生物发酵罐集中控制系统的实现①

Implementation of the Contral System of Bioreactors

朱　宇　（集美大学计算机工程学院　福建厦门 361021）
杜翠红　（集美大学生物工程学院 福建厦门 361021）

摘　要： 本文以西门子的 S7－200 可编程控制器(PLC)作为生物发酵罐的主控制器，利用传感器检测发酵过程的动态数据，采用 VC6.0 作为开发工具设计上位机的监控程序，实现了上位机(PC)与可编程控制器(PLC)之间的数据通信，达到了多台发酵罐进行集中控制的目的。

关键词： 可编程控制器(PLC)　上位机(PC)　Vc6.0　发酵罐

1　引言

近年来，随着生物技术的迅速发展.许多研究成果都需要经过发酵工业而转化为产品，但在进行大规模产业化之前，需要在实验室进行大量的发酵条件的优化，而对于微生物发酵过程，机理十分复杂，影响因素错综复杂，为此科研人员需要进行多次数据采集。长期以来，科研人员必须在实验室对发酵罐数据进行观察，靠经验对发酵罐进行实时控制。这样势必造成实验重复率高，大量耗费实验材料，往往实验结果还不能令人满意。为此，作者利用西门子的 S7－200 可编程控制器(PLC)作为下位机，采用 Vc6.0 作为开发软件，建立了多台发酵罐集中控制系统，这样一方面可以改善科研人员的实验环境，使工作人员可以在办公室就可以实时监视控制发酵罐的作业；另一方面通过实时的图形显示，可以更直观地观察到发酵罐中各参数的变化过程，从而对发酵罐进行自动控制。另外可以使科研人员同时控制多台发酵罐，从而提高工作效率，节省人力物力，加速发酵工艺产业化的步伐。

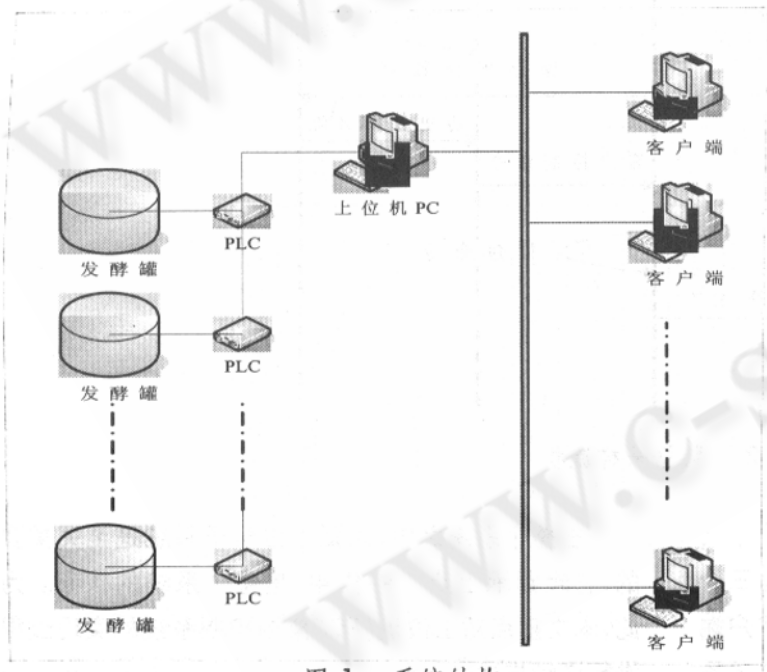

图 1　系统结构

2　系统设计

2.1　系统组成

系统是由发酵罐、S7－200PLC、编成电缆和通信适配器、上位机 PC 及客户机组成。S7－200PLC 各下位机通过 RS485 端口进行连接，通过编成电缆和通信适配器将 RS485 转换为 RS232 串行端口与上位机 PC 连接，上位机与客户机通过网线相连。(见图 1)

2.2　数据采样流程

本系统以西门子 S7－200 的 PLC 作为下位机对发

① 项目来源：集美大学科研启动基金资助项目(C60610)

