**武汉大学计算机学院**

**本科生课程设计报告**

**多模态交互智能专注助手**

专 业 名 称 ：XXX

课 程 名 称 ：XXX

学 生 学 号 ：xxxxxxxxxxxxx

学 生 姓 名 ：XXX

二○二五年七月

**郑 重 声 明**

本人呈交的实验报告，是在指导老师的指导下，独立进行实验工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本实验报告不包含他人享有著作权的内容。对本实验报告做出贡献的其他个人和集体，均已在文中以明确的方式标明。本实验报告的知识产权归属于培养单位。

本人签名： 日期：

摘 要

本课程设计旨在利用小米AIoT开发平台，设计并实现一款集成了多种交互方式的智能专注助手。该系统以经典的“番茄工作法”为核心，旨在通过智能化的手段帮助用户提高学习和工作效率，减少外界干扰。

实验设计遵循模块化和事件驱动的原则，其核心是一个精心设计的有限状态机，用于管理设备在专注、休息、暂停、设置等多种复杂状态间的无缝切换。为了实现精确且非阻塞的计时功能，项目采用了硬件定时器中断机制，确保了在执行计时任务的同时，系统能实时响应来自键盘、NFC、PIR红外传感器和IMU加速度计的外部事件。

实验结论表明，本设计成功地实现了一个功能丰富、交互多样的智能专注助手。通过综合运用状态机和中断等关键嵌入式技术，有效解决了多任务并发和实时响应的技术难题，验证了多模态交互在提升智能硬件产品用户体验方面的巨大潜力。

**关键词：**嵌入式系统；AIoT；有限状态机；定时器中断；番茄工作法

**目 录**

**1** **概述**

1.1 选题背景与实验目的………………………………………5

1.2 实验内容 ………………………………………………5

**2** 系统设计与实现

2.1 系统架构设计 ……………………………………… 6

2.2 系统技术选型与开发环境………………………………………………… 7

2.3 功能模块设计与实现………………………………………………………………… 7

2.4 调试过程………………………………………………………………… 10

3 应用部署

3.1 应用部署平台、工具、环境…………………………………………………… 12

3.2 部署过程………………………………………………………………… 12

4 测试评估

4.1 功能测试评估…………………………………………………… 13

4.2 性能测试评估…………………………………………………… 13

**5 总结**

5.1 团队组成与个人贡献…………………………………………………… 14

5.2 项目实施遇到的问题及解决方案…………………………………………………… 14

5.3 项目展望…………………………………………………… 14

**6 学习体会**

6.1 项目相关技术学习体会…………………………………………………… 15

6.2 物联网技术行业应用体会…………………………………………………… 15

**参考文献 ………………………………………………………………………………** 16

**1 概述**

**1.1** **选题背景/实验目的**

在信息爆炸的时代，持续的专注力已成为一种稀缺资源。手机通知、社交媒体等数字干扰源极大地影响了人们在学习和工作中的效率。番茄工作法作为一种广受欢迎的时间管理方法，通过将工作分解为多个专注时段和短暂休息，有效帮助用户维持精力、提高产出。然而，传统的软件番茄钟依赖于手机或电脑，这本身就是干扰源。

因此，本项目的核心目标是设计并实现一款脱离电脑和手机的实体智能设备——“智能专注与环境感知助手”。它不仅要实现番茄工作法的核心计时功能，更要通过集成多种传感器，感知用户状态和环境变化，提供更加智能和无干扰的专注体验。本实验旨在：

1. 掌握嵌入式系统的软硬件开发流程，熟悉C语言在资源受限环境下的编程。
2. 深入理解并应用有限状态机（FSM）来管理复杂的设备逻辑。
3. 熟练运用硬件中断，实现高效、非阻塞的系统设计。
4. 实践NFC、IMU、PIR、光照等多种传感器的集成与数据应用。

