

学士学位论文

网络与串口调试助手的设计与实现

姓 名 苏卓锐

学 号 16852111

院 系 计算机与信息技术学院

专 业 软件工程

指导教师 徐林雪 主教

二〇二〇年 5月 日

学位论文原创性声明

本人所提交的学位论文《网络与串口调试助手的设计与实现》，是在导师的指导下，独立进行研究工作所取得的原创性成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中标明。

本声明的法律后果由本人承担。

论文作者（签名）： 指导教师确认（签名）：

年 月 日 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解北京交通大学海滨学院有权保留并向国家有关部门或机构送交学位论文的复印件和磁盘，允许论文被查阅和借阅。本人授权北京交通大学海滨学院可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或其它复制手段保存、汇编学位论文。

保密的学位论文在\_\_\_\_\_\_\_年解密后适用本授权书。（论文如无需保密，则删除此行）

论文作者（签名）： 指导教师（签名）：

年 月 日 年 月 日

摘 要

1980年起，由于一种状态信息病毒的横空出现，并在一种令人出乎意料的速度下开始繁衍，迫使ARPANET完全停止运行。美国政府的高级研究工程机构（ARPA）给加利福尼亚大学Berkeley分销提供了资金，让他们实现TCP/IP协议。在这个项目中Socket（套接字）横空出世。今天，SOCKET是TCP/IP网络中最为通用的API，也是在INTERNET上进行应用程序开发最为通用的API。

本软件基于Qt Creator编译器，C++语言开发。应用了程序设计技术、网络技术和图形化界面技术。使用了Qt封装好的Network模块和SerialPort模块。本课题的研究内容为，利用socket和串口协议编写TCP/UDP/串口的信息收发工具，以此功能为基础来达到调试设备的目的。本调试助手包括5种设备：TCP服务器设备，TCP客户端设备，UDP服务器设备，UDP客户端设备，串口设备。每个模块代表一个类型的设备。用户可创建多个类型的设备或者创建同一类型的多个设备。每种类型设备均可进行数据的收发。在发送数据时可设置数据格式为ASCII或Hex数据，支持单次发送和定时多次发送。支持数据文件导出等功能。TCP服务器用采用多线程来连接客户端。本软件的亮点在于集成了TCP，UDP，串口的调试功能，可提高学习和工作的效率。使本软件的面向群体更加广泛。通过测试，本软件界面美观，操作简单，功能模块一目了然，运行也比较稳定，可以广泛使用。

关键词：Qt；调试助手；TCP；UDP；串口通信

ABSTRACT

In the early 1980s, ARPANET was stopped completely due to an unexpected self-reproduction of a state information virus. The US Government's Advanced Research Engineering Agency (ARPA) has funded the University of California, Berkeley Distribution, enabling them to implement the TCP / IP protocol. Socket was born in this project. Today, SOCKET is the most common API in TCP / IP networks and the most common API for application development on the Internet.

This software is based on Qt Creator compiler and C ++ language development. Application of programming technology, network technology and graphical interface technology. Qt packaged Network module and SerialPort module are used. The research content of this subject is to use the socket and serial port protocols to write TCP / UDP / serial port information receiving and sending tools, and use this function as the basis to achieve the purpose of debugging equipment. The software includes 5 modules: TCP server, TCP client, UDP server, UDP client, serial port. Each module represents a type of device. Users can create multiple types of devices or multiple devices of the same type. Each type of device can send and receive data. When sending data, the data format can be set to ASCII or Hex data, which supports single transmission and timed multiple transmission. Supports data file export and other functions. The TCP server uses multiple threads to connect to the client. The highlight of this software is the integration of TCP, UDP, serial port debugging functions, which can improve the efficiency of learning and work. Make the software more community-oriented. Through testing, the software has a beautiful interface, simple operation, clear function modules at a glance, stable operation, and can be widely used.

**Key words：**Qt；Commissioning assistant；TCP；UDP；Serial communicatio

目 录

[第1章 绪论 1](#_Toc41205428)

[1.1 项目背景 1](#_Toc41205429)

[1.2 国内外研究现状 1](#_Toc41205430)

[1.3 主要研究内容 2](#_Toc41205431)

[1.4 论文组织结构 2](#_Toc41205432)

[第2章 软件需求分析 3](#_Toc41205433)

[2.1 可行性分析 3](#_Toc41205434)

[2.1.1 经济可行性 3](#_Toc41205435)

[2.1.2 技术可行性 3](#_Toc41205436)

[2.1.3 运行可行性 3](#_Toc41205437)

[2.2 需求概述 3](#_Toc41205438)

[2.3 功能需求分析 6](#_Toc41205439)

[2.3.1 设备模块 6](#_Toc41205440)

[2.3.2 设备信息模块 8](#_Toc41205441)

[2.3.3 发送数据模块 9](#_Toc41205442)

[2.3.4 数据接收模块 12](#_Toc41205443)

[2.3.5 系统设置模块 13](#_Toc41205444)

[2.3.5 其他功能模块 14](#_Toc41205445)

[2.4 非功能需求分析 21](#_Toc41205446)

[2.4.1 性能需求分析 21](#_Toc41205447)

[2.4.2 可靠性分析 21](#_Toc41205448)

[2.5 本章小结 21](#_Toc41205449)

[第3章 软件系统设计 21](#_Toc41205450)

[3.1 总体设计 21](#_Toc41205451)

[3.1.1 基本框架设计 21](#_Toc41205452)

[3.1.2 功能结构设计 22](#_Toc41205453)

[3.1.3 关键技术介绍 23](#_Toc41205454)

[3.2 详细设计 24](#_Toc41205455)

[3.2.1 设备模块 24](#_Toc41205456)

[3.2.2 设备信息模块 26](#_Toc41205457)

[3.2.3 发送数据模块 27](#_Toc41205458)

[3.2.4 数据接收模块 30](#_Toc41205459)

[3.2.5 系统设置模块 31](#_Toc41205460)

[3.2.6 其他功能模块 32](#_Toc41205461)

[3.3 本章小结 36](#_Toc41205462)

[第4章 软件实现 36](#_Toc41205463)

[4.1 整体界面 37](#_Toc41205464)

[4.1.1 界面设计 37](#_Toc41205465)

[4.1.2 软件架构设计 37](#_Toc41205466)

[4.2 设备模块 38](#_Toc41205467)

[4.2.1 MyObject类的设计 38](#_Toc41205468)

[4.2.2 设备创建 40](#_Toc41205469)

[4.2.3 设备删除 42](#_Toc41205470)

[4.3 设备信息模块 44](#_Toc41205471)

[4.4 发送数据模块 46](#_Toc41205472)

[4.4.1 单次发送 46](#_Toc41205473)

[4.4.2 定时发送 47](#_Toc41205474)

[4.4.3 设置发送模式 48](#_Toc41205475)

[4.5 数据接收模块 49](#_Toc41205476)

[4.6 系统设置模块 49](#_Toc41205477)

[4.7 其他功能模块 50](#_Toc41205478)

[4.7.1 清空数据接收框 51](#_Toc41205479)

[4.7.2 清空数据发送框 51](#_Toc41205480)

[4.7.3 加载文件 51](#_Toc41205481)

[4.7.4 导出数据 52](#_Toc41205482)

[4.7.5 显示字节数与重置计数 52](#_Toc41205483)

[4.7.6 功能详解 53](#_Toc41205484)

[4.8 本章小结 54](#_Toc41205485)

[第5章 软件测试 54](#_Toc41205486)

[5.1 测试原则 54](#_Toc41205487)

[5.2 测试环境 55](#_Toc41205488)

[5.3 设备模块测试 55](#_Toc41205489)

[5.3.1 创建设备 55](#_Toc41205490)

[5.3.2 删除设备 56](#_Toc41205491)

[5.4 设备信息模块测试 56](#_Toc41205492)

[5.5 发送数据模块测试 57](#_Toc41205493)

[5.5.1 单次发送功能测试 57](#_Toc41205494)

[5.6.2 定时发送功能测试 58](#_Toc41205495)

[5.6.3 设置发送模式功能测试 58](#_Toc41205496)

[5.6 数据接收模块测试 60](#_Toc41205497)

[5.7 系统设置模块 60](#_Toc41205498)

[5.8 其他功能模块测试 61](#_Toc41205499)

[5.8.1 清空数据接收框功能测试 61](#_Toc41205500)

[5.8.2 清空数据发送框功能测试 62](#_Toc41205501)

[5.8.3 加载文件功能测试 63](#_Toc41205502)

[5.8.4 导出数据功能测试 63](#_Toc41205503)

[5.8.5 显示字节数功能测试 64](#_Toc41205504)

[5.8.6 计数清零功能测试 64](#_Toc41205505)

[5.8.7 显示设备数功能测试 65](#_Toc41205506)

[第6章 总结与展望 66](#_Toc41205507)

[6.1 总结 66](#_Toc41205508)

[6.2 展望 66](#_Toc41205509)

[参考文献 67](#_Toc41205510)

[致 谢 68](#_Toc41205511)

