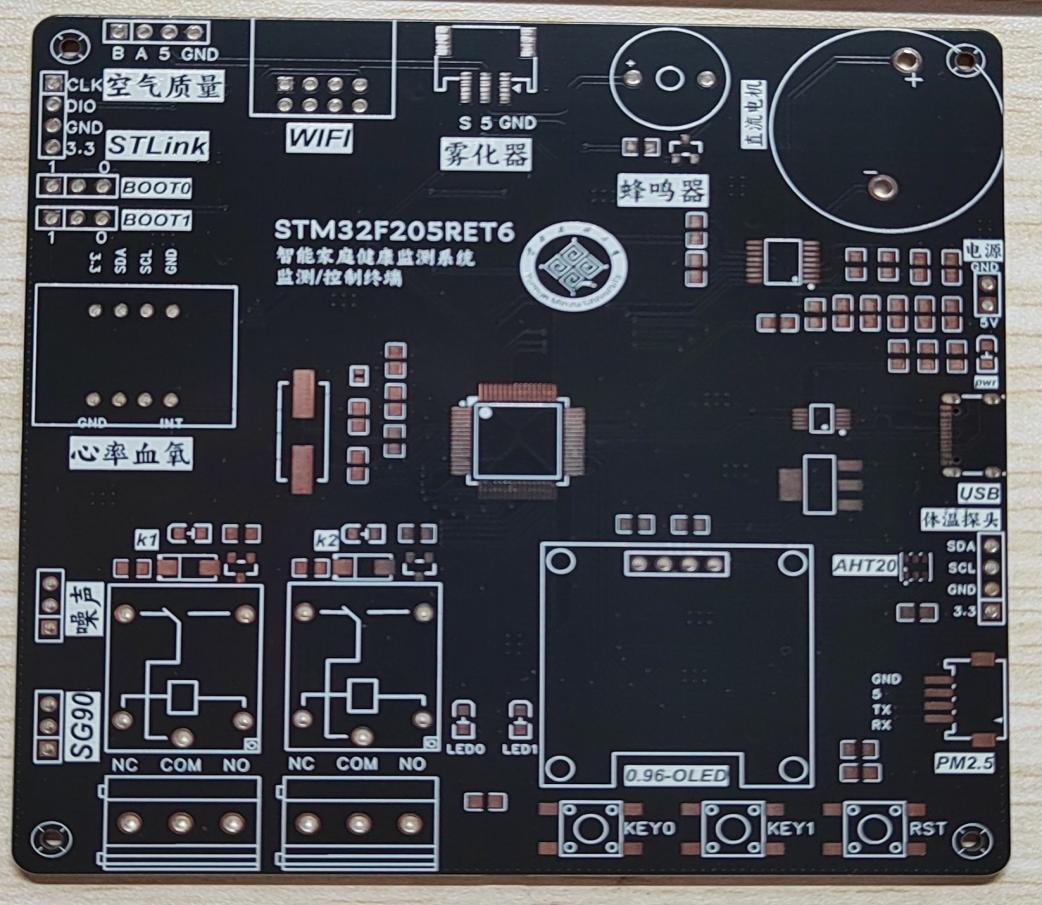


图 5.1智能家庭健康监测系统电路板

1. 硬件电路焊接完成后进行电压测试，确保整个系统的供电正常。如图 5.2和图 5.3所示。

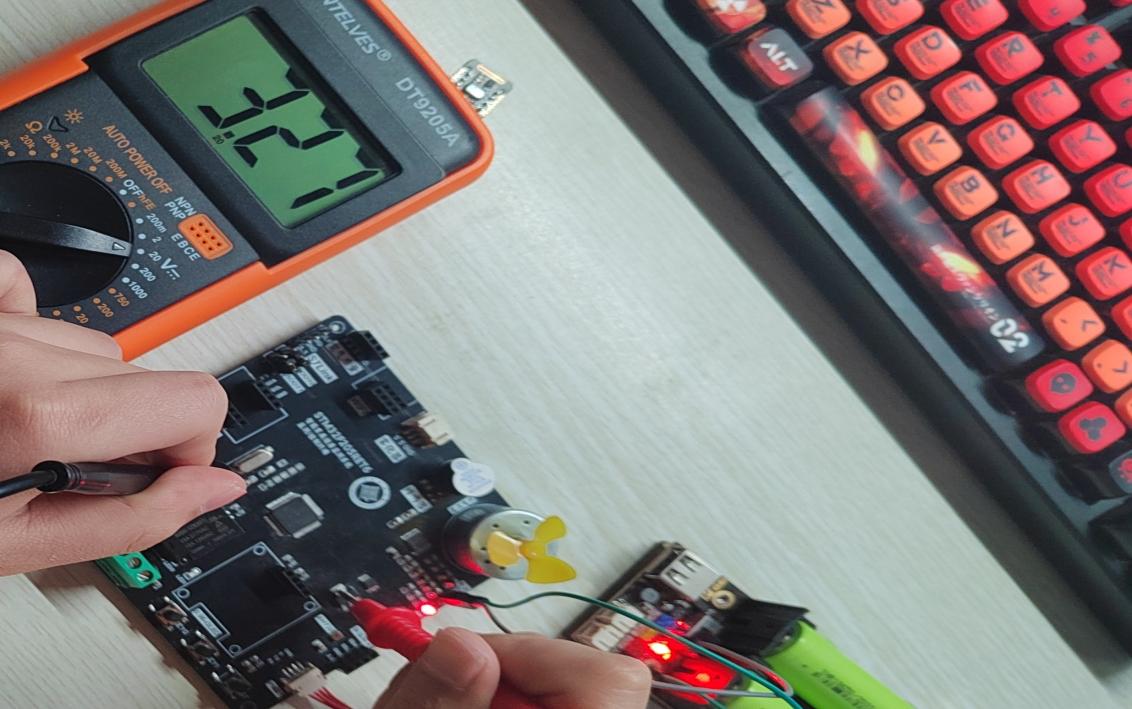


图 5.2 3.3V电压测试

