安徽建筑大学
ANHUI JIANZHU UNIVERSITY

毕 业 论 文

题　　目：多参数水质监测系统研制

姓　　名：夏庆生

学　　号：17205040229

学　　院：电子与信息工程学院

专　　业：通信工程

指导教师：徐荃

完成时间：2021 年 6 月 4 日

摘　要

随着经济发展和公众卫生意识的提升，人们对饮用水、生活用水和养殖用水的水体质量越来越关注。然而，我国部分地区的水体质量令人堪忧，水体污染问题时有发生，对人们的身体健康产生了极大的危险，因此及时掌握水体的水质情况具有十分重要的意义。

目前，市面上现有的水质监测设备因为价格昂贵而令人望而却步。为此，本文设计了一种简单实用且价格低廉的多参数水质检测系统，用以满足水质监测的需要。系统以 STM32F407ZGT6 为主控芯片，结合温度传感器、PH 值传感器、浑浊度传感器、电导率传感器、液晶显示屏等，实现对水体中水温、PH 值、浑浊度、电导率等重要水质参数的实时采集和显示。

温度传感器输出的数字量直送到主控单元，经处理后送往显示屏显示，而其余三个传感器模块的输出均需由主控单元的 ADC 外设进行抽样、量化、编码形成数字量再通过 DMA 外设传输到主控芯片的内存中，最后交由显示模块显示。显示屏自带液晶控制器 NT35510，与 STM32 单片机通过主控芯片中的 FSMC 外设模拟 8080 时序进行通讯。

关键词：STM32F407ZGT6；KEIL；温度；PH 值；浑浊度；电导率

ABSTRACT

With the development of economy and the improvement of public health awareness, people pay more and more attention to the water quality of drinking water, domestic waterand aquaculture water.However, the quality of water in some areas of our country is worrying. Water pollution occurs from time to time, which poses a great danger to people's health. Therefore, it is of great significance to grasp the water quality in time.

At present, the existing water quality monitoring equipment in the market is prohibitive because of its high price.Therefore, this paper designed a simple, practical and low-cost multi-parameter water quality detection system to meet the needs of water quality monitoring.The system uses STM32F407ZGT6 as the main control chip, combined with temperature sensor, pH sensor, turbidity sensor, conductivity sensor, liquid crystal display screen, etc., to realize real-time acquisition and display of important water quality parameters such as water temperature, pH value, turbidity, conductivity and so on.

The digital output of the temperature sensor is sent directly to the master control unit, and then sent to the display screen after processing. The output of the other three sensor modules shall be sampled, quantified and encoded by the ADC peripheral of the master control unit, and then transmitted to the memory of the master control chip through the DMA peripheral, and finally displayed by the display module.The display comes with LCD controller NT35510, which communicates with STM32 microcontroller through the FSMC peripheral in the main control chip to simulate the 8080 timing sequence.

Keywords: STM32F407ZGT6; KEIL; temperature; PH value; Electrical conductivity; Turbidity

目　录

第一章 绪论

1.1 本课题背景

水资源是人类赖以生存的自然资源之一，然而，近些年来，全球许多地区的水污染问题已成愈演愈烈之势，有些地方的生活用水已被严重污染，以至于根本无法供人类正常生产生活所用，必须要经过复杂的污水处理工序才能被使用。因此，如何确保水质安全已逐渐成为一项全民关注的热点问题。

我们在日常生活中，通常既不清楚自己生活用水的来源，也不清楚自己饮用水的卫生状况如何，以及究竟是否可以供我们正常生产生活所用。为此，本文根据人们对于生活用水水质的需求，设计了一款基于 STM32F407ZGT6 单片机的多参数水质监测系统，它可以从所取水样中实时采集温度、pH 值、电导率、浑浊度等数据，经分析处理后，以一种简单易懂的方式将其显示在液晶屏上以作监测之用。

除了温度、pH 值、电导率和浑浊度之外，表征水质状况的参数还有游离氯、溶解氧等，但这些参数测量起来比较复杂，而温度、pH 值、电导率和浑浊度这四项参数指标已基本可以反映我们日常饮用水的水质状况。

1.2 本课题的研究现状及发展趋势

过去几十年来，“智慧水”作为智慧城市的一个重要组成部分被大众熟知，因此，在近些年来，人们不断提出新的有关于水资源管理、检测和治理的方法，在论文《基于物联网的实时饮用水管道污染检测系统》中，作者们提出了一种监测水管中水质的方法：使用智能传感器实时测量水管中的数据，使用 ZigBee 通信，并将 LED 显示屏用于显示结果，但用户需要手动检查各项参数。在论文《一种低功耗远程实时监测评估水质系统》中，作者提出了一种远程实时水质检测的低功耗系统。在论文《基于 STC89C52 的鱼塘水质检测系统》中，作者设计了一种可以检测并调节多种水质参数的水质检测系统，其以 STC89C52 作为主控芯片，使用 DS18B20 作为温度传感器，采用 AD 转换器来采集数字信号并处理，同时系统还具有报警功能。

根据上述的研究现状综述，不难发现，目前国内外关于水质检测呈现出以下发展趋势：

1. 参数多元化。水体中含有多种矿物质及多种元素，因而也会呈现出不同的物理

和化学特性，显然要想较为准确地描述一份水样的具体水质状况，参数越多，则描述越为精确。

2.标准化。目前制作一种简易实用的水质检测系统已有多套较为成熟的设计方案，器件种类也基本相对固定。

3.成本低。随着制造工艺的提升及检测系统设计方案的标准化规范化，目前无论是批量生产还是自制水质检测系统所花费的成本都越来越低，并且检测精度越来越高。

1.3 本课题的研究目的及意义

基于单片机的多参数水质检测系统会给人们的生产生活带来许多便捷，实时数据测量、传输、存储和分析的成本更低。我们可以开发出一种即插即用的产品，它可以被放置在任意的水源处，并测量一些重要的指标，进行分析并向用户展示分析结果，使得即便是非技术人员也能够了解检测的水质情况。类似于这样的技术方案有助于提升大众卫生意识，用户们将会了解水质污染所带来的影响及危害。

1.4 本课题的主要研究工作

本文主要研究基于 STM32F407ZGT6 单片机的多参数水质参测系统，主要实现的功能是实时检测水样中的温度、pH 值、电导率和浑浊度。本系统的设计流程为：查阅文献资料→确定系统设计方案及器件选材→购买元器件→硬件系统搭建→绘制原理图→源程序开发→系统调试。其中，各步骤主要内容如下所示：

1. 查阅中英文文献资料，确定一个简易的水质检测系统应包括哪些模块。
2. 查阅相关器件手册及传感器数据手册，了解元器件参数，确定系统设计方案。
3. 根据系统设计方案及所选元器件型号购买器件。
4. 根据器件手册上的引脚说明将传感器和显示模块分别同开发板连接起来。
5. 用 AD 软件根据各个模块手册上的参考原理图及元器件封装库绘制电路原理图。
6. 查阅 STM32F407 数据手册及传感器数据手册，据此编写各个外设及传感器模块的驱动程序（其中 IO 口的配置与步骤 3 中的硬件连接要相对应）。
7. 检查各个模块的硬件连接，将源程序下载到开发板并调试。

第二章 系统方案设计

2.1 系统功能分析

本文旨在设计一种能够实时精确检测水质参数并向用户反馈的水质检测系统。如前所述，本系统应能够同时检测所取水样中的温度、浑浊度、pH 值和电导率，并直观地显示在显示屏上，以供用户实时查看。

2.2 系统方案选择

方案一：

采用 STC89C51 作为主控芯片，外加水质参数传感器模块、ADC 转换芯片、DMA 芯片以及 1602 显示屏。

STC8C51 系列单片机是一种 8 位微控制器，内嵌 64KB 的 Flash，支持 ISP(In-System-Programming)，即系统内编程--支持用户在系统内修改程序，而不需将微控制器拆卸下来，芯片内部还有 1280 字节或 512 字节的片上 RAM，以供程序开发备用。STC89 系列单片机保留了标准 80C51 单片机的所有特征，另外，STC89 系列单片机还有额外的 I/O 端口(P4),片上晶振。但是，STC89C51 没有片上 ADC 外设，以及 DMA 外设，如果其他模块需要用到这两个外设，则需要另外添加。它使用寄存器编程方法。

1602 字符型液晶显示器是一种点阵式 LCD，目前常用的模块大小分为 16×1,16×2,20×2 和 40×2 行等。它具有显示质量高、体积小、重量轻以及功耗低等优点。常见的 1602 显示屏其驱动电压为 3.3V 或 5V，工作温度为-20℃～+70℃。

方案二：

采用 STM32F407ZGT6 芯片作为主控芯片，由于该芯片内部集成了 ADC、DMA、USART 等外设，外设只需要接水质参数传感器模块和 4.3 寸的液晶显示屏即可。

STM32F407ZGT6 是 STM32F407 家族的一员，它基于高性能的 ARM Cortex-M4 的 32 位 RISC 内核，工作主频高达 168MHz,Cortex-M4 内核有一个 FPU(Floating point Unit)，即浮点运算单元，可以用于进行高速浮点运算。同时 STM32F407 家族芯片内嵌高速存储器，包含存储容量高达 1MB 的 Flash，以及 192KB 的 SRAM，4KB 的后备 SRAM。同时，STM32F407ZGT6 内部还有 3 个 12 位 ADC，3 个 I2C,4 个 USART

以及其他外设。它使用固件库函数编程方法。

4.3 寸液晶显示屏的屏幕分辨率为 800×480，采用 16 位并口 8080 时序，供电电压为 5V 和 3.3V，需要同时供电，控制器为自带的 NT35510 控制器，触摸芯片为 GT917S。

若采用方案一，即使用 STC89C51 作为主控芯片，则由于 PH 值、电导率和浑浊度传感器都需要配套 ADC 进行模数转换，而需要另外添加 ADC 外设，同时也要添加 DMA 外设，但如果使用 STM32 作为主控芯片，由于其内嵌 ADC 和 DMA，则硬件系统搭建比较方便。另外，STC89C51 使用寄存器编程方法，而 STM32F407ZGT6 使用固件库编程方法，后者由前者发展而来，采用固件库编程时开发程序更为快捷，基于 STM32F407ZGT6 的这些优点，本文使用它作为主控芯片。由于 1602 显示屏分辨率较低，一般只能显示两行，而本文所设计的检测系统需同时检测四种水质参数，至少需要同时显示四行信息，而 4.3 寸和 4,5 寸液晶显示屏的分辨率均可满足需求，从经济角度出发，本文选择 4.3 寸液晶显示屏。

综上所述，本文使用方案二来设计硬件系统。

2.3 系统硬件框图

如图 2.1，温度传感器的输出数据为数字量，故可以直送到单片机的 IO 口，而浑浊度、pH 值和电导率传感器的输出数据均为模拟量，故均需通过模数转换器(ADC)再送往主控芯片 STM32F407ZGT6 中，再将这些数据发往液晶显示屏。