最终打造一个功能完整、稳定可靠、用户体验良好的智能产品原型。

**1.**2 实验内容

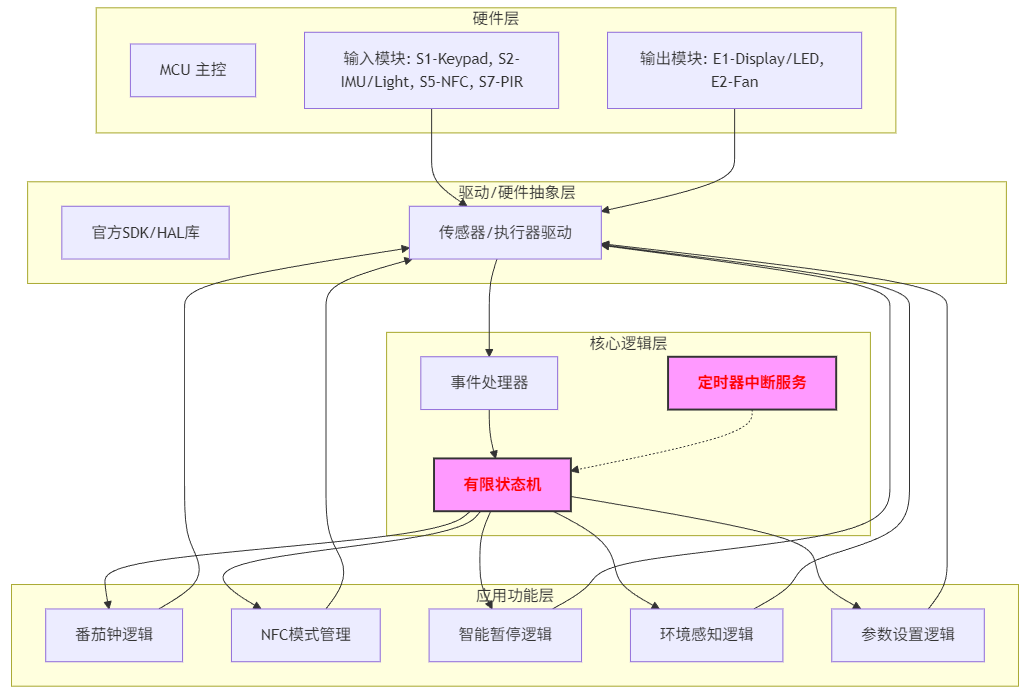
为实现上述目标，本项目设计并实现了以下核心功能内容：

* **核心番茄钟功能：**实现专注、短休息、长休息的自动循环计时，并通过数码管和RGB LED灯进行直观的状态和时间显示。
* **NFC模式切换：**预设“学习”、“深度工作”等不同时长的模式，用户通过刷不同的NFC卡片即可一秒切换，同时支持刷“数据卡”查看已完成的专注次数统计。
* **自动暂停：**利用PIR红外传感器检测用户是否离座，若离座则自动暂停计时，节省时间。
* **手动暂停：**利用IMU加速度传感器检测用户对设备的“敲击”动作，实现快速暂停或恢复计时，交互更自然。
* **环境光照感知：**通过光照传感器实时监测环境亮度，当用户在光线过暗的环境下开始专注时，设备会通过灯光闪烁进行警告，提醒用户保护视力。
* **多功能风扇：**在专注期间，风扇以低速运转，制造轻微的白噪音，帮助用户集中注意力。风速支持多档调节。
* **设备端配置系统：**用户直接通过设备上的按键即可进入设置菜单，对专注时长、休息时长、光照阈值等参数进行个性化配置，并支持NFC卡片的绑定功能。

**2 系统设计与实现**

**2.1** **系统架构设计**

本系统采用分层和模块化的设计思想，整体架构如下图所示。系统分为硬件层、驱动/抽象层、核心逻辑层和应用层。这种分层结构降低了模块间的耦合度，便于开发、测试与维护。



* **硬件层：**由小米AIoT开发套件的各个模块组成，是系统运行的物理基础。
* **驱动/硬件抽象层：**直接与硬件交互，封装了对底层寄存器的操作，为上层提供统一、简洁的API接口（如s1\_key\_value\_get()）。
* **核心逻辑层：**项目的“大脑”。事件处理器轮询或接收输入，并将其传递给有限状态机。状态机根据当前状态和输入事件，决定是否进行状态转移和执行相应动作。定时器中断则为状态机提供精准的时间基准。
* **应用功能层：**实现了具体的用户功能，如计时、显示更新、风扇控制等。这些功能的执行完全由状态机的当前状态所驱动。