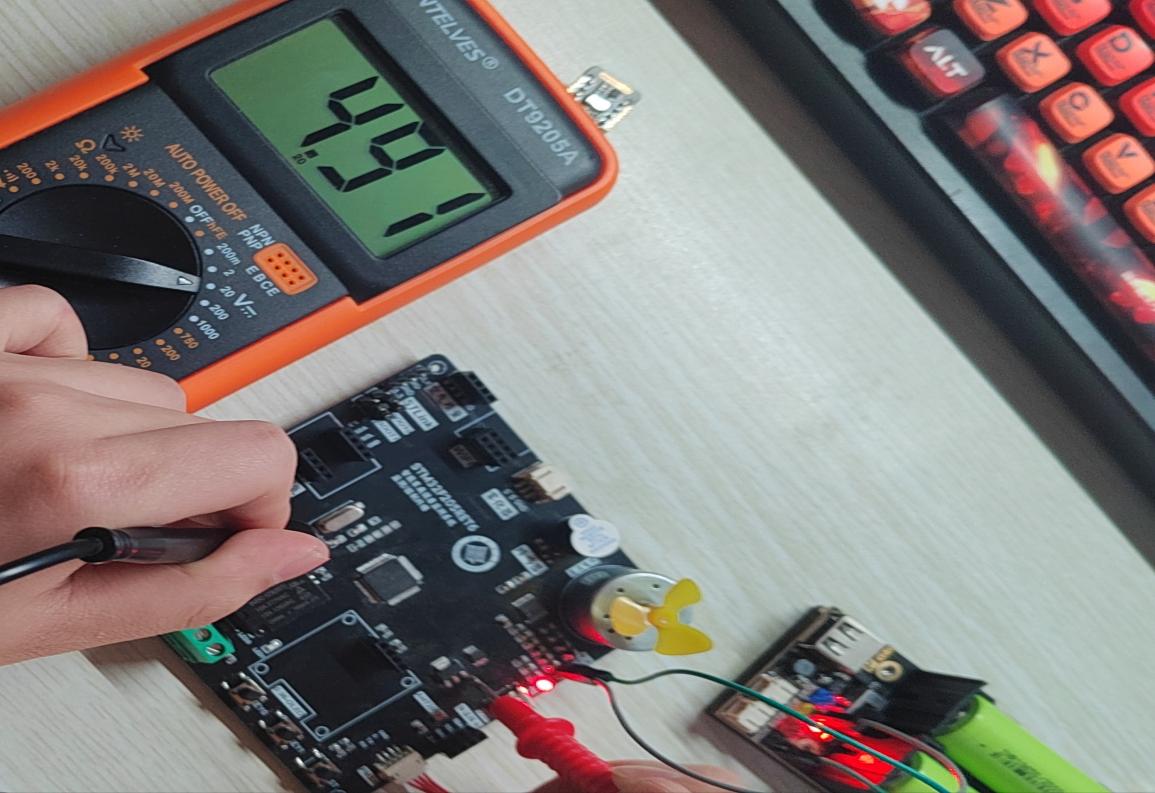


图 5.3 5V电压测试

1. 本系统采用的是外部8MHz的晶振，要确保外部晶振启振，验证如图 5.4所示为8MHz。

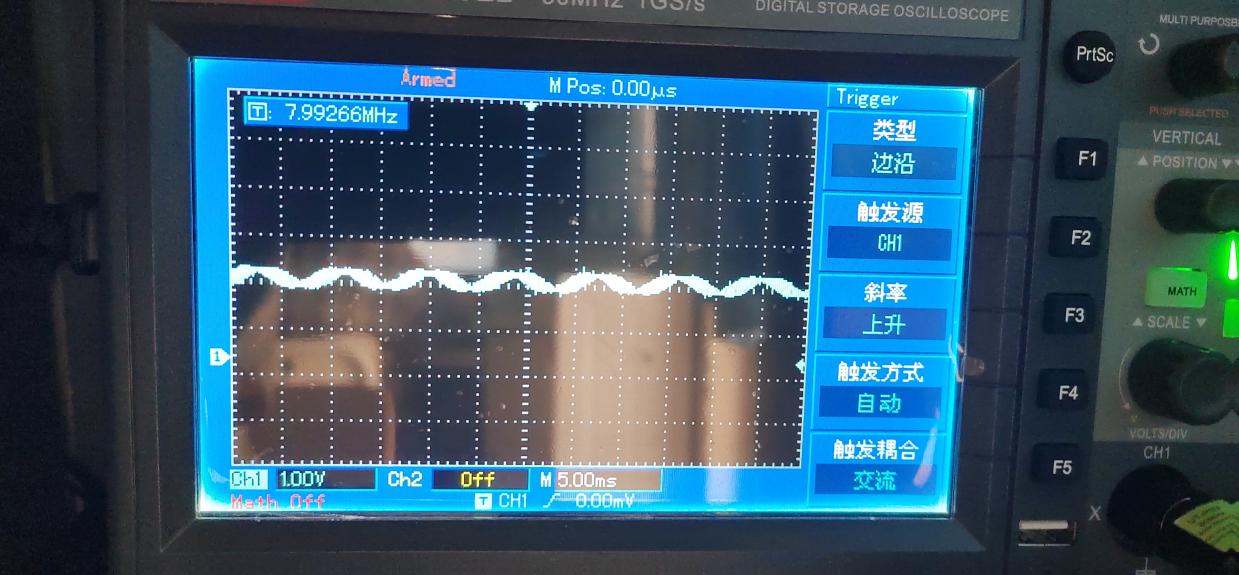


图 5.4外部晶振测试

## 软件编译

### 单片机程序编译

单片机程序的开发采用了 KEIL5 和 STM32CubeMX 两种工具进行协同设计，其中 STM32CubeMX 用于生成初始化的代码框架，而具体的 STM32 程序则基于 HAL 库进行编写，以简化开发过程并提高代码的可移植性。程序编写完成后，利用 KEIL5 自带的集成开发环境进行编译，检查代码的语法和逻辑是否正确。编译通过后，通过 STLink 调试工具将程序下载到单片机中，确保程序能够正确加载并运行于目标硬件上。

