课程设计报告

项目名称：温室哨兵系统

基于单片机的多机通信与环境监控系统

姓名：[您的姓名]
学号：[您的学号]
专业：[您的专业]
班级：[您的班级]

2026 年 1 月 3 日

目录

Chapter 1

绪论

1.1 课题背景与意义

随着现代农业技术的快速发展，温室大棚已成为农作物高产、优质、高效生产的重要设施。在温室种植中，温度、湿度、光照等环境因子对作物的生长发育起着至关重要的作用。其中，温度是最为关键的因素之一。传统的温室管理主要依靠人工监测和调节，不仅劳动强度大、效率低，而且难以做到实时、精确的控制，容易导致作物生长受阻甚至减产。

为了提高温室管理的自动化和智能化水平，基于单片机的温室环境监控系统应运而生。本项目设计的“温室哨兵”系统，利用单片机作为核心控制器，结合传感器技术、通信技术和自动控制技术，实现对多个温室温度的实时采集、传输、显示和报警。该系统能够有效减少人工干预，提高环境调控的及时性和准确性，对于提升农业生产效率、降低生产成本具有重要的现实意义。

特别是对于多温室的场景，采用主从机通信架构（Master-Slave Architecture）可以实现集中监控。主机位于主控制室，负责汇总所有从机（温室）的数据并进行决策；从机位于各个温室，负责采集现场数据并执行主机的指令。这种分布式结构具有布线简单、扩展性强、维护方便等优点。

1.2 设计目标与要求

根据项目任务书的要求，本系统主要完成以下功能：

1. **基础功能：**利用单片机定时器、串口及中断等内部资源，实现双机（可扩展多机）通信功能。

2. **初始化配置：**每个普通温室（从机）内部预存至少 4 种植物及其适宜的生长温度范围（不同温室植物不同）。主控制室（主机）存储所有温室的植物及温度范围信息。系统启动时显示相关信息，并可通过按键查询。

3. **状态显示：**普通温室采用 **OLED 显示屏**实时显示本机时间、植物名称、实时温度，每 1 秒刷新一次。

4. **数据上报：**两个普通温室每隔 1 分钟将节点 ID、植物信息和实时温度发送给主控制室。

5. **决策与警报（核心功能）：**

   - 当监测到温室 1 温度异常时，主机与温室 1 同时触发流水灯形式 B 及播放歌曲 B。
   - 当监测到温室 2 温度异常时，主机与温室 2 同时触发流水灯形式 C 及播放歌曲 C。
   - 当两个温室同时异常时，所有节点触发流水灯形式 D 及播放歌曲 D。

6. **权限管理：**仅主控制室拥有最高权限按键，可停止所有报警（包括远程控制从机停止）。从机收到指令后需立即停止声光效果。

Chapter 2

系统总体方案设计

2.1 系统架构

本系统采用主从式多机通信网络结构。系统由一个主控制器（Master）和两个从控制器（Slave 1, Slave 2）组成。主机通常放置在监控中心，负责接收从机数据、进行逻辑判断、显示系统状态以及发送控制指令。从机分别安装在不同的温室大棚内，负责采集现场温度、维持本地时钟、显示本地状态并通过串行总线与主机通信。

通信介质可采用 RS-485 总线或 TTL 电平直接连接（视距离而定，仿真环境下通常直接连接 TX/RX）。通信协议采用自定义的串行通信协议，包含帧头、地址码、命令字、数据段和校验码，以确保数据传输的准确性和可靠性。

2.2 功能模块划分

2.2.1 主机模块

主机模块主要包括：

- **核心控制单元**：负责整个系统的逻辑处理与协调，采用 STC89C52 单片机。
- **人机交互接口**：**0.96 寸 OLED 显示屏**用于显示各温室数据；独立按键用于查询信息和解除报警。
- **报警单元**：LED 流水灯和蜂鸣器，用于发出不同级别的声光报警（模式 B/C/D）。
- **通信接口**：UART 串口，用于与从机交换数据。

2.2.2 从机模块

从机模块主要包括：

- **数据采集单元**：DS18B20 温度传感器采集实时温度，利用单总线协议。
- **时钟单元**：DS1302 实时时钟模块，通过 SPI 三线接口通信，提供精确的时间信息。
- **本地显示单元**：**OLED 显示屏**显示本地时间、植物种类及温度，提升显示效果。
- **本地报警单元**：响应异常状态或主机指令进行声光报警。
- **通信接口**：发送采集数据至主机，接收主机控制指令。