酵罐数据进行采样，通过编成电缆和通信适配器与上位机 PC 进行连接，将采样数据通过串口上传至上位机，经上位机对上传数据进行统一处理。将最终数据保存入库，并在客户端将数据直接或以图形两种方式进行显示。同时可按用户的要求对发酵罐进行控制，将命令通过上位机 PC 下传至下位机 PLC，最终由下位机 PLC 将命令传至发酵罐的传感器，从而达到发酵罐自动控制的目的。（见图 2）

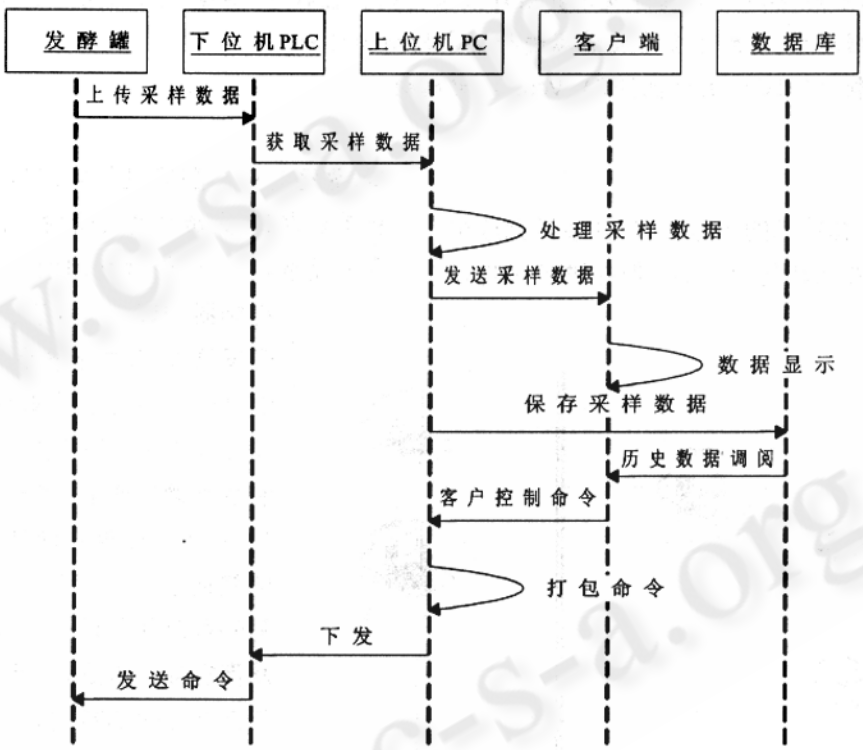

图 2　数据采样流程

2.3　系统软件结构

根据系统结构及部署情况，将整个系统分为三个子系统，分别是下位机子系统、上位机子系统、客户端子系统。通过系统数据采样流程的分析，将各个子系统完成的功能确定如下：

下位机子系统包括四个功能模块，分别为接收发酵罐采样数据模块、上传上位机采样数据模块、接收上位机命令模块及向发酵罐发送命令模块。

上位机子系统包括八个功能模块，分别为接收下位机采样数据模块、处理采样数据模块、向客户端发送采样数据模块、保存采样数据模块、接收客户端命令模块、命令打包模块、向下位机下发命令模块及异常处理模块。

客户端子系统包括四个功能模块，分别为接收上位机采样数据模块、数据显示模块、历史数据调阅模块及向上位机下发命令模块。

系统软件结构如图 3 所示。

3　上位机（PC）集中控制系统的实现

在整个系统之中，无疑上位机子系统是至关重要的，它起到承上启下的作用，是整个系统的核心。为此，本文重点对上位机（PC）集中控制系统的实现过程进行详细描述。

对于整个子系统的程序实现，我们以 VC + +6.0 作为开发工具，采用一体化程序设计思想（UML），使系统具有良好的健壮性和可塑性。

在系统程序设计中，存在的主要难点包括：对各个 PLC 输出数据包的识别、对每个 PLC 的采样数据的接收（要同时接收一组输出数据包）、对接收及命令数据包的重新定义、对多个 PLC 同时采样及多线程处理等。