### Qt程序的编译

Qt 程序的开发过程首先在 Windows 系统中进行，利用 Qt Creator 集成开发环境编写代码。在 Windows 环境下，开发者可以充分利用其便捷的调试工具和图形化界面，对程序的功能进行全面的测试和优化，确保代码逻辑正确、界面交互流畅以及各项功能正常运行。测试成功后，将完整的源代码通过文件传输工具发送至树莓派 5 设备上。在树莓派 5 的 Linux 环境中，需先安装必要的 Qt 开发库和编译工具，然后对源码进行适配性调整，接着使用命令行或构建工具进行编译，生成适用于树莓派硬件的可执行文件。编译完成后，直接在树莓派 5 上运行程序，验证其在目标平台上的性能和稳定性，确保程序能够正常工作并满足设计需求。整个过程从开发到部署，充分结合了 Windows 的开发便利性和树莓派的硬件优势。

## 系统功能测试

### 系统通信功能测试

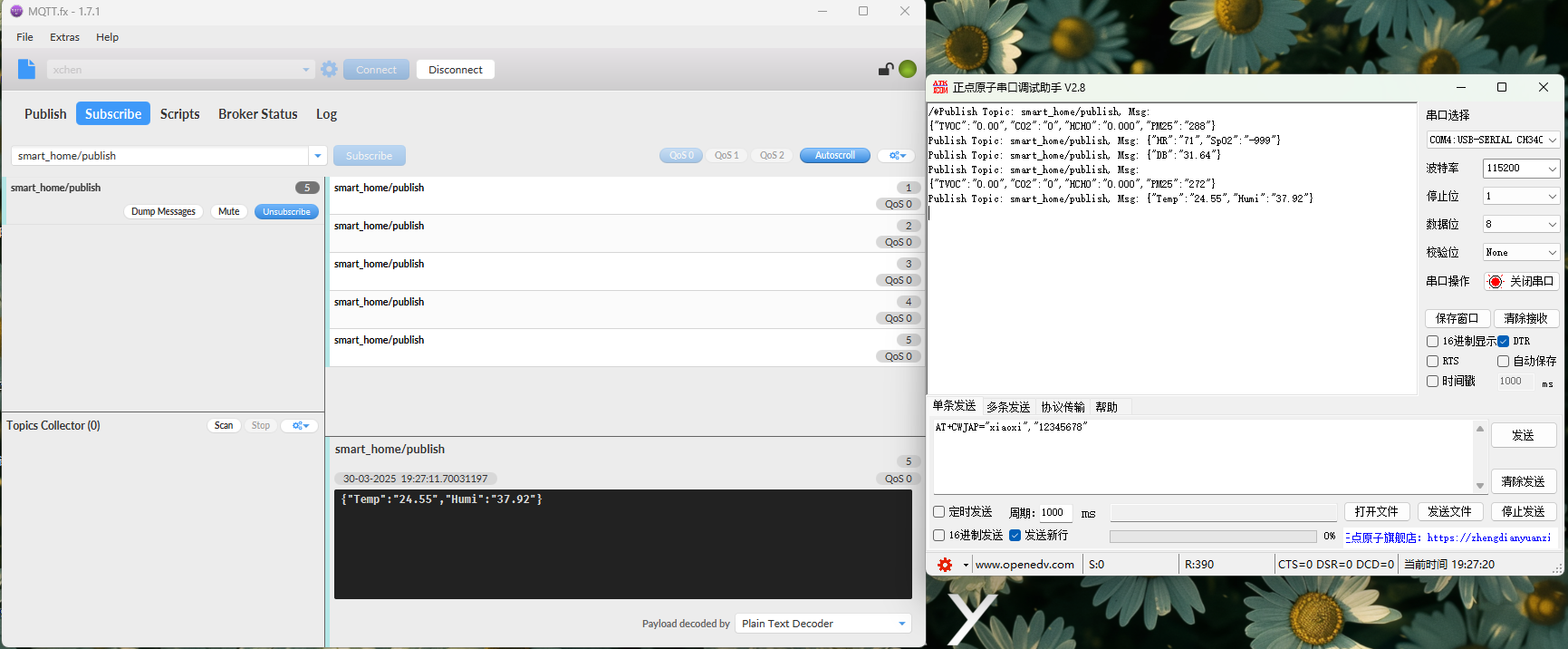
