湖南理工职业技术学院

毕业设计说明书

（产品设计说明书√、工艺设计说明书□、方案设计说明书□）

题　　目：	基于Atmel328P的开源PLC的设计
年级专业：	2015级 电气自动化
学生姓名：	黎武鑫
指导教师：	陈揆能
企业教师：	钱仪

2018 年 5 月 8 日

摘　要

可编程控制器（PLC）是一种工业控制设备，其编程方式简单，硬件电路和软件系统可靠性高，在自动化生产中有着极其重要作用，随着社会和科技快速的发展，生产水平的逐步改善，生产上也需要达到相应的生产水平，生产线也逐步往自动化方向发展，这个发展过程中 PLC 成为了关键。同时可编程控制器但是由于大厂垄断，价格昂贵，没有完全开源的 PLC 硬件和软件供选择使用。

本设计将设计一套开源 PLC 硬件和 PLC 编程软件，PLC 硬件以 Ateml328P 为控制核心，具有 8 点高速数字 I/O 输入，8 点数字 I/O 输出，两通道 4096 分之一精度的模拟 I/O 输入，一通道 RS232 通讯串口，自带 24 伏电源输出，具备 PLC 硬件的基本硬件。PLC 编程软件用 Qt 框架对 ArduinoIDE 进行二次开发，程序都已项目的方式存在，编程软件(IDE)具有梯形图和高级编程语言两种方式，可以实现编程软件对 PLC 的编程和程序下载。

关键字 可编程控制器（PLC）单片机 梯形图

目 录

第一章 绪论

1.1 课题研究背景及意义

目前，国内工业制造业自动化智能化升级，对自动化设备需求量大，考虑到设备的的高可靠性，国内自动化厂商一般选择进口 PLC，但因为其价格昂贵，导致自动化厂家面临成本压力，不利于长期发展。虽然也有一些国内 PLC 厂家，但主要以模仿进口为主，品质和性能上难以让厂家普遍接受[3]。而且 PLC 生产行业属于大厂垄断，实际成本并不高，其中的利润极大。

在编程方式上也需要变革，早期 PLC 主要是简单的逻辑控制，而现在各个行业自动化程度智能化程度都在提高。早期的编程方式已经不能满足现有需求。原先的梯形图语言（LD）编程方式为主，结构化编程语言（ST）使用为辅，且 ST 使用不便；现在顺应发展，PLC 的结构化编程语言也开始变成主要编程方式之一了，在复杂的程序编写时效率相对梯形图编程语言（LD）高很多。

在这种背景下，支持高级语言编程的开源共享的 PLC，是非常有益于我国现代工业生产的自动化智能化发展的。

1.2 研究现状

国外有不少开源可编程逻辑控制器项目，开源 PLC 最为人知的就是 openPLC 项目。OpenPLC 标准且优秀，缺点是不支持中文，不适合国人使用，这样对英文水平有限的技术人员是一道难跨的门槛。

我国的 PLC 研发水平与工业发达国家相比有滞后。而且开源 PLC 属于冷门，更多都是在仿造抄袭着国外技术透明的 PLC。而自动化控制的 DIY 市场迅速发展，比如一些微型的 CNC 设备，价格足够低，让很多 DIY 爱好者可以自己动手设计一些设备，但他们缺少资金，哪怕是国产的 PLC，对他们来说，价格都太贵，他们更喜欢选择嵌入式作为控制系统，但因为嵌入式涉及面太广，太随意，且各自为政，没有一个可以持续的发展标准和长期验证、改善和演化的基础，所以到今天，只能在自己设计的项目中使用，限制了规模[4]。并没有出现开源出来的 PLC 项目。

总的来说，PLC 被广泛应用于生产、控制等领域，数量日渐上升。近几十年的发展大致经历了两个阶段：1.具备简单逻辑控制和通讯；3.具备复杂的逻辑控制支持网络化通讯。目前，国际上新型 PLC 正向集成化向智能化、网络化的方向上发展[10]。虽然 PLC 这个方面已有很久的发展，但是开源软硬件整套的 PLC 项目还是非常少见的，虽然国外已有一定尝试，但是对于这个庞大目标用户的领域，一两个开源 PLC 不能所有人的需求，而且这是一个需要大量探索者进入的领域，并且国外的开源 PLC 不一定适于国内的使用需求。需要一个对中文支持友好的 PLC 编程系统软件。

1.3 课题研究的内容

研究设计一款基于 Atmel328P 的开源可编程控制器（PLC），具备 16 点光耦隔离输入输出 IO，两通道 12 位精度模拟 IO 输入，一通道 RS232 通讯串口。符合 PLC 硬件要求。

软件方面，设计可以配合该 PLC 的一款编程软件，编程软件可使用梯形图和结构化文本进行编程，且可以通过 USB 延长线连接 PLC，通过软件直接下载编写的程序。

第二章 控制系统硬件设计

基于上章的分析，决定硬件系统方案的设计如图 2-1 所示。本章主要介绍系统的硬件设计，主要从电路硬件设计和外形 3D 设计进行阐述。

2.1 硬件系统设计方案

本设计 PLC 使用 Atmel328p 做核心处理器，外设有数字量采集模块、模拟量采集模块、USB 通讯模块、RS232 通讯模块、数字输出模块。

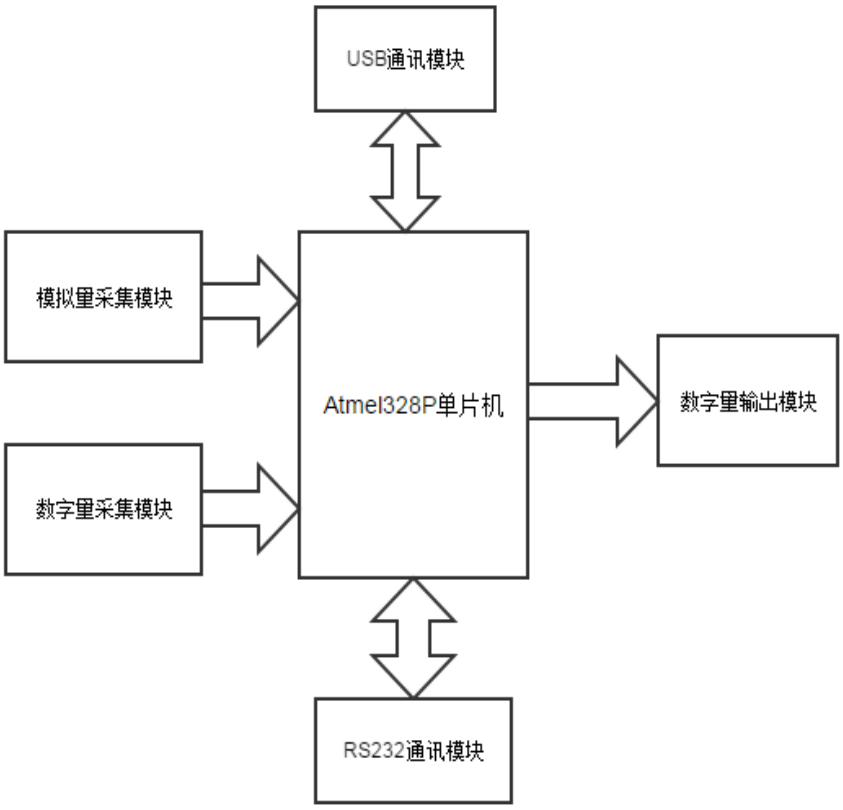

图 2-1 硬件系统框图

系统运行频率 8MHZ；PLC 输入 IO 8 个，输出 IO 8 个采用继电器输出，PLC 里面集成 5V 和 24V 电源，电源模块采用的是 SHB-5W-05V，选用这样需要 24v 电源就不需要外部电源设备供电了，电源直接输入 220v 交流就可以，设置两通道的 0-24v 模拟电压检测，当中的运放芯片采用的是 MAX4424；通讯采用 232 通讯，使用 MAX232 芯片进行 TTL 转 RS232；下载采用 USB 转接线和电脑相连接进行下载，然后 PLC 内部使用 GH340 芯片把 USB 转成 TTL 串口；PLC 外壳用 3D 打印生成。