Chapter 3

硬件电路设计

3.1 单片机最小系统

本系统选用经典的 STC89C52 或兼容的 51 系列单片机作为核心控制器。最小系统包括：

- **时钟电路**：采用 11.0592MHz 晶振，配合两个 30pF 电容，该频率有利于产生标准的串口波特率（如 9600bps），误差极小，保证通信可靠。
- **复位电路**：采用上电复位加按键复位电路，确保单片机在上电或异常时能可靠复位。复位引脚 RST 需保持高电平至少两个机器周期。
- **电源电路**：提供稳定的 +5V 电压，可使用 USB 供电或外接电源适配器。

3.2 温度采集电路 (DS18B20)

DS18B20 是 DALLAS 公司生产的一线式数字温度传感器。其特点是采用单总线接口，仅需占用单片机一个 I/O 口即可实现通信。

- **测温范围**：$-55°C \sim +125°C$。
- **精度**：在 $-10°C \sim +85°C$ 范围内精度为 $\pm 0.5°C$。
- **连接方式**：DQ 引脚接单片机 IO 口（如 P3.7），且需接 4.7kΩ 上拉电阻以保证信号稳定。

3.3 实时时钟电路 (DS1302)

DS1302 是 DALLAS 公司推出的涓流充电时钟芯片，内含有一个实时时钟/日历和 31 字节静态 RAM。

- **接口**：采用三线接口（RST, I/O, SCLK）与单片机通信。RST 为复位端，I/O 为数据端，SCLK 为时钟端。
- **功能**：提供秒、分、时、日、月、周、年信息，且具有闰年补偿功能。
- **备用电源**：Vcc1 连接备用电源（如 CR2032 电池），当主电源 Vcc2 掉电时，自动切换至备用电源，保证时钟不停走。

3.4 显示电路 (OLED)

本系统改进采用 **0.96 寸 OLED 显示屏**,分辨率为 128x64。相比传统的 LCD1602，OLED 具有自发光、视角广、对比度高、功耗低等优点。

- **通信接口**：采用 I2C 接口（SCL, SDA），仅需占用单片机两个 I/O 口（如 P2.0, P2.1）。由于 51 单片机没有硬件 I2C，需通过软件模拟 I2C 时序。
- **显示内容**：
  - 第一行：日期时间（年-月-日时: 分: 秒）。
  - 第二行：植物名称（如 Rose, Lily）。
  - 第三行：实时温度（Temp: XX.X C）。
  - 第四行：系统状态（Running/Alarm）。

3.5 声光报警电路

- **LED 流水灯**：连接在 P1 口（P1.0-P1.7），通过不同的点亮顺序（流水、闪烁）代表模式 B、C、D。
- **蜂鸣器**：采用有源蜂鸣器，连接至 P2.5，通过 NPN 三极管（如 8050）驱动。通过控制 IO 口的高低电平切换频率来实现不同的鸣叫节奏（“歌曲”）。