1. 监测/采集终端数据上传测试

图 5.5监测采集/终端数据上传测试

1. 控制/显示终端指令下发测试

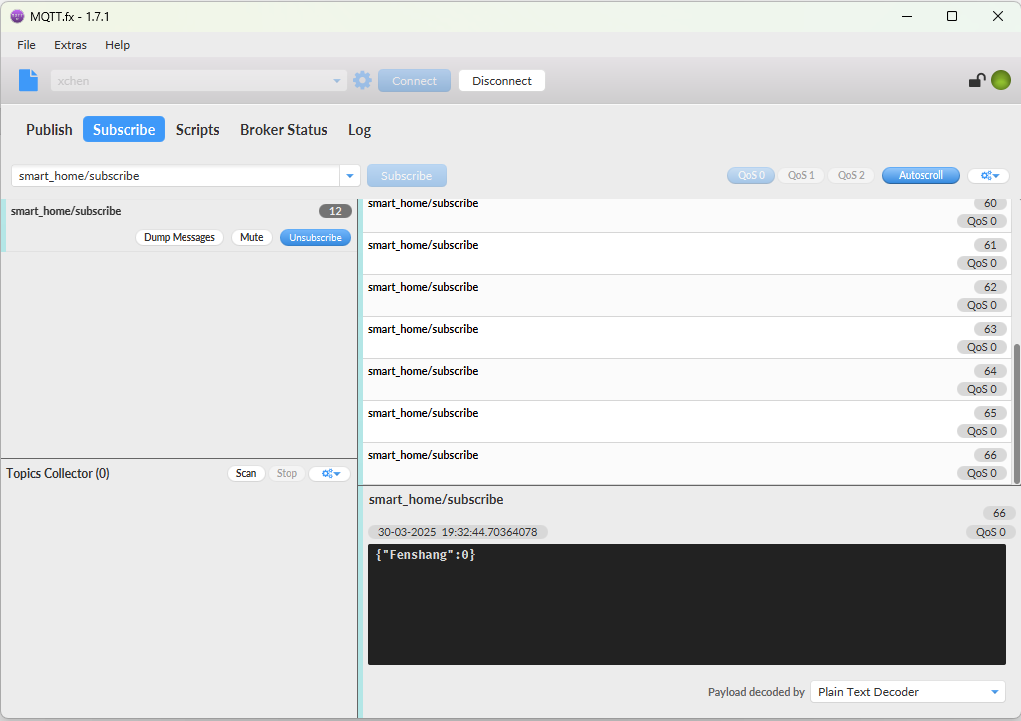


图 5.6控制/显示终端指令下发测试

### 功能测试总结

在监测/采集终端数据上传测试中，通过图 5.5展示了测试过程与结果，验证了终端设备能够稳定地将采集到的数据上传至系统，确保数据传输的准确性和可靠性。在控制/显示终端指令下发测试中，如图 5.6所示，测试了指令从控制端下发至显示终端的执行情况，确认指令能够被正确接收并顺利执行，通信链路的响应性得到了验证。

基于树莓派的智能家庭健康监测系统，监测/采集终端的数据上传测试以及控制/显示终端的指令下发测试均顺利完成，系统各部分之间的通信稳定无误，数据传输和指令执行均达到预期效果。这表明系统的通信模块设计合理，能够有效支持智能家庭健康监测的应用需求。

