基于 STM32 的数据采集系统设计

摘要：随着社会时代的发展和科学技术的不断进步，如今在各个领域都能见到数据采集的运用。而且对于满足社会的要求传统的数据采集系统也逐渐不够满足，于是开始发展了一种新的数据采集方式：远程数据采集。这种方式在距离上面不会送到牵制，远距离也能够实时采集、查看数据。本设计根据网络资源以及其他相关项目资料，结合 GSM 无线通信技术，设计了一种能够将数据采集发送到远程并且实时查看数据的采集系统。这使得数据采集更加方便、快捷，突破了距离的限制。本设计包括数据采集，控制，传输以及远端服务平台。数据采集端模块负责采集和处理数据；控制端模块负责采集数据的存储、系统的控制以及界面显示；数据传输端模块主要负责通过 GSM 无线传输把相关数据上传到服务器。根据远程数据采集系统的功能要求，设计系统的硬件电路包括 STM32F103 控制端最小系统电路、数据采集电路、GSM 无线通信电路、实时时钟电路、液晶显示电路以及系统电源电路等，并基于硬件电路设计相应的软件程序。最后对本设计系统进行各模块测试及数据上传测试。

关键词：STM32； 数据采集； GSM； SIM808； 物联网； 发展； ONENET

Design of Data Acquisition System Based on STM32

Abstract: With the development of the social age and the continuous advancement of science and technology, the use of data collection can now be seen in various fields. In addition, the traditional data acquisition system is not enough to satisfy the social requirements. Therefore, a new data acquisition method has been developed: remote data acquisition. This way will not be sent to the upper limit in the distance, and it can also collect and view data in real time from a distance. This design, based on network resources and other related project data, combined with GSM wireless communication technology, designed an acquisition system that can send data collection to remote and view data in real time. This makes data collection more convenient and faster, breaking the distance limit. The design includes data acquisition end module, control end module (STM32F103), transmission end module and remote service platform. The data collection module is responsible for collecting and processing data. The control module is responsible for the storage of data, system control and interface display. The data transmission module is mainly responsible for uploading relevant data to the server through GSM wireless transmission. According to the functional requirements of the remote data acquisition system, the hardware circuit of the design system includes STM32F103 minimum control system control circuit, data acquisition circuit, GSM wireless communication circuit, real-time clock circuit, liquid crystal display circuit and system power supply circuit, etc., and based on the hardware circuit design Software program. Finally, each module test and data upload test of this design system.

Keywords:STM32, Data Acquisition, GSM, SIM808, Internet of Things, Developm ent, ONENET

目录

目录 III
第 1 章 绪论 1
1.1 数据采集系统的研究背景和意义 1
1.2 数据采集系统的发展情况[3] 1
1.2.1 系统器件模块化单片化 1
1.2.2 数据采集接口标准化 1
1.2.3 测量与控制一体化 1
1.2.4 器件智能化 2
1.3 GMS 无线通信 2
1.4 本论文的构成摘要 3
第 2 章 数据采集系统的方案设计 5
2.1 系统的组成 5
2.2 控制与显示设计与分析 5
2.2.1 STM32F103 芯片简介 6
2.2.2 显示端设计简介 7
2.2.3 GSM 无线通信 7
2.2.4 服务器远端简介 9
第 3 章 数据采集系统的硬件设计 12
3.1 系统电源电路 12
3.2 STM32MCU 最小系统及外围 13
3.2.1 内核 ARM Contex-M3 简介 13
3.2.2 STM32 最小系统电路 14
3.2.3 USB 串口接口 15
3.2.4 JTAG/SWD 程序下载接口 15
3.2.5 复位电路 15
3.2.6 LED 电路 16
3.3 数据采集电路设计 16
3.3.1 STM32 内部温度传感器 16

3.3.2 DS18B20 数字温度传感器采集电路 17
3.4 LCD 液晶显示电路 18
3.4.1 LCD 液晶显示屏 18
3.4.2 STM32 的 FSMC 功能 19
3.5 SIM808 通信模块 20
第 4 章 数据采集系统的软件设计 22
4.1 系统整体软件设计概述 22
4.2 系统初始化 23
4.3 数据采集端的软件设计 23
4.3.1 STM32 内部温度传感器的数据采集 23
4.3.2 DS18B20 温度传感器的数据采集 24
4.4 数据处理和上传的软件设计 25
4.4.1 AT 指令 25
4.4.2 设备连接至 ONENET 25
4.4.3 数据包封装 26
4.5 ONENET 平台自定义设计 27
4.6 小结 28
第 5 章 数据采集系统的调试及功能实现 29
5.1 电脑直连通信模块调试 29
5.2 数据封装及上传程序调试 30
5.3 设计成果展示 32
结论 34
致谢 35
参考文献 36
附录 1 系统原理图 37
附录 2 程序关键函数 38

第1章 绪论

1.1 数据采集系统的研究背景和意义

随着时代的进步和科学技术的日益更新，数据采集在汽车，家电，通讯，工业生产，医疗等各行各业都占据重要地位。压力，声音，流量，光线，重量等信号一般都需要经过转换之后计算机才能进行处理，而能够进行这样的信号转换的电路就叫做数据采集电路，能够实现数据采集的系统叫数据采集系统[1]。

在最近几年，数据采集系统以及其相关应用受到了人们越来越广泛的关注，数据采集系统也一直在飞速的发展，数据采集系统的发展趋势也朝着小型化、微型化、便携式、低电压、低功耗等方向发展。但市场上的数据采集卡，都是通过 USB 口，或者串口等与计算机连接进行操作，这样就限制的采集卡的使用距离，开发远程监控、采集、查询数据系统具有重大意义[2]。

1.2 数据采集系统的发展情况[3]

1.2.1 系统器件模块化单片化

在六七十年代，当时的数据采集系统主要是以半导体分器件来进行组建，直到八十年代左右，由于大规模以及超大规模集成电路的广泛应用，于是开始出现有单芯片器件和集成度高的数据采集器件。随着制造技术以及制造工艺的不断完善改进，器件的性能和功能有显著提高，市场上已经有 8 位、12 位、14 位的 A/D 转换芯片。由于芯片器件有小体积，低功耗，低造价，高可靠性，低热效应等诸多优点，而且还能与微控制器直接接口，将成为采集系统的主题器件。

1.2.2 数据采集接口标准化

测试设备以及数据采集设备在与计算机的连接是很重要的问题，若要用一台计算机控制多种不同的一起就必须要通用接口，如今已有很多的通用接口标准例如 232 串行接口，485 串行接口，CAN 总线接口等等。

1.2.3 测量与控制一体化

随着微处理器的普及和芯片技术的快速发展，除了处理器性能的飞速发展外，芯片制造商也开始将数据采集与转换集成在处理器内部，并且也是有很可观的发展，如今很多芯片内设 ADC，DAC，甚至有内置传感器，例如温度传感器。

1.2.4 器件智能化

数据采集与微处理机或者微型计算机相兼容是现代采集系统的发展方向，也就是以微处理器为基础和控制的性能化数据采集系统。由于微处理器可以进行软件控制，所以很容易进行定制从而满足特殊要求。近几年非常的火热也发展迅速的大数据与人工智能也开始融入数据采集系统，构成智能化的物联网系统。

1.3 GMS 无线通信

GSM 是欧洲电信标准组织 ETSI 制订的数字移动通信标准。简称全球移动通信系统(Global System of Mobile Communication)。该通信的空中接口用的是时分多址的技术，从 90 年代商用以来，已有超过 100 个国家在采用。GSM 标准的设备占据当前全球蜂窝移动通信设备的比例超过 80%。一般 2 我们所说的 GSM 是属于第 2 代蜂窝移动通信技术。所有用户可以在签署了"漫游协定"移动电话运营商之间自由漫游。GSM 较之它以前的标准不同点是信令和语音信道全部都是数字式的，因此 GSM 被看作是第二代(2G)移动电话系统。从用户观点出发，GSM 有一个主要的优势就是用户可以从更高的数字语音质量以及低费用的[SMS]之间作出选择。不同的网络运营商有各自的优势便是他们可以不同的客户定制他们的设备配置，由于 GSM 拥有开放标准而提供了更容易的互操作性。这样一来，就允许不同的网络运营商提供漫游服务，用户就可以在全球范围使用移动电话[5]。

随着信息化不断发展，GSM 在各行各业应用非常广泛，其发展趋势如下：

1、标准正在不断地升级。GSM 有一套很完整的发展计划，也已经确定了它的发展目标，即，GSM 在经过三个阶段(Phase I，Phase II，Phase III)的发展后，常规蜂房(GSM900)和移动通信微蜂房(PCN)最终能够结合起来，得以保证步行用户和正在高速运动的用户的移动通信需求以满足，并且，逐步实现话音通信、短信息服务、辅助业务，最后进入到 ISDN。

2、不断地寻求与 LMSS 结合方式。在 GSM 通信系统的标准中，已经对常规蜂房(GSM－900)和微蜂房(DCS1800)有所考虑，同时，一些国家还在研究 GSM 通信系统与陆地移动卫星系统(LMSS)的结合。它们充分结合形成一个系统，可以互相兼容，互相补充。

3、随着 IP 技术的兴起，电信网络以及它未来技术的发展走向，正被深刻地改变着的自己原有的面貌。GSM 移动通信技术的发展趋势主要是体现在个人化、分组化、宽带化和综合化，主要表现为：

(1) 通信信息技术发展的宽带化，由于高通透量的网络以及光纤传输技术的广泛应用，全球范围都在进行有线的网络宽带化，移动通信技术也在无线宽带化发展。

(2) 随着网络中数据业务量的地位越来越高，依靠传统电路交换的通信技术转向以分组，以及 IP 技术的网络方向发展。IP 协议终将全范围取代旧协议。

(3) 接入网络的多样化以及核心网络的综合化。未来的信息网络结构模式是向核心网、接入网转变的，网络的宽带化与分组化使得在同一个核心网络上传送多个综合业务信息成为可能。网络的综合化，管制的开放化和市场竞争的需要，也将推动传统的电信网络和新兴的计算机网络融合。

(4)信息的个人化。为了更好的推动个人通信行业的发展，必定是 IP 技术与移动智能网技术的有机结合。

(5) 所有的信息业务都需要根据市场的实际来需要提出，经过后台的软硬件集成后再“生产”出新的产品。

(6) 关于移动通信的网络结构的变革正在发展，未来的网络核心技术必定将是 IP 技术。将在业务控制分离的基础上进一步进行分离核心交换传送网和网络呼叫控制，最终网络层将由业务控制层、应用层和由核心网和接入网共同组成的。

1.4 本论文的构成摘要

本数据采集系统主要由数据采集模块、控制模块、显示模块、GSM 通信模块和服务器端组成，其中数据采集模块由控制器内置温度传感器，数字温度传感器模两个模块组成，内置的温度传感器集成在控制器芯片的内部并且与 ADC 通道直接相连接并且经控制器 AD 转换为数字值，而数字温度传感器输出为就是数字值，然后通过控制器处理两个通道的温度值，在显示模块进行显示，同时通过 GSM 模块将数据实现与远程服务器端的传输，用户可以远程登录服务器，查询数据的采集情况。本论文分为五个章节，各章节内容安排如下：

第 1 章为绪论，主要介绍本设计传统的研究背景和意义，发展情况以及本论文的构成摘要。

第 2 章对数据采集系统进行简单介绍，并且对每个模块做了相应的阐述，包括两个温度传感器模块，STM32 控制模块，GSM 通信模块，以及中国移动 ONENET 物联网开放平台的介绍。

第 3 章详细介绍本设计系统的硬件设计以及相关原理，其中包括电源电路，STM32 最小系统以及其外围电路，DS18B20 数据采集电路，LCD 液晶显示电路以

及 SIM808 通信模块电路。

第 4 章详细介绍本设计系统的软件和运行流程，包括系统初始化，数据采集的程序设计，数据封装的程序处理，数据发送的程序设计，连接 ONENET 服务器，以及 ONENET 平台端的产品设备设置数据流添加和应用的建立。