2.2 外形设计建模

外壳可以方便进行组装和实际时使用，对内部电路起到了保护作用，也保证了使用过程中的人生安全。为了系统的实用性和安全性，于是进行外壳设计。

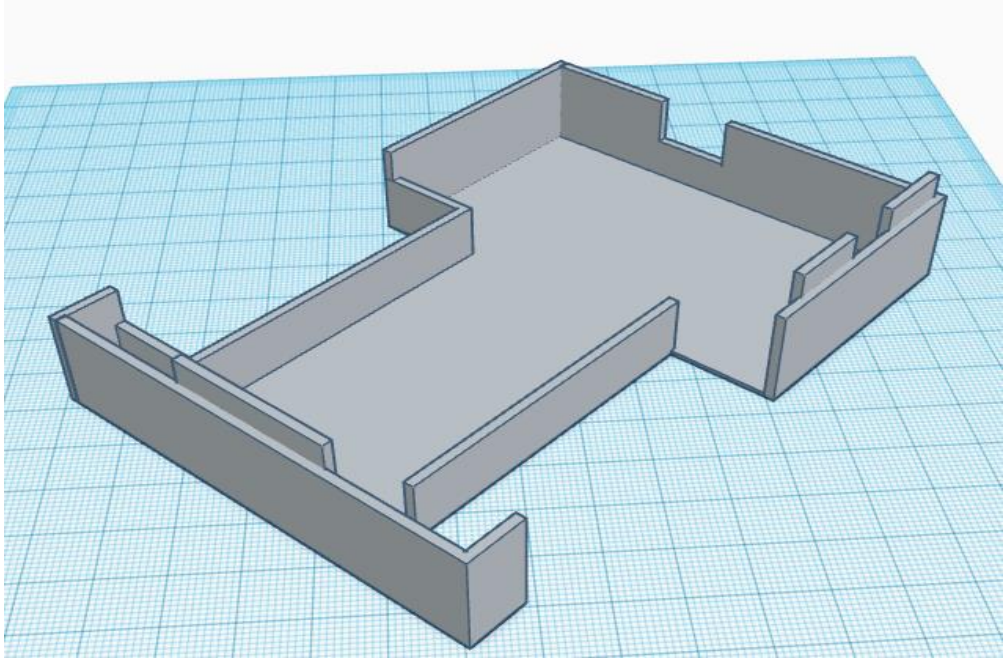

图 2-2　PLC 外壳 3D 视图

本设计的外壳由 3DMAX 软件设计，最终导出成 STL 格式 3d 模型文件。然后采用 3D 打印技术制作。

其 3D 打印是一种快速成型技术，它是一种以数字模型文件为基础，运用粉末状金属或塑料等可粘合材料，通过逐层打印的方式来构造物体的技术。3D 打印通常是采用数字技术材料打印机来实现的。常在模具制造、工业设计等领域被用于制造模型，后逐渐用于一些产品的直接制造，已经有使用这种技术打印而成的零部件。该技术在珠宝、鞋类、工业设计、建筑、工程和施工（AEC）、汽车，航空航天、牙科和医疗产业、教育、地理信息系统、土木工程、枪支以及其他领域都有所应用[5]。

本次设计的实体外壳由三角洲 3D 打印机打印，分为底座和盖两部分。组装起来，长宽高分别为 84.5mm、121mm、80mm。材料为 PLA，形变温度为 120℃左右。

2.3 系统硬件电路设计

本节主要介绍阐述系统的电子硬件，系统包括多个不同功能的模块，有输入输出电路、通讯电路、模拟采样电路、电源模块、嵌入式系统电路等主要模块。

2.3.1 单片机电路

将运算器、控制器、存储器和各种输入 / 输出接口等计算机的主要部件集成在一块芯片上，就能得到一个单芯片的微型计算机。它虽然只是一个芯片，但在组成和功能上已经具有了计算机系统的特点，因此称之为单片微型计算机，简称单片机[1] 。因为其体积小、功耗低、价格低廉、抗干扰能力强且可靠性高，适合应用于工业过程控制、智能仪器仪表和测控系统的前端装置[2]。本次毕业设计所采用的是 Atmel328P。以下简述本次毕业设计的单片机电路。

主要特性：

（1）先进的 RISC 结构，低功耗的 AVR（R）8 位微控制器

（2）系统内可编程的 Flash32K 字节 ，寿命为 10 万次写/擦循环，数据可保留时间为 10 年

（3）1K 字节的 EEPROM，2K 字节的片内 SRAM

（4）6 通道 10 位 ADC PDIP 封装

（5）可编程的串行 USART

（6）23 可编程 I / O 线

（7）28 引脚 PDIP

（10）低功耗的闲置和掉电模式

（11）内部校准振荡器

管脚说明：

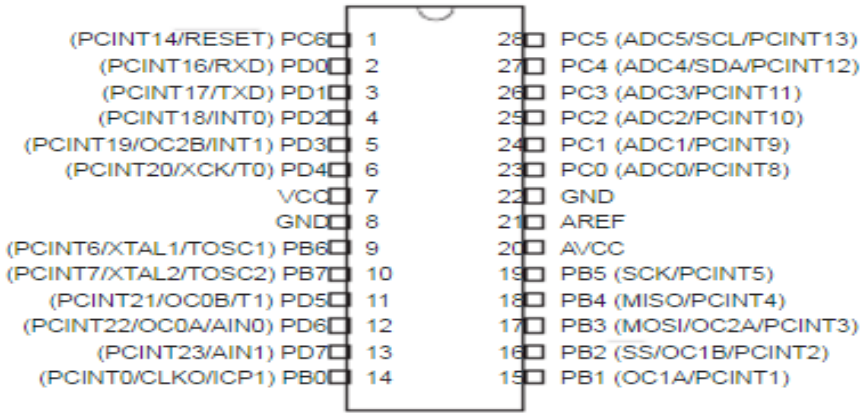

图 2-3 单片机引脚图

VCC：供电电压。　GND：接地电压

管脚 1：RESET 复位引脚

管脚 2：串口 RX 端

管脚 3：串口 TX 端

管脚 4-6：对应 Q0.0-Q0.2

管脚 9-10：时钟引脚

管脚 5-9：对应输出 Q0.3-Q0.7

管脚 10-13：对应 I0.0-I0.3

管脚 23-26：对应 I0.4-I0.7

管脚 27-28：对应 A0 和 A1

I0.0-I0.7 和 Q0.0-Q0.7 引脚都属于通用引脚，可配置成推挽，开漏，上拉，浮空等模式，I0.0-I0.7 被配置成了浮空模式，用于信号采集输入。Q0.0 到 Q0.7 引脚都被配置成了推挽输出模式，接输出电路，用于信号输出。A0 和 A1 被配置成模拟输入引脚。

时钟电路说明：

时钟电路产生系统时钟，该时钟电路由晶体振荡器、单片机内部的晶震控制芯片和电容组成。

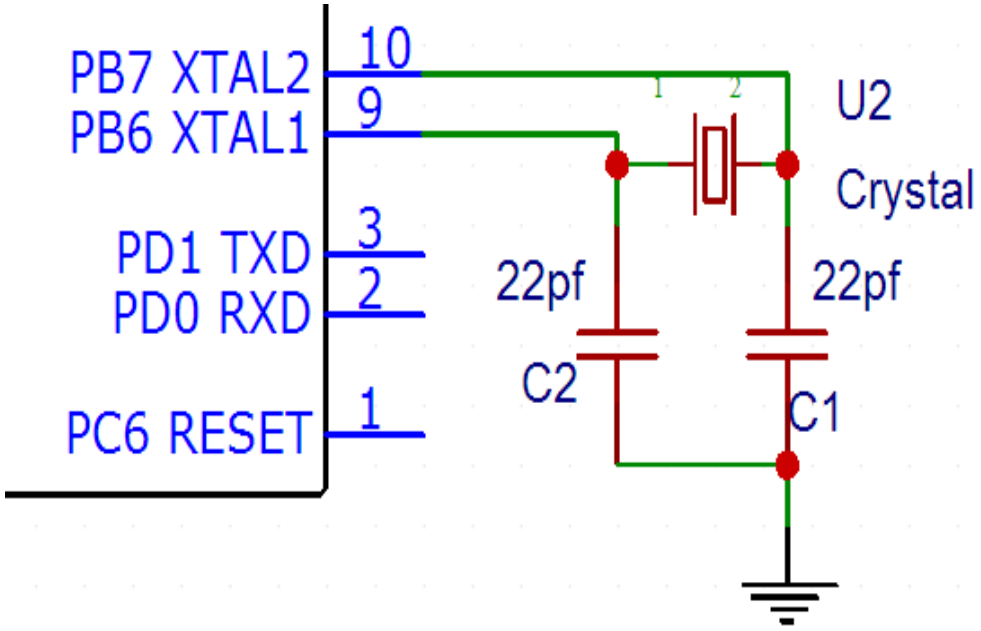

图 2-4 单片机时钟电路

其中 Atmel328P 单片机的 9、10 引脚为时钟引脚，时钟电路产生的时钟周期决定了该单片机的最小时钟单位。系统采用的是 8.00Mhz 的时钟晶振，所以时钟电路的周期为 8 X 10^8 分之一。