# 总结与展望

## 研究总结

至此，基于树莓派的智能家庭健康监测系统的研究任务已顺利完成，在研究设计的过程中克服了多项困难与挑战，最终成功的开发出了功能比较完善的硬件与软件系统，总体基本达到了预期的目标。总结如下：

1. 深入剖析“智能家庭健康监测”概念并提出个人见解

本研究通过调研国内外学者对智能家庭健康监测系统的理论阐述与实践案例，结合当前老龄化社会和健康管理需求的背景，形成了对该系统的独特理解。智能家庭健康监测不仅是对生理数据的简单采集，更应是一个融合环境感知、智能反馈与用户交互的生态系统。本文强调系统应以家庭为核心，为用户提供预防性健康管理，而非仅停留于被动监测。

1. 基于实际需求设计系统功能与架构并完成设备选型

在分析家庭成员对健康监测的多样化需求后，本研究明确了系统的核心功能，包括多参数健康监测、环境智能调节、数据可视化管理以及AI驱动的健康建议。基于此，设计了由感知层、网络层和应用层构成的系统架构，选用STM32F205RET6微控制器负责高效的数据采集与实时控制，树莓派5作为与用户的交互中枢，并搭配多样化的传感器与执行器，结合FreeRTOS、Qt、MQTT和MySQL等技术，完成了软硬件的科学选型，确保系统的高效性与扩展性。

1. 搭建开发环境并实现系统全流程开发

本研究在树莓派5上搭建了嵌入式Linux开发环境，利用Ubuntu 24.10支持Qt图形界面的流畅运行；在STM32端移植FreeRTOS，开发了多任务并发的采集与控制程序。同时，通过阿里云服务器部署MQTT通信服务和MySQL数据存储平台，实现了从数据采集到云端管理的完整流程。系统的开发过程注重模块化设计与软硬件协同，成功构建了一个功能齐备的智能家庭健康监测平台。

1. 系统测试与性能优化

通过分阶段测试，本研究验证了硬件电路的供电稳定性与传感器采集精度，完成了软件功能的调试与优化，消除了开发中出现的潜在问题。测试结果显示，系统在健康数据监测的实时性、环境控制的响应速度以及用户界面的操作便捷性方面表现良好。通过反复迭代，系统稳定性逐渐增强。

## 研究展望

尽管本次研究在功能实现和性能验证方面取得了较好的成果，但系统仍存在一定的局限性，未来的研究与开发可在以下几个方向进一步探索和完善：

1. 功能的扩展与精准化

当前系统主要监测心率、血氧、体温和部分环境参数，未来可集成更多健康指标如血压、血糖和睡眠的监测功能，以满足不同人群的个性化需求。同时，通过优化传感器算法和数据处理技术，进一步提升监测精度，减少环境噪声或用户操作对数据的影响。

1. 硬件的小型化与能效优化

本系统的硬件设计虽已实现较高集成度，但体积和功耗仍可进一步优化。未来可采用更先进的芯片工艺和低功耗设计，开发更小型化、便携化的设备，提升系统的易用性和家庭部署的灵活性。

1. 智能化与本地化处理能力提升

当前系统依赖云端AI进行健康分析，未来可引入边缘计算技术以及本地大模型，将部分数据处理和分析任务下沉至树莓派或STM32端，减少对网络的依赖，提升响应速度并增强数据隐私保护。此外，可结合深度学习模型，基于用户长期数据进行健康趋势预测和风险预警，进一步提升智能化水平。

1. 安全性和隐私保护的强化

系统通过MQTT和HTTP协议传输数据，通信稳定性较好，但安全性有待加强。未来可加入端到端加密、用户认证机制以及数据匿名化处理，确保用户健康数据的隐私安全，增强系统在实际应用中的可信度。

1. 应用场景的多元化

当前系统以家庭健康监测为主，未来可扩展至养老机构、社区医疗或远程诊疗等场景，通过与医疗系统的对接实现数据共享和远程支持。同时，可开发配套的移动端应用，实现随时随地的健康管理，拓宽系统的应用范围和实用价值。