第 5 章对本系统的设计调试过程以及实现情况进行详细的介绍。

第2章 数据采集系统的方案设计

2.1 系统的组成

本系统主要分为数据采集端，控制及显示端，数据传输端和服务器远端。系统整体结构如图 2-1 所示。

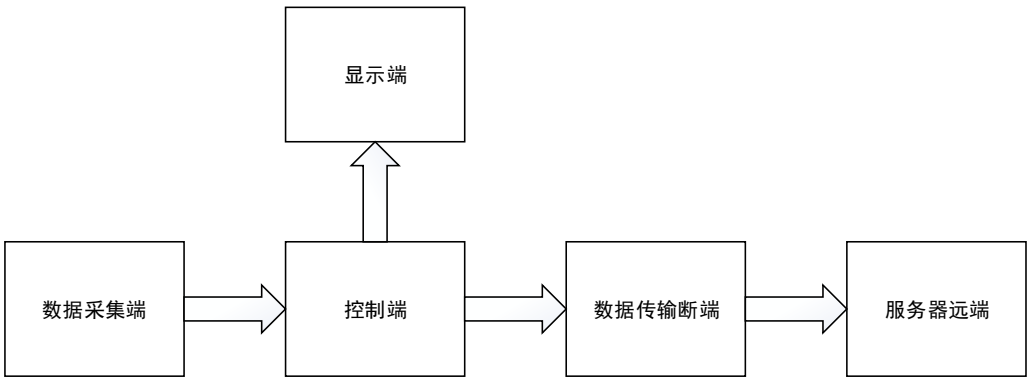


图 2-1 系统整体结构图

数据采集端对数字模拟信号进行采集，并将数字或者模拟信号传输到控制器中进行处理。本系统使用了控制器内置温度传感器和 DS18B20 数字温度传感器这两组传感器进行数据采集。控制器内置温度传感器直接连接控制器 ADC 端口，经过 AD 转换为数字信号。而 DS18B20 数字温度传感器输出直接就是数字信号。

控制显示端是整个系统的核心，系统中信号的处理，对数据发送的准备，和显示等功能都是由控制端进行控制。本设计系统主控芯片是采用意法半导体基于 ARMCortex-M3 核心的 STM32F103 芯片。

数据传输端实现控制端信号向服务器远端的数据传输，本系统采用 GSM 无线通信模块，采用 GPRS 网络上传数据。

服务器远端是接收数据的设备，它就是远端的服务器，可远程查询数据，本系统采用了中国移动 ONENET 物联网开放平台进行数据上传。

2.2 控制与显示设计与分析

控制器是整个数据采集系统的核心，它既要完成对数据的处理和传输，又要对各种外围设备进行控制，根据设计功能需求，控制器芯片选型需要考虑到性价比，研发周期，功耗，控制器性能等因素[2]。市场主流的控制器很多，运用比较多的有 51 系列芯片，TI 公司的 MSP430 系列，ARM7、ARM9 系列控制器以及基于 Cortex-

M3 架构的 STM32 系列等等。基于系统要求功耗较低，主要硬件功能应有时钟模块、异步串行通信模块、定时器模块、ADC 模块等。

综合各种控制芯片的优缺点，基于成本控制，系统使用裸机程序设计即可，不需要运行操作系统，所以 ARM7、ARM9 系列不能使用，而 51 系列单片机虽然应用非常广泛，但是自身资源有限，而且功耗较高，也不纳入设计，MSP430 系列和 STM32 均较符合设计要求，但考虑普及性 STM32 更加广泛，所以设计选用意法半导体 STM32 系列芯片的 STM32F103ZET6。

2.2.1 STM32F103 芯片简介

意法半导体推出的 STM32 系列处理器，在内核基础上将极高性价比的外围设备进行了扩展[18]。

STM32 处理器最显著的特性包括处理速度提高 35%，代码量减少了 45%，具有支持延时操作和实时性能的快速中断控制器。同时还具有节能降耗的有点，只要六个 CPU 时钟周期即可从待机状态唤醒，另外还支持高精度的电源管理。驱动电压也比较广，可在 2V 到 3.6V 之间直接驱动。

在晶振频率为 72MHZ 的正常工作模式下，电流消耗也只有 27mA。除了正常工作模式下有较高的能耗外，还具有其他三种低功耗工作模式和多种用途的时钟设置方案。实时时钟部分还设置了一个内嵌转换电路实现纽扣电池与电源之间的转换。

本系统选用 STM32 具体型号为 STM32F103ZET6,采用 LQFP144 封装，芯片引脚图如图 2-3 所示。

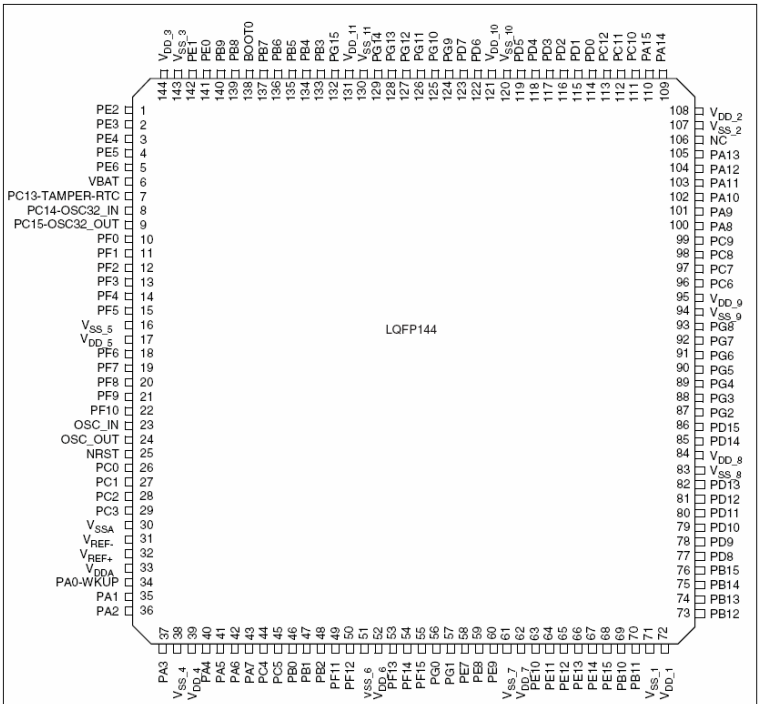


图 2-2 STM32F103ZET6 芯片引脚图

STM32F103ZET6 控制器具有非常高的性能以及丰富的外设资源。内核方面该芯片具有 64KBSRAM、52KBFLASH、2 个基本定时器、4 个通用定时器、2 个高级定时器。通信接口方面具有 3 个 SPI、2 个 IIC、5 个串口、1 个 USB、1 个 CAN、3 个 12 位 ADC、1 个 SDIO 接口、1 个 FSMC 接口以及 112 个通用 IO 口[18]。

2.2.2 显示端设计简介

显示端将采集的数据进行直观显示，常用的显示器件有 1602、128624 液晶显示屏，LCD 液晶显示屏，结合系统情况以及 STM32 的控制特性，本系统采用 LCD 液晶显示屏。

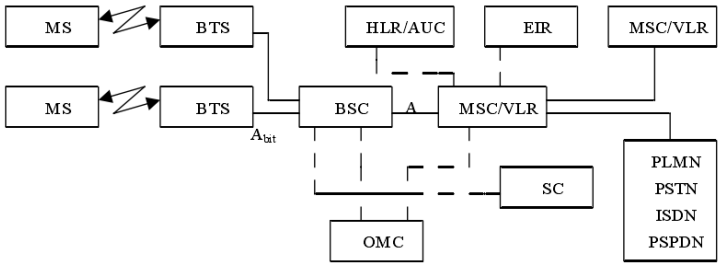
2.2.3 GSM 无线通信

GSM 英文全称 Global System for Mobile Communications，也就是全球移动通信系统。欧洲各国为了建立统一的数字蜂窝通信系统在 1982 年成立了移动通信特别小组。提出目标开发数字蜂窝通信系统。第一个实用的系统于 1991 年在欧洲开通，并且把名字改成全球移动通信系统[5]。在那之后，GSM 系统又被不断的改良和升级，很快在全球范围得到广泛的应用[6]。

2.2.3.1 GSM 系统结构

GSM 的每个子系统又由若干个功能实体构成，功能实体即是通信系统中的每一

个具体设备，它们各自具有一定的功能，完成相应的工作，如 VLR，HLR 等。各个子系统中间通过接口连接，各个子系统之间的路由可以分为通信路由和信路由。GSM 系统结构框图如图 2-4 所示。



MS:移动台　　BSS：基站子系统　　NSS：网络交换子系统

SC：短信息业务中心　　OMC：操作维护中心　　MSC：移动业务交换中心

VLR：来访位置寄存器　　HLR：归属位置寄存器 AUC：鉴权中心

EIR：移动设备识别寄存器　　BSC：基站控制器　　BTS:基站收发台

PSPND：分组交换公用数据网　　PSTN：公用电话网　　ISDN：综合业务数字网

图 2-3 GSM 系统结构框图

GSM 全球移动通信系统由网络交换子系统、无线基站子系统和移动台着这三个系统组成。其中无线基站子系统介于网络交换子系统和移动台之间，提供和管理它们之间的信息传输。所以无线基站字系统主要是对另外两个系统功能实体之间的无线接口的管理。

2.2.3.2　GSM 系统业务

GSM 系统主要基本电信业务、承载业务、和补充业务这三个业务组成。其中电信业务提供移动台与其他应用通信。承载业务传输用户界面之间的信息。补充业务集中体现全部使用方便和完善的服务。

电信业务包括语言业务和非话业务。语言业务是 GSM 系统提供的基本业务，也就是我们常说的打电话功能。非话业务又叫做数据业务，提供固定用户和 ISDN 用户所能享用业务中的大部分业务，包括文字、图像、传真、计算机文件、Internet 访问等服务。其中电信业务有：电话业务、紧急呼叫业务、语言信箱业务、短信业务、可视图文接入业务、智能用户电报传送业务、传真业务、GPRS 业务。

在数据传输系统，用的较多的是短信 SMS 业务和 GPRS 业务，但随着 GPRS 业务的不断发展，SMS 的弊端越来越多的展现出来：信息长度受限、信息格式单一、消息结构不灵活、采用的信令通道速率低、存储转发机制效率低、产生费用高。GPRS 业务是第二代移动通信数据传输技术。是以封包（Pocket）方式传输，传输费用以传

输资料单位计算，并使用的是整个频道。所以产生费用远比 SMS 便宜，同时传输速率高很多。

综合考虑传输速率以及产生的费用，本系统采用 GPRS 业务进行数据传输。

2.2.3.3 SIM808GSM 通信模块简介

本系统 GSM 通信模块采用的是 SIM808 通信模块。SIM808 模块是一个完整的四频 GSM/GPRS 模块，结合了卫星导航的 GPS 技术[20]。将 GPRS 和 GPS 集成在 SMT 封装中。对需要同时开发 GPRS 以及 GPS 应用的用户很大程度降低了开发成本和难度。采用行业标准接口和 GPS 功能，可在任何位置和任何时间以信号覆盖的方式无缝跟踪。SIM808 模块的实物如图 2-5 所示。

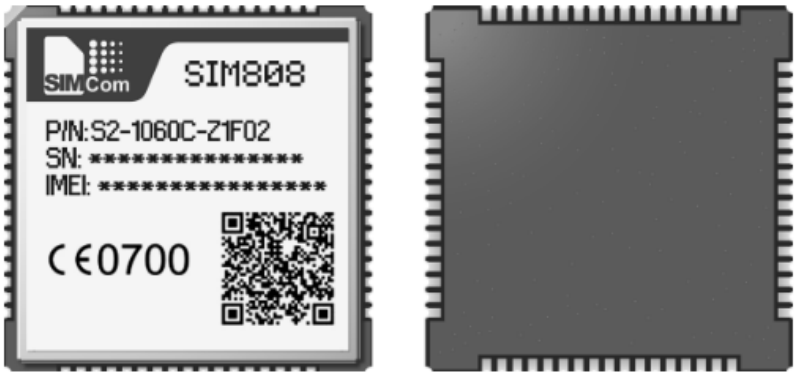


图 2-4 SIM808 模块实物图

2.2.4 服务器远端简介

采集数据最后还要上传到服务器使之可以远程查询。一般情况用户可以使用计算机进行服务器搭建，但是出于安全性和稳定性考虑，用户可以选择专业的服务器公司购买服务器的使用权。而在近几年，一个新的名词“物联网”快速普及，相关企业也推出开放平台使得远程服务器数据传输门槛更低。

2.2.4.1 物联网

物联网是美国在 2000 年提出的概念。当时叫传感网。通过射频识别（RFID）、红外感应器、全球定位系统、激光扫描器等设备。按找约定的协议可以把任何物品通过物联网域名进行连接，进行信息交换，进行通信。进而将智能化识别、定位、跟踪、监控和管理等功能进行实现。“物联网”概念是在“互联网概念”的基础上，将在任何物品与物品之间进行用户端延伸和扩展。并且进行信息交换和通信的一种网络概念。