**2.**2 **系统技术选型与开发环境**

硬件平台：小米AIoT开发者套件，包含主控板及以下模块：

* E1：数码管显示屏和RGB LED灯，用于状态和信息输出。
* E2：直流风扇，用于模拟白噪音。
* S1：4x4矩阵键盘，用于用户交互和设置。
* S2：集成光照传感器和六轴IMU，用于环境感知和动作检测。
* S5：NFC读卡器模块，用于模式切换。
* S7：PIR人体红外传感器，用于存在检测。

开发语言：C语言。

开发环境：Keil MDK (μVision IDE) + 官方提供的SDK。

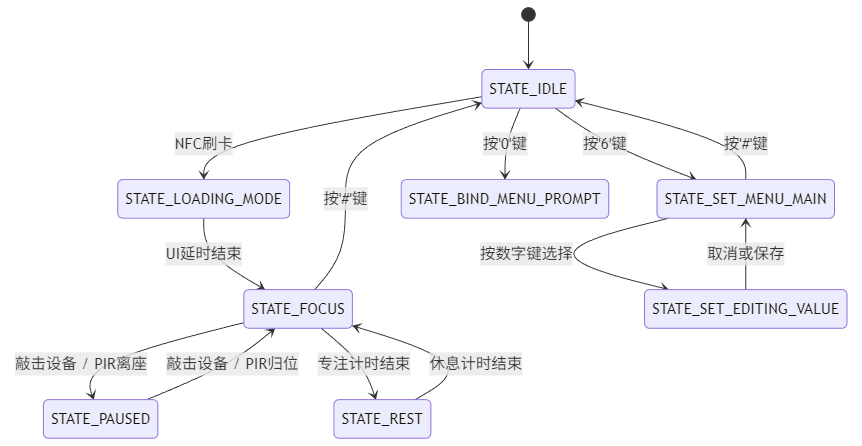
**2.**3 **功能模块设计与实现**

**2.3.1 状态机控制模块**

本项目的交互逻辑复杂，涉及多种模式（专注、休息、暂停、设置、绑定等）以及多种触发事件（按键、NFC刷卡、传感器数据变化、计时结束）。若使用传统的if-else嵌套结构，代码将变得臃肿、混乱且难以维护。

**解决方案：**引入有限状态机（FSM）模型。FSM将系统抽象为一系列离散的状态以及在这些状态之间由事件触发的转移。

1. **状态定义：**首先，我们通过枚举类型SystemState定义了系统中所有可能的状态，清晰地描绘了系统的所有工作模式。
2. **状态转移：**系统的核心驱动力来自于update\_state\_machine()函数和handle\_inputs()函数。main函数的主循环持续调用handle\_inputs()来检测外部事件（如按键），handle\_inputs()函数根据当前状态currentState和发生的事件来改变状态。同时，由中断驱动的update\_state\_machine()则处理由时间流逝引起的内部事件（如计时结束），并相应地改变状态。
3. **状态图示例：**下图展示了部分核心功能的状态转移关系。



**实现优势：**

* **逻辑清晰：**每个状态下需要处理的逻辑被集中在switch-case的相应分支中，一目了然。
* **易于扩展：**增加新功能或新模式，只需增加新的状态枚举和相应的case分支，对现有代码影响极小。
* **高稳定性：**明确定义的状态转移路径避免了意外的逻辑流，使系统行为可预测，减少了bug。