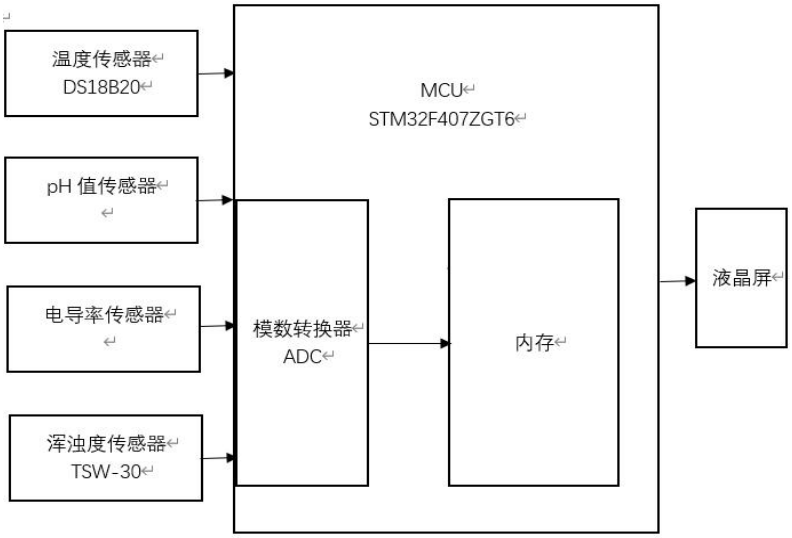


图 2.1 系统硬件框图

第三章 硬件电路设计

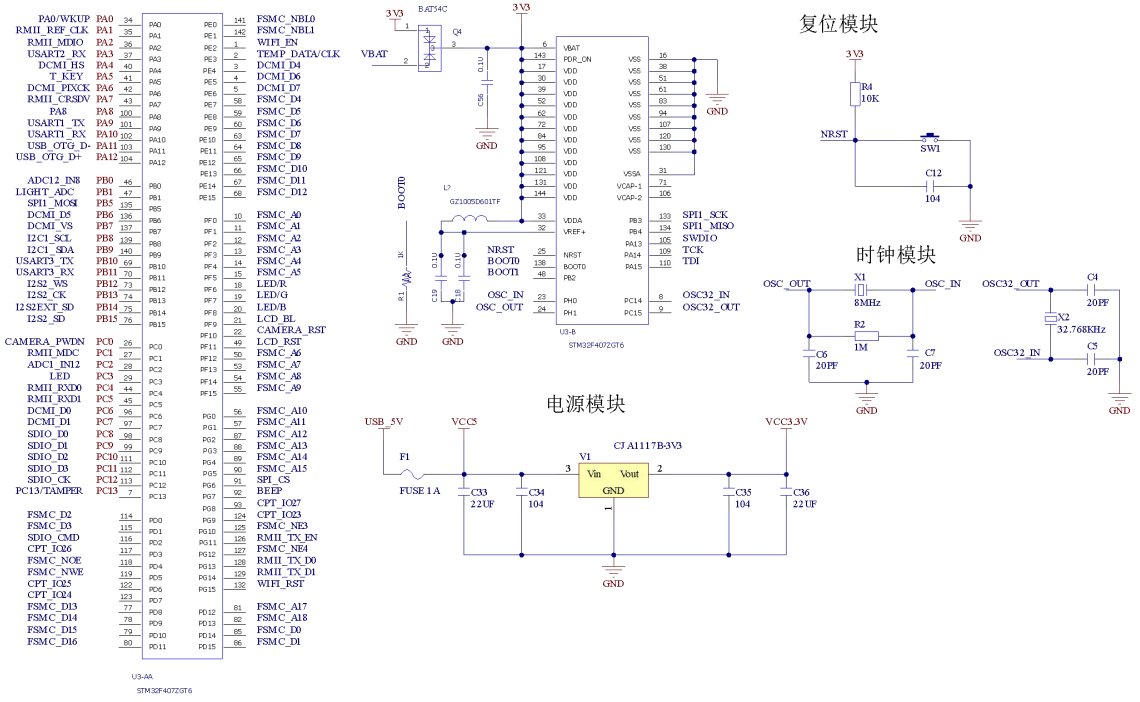
3.1 原理图绘制软件简介

本文绘制原理图的工具为 Altium Designer（AD），它是由澳大利亚公司 Altium Limited 开发出来的，在 2005 年之前，这款软件名叫 Protel，2005 年之后更名为 Altium Designer。AD 画原理图步骤：

1. 新建原理图文件。
2. 导入所要绘制的模块(如 STM32F407ZGT6 最小系统版)的元器件封装库。
3. 放置元器件、布线、合理设置器件参数。
4. 保存原理图。

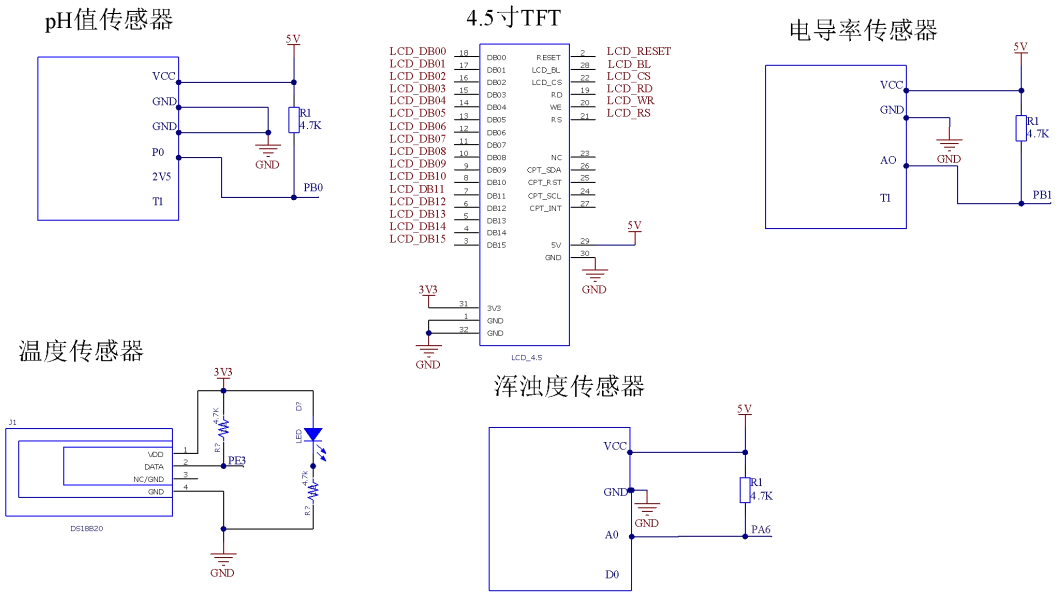
3.2 硬件电路总原理图

图 3.1（a）和图 3.1（b）为硬件电路总原理图，由单片机最小系统和传感器以及显示屏模块组成。单片机最小系统为使用最少的元器件组成使单片机可以正常工作的系统，故最小系统由电源、时钟、复位和主控芯片模块四部分组成。而为了实现检测并显示水质参数至少还需要四个传感器外加一个显示屏模块。



单片机最小系统

（a）STM32 最小系统



（b）传感器及显示屏模块

图 3.1 硬件电路总原理图

3.3 STM32 单片机主控模块

3.3.1 STM32 单片机简介

本文选用的单片机 STM32F407ZGT6 是 STM32F4 系列家族的一员,它采用高性能的 ARM Cortex-M4 内核，工作频率高达 168MHz。Cortex-M4 内核的特征是有一个单精度浮点运算单元(FPU)，可以用于进行高速浮点运算，它也可以执行所有的 DSP 指令，另外，它还有一个可以提升应用安全的内存保护单元(MPU)。

STM32F4 系列单片机拥有高速内存(Flash 容量高达 1MB,SRAM 容量高达 192KB)，后备存储空间 SRAM 容量高达 4KB，同时有许多 I/O 口和外设连接到 2 根 APB 总线，3 根 AHB 总线和一个 32 位多 AHB 总线矩阵。

STM32F4 系列单片机提供 3 个 12 位 ADC，2 个 DAC，一个低压 RTC，12 个 16 位通用定时器。它们还提供一个随机数发生器(RNG),同时，该系列单片机拥有下列标准和先进的通信接口：

• 3 个I^2C外设。

• 3 个 SPI 外设，2 个采用全双工的I^2S 外设。为了实现声音分类的精度，I^2S 外设的同步时钟可由一个专用的内部音频 PLL(Phase Locked Loops)或外部时钟提供。

- 4 个 USART 外设和 2 个 UART 外设
- 2 个 CAN 总线接口。
- 以太网和摄像头接口。
- 灵活的静态存储器控制接口 FSMC(Flexible Static Memory Controller)。

3.3.2 STM32 最小系统原理图

最小系统电路原理图如图 3.2 所示，包括 STM32F407ZGT6 主控芯片、复位电路、时钟电路、电源电路。

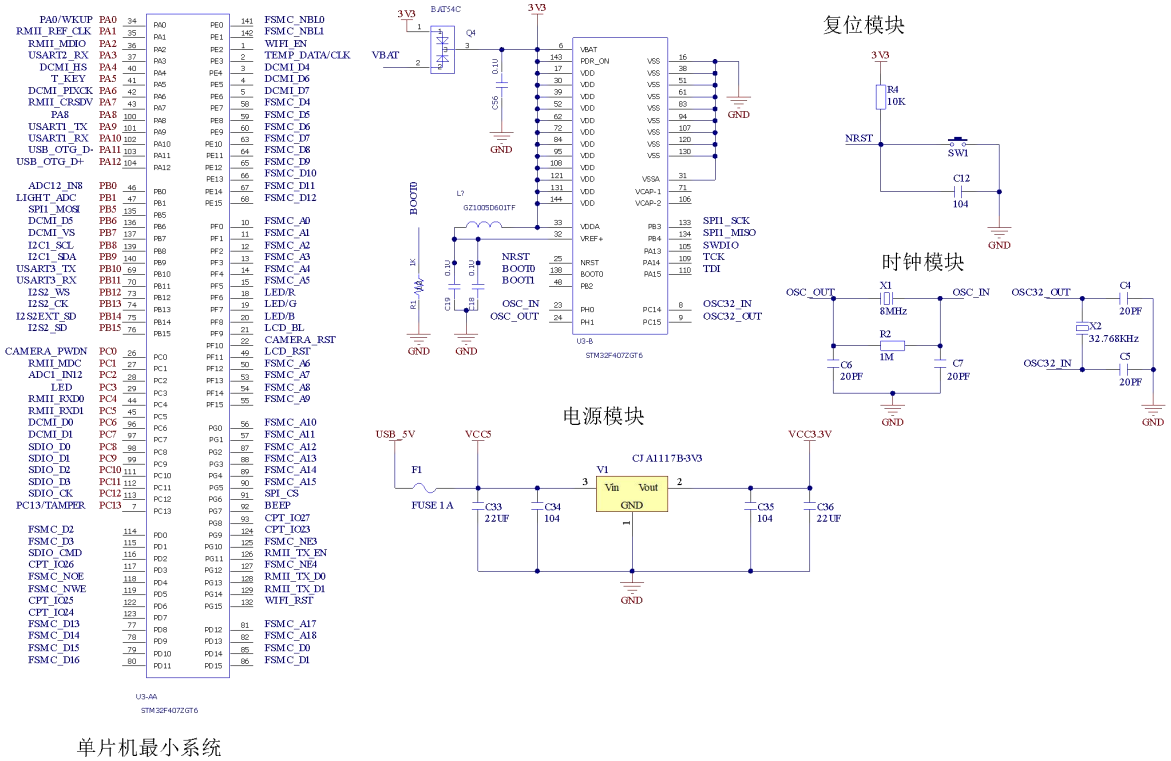


图3.2 STM32F407ZGT6最小系统原理图

图 3.3 为电源模块，电源模块负责整个系统包括单片机和传感器的供电，其中，USB_5V 为 USB 接口，它的输入电压为 5V，VCC5 给需要 5V 电压的模块供电，F1 为自恢复保险丝，用于保护 USB，CJA1117B-3.3 为稳压芯片，通过 USB_5V 输入的 5V 电压经过保险丝，再经过 VCC5 和稳压芯片，最终到达 VCC3.3V 输出口，可为系统提供 3.3V 的电压。