随着大数据和智能化的快速发展，物联网已有非常广泛的应用领域，包括智能家居、智慧交通、智能医疗、智能电网、智能物流以及在农业、安防、汽车、建筑等诸多领域的智能化应用。

2.2.4.2 物联网开放平台

物联网开放平台也叫做物联网服务平台。用户可以通过企业研发和开放连接接口（API）以及协议，进行注册并自定义上传数据和处理数据。有付费平台也有免费平台，现在很多互联网公司都有提供服务平台如腾讯的 QQ 物联、阿里巴巴的阿里云物联网套件、中国移动的 ONENET 平台、京东的京东微联、百度的 IoT Hub 等等。本系统设计选用中国移动的 ONENET 物联网开放平台进行数据上传。

ONENET 是中国移动旗下的物联网开放平台。开发者能轻松的进行对设备的接入与设备的连接，用较短时间完成产品开发。目前为止 ONENET 是完全免费的并且不限制接入设备数量。ONENET 具有高并发可用、多协议接入、丰富 API 支持、快速应用孵化、数据安全存储、全方位支撑等优势。

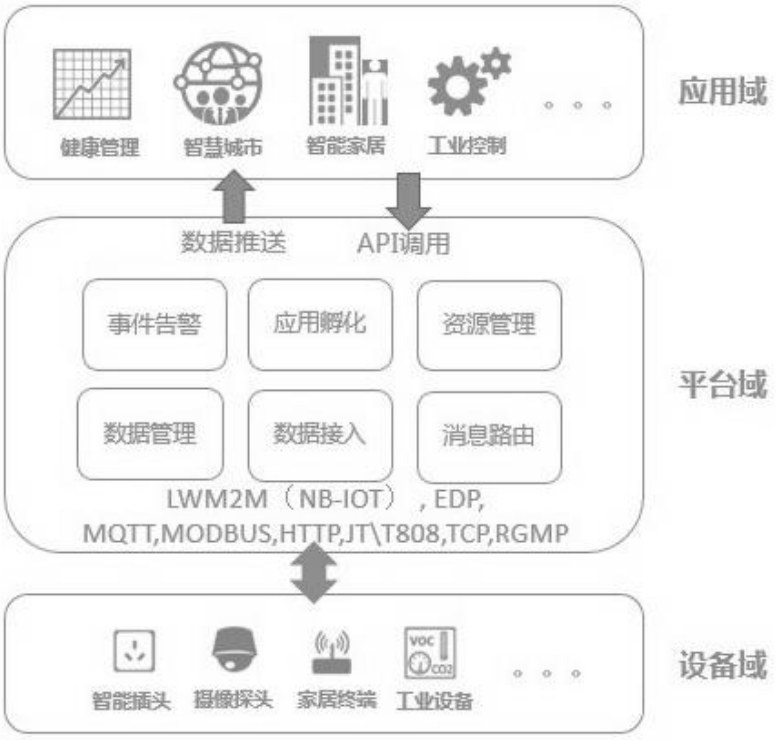


图 2-5 ONENET 平台架构

通过查阅官方给出的开发文档，可以很快上手开发，ONENET 平台支持 NB-IOT、EDP、MQTT、HTTP、TCP、MODBUS 等多种协议接入，支持模块应用的 DIY，并

且提供部分应用元素模板。ONENET 的平台架构如 2-6 所示。

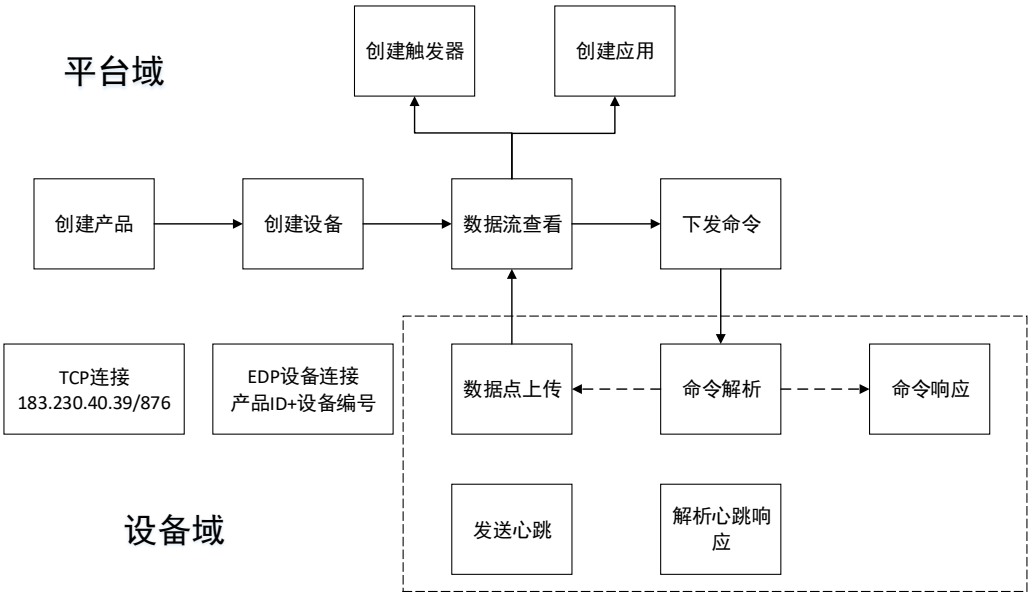


图 2-6 EDP 接入流程图

考虑到本系统传输数据的内容结构数据量以及连接平台的方式（GSM），最终采用 EDP 协议中的 RestFulAPI 上传数据点，这种方式设备不会保持在线，数据点上传成功后服务器端自动断开连接，下一次上传又重新连接。

EDP 的接入流程如图 2-7 所示，ONENET 平台还提供多种协议多种编程语言的接入 SDK，平台注册后，可按照开发文档创建产品和设备，将平台提供的 EDP 协议 SDK 移植调试，便可进行数据上传。

第3章 数据采集系统的硬件设计

本数据采集系统硬件设计主要由 STM32MCU 控制电路，电源电路，DS18B20 数字温度采集电路，LCD 液晶显示电路，SIM808GSM 通信电路构成。系统整体硬件结构如图 3-1 所示。

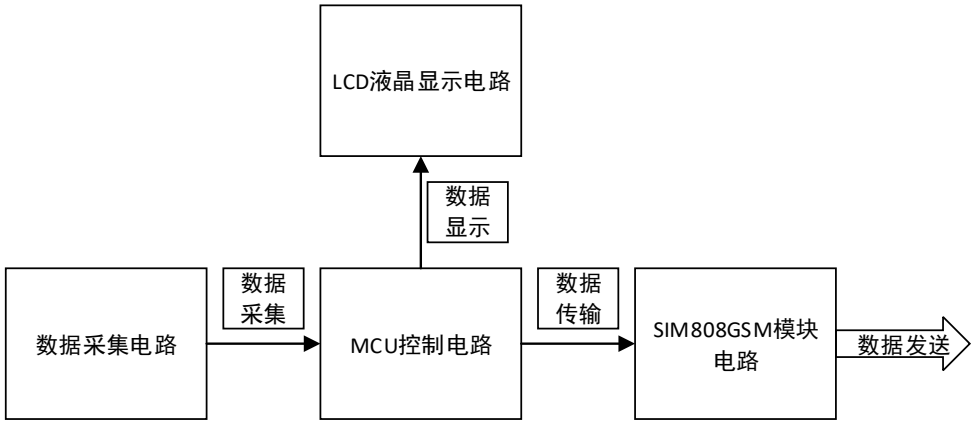


图 3-1 系统整体硬件结构图

3.1 系统电源电路

系统电源电路给系统各个模块供电，包括 STM32 控制模块和 SIM808 通信模块，以保证系统正常运行。总共有三块稳压芯片：U16、U17 和 U19，DC_IN 用于外部直流电源输入，输入电压范围为 6-12V，经过 U17DC-DC 芯片转换为 5V 电源输出，其中 D4 为防反接二极管，避免外部直流电源极性的接反而烧坏系统，F1 为 500mA 自恢复保险丝，用于保护电源，U16 为 3.3V 稳压芯片，给 STM32 提供 3.3V 电源。系统电源的电路图如图 3-2 所示。

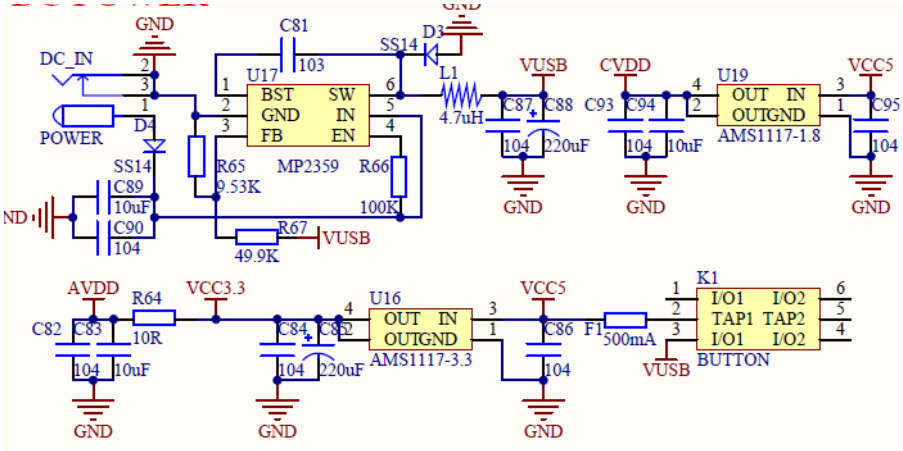


图 3-2 整体电源电路图

3.2 STM32MCU 最小系统及外围

控制电路主要是对采集到的数据进行处理以及对各模块的功能控制，程序下载以及串口通信。

3.2.1 内核 ARM Contex-M3 简介

Cortex-M3 内核是具有 32 位的处理器。内部的数据路径是 32 位的，寄存器是是 32 位的，存储器接口也是 32 位的。哈佛结构是 Cortex-M3 的一大特点，独立拥有指令和数据两条总线，所以在访问指令时也可以同时访问数据。很大程度提高了性能。

Cortex-M3 有好几条总线接口，每条都为自己的应用场合优化过，并且可以同时工作。但是指令总线和数据总线共享同一个存储器空间（一个统一的存储器系统）。对于有些需要更多的存储系统功能的复杂应用，有一个可选的 MPU 供用户选择，在有必要的环境下也可以增加外部存储。

Cortex-M3 的处理器为了缩小芯片体积，内部紧密结合了系统各组件。使用了代码很密集的 Thumb-2 指令集架构，很大程度的降低了对存储器的使用量。

Cortex-M3 系列处理器采用了满递减堆栈方式，也就是堆栈的指针总是指向堆栈中的最后一个数据单元，当有新的数据入栈，指针逐次递减，同时在新的内存位置上写入新的数据。

另外，堆栈空间分为主堆栈、进程堆栈，在工作模式下，有控制寄存器控制堆栈的使用，默认使用的是主堆栈。

Cortex-M3 处理器有 R0-R15 一共 16 个寄存器组，之中的 R0 到 R12 为 32 位通用寄存器，可以对数据进行操作。一般 16 位的 Thumb 指令只能访问 R0-R7， 32 位的 Thumb-2 指令才可以对所有的指令进行访问。R13 是堆栈指针寄存器，用来存放堆栈指针。R14 是连接寄存器，用来调用主程序的信息。R15 是程序寄存器，它指向的是当前程序的地址。