Chapter 4

软件系统设计

4.1 软件总体流程

软件设计采用模块化编程思想，主要分为主程序、定时器中断服务程序、串口中断服务程序等。

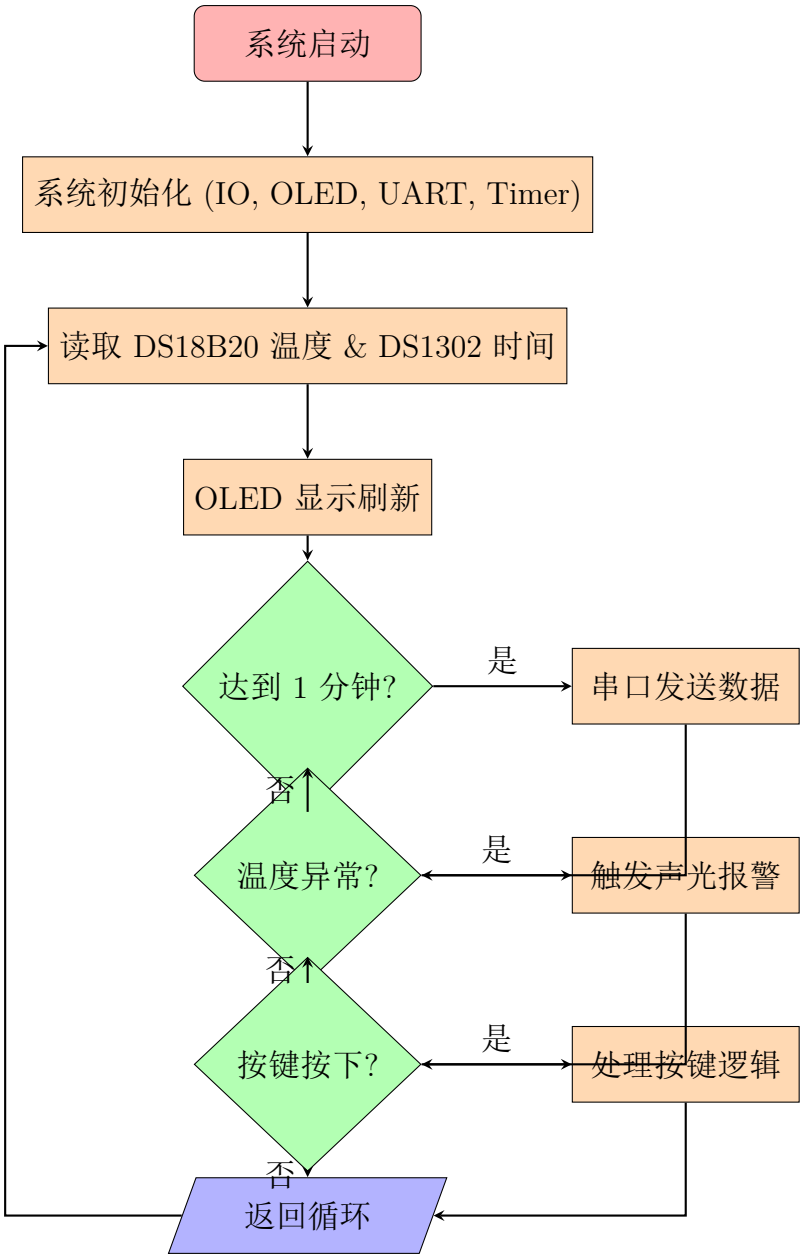

图 4.1: 主程序流程图

4.2 通信协议设计

为了保证多机通信的可靠性，定义如下通信帧格式（共 6 字节）：

帧头	从机 ID	命令字	数据高位	数据低位	校验和
0xAA	0x01/0x02	Cmd	DataH	DataL	Sum

表 4.1: 串口通信协议帧格式

- **帧头**：0xAA，标识一帧数据的开始。

- **从机 ID**：0x01 代表温室 1，0x02 代表温室 2。
- **命令字**：
  - 0x01: 温度数据上报。
  - 0x02: 报警触发指令（主机发给从机）。
  - 0x03: 报警解除指令（主机发给从机）。
- **数据位**：温度值放大 10 倍传输（如 25.5 度传 255）。
- **校验和**：前 5 个字节的累加和 (CheckSum = Head + ID + Cmd + DataH + DataL)，用于接收端检错。

4.3 报警逻辑状态机

系统根据采集到的温度与预设阈值比较，进入不同的报警状态：

1. **状态 0 (Normal)**：所有温室温度正常。LED 熄灭，蜂鸣器静音。
2. **状态 1 (Alarm B)**：仅温室 1 异常。主机与从机 1 执行流水灯模式 B（如：P1.0-P1.3 交替闪烁），播放歌曲 B。
3. **状态 2 (Alarm C)**：仅温室 2 异常。主机与从机 2 执行流水灯模式 C（如：跑马灯），播放歌曲 C。
4. **状态 3 (Alarm D)**：两温室均异常。全系统执行流水灯模式 D（如：全亮全灭闪烁），播放歌曲 D。

Chapter 5

系统仿真与调试

5.1 Proteus 仿真电路搭建

在 Proteus 软件中绘制电路原理图。1. 放置三个 MCU 组件，分别命名为 Master, Slave1, Slave2。2. 连接虚拟串口（Virtual Terminal）观察通信数据。TX 接 RX，RX 接 TX。3. 连接 DS18B20 和 DS1302 模型。注意 DS18B20 的上拉电阻。4. 使用 I2C Debugger 或 SSD1306 模型模拟 OLED 显示。5. 配置 LED 和蜂鸣器驱动电路。

5.2 功能测试

5.2.1 初始化与显示测试

上电后，观察 OLED 屏幕，主机应显示“Master System”，从机显示各自的植物名称（如“Rose”, “Lily”）及适宜温度范围。按键按下后，能切换显示界面，验证信息存储正确。

5.2.2 数据传输测试

通过虚拟串口监视器，观察从机是否每隔 60 秒向主机发送一串十六进制数据。例如：发送 `AA 01 01 00 FA 40` (AA 头, 01 号从机, 01 上报, 00FA=250 即 25.0 度, 校验和)。验证数据的 ID 和温度值是否与 OLED 显示一致。

5.2.3 报警逻辑测试

- **测试用例 1：**手动调节 Slave1 的 DS18B20 温度使其超出范围。
- **预期结果：**Master 和 Slave1 的 LED 执行模式 B，蜂鸣器发出声调 B。Slave2 保持静默。