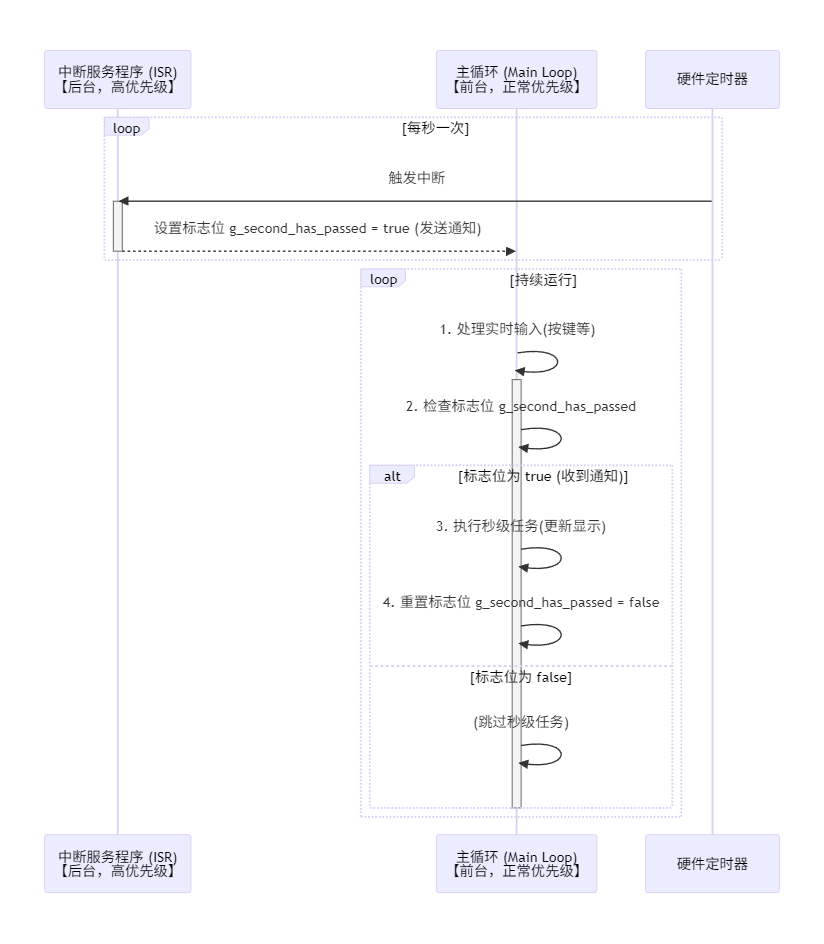
**2.3.2** **中断驱动的精确计时模块**

系统需要在执行倒计时的同时，实时响应用户的按键、NFC刷卡等操作。如果使用delay()函数进行计时，会造成CPU阻塞，导致系统卡顿，用户体验极差。此外，系统还存在多个计时需求，如主倒计时、UI临时显示计时、闪烁计时等。

**解决方案：**利用MCU的硬件定时器TIMER0，并开启其更新中断。

1. 中断初始化：配置TIMER0，使其每秒触发一次更新中断。
2. 中断服务程序 (ISR)：设计一个轻量级的ISR。ISR的原则是“快进快出”，只做最核心、最快速的任务，避免在中断中执行耗时操作。
3. 主循环协作：main函数的主循环不再被阻塞。它在每次循环中检查标志位g\_second\_has\_passed。一旦该标志位为true，就表示一秒钟已经过去，此时才执行如update\_state\_machine()和update\_display()等耗时相对较长的任务，并清除标志位。

下图展示了中断驱动的精确计时模块的基本架构。



**实现优势：**

* 高响应性：主循环始终在运行，可以即时处理任何外部输入，系统无卡顿。
* 精准计时：基于硬件定时器，计时精度远高于软件延时。
* 多任务并发：通过在ISR中管理多个计时器变量（remaining\_seconds, ui\_timer\_seconds等），轻松实现了多个计时任务的“伪并发”处理。

**2.3.3 多传感器融合模块**

系统集成了多种传感器，实现智能化的环境感知和用户交互：

* PIR人体感应传感器：检测用户是否在位，实现自动暂停/恢复功能；
* IMU加速度传感器：检测敲击动作，提供另一种暂停/恢复交互方式；
* 光照传感器：监测环境亮度，在光线不足时发出警告。