另外，还有几个特殊功能寄存器，包括中断屏蔽寄存器组、控制寄存器、程序状态字寄存器组。Cortex-M3 的内部结构如图 2-2 所示。

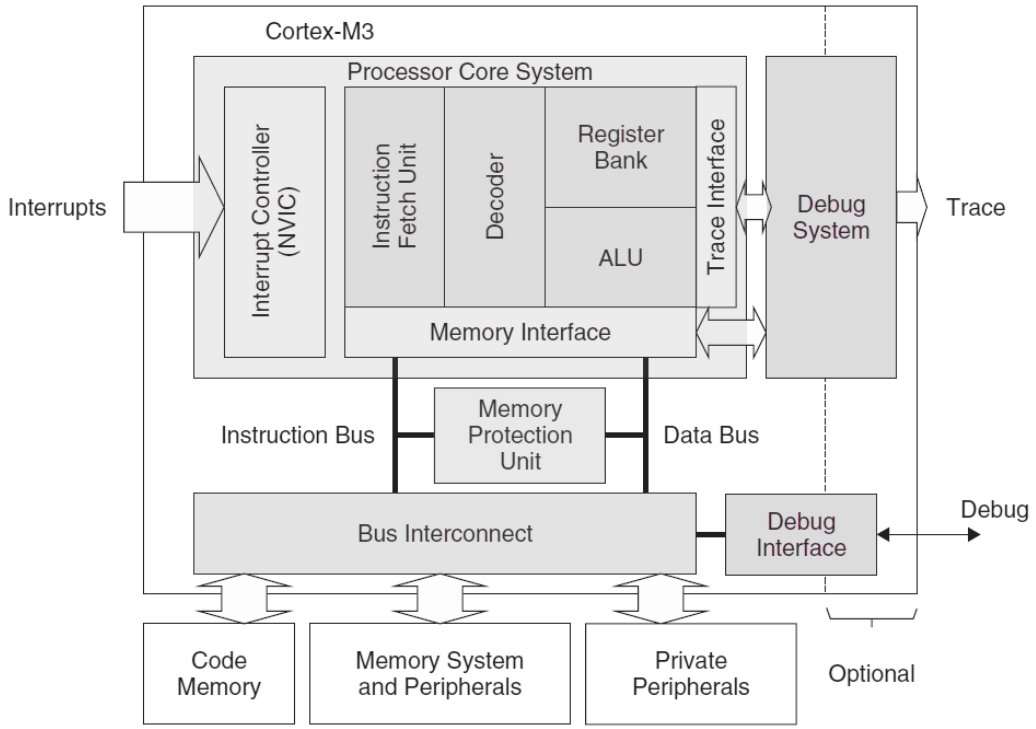


图 3-3 Cortex-M3 内部结构图

从数据采集系统的软件设计和硬件设计等方面考虑，Cortex-M3 处理器有很多优点：更小的内核和系统降低整个系统的成本；电源管理完整，功耗更低；处理器的性能以及可以满足应用需求；普及性广，较完善的系统调试功能使系统的开发过程加快。

3.2.2 STM32 最小系统电路

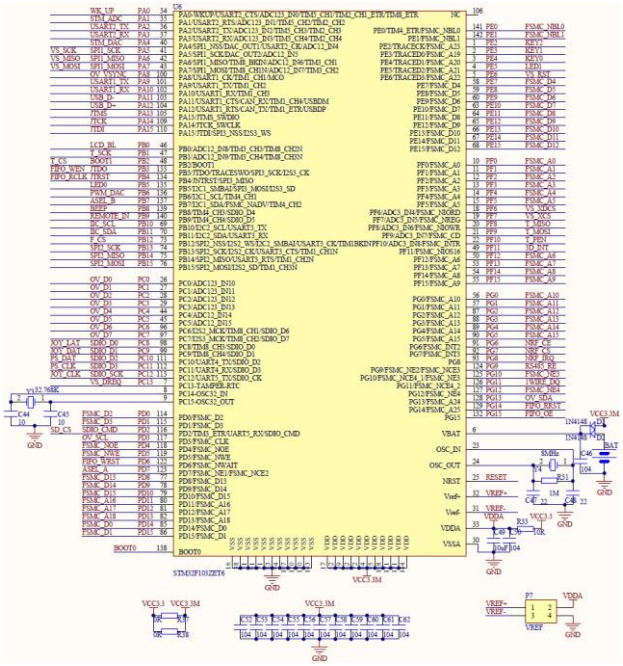


图 3-4 STM32 最小系统原理图

STM32 最小系统电路如图 3-3 所示。芯片的额定电压是 2.0-3.6V，故本设计系统以 3.3V 为芯片供电。芯片内置 8MHZ 内部高速 RC 制动器 HIS 和 40KHz 内部 RC 制动器 LSI，设计过程中用户只用在外部添加晶振源即可，通过相应的寄存器配置可以完成各个功能所需要的时钟源。

本系统设计 HSE 端工作主时钟使用 8MHZ 无源晶振。LSE 端采用的是 32.768Hz 晶振作为 RTC 时钟源。

3.2.3 USB 串口接口

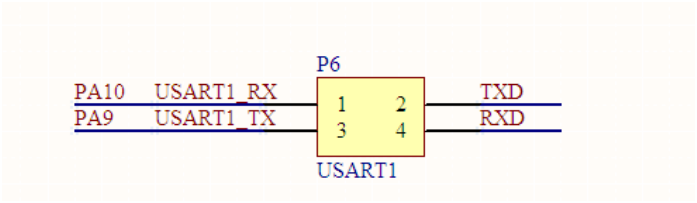


图 3-5 USB 串口接口图

USB 串口用来连接电脑以方便进行数据上传时的程序调试，本设计将 USB 串口 TXD、RXD 分别连接 STM32 的 PA10、PA9 口，通过接线柱插入跳线帽将端口连接。电路图如图 3-4 所示。

3.2.4 JTAG/SWD 程序下载接口

本设计把标准 JTAG 下载接口有接出，还可以兼容 SWD 接口进行程序下载。SWD 就只需要连接 SWCLK 和 SWDIO 以及 GND 便下载和调试程序。电路图如图 3-5 所示。

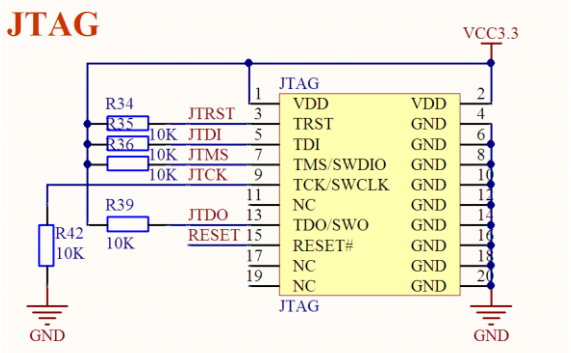


图 3-6 STM32 程序下载接口原理图

3.2.5 复位电路

STM32 的复位电路如图 3-6 所示。STM32 是低电平复位的。所以设计电路也是

低电平复位，R32 和 C51 构成上电复位电路，同时，LCD 液晶屏的复位引脚也连接在 RESET 上，这样液晶屏与 STM32 可以同时复位。

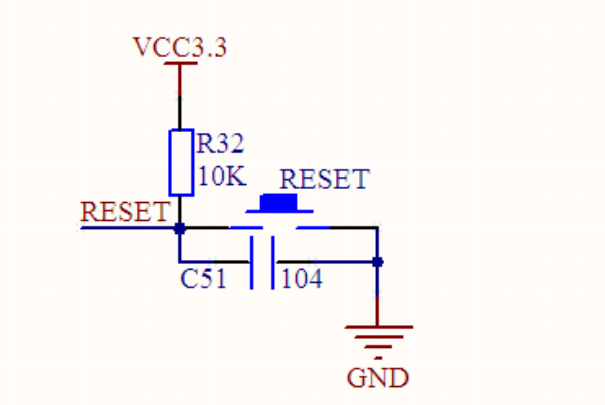


图 3-7 STM32 复位电路原理图

3.2.6 LED 电路

系统使用了三个 LED 灯，用于指示，电路如图 3-7 所示。

PER 是系统为蓝色电源指示灯，LED0 和 LED1 分别接在 STM32 的 PB5 和 PE5 上。

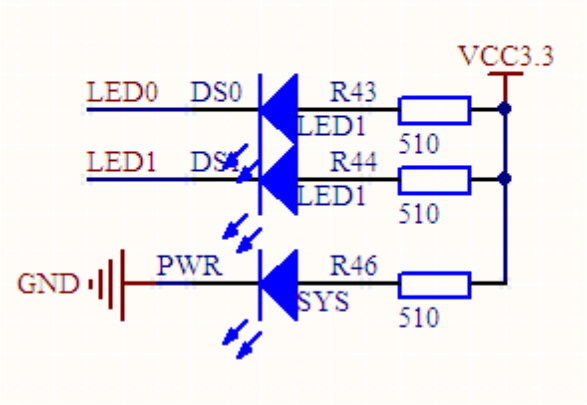


图 3-8 LED 电路原理图

3.3 数据采集电路设计

3.3.1 STM32 内部温度传感器

STM32 自带一个内部温度传感器，可以对 CPU 和周围的温度进行测量。该温度传感器在内部连接 ADCx_IN16 输入通道，此通道把传感器输出的电压转换成数字值，该温度传感器支持的温度范围为-40~125 度。精度比较差，误差在正负 1.5 摄氏度左右。由于该传感器集成在 STM32 芯片内部，所以没有相关电路。

3.3.2 DS18B20 数字温度传感器采集电路

DS18B20 是非常常用的一款数字温度传感器，美国 Dallas 半导体公司的数字化温度传感器 DS1820 是世界上第一片支持"一线总线"接口的温度传感器，在其内部使用了在板（ON-B0ARD）专利技术。全部传感元件及转换电路集成在形如一只三极管的集成电路内。

DS18B20 具有以下特性：

1、电压范围适应程度宽，电压范围：3.0～5.5V。供电条件简单。

2、接线方式为单线接口方式。DS18B20 仅需要一条串口线即可双向在微处理器与 DS18B20 之间进行通讯。

3、支持多点组网功能。可以通过并联将多个 DS18B20 进行连接，实现多通道温度测量。

4、DS18B20 不需要再搭建电路可以直接使用，所有传感元器件和转换电路都集成在形如一只三极管的集成电路内。

5、温度测量范围为－55℃～＋125℃，其中在-10～+85℃时精度为±0.5℃。

DS18B20 的内部结构如 3-8 所示：

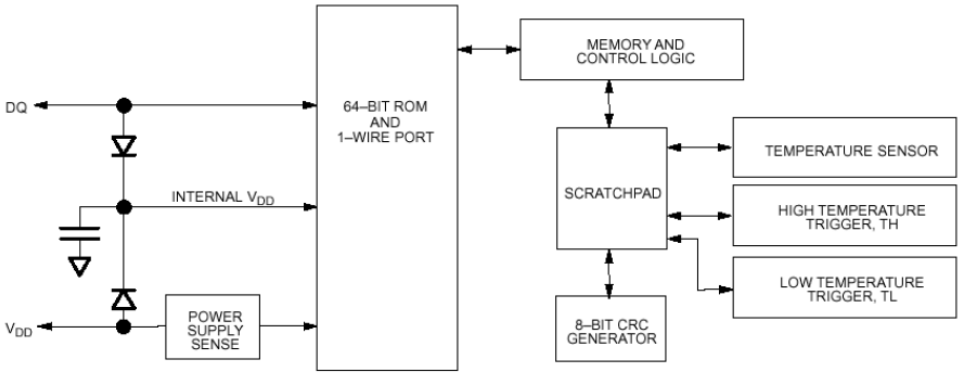


图 3-9 DS18B20 内部结构图

ROM 中有 64 位出厂前标记好的序列号，也就是是传感器的地址序列码，ROM 可以使每一个传感器都不相同，从而可以在一根总线上挂多个传感器。

所有的单总线为了保证其数据的完整性都需要采用严格的信号时序。DS18B20 共有 6 种信号类型：复位脉冲、应答脉冲、写 0、写 1、读 0 和读 1，除了应答脉冲外，其他信号都由主机发送同步信号，并且命令和数据都是低位在前。

本系统将 DS18B20 传感器连接在 STM 的 PG11 引脚，通过对该引脚的操作便

可以采集温度数据，具体电路如图 3-9 所示。

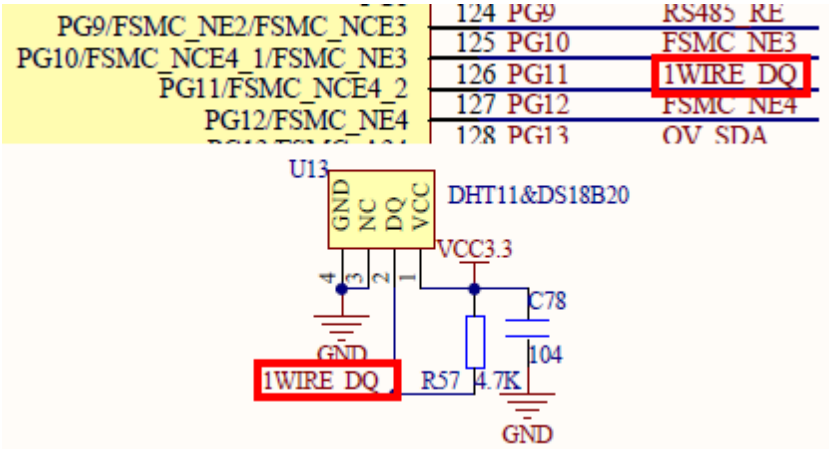


图 3-10 DS18B20 温度采集电路原理图

3.4 LCD 液晶显示电路

STM32 通过 FSMC 接口与 LCD 液晶屏连接，用来显示采集的数据以及当前时间信息。

3.4.1 LCD 液晶显示屏

液晶显示屏的每一个像素上都有一个薄膜晶体管，可有效克服非选通时的串扰，是显示的静态特性与扫描线数无关，图像质量大大提高。

本系统使用 2.8 寸 LCD 液晶显示屏进行显示。这块液晶屏支持 65K 色显示，分辨率为 320*240，接口是 16 位的 80 并口。

模块采用 2*17 双排座进行连接，所以该系统在 STM32 引出相应 FSMC 接口连接双排针，因为彩色屏的数据量比较大，所以采用 16 位的并行方式与液晶屏相连接，具体接线如图 3-10 所示。