本文中对以上难点均有详细说明。

3.1 输入输出数据的定义

上位机输入数据也就是 PLC 的输出数据包，通过上位机发送取参数命令来获取相应的输出数据包，因此在程序中采用定时处理事件 void CPLCController-Dlg::OnTimer(UINT nIDEvent)，定时地向 PLC 发送取参数命令来获取相关数据(即按要求设定采样周期定时采样)。

向 PLC 发送数据是以包的形式规定的，计算机每次发送一个 33 字节长的指令来实现一次读、写操作，当从上位机发出一个命令时，按命令帧格式准备数据(命令帧格式见图 3)。PLC 在接到上位机指令后，将发送一个 21 字长的反馈信息(响应帧格式见图 4)。

起始字符 B0	指令类型 B1	PLC 地址 B2 – B3	目标寄存器地址 B4 – B11	读写字节数 B12 – B13	数据区 B14 – B29	BCC 校验码 B30 – B31	结束字符 B32

图 4　PLC 命令帧格式

起始字符 B0	状态信息 B1	数据区 B2 – B17	BCC 校验码 B18 – B19	结束字符 B20

图 5　从 PLC 返回的响应帧格式

数据区不能够容纳全部的采样数据(包括温度、PH 值、溶解氧、搅拌转速、时间等)，因此规定每个响应帧只发送一个参数的数据，对每个发酵罐一次采样将向 PLC 发送一组采样命令，同时返回一组响应帧。因此把数

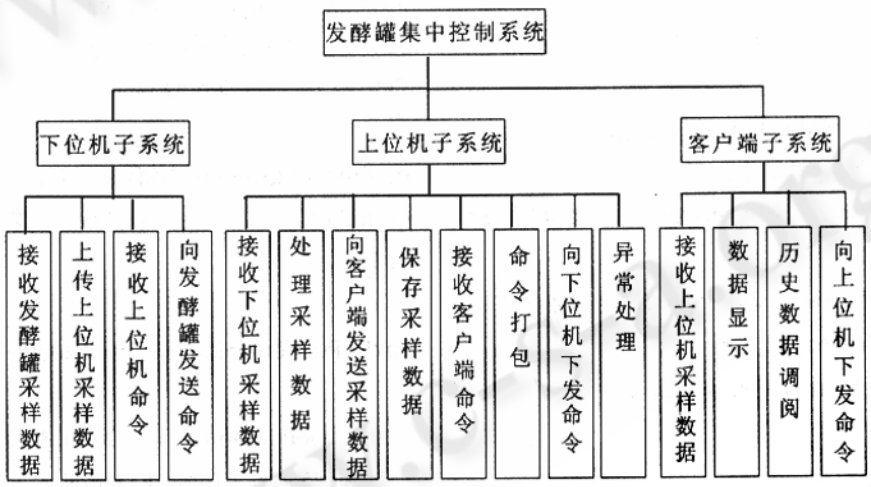

图 3　软件层次结构

原有输入输出数据包的数据定义不能满足程序的需要，根据系统的具体情况对原有数据包的数据做了一定的改动。首先，系统可向多台发酵罐同时采样，为了识别是哪一台 PLC 所发的数据包，我们对 PLC 的响应帧的起始字符作了改动。原来命令帧与响应帧都是以“g”作为开始符的，我们采用 ASC II 码 21H – 2EH 和 3AH – 3EH 这二十个符号来作为响应帧的起始字符，以区分来自 20 个发酵罐的输出报告。其次，对于发酵罐的一次采样，一个响应帧十六字节(B2 – B17)据区 B10 定义为参数标示位，以区分响应的是那个参数。B11 – B14 位定义为数据整数位，B16 – B17 定义为数据小数位。这样可根据试验员的需要随意设定响应参数的个数，使实验更方便灵活。

3.2 采样数据的处理方式

一方面由于采样数据比较复杂，对于发酵罐的每次参数(包括温度、PH 值、溶解氧、搅拌转速、时间等)的提取，都要向 PLC 发送一组采样命令同时 PLC 返回一组响应帧。另一方面可并发的接收来自多台 PLC 数