通过以上方向的持续改进，基于树莓派的智能家庭健康监测系统有望在功能性、便捷性和安全性上实现更大突破，为智能家居和健康管理领域提供更全面、高效的解决方案，推动技术成果向更广泛的社会应用转化。

# 参考文献

1. 耿新,邢鹏飞,王立群.人口老龄化视域下的适老化居家体系设计研究[J].设计,2024,37(19):90-93.DOI:10.20055/j.cnki.1003-0069.002182.
2. 刘浩源,张慧颖.基于物联网技术的人体健康数据检测系统设计[J].吉林化工学报,2023,40(03):42-47.DOI:10.16039/j.cnki.cn22-1249.2023.03.009.
3. 梁峻阁,宋怡然,孙杨帆,等.基于可穿戴与可植入技术的人体健康物联网研究进展[J].物联网学报,2023,7(02):26-34.
4. 韩改宁,李永锋,高伊腾.基于嵌入式Qt下的MySQL数据库设计与开发[J].微型电脑应用,2020,36(05):25-27.
5. 汤晓.健康中国背景下家庭医生服务能力建设研究[D].上海工程技术大学,2020.DOI:10.27715/d.cnki.gshgj.2020.000094.
6. 顾彦.《“健康中国2030”规划纲要》发布看健康中国如何“三步走”?[J].中国战略新兴业,2017,(21):80-81.DOI:10.19474/j.cnki.10-1156/f.001171.
7. 牛帅,张弛,董振华,等.WM8978音频处理器的百度AI语音识别[J].单片机与嵌入式系统应用,2023,23(02):42-44.
8. 凌信航,王航蜀,吴俊.基于物联网的人体健康及家庭环境监测系统[J].电子器件,2022,45(05):1272-1278.
9. 尚宪茹,刘子建,胡志刚,等.国内外老年智能健康穿戴产品研究态势分析[J].丝绸,2025,62(02):82-93.
10. 赵有明,汪涛,张庆哲,等.基于互联网和传感器的家庭健康监护系统[J].中国科技论文在线精品论文,2020,13(04):491-496.
11. 凌信航,王航蜀,吴俊.基于物联网的人体健康及家庭环境监测系统[J].电子器件,2022,45(05):1272-1278.

# 致谢

时光荏苒，转眼间本科四年的学习生活即将画上圆满的句号，而这篇毕业论文的完成，不仅是我大学阶段学术探索的总结，也是对这段旅程中所有支持与陪伴的最好回馈。在论文的研究与撰写过程中，我得到了来自多方的帮助与鼓励，借此机会，我想向所有在我成长道路上给予关怀与指导的人们表达最诚挚的谢意。

首先，我要衷心感谢我的指导教师王武副教授。从论文选题的确定到研究方案的设计，再到系统开发与论文撰写的每一个环节，王老师始终给予我耐心的指导和无私的帮助。同时，我要感谢数学与计算机科学学院的各位老师。四年来，他们在课堂上为我传授了扎实的理论基础，在课外为我提供了丰富的实践机会。特别是物联网工程专业的课程体系，让我从嵌入式技术到物联网通信，再到数据分析与人工智能，逐步构建起完整的知识框架。这些宝贵的学习经历为本次毕业设计提供了坚实的支撑，使我能够将理论与实践相结合，完成智能家庭健康监测系统的设计与实现。

还要感谢云南民族大学为我提供的优越学习环境。学校先进的实验室设备、丰富的图书资源以及开放的学术氛围，为我的研究工作创造了良好的条件。特别是在嵌入式开发和服务器搭建过程中，实验室的硬件支持与技术资源让我能够高效地完成实验验证与系统测试。

毕业论文的完成是我本科生涯的一个重要节点，但它绝非终点，而是新的起点。带着这一路的收获与感恩，我将继续前行，探索更广阔的知识领域，以回报所有支持我的人。再次向所有帮助过我的老师、家人、朋友以及同学致以最深的谢意！愿未来的日子里，我们都能不忘初心，砥砺前行。

陈玄

2025年3月