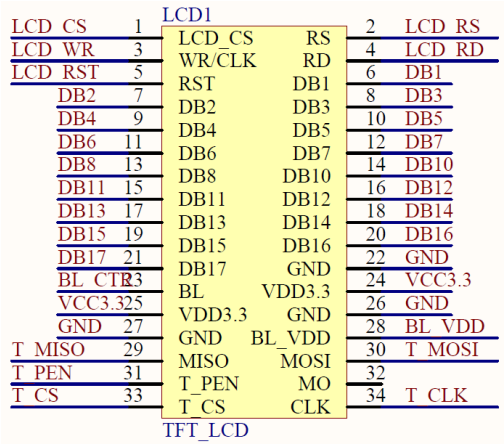


图 3-11 STM32 液晶屏接口原理图

该液晶屏采用 ILI9320 液晶控制器进行控制，此控制器自带显存，总大小为172820，也就是 18 位模式下的显存量。模块的 16 位数据线与显存的对应关系为 565 方式，如图 3-11 所示;

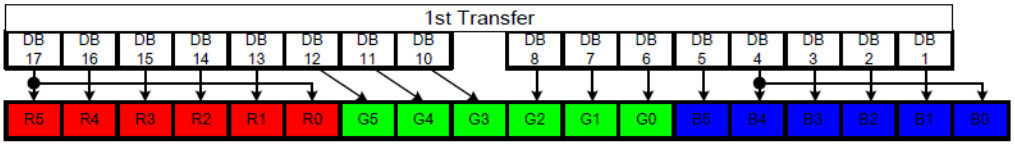


图 3-12 16 位显存寄存器

最低的 5 位代表蓝色，中间的 6 位代表绿色，最高的 5 位代表红色，数值越大，代表相应颜色越深。要控制液晶屏显示，还需要控制 ILI9320 的几个重要命令，部分液晶控制器命令如表 3-1 所示

根据液晶屏的主要控制命令，可以由以下步骤得到液晶屏的显示：

1、设置 STM32 与 LCD 液晶屏模块相连接的 IO。

2、初始化 LCD 模块。

3、LCD 液晶屏显示。

表 3-1 液晶屏控制器 ILI9320 几个重要命令

编号	指令	各位描述																命令
	HEX	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
R0	0X00	1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	OSC	打开震荡器/
		1	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	读取控制器信号
R3	0X03	TRT	DFM	0	BGR	0	0	HWM	0	0	0	I/D1	I/D0	AM	0	0	0	人口模式
R7	0X07	0	0	PTDE1	PTDE0	0	0	0	BASEE	0	0	GON	DTE	CL	0	D1	D0	显示控制
R32	0X20	0	0	0	0	0	0	0	0	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	行地址（X）设置
R33	0X21	0	0	0	0	0	0	AD16	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8		列地址（Y)设置
R34	0X22	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	NC	写数据到GRAM
R80	0X50	0	0	0	0	0	0	0	0	HSA7	HSA6	HSA5	HSA4	HSA3	HSA2	HSA1	HSA0	行起始地址（X)设置
R81	0X51	0	0	0	0	0	0	0	0	HEA7	HEA6	HEA5	HEA4	HEA3	HEA2	HEA1	HEA0	行结束地址（X)设置
R82	0X52	0	0	0	0	0	0	0	VSA8	VSA7	VSA6	VSA5	VSA4	VSA3	VSA2	VSA1	VSA0	列起始地址（Y)数组
R83	0X53	0	0	0	0	0	0	0	VEA8	VEA7	VEA6	VEA5	VEA4	VEA3	VEA2	VEA1	VEA0	列结束地址（Y)设置

3.4.2 STM32 的 FSMC 功能

FSMC 英文全称为 Flexible Static Memory Controller，即可变静态存储控制器。是 STM32 系列采用的一种新型的存储器扩展技术。

大容量，并且引脚数在 100 脚以上的 STM32F103 芯片都带有 FSMC 接口。FSMC 能够与同步或异步存储器和 16 位 PC 存储卡连接，STM32 的 FSMC 接口包括 SRAM、NAND、FLASH、NORFLASH、和 PSRAM 等存储器。

FSMC 有以下技术优势：

1、支持多种静态存储器类型。通过 FSMC，STM32 可以与 SRAM、ROM、PSRAM、NORFlash 和 NANDFlash 存储器的引脚直接相连。

2、有多种方法进行存储操作。FSMC 不仅支持多种数据宽度的异步读/写操作。而且支持对存储器的同步突发访问。

3、可以同时扩展多种存储器。FSMC 的映射地址空间中，不同的 BANK 是独立的并且可用于扩展不同类型的存储器。当系统中扩展和使用多个外部存储器时，FSMC 会通过总线悬空延迟时间参数的设置，防止各存储器对总线的访问冲突。

4、支持更为广泛的存储器型号。通过对 FSMC 的时间参数设置，扩大了系统中可用存储器的速度范围，为用户提供了灵活的存储芯片选择空间。

5、支持代码从 FSMC 扩展的外部存储器中直接运行，而不需要首先调入内部 SRAM。

LCD 模块通过 RS 信号来决定传输的数据是数据还是命令，本质上可以理解为一个地址信号，比如我们把 RS 接在 A0 上面，那么当 FSMC 控制器写地址 0 的时候，会使得 A0 变为 0，对 LCD 来说，就是写命令。而 FSMC 写地址 1 的时候，A0 将会变为 1，对 LCD 来说，就是写数据了。这样，就把数据和命令区分开了，其实就是对应 SRAM 操作的两个连续地址。当然 RS 也可以接在其他地址线上，在本系统中是把 RS 连接在 STM32 的 A10 上面的。另外 STM32 的 FSMC 支持 8 位、16 位、32 位，在控制 LCD 屏时位宽需选择 16 位。

3.5 SIM808 通信模块

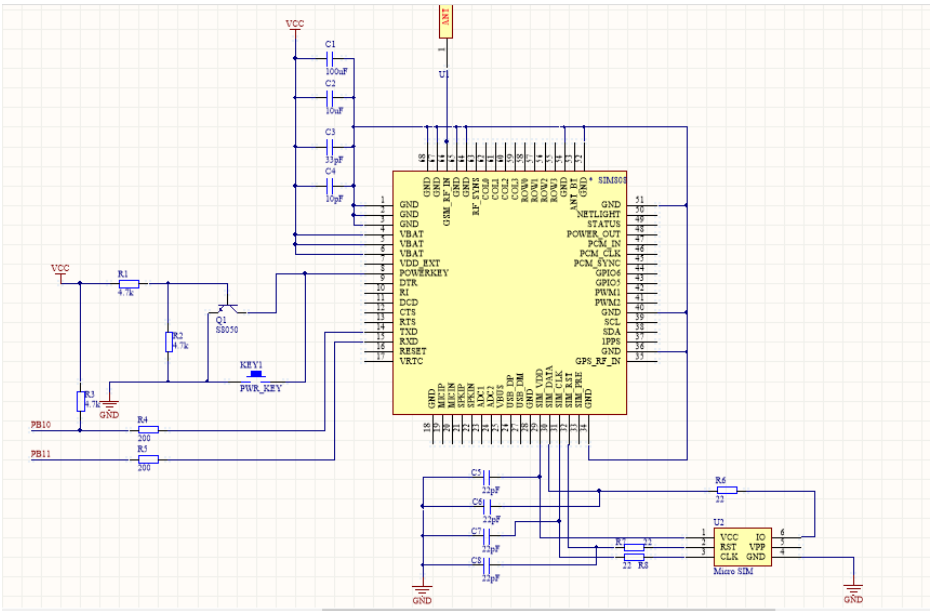


图 3-13 SIM808 模块电路原理图

SIM808 模块是一个完整的四频 GSM/GPRS 以及卫星 GPS 导航技术相结合的

通信模块。SIM808 支持 GPRSmulti-slotclass10/class8(可选)和 GPRS 编码格式 CS-1，CS-2，CS-3andCS-4。SIM808A 采用了省电技术设计，在 SLEEP 模式下最低耗流低至 1mA。此外，该模块内嵌 TCP/IP 协议，扩展的 TCP/IP 命令让用户能够很容易使用 TCP/IP 协议，这些在用户做数据传输方面的应用时都非常有用。SIM808 模块电路如图 3-12 所示：

本模块通过串口连接至 STM32 串口通道 3 的 PB10 和 PB11。由于本系统只用到 SIM808 模块的 GPRS 功能，所以 GPS 和模块语音接口没有连接。另外 U2 是 SIM 卡槽，需要准备一张能够使用的中国移动电话卡。

第4章 数据采集系统的软件设计

本系统软件的集成开发环境使用 Keil MDK 进行程序设计。Keil MDK-ARM 软件为基于 cortex-M、cortex-R4、ARM7\9 处理器设备提供一个完整的开发环境，其专为微控制器应用而设计，具有完善的 C/C++开发环境，不仅易学易用，且功能非常强大。系统设计采用的微控制器 STM32F103 自带丰富的库函数，方便移植和阅读，用户可直接在 MDK 开发环境中进行编译和调试，大大缩短开发周期和开发难度。本系统的软件使用 C 语言编写，采用结构化的程序设计方法，各个模块的程序相对独立，方便后期对代码的维护、移植和升级，同时也使代码的调试难度大大降低。

4.1 系统整体软件设计概述

在该数据采集系统中，选用微控制器 STM32F103ZET6 为主控单元。系统的整个软件部分设计，包括硬件部分功能实现编程和设计，都是在该处理器平台上完成的。系统上电后，在运行之前需进行各模块的初始化，然后进行数据采集处理，数据显示和数据发送。系统流程框图如图 4-1 所示

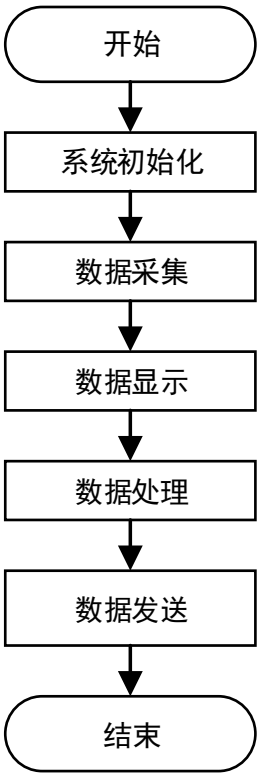


图 4-1 系统流程图

4.2 系统初始化

系统初始化是数据采集系统运行的第一步，其中包括 STM32 时钟初始化，串口初始化，LED 初始化，ADC 内部温度传感器初始化，DS18B20 初始化，LCD 显示屏初始化。系统初始化流程图如图 4-2 所示。