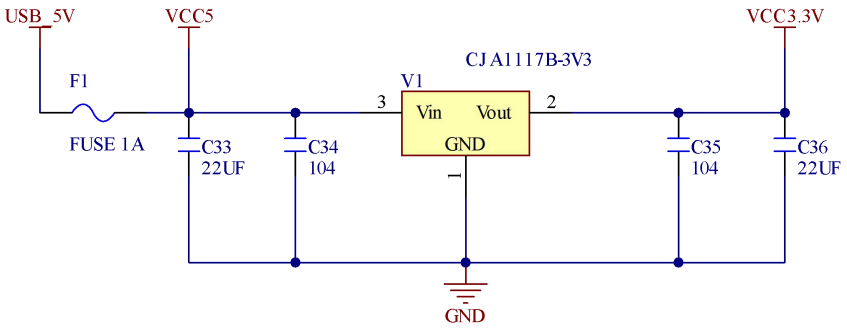


图3.3 电源模块

图 3.4 为时钟模块，OSC_out 和 OSC_IN 引脚分别接单片机的 PH0 和 PH1 引脚，OSC_out 和 OSC32_IN 引脚分别接单片机的 PC14 和 PC15 引脚。图中 X1 为晶振，晶振的全称为晶体振荡器，它的作用是作为单片机的脉搏，为其提供时钟信号以控制程序的运行，晶振分为有源晶振和无源晶振，图 3.4 中使用的 2 个晶振均为无源晶振，分别为 8MHz 和 32.768KHz,分别作为单片机的高速时钟输入和低速时钟输入。晶振两侧的引脚相当于电阻的两个引脚，分别接于单片机的两个相邻的晶振引脚上。晶振两侧所接电容的作用均为帮助晶振起振和稳定振荡信号，其容值一般选在 10～40pF 之间，这里我们选择经验值 20pF。

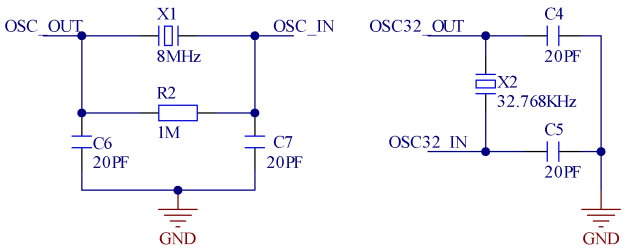


图 3.4 时钟模块

STM32 单片机采用低电平复位，图 3.5 为低电平有效的复位电路。复位电路的 NRST 接口与主控芯片的 25 号引脚 NRST 相连。当按键断开时，在上拉电阻的作用下，NRST 接口为高电平，相应地主控芯片 NRST 口也为高电平，主控芯片可以控制程序正常运行。当按键 SW1 按下时，复位电路的 NRST 接口与主控芯片的 NRST 接口同时变为低电平，由于电容两端的电压不能突变，故电容两端的电压会缓慢抬高，直到 NRST 接口变为高电平，复位过程结束，程序正常运行。

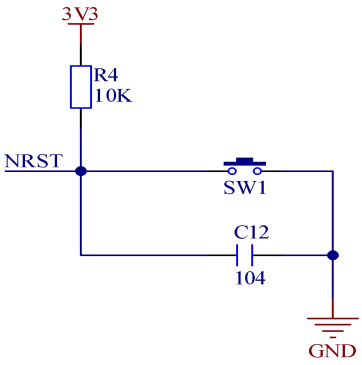


图 3.5 复位模块

3.4 DS18B20 温度传感器

3.4.1 DS18B20 简介

DS18B20 温度传感器提供 9～12 位(二进制)的测量精度，其可通过 1-Wire 接口总线收发数据。采用 1-Wire 总线连接时，中央处理器与 DS18B20 之间仅需通过一根总线来连接，这根总线可以完成所有的任务：供电、读写数据以及实现温度转换(不需要提供外部供电电源)。DS18B20 温度测量范围为-55℃～+125℃，并且在-10℃～+85℃的测量范围内精度为±0.5℃，足以满足大多数情况下测量范围与精度要求。它的一项重要特征是温度检测结果可直接在传感器模块内部转换为数字值，出厂默认的数字位数为 12 位，不需要通过 ADC 转换即可直接将测量结果传输给 MCU。温度传感器实物图如图 3.6 所示：

图 3.6 温度传感器实物图

3.4.2 DS10B20 电路原理图

图 3.7 为温度传感器 DS18B20 的电路原理图，传感器模块使用 3.3V 供电电压，故电源模块的 3V3 引脚与 DS18B20 的 VDD 引脚相连，而 DS18B20 的 DATA 引脚输出的数据直接为数字量，故可直接与主控芯片的 PE3 引脚相连，DS18B20 的 GND 引脚接地，NC/GND 引脚直接悬空。

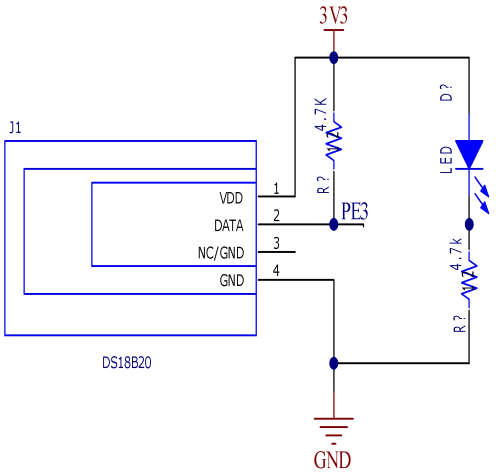


图 3.7　DS18B20 电路原理图

3.5 pH 值传感器

3.5.1 pH 值传感器简介

酸碱度(pH 值)是溶液的一种重要特性，本文所使用的 pH 值传感器模块由一个电压转换模块及 pH 复合电极组成电压转换模块通过 BNC 接头与 PH 复合电极相连。pH 复合电极的输出信号为 mV 级的电压信号，单片机无法识别并处理，而通过电压转换模块进行放大，就可以输出 0～3V 或 0～5V 输出信号，再通过 ADC 对此输出模拟信号进行采集得到数字信号。调节电位器旋钮可以改变放大倍数。模块电源为+5.00V,模块尺寸为 37mm×28mm，测量范围为 0～14PH，测量温度为 0～60℃，测量精度为±0.01PH(25℃)，响应时间≤1min。图 3.8 为 pH 传感器实物图：

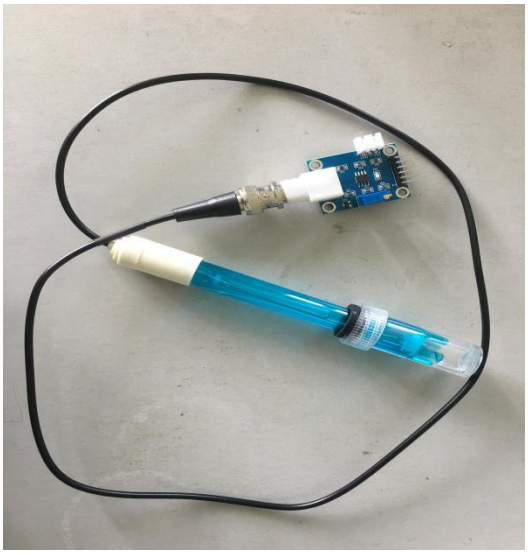


图 3.8 pH 传感器模块实物图

表 3.1 电压转换模块基本参数

序号	引脚定义	功能描述	备注
1	VCC	供电电压正极，5V	
2	GND	供电电压负极	
3	GND	模拟信号输出负极	
4	P0	模拟信号输出正极	输出电压范围 0~5V
5	2V5	基准电压 2V5 输出口	测试用，不外接电源
6	T1	温度传感器 DS18B20 信号输出口	可通过软件进行温度补偿

3.5.2 pH 值传感器原理图

图 3.9 为 pH 值传感器原理图，pH 值传感器供电电压为 5V，输出模拟量，VCC 与电源模块的 5V 引脚相连，P0 引脚接主控芯片的 PB0 引脚，2V5 为基准电压 2V5 输出口,用于测试，T1 为温度补偿模块接口，本系统未使用 2V5 和 T1 引脚，故使其悬空。

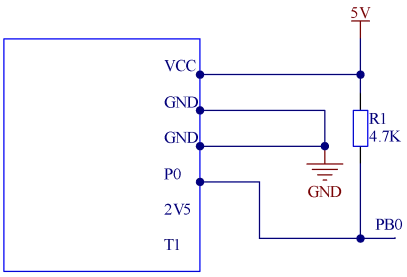


图 3.9 pH 值传感器原理图

3.6 浑浊度传感器

3.6.1 浑浊度传感器简介

浑浊度传感器模块由一个电压转换模块和浊度传感器 TSW-30 组成，电压转换模块通过 3pin XH-2.54 接头与浊度传感器 TSW-30 连接，电压转换模块的作用也是将 TSW-30 的输出电压转换到单片机可识别的范围内。本传感器模块同时具有数字量和模拟量输出接口，本文使用模拟量输出接口。模拟量可通过主控芯片中的 ADC 采集为数字量，从而得知所取水样的浑浊度。浑浊度传感器在出厂时会由厂家设定一默认阈值，用户也可在程序中自己设定阈值，一旦浑浊度达到预先设定的阈值，传感器模块上的绿色指示灯就会被点亮，模块输出也会发生高低电平转换，观察指示灯的亮和灭或通过监测模块输出高低电平的变化就能发现水样的浑浊度是否超标。图 3.10 为浑浊度传感器模块：