据包,同时也可向多台 PLC 发送指令。

针对以上采样数据的特点,我们对输入输出数据采取相应的处理策略。对于接受数据包的处理相对较为简单,串口通信编成,VC + +开发通常采用 Microsoft Visual C + +的通信控件(MSComm)。通过 MSComm 控件 OnComm 事件处理响应帧数据。当 MSComm 控件接收到数据包时,将生成一个接受报告(程序中我们定义为 CRecData 类,通过 CRecData 类对响应帧进行解包,生成所有相关数据,并对数据包及包数据出现的异常进行处理)实例来处理相应的接受报告。

```
void CPLCControllerDlg::OnCommctrl()
{
    VARIANT variant_inp;
    COleSafeArray safearray_inp;
    LONG len,k;
    BYTE rxdata[512]; //设置 BYTE 数组
    switch(m_ComPort.GetCommEvent())
    {
      case 1: // comEvSend 发送数据
          break;
      case 2: // comEvReceive 读取数据
          Sleep(50);//接受时延
          variant_inp = m_ComPort.GetInput();//
读缓冲区
          safearray_inp = variant_inp;//VARIANT 型
变量转换为 ColeSafeArray 型变量
          len = safearray_inp.GetOneDimSize();//
得到有效数据长度
          // 接受数据
          for(k=0; k<len; k++)
            safearray_inp.GetElement(&k,rxdata +
k); //转换为 BYTE 型数组
          m_pRecData = new CRecData(rxdata,
len); //生成接受报告实例
          treatRecData(m_pRecData) ;//处理接受
报告
          delete m_pRecData ;//清除接受报告实例
          m_pRecData = NULL ;
          break;
      default: // 传输事件出错
          m_ComPort.SetOutBufferCount(0);//清
空缓冲区数据
          break;
    }
    m_ComPort.SetInBufferCount(0);//清空缓冲区
数据
} 发送命令数据的处理相对要复杂的多,一方面可同时向不同 PLC 发送命令,另一方面每次向一台 PLC 发送的是一组命令。首先在程序中我们建立专门的线程来处理指令流。

UINT CPLCControllerDlg::runThreadProc(void * p)
{
    ((CPLCControllerDlg *)p) -> treatQueue();//
处理指令流
    return 0;
}
```

其次分 两步来处理指令流,第一步当定时采样要向某台 PLC 发出请求时,先将要请求的 PLC 进行排队,按先进先出的原则依次处理。

```
CPtrList m_autoSendQueue ; //定义 PLC 采样队列
.....
m_autoSendQueue.AddTail( new CQueueMember(PL-
Cnumber) ) ; //通过 ontimer()事件将 PLC 加入到指
令队列
.....
void CPLCControllerDlg::treatQueue()//处理 PLC 队列
{
    CQueueMember * pQueueMember = NULL ;
    while (m_nThreadController == 1)
    {
      if (! m_autoSendQueue.IsEmpty())
      {
        pQueueMember = (CQueueMember *)m_
autoSendQueue.GetHead() ;//取队列的头部
        treatPLCCommandFloat(pQueueMember ->
m_nQueueMember);//处理向某一台 PLC 发送的命
令流
        m_autoSendQueue.RemoveHead();//清除
队列的头部
        delete pQueueMember ;
```

```
            pQueueMember = NULL ;
        }
    }
}
```

第二步对当前将要发往某一 PLC 的指令组，再次形成指令队列，逐条向串口发送数据，每个发往 PLC 的指令组都有自己的队列用以区分和修改指令参数。

```
CPtrList m_autoSendList[20] ;//按 PLC 号定义 20 个 PCL 发送指令队列
.....
m_autoSendList[PLCNum].AddTail( new CSendData(num+1,'r','v',"07D0",4)) ;//向发往某一台 PLC 的指令队列中压入指令
.....
//处理发往某一台 PLC 指令队列
void CPLCControllerDlg::treatPLCCommandFloat(int PLCNum)
{
    CSendData *pSendData = NULL ;
    POSITION pos;
    //依次调用指令队列中的指令
    for( pos = m_autoSendList[PLCNum-1].GetHeadPosition(); pos != NULL; )
    {
      pSendData = (CSendData *)(m_autoSendList[PLCNum-1].GetNext(pos)) ;//逐条取指令
      OnCommSend(pSendData);//逐条发