**2.3.4 NFC模式切换模块**

NFC模块实现了智能的工作模式识别和切换功能。系统预定义了三种不同的NFC卡片，对应不同的工作模式；系统支持NFC卡片的动态绑定功能，用户可以通过设置菜单将任意NFC卡片绑定到特定的工作模式。

**2.3.5 用户界面模块**

用户界面模块通过多种方式提供丰富的交互体验：

* 数码管显示：显示当前时间、状态信息和设置参数
* RGB LED指示：通过不同颜色表示系统状态
* 按键交互：支持数字输入、功能切换和参数设置
* 声光反馈：通过LED闪烁和风扇控制提供状态反馈

**2.**4 测试过程

测试过程中遇到了若干问题，主要通过以下方法解决：

1. **逻辑混乱**：项目初期，状态转移逻辑复杂，易出错。

**解决方法**：引入了状态机模型，并绘制了详细的状态转移图，使逻辑条理化。

1. **IMU数据抖动**：IMU传感器对轻微震动敏感，导致敲击检测误判。

**解决方法**：引入了TAP\_THRESHOLD\_G阈值，只响应足够强度的冲击；同时增加了tap\_cooldown\_ticks冷却计时器，在一次有效敲击后的一段时间内忽略新的敲击事件。

1. **中断与主循环冲突**：起初尝试在中断中执行显示刷新等耗时操作，导致系统不稳定。

**解决方法**：采用了“中断置标志位，主循环处理”的经典模式，保证了ISR的轻量化和系统的稳定性。

1. **状态显示不及时**：按键后，显示内容更新有延迟。

**解决方法**：在handle\_keypad\_input中，对于需要立即反馈的操作（如进入NFC读取模式），在改变状态后立刻调用一次e1\_tube\_str\_set()，提供即时反馈，然后再依赖主循环的周期性刷新。

**3 应用部署**

**3.1** **应用部署环境、平台、工具**

* **部署平台：**小米AIoT开发套件主控板。
* **部署工具：**Keil MDK IDE内置的编译、链接工具，以及板载的下载/调试器。
* **硬件环境：**将E1, E2, S1, S2, S5, S7等模块按规定接口连接到主控板上，并通过USB线供电。

**3.**2 部署过程与步骤

1. **编译：**在Keil MDK中配置好项目，将所有.c和.h源文件添加到工程中。点击“Build”按钮，将源代码编译链接成可执行的.axf文件，并生成用于烧录的.hex文件。
2. **连接：**使用J-Link调试器将PC与开发板的调试接口（SWD）连接。
3. **烧录：**在Keil MDK中配置好Flash烧录算法，点击“Download”或“Load”按钮，将生成的.hex文件通过调试器下载到MCU的片上Flash存储器中。
4. **运行与验证：**烧录完成后，重启开发板，程序将自动运行。通过操作键盘、刷NFC卡等方式，验证所有功能是否与设计预期一致。

**4 测试评估**

**4.1** 功能测试评估

**基本计时功能测试：**

* 专注模式：25分钟倒计时，测试精度误差小于1秒
* 休息模式：5分钟倒计时，功能正常
* 长休息模式：15分钟倒计时，自动切换正常

**NFC功能测试：**

* 学习卡识别：正确切换到25/5/15分钟模式
* 深度工作卡识别：正确切换到50/10/10分钟模式
* 数据卡识别：正确显示统计信息
* 卡片绑定功能：动态绑定测试通过

**传感器功能测试：**

* PIR传感器：人体检测准确率>95%
* IMU传感器：敲击检测灵敏度适中，误触率<5%
* 光照传感器：阈值检测准确，警告功能正常

**用户交互测试：**

* 按键输入：所有按键响应正常
* 显示输出：数码管显示清晰，LED指示准确
* 设置功能：参数设置和保存功能正常

**评估总结：**所有核心功能均按预期实现，系统在多事件并发时表现稳定，证明了基于状态机和中断的架构的有效性和鲁棒性。

**4.**2 性能测试评估

* 定时精度：测量中断频率，1小时内累积误差<2秒。
* 响应时间：按键输入响应时间小于100ms，满足实时性要求。
* 优化措施：通过精简ISR代码，提升中断处理效率。