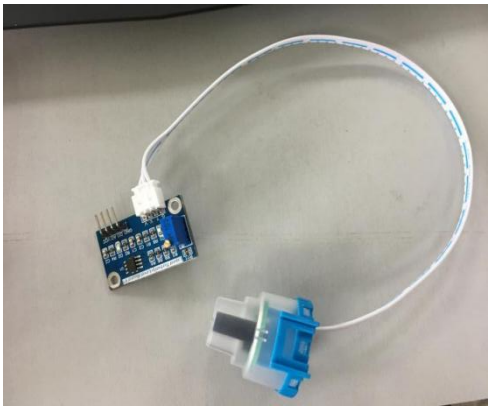


图 3.10 浑浊度传感器模块

浑浊度传感器模块参数如表 3.2 所示：

表 3.2 浑浊度传感器基本参数

序号	引脚定义	功能描述	备注
1	VCC	供电电压，5V	
2	AO	模拟信号输出	输出电压范围 0～5V
3	DO	数字信号输出	输出 1 或 0 代表高低电平
4	GND	供电电压负极	与开发板 GND 引脚相连

3.6.2 浑浊度传感器电路原理图

图 3.11 为浑浊度传感器电路原理图，浑浊度传感器为 5V 电压供电，故 VCC 与电源模块的 5V 引脚相连，AO 引脚为模拟量输出口，直接与单片机的 PA6 引脚相连，以供 ADC 采集数据，DO 为数字量输出口，本系统不使用数字信号输出口，故 DO 引脚悬空。

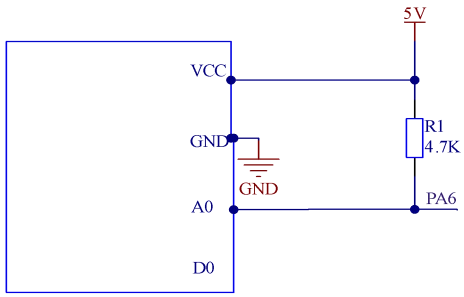


图 3.11 浑浊度传感器电路原理图

3.7 电导率传感器

3.7.1 电导率传感器简介

专业的电导率检测设备价格昂贵，且缺少相关技术资料，本文所用电导率传感器模块如图 3.12 所示，由一个电压转换模块和电导率电极组成，电压转换模块通过 BNC 接口与电极相连，其作用与 pH 传感器及浑浊度传感器模块中的电压转换模块均相同。本传感器模块具有价格低廉和使用方便等优点，可用于评估水质、水产养殖和环境水样检测等领域。

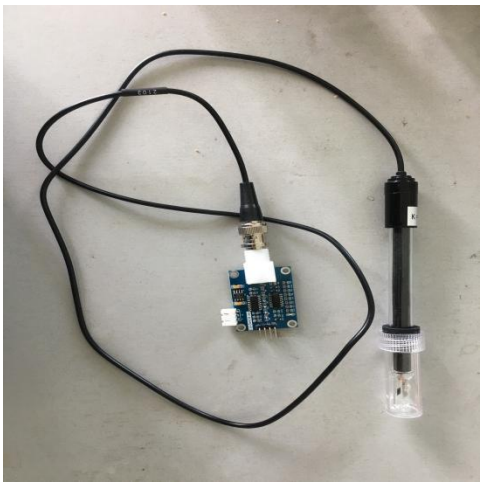


图 3.12 电导率传感器模块实物图

电导率传感器模块参数如表 3.3 所示：

表 3.3 电导率传感器基本参数

序号	引脚定义	功能描述	备注
1	VCC	供电电压正极，5V	不使用 3V3
2	GND	供电电压负极	与开发板 GND 引脚相连
3	T1	温度传感器数据输出口	悬空
4	AO	模拟量输出口	输出电压范围 0～3.4V

3.7.2 电导率传感器电路原理图

图 3.13 为电导率传感器电路原理图，电导率传感器使用 5V 供电电压，故 VCC 引脚与电源模块的 5V 引脚相连，GND 引脚接地，AO 引脚为传感器模拟量输出口，与单片机 PB1 引脚相连，以供 ADC 采集，T1 引脚为温度传感器数据输出口，本系统有单独的温度传感器 DS18B20，故不使用 T1，令其处于悬空状态。

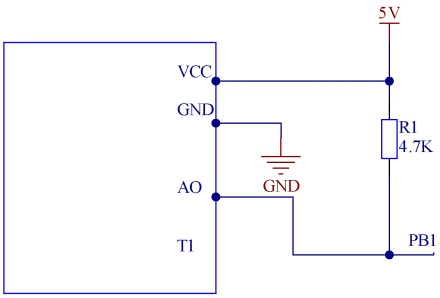


图 3.13 电导率传感器电路原理图

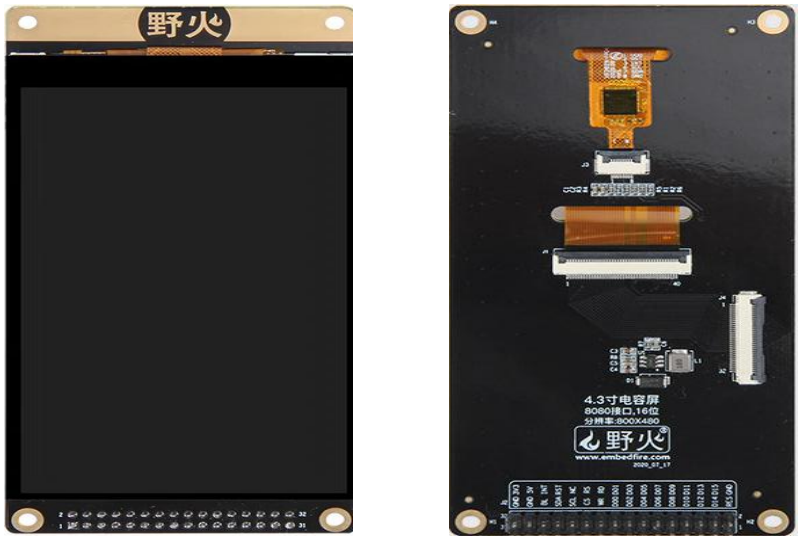
3.8 液晶显示模块

3.8.1 液晶显示屏简介

本文所用的显示屏为自带NT35510液晶控制器的野火4.3寸薄膜晶体管液晶显示屏(TFT LCD)，屏幕分辨率为800×480，它由以下三部分组成：液晶显示面板、电容触摸面板和PCB底板。电容触摸面板上有触摸控制芯片，PCB底板上带有液晶控制器。表3.4为液晶显示屏硬件参数，图3.14（a）和3.14（b）分别为液晶屏实物图的正面和背面。

表 3.4 液晶显示屏硬件参数表

PCB 尺寸	124*62mm
TFT	4.3 寸、分辨率 480*800
触摸	5 点触控电容屏
接口	16 位并口 8080 时序
供电	5V 和 3.3V，需同时供电
背光	200mA 即可点亮
帧率	75 帧
控制器	自带 NT35510 控制器
颜色格式	RGB565
触摸芯片	GT917S



（a）正面　　（b）背面

图 3.14 液晶屏实物图

图 3.15 为液晶屏与主控芯片连接示意图，本文所使用的液晶显示屏自带液晶控制器 NT35510，且 NT35510 中含显存，当需要使用显示屏时，STM32 通过 FSMC 外设模拟 8080 时序来传送要显示的字符数据到 NT35510 的显存中，NT35510 再通过 RGB 接口向液晶面板传输像素数据。

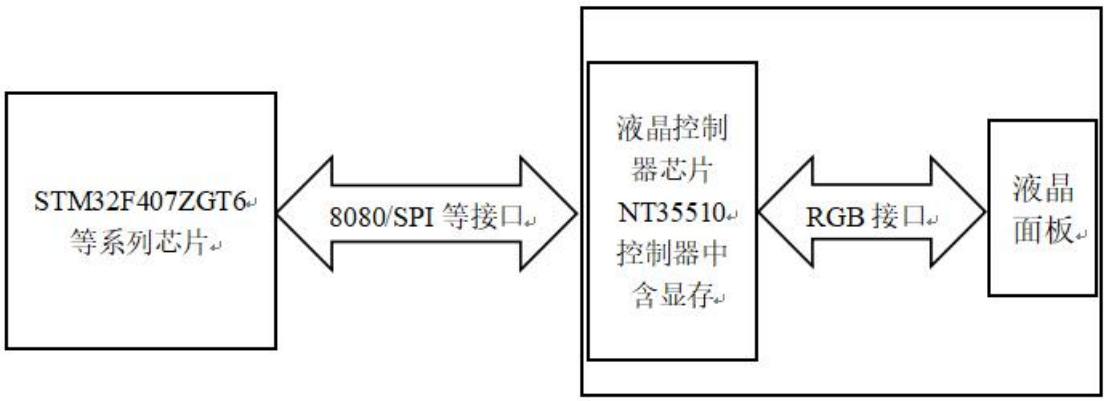


图 3.15 液晶屏与主控芯片连接示意图

3.8.2 液晶显示屏电路原理图

液晶显示屏电路原理图如图 3.16 所示，由于主控芯片采用 FSMC 外设模拟 8080 时序与液晶控制器 NT35510 进行通信，故显示屏引出的这些信号线实际上为 8080 通讯接口，其中 LCD_DB[15:0]引脚用于传输数据信号，LCD_RD 引脚用于传输读数据信号，低电平有效，LCD_RS 引脚用于选择传输数据/命令信号，本引脚为高电平时，

表示 LCD_DB[15:0]传输的是数据，反之表示传输的是命令。LCD_RESET 引脚用于传输复位信号，低电平有效，LCD_WR 引脚用于传输写数据信号，低电平有效。LCD_CS 用于传输片选信号，低电平有效。LCD_BL 用于传输背光信号，低电平有效。

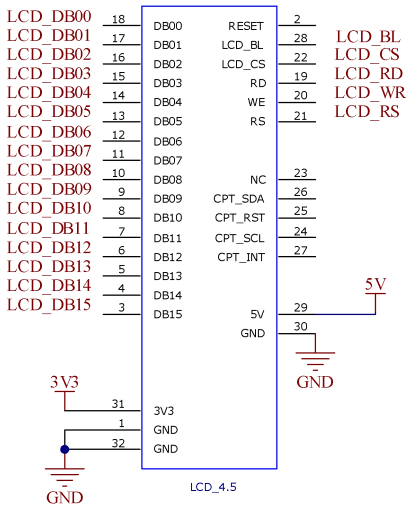


图 3.16 液晶显示屏电路原理图

第四章 软件程序设计

4.1 开发软件简介

本文所使用的的软件开发环境是 Keil MDK（Microcontroller Development Kit），它主要包括 MDK-Core 以及 Arm C/C++编译器。其中，MDK-Core 基于μVsion，它支持所有的 Cortex-M 设备(本文使用的设备为 Cortex-M4 内核)，包括最新版的 ARMv8-M 架构。ARM C/C++ 编译器包括汇编器、链接器以及高性能的运行环境库。用户也可以随时将软件包添加到 MDK-Core 中，从而支持新设备，并使中间件更新独立于工具链之外。