2.3.2 输入输出电路

图 2-5 是 I0.0 到 I0.7 输入电路图，采用光耦隔离输入，RC 滤波，防止输入过压。光耦采用 PC817C 其输入电压具有 5000v 隔离，可有效的将外部引脚和单片机引脚隔离起来。R2 和 C1 起到滤波作用防止信号误动作，齐纳二极管起到对光耦的两端过压保护作用[8]。LED 时输入信号指示灯，输入信号为高电平时亮。

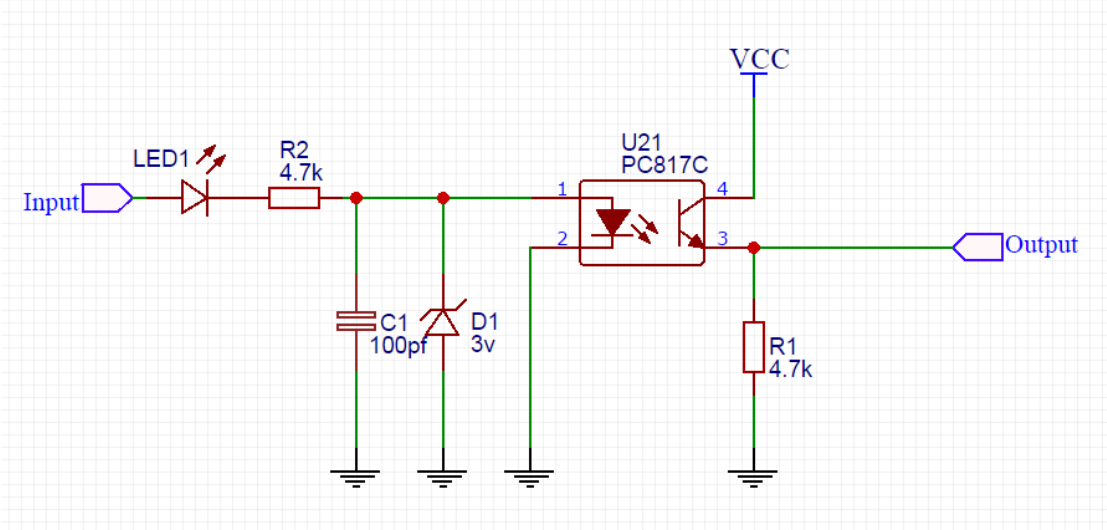

图 2-5 输入电路图

图 2-6 输出电路，Q0.0 到 Q0.7 是接输出电路的引脚，被配置成推挽输出模式，通过光耦 PC817C 传递信号，来隔离外部输出和单片机引脚[9]；LED9 为信号指示灯，亮灯时说明输出闭合；三极管 S8050 用来放大电流，来达到可以驱动继电器的电流大小，继电器只使用了 2 和 5 引脚输出，默认是常开；二极管 D2 用来防止感应电流产生的反电动势损坏电子元器件，可控制电流可以小于 5A，可以直接驱动一些小负载电气设备，但是继电器存在一定的缺点，就是开关速度不够快，Q0.0-Q0.3 共用第一个 COM 端，Q0.4-Q0.7 共用第二个 COM 端。

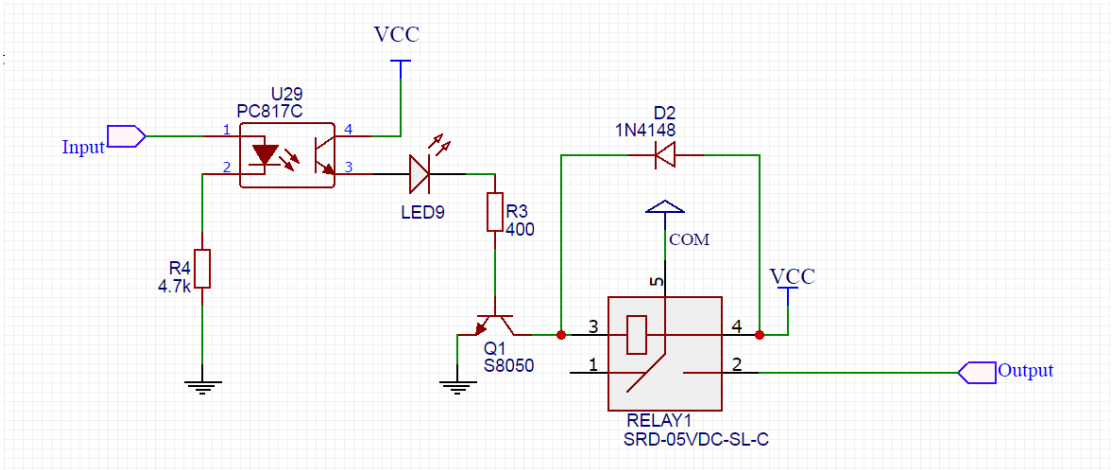

图 2-6 输出电路图

2.3.3 外部通讯电路

PLC 和上位机的通讯可以采用 RS232 线连接，PLC 中集成了一块 TTL 转 RS232 的芯片-MAX232；MAX 芯片的 11 引脚 TX 接单片机的 RX 引脚，MAX232 的 12 引脚 RX 接单片机的 TX 引脚，电容 C10 起稳定电源作用，MAX232 的 14 引脚为 TX，13 引脚为 RX，分别接到 DB9 接头的 3 和 2 引脚上；外部连接时，可以直接通过 DB9 线和带有 DB9 对的上位机通讯。

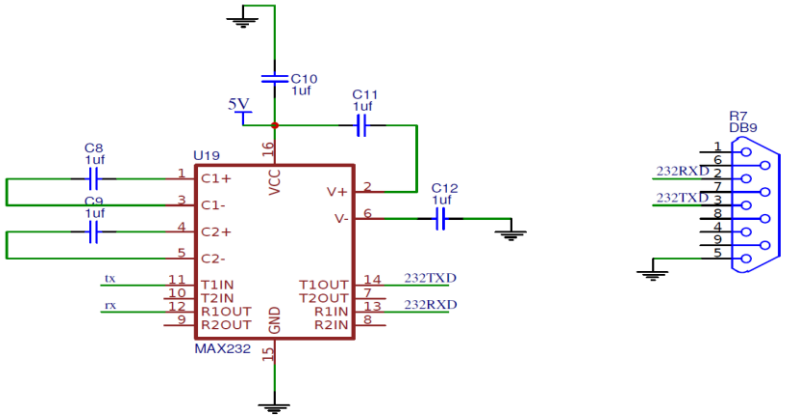

图 2-7 TTL 串口转 RS232 电路原理图

2.3.4 模拟电压检测电路

PLC 具有两通道的模拟电压检测电路，一共有两个通道，A0 和 A1。如图 3-10；电压检测范围为 0V 到 24V，其精度可以达到 12 位，AnalogInput 接外接端子，AnalogOutput 接单片机 27/28 号模拟引脚；电压信号从 AnalogInput 输入接运放 5 脚，运放 5 脚输入电阻超过 500K，对外部信号几乎不影响，运放对电压 1：1 放大输出，然后通过精密可调电阻分压，端子 AnalogInput 的 0-24v 电压就可以 24：5 比例放大映射到了 AnalogOutput 端子的 0-5V 间。