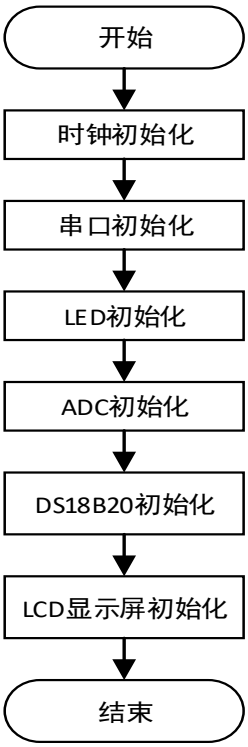


图 4-2 系统初始化流程图

4.3 数据采集端的软件设计

本系统采用了两个通道进行数据采集，分别是内部温度传感器和 DS18B20 温度传感器。

4.3.1 STM32 内部温度传感器的数据采集

只需设置内部 ADC，并激活其内部通道便可以使用内部温度传感器。通过设置 ADC_CR2 的 AWDEN 位（bit23），设置该位为 1 启用内部温度传感器。另外因为 STM32 的内部温度传感器是芯片内部直接连接在 ADC 通道 16 上，所以设置好 ADC 后只需要读取通道 16 的值，就是传感器返回的电压值，根据这个数值可以计算出当前温度，计算公式如下：

$$T（℃）=[(V25-Vsence)/Avg\_Slope]+25$$

上式中：

V25：Vsencs 在 25 摄氏度时的数值（典型值 1.43）

Avg_Slope：温度与 Vsence 曲线的平均斜率（典型值 4.3）

利用公式，便可以计算出温度值。

4.3.2 DS18B20 温度传感器的数据采集

要对 DS18B20 进行温度读取，需要对传感器复位脉冲和应答脉冲控制，单总线上的所有通信都是以初始化序列开始。STM32 控制通道输出低电平，保持低电平时间至少 480us，，以产生复位脉冲。接着主机释放总线，4.7K 的上拉电阻将单总线拉高，延时 15～60us，并进入接收模式(Rx)。接着 DS18B20 拉低总线 60~240us，以产生低电平应答脉冲，若为低电平，再延时 480us。

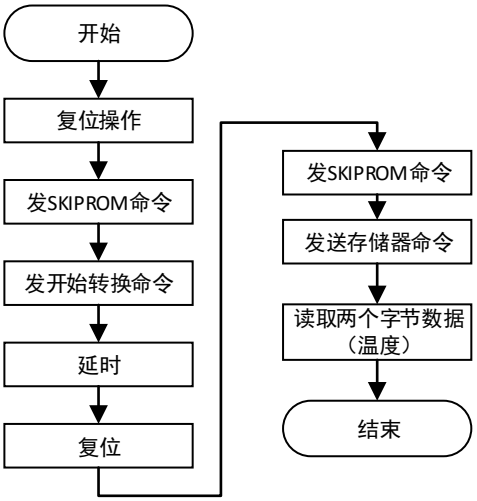


图 4-3 DS18B20 温度采集流程图

需要对时序进行写操作和读操作。写时序包括写 0 时序和写 1 时序。所有写时序至少需要 60us，且在 2 次独立的写时序之间至少需要 1us 的恢复时间，两种写时序均起始于主机拉低总线。写 1 时序：主机输出低电平，延时 2us 释放总线再延时 60us。写 0 时序：主机输出低电平，延时 60us 释放总线再延时 2us。单总线器件仅在主机发出读时序时，才向主机传输数据。所以，在主机发出读数据命令后，必须马上产生读时序，以便从机能够传输数据。所有读时序至少需要 60us，且在 2 次独立的读时序之间至少需要 1us 的恢复时间。每个读时序都由主机发起，至少拉低总线 1us。主机在读时序期间必须释放总线，并且在时序起始后的 15us 之内采样总线状态。典型的读时序过程为：主机输出低电平延时 2us，然后主机转入输入模式延时 12us，然后读取单总线当前的电平，然后延时 50us。DS18B20 的温度采集流程如图 4-3 所示。

4.4 数据处理和上传的软件设计

传感器采集到的温度值经过简单的格式转换就可以在 LCD 显示，但要通过 SIM808 通信模块上传至 ONENET 服务器端，则需要按照 ONENET 提供的 API 协议进行数据包封装。另一方面 GSM 通信模块是使用 AT 指令进行通信和数据传输。

4.4.1 AT 指令

AT 即 Attention，AT 指令集是从终端设备或数据终端设备向终端适配器或数据电路终端设备发送的。其对所传输的数据包大小有定义。即对于 AT 指令的发送，除 AT 两个字符外，最多可以接收 1056 个字符的长度（包括最后的空字符）。每个 AT 命令行中只能包含一条 AT 指令。对于由终端设备主动向 PC 端报告的 URC 指示或者 response 响应，也要求一行最多有一个。不允许上报的一行中有多条指示或者响应。AT 指令必须以“AT”或“at”开头。以回车（<CR>）结尾。模块的响应通常紧随其后，格式为：<回车><换行><响应内容><回车><换行>。

部分 AT 指令如表 4-1 所示：

表 4-1 部分 AT 指令

命令	作用
AT	测试连接是否正确
ATE0	关闭回显
ATE1	打开回显
AT+CGMI	得到厂商信息
AT+CSQ	当前信号
AT+COPS	网络营运商
AT+CGMR	获得改订的软件版本
ATD	拨号命令
ATH	接电话
AT+CIPSTART	建立 TCP/UDP 连接，设置 IP 地址端口号
AT+CIPSEND	通过 TCP/UDP 传输数据
AT+CIPSHUT	断开 TCP/UDP 连接
AT+CIPCLOSE	关闭 TCP/UDP 连接
AT+CGATT?	覆盖到 GPRS 网络

4.4.2 设备连接至 ONENET

首先要在 ONENET 平台进行注册，并且创建一个 EDP 协议的产品，然后新建一个设备。会有一个设备 ID 以及设备的 APIkey，在连接 ONENET 以及数据上传时会用上设备 ID 以及 APIkey。此外还需要在设备中添加两个数据流用在存放上传的

CPU 内部温度以及 DS18B20 采集的温度值。

平台基本设置完成后，可以使用对 SIM808 通信模块发送 AT 指令进行服务器连接。每一条 AT 指令发送后，不管成功或者失败，SIM 模块都会有相应的数据回复，可以通过回复确定下一步的操作，进而一步一步实现连接。

4.4.3 数据包封装

本系统采用 EDP 中的 RestFulAPI 方式进行数据上传，这种方式以数据点上传，设备不会保持在线。RestFulAPI 基于 HTTP 协议和 json 数据格式，适合平台资源管理、平台与平台之间数据对接、使用短连接上报终端数据、时间序列化数据存储等场景。

数据的上传格式如图 4-4 所示：

```
POST /devices/29547977/datapoints HTTP/1.1
api-key:4=FIl6GSKGpTo5MFQrDHuxVrlUA=
Host:api.hecloud.com
Content-Length:64

{"datastreams":[{"id":"sys_time","datapoints":[{"value":20}]}]}

```

图 4-4 RestFulAPI 数据上传格式

其中“devices/”后面的一串数字为设备 ID，“api-key:”后面的字符串为设备的 APIkey，“64”为后面的 JSON 串长度，该长度必须准确，否则服务器端无法解析上传的数据，“sys_time”是设备里面添加的数据名称，“20”是上传的数据值。按照 RestFulAPI 方式封装数据包需要以下注意：

1、每行结束的回车换行符不能少。

2、Content-Length 字段的值必须准确。

图 4-4 的数据包即是上传一个数值 20 到设备 29547977 的 sys_time 数据流。通过程序设计，可以将数据包的封装成一个函数，在 SIM808 模块连接服务器成功后，便可以调用该函数实现数据上传。在程序设计中，将两个通道的温度采集放在函数 get_temp()里面，定义全局变量 cpu_temp 和 b_temp 分别存放 CPU 内部温度和 DS18B20 采集的温度值，通过 SendCmd()函数发送 AT 指令，并把返回的数据通过串口 1 发送到电脑串口调试助手显示，可以直观的发现指令以及数据发送情况。

restful_send()数据包封装和发送的函数，通过 aprintf 函数将设置的数据包格式化在一个字符串中，再通过 SendCmd()函数发送至服务器。程序先发送 CPU 温度，再发送 DS18B20 采集温度值。

4.5 ONENET 平台自定义设计

STM32 控制端软件设计完成后，还需要对 ONENET 服务器平台进行设置和应用设计。

平台注册完成后，新建一个公开协议 EDP 协议的产品 loco'sgrad-proj，并新建一个设备，如图 4-5 所示

图 4-5 新建产品和设备

然后添加两个数据流 cpu_temp 和 18b_temp，用来存放上传的 CPU 内部温度以及 DS18B20 采集的温度，两个数据流在设备中的情况如图 4-6 所示。

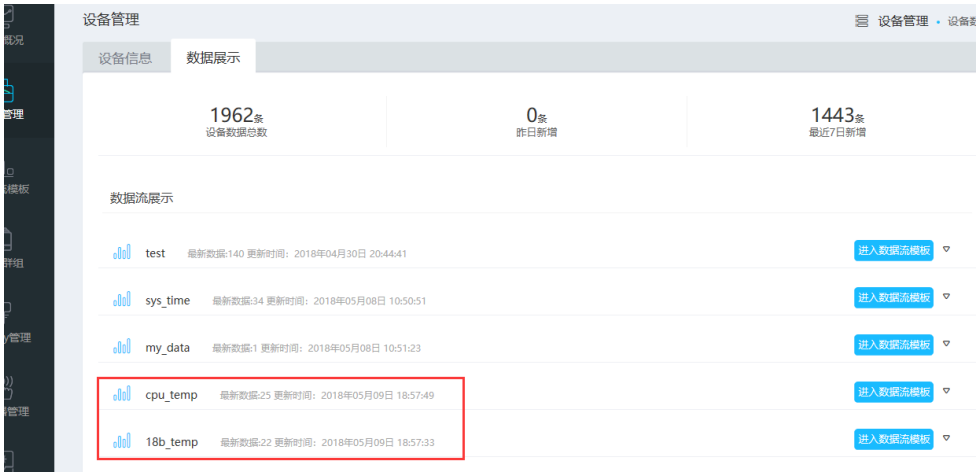


图 4-6 数据流界面

然后，还要创建一个模板应用，通过 ONENET 应用编辑器，用户可以方便快捷地实现 ONENET 平台上的设备数据流可视化。

编辑页面如图 4-7 所示：

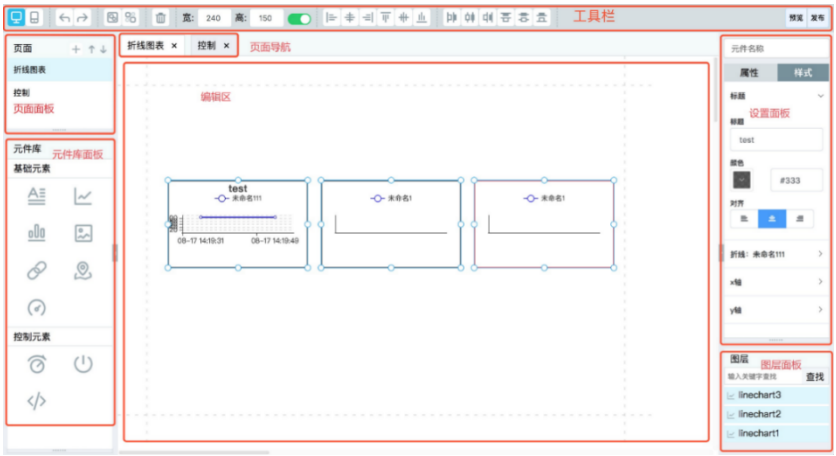


图 4-7 ONENET 设备应用编辑界面

该应用具有多页面支持、手机页面、页面设置等特点。

4.6 小结

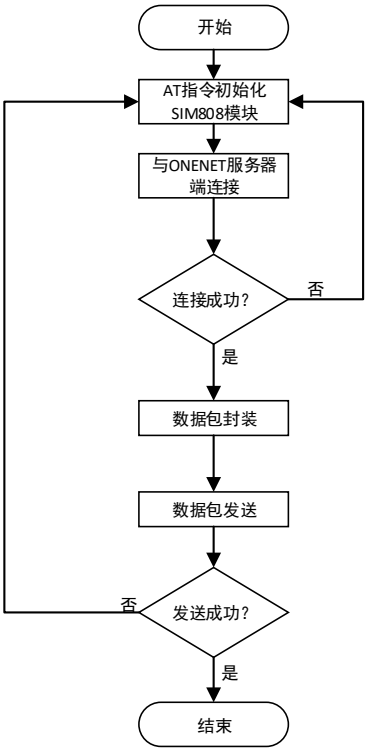


图 4-8 数据处理及上传流程图

数据处理与上传的流程如图 4-8 所示。在设计 ONENET 服务器连接以及数据包上传时，可以先利用电脑端提供 USB 转串口模块总结与 SIM808 模块连接，再使用串口调试助手发送 AT 指令以及数据包进行调试。在程序设计时利用 STM32 的串口 1 通过 USB 转串口连接电脑，将 SIM808 回复的数据通过串口 1 转发至电脑串口调试助手，可以更直观的看到连接情况和数据发送情况。

第5章 数据采集系统的调试及功能实现

在系统硬件搭建完成后，程序的设计与调试非常重要，数据处理和上传部分是这个系统的关键，数据需要按照 ONENET 提供的格式封装，先通过串口调试助手直接连接 SIM808 调试，上传数据成功后，进行程序的数据发送封装函数设计，是花费时间较多的点。