使用 Keil 软件编写源程序，在创建工程时需要根据所使用的芯片型号和仿真工具来设置下载和仿真环境，其余步骤与编写一个普通的 C 语言源程序相同，如图 4.1 所示：

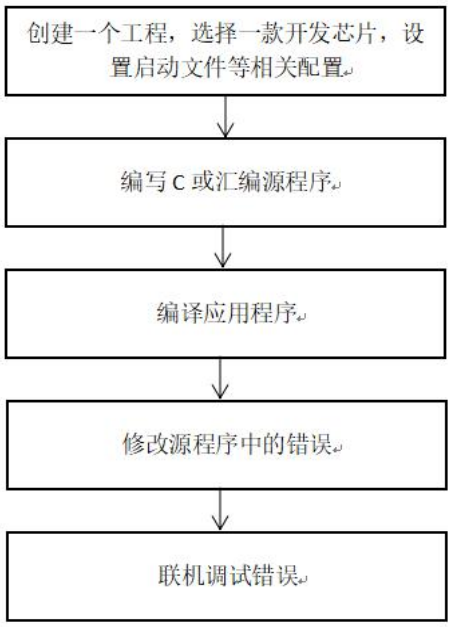


图 4.1 KEIL 软件开发程序流程图

4.2 系统主程序设计

本文所设计的水质参数检测系统主要检测四种参数：温度、浑浊度、pH 值和电导率。由于对这四个参数的检测并显示是相对独立的过程，故采用模块化程序设计方法。同时由于温度检测模块驱动程序的设计方法与其他三种区别较大，pH 值、电导

率、浑浊度检测模块的源代码设计方法基本相同，而液晶显示方法原理均相同，故按照温度→pH 值→电导率→浑浊度，以及“先测量、后显示”的顺序编写源程序。

主函数流程图如图 4.2 所示：

1. 声明并初始化主函数中需要的用到的变量：如定义一个显示缓冲区变量 dis_buf，将要显示地字符插入 dis_buf 中即可。

2. 初始化系统定时器、液晶控制器 NT35510、ADC、DMA、LED 端口，初始化各个传感器检测模块。

3. 读取温度并送往显示缓冲区中，显示温度。

4. 浑浊度、pH 值和电导率检测模块程序设计原理均相同：ADC 端口实时采集传感器模块输出的模拟量，并转化为数字量，再将数字量转化为可用于显示的字符，最后将要显示的字符插入到显示缓冲区 dis_buf 中即可显示。

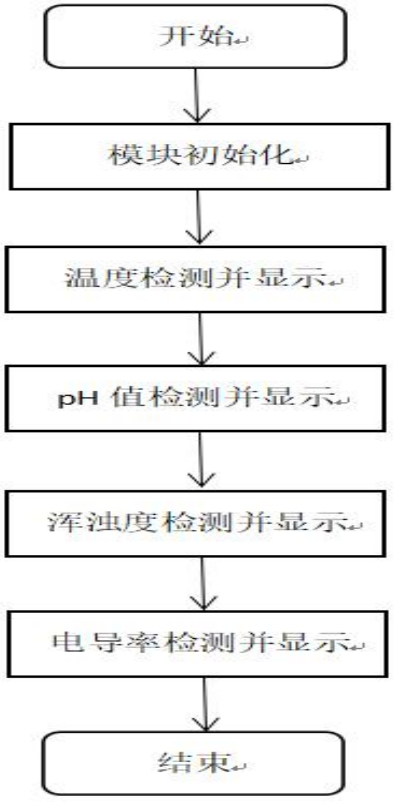


图 4.2 主函数流程图

主函数部分程序如下所示：

```
for( ; ; )
{
```

```
//1.循环检测温度并显示
temperature=DS18B20_Get_Temp();
sprintf( (char*)dis_buf, "T:%0.3f degree Celsius", temperature);
NT35510_DispStringLine_EN(LINE(1),dis_buf);
//2.循环检测 pH 并显示
ADC_ConvertedValueLocal[0] =(float) ADC_ConvertedValue[0]/4096*3.3; //读取转换的 AD 值
PH_Value=-5.7541*ADC_ConvertedValueLocal[0]+16.654;
if(PH_Value<=0.0){PH_Value=0.0;}
if(PH_Value>=14.0){PH_Value=14.0;}
/*显示电压*/
Tx_PH[0]=(int)(PH_Value*100)/1000+'0'; //将数字转化为对应字符串
Tx_PH[1]=(int)(PH_Value*100)%1000/100+'0';
Tx_PH[2]='.';
Tx_PH[3]=(int)(PH_Value*100)%100/10+'0';
Tx_PH[4]=(int)(PH_Value*100)%10+'0';
Tx_PH[5]='\0';    //字符串结束符
NT35510_DispStringLine_EN(LINE(3),"2. pH test ");
sprintf((char*)dis_buf,"pH:%s",Tx_PH);
NT35510_DispStringLine_EN(LINE(4),dis_buf);
```

4.3 DS18B20 温度传感器模块设计

温度传感器 DS18B20 需要单独编写驱动程序。编写 DS18B20 的驱动程序与编写其他所有外设驱动程序的方法类似：首先要配置 DS18B20 用到的通用输入输出接口口(GPIO 口)，即定义一个外设初始化结构体，并对这个初始化结构体的成员如引脚和输入输出方式等一一赋值。之后还需要编写一些用于主控芯片与 DS18B20 之间通信的函数，如用于主机(MCU)给从机(DS18B20)发送复位脉冲的函数、检测从机给主机发送存在脉冲的函数、读写 DS18B20 的函数以及读取温度的函数。另外，DS18B20 的直接输出为数字量(二进制数据)，且其数据精度可调(可选 9 位、10 位、11 位或 12

位)，默认读取精度为 12 位。下表显示了测量温度与输出数据之间的关系：

表 4.1 测量温度与输出数据之间的关系

S	S	S	S	S	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}

上表假设输出精度为 12 位，共 16 位(二进制补码)，高 5 位为符号位，中间 7 位为整数位，低 5 位为小数位。其中如果将 DS18B20 的输出精度配置为更低的位数，有些位将会置 0。

温度传感器模块的读取温度函数程序框图如图 4.3 所示：

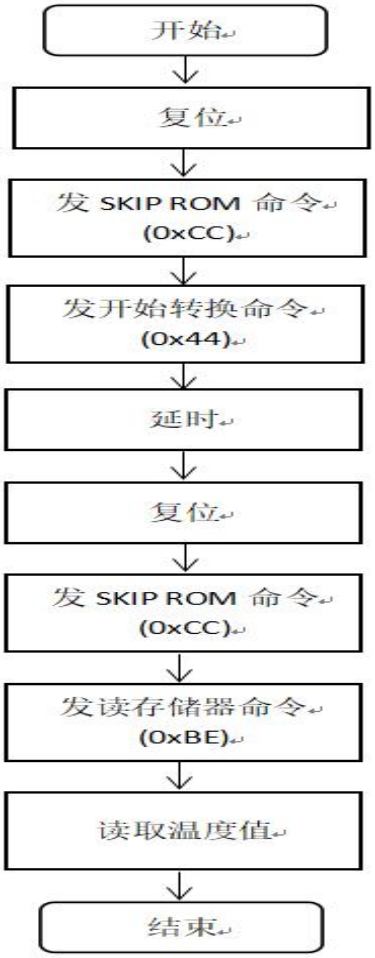


图 4.3 读取温度函数程序框图

温度传感器模块部分程序如下所示：

```
float DS18B20_Get_Temp(void)
{
uint8_t tpmsb, tplsb;
```

```
short s_tem;
float f_tem;
DS18B20_Rst();
DS18B20_Presence();
DS18B20_Write_Byte(0XCC);                    /* 跳过 ROM */
DS18B20_Write_Byte(0X44);                    /* 开始转换 */
DS18B20_Rst();
DS18B20_Presence();
DS18B20_Write_Byte(0XCC);                    /* 跳过 ROM */
DS18B20_Write_Byte(0XBE);                    /* 读温度值 */
tplsb = DS18B20_Read_Byte();          //读取温度低字节
tpmsb = DS18B20_Read_Byte();          //读取温度高字节
s_tem = tpmsb<<8;
s_tem = s_tem | tplsb;
if( s_tem < 0 )          /* 负温度 */
f_tem = (~s_tem+1) * 0.0625;
else
f_tem = s_tem * 0.0625;
return f_tem;
}
```

4.4 pH 值、浑浊度、电导率传感器模块设计

如前所述，浑浊度、pH 值和电导率传感器模块检测原理均相同：传感器模块输出模拟量，该模拟量送往主控芯片中的 ADC，ADC 对其进行抽样、量化、编码生成数字量，再将生成的数字量通过 DMA 传送方式送往内存中存储，最后将其插入显示缓冲区中进行显示。使用 ADC 进行模数转换和 DMA 传送方式分别需要用到芯片中的 ADC 外设和 DMA 外设，因而，对这三个模块的源程序设计的主要任务就是设计芯片中 ADC 外设的驱动程序以及 DMA 外设的驱动程序，ADC 外设和 DMA 外设驱动程序框图如图 4.4 所示：

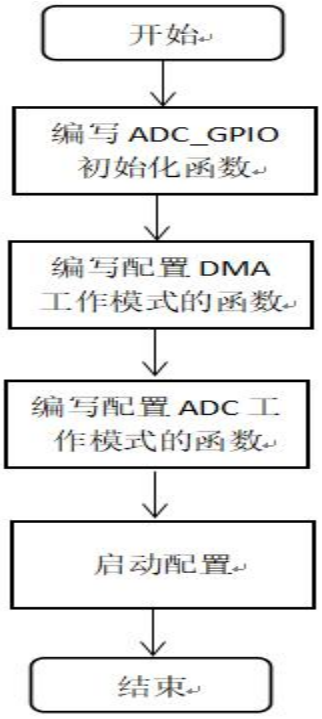


图 4.4 ADC 外设和 DMA 外设驱动程序的程序框图

DMA 外设的部分驱动程序如下所示：

```
{
// ADC1 使用 DMA2，数据流 0，通道 0，这个是手册固定死的
// 开启 DMA 时钟
RCC_AHB1PeriphClockCmd(RHEOSTAT_ADC_DMA_CLK, ENABLE);
// 外设基址为：ADC 数据寄存器地址
DMA_InitStructure.DMA_PeripheralBaseAddr = RHEOSTAT_ADC_DR_ADDR;
// 存储器地址，实际上就是一个内部 SRAM 的变量
DMA_InitStructure.DMA_Memory0BaseAddr = (u32)ADC_ConvertedValue;
// 数据传输方向为外设到存储器
DMA_InitStructure.DMA_DIR = DMA_DIR_PeripheralToMemory;
// 缓冲区大小为，指一次传输的数据量
DMA_InitStructure.DMA_BufferSize = RHEOSTAT_NOFCHANEL;
// 外设寄存器只有一个，地址不用递增
DMA_InitStructure.DMA_PeripheralInc = DMA_PeripheralInc_Disable;
```

```
// 存储器地址固定
DMA_InitStructure.DMA_MemoryInc = DMA_MemoryInc_Enable;
//外设数据大小为半字，即两个字节
DMA_InitStructure.DMA_PeripheralDataSize =
DMA_PeripheralDataSize_HalfWord;
...
DMA_Cmd(RHEOSTAT_ADC_DMA_STREAM, ENABLE);
}
```