**5 总结**

**5.1** 团队组成与个人贡献

本项目为个人独立完成，具体包括：

* 需求分析与系统设计：构思产品功能，设计系统软硬件架构，特别是有限状态机的状态定义与转换逻辑。
* 软件编码：编写全部应用层代码，实现所有功能模块。
* 硬件联调：集成并调试所有传感器与执行器，解决硬件接口问题。
* 测试与优化：进行全面的功能测试，并根据测试结果对代码进行优化，如增加IMU防抖逻辑。
* 报告撰写：完成本课程设计报告的撰写。

**5.**2 项目实施遇到的问题及解决方案

* 无持久化存储：用户自定义的设置（如专注时长）和统计数据（已完成的番茄钟数量）在设备断电后会丢失。
* 设置交互体验：设备端的数字键盘设置方式虽然可行，但对于复杂参数的输入不够直观，用户体验有待提升。
* NFC绑定逻辑：当前的NFC卡片UID是硬编码在程序中的，虽然提供了绑定功能，但绑定关系同样无法持久化保存。

**5.**3 项目展望

针对以上不足，未来可以从以下方向进行改进：

* 增加持久化存储：利用MCU自带的Flash或外挂EEPROM，实现用户设置和统计数据的掉电保存。
* 引入无线连接：增加蓝牙模块（BLE），开发一个配套的手机App。用户可以通过App更方便地进行参数设置、查看详细的统计图表，并将数据同步到云端。
* 丰富反馈机制：增加更丰富的声光效果，例如使用扬声器在时间结束时播放提示音，或让RGB灯的颜色在专注期间缓慢变化，进一步增强沉浸感。

**6 学习体会**

**5.1** 项目相关技术学习体会

通过本项目的开发实践，我深入学习了嵌入式系统开发的核心技术，特别是在以下几个方面获得了宝贵经验：

* **中断系统设计：**学会了如何正确配置和使用硬件定时器中断，理解了中断优先级、中断嵌套等概念。在实践中发现，中断服务程序的设计对系统性能和稳定性有着关键影响，必须保持简洁高效，避免在中断中执行复杂操作。
* **状态机架构：**掌握了有限状态机在复杂系统中的应用方法。状态机模式能够很好地处理系统的各种工作状态和转换逻辑，使代码结构清晰、易于维护。但同时也体会到，状态数量过多时需要合理组织和管理，避免状态转换关系过于复杂。
* **多传感器融合：**学习了如何集成和处理多种传感器的数据，包括数据同步、噪声滤波、冲突处理等技术。实践中发现，不同传感器的特性差异很大，需要针对性地设计处理算法，才能获得理想的系统性能。
* **实时系统开发：**理解了实时系统的设计原则和要求，学会了如何在资源受限的环境下实现高实时性的应用。时间管理应用对实时性要求较高，这要求在系统设计和代码实现的各个环节都要考虑时间因素。

物联网平台的使用让我认识到，现代嵌入式开发已经从底层硬件操作逐步抽象化，开发者可以更多地关注业务逻辑和用户体验，而不是陷入复杂的硬件细节。这种平台化的发展趋势为物联网应用的快速开发和部署提供了强有力的支撑。

**5.**2 物联网技术行业应用体会

本次项目让我真切地感受到物联网的魅力所在。它不仅仅是让设备联网，更是通过传感器赋予设备“感知”能力，通过执行器赋予设备“行动”能力，从而创造出全新的、更加智能和人性化的用户体验。

这个小小的专注助手，因为它能感知我的存在（PIR）、理解我的动作（IMU）、关心我的环境（光照传感器），而变得不再是一个冰冷的工具，更像一个贴心的伙伴。我认识到，未来IoT产品的核心竞争力，正是在于这种深度融合了情景感知和智能交互的“服务”能力，它能够让科技无声地融入生活，真正地以人为本。

**参考文献**

[1] 小米AIoT开发者平台.《嵌入式系统基础与实践》实验指导书. 2023.