5.1 电脑直连通信模块调试

电脑串口调试助手按照波特率 115200 配置，通过串口调试助手向 SIM808 模块发送以下 AT 指令：

AT

AT+CGCLASS="B"

AT+CGDCONT=1,"IP","CMNET"

AT+CGATT=1

AT+CIPCSGP=1,"CMNET"

AT+CLPORT="TCP","2000"

每条指令若发送成功，模块会返回“OK”。指令发送情况如图 5-1 所示。

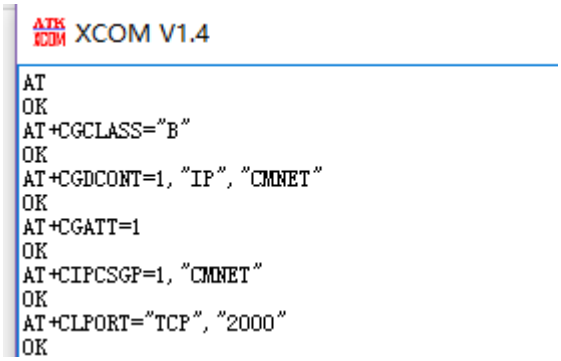


图 5-1 串口调试指令发送情况

这几个命令用于设置移动台类别、连接方式、接入点、附着 GPRS 业务、设置本地端口等。起到一个前期准备的作用。接下来就可以和 ONENET 建立 TCP 连接了，发送命令：AT+CIPSTART="TCP","183.230.40.33","80"和 ONENET 建立 TCP 连接，若连接成功，模块会返回“CONNET OK”此时发送命令 AT+CIPSEND 开始数据透传，模块回复“>”后便可将封装好的数据包发送。数据包发送完成后需要再以十六进制发送 0x1a 表示透传结束。

指令发送情况如图 5-2 所示。

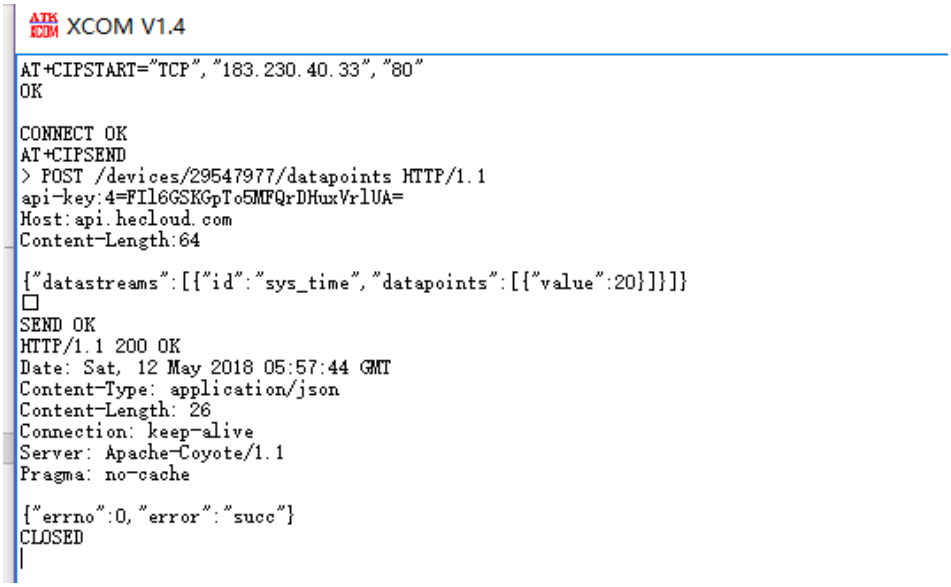


图 5-2 数据发送成功后串口显示情况

图中数据包下面一行的“□”字符即是 0x1a 的十六进制在窗口上显示的形式，数据发送成功服务器会返回相关参数，{"errno":0,"error":"succ"}表示错误数为 0，发送成功，并且服务器关闭了连接，这时可以在 ONENET 设备里面数据流 sys_time 会有一个数据点，并且可以看到该数据点上传的时间，数据上传的调试就成功了。如图 5-3 所示：

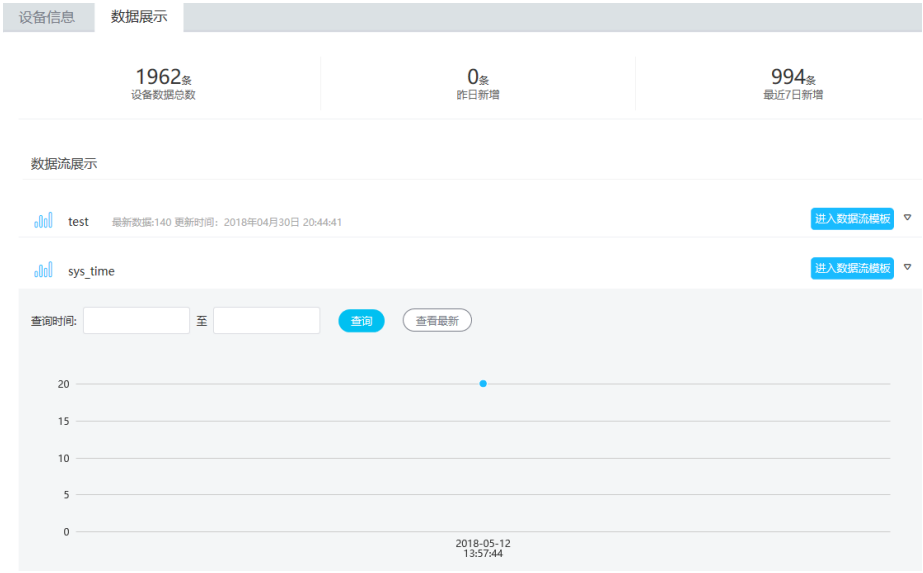


图 5-3 ONENET 端数据点上传情况

5.2 数据封装及上传程序调试

根据 4.4.3 节的介绍，数据的封装和上传过程主要是 SendCmd()和封装的 sprintf

函数的调试，根据串口调试助手返回的数据情况，更改程序，正确的返回情况如图5-4所示。

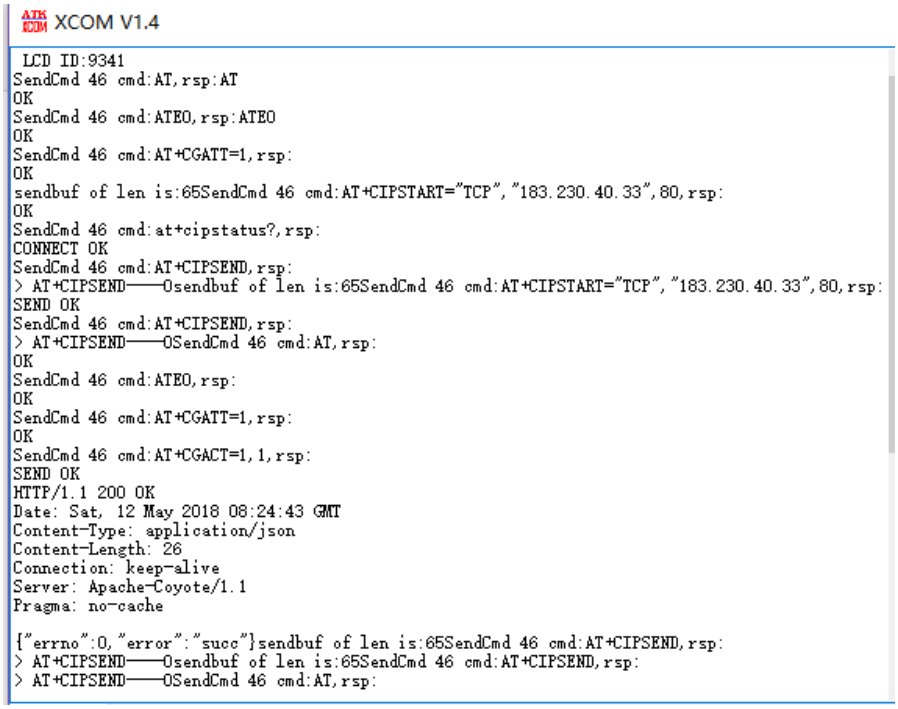


图 5-4 STM32 控制数据发送情况串口显示图

指令 at+cipstatus?在判断连接服务器是否成功，回复 CONNECT OK 后发送AT+CIPSEND 开始发送数据，窗口显示 send buf of len is:65 表是发送的数据长度，即数据发送成功。

图 5-5 ONENET 端数据流数据上传情况图

程序会一直循环发送两个采集的温度值，ONENET 平台该设备的数据流可以看到上传的数值，因为 STM32 在运行会有发热，所以内部温度传感器采集的温度值会比 DS18B20 采集的温度高一点。ONENET 平台数据流显示如图 5-5 所示。

最后，在 ONENET 平台设计一个模板应用，添加一个折线图并把相关的两个数据流 18b_temp 和 cpu_temp 关联至图例，然后添加两个仪表盘同样关联两个数据流，可以直观的显示数据情况，如图 5-6 所示。

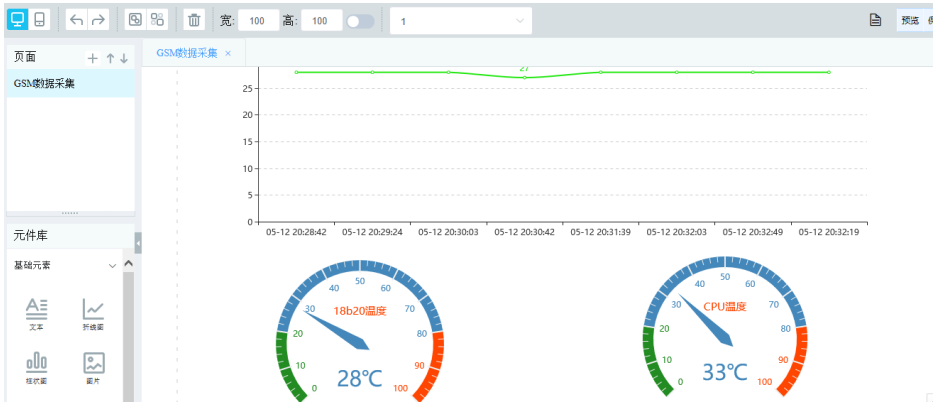


图 5-6 ONENET 设备模板应用运行图

5.3 设计成果展示

本数据采集系统是以 GSM 通信为研究对象，基于 STM32 高性能处理器实现数据采集以及无线远端上传，同时在 LCD 屏进行数据的显示。控制和采集端运行效果如图 5-7 所示。

图 5-7 STM32 端系统运行情况

ONENET 平台设备模板应用数据上传情况可以在任何网络浏览器（包括手机浏览器）进行个人 ONENET 账户登录后查看，如图 5-8 所示。

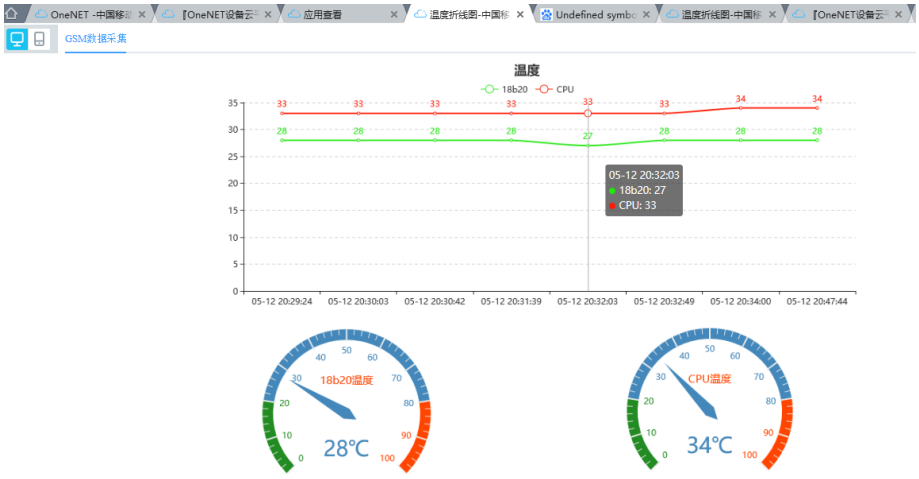


图 5-8 ONENET 端应用数据上传情况图

ONENET 还针对 Android 系统开发了 APP，也可以用手机安装 APP 进行登录查看数据情况。

结论

为满足社会相关行业对远程数据采集，监控和查询的需求，本文将 GSM 无线通信与物联网开发平台引入数据采集系统，设计了一种用户可远程通过浏览器登陆物联网平台查询采集数据的系统，本文在分别硬件和软件方面对该系统进行了深入分析和详细介绍，主要包括以下几点:

1、基于数据采集相关技术要求，设计数据采集，数据处理，数据上传的系统运行结构，并针对具体功能要求，完成了相关硬件电路的设计。

2、基于本数据采集系统的硬件电路结构，设计了相应软件程序，完成从局部到整体的调试，并且整体系统运行稳定，达到设计要求。

3、根据数据传输速率和成本考虑，本设计弃用成本较高，格式单一，传输效率慢的 SMS 传输，而是采用 GSM 的 GPRS 进行数据上传，每次上传仅消耗几十个字节，也就是 1KB 流量可以发送 10 余次数据，成本非常低。

4、根据近几年发展飞速的物联网技术，本系统采用中国移动 ONENET 物联网开放平台进行远端数据的上传，实现远程查询与监控。

由于个人以及时间等因素，本数据采集系统在硬件电路设计，关键技术和软件设计等方面还存在着许多待改进和完善的地方，以后可以在一下几方面进行深入研究和进一步完善设计：

1、控制端和 GSM 通信端集成在一块电路板，设计更多的数据采集接口拓展更多的数据采集。

2、控制端增加对数据采集的逻辑控制和条件控制，使数据采集过程可以满足更多条件。

3、数据上传至 ONENET 可以使用 EDP 封装，保持设备在线，并在 ONENET 应用中添加控制以发送数据对本地数据采集进行控制。

致谢

四年的大学生活即将结束，在此期间，无论是学习上还是生活中，许多老师、同学和朋友都给予了我很大的帮助，使我在多个方面都取得了很大的进步。

首先，我要感谢父母，父母一直是我最大的依靠，每当我遇到困难，他们总是能给我最大的帮助，时光飞逝，我在成长，他们却变得长老，正是他们坚定的支持与无微不至的关怀才使我顺利完成四年大学学业。

然后还要感谢学校和老师，老师们学识渊博，在他们的带领和教育下四年来我在专业水平方面有了明显的提升。还要特别感谢 108，和 112 两个实验室的指导老师们，在他们的指导和带领下我在相关项目设计和制作以及动手能力有非常大的提升。

感谢两个实验室的学姐学长以及小伙伴们，这四年我们一起学习，一起娱乐，一起努力，一起奋斗，一起通宵……这些经历，是我这大学四年最宝贵的收获，谢谢你们！

感谢实习公司提供的 SIM808 通信模块，以及表哥提供的 STM32 开发板和 LCD 液晶屏，我才能顺利完成本设计。

最后，感谢所有的论文审阅老师，你们的意见非常宝贵！

参考文献

[1] 刘辉.基于 DSP 和 miniPIC 技术数据采集卡的设计[D].西安电子科技大学，2012.

[2] 孟强.基于 STM32 的数据采集系统设计[D].南京林业大学，2014

[3] 梁恺.数据采集系统的发展趋向.[J]航空测试技术,1982

[4] 刘宣.用电信息采集系统数据传输协议的发展趋势研究[J]通信技术.2016

[5] 王明新.基于 SIM900A 的 GSM 远程监控系统设计[J]电脑知识与技术.2014

[6] 赵圣飞.基于 STM32 的数据采集存储系统的设计与实现[D]中北大学.2014

[7] 樊龙.基于 STM32 的智能仪表数据采集系统的设计[D]太原理工大学.2014

[8] 赵国峰.基于 GSM 网络的嵌入式分布测控系统[D]浙江大学.2006

[9] 李家福.基于 GSM 网络的智能监控模块设计[D]西南交通大学.2006

[10] 徐建.基于 GSM 的温度采集与报警系统的设计[J]湖北民族学院学报.2014

[11] 田丽.基于 GSM 的温度测控系统设计[J]工业控制计算机.2017

[12] 陈宇瑞.基于 GSM 的智能家居安防系统设计[J]电子制作.2017

[13] 赵金峰.基于 GSM 网络的远程监控终端的设计与实现[D].武汉理工大学.2008

[14] 范存玉.基于 GSM 无线传输系统的研究与实现[D]太原理工大学.2009

[15] 史文证.基于 GSM 无线传输血氧仪的设计与实现[D]山东师范大学.2016

[16] WolfgangGerstacker.TransmissionTechniquesForVamosGSMInDownlink[Z].2011

[17] JosephYiu.TheDefinitiveGuidetotheARMCortex-M3[Z].STMicroelectronics

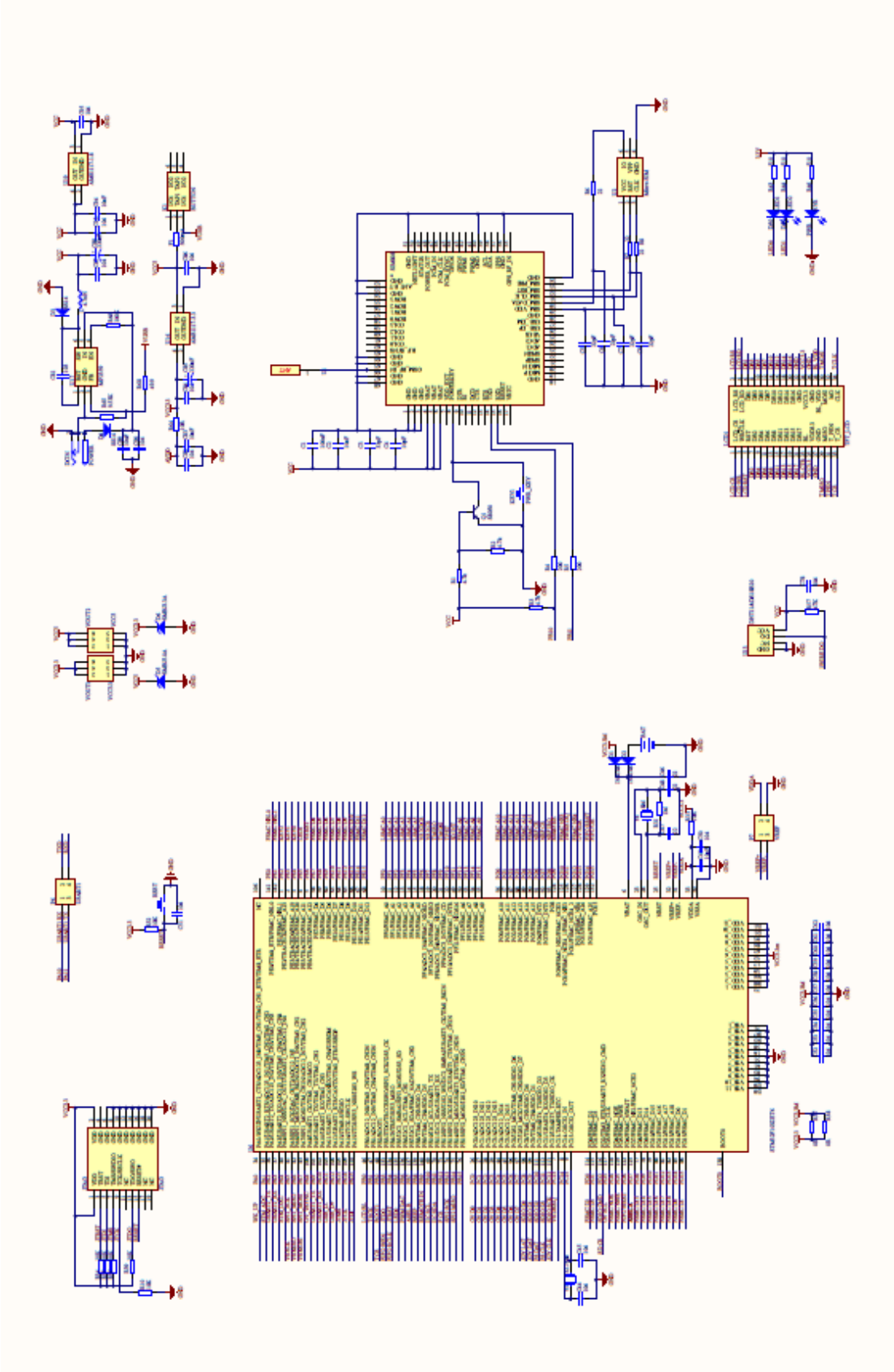
[18] STM32 中文参考手册 V10[Z]

[19] STM32F103x8STM32xBDATASHEET[Z].STMicroelectronics

[20] SIM808HardwareDesign[Z].SIMComWireless

[21] 中国移动 ONENET.设备终端接入协议-EDP[Z].V1.6

附录 1 系统原理图



附录 2 程序关键函数

系统初始化相关函数：

```
delay_init();            //延时函数初始化
NVIC_Configuration();//设置 NVIC 中断分组
USART2_Init(36,115200);
usart3_init(36,115200);
uart_init(9600); //串口 1 初始化
LED_Init();              //LED 初始化
KEY_Init();//按键初始化
LCD_Init();
T_Adc_Init();                    //ADC 初始化
DS18B20_Init();
usmart_dev.init(SystemCoreClock/1000000); //USMART 初始化
RTC_Init();                  //RTC 实时时钟初始化
```

两通道温度数据采集：

```
voidget_temp()
{
    floattemp,temp1,temp2;
    adcx=T_Get_Adc_Average(ADC_CH_TEMP,10);
    temp=(float)adcx*(3.3/4096);
    cpu_temp=temp;
    adcx=temp;
    b_temp=DS18B20_Get_Temp();
    if(b_temp<0)
    {
        LCD_ShowChar(60+40,70,'-',16,0);
        b_temp=-b_temp;
    }
    elseLCD_ShowChar(60+40,70,'',16,0);
    LCD_ShowNum(60+40+8,70,b_temp/10,2,16);
```

```
LCD_ShowNum(60+40+32,70,b_temp%10,1,16);

    cpu_temp=(1.43-cpu_temp)/0.0043+25;
    LCD_ShowxNum(60+14*8,90,(u8)cpu_temp,2,16,0);
    temp2=cpu_temp-(u8)cpu_temp;
    LCD_ShowxNum(60+24+14*8,90,temp2*100,2,16,0X80); LED0=!LED0;
    delay_ms(250);

}
```

与 ONENET 服务器连接;

```
SendCmd("AT","OK",100);
SendCmd("ATE0","OK",200);
SendCmd("AT+CGATT=1","OK",1000);
SendCmd("AT+CGACT=1,1","OK",1000);
```

封装温度数据包以及发送：

```
/*
//edprestfulAPI 数据上传
author：loco
相关参数:devid 上传设备 ID
api_key 上传设备 apiKey
de_id 数据流 ID
send_data 数据
*/
voidrestful_send(int8_t*devid,int8_t*api_key,constchar*de_id,intsend_data)
{
    intt=20,dosend_flag=0;
    charsendbuf[200];
    charlenbuf[200];
    intsend_data_len=0;
    SendCmd("AT+CIPSTART=\"TCP\",\"183.230.40.33\",80","OK",500);
```

```
        sprintf(sendbuf,"{\"datastreams\":[{\"id\":\"%s\",\"datapoints\":[{\"value\":%d}]
}]}",de_id,send_data);
        send_data_len=strlen(sendbuf)+2;
        printf("sendbufoflenis:%d\n",send_data_len);
    //delay_ms(100);
        SendCmd("at+cipstatus?","CONNECTOK",1000);
    cntflag=SendCmd("AT+CIPSEND",">",500);
        printf("AT+CIPSEND----%d",cntflag);
    //SendCmd("POST/devices/29547977/datapointsHTTP/1.1\r\napi-
key:4=FIl6GSKGpTo5MFQrDHuxVrlUA=\r\nHost:api.hecloud.com\r\nContent-
Length:64\r\n\r\n{\"datastreams\":[{\"id\":\"sys_time\",\"datapoints\":[{\"value\":20}]}]}
\r\n",NULL,100);
        sprintf(sendbuf,"POST/devices/%s/datapointsHTTP/1.1\r\napi-
key:%s\r\nHost:api.hecloud.com\r\nContent-
Length:%d\r\n\r\n{\"datastreams\":[{\"id\":\"%s\",\"datapoints\":[{\"value\":%d}]}]}\r\n"
,devid,api_key,send_data_len,de_id,send_data);
    SendCmd(sendbuf,NULL,100);
    DoSend(0,buf,strlen(buf));
    }
```