ADC 外设的部分驱动程序如下所示：

```
{
// 开启 ADC 时钟
RCC_APB2PeriphClockCmd(RHEOSTAT_ADC_CLK , ENABLE);
// ------------------ADC_Common 结构体参数初始化------------------------
// 独立 ADC 模式
ADC_CommonInitStructure.ADC_Mode = ADC_Mode_Independent;
// 时钟为 fpclk_x 分频
ADC_CommonInitStructure.ADC_Prescaler = ADC_Prescaler_Div4;
// 禁止 DMA 直接访问模式
ADC_CommonInitStructure.ADC_DMAAccessMode =
ADC_DMAAccessMode_Disabled;
// 采样时间间隔
ADC_CommonInitStructure.ADC_TwoSamplingDelay =
ADC_TwoSamplingDelay_20Cycles;
ADC_CommonInit(&ADC_CommonInitStructure);
// ------------------ADC_Init 结构体参数初始化-------------------------
ADC_StructInit(&ADC_InitStructure);
// ADC 分辨率
ADC_InitStructure.ADC_Resolution = ADC_Resolution_12b;
// 扫描模式，多通道采集需要
```

```
...
// 使能 ADC
ADC_Cmd(RHEOSTAT_ADC, ENABLE);
//开始 ADC 转换，软件触发
ADC_SoftwareStartConv(RHEOSTAT_ADC);
}
```

4.5 液晶模块设计

本文所使用的显示屏为野火 4.3 寸液晶屏，内置液晶控制器 NT35510，NT35510 中内置显存，当我们想要显示数据时，其实就是将数据写入显存中，STM32 是使用 FSMC 外设来模拟 8080 时序来控制液晶屏的。

STM32F407ZGT6 内含 FSMC (Flexible Static Memory Controller)，它可以用来管理对用于单片机存储容量进行扩展的存储器，包括 SRAM、NOR Flash 以及 NAND Flash 等。由于 MCU 控制液晶屏显示实际上是往液晶控制器的显存(通常为 SRAM)中写入像素数据，故 FSMC 也可以用来控制液晶屏。

图 4.5 为 FSMC 外设框图。由于 FSMC 可以控制不同种类的存储器，相应地控制引脚也比较多，在图 4.5 中，右侧为 FSMC 外设的控制引脚，其中，NOR/PSRAM 信号左边的花括号括起来的引脚 FSMC[4:1]及 FSMC_NL 等引脚为 NOR Flash 以及 PSRAM 的控制器专用引脚，共享信号左边的花括号括起来的引脚为所有类型存储器的共用引脚。

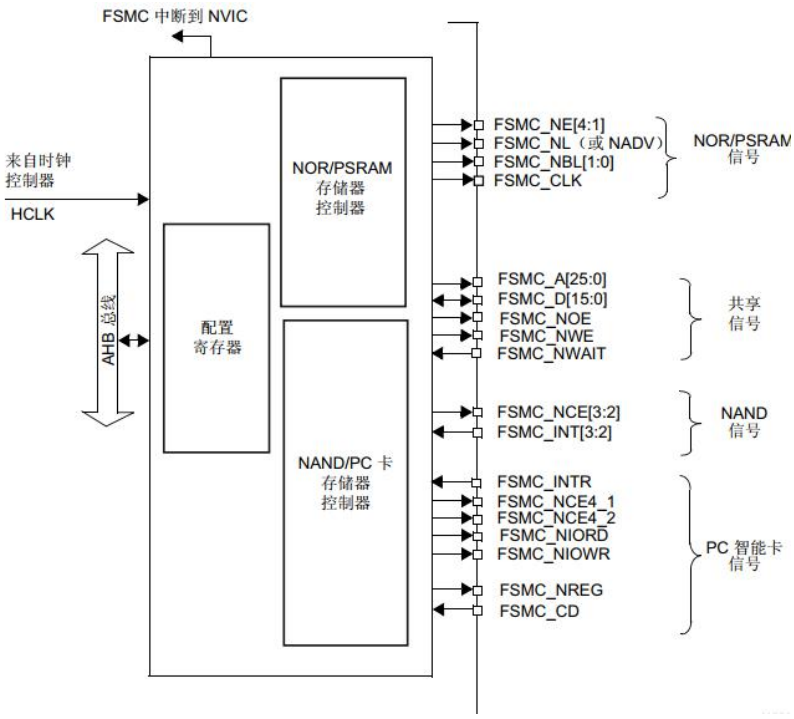


图 4.5 FSMC 外设框图

采用 FSMC 外设控制访问存储器与采用I^2C 和 SPI 访问存储器的方式不同，后二者是给出一个要访问的存储器地址，再往地址中读写数据，地址和数据都要存储在存储器中的不同地址单元，读写数据时还要使用代码发送命令。而使用 FSMC 访问存储器时，通过一种地址映射的方式将 STM32 内核地址空间中的 0x60000000～0x6FFFFFFF 这段地址范围映射到外部扩展的存储器中，如图 4.6 所示，FSMC 外设将内核地址空间映射到外部存储器中，若我们在程序中设一地址指针指向内核地址空间 0x60000000～0x6FFFFFFF，则我们通过地址指针对该地址范围读写数据时，FSMC 会自动进行转换，并实现对外部存储器对应存储区域的读写。

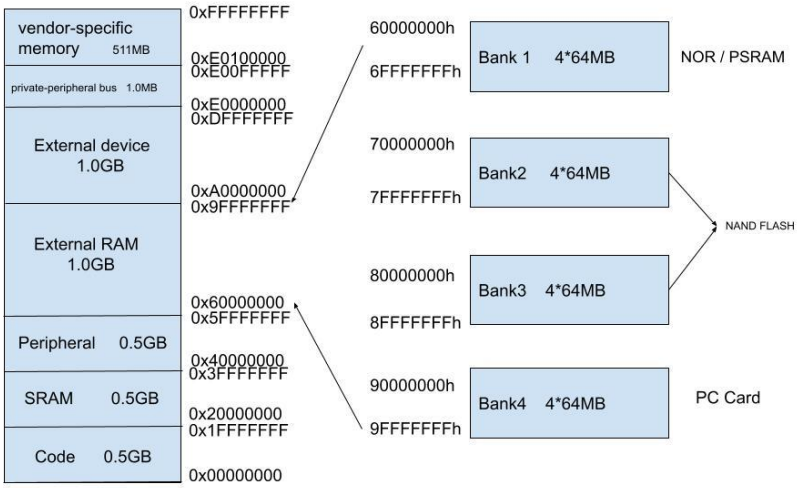


图 4.6 FSMC 外设对内核地址的映射

8080 接口是由 Intel 公司推出的一种 LCD 接口通讯方式，MCU 可以通过 8080 接口快速访问液晶显示屏。MCU 通过 8080 接口向液晶屏（实际上为液晶控制器）的显存中发送命令与数据。同I^2C和 SPI 总线接口类似，8080 总线接口也有它对应的读写时序。图 4.7 为 8080 总线写时序图(本文只用到了写时序)，当 MCU 要向 LCD 写入命令(带参数)时，分为两个阶段传输。第一阶段传输命令地址，先将片选信号 CS 拉低，将数据/命令选择信号拉低，表示写入 LCD 的是命令地址，写信号 WR 为低电平，读信号 RD 为高电平，表示读写方向为 MCU→LCD,同时在数据线 DB[23:0]上传输命令地址。第二阶段传输命令参数，将 DC 信号置为高电平，表示传输的是参数(数据)，其余信号与第一阶段相同。当 MCU 要向 LCD 写入像素数据时，传输过程与上述写入命令的第二阶段相同。

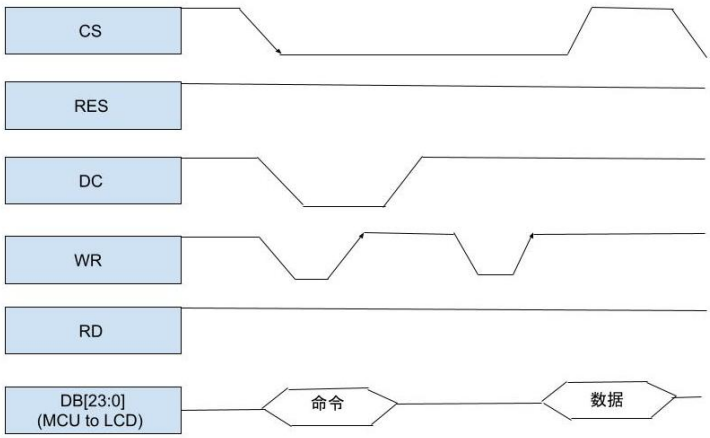


图 4.7 为 8080 总线写时序图

对于液晶控制器模块的程序设计主要是编写 NT35510 的驱动程序和 FSMC 外设的驱动程序。液晶控制器模块的程序框图如图 4.8 所示：

图 4.8 液晶控制器模块的程序框图

液晶控制器模块的部分驱动程序如下所示：

```
/**
 * @brief   NT35510 初始化函数，如果要用到 LCD，一定要调用这个函数
 * @param   无
 * @retval  无
 */
void NT35510_Init ( void )
{
NT35510_GPIO_Config ();
NT35510_FSMC_Config ();
NT35510_Rst ();
NT35510_REG_Config ();
//设置默认扫描方向，其中 6 模式为大部分液晶例程的默认显示方向
```

```
NT35510_GramScan(LCD_SCAN_MODE);
NT35510_Clear(0,0,LCD_X_LENGTH,LCD_Y_LENGTH);/* 清屏，显示全黑 */
NT35510_BackLed_Control ( ENABLE );          //点亮 LCD 背光灯
}
```

第五章 系统制作与调试

5.1 系统硬件制作

将液晶显示屏插入开发板的排母上，并将四个传感器模块的各个引脚分别连接到开发板上对应引脚处，连接状况如下：温度传感器模块: VCC→3.3V; GND→GND; DO→PE3; PH 传感器模块: VCC→5V; GND→GND; P0→PB0; EC 传感器模块: VCC→5V; GND→GND; A0→PB1; 浑浊度传感器模块: VCC→5V; GND→GND; A0→PA6; 硬件连接实物图如图 5.1 所示：