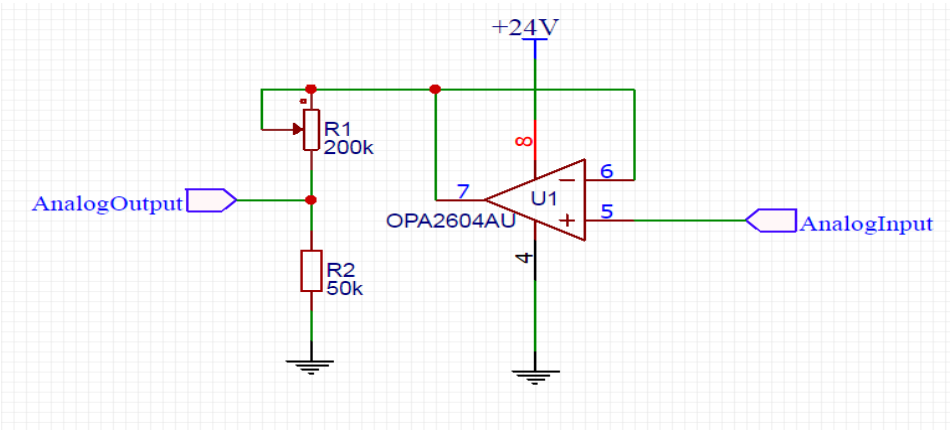

图 2-8 模拟电压检测电路

第三章 系统软件设计

设计中软件有两部分，一个是单片机里的硬件代码，还有一个是 PLC 编程软件。两处的代码相互配合衔接，来实现 PLC 的功能。本章将对核心代码的实现进行阐述。

3.1 控制系统程序设计

在一个周期内 PLC 的执行时主要分三个步骤，先读取输入寄存器状态，再执行程序，然后修改输出寄存器状态值；另外 PLC 的主程序是循环扫描执行的，PLC 的一个运行周期，一般时间在几毫秒到十几毫秒不等，所以程序当中一定不能出现阻塞的程序。

在上面这几点的要求上进行程序设计。本设计的单片机采用的 Atmel328P，它是 Arduino IDE 平台默认支持的单片机（Arduino 是一款开源的单片机软硬件）[7]，Arduino IDE 里集成了 AVR GCC 编译器，并且程序代码开源，方便设计开发。

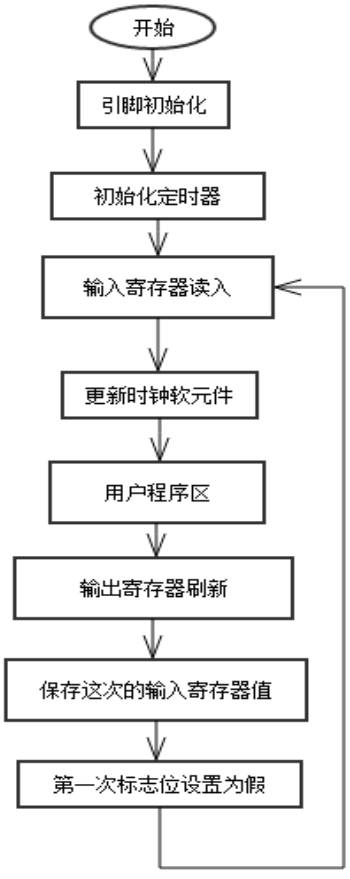

图 3-1 主函数流程图

主函数开始进行引脚初始化，把输入引脚配置成浮空输入模式，输出引脚设置成推挽模式；然后初始化定时器，定时器设置为 10ms；接着进入周期循环；周期循环首先读取输入寄存器；然后更新时钟软元件；接着就执行梯形图生成的用户程序；然后输出寄存器更新；接着讲这次输入寄存机的值保存起来，用于下一个周期上升沿和下降沿判断使用；然后把第一次周期标志位清除；接着又回到循环第一步。就这样一直循环扫描执行程序。

3.1.1 定时软元件实现

图 3-2 为定时中断函数流程图，本设计一共 0-9 十个定时软元件；定时中断函数作用是，更新定时软元件的时间值。

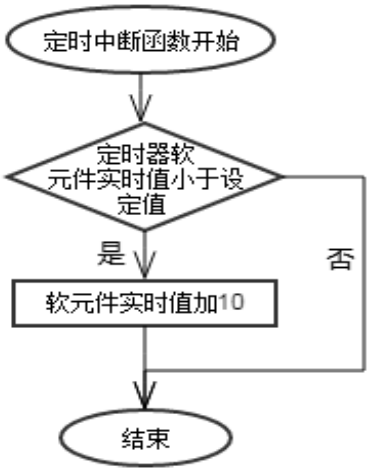

图 3-2 定时中断函数流程图

图 3-3 为定时中断函数，Tvar[]为定时软元件实时值，Tset[]为定时软元件设定的值，当进入中断函数后，对 T0 到 T9 判断实时值小于设定值的软元件通通自加 10。

```
void flash()//定时中断函数
{
  if (Tvar[0] < Tset[0])Tvar[0]+=10;
  if (Tvar[1] < Tset[1])Tvar[1]+=10;
  if (Tvar[2] < Tset[2])Tvar[2]+=10;
  if (Tvar[3] < Tset[3])Tvar[3]+=10;
  if (Tvar[4] < Tset[4])Tvar[4]+=10;
  if (Tvar[5] < Tset[5])Tvar[5]+=10;
  if (Tvar[6] < Tset[6])Tvar[6]+=10;
  if (Tvar[7] < Tset[7])Tvar[7]+=10;
  if (Tvar[8] < Tset[8])Tvar[8]+=10;
  if (Tvar[9] < Tset[9])Tvar[9]+=10;
}
```

图 3-3 定时中断函数

图 3-4 为定时器软元件更新函数，这个函数在一个循环周期里的开始被调用，用来更新定时软元件的闭合关断，本设计一共有十个软元件。

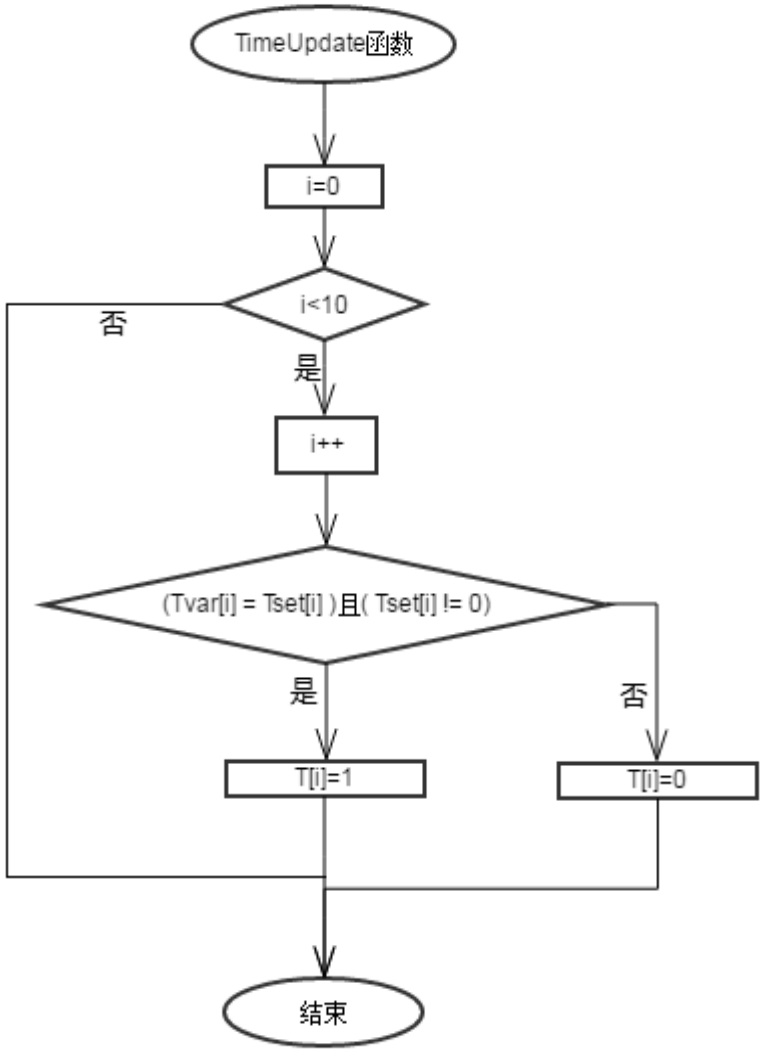

图 3-4 定时元件更新函

用 for 循环遍历 0 到 9 的每个软元件，当定时软元件当时值等于设定值且不等于 0 时，定时软元件闭合。实现代码如图

```
void TimeUpdate() {//更新定时器状态
  for(int i = 0; i < 10; i++)
    T[i] = (Tvar[i] == Tset[i] && Tset[i] != 0) ? 1 : 0;
}
```