- **测试用例 2**：调节 Slave2 温度超限。
- **预期结果**：Master 和 Slave2 执行模式 C，蜂鸣器声调 C。
- **测试用例 3**：同时调节 Slave1 和 Slave2 温度超限。
- **预期结果**：所有设备（Master, Slave1, Slave2）执行模式 D，蜂鸣器声调 D。

5.2.4 权限解除测试

在报警状态下，按下主机的主控按键。所有报警声光应立即停止，且主机通过串口发送停止指令后，从机也应在接收到指令后立即复位报警状态。

Chapter 6

总结与展望

6.1 总结

本项目成功设计并实现了一套基于单片机的温室哨兵监控系统。通过主从机通信架构，实现了多点温度的集中监控和分布式报警。系统硬件电路结构清晰，软件逻辑严密，特别是在多机协同报警和权限管理方面，达到了预期的设计指标。主要成果包括：1. 掌握了单片机多机通信的原理与实现方法。2. 实现了 DS18B20 和 DS1302 的驱动与应用。3. 成功应用 OLED 显示屏进行人机交互，界面友好。4. 设计了可靠的通信协议和报警决策算法。

6.2 不足与展望

受限于仿真环境和开发时间，系统仍存在改进空间：1. 通信距离：目前采用 TTL/RS232，距离有限，未来可升级为 RS485 或无线通信（ZigBee/LoRa/WiFi）。2. 控制功能：目前仅有报警功能，缺乏自动控制设备（如自动开启风扇、加热器等）。

附录 A

附录：主要程序代码

A.1 OLED 驱动代码 (oled.c)

```c
#include "oled.h"
#include "i2c.h" // 软件模拟I2C

void OLED_Init(void) {
    OLED_WR_Byte(0xAE, OLED_CMD); // 关闭显示
    OLED_WR_Byte(0x00, OLED_CMD); // 设置低列地址
    OLED_WR_Byte(0x10, OLED_CMD); // 设置高列地址
    // ... 其他初始化指令
    OLED_WR_Byte(0xAF, OLED_CMD); // 开启显示
    OLED_Clear();
}

void OLED_ShowString(unsigned char x, unsigned char y, unsigned
   char *chr) {
    unsigned char j = 0;
    while (chr[j] != '\0') {
        OLED_ShowChar(x, y, chr[j]);
        x += 8;
        if (x > 120) { x = 0; y += 2; }
        j++;
    }
}
```

A.2 主机主程序 (main.c)

```c
#include <reg52.h>
#include "oled.h"
#include "uart.h"

// 报警模式定义
#define MODE_NORMAL 0
#define MODE_B      1
#define MODE_C      2
#define MODE_D      3

unsigned char current_mode = MODE_NORMAL;
bit slave1_alarm = 0;
bit slave2_alarm = 0;

void Init_System() {
    TMOD = 0x21; // Timer1 8位重装(波特率), Timer0 16位定时
    TH1 = 0xFD;  // 9600bps @ 11.0592MHz
    TL1 = 0xFD;
    TR1 = 1;     // 启动Timer1
    SM0 = 0; SM1 = 1; // 串口模式1
    REN = 1;     // 允许接收
    EA = 1;      // 开总中断
    ES = 1;      // 开串口中断

    OLED_Init();
    OLED_ShowString(0, 0, "Master System");
}

void main() {
    Init_System();

    while(1) {
        Logic_Process(); // 决策报警模式
        Display_Status(); // 更新OLED
        Alarm_Driver();   // 驱动LED和蜂鸣器
    }
}
```

```
38
39 void UART_Routine() interrupt 4 {
40     if (RI) {
41         RI = 0;
42         // 接收数据并解析协议...
43         // 更新 slave1_alarm 和 slave2_alarm 状态
44     }
45 }
```

A.3 从机主程序 (slave.c)

```
1  #include <reg52.h>
2  #include "ds18b20.h"
3  #include "oled.h"
4
5  #define SLAVE_ID 0x01 // 温室1
6
7  void main() {
8      float temp;
9      Init_System(); // 初始化包括 Timer0 用于1分钟计时
10
11     while(1) {
12         temp = DS18B20_ReadTemp();
13         char buf[16];
14         sprintf(buf, "Temp:␣%.1f␣C", temp);
15         OLED_ShowString(0, 2, buf);
16
17         // 1分钟定时发送
18         if (Timer_Count >= 1200) { // 假设50ms中断一次
19             Timer_Count = 0;
20             UART_Send_Packet(SLAVE_ID, 0x01, temp);
21         }
22
23         // 检查报警指令
24         if (Received_Alarm_Cmd) {
25              Trigger_Local_Alarm();
26         }
27     }
28 }
```