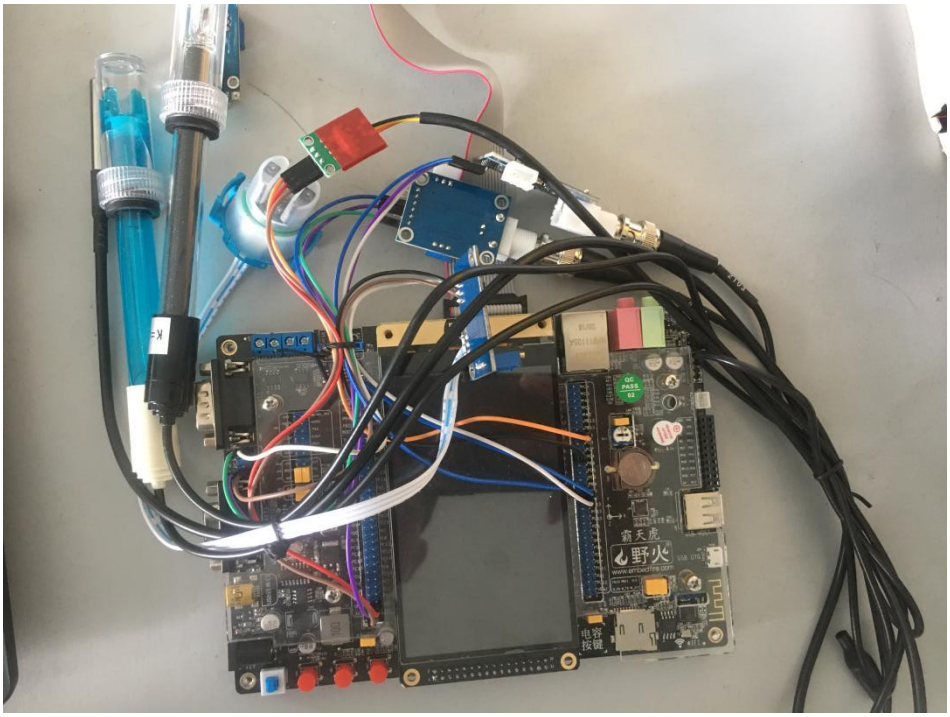


图 5.1 硬件连接实物图

5.2 软件调试

在程序调试时，共进行 5 个检测实验，实验 1～4 用于调试各传感器模块源程序，实验 5 为综合检测实验。

实验 1 检测温度传感器模块，将程序下载到开发板中，再将温度传感器探头分别放置到热水与冷水中，打开电源并按下复位按键，结果分别如图 5.2 和 5.3 所示。检测 66℃热水以及冷水水温时，液晶显示屏中温度示数分别为 63.188 和 24.125，说明温度传感器模块检测效果较好。

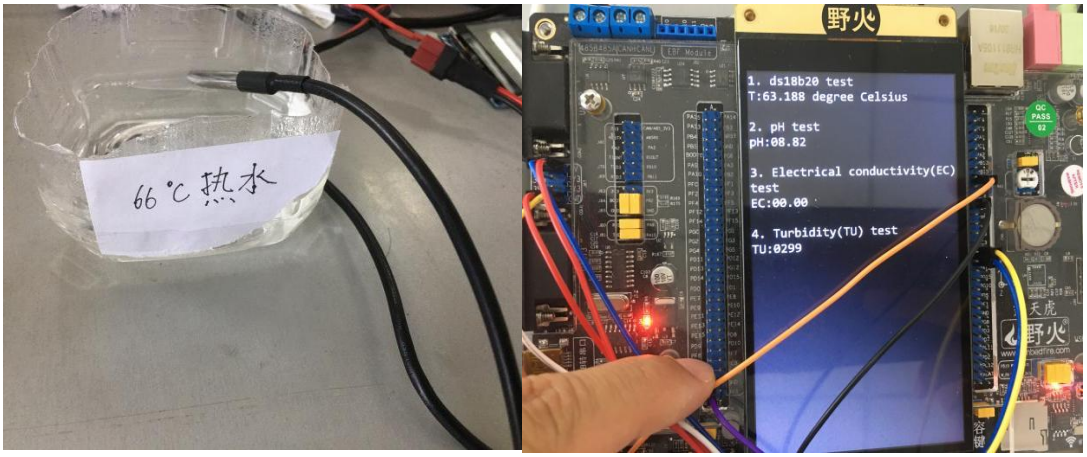


图 5.2 检测热水水温

图 5.3 检测冷水水温

实验 2 检测 pH 传感器模块，将 pH 传感器探头分别放置到 pH 值 4.00 为与 9.18 的溶液中中，打开电源并按下复位按键，结果分别如图 5.4 和 5.5 所示。检测检测 pH 值分别为 4.00 和 9.18 的溶液 pH 时，显示屏的 pH 值示数分别为 5.08 和 9.05，酸性溶液检测结果与正确值差 1，说明 pH 传感器模块检测酸性溶液时还有一定误差，检测碱性溶液时相差 0.13，说明 pH 传感器模块检测碱性溶液精度较高。

图 5.4 检测 pH 为 4.00 的溶液 PH 值

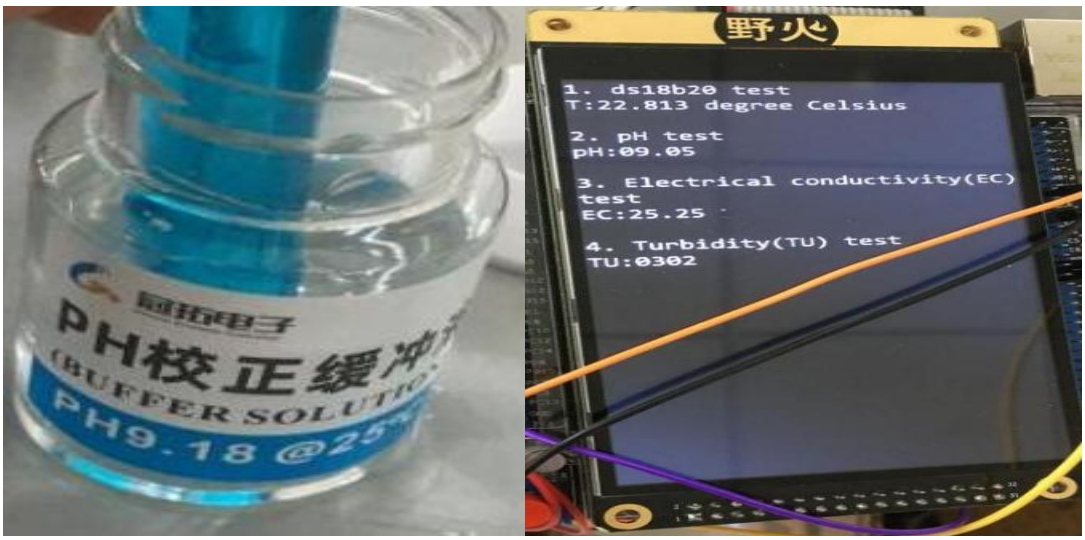


图 5.5 检测 pH 为 9.18 的溶液 PH 值

实验 3 检测浑浊度传感器模块，将浑浊度传感器探头分别放置到墨水以及清水中，打开电源并按下复位按键，结果分别如图 5.6 和 5.7 所示。检测墨水和清水浑浊度时，显示屏的浑浊度示数分别为 2506 和 191，墨水浑浊度示数远高于清水浑浊度，检测效果较好。

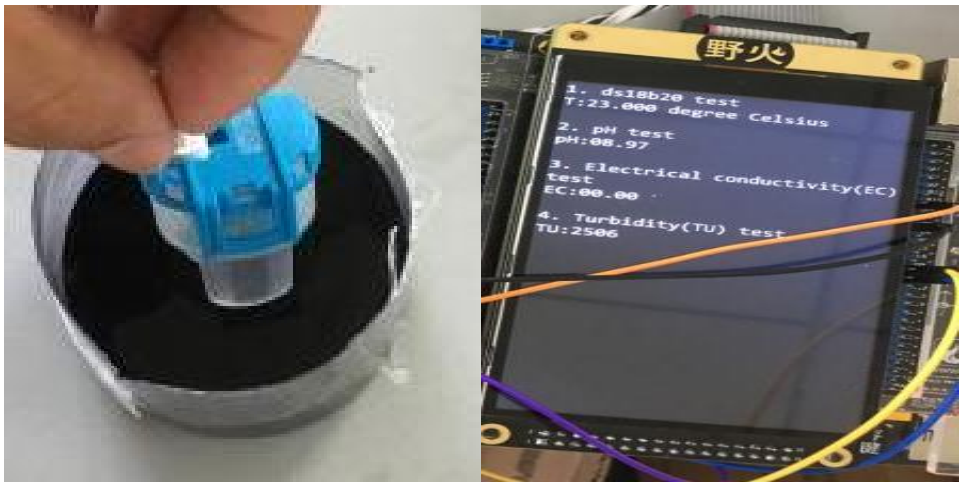


图 5.6 检测墨水浑浊度

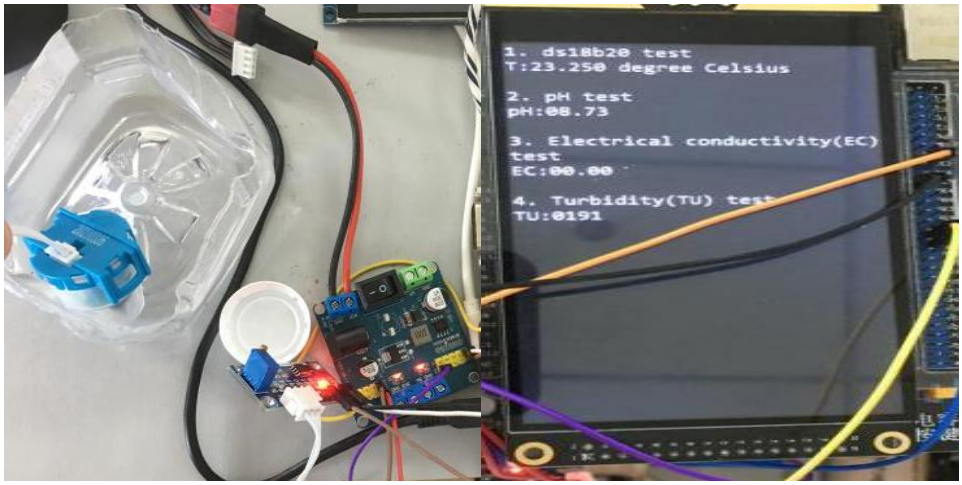


图 5.7 检测清水浑浊度

实验 4 检测电导率传感器模块，将电导率传感器探头分别放置到电导率为 1413μs/cm 和 12.88ms/cm 的溶液中，打开电源并按下复位按键，结果分别如图 5.8 和 5.9 所示。检测电导率为 1413μs/cm 和 12.88ms/cm 的溶液电导率时，显示屏的电导率示数分别为 1.01 和 11.11，检测效果较好。