图 3-5 定时更新函数代码

3.1.2 上升下降沿软元件实现

上升沿下降沿是 PLC 的基本功能之一，本系统通过对上一次数据保存，达到判断上升沿下降沿的功能，在 PLC 中使用上升下降沿检测时，IDE 软件将生成如下代码，在用户程序当中，来实现其检测功能。xsc[]/ysc[]数组存储着上个周期的 x/y 输入输出点的状态，x[]/y[]存储着本次的 x/y 输入输出点的状态，one 是第一个循环周期的标志变量，当上次周期的记录为 FALSE 本次记录的为 TRUE，且不为第一个周期时，上升沿软元件将闭合。当上次周期的记录为 TRUE 本次记录的为 FALSE，且不为第一个周期时，下降沿软元件将闭合。

```
(((xsc[1]==false && x[1]==true && one==false)?true:false))//x检测上升沿
(((ysc[1]==false && y[1]==true && one==false)?true:false))//y检测上升沿
(((xsc[1]==true&& x[1]==false&& one==false)?true:false))//x检测下降沿
(((ysc[1]==true&& y[1]==false&& one==false)?true:false))//y检测下降沿
```

图 3-6 上升沿下降沿软元件实现

3.1.3 其它软元件实现

PLC 还有其它的各种软元件功能；输入软元件 I0.0-I0.7 是在程序中新建了一个 BOOL 类型的一维数组，通过程序将输入寄存器的值映射到一维数组当中，当使用输入软元件之时，实际调用的是数组元素的值；同样输出软元件 Q0.0-Q0.7 同样是在程序当中新建了一个 BOOL 类型的一维数组，在用户程序中使用输出线圈时，实际修改的是数组元素的值，然后再循环周期结束时更新输出；辅助寄存器 M 一共有 50 个，本质就是一个元素为大小为 50 的一维数组，因为篇幅有限不能一一详述。文章将在下节软件程序设计时也会有介绍，由于是软硬配合实现的其功能，所以不能界限分明的介绍各个部分，

3.2 编程软件程序设计

软件方面本设计使用 Qt Creator 开发，PLC 可以进行梯形图编程、结构代码编程、PLC 通讯及下载、项目管理等功能。软件内部使用了 Qt 库、Arduino、cmd 等这几个主要资源来实现其功能。

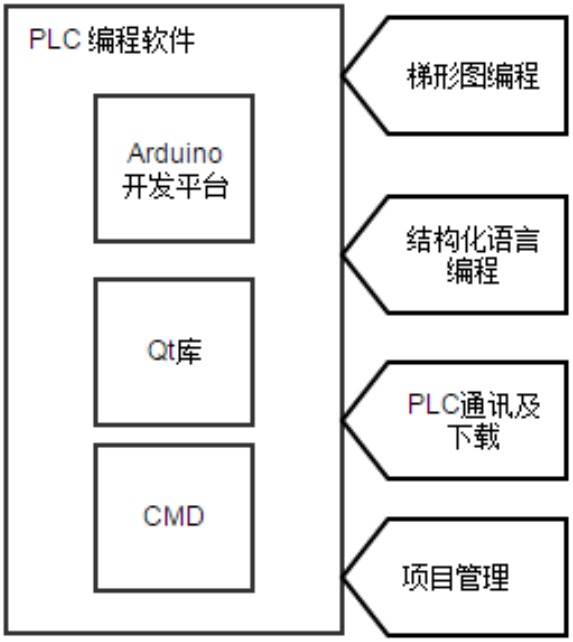

图 3-7 软件结构及功能

在设计较复杂的程序时，一般采用自顶向下的方法，将问题划分为几个部分，各个部分再进行细化，直到分解为较好解决问题为止。模块化设计，简单地说就是程序的编写不是一开始就逐条录入计算机语句和指令，而是首先用主程序、子程序、子过程等框架把软件的主要结构和流程描述出来，并定义和调试好各个框架之间的输入、输出链接关系逐步求精的结果是得到一系列以功能块为单位的算法描述。以功能块为单位进行程序设计，实现其求解算法的方法称为模块化。模块化的目的是为了降低程序复杂度，使程序设计、调试和维护等操作简单化[6]。本设计采用模块化设计，其中 Arduino 主要对底层的代码进行编译；CMD 进行硬件系统信息查询；Qt 库函数编程来实现其它主要功能。本节将主要对 IDE 软件的功能和梯形图编译的实现方法进行介绍。

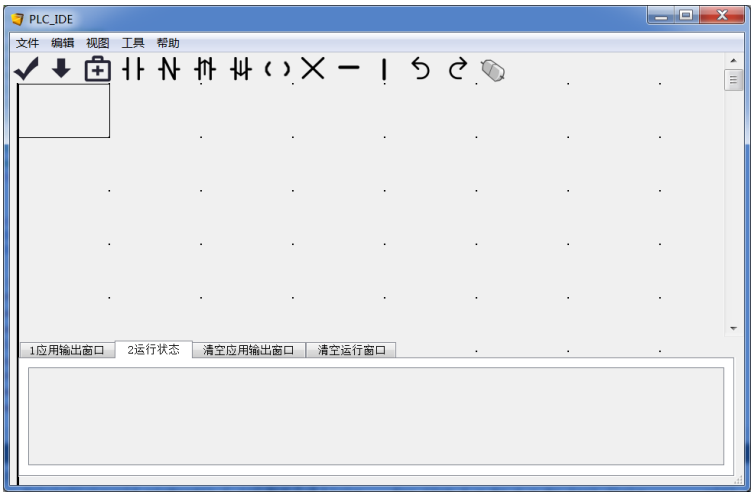

图 3-8 软件界面

3.2.1 IDE 功能介绍

IDE 编程软件的功能很多，本节从菜单栏的顺序来介绍这些功能.首先是文件菜单栏，点击(打开/新建/保存/另存)都会出现文件窗口，项目的后缀名都为".plcpro"，项目名称支持中文字母数字下划线，操作目录可任意切换。

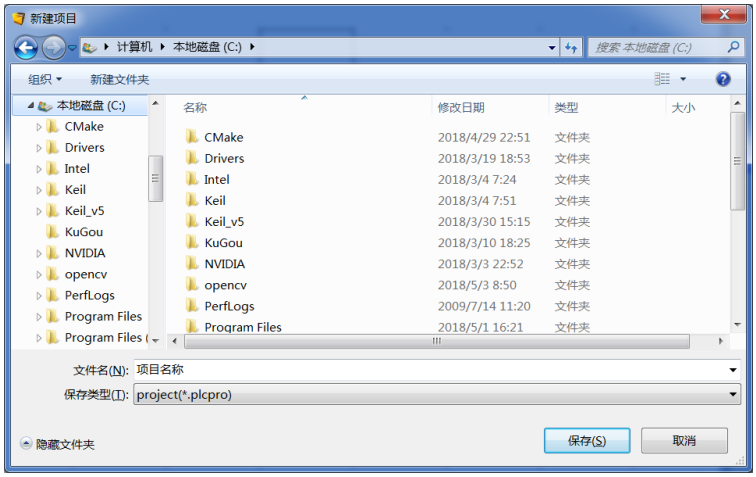

图 3-9 文件窗口

关闭选项卡点击时会默认保存项目文件后再保存；示例选项卡下有示例项目，示例项目可以方便用户熟悉软元件的使用方法，最近访问选项卡下有最近操作的十个项目文件；设置选项卡下可以进行软件的配置，退出选项卡点击后会保存本次操作记录然后关闭 IDE 软件。

编辑菜单栏下有还原、撤回、剪切、复制、黏贴等选项卡，可以点击操作也可以快捷键操作。

视图菜单栏下有：查看生成代码选项卡、关闭/打开输出窗口选项卡、视图语言切换窗口、字体加减选项卡；点击“查看生成代码”选项卡可以查看到梯形图编译后的用户程序；

“关闭/打开输出窗口”选项卡可以决定窗口下面信息输出窗口是否显示；“视图语言切换窗口”可以选择窗口是梯形图编辑页面还是代码编辑页面。工具菜单栏下有查找选项卡可以查找代码里的关键词。最后一个是帮助菜单栏，遇到技术问题或有疑问可以在当中查找。

图 3-10 窗口上方图标

图 3-10 时 IDE 软件窗口上的图标，从左到右分别是编译、下载、调试、然后接下的几个是梯形图中的逻辑单元选择，依次是常开、常闭、上升沿、下降沿、输出线圈、万能逻辑单元、横向连接、竖直方向连接，接着是对梯形图编辑的撤回和还原，然后是串口连接图标。