# 第1章 绪论

本章主要讲述了本调试助手的项目背景与意义、国内外研究状况、主要研究内容和本论文的组织结构。

## 1.1 项目背景

计算机技术在不断发展的过程中，其具有很广的应用领域，在其应用的过程中，需要有效提升计算机应用的效果，在其具体实际的应用过程中存在一定问题，必须根据其实际应用领域不断完善，让计算机技术适合现代社会发展需要，为其它行业的发展提供技术支持，现在掌握计算机技术，利用计算机技术解决工作中的实际问题，已经成为人们的基本技能之一[[1]](#endnote-2)[1]。随着计算机的普及与科技时代的到来，嵌入式设备也得到了空前的发展与应用。与此同时，很多嵌入式设备为了方便系统的调试以及适应互联网的发展与更高的用户需求，需要连接网络与串口。它们不仅可以作为一种嵌入式设备的调试手段，也是嵌入式设备与外界进行信息交流的一种手段。另一方面，可以帮助与此技术相关的人员检查所开发的软硬件设备的通信情况，提高开发和调试的速度，让他们更容易了解网络通信应用。因此，有一个比较完善的调试助手用于开发和调试，许多事情将会变的更加简单。

## 1.2 国内外研究现状

中国计算机的发展从1955年左右开始，和国外同一时期的计算机水平相比，中国的计算机起步晚了约 10 年。在中国计算机的发展过程中，随着科研人员艰苦卓绝的奋斗，使中国的研制水平从与国外的差距整整一代直至达到国际前沿水平[[2]](#endnote-3)[2]。

目前，由于网络和通信技术的发展，嵌入式系统在经历了几十年的发展历程后，又进入了一个新的历史发展阶段，即从普遍低端应用进入到一个高、低端并行发展，并且不断提升低端应用技术水平的时代，其标志就是32位嵌入式系统的发展。随着城市信息化和行业信息化的持续深入，嵌入式技术已成为信息产业中发展最快、应用最广的计算机技术之一。网络、通信、多媒体、信息家电时代的到来，以及未来物联网的发展，为32位的嵌入式系统提供了巨大的发展空间[[3]](#endnote-4)[3]。而在这些行业快速发展时期，面向网络应用和嵌入式系统的调试助手，更是被许多开发爱好者设计出了多个版本。但是大多数年调试助手都只是针对某个连接方式的具体实现，没有整合成一个应用。

## 1.3 主要研究内容

本文主要讲述的是基于TCP、UDP和串口调试助手的设计与实现过程，包括需求分析，软件系统设计，软件实现与软件测试阶段。此研究的目的是为从事与嵌入式设备、计算机网络和工控技术相关的设计、开发、测试人员检查所开发的软硬件设备的数据收发状况，提高开发和调试的速度。

## 1.4 论文组织结构

第1章介绍了关于网络与串口调试助手的选题背景与选题意义，以及我国在目前在计算机应用方面的研究现状。

第2章介绍了网络与串口调试助手在需求分析阶段，该应用需要完成的功能模块，和软件的用例模型。

第3章介绍了网络与串口调试助手在设计时的思路，系统架构与公共数据定义。

第4章介绍了网络与串口调试助手各个功能模块的具体实现步骤、核心代码与效果展示。

第5章详细介绍了网络与串口调试助手在完成之后对其功能的测试以及各项的测试结果。

第6章介绍了本次毕业设计中对于软件功能的实现过程和优缺点的总结以及对于本软件的后续完善计划。

# 第2章 软件需求分析

需求分析是软件计划阶段的重要活动，也是软件生存周期中的一个重要环节 [[4]](#endnote-5)[4]。本章节主要介绍了关于此调试助手的需求分析和可行性分析的全部内容。分析了软件的可行性和其面向的对象及其模块和功能。

## 2.1 可行性分析

本小结主要讲述了从经济可行性、技术可行性和运行可行性三方面对本软件的可行性分析。

### 2.1.1 经济可行性

本调试助手的开发人员仅为本人一人。从毕设选题到项目完成，预估花费2个月时间，所花费的仅由个人的每日日常消耗和时间。此消耗完全在个人的承受范围内。

### 2.1.2 技术可行性

本软件的开发环境为windows系统下的Qt Creator编译器。编程语言为C++语言。本软件开发所用到的技术有：TCP/IP协议、面向Qt的socket编程、Qt的多线程、Qt的串口通信、Qt界面开发。除了串口通信和多线程外，其他均为本人在校期间所学过或用过的知识或技能。而串口通信和多线程，本人可以在项目过程中自行学习掌握。因此本项目的开发，在本人的技术范围之内。

### 2.1.3 运行可行性

本软件的开发环境为windows7 64位系统，4GB内存，处理器为i5-7200U。随着国民经济的发展，个人电脑的配置也逐步提高，大多数PC机均可达到运行配置。

## 2.2 需求概述

本小节针对此调试助手的功能需求进行简单的概述，从整体角度出发设计用力图与用例表。软件用户用例图如图2-1所示。



图2-1 软件用户用例图

下表是对设计好的用例图中个用例的说明。见表2-1。

表2-1 用例说明表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例名 | 入口条件 | 事件流 | 出口条件 | 异常事件 |
| 创建设备 | 用户选择一个类型的设备 | 用户输入所要创建的设备信息 | 关闭创建窗口，在设备区添加设备 | 输入错误的设备信息，创建失败 |
| 删除设备 | 用户算则一个已创建好的设备 | 删除所选设备 | 设备区删除设备 | 无 |
| 连接TCP服务器 | 用户选择一个未就绪的TCP客户端 | 击连接服务器按钮 | 设备已就绪 | 服务器不存在或为启动监听，连接失败 |

续表2-1 用例说明表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例名 | 入口条件 | 事件流 | 出口条件 | 异常事件 |
| 监听TCP服务器 | 用户选择一个未就绪的TCP服务器 | 点击监听按钮 | 设备已就绪 | 服务器端口被占用，或超出端口范围，监听失败。 |
| 打开串口 | 用户选择一个未就绪的串口设备 | 点击打开串口按钮 | 设备已就绪 | 串口已经不存在，打开失败 |
| 关闭串口 | 用户选择已就绪的串口设备 | 点击关闭串口按钮 | 设备未就绪 | 无 |
| 停止监听TCP服务器 | 用户选择一个已就绪的TCP服务器 | 点击停止监听按钮 | 设备未就绪 | 无 |
| 断开连接TCP服务器 | 用户选择一个已就绪的TCP客户端 | 点击断开连接按钮 | 设备未就绪 | 无 |
| 显示设备信息 | 用户选择一个已就绪的设备 | 更改选择已就绪的设备 | 在设备区显示当前设备信息 | 显示不正确 |
| 单次发送 | 用户选择一个已就绪的设备 | 在数据输入框中输入数据后点击发送按钮 | 发送成功，在数据接收框显示数据 | 不存在发送对象，发送失败 |
| 定时发送 | 用户选择已就绪的设备 | 在重复发送区域设置定时发送的规则后点击定时发送 | 发送成功，在数据接收框显示数据 | 不存在发送对象，发送失败 |
| 停止定时发送 | 用户选择一个已就绪并且已经定时发送的设备 | 点击停止定时发送 | 停止发送成功 | 无 |
| 加载文件 | 用户选择一个已就绪的设备 | 点击加载文件，选择一个数据文件 | 加载成功，在数据发送框显示文件路径 | 无 |
| 接收数据 | 用户选择一个已就绪的设备 | 接收从其他设备传来的数据 | 显示在数据接收框 | 无 |
| 导出数据到文件 | 用户选择一个已就绪的设备 | 点击导出数据按钮，选择导出路径 | 导出数据成功 | 无 |

续表2-1 用例说明表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用例名 | 入口条件 | 事件流 | 出口条件 | 异常事件 |
| 发送接收字节数 | 用户选择一个已就绪的设备 | 记录自己发送的字节长度，记录从其他设备传来的数据字节长度。 | 显示在软件下方 | 无 |
| ASCII模式 | 用户选择一个已就绪的设备 | 选择单选框字符发送 | 输入框切换为ASCII输入框 | 无 |
| Hex模式 | 用户选择一个已就绪的设备 | 选择单选框十六进制发送 | 输入框切换为Hex输入框 | 无 |
| 更改最大连接数 | 无 | 点击更改最大连接数，设置最大连接数 | 设置成功 | 设置数小于当前设备数，设置失败 |

## 2.3 功能需求分析

需求分析是软件生命周期中一个非常重要的过程，它决定着整个软件项目的质量，也是整个软件项目的成败所在。要求软件各个功能都可正常使用，且界面美观。

### 2.3.1 设备模块

调试助手的设备模块主要是使用户进行创建和删除设备的操作。对于创建设备，用户可以选择创建TCP类型的客户端和服务器或者UDP类型的客户端和服务器。串口设备自动检测创建。对于删除设备，是对已创建的设备进行删除。创建设备的活动图如图2-2所示。



图2-2 创建设备的活动图

删除设备的活动图如图2-3所示。



图2-3 删除设备的活动图

### 2.3.2 设备信息模块

用户在进行真实调试过程中，往往需要查看已创建设备的信息，而为了方便用户使用，因此创建了设备信息模块来显示或设置不同设备的基本信息。显示设备信息活动图如图2-4所示。



图2-4 显示设备信息活动图

### 2.3.3 发送数据模块

发送数据模块主要实现了用户用一个设备向目标设备发送信息。包括单次发送、重复发送和设置发送模式三个功能。单次发送功能为仅发送一次信息。重复发送功能为定时的向目标设备发送重复的信息。设置发送模式功能为设置发送的数据格式为Hex模式，还是ASCII模式。单次发送功能的活动图如图2-5所示。



图2-5 单次发送功能的活动图

重复发送功能的活动图如图2-6所示。



图2-6 重复发送功能的活动图

设置发送格式功能的活动图如图2-7所示。



图2-7 设置发送格式功能的活动图

### 2.3.4 数据接收模块

调试助手的数据接收模块主要是使设备接收到来自其他设备所发送来的消息。数据接收功能的活动图如图2-8所示。

. 

图2-8 数据接收功能的活动图

### 2.3.5 系统设置模块

系统设置模块主要为显示当前设备数与设备总数，更改最大连接数。更改最大连接数活动图如图2-9所示。



图2-9 更改最大连接数活动图

### 2.3.5 其他功能模块

其他功能模块主要实现了为了提高软件实用性和舒适性的一些小功能。包括清空数据接收框、清空数据发送框、加载文件、导出数据、显示字节数、重置计数和功能详解。清空数据接受框活动图如图2-10所示。



图2-10 清空数据接受框活动图

清空数据发送框活动图如图2-11所示。



图2-11 清空数据发送框活动图

加载文件的活动图如图2-12所示。



图2-12 加载文件的活动图

导出数据的活动图如图2-13所示。



图2-13 导出数据的活动图

显示字节数的活动图如图2-14所示。



图2-14 显示字节数的活动图

重置计数的活动图如图2-15所示。



图2-15 重置计数的活动图

## 2.4 非功能需求分析

### 2.4.1 性能需求分析

本软件可创建TCP服务器使多个TCP客户端连接，但为了保证软件及其操作系统的流畅性，本软件默认设定所有设备的最大连接数为10，即所有设备的数量超过10台后不可在进行创建设备。但在特殊情况下可能需要超过10台设备，我们提供了一个不负责任的按钮用来更改最大连接数。

### 2.4.2 可靠性分析

软件健壮性强，需要能处理用户的错误操作或常见错误，做到主动规避错误。对输入数据有检查，防止数据异常。

## 2.5 本章小结

本章节介绍了本项目在需求分析阶段所应该做的内容，其中包含了需求分析和可行性分析。其中，需求分析阶段包括了功能需求分析、用例模型设计、性能需求分析和可靠性分析。而可行性分析中包含了经济可行性、技术可行性和运行可行性等。通过这些方面的分析来确保项目的正常实施。

# 第3章 软件系统设计

本章节对软件的大概结构和总体设计做出了详细的说明。也对项目中的重要模块进行了单独的讲述与介绍。

## 3.1 总体设计

总体设计是整个系统设计的关键，它的可行性、正确性等直接决定了系统的功能与性能。因此在本章中，我们结合组态软件的特点、用户需求分析、功能模块等来确定合理的总体设计方案[[5]](#endnote-6)[5]。本软件作为一款网络和串口调试助手，需要包括对TCP、UDP和串口的调试作用。（如：发送信息以及接受信息。）以及为了调试过程中的便利而对应的一些相应功能。（如：定时发送和Hex模式发送等。）

### 3.1.1 基本框架设计

本软件包含了五个模块：TCP类型的服务器设备、TCP类型的客户端设备、UDP类型的服务器设备、UDP类型的客户端设备、串口设备。而为了用户能准确的分辨自己所创建的设备是哪种设备和设备的基本信息，我们有必要创建出设备区域模块和设备信息模块。而对调试功能来说，需要的是发送数据模块和接收数据模块。以上为软件的基本模块。而对于各项目而言，我们有必要设计一个公共系统模块来设置和显示系统信息和数据。软件整体界面设计图如图3-1所示。



图3-1 软件整体界面设计图

### 3.1.2 功能结构设计

根据本文上文所述的功能需求分析，可以总结出以下功能。

（1）可创建和删除五种类型设备：TCP类型的服务器设备、TCP类型的客户端设备、UDP类型的服务器设备、UDP类型的客户端设备、串口设备。

（2）可同时创建多设备。

（3）ASCII/Hex模式发送数据。

（4）[Hex]ASCII模式接收数据。

（5）单次发送/定时多次发送。

（6）清空数据显示框和清空发送框。

（7）显示发送接收字节数和重置计数。

（8）导入外部数据源文件。

（9）导出数据到文件。

（10）更改最大设备连接数。

软件功能结构图如图3-2所示。



图3-2 软件功能结构图

### 3.1.3 关键技术介绍

本小结重点介绍了本软件开发过程中使用到的一些关键技术，包括开发工具与相应的开发环境。

（1）Qt

Qt是一个全面的 C+ + 应用程序开发框架。它包含一个类库 ,和用于跨平台开发及国际化的工具。由挪威 Tro lltech 公司推出 ,主要通过汇集 C+ + 类的形式来实现提供开发应用程序用户界面部分的需求[[6]](#endnote-7)[6]。它既可以开发GUI程序，也可用于开发非GUI程序，比如控制台工具和服务器。Qt是面向对象的框架，使用特殊的代码生成扩展（称为元对象编译器(Meta Object Compiler, moc)）以及一些宏，Qt很容易扩展，并且允许真正地组件编程。Qt的优点有：具有优良的跨平台特性、面向对象、丰富的API、支持2D/3D图形渲染，支持OpenGL、拥有大量的开发文档、支持XML。

（2）TCP/IP和SOCKET

TCP/IP 是目前广泛采用的一组完整的网络协议,在网络通信中被广泛应用。 TCP/IP协议的核心是TCP、UDP和IP协。TCP /IP网络环境下应用程序通过网络系统编程界面套接字(Socket)实现与内核的交互。用户开发的网络通信应用程序不直接与TCP/IP核心打交道,而是与网络应用编程界面Windows Sockets API 打交道。WinSock(Windows Socket)是Windows操作系统下的通用的TCP/IP应用程序的网络编程接口,由于Windows操作系统本身和开发工具均提供了较多的Socket支持, 故应用程序采用Windows Sockets(WinSock)实现网络通信[[7]](#endnote-8)[7]。

Socket 为套接字，是一种双向的通信端口。网络程序设计全靠套接字接受和发送信息[[8]](#endnote-9)[8]。

（3）串口通信

串行通信是指仅使用一条数据线，将数据从低位到高位或从高位到低位由一个通信系统一位一位地依次传输到另一个通信系统，每一位数据传输时使用一定的时间长度，因其所需传输线少，一般只需少数几根传输线就能完成多系统之间的信息交换，而且一个方向顶多只需要一条传输线，成本低，适合远距离传输，也经常用于计算机与其串行外部设备之间的信息交换[[9]](#endnote-10)[9]。

## 3.2 详细设计

### 3.2.1 设备模块

本调试助手的设备模块主要用来显示设备名称与设备数量。用树形结构实现。初始化时显示：TCP服务器、TCP客户端、UDP服务端、UDP客户端、串口五个类型名称。通过鼠标点击选中一个类型名称来进行创建设备（串口设备无需主动创建，在打开软件后自行检测串口设备）。创建设备，在点击创建后弹出对应设备类型的设置设备基本信息的对话框。如：设置IP地址或设置端口号。创建完成后在设备类型名下新建一个节点来显示当前设备名称。创建设备流程图如图3-3所示。



图3-3 创建设备流程图

删除设备，通过鼠标点击选中一个已经创建好的设备，点击删除后，删除所选中设备。删除设备流程图如图3-4所示。



图3-4 删除设备流程图

### 3.2.2 设备信息模块

用户选择一个已创建设备，在软件上部的当前设备信息区显示当前所选中设备的信息。TCP的设备信息模块中包括：设备名称、设备当前状态、本机IP地址、监听端口号、已连接的客户端、监听按钮。TCP客户端的设备信息模块中包括：设备名称、设备当前状态、本机IP地址、服务器IP地址、服务器端口号、连接按钮。UDP服务器的设备信息模块中包括：设备名称、设备当前状态、本机IP地址、本机端口、 对方IP地址、对方端口号。UDP客户端的设备信息模块中包括：设备名称、设备当前状态、本机IP地址、对方IP地址、对方端口号。串口设备的设备信息模块包括：设备名称、设备状态、波特率、校验方式、停止位、数据位、打开串口按钮。显示设备信息流程图如图3-5所示。



图3-5 显示设备信息流程图

### 3.2.3 发送数据模块

发送数据模块主要用来让每个已就绪的设备发送信息或设置发送模式。单次发送，在设备区域选择一个已就绪的设备后在数据发送窗口中输入需要发送的数据后点击发送数据按钮即可以向对方发送数据。单次发送流程图如图3-6所示。



图3-6 单次发送流程图

定时发送，在设备区域选择一个已就绪的设备后在数据发送窗口中输入需要发送的数据后，在重复发送框中设置发送次数和发送间隔后点击定时发送按钮即可以向对方定时循环发送数据。定时发送流程图如图3-7所示。



图3-7 定时发送流程图

设置发送模式，在用户发送数据之前可以选择发送模式为字符发送（ASCII）和十六进制发送（Hex） 。设置发送模式流程图如图3-8所示。



图3-8 设置发送模式流程图

### 3.2.4 数据接收模块

数据接收模块主要负责接收其他应用或设备向本设备发送来的消息并处理消息为ASCII模式和Hex模式两种，减少了用户的操作负担。数据接收模块的流程图如图3-9所示。



图3-9 数据接收模块的流程图

### 3.2.5 系统设置模块

系统设置模块主要为显示当前设备数与设备总数，更改最大连接数。显示当前设备数与设备总数时，在软件底部显示，显示格式为：当前设备数/最大设备数。更改最大连接数时，在弹出的对话框中选择是默认设置、自定义还是无限制。默认设置中最大连接数为10个。自定义中最大连接数随意设置，但若设置的连接数小于当前设备数，则设置失败。无限制为最大连接数不做限制。若设置最大连接数时失败，则先删除多余设备后，在进行更改。更改最大连接数流程图如图3-10所示。



图3-10 更改最大设备数流程图

### 3.2.6 其他功能模块

其他功能模块中的功能主要用来方便用户使用，提高软件的使用舒适性。

清空数据接收框，点击清空数据接收框按钮后，使所选已就绪设备的数据接收框数据清空，避免数据量过多而看不清数据内容。清空数据接收框流程图如图3-11所示。



图3-11 清空数据接收框流程图

清空数据发送框，点击清空数据发送框按钮后，使所选已就绪设备的数据发送框中数据清空。清空数据发送框流程图如图3-12所示.



图3-12 清空数据发送框流程图

加载文件，选择一个数据文件导入数据发送框中，设置数据发送框为不可更改，点击发送数据后，把文件中的内容发送出去。点击停止加载后设置数据发送框可更改。加载文件流程图如图3-13所示。



图3-13 加载文件流程图

导出数据，点击导出数据后，打开一个保存文件对话框，设置文件路径后使所选已就绪设备中的数据接收框中数据写入文件中。导出数据流程图如图3-14所示。



图3-14 导出数据流程图

显示字节数与重置计数，选择一个已就绪设备后为显示该设备的接收字节数和发送字节数。在有数据传来时，使接收字节数累加。在发送数据时，使发送字节数累加。在点击计数清零后使接收字节数和发送字节数清零。计数清零流程图如图3-15所示。



图3-15 计数清零流程图

功能详解，对本软件进行了介绍，对软件所有功能进行简单的叙述。

## 3.3 本章小结

本章主要介绍了本软件的软件设计方案。在基本框架小节里，对软件的界面规划与功能模块进行了划分。剩余的设备模块小节、设备信息模块小节、发送数据模块小节、数据接收模块小节、其他功能模块小节，对每个模块中的主要功能进行了详细的叙述。

# 第4章 软件实现

本章节主要是继以上章节的分析与设计后对网络与串口调试助手的各个功能与模块的实现方式与效果进行详细的讲述与描述。为后续的开发与维护做好铺垫。

## 4.1 整体界面

软件界面用Qt带有的Qt设计师界面类实现。Qt设计师界面主要包括一个.h页面、一个.cpp页面和一个.ui页面。通过在.ui页面中放置和设置控件来实现页面的框架设计，通过.h与.cpp来进行页面中控件代码的控制。

### 4.1.1 界面设计

在mainwindow.ui中放置QTreeView来显示设备区域。通过textBrower与textEdit来进行设备的输入与输出。通过多个按钮来进行功能实现和页面跳转。通过多个QLabel来显示一些设备信息。通过QLineEdit与QComboBox来进行输入和选择设备信息。下面是软件界面截图，如图4-1所示。

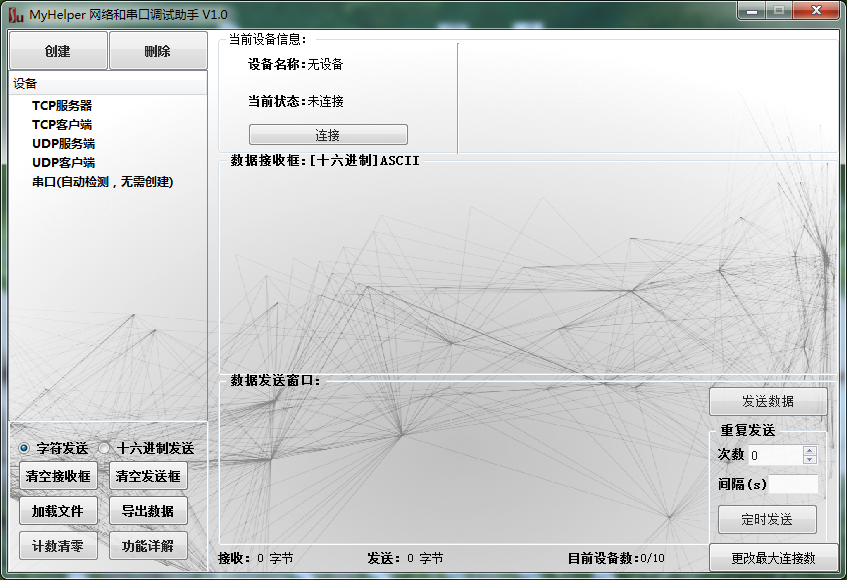


图4-1 软件界面截图

### 4.1.2 软件架构设计

（1）软件有5种功能类似的设备类型，所以创建一个基类MyObject来让五种类型的设备继承来减少不必要的代码量。

（2）用QList<ClassType>链表来管理五种设备。通过泛型来控制设备类型。

（3）代码与控件之间的联系，不同页面代码与代码之间的联系，用Qt独有的一个方法信号与槽来实现。

（4）TCP服务器，创建一个tcpServer类用来实现TCP服务器，该类继承MyObject和QTcpServer。在创建一个类socketRun，用来实现多线程客户端的连接。

（5）创建一个类tcpServerManagement来管理TCP服务器，该类继承QObject。

（6）TCP客户端，创建一个tcpClient类用来实现TCP客户端，该类继承QObject和MyObject。

（7）UDP服务器，创建一个udpServer类用来实现UDP服务端，该类继承QObject和MyObject。

（8）UDP客户端，创建一个udpClient类用来实现UDP客户端，该类继承QObject和MyObject。

（9）串口，创建一个SerialPort类用来实现串口设备，该类继承QSerialPort和MyObject。

（10）在创建设备时会弹出一个相应的对话框来进行设备的创建，所以创建界面设计师界面：类createTcpClient、类createTcpServer2、类createUdpClient、类createUdpServer来进行设备的创建。

（11）设置界面，创建一个界面设计师界面SettingDialog类。

（12）功能详解界面，创建一个界面设计师界面HelpDialog类。

（13）公有类，创建一个类common，继承QTextEdit，重写键盘事件来进行限制输入。

## 4.2 设备模块

设备模块公有五种类型设备：TCP服务器、TCP客户端、UDP服务器、UDP客户端、串口。而五种类型全部继承MyObject类。

### 4.2.1 MyObject类的设计

MyObject类作为五种设备的基类拥有所有设备类的共有属性和方法。而五种设备的不同也导致部分方法内容的不同。通过使用纯虚函数来使各个子类实现重写。设备共有属性表见表4-1。

表4-1 设备共有属性表

|  |  |  |  |
| --- | --- | --- | --- |
| 修饰类型 | 成员类型 | 成员名 | 描述 |
| Public | QTextEdit \* | textEdit | 设备ASCII输入 |
| Public | QTextBroser \* | textBroser | 显示数据 |
| Public | common \* | hexEdit | 设备Hex输入 |

续表4-1 设备共有属性表

|  |  |  |  |
| --- | --- | --- | --- |
| 修饰类型 | 成员类型 | 成员名 | 描述 |
| Public | bool | isFile | 是否正在文件加载 |
| protected | QString | myIP | 自身IP |
| protected | QString | otherIP | 对方IP |
| protected | int | myPort | 自身端口 |
| protected | int | otherPort | 对方端口 |
| protected | int | falgcount | 设备标志数 |
| protected | int | type | 设备类型 |
| protected | int | ricvBit | 接收字节数 |
| protected | int | sendBit | 发送字节数 |
| protected | QTimer \* | m\_timer | 定时器对象 |
| protected | int | timerCount | 总发送次数 |
| protected | int | nowCount | 当前发送次数 |
| protected | QString | addr | 发送对象 |
| protected | QString | msg | 发送消息 |
| protected | int | sendType | 发送的消息类型 |
| protected | bool | isTimeSending | 是否正在定时发送 |
| protected | QByteArray | data | 字节数据 |

设备共有函数表见表4-2。

表4-2 设备共有函数表

|  |  |  |
| --- | --- | --- |
| 修饰类型 | 函数 | 描述 |
| public | virtual void sendMsg(QString str,QString addr="")=0 | 设备发送ASCII消息 |
| public | virtual void sendHexMsg(QString msg,QString addr="")=0 | 设备发送Hex消息 |
| public | virtual void timerSend(QString msg, float s,int count,int sendType,QString add="")=0 | 定时发送 |
| public | void stopTimer() | 停止定时发送 |
| public | void clearData() | 清空接收框数据 |

续表4-2 设备共有函数表

|  |  |  |
| --- | --- | --- |
| 修饰类型 | 函数 | 描述 |
| public | void saveData() | 导出数据到文件 |
| public | void setDeviceType(int type) | 设置设备类型 |
| public | int getDeviceType() | 获取设备类型 |
| public | QString getMyIP() | 获取自身IP |
| public | QString getOtherIP() | 获取对方IP |
| public | int getMyPort() | 获取自身端口 |
| public | int getOtherPort() | 获取对方端口 |
| public | int getRicvBit() | 获取接收字节数 |
| public | int getSendBit() | 获取发送字节数 |
| public | int getflagcount() | 获取设备标志数 |
| public | void setZero() | 重置计数 |
| public | bool isTimerSending() | 是否定时发送 |
| public | void clearSend(int type) | 清空发送框 |
| public | void loadFile(QString path,int type) | 加载文件 |
| public | QString getRedString(QString str) | 获取红色QString |
| public | QString getBlueString(QString str) | 获取蓝色QString |
| private | QString getGreenString(QString str) | 获取绿色QString |

### 4.2.2 设备创建

通过在QTreeView选择相应的设备类型后点击创建按钮进行创建设备。串口设备无需主动创建。创建服务器的对话框如图4-2所示。

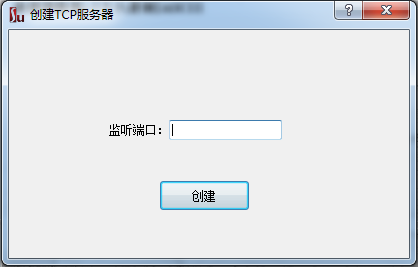


图4-2 创建服务器对话框

创建客户端对话框如图4-3所示。

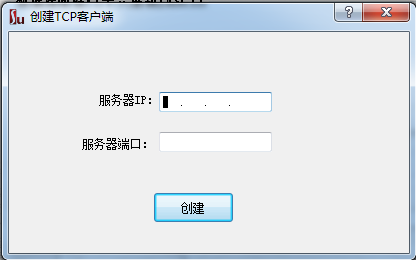


图4-3 创建客户端对话框

部分代码如下。

1. //创建子树设备
2. void MainWindow::createTreesun()
3. {
4. if(DeviceNum>=MaxDeviceNum)
5. {
6. QMessageBox::warning(this,"ERROR","超过最大设备数，创建失败!",nullptr,nullptr);
7. return;
8. }
9. if(ui->treeView->currentIndex().data().toString()=="TCP服务器")
10. {
11. class CreateTcpServer2 \*dialog=new class CreateTcpServer2();
12. dialog->show();
13. connect(dialog,&CreateTcpServer2::createSignal,this,&MainWindow::createTcpServer);
14. }
15. else if (ui->treeView->currentIndex().data().toString()=="TCP客户端") {
16. class createTcpClient \*dialog=new class createTcpClient();
17. dialog->show();
18. connect(dialog,&createTcpClient::createSignal,this,&MainWindow::createTcpClient);
19. }
20. else if (ui->treeView->currentIndex().data().toString()=="UDP服务端") {
21. class createUdpServer \*dialog=new class createUdpServer();
22. dialog->show();
23. connect(dialog,&createUdpServer::createSignal,this,&MainWindow::createUdpServer);
24. }
25. else if (ui->treeView->currentIndex().data().toString()=="UDP客户端") {
26. class createUdpClient \*dialog=new class createUdpClient();
27. dialog->show();
28. connect(dialog,&createUdpClient::createSignal,this,&MainWindow::createUdpClient);
29. }
30. }

### 4.2.3 设备删除

通过在QTreeView中选择所要删除的设备后点击删除按钮后删除设备。

部分代码如下。

1. //删除子树
2. void MainWindow::deleteTreesun()
3. {
4. //判断所选item的父亲来确定类型
5. qDebug()<<"点击删除";
6. if(ui->treeView->currentIndex().parent().data().toString()=="TCP服务器")
7. {
8. qDebug()<<"点击删除TCP服务器";
9. tcpServerManagementList.at(ui->treeView->currentIndex().row())->CloseAndDelListen();
10. delete tcpServerManagementList.at(ui->treeView->currentIndex().row());
11. tcpServerManagementList.removeAt(ui->treeView->currentIndex().row());
12. m\_Tree->removeRow(ui->treeView->currentIndex().row(),ui->treeView->currentIndex().parent());
13. DeviceNum--;
14. //\*最后在移除item，如果先移除那么当前所选中的就变了
15. }
16. else if (ui->treeView->currentIndex().parent().data().toString()=="TCP客户端") {
17. //在删除客户端时，要把连接服务器的客户端从conbobox中删掉,要把服务器中保存客户端信息的list中的数据删掉
18. qDebug()<<"点击删除TCP客户端";
19. ui->comboBox\_TCPclients->removeItem(ui->treeView->currentIndex().row()+1);
20. tcpClientList.at(ui->treeView->currentIndex().row())->disConnectToHost();
21. delete tcpClientList.at(ui->treeView->currentIndex().row());
22. tcpClientList.removeAt(ui->treeView->currentIndex().row());
23. m\_Tree->removeRow(ui->treeView->currentIndex().row(),ui->treeView->currentIndex().parent());
24. DeviceNum--;
25. }
26. else if (ui->treeView->currentIndex().parent().data().toString()=="UDP服务端") {
27. qDebug()<<"点击删除UDP服务器";
28. delete udpServerList.at(ui->treeView->currentIndex().row());
29. udpServerList.removeAt(ui->treeView->currentIndex().row());
30. m\_Tree->removeRow(ui->treeView->currentIndex().row(),ui->treeView->currentIndex().parent());
31. DeviceNum--;
32. }
33. else if (ui->treeView->currentIndex().parent().data().toString()=="UDP客户端") {
34. qDebug()<<"点击删除UDP客户端";
35. delete udpClientList.at(ui->treeView->currentIndex().row());
36. udpClientList.removeAt(ui->treeView->currentIndex().row());
37. m\_Tree->removeRow(ui->treeView->currentIndex().row(),ui->treeView->currentIndex().parent());
38. DeviceNum--;
39. }
40. QString str=QString::number(DeviceNum)+"/"+QString::number(MaxDeviceNum);
41. ui->label\_DeviceNum->setText(str);
42. }

## 4.3 设备信息模块

通过选择相应的设备在设备信息框中显示或设置相应设备信息。TCP服务器点击开始监听按钮后按钮变为停止监听，TCP服务器开始监听。TCP服务器设备信息框如图4-4所示。



图4-4 TCP服务器设备信息框

TCP客户端点击连接按钮后按钮变为断开连接，TCP客户端连接TCP服务器。TCP客户端设备信息框如图4-5所示。



图4-5 TCP客户端设备信息框

UDP服务端在有UDP客户端发来消息前不能发送消息。UDP服务端设备信息框如图4-6所示。



图4-6 UDP服务器设备信息框

UDP客户端设备信息框如图4-7所示。



图4-7 UDP客户端设备信息框

串口设备在设置串口信息后点击打开串口后，，串口打开，按钮变为关闭串口。串口设备信息框如图4-8所示。



图4-8 串口设备信息框

## 4.4 发送数据模块

选择一个已就绪设备后在输入框中输入消息，选择发送模式为单次发送还是定时发送。发送数据图如图4-9所示。



图4-9 发送数据图

### 4.4.1 单次发送

部分代码如下所示。

1. int currIndex=ui->treeView->currentIndex().row();
2. QString parentName=ui->treeView->currentIndex().parent().data().toString();
3. //通过判断当前所选中item的parent来更改相应的标签
4. if(parentName=="TCP服务器")
5. {
6. if(tcpServerManagementList.at(currIndex)->isListening())
7. {
8. qDebug()<<"服务器点击发送";
9. if(ui->radioButton->isChecked())
10. {
11. QString str=tcpServerManagementList.at(currIndex)->server->textEdit->toPlainText();
12. tcpServerManagementList.at(currIndex)->server->
13. sendMsg(str,ui->comboBox\_TCPclients->currentText());
14. }
15. else {
16. QString str=tcpServerManagementList.at(currIndex)->server->hexEdit->toPlainText();
17. tcpServerManagementList.at(currIndex)->server->
18. sendHexMsg(str,ui->comboBox\_TCPclients->currentText());
19. }
20. }
21. }

### 4.4.2 定时发送

部分代码如图所示。

1. if(sendType==10)
2. {
3. if(nowCount<timerCount)
4. {
5. nowCount++;
6. sendMsg(msg,addr);
7. }
8. else {
9. isTimeSending=false;
10. emit sendDateSIGNAL(sendBit,ricvBit,falgcount,type);
11. m\_timer->stop();
12. nowCount=0;
13. delete m\_timer;
14. m\_timer=nullptr;
15. return;
16. }
17. }
18. else if(sendType==16){
19. if(nowCount<timerCount)
20. {
21. nowCount++;
22. sendHexMsg(msg,addr);
23. }
24. else {
25. isTimeSending=false;
26. emit sendDateSIGNAL(sendBit,ricvBit,falgcount,type);
27. m\_timer->stop();
28. nowCount=0;
29. delete m\_timer;
30. m\_timer=nullptr;
31. return;
32. }
33. }

### 4.4.3 设置发送模式

设置发送模式图如图4-10所示。



图4-10 设置发送模式图

部分代码如下所示。

1. //只能输入16进制数和回退符
2. void common::keyPressEvent(QKeyEvent \*event)
3. {
4. if(event->key()==Qt::Key\_0||event->key()==Qt::Key\_1||event->key()==Qt::Key\_2||event->key()==Qt::Key\_3||event->key()==Qt::Key\_4||event->key()==Qt::Key\_5||event->key()==Qt::Key\_6||event->key()==Qt::Key\_7||event->key()==Qt::Key\_8||event->key()==Qt::Key\_9||event->key()==Qt::Key\_A||event->key()==Qt::Key\_B||event->key()==Qt::Key\_C||event->key()==Qt::Key\_D||event->key()==Qt::Key\_E||event->key()==Qt::Key\_F)
5. {
6. QString str=this->toPlainText();
7. str.append(event->text().toUpper());
8. this->clear();
9. this->append(str);
10. }
11. else if(event->key()==Qt::Key\_Backspace) {
12. QString str=this->toPlainText();
13. QString str1=str.remove(str.length()-1,1);
14. this->clear();
15. this->append(str1);
16. }
17. }

## 4.5 数据接收模块

在数据接收框中显示对方和自己发送的消息。数据接收框如图4-11所示。

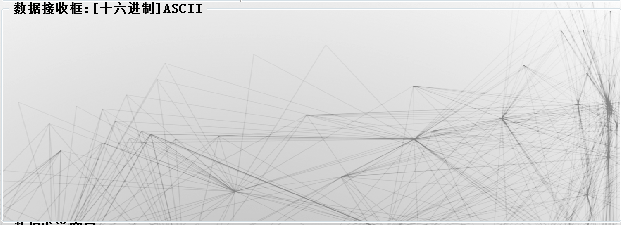


图4-11 数据接收框

## 4.6 系统设置模块

在软件底部显示当前的设备数与最大设备数。点击更改最大连接数后弹出更改界面设置最大连接数。创建一个全局变量DeviceNum来表示当前设备数。软件初始化设置最大连接数为10。当设备数超过10便不可创建设备。更改最大连接数界面图如图4-12所示。

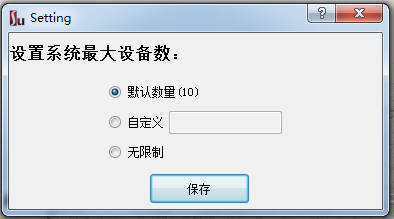


图4-12 更改最大连接数界面图

部分代码如下所示。

1. void SettingDialog::saveSLOT()
2. {
3. if(ui->radioButton\_renyi->isChecked())
4. {
5. num=ui->lineEdit->text().toInt();
6. }
7. qDebug()<<"设置num="<<num;
8. if(num>=DeviceNum)
9. {
10. emit sendDeviceNumSIGNAL(num);
11. }
12. else {
13. QMessageBox::warning(this,"ERROR","设置最大设备数失败（设置数小于当前设备数，请删除多余设备或重新设置）",nullptr,nullptr);
14. }
15. this->deleteLater();
16. }

## 4.7 其他功能模块

清空数据接收框、清空数据发送框、加载文件、导出数据、功能详解、技术清零的按钮设计图如图4-13所示。

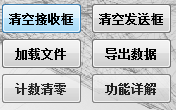


图4-13 按钮设计图

字节数与设备数显示图如图4-14所示。



图4-14 字节数与设备数显示图

### 4.7.1 清空数据接收框

点击清空数据接收框后，清空接收框数据。

部分代码如下所示。

1. void MyObject::clearData()
2. {
3. textBrowser->clear();
4. }

### 4.7.2 清空数据发送框

点击清空数据发送框后，根据所选发送模式清空当前发送框数据。

部分代码如下所示。

1. void MyObject::clearSend(int type)
2. {
3. if(type==10)
4. {
5. textEdit->clear();
6. }
7. else if (type==16) {
8. hexEdit->clear();
9. }
10. }

### 4.7.3 加载文件

点击加载文件后选择数据文件，设置设备当前状态位正在加载文件，设置当前输入框不可更改。

部分代码如下所示。

1. void MyObject::loadFile(QString path,int type)
2. {
3. isFile=true;
4. qDebug()<<type<<path;
5. if(type==10)
6. {
7. textEdit->clear();
8. textEdit->append("外部数据源文件["+path+"]");
9. textEdit->setEnabled(false);
10. }
11. else if(type==16)
12. {
13. hexEdit->isFile=true;
14. hexEdit->clear();
15. hexEdit->append("外部数据源文件["+path+"]");
16. hexEdit->setEnabled(false);
17. }
18. }

### 4.7.4 导出数据

点击导出数据后，设置导出路径与文件名，把数据接收框中数据导入文件。

部分代码如下所示。

1. void MyObject::saveData()
2. {
3. QString filePath = QFileDialog::getSaveFileName();
4. QFile file(filePath);
5. if(!file.open(QIODevice::WriteOnly))
6. {
7. qDebug()<<"open file ERROR";
8. return;
9. }
10. QString str=textBrowser->toPlainText();
11. qDebug()<<str;
12. QStringList strList=str.split("\n");
13. foreach(QString str,strList)
14. {
15. str.append("\r\n");
16. file.write(str.toUtf8());
17. }
18. file.close();
19. QMessageBox::warning(nullptr,"消息","保存数据成功！（"+filePath+")",nullptr,nullptr);
20. }

### 4.7.5 显示字节数与重置计数

选择一个已就绪设备在软件底部显示当前字节数。点击计数清零后重置当前设备的字节数。

部分代码如下所示。

1. if(ui->treeView->currentIndex().parent().data().toString()=="TCP服务器")
2. {
3. qDebug()<<"tcp服务器更新字节和定时按钮";
4. if(tcpServerManagementList.at(ui->treeView->currentIndex().row())->getfalgcount()==falgcount)
5. {
6. ui->label\_sendNum->setText(QString::number(send)+" 字节");
7. ui->label\_ricvNum->setText(QString::number(ricv)+" 字节");
8. qDebug()<<"upDateBit"<<send<<ricv;
9. if(tcpServerManagementList.at(ui->treeView->currentIndex().row())->server->isTimerSending()==true)
10. ui->toolButton\_timerStartstop->setText("停止发送");
11. else
12. ui->toolButton\_timerStartstop->setText("定时发送");
13. }
14. }

### 4.7.6 功能详解

点击功能详解按钮弹出对话框。功能详解界面图如图4-15所示。

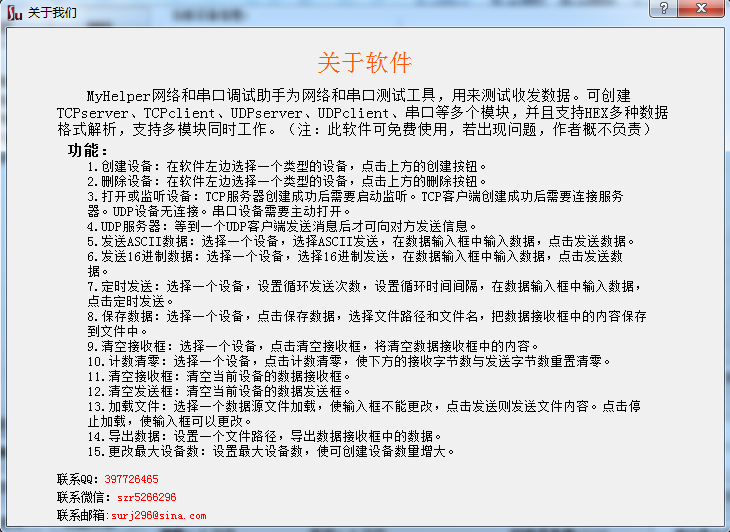


图4-15 功能详解界面图

## 4.8 本章小结

本章主要是对于本次毕业设计开发阶段的概述，介绍了本软件在设计时的思路与各个功能的实现方法，将这些模块与功能的界面实现图展现在文章中。本软件在此便已经基本开发完成，剩下的工作就是对软件的测试。

# 第5章 软件测试

软件测试是软件质量保证的关键步骤。软件测试保障了软件的正确性、完整性、安全性和软件的质量。

## 5.1 测试原则

软件测试就是为了保证软件质量，如果没有很好地完成软件测试任务，产品

的质量得不到保证。如果没有软件测试，就不能了解软件产品的质量。测试是软

件工程中不可缺少的一部分，特别是当软件无处不在、越来越贴近人们的生活和工作的时候，软件测试的必要性就越来越明显[[10]](#endnote-11)[10]。

## 5.2 测试环境

一台拥有串口或者拥有虚拟串口的windows操作系统的电脑。

## 5.3 设备模块测试

本小节是对网络与串口调试助手的设备模块进行测试。

### 5.3.1 创建设备

打开软件后能否正常创建设备。创建设备测试用例表见表5-1。

表5-1 创建设备测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 创建设备 | 创建TCP服务器成功 | 选择TCP服务器点击创建，输入123123 | 关闭创建窗口，在设备区添加设备，显示目前设备数为：1/10 | 正常 |
| 创建设备 | 创建TCP客户端成功 | 选择TCP客户端点击创建，输入192.168.18.26,123123 | 关闭创建窗口，在设备区添加设备，显示目前设备数为：1/10 | 正常 |
| 创建设备 | 创建UDP服务器成功 | 选择UDP服务端点击创建，输入456456 | 关闭创建窗口，在设备区添加设备，显示目前设备数为：1/10 | 正常 |
| 创建设备 | 创建UDP客户端成功 | 选择UDP客户端点击创建，输入192.168.18.26,456456 | 关闭创建窗口，在设备区添加设备，显示目前设备数为：1/10 | 正常 |
| 创建设备 | 检测串口成功 | 在打开软件前创建虚拟串口COM1和COM2，打开软件 | 在设备区添加串口设备，显示目前设备数为：2/10 | 正常 |
| 创建设备 | 检测串口失败 | 打开本软件，创建虚拟串口COM1和COM2 | 软件无反应 | 正常 |

续表5-1 创建设备测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 创建设备 | 创建TCP服务器成功 | 选择TCP服务器点击创建，输入asd | 不能关闭创建窗口，不能添加设备 | 正常 |
| 创建设备 | 创建TCP服务器失败 | 选择TCP服务器点击创建，输入-123 | 不能关闭创建窗口，不能添加设备 | 正常 |

### 5.3.2 删除设备

对删除设备功能进行测试，看能否删除所创建的设备。删除设备测试用例表见表5-2。

表5-2 删除设备测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 删除设备 | 删除TCP服务器成功 | 选择TCP服务器点击删除 | 在设备区删除该设备 | 正常 |
| 删除设备 | 删除TCP客户端成功 | 选择TCP客户端点击删除 | 在设备区删除该设备 | 正常 |
| 删除设备 | 删除UDP服务器成功 | 选择UDP服务器点击删除 | 在设备区删除该设备 | 正常 |
| 删除设备 | 删除UDP客户端成功 | 选择UDP客户端点击删除 | 在设备区删除该设备 | 正常 |
| 删除设备 | 删除串口设备失败 | 选择串口点击删除 | 不能删除 | 正常 |

## 5.4 设备信息模块测试

本小节是对网络与串口调试助手的设备信息模块进行测试。显示设备信息测试用例表见表5-3。

表5-3 显示设备信息测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |

续表5-3 显示设备信息测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 显示设备信息 | 显示TCP服务器信息成功 | 选择一个TCP服务器 | 在设备区显示该设备信息 | 正常 |
| 显示设备信息 | 显示TCP客户端信息成功 | 选择一个TCP客户端 | 在设备区显示该设备信息 | 正常 |
| 显示设备信息 | 显示UDP服务器信息成功 | 选择一个UDP服务器 | 在设备区显示该设备信息 | 正常 |
| 显示设备信息 | 显示UDP客户端信息成功 | 选择一个UDP客户端 | 在设备区显示该设备信息 | 正常 |
| 显示设备信息 | 显示串口设备信息成功 | 选择一个串口设备 | 在设备区显示该设备信息 | 正常 |

## 5.5 发送数据模块测试

本小节是对网络与串口调试助手的发送数据模块进行测试。

### 5.5.1 单次发送功能测试

单次发送测试用例表见表5-4。

表5-4 单次发送测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 单次发送 | 设备单次发送成功 | 选择已就绪TCP服务器，在数据输入框输入123，点击发送数据 | 在数据接收框显示发送数据 | 正常 |
| 单次发送 | 设备单次发送成功 | 选择已就绪TCP客户端，在数据输入框输入123，点击发送数据 | 在数据接收框显示发送数据 | 正常 |
| 单次发送 | 设备单次发送成功 | 选择已就绪UDP服务器，在数据输入框输入123，点击发送数据 | 在数据接收框显示发送数据 | 正常 |
| 单次发送 | 设备单次发送成功 | 选择已就绪UDP客户端，在数据输入框输入123，点击发送数据 | 在数据接收框显示发送数据 | 正常 |
| 单次发送 | 设备单次发送成功 | 选择已就绪串口设备，在数据输入框输入123，点击发送数据 | 在数据接收框显示发送数据 | 正常 |
| 定时发送 | 设备单次发送失败 | 选择未就绪设备，点击发送数据 | 软件无反应 | 正常 |

### 5.6.2 定时发送功能测试

定时发送测试用例表见表5-5。

表5-5 定时发送测试用例表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 功能 | | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 定时发送 | 设备定时发送成功 | | 选择已就绪TCP服务器，在数据输入框输入123，设置发送次数5次，设备间隔1秒 | 在数据接收框每隔1秒显示发送数据五次 | 正常 |
| 定时发送 | 设备定时发送成功 | | 选择已就绪TCP客户端，在数据输入框输入123，设置发送次数5次，设备间隔1秒 | 在数据接收框每隔1秒显示发送数据五次 | 正常 |
| 定时发送 | 设备定时发送成功 | | 选择已就绪UDP服务器，在数据输入框输入123，设置发送次数5次，设备间隔1秒 | 在数据接收框每隔1秒显示发送数据五次 | 正常 |
| 定时发送 | 设备定时发送成功 | | 选择已就绪UDP客户端，在数据输入框输入123，设置发送次数5次，设备间隔1秒 | 在数据接收框每隔1秒显示发送数据五次 | 正常 |
| 定时发送 | 设备定时发送成功 | | 选择已就绪串口设备，在数据输入框输入123，设置发送次数5次，设备间隔1秒 | 在数据接收框每隔1秒显示发送数据五次 | 正常 |
| 定时发送 | 设备定时发送失败 | | 选择未就绪设备，点击定时发送 | 软件无反应 | 正常 |

### 5.6.3 设置发送模式功能测试

设置发送模式测试用例表见表5-6。

表5-6 设置发送模式测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 设置发送模式 | 设置ASCII模式成功 | 选择已就绪的TCP服务器，设置字符模式 | 在数据输入框能输入所有字符 | 正常 |
| 设置发送模式 | 设置ASCII模式成功 | 选择已就绪的TCP客户端，设置字符模式 | 在数据输入框能输入所有字符 | 正常 |
| 设置发送模式 | 设置ASCII模式成功 | 选择已就绪的UDP服务器，设置字符模式 | 在数据输入框能输入所有字符 | 正常 |
| 设置发送模式 | 设置ASCII模式成功 | 选择已就绪的UDP客户端，设置字符模式 | 在数据输入框能输入所有字符 | 正常 |
| 设置发送模式 | 设置ASCII模式成功 | 选择已就绪的串口设备，设置字符模式 | 在数据输入框能输入所有字符 | 正常 |
| 设置发送模式 | 设置Hex模式成功 | 选择已就绪的TCP服务器，设置十六进制模式 | 在数据输入框只能输入十六进制字符 | 正常 |
| 设置发送模式 | 设置Hex模式成功 | 选择已就绪的TCP客户端，设置十六进制模式 | 在数据输入框只能输入十六进制字符 | 正常 |
| 设置发送模式 | 设置Hex模式成功 | 选择已就绪的UDP服务器，设置十六进制模式 | 在数据输入框只能输入十六进制字符 | 正常 |
| 设置发送模式 | 设置Hex模式成功 | 选择已就绪的UDP客户端，设置十六进制模式 | 在数据输入框只能输入十六进制字符 | 正常 |
| 设置发送模式 | 设置Hex模式成功 | 选择已就绪的串口设备，设置十六进制模式 | 在数据输入框只能输入十六进制字符 | 正常 |

## 5.6 数据接收模块测试

本小节是对网络与串口调试助手的数据接收模块进行测试。数据接收功能测试用例表见表5-7。

表5-7 数据接收功能测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |

续表5-7 数据接收功能测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 数据接收 | TCP服务器能接收到数据成功 | 在TCP服务器所连接的设备中发送数据123 | 接收数据[313233]123 | 正常 |
| 数据接收 | TCP客户端能接收到数据成功 | 在TCP客户端所连接的服务器中发送数据456 | 接收数据[343536]456 | 正常 |
| 数据接收 | UDP服务端能接收到数据成功 | 在UDP服务端所连接的UDP客户端中发送123 | 接收数据[313233]123 | 正常 |
| 数据接收 | UDP客户端能接收到数据成功 | 在UDP客户端所连接的UDP服务端中发送456 | 接收数据[343536]456 | 正常 |
| 数据接收 | 串口设备能接收到数据成功 | 在串口设备的另一端设备发送123 | 接收数据[313233]123 | 正常 |

## 5.7 系统设置模块

本小节是对网络与串口调试助手的系统设置模块进行测试。更改最大连接数测试用例表见表5-8。

表5-8 更改最大连接数测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 更改最大连接数 | 更改设备最大连接数成功 | 当前设备数为0，点击更改最大连接数，选择自定义，输入5 | 显示目前设备数为：0/5 | 正常 |
| 更改最大连接数 | 更改设备最大连接数成功 | 当前设备数为0，点击更改最大连接数，选择无限制 | 显示目前设备数为：0/9999999 | 正常 |
| 更改最大连接数 | 更改设备最大连接数成功 | 当前设备数为0，点击更改最大连接数，选择默认数量 | 显示目前设备数为：0/10 | 正常 |

续表5-8 更改最大连接数测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 更改最大连接数 | 更改设备最大连接数失败 | 当前设备数为6，点击更改最大连接数，选择自定义，输入5 | 弹出错误框，提示设置最大设备数失败，显示目前设备数为：6/10 | 正常 |

## 5.8 其他功能模块测试

本小节是对网络与串口调试助手的其他功能模块进行测试。

### 5.8.1 清空数据接收框功能测试

清空数据接收框测试用例表见表5-9。

表5-9 清空数据接收框测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 清空数据接收框 | 清空已就绪TCP服务器数据接收框成功 | 在该TCP服务器所连接的TCP客户端中发送123后，点击服务器的清空数据接收框 | 数据接收框清空 | 正常 |
| 清空数据接收框 | 清空已就绪TCP客户端数据接收框成功 | 在该TCP客户端所连接的TCP服务器中发送123后，点击客户端的清空数据接收框 | 数据接收框清空 | 正常 |
| 清空数据接收框 | 清空已就绪UDP服务器数据接收框成功 | 在该UDP服务器所连接的UDP客户端中发送123后，点击服务器的清空数据接收框 | 数据接收框清空 | 正常 |
| 清空数据接收框 | 清空已就绪UDP客户端数据接收框成功 | 在该UDP客户端所连接的UDP服务器中发送123后，点击客户端的清空数据接收框 | 数据接收框清空 | 正常 |

续表5-9 清空数据接收框测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 清空数据接收框 | 清空已就绪串口设备数据接收框成功 | 在串口设备的另一端设备发送123后，点击清空数据接收框 | 数据接收框清空 | 正常 |
| 清空数据接收框 | 清空未就绪设备数据接收框成功 | 点击清空数据接收框 | 软件无反应 | 正常 |

### 5.8.2 清空数据发送框功能测试

清空数据发送框测试用例表见表5-10。

表5-10 清空数据发送框测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 清空数据发送框 | 清空已就绪设备的数据发送框成功 | 选择字符模式，在已就绪设备的数据发送框中输入123，点击清空发送框 | 数据发送框清空 | 正常 |
| 清空数据发送框 | 清空已就绪设备的数据发送框成功 | 选择十六进制模式，在已就绪设备的数据发送框中输入123，点击清空发送框 | 数据发送框清空 | 正常 |
| 清空数据发送框 | 清空未就绪设备数据发送框成功 | 点击清空发送框 | 软件无反应 | 正常 |

### 5.8.3 加载文件功能测试

加载文件测试用例表见表5-11。

表5-11 加载文件测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 加载文件并发送 | 已就绪设备加载文件并发送成功 | 选择字符模式，点击加载文件选择文件asd.txt，点击发送 | 数据发送框，接收框显示文件路径，发送框不可更改 | 正常 |

续表5-11 加载文件测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 加载文件并发送 | 已就绪设备加载文件并发送成功 | 选择十六进制模式，点击加载文件选择文件asd.txt，点击发送 | 数据发送框，接收框显示文件路径，发送框不可更改 | 正常 |
| 加载文件并发送 | 未就绪设备加载文件并发送失败 | 点击加载文件 | 软件无反应 | 正常 |

### 5.8.4 导出数据功能测试

导出数据测试用例表见表5-12。

表5-12 导出数据测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 导出数据 | 已就绪设备导出数据成功 | 在数据接收框有数据的设备中点击导出数据，设置文件路径为D://1.txt | 弹出保存数据成功对话框，D盘出现文件1.txt并且文件内容为该设备数据接收框中内容。 | 正常 |
| 导出数据 | 已就绪设备导出数据成功 | 在数据接收框无数据的设备中点击导出数据，设置文件路径为D://2.txt | 弹出保存数据成功对话框，D盘出现文件2.txt并且文件内容为空 | 正常 |
| 导出数据 | 未就绪设备导出数据失败 | 点击导出数据 | 软件无反应 | 正常 |

### 5.8.5 显示字节数功能测试

显示字节数测试用例表见表5-13。

表5-13 显示字节数测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |

续表5-13 显示字节数测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 显示字节数 | 已就绪设备显示字节数成功 | 已就绪设备发送数据123 | 显示发送字节数为3，接收字节数为0 | 正常 |
| 显示字节数 | 已就绪设备显示字节数成功 | 其他设备向已就绪设备发送数据123 | 显示接收字节数为3，发送字节数为0 | 正常 |
| 显示字节数 | 未就绪设备显示字节数成功 | 无 | 显示发送字节数为0，接收字节数为0 | 正常 |

### 5.8.6 计数清零功能测试

计数清零测试用例表见表5-14。

表5-14 计数清零测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 计数清零 | 已就绪设备计数清零成功 | 选择发送字节数和接收字节数不为0的已就绪设备，点击计数清零 | 显示发送字节数为0，接收字节数为0 | 正常 |
| 计数清零 | 已就绪设备计数清零成功 | 选择发送字节数和接收字节数为0的已就绪设备，点击计数清零 | 显示接收字节数为0，发送字节数为0 | 正常 |
| 计数清零 | 未就绪设备计数清零成功 | 无 | 显示发送字节数为0，接收字节数为0 | 正常 |

### 5.8.7 显示设备数功能测试

显示设备数测试用例表见表5-15所示。

表5-15 显示设备数测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |

续表5-15 显示设备数测试用例表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 功能 | 目标 | 测试输入 | 相对预期效果 | 实际效果 |
| 显示设备数 | 显示当前设备数和最大设备数成功 | 创建一个设备 | 显示目前设备数为：1/10 | 正常 |
| 显示设备数 | 显示当前设备数和最大设备数成功 | 当前有一个设备，删除该设备 | 显示目前设备数为：0/10 | 正常 |
| 显示设备数 | 显示当前设备数和最大设备数失败 | 在当最大设备数为10并且当前设备数为10时，创建一个设备 | 弹出错误框，提示超过最大设备数，创建失败 | 正常 |

## 5.9 本章小结

本章主要是对于本次毕业设计中的软件测试进行概述，介绍了本软件在进行功能测试时各个功能的测试结果，并将这些测试结果以表的格式展现在文章中。本次毕业设计基本完成。

# 第6章 总结与展望

## 6.1 总结

在本次毕业设计的应用开发中，使用了一个新的编译器Qt Creater,与带有Qt风格的C++语言。通过这次对网络与串口调试助手应用的设计与开发，不仅对Qt Creator有了全新的认识，更对C++有了更深度的了解。在查阅书籍与上网查阅资料时更是自学了Qt中的GUI设计、socket编程、多线程与串口通信。不但巩固了自己在校期间所学到的专业知识，更是拓展了自己的知识面，提高了自己自学的能力与独立开发的能力，这在毕业后的工作中是非常重要的。

本软件包括了TCP设备连接、UDP设备连接、串口设备连接、各种设备的数据发送与接收、加载文件、导出数据等主要功能，也实现了提升用户体验的一些实用功能，比如说显示接收与发送的字节数，显示当前设备数等功能。但是也有许多的不足之处，如串口设备不能在打开软件后检测到，串口设备不能检测到设备连接断开等。

## 6.2 展望

本小结讲述了本软件的缺陷与应改进的部分。

（1）软件的界面设计与样式不够新颖，用的是Qt默认的界面样式，需要在以后对软件的整体样式进行修改。

（2）上文总结时说的串口设备不能在打开软件后随时检测到新的串口设备，只可以检测到打开软件前所连接的串口。希望以后在提高自身的专业知识后改进。

（3）上文总结时说的串口设备未能实现检测断开连接的功能。希望以后在提高自身的专业知识后改进。

（4）Hex模式发送框中，只做到了基本的输入与单字符删除，不能进行常用的鼠标点击位置插入与选择性删除，需要在以后改进该功能。

# 参考文献

# 致 谢

本毕业设计（论文）的工作是在徐林雪老师的悉心指导下完成的，徐林雪老师严谨的治学态度和科学的工作方法给了我极大的帮助和影响。在此衷心感谢四年来徐林雪老师对我的关心和指导。

徐林雪老师悉心指导我们完成了毕业设计（论文），在学习上和生活上都给予了我很大的关心和帮助，在此向徐林雪老师表示衷心的谢意。

徐林雪老师对于我的毕业设计（论文）都提出了许多的宝贵意见，在此表示衷心的感谢。

在撰写毕业设计（论文）期间，武志强、齐睿、陈可强等同学对我毕业设计（论文）中的软件测试研究工作给予了热情帮助，在此向他们表达我的感激之情。

另外也感谢家人，他们的理解和支持使我能够在学校专心完成我的学业。

1. [1] 陶建华,刘瑞挺,徐恪,韩伟力,张华平,于剑,田丰,梁晓辉. 中国计算机发展简史[J]. 科技导报,2016,34(14):12-21.

   [2]张葛. 计算机应用发展现状及发展趋势[J]. 山东工业技术,2019(05):157.

   [3] 傅篱. 嵌入式系统在我国的应用现状与发展趋势[J]. 管理观察,2013(31):164-166.

   [4]余久久.软件工程简明教程[M].清华大学出版社:北京,2015.05.

   [5]李冬梅. 基于Qt的组态软件的设计与开发[D].大连理工大学,2012. [↑](#endnote-ref-2)
2. [6]王浩南,刘益成. 基于嵌入式Linux系统下的Qt开发[J]. 电脑开发与应用,2010,23(01):11-13.

   [7]束长宝,于照,张继勇. 基于TCP/IP的网络通信及其应用[J]. 微计算机信息,2006(36):157-159.

   [8]欧军,吴清秀,裴云,张洪. 基于socket的网络通信技术研究[J]. 网络安全技术与应用,2011(07):19-21.

   [9]刘先博. 基于FPGA与MCU的多串口通信接口设计与实现[D].南京理工大学,2014.

   [10] 朱少民．软件测试方法和技术[M] ．北京：清华大学出版社，2014． [↑](#endnote-ref-3)
3. [↑](#endnote-ref-4)
4. [↑](#endnote-ref-5)
5. [↑](#endnote-ref-6)
6. [↑](#endnote-ref-7)
7. [↑](#endnote-ref-8)
8. [↑](#endnote-ref-9)
9. [↑](#endnote-ref-10)
10. [↑](#endnote-ref-11)