图 5.8 检测电导率为 1413μs/cm 溶液的电导率

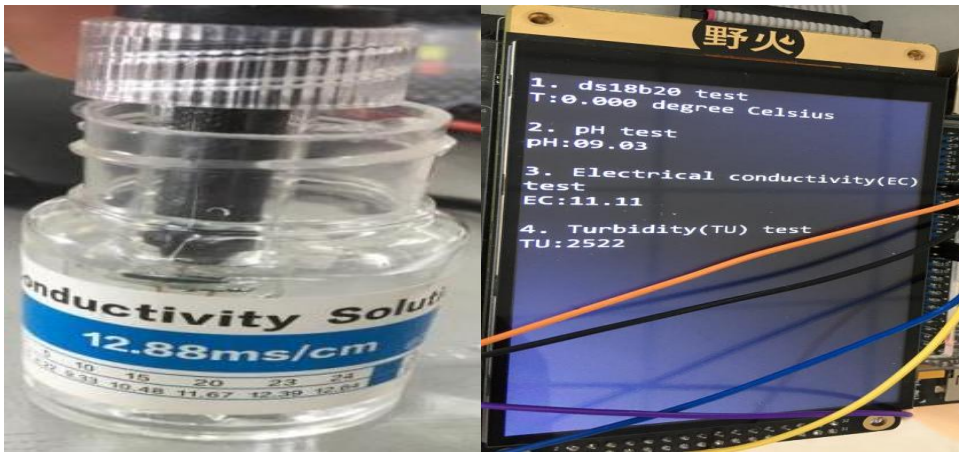


图 5.9 检测电导率为 12.88ms/cm 溶液的电导率

实验 5 为对掺杂了多种溶液的混合溶液的各个参数同时检测。结果如图 5.10 所示：

图 5.10 综合检测

第六章 总结与展望

本次毕业设计历时一个学期，经过查阅资料、方案设计、购买元器件、硬件系统设计、原理图绘制、查阅器件手册、编写源程序等多个步骤，设计出了一种多参数水质监测系统。该系统采用 STM32F407ZGT6 作为主控芯片，结合温度传感器、pH 值传感器、电导率传感器、浑浊度传感器及 STM32 中内嵌的外设（ADC、DMA、FSMC）等，最终实现了以下功能：

1. 能够实时检测水样的温度、浑浊度、PH 值和电导率。

2. 液晶显示屏能够实时显示检测数据，方便用户实时观察水质。

经测试，本系统性能稳定、检测精度高，此外系统还具有使用便捷、价格低廉等特点。

当然，还可对本检测系统添加蜂鸣器或 LED 闪烁等报警功能模块，可对四项参数均事先设置安全阈值，一旦检测结果超出安全阈值，报警模块即发出蜂鸣声或红灯闪烁以提醒用户当前水质可能已不能供正常使用。另外，本系统对检测数据也未进行存储，每进行一次新的检测就会将上一次的检测数据覆盖掉，系统关机重启之后所有的检测数据也会丢失。若可为本系统建立一个服务器端数据库，将本系统作为客户端，并添加网络通信功能，一旦客户端检测到数据就立即上传到服务器端，而服务器端的海量数据可以应用于污水处理、水质监测等领域。

致谢

时光荏苒，大学四年如白驹过隙，一晃而过，犹记得刚踏进这所校园时我对未来的生活充满迷茫，不知该何去何从。幸运的是，回忆大学期间的点点滴滴，我惊喜地发现这四年里自己一直在成长。

我的父亲母亲和姐姐在这四年里仍同过去的二十多年里一样无条件地支持我做出的任何选择，没有他们，我恐怕不能够如此顺利地走过这么多年的求学生涯。我也要感谢吉吉一直陪伴我、鞭策我、督促我, 没有她的陪伴，我的考研之路会更加困难。

我的多位专业课老师们在这四年里用他们严谨治学的态度感染着我，我的室友们、我在实验室和球场上的朋友们，在这四年里也与我一同成长，谢谢你们。

最后，我还要感谢自己，谢谢你没有放弃当初那个满身臭毛病的夏庆生。

参考文献

[1] 申伟. 杜文娟. 基于ZigBee无线传感技术的水库水质多参数监测系统[J]. 自动化技术与应用, 2018, 37(4): 40-43.

[2] 周皓东. 黄燕, 刘炜. 基于WiFi无线传感器网络的水质监测系统设计[J]. 传感器与微系统, 2015, 34(5): 99-101+105.

[3] 赵敏华. 李莉, 呼娜. 基于无线传感器网络的水质监测系统设计[J]. 计算机工程, 2014, 40(2): 92-96.

[4] 梁正宁. 黄晟. 基于Lora的水质检测系统设计[J]. 无线通信技术, 2018, 4: 46-50+55.

[5] 杨磊, 熊卫华, 姜明. 基于NB_IoT技术的家庭水质检测系统[J]. 计算机系统应用, 2019, 28(12): 129-133

[6] 王英, 陈聪伟, 肖金球. 水质多参数监测系统设计[J]. 苏州科技学院学报(自然科学版), 2015, 32(1): 54-58+63

[7] 谷春英, 姚青山. 基于无线传感器的水质监测系统仿真设计[J]. 计算机仿真, 2013, 30(1): 340-343.

[8] 孙雷霸. 基于无线传感器网络的水环境多参数监测系统的研究与实现[D]. 江苏大学, 2009.

[9]周新. 从零开始学Altium Designer电路设计与PCB制版[M]. 北京: 化学工业出版社, 2020.

[10] 马忠梅, 王美刚, 王拓. 单片机的C语言应用程序设计（第6版）[M]. 北京: 北京航空航天大学出版社, 2017.

[11] Ahmed Abbas Fadel, Mohamed Ibrahim Shujaa. Water Quality Monitoring System Based on IOT Platform[J]. IOP Conference Series: Materials Science and Engineering, 2020, 928(3): 032054.

[12] Himanshu Jindal, Sharad Saxena, Singara Singh Kasana. Sewage Water Quality Monitoring Framework Using Multi-parametric Sensors[J]. Wireless Personal Communications, 2017, 97(1): 881-913.

附录

```
int main ( void )
{
float temperature;      //存放温度检测值
uint8_t uc, ucDs18b20Id [ 8 ];    //存放 DS18B20 序列号
uint8_t DS18B20Id_str[20];/* 配置 SysTick 为 1us 中断一次 */
SysTick_Init();              //系统定时器初始化
NT35510_Init ();             //LCD 初始化
// Debug_USART_Config();
//USART 初始化，当用串口调试助手显示温度时需要用到 USART，用液晶显示时
                                不需要用到 USART
Rheostat_Init();            // ADC 初始化
/* LED 端口初始化 */
LED_GPIO_Config();
//其中 0、3、5、6 模式适合从左至右显示文字，
//不推荐使用其它模式显示文字    其它模式显示文字会有镜像效果

//其中 6 模式为大部分液晶例程的默认显示方向
NT35510_GramScan ( 6 );
NT35510_Clear(0,0,LCD_X_LENGTH,LCD_Y_LENGTH);    /* 清屏，显示全黑
*/
NT35510_DispStringLine_EN(LINE(0),"1. ds18b20 test");
}
for(;;)
{
//1.循环检测温度并显示
temperature=DS18B20_Get_Temp();
sprintf((char*)dis_buf,"T:%0.3f degree Celsius",temperature);
```

```
NT35510_DispStringLine_EN(LINE(1),dis_buf);
//2.循环检测 pH 并显示
ADC_ConvertedValueLocal[0] =(float) ADC_ConvertedValue[0]/4096*3.3;
// 读取转换的 AD 值
PH_Value=-5.7541*ADC_ConvertedValueLocal[0]+16.654;
if(PH_Value<=0.0)
{
PH_Value=0.0;
}
if(PH_Value>=14.0)
{
PH_Value=14.0;
}
/*显示电压*/
Tx_PH[0]=(int)(PH_Value*100)/1000+'0';
Tx_PH[1]=(int)(PH_Value*100)%1000/100+'0';
Tx_PH[2]='.';
Tx_PH[3]=(int)(PH_Value*100)%100/10+'0';
Tx_PH[4]=(int)(PH_Value*100)%10+'0';
Tx_PH[5]='\0';    //字符串结束符
NT35510_DispStringLine_EN(LINE(3),"2. pH test ");
sprintf((char*)dis_buf,"pH:%s",Tx_PH);
NT35510_DispStringLine_EN(LINE(4),dis_buf);
//3.循环检测电导率并显示
EC_voltage =(float) ADC_ConvertedValue[1]/4096*3300; // 读取转换的 AD 值
rawEC = 1000*EC_voltage/RES2/ECREF;
EC_valueTemp=rawEC*kValue;
/*First Range:(0,2); Second Range:(2,20)*/
if(EC_valueTemp>2.0)
```

```
{
    kValue=kValue_High;
}
else if(EC_valueTemp<=2.0)
{
    kValue=kValue_Low;
}
EC_value=rawEC*kValue;
compensationCoefficient=1.0+0.0185*((temp_data/10)-25.0);
EC_value=EC_value/compensationCoefficient;
//if((EC_value<=0)){EC_value=0;}
//if((EC_value>20)){EC_value=20;}//20mS/cm
/*显示 EC*/
Tx_EC[0]=(int)(EC_value*100)/1000+'0';
Tx_EC[1]=(int)(EC_value*100)%1000/100+'0';
Tx_EC[2]='.';
Tx_EC[3]=(int)(EC_value)%100/10+'0';
Tx_EC[4]=(int)(EC_value)%10+'0';
Tx_EC[5]='\0';    //字符串结束符
NT35510_DispStringLine_EN(LINE(6),"3. Electrical conductivity(EC)");
NT35510_DispStringLine_EN(LINE(7),"test");
sprintf((char*)dis_buf,"EC:%s",Tx_EC);
 NT35510_DispStringLine_EN(LINE(8),dis_buf);
//4.循环检测浑浊度并显示
TU =(float) ADC_ConvertedValue[2]/4096*3.3; // 读取转换的 AD 值
TU_calibration=-0.0192*(temp_data/10-25)+TU;
TU_value=-865.68*TU_calibration + K_Value_TU;
if(TU_value<=0){TU_value=0;}
if(TU_value>=3000){TU_value=3000;}
```

```
/*显示 TDS*/
Tx_TU[0]=(int)(TU_value)/1000+'0';
Tx_TU[1]=(int)(TU_value)%1000/100+'0';
Tx_TU[2]=(int)(TU_value)%100/10+'0';
Tx_TU[3]=(int)(TU_value)%10+'0';
Tx_TU[4]='\0';    //字符串结束符
NT35510_DispStringLine_EN(LINE(10),"4. Turbidity(TU) test");
sprintf((char*)dis_buf,"TU:%s",Tx_TU);
NT35510_DispStringLine_EN(LINE(11),dis_buf);
//延时
CPU_TS_Tmr_Delay_MS(1000);
}
}
/*********************************END OF FILE**********************/
```