3.2.2 编译程序模块

IDE 软件最核心的部分就在于其编译模块部分了，编译模块将梯形图编译成编程简单的与或非逻辑语句，下面将介绍编译函数是如何建梯形图编译成与或非逻辑语句的。

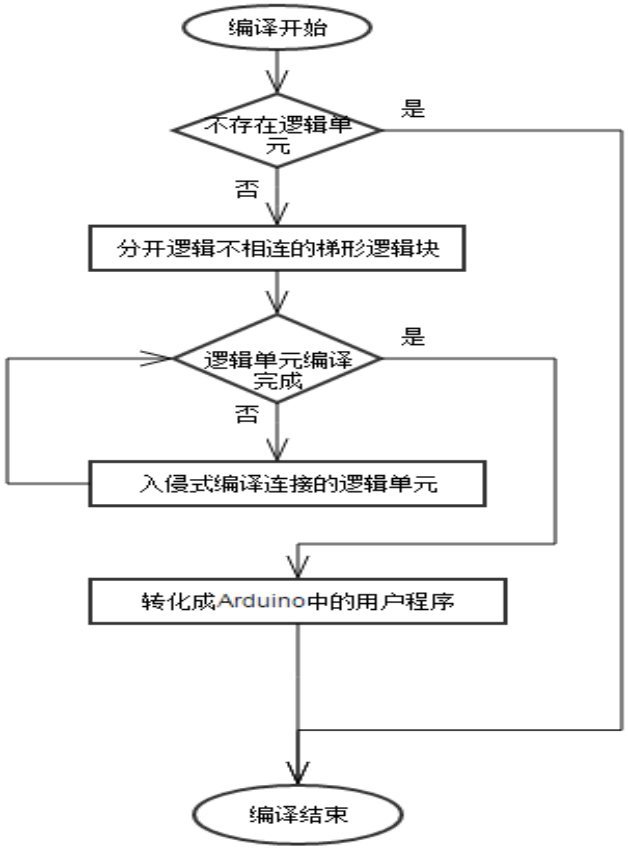

图 3-11 编译函数流程图

编译开始首先判断项目的梯形图程序是否存在逻辑单元，如果没有逻辑单元直接结束编译；然后是对梯形图逻辑连接在一起的块进行拆分；接着是将拆分成一块一块的梯形图块进行编译，编译完所有梯形图块后将其按顺序转化成可下载的用户程程序。

梯形图中的逻辑单元都是已链表的形式存储，进行编译主要就是对链表里的数据进行操作。如果添加了逻辑单元就会在链表的结尾添加一个节点链，每个节点链都是一个结构体变量。

```
struct logic{
    int x=5555;
    int y=5555;
    QString leiXin="";//常开，常闭，上身，下降
    QString pin_leiXin="";//x或y或m
    int bianHao=0;//引脚编号
    logic* addr=NULL;//下一个链表单元地址
    logic* addr2=NULL;//上一个链表单元地址
    bool Del=false;//撤回还原时候做相反操作
    int changDu=0;
    logic *shi=NULL;//撤回还原(多个逻辑时)时起启动停止作用
    logic *mou=NULL;//撤回还原(多个逻辑时)时起停止作用
    bool compileOK=0;//是否编译了
};
```

图 3-12 程序当中结构体原型

x 和 y 是该逻辑单元的位置信息；QString 类型是 Qt 框架的字符串类型，leiXin 字符串变量存储逻辑单元的类型；pin_leixin 字符串变量也是指定逻辑单元类型；biaoHao 是指定的引脚编号或者软元件编号；addr 和 addr2 是指向本类型的结构体指针变量，用于链表查询，addr 是存储的上下个节点链的地址，sddr2 是存储的下个节点链的地址；Del、changDu、shi、mou、这几个变量是编辑逻辑单元时用来进行撤回还原操作的；compileOK 标志该逻辑单元是否编译过了。

编译第一步，通过链表的长度，来判断是否有梯形图逻辑单元。第二步分开不相连的逻辑单元块，首先把链表里的节点的信息全部读取出来，存成数组的模式；然后在梯形图中找到满足最上方最左方的一个逻辑单元，接着由它连接前后左右与其互通的逻辑单元，连接到的新逻辑单元，也执行同样操作，连接自己四周的逻辑单元，直到连接不到新的逻辑单元后。连接到的所有逻辑单元即是一块逻辑单元块。第三步判断是否编译完成，通过读取链表节点的 compileOK 单元可以知道节点是否编译完成。如果没有编译完成，则执行第四步，第四步是将第二步中的逻辑块进行逻辑连接并生成代码；最后第五步将生成的代码转换成可下载的用户程序。

第四章 实际测试结果

4.1 单项功能测试

为了发现尽迄今尚未发现的问题，保证软件的可靠性和用户体验，所以需要进行测试来找到潜伏在软件里的缺陷。本章先对要执行测试的软件进行了分析，确定测试策略为，常用功能测试进行 80%的测试，非常用功能测试进行 20%的测试。测试步骤，先测试菜单单个功能，在测试组合功能，然后进行实际使用过程测试。最后将对软件的多个极限进行测试。

首先测试菜单栏功能，打开项目功能测试：打开已存在项目、打开刚新建项目、打开一个项目后接着打开另一个项目、打开一个项目后再次重复打开这个项目。新建项目测试：新建一个项目、新建一个带有符号文件名的项目、同一文件下新建一个重复项目名，关闭项目测试：关闭一个空项目，关闭一个存在逻辑单元的项目，保存项目测试：保存一个空项目、保存一个修改后的文件，另存项目测试：另存在一个文件夹、存在同一个文件夹，示例使用测试：打开示例项目、重复打开项目、连续多次打开不同项目，最近访问项目文件打开测试，重复打开同一个文件最近文件。

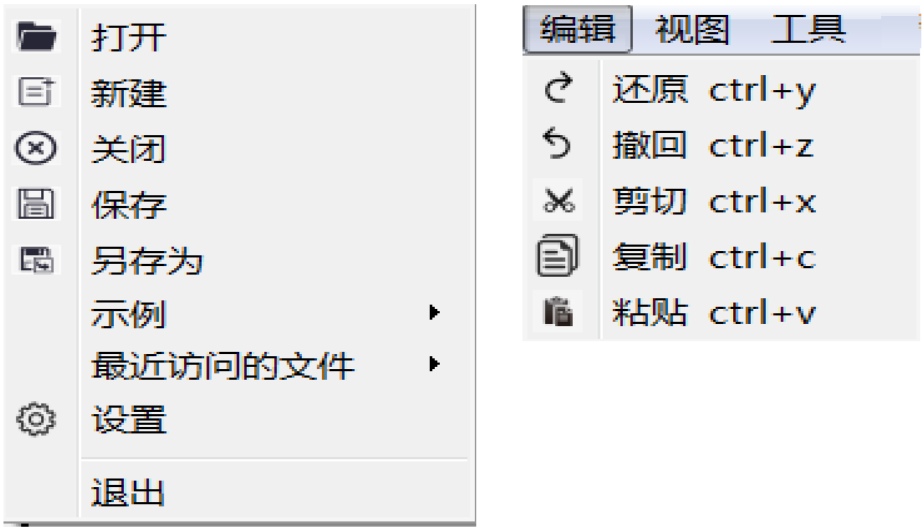

图 4-1 文件和编辑菜单栏下功能

上面对项目文件有关的操作进行的一系列操作，发现了打开重复文件时系统报错软件不可用，然后进行了改正，改正后再次测试不存在文件打开时，软件系统报错消失。

然后进行的测试是，梯形图测试编辑测试，测试梯形编辑的删除、撤回、还原、复制、剪切等操作。进行单个功能测试和多个功能复合测试。测试中发现撤回和还原功能存在问题，无法对竖向连接单元进行撤回操作，发现程序中撤回没有包括该种类型，在程序中添加类型后问题解决。接着又测试了视图工具等功能。

4.2 综合测试

上节进行了软件的单个功能测试，由于硬件测试需用软件配合使用，于是这里将软硬件一起测试结果。选定一个流水灯项目作为实际测试项目；按照如下步骤：首先新建一个名为流水灯的项目；然后进行梯形图编程，梯形图如图 4-2：

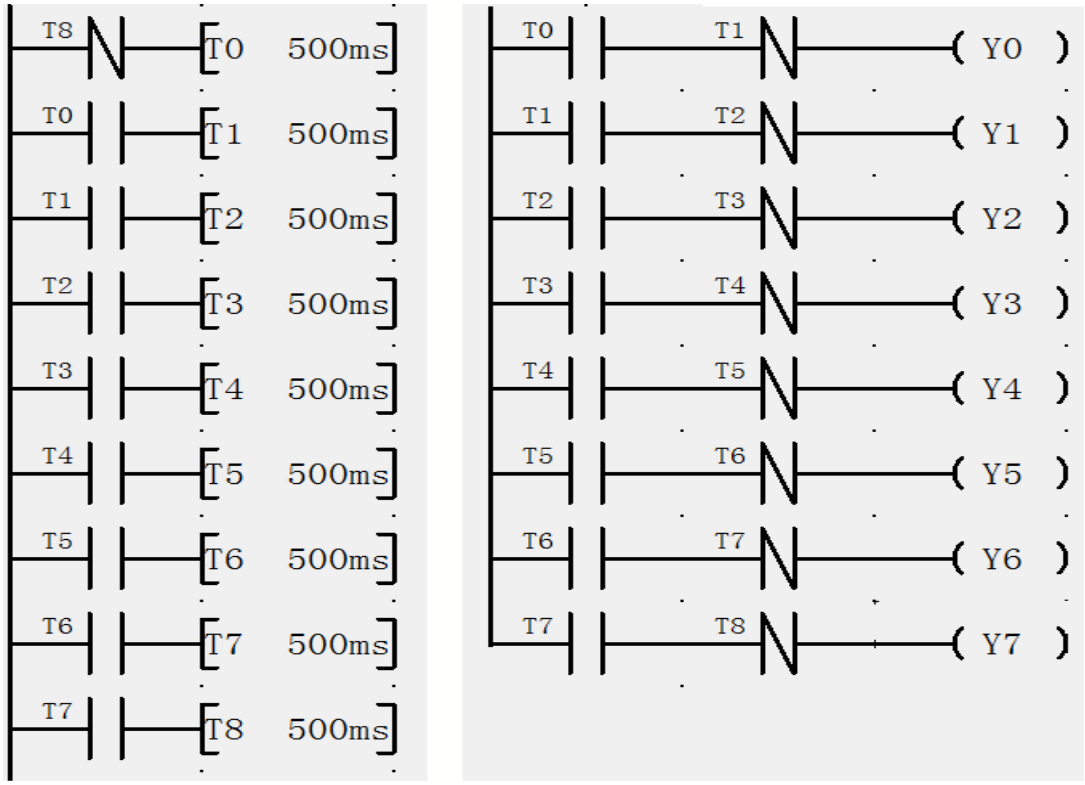

图 4-2 流水灯程序

梯形图输入结束，然后进行编译，测试编译结果，点击编译按钮后，查看生成的代码是否正确；发现生成代码如下：

```
if((((!T[8])))&&(Tvar[0]==0))
Tset[0]=500;
else if(((!((!T[8])))&&(Tvar[0]!=0))
Tvar[0]=Tset[0]=0;
if((((T[0])))&&(Tvar[1]==0))
Tset[1]=500;
else if((!((T[0])))&&(Tvar[1]!=0))
Tvar[1]=Tset[1]=0;
if((((T[1])))&&(Tvar[2]==0))
Tset[2]=500;
else if((!((T[1])))&&(Tvar[2]!=0))
Tvar[2]=Tset[2]=0;
if((((T[2])))&&(Tvar[3]==0))
Tset[5]=500;
else if((!((T[4])))&&(Tvar[5]!=0))
Tvar[5]=Tset[5]=0;
if((((T[5])))&&(Tvar[6]==0))
Tset[6]=500;
else if((!((T[5])))&&(Tvar[6]!=0))
Tvar[6]=Tset[6]=0;
if((((T[6])))&&(Tvar[7]==0))
Tset[7]=500;
else if((!((T[6])))&&(Tvar[7]!=0))
Tvar[7]=Tset[7]=0;
if((((T[7])))&&(Tvar[8]==0))
Tset[8]=500;
```

图 4-3 生成代码-上

```
Tset[3]=500;
else if((!((T[2])))&&(Tvar[3]!=0))
Tvar[3]=Tset[3]=0;
if((((T[3])))&&(Tvar[4]==0))
Tset[4]=500;
else if((!((T[3])))&&(Tvar[4]!=0))
Tvar[4]=Tset[4]=0;
if((((T[4])))&&(Tvar[5]==0))
Tset[5]=500;
else if((!((T[4])))&&(Tvar[5]!=0))
else if((!((T[7])))&&(Tvar[8]!=0))
Tvar[8]=Tset[8]=0;
y[0]=((T[0]))&&((!T[1]));
y[1]=((T[1]))&&((!T[2]));
y[2]=((T[2]))&&((!T[3]));
y[3]=((T[3]))&&((!T[4]));
y[4]=((T[4]))&&((!T[5]));
y[5]=((T[5]))&&((!T[6]));
y[6]=((T[6]))&&((!T[7]));
y[7]=((T[7]))&&((!T[8]));
```

图 4-4 生成代码-下

经过确认发现生成代码正确，然后就可以将程序下载进 PLC 当中观察实际结果了；第一步先通过下载线将 PLC 和电脑连接，然后查看电脑是否搜索到该串口，接着点击下载，会弹出提示窗口图 4-5，点击 OK 确认；然后过一会下载成功会显示提示窗口图 4-6。

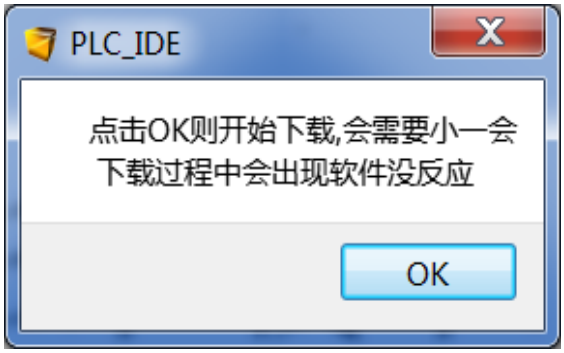

图 4-5 提示窗口

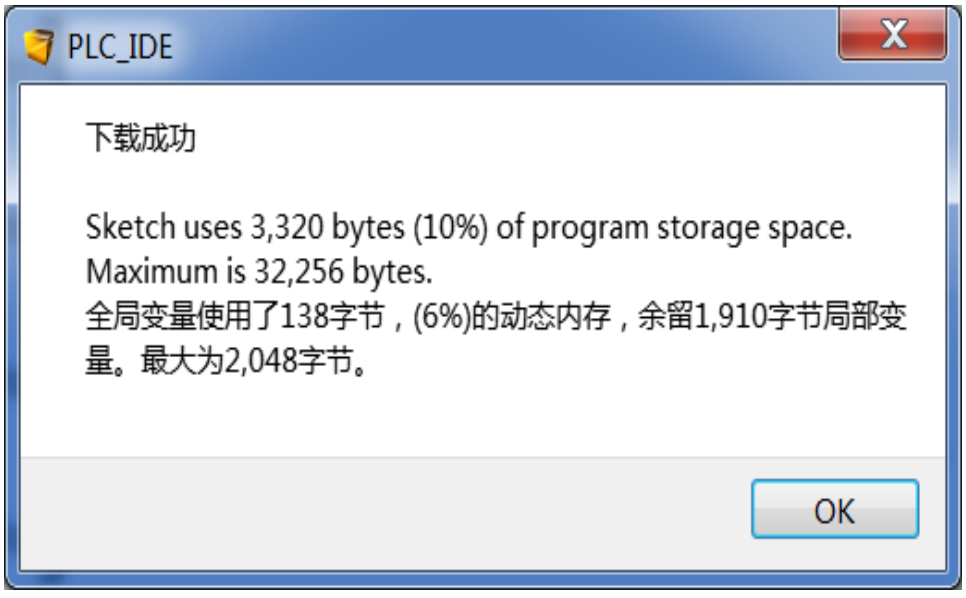

图 4-6 提示窗口

这时流水灯程序已经成功下载到了 PLC，并且开始在 PLC 上正常启动运行了。

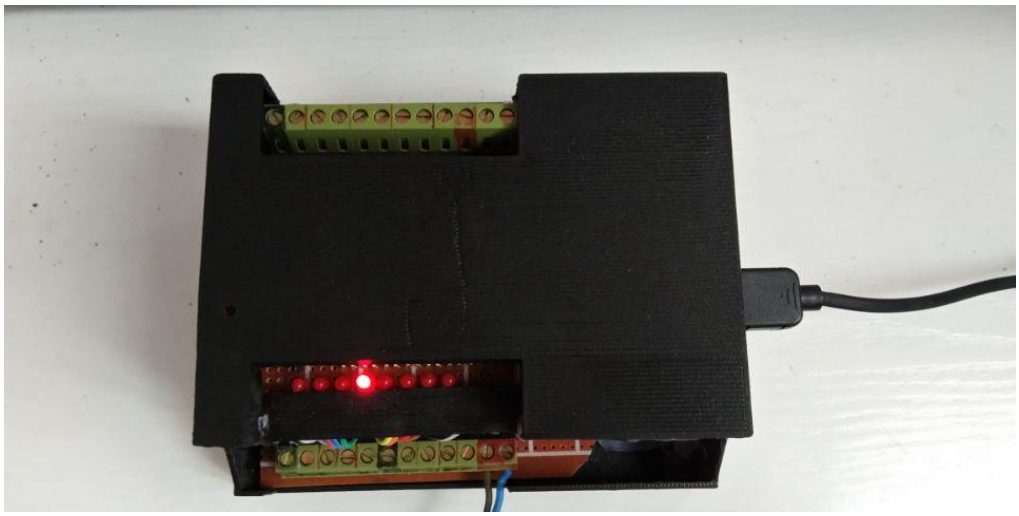

图 4-7 实际运行图

效果是 Y1 到 Y7 指示灯依次亮灯，中间间隔 500 毫秒。效果和程序中编写的流程相同，所以本次测试成功。实物如图 4-7，看到的只是一个时刻。

第五章 总结与展望

5.1 总结

本设计软件硬件系统都达到了预期的要求。本次毕业设计主要完成的工作有：硬件电路图设计硬件代码设计编程、硬件制作、软件设计编程、PLC 外形设计。以下是具体的总结：

（1）硬件以 Atmel328P 单片机为核心进行系统设计，完成输入采集、继电器输出、模拟信号采集；数字输入输出通道采用光电耦合器隔离。模拟输入通过运放输入引脚输入。由于输入端为运放输入引脚所以输入电阻极大，不会干扰输入源。

（2）软件 Qt 为框架进行设计，完成了梯形图编程功能，高级语言编程功能，项目管理功能，程序下载功能。

（3）外壳设计采用 3DMAX 设计 3D 打印机打印，起到了保护内部元件、美观、安全的功能。

软硬件功能可以很好的配合，IDE 软件编写的程序可以在 PLC 上正常运行。本次设计成果，具有 PLC 基本功能，总的来说此次设计达到了预期要求。

5.2 展望

虽然本次设计已近结束了，但是设计的软硬件功能还有很多提升的空间，比如 PLC 的指令功能方面，硬件的合理性和功能方面，外壳安装的方便性、稳固性，等多方面都有提升的空间。

（1）在软件方面，PLC　IDE 软件中三菱的 Works 和西门子公司的 TIA(博途)都是非常优秀的 IDE 软件，在以后的设计中可以借鉴它们软件的设计理念和 UI 界面。

（2）在 PLC 的硬件方面，设计采用的 Atmel328P 芯片硬件不强，采用 Atmerl2560 芯片可以增加到平时的 3 到 4 倍的输入输出引脚数量，而且 RAM 和 ROM 空间都相应增大了，可以运行更大的用户程序。

（3）其它的方面，以后可以设计不同类型的 PLC，来适应不同场合和环境。而且 PLC 的稳定性是非常重要，由于底层使用的 Arduino 开发环境，所以 PLC 稳定性由 Arduino 开发环境决定。不能达到有些冗余 PLC 和多核表决式 PLC 的系统稳定性。

在以上这些方面本设计的软硬件仍然有提升和改进的潜力。

致 谢

本次毕业设计是在导师陈揆能老师的细心指导下完成的。本设计的撰写过程中，老师给予了大力支持和细心指导,指出了我设计中的很多不足和需要改进的地方。在此真诚感谢老师的辅导。同时也要感谢 Arduino 团队的人员，开发出 Arduino 这样优秀的开发板和使用环境，并将其免费开源软硬件出来方便大家学习使用。没有 Arduino 的存在我的 PLC IDE 也将不可能成功。这样的付出是伟大的，开源让这个世界变得更好，因此我也将传承这种开源共享的精神。将我的本设计的所有源码以及资料公开在 Github 上让其变得更好。

参考文献

[1]高玉萍.AT89S51 单片机实验系统的开发与应用[J].现代电子技术，2015.

[2]李乔.基于单片机的便携式温度控制系统[J].自动化与仪器仪表，2011.

[3]王绍伟.PLC 开源协议[Z],2013.

[4]王绍伟.PLC 开源协议[Z],2013.

[5]孙聚杰.3D 打印材料及研究热点[M].丝网印刷,2013.

[6]吴登峰，邢鹏飞.C 语言程序设计[M].中国水利水电出版社，2015.

[7]张菁. 单片机控制系统方案的研究[J].上海交通大学学报，2007.

[8]王永利. 光电耦合器的工作原理及检测[J]. 家电维修技术， 2010.

[9]易小月，张斌，李文启. 光电耦合器件的研究[J]. 才智，2009.

[10]郝久清，肖立.PLC 控制系统的可靠性设计[M].自动化仪表，2005.

附 录

PLC 硬件程序：

```
#include <FlexiTimer2.h>

bool one = true;
int x_pin[8] = {10, 11, 12, 13, 14, 15, 16, 17};
int y_pin[8] = {2, 3, 4, 5, 6, 7, 8, 9};
bool x[8] = {0, 0, 0, 0, 0, 0, 0, 0};
bool y[8] = {0, 0, 0, 0, 0, 0, 0, 0};
bool xsc[8] = {0, 0, 0, 0, 0, 0, 0, 0};
bool ysc[8] = {0, 0, 0, 0, 0, 0, 0, 0};
bool T[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0};//定时器是否闭合
int Tvar[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; //定时器当前的值
int Tset[10] = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}; //定时器设定的值
bool m[50]={0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 0,0, 0, 0,
0, 0, 0, 0, 0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

void flash()//定时中断函数
{
if (Tvar[0] < Tset[0])Tvar[0]+=10;
if (Tvar[1] < Tset[1])Tvar[1]+=10;
if (Tvar[2] < Tset[2])Tvar[2]+=10;
if (Tvar[3] < Tset[3])Tvar[3]+=10;
if (Tvar[4] < Tset[4])Tvar[4]+=10;
if (Tvar[5] < Tset[5])Tvar[5]+=10;
if (Tvar[6] < Tset[6])Tvar[6]+=10;
if (Tvar[7] < Tset[7])Tvar[7]+=10;
if (Tvar[8] < Tset[8])Tvar[8]+=10;
if (Tvar[9] < Tset[9])Tvar[9]+=10;
}
void TimeUpdate() {//更新定时器状态
for(int i = 0; i < 10; i++)
T[i] = (Tvar[i] == Tset[i] && Tset[i] != 0) ? 1 : 0;
}
void pinRead() { //引脚状态读入
```

```
for (int i = 0; i < 8; i++) {
x[i] = digitalRead(x_pin[i]);
y[i] = digitalRead(y_pin[i]);
}
}
void pinWrite() { //引脚状态输出
for (int i = 0; i < 8; i++) {
digitalWrite(y_pin[i],y[i]);
}
}
void pinInit() { //引脚配置
for (int i = 0; i < 8; i++) {
pinMode(x_pin[i],INPUT);
pinMode(y_pin[i],OUTPUT);
}
}
void pinSC() { //保存上次旧值
ysc[0] = y[0]; ysc[1] = y[1]; ysc[2] = y[2]; ysc[3] = y[3];
ysc[4] = y[4]; ysc[5] = y[5]; ysc[6] = y[6]; ysc[7] = y[7];
xsc[0] = x[0]; xsc[1] = x[1]; xsc[2] = x[2]; xsc[3] = x[3];
xsc[4] = x[4]; xsc[5] = x[5]; xsc[6] = x[6]; xsc[7] = x[7];
}


void setup() {
pinInit();//引脚初始化
FlexiTimer2::set(10, flash); //10 毫秒，定时器 2 开启，10 毫秒后调用 flash 函数
FlexiTimer2::start(); //定时器 2 开启
}

void loop() {
pinRead();//引脚状态读入
TimeUpdate();//时钟更新

//inputCode
```

```
pinWrite();//引脚输出
pinSC();//保存上次旧值
one = false;//第一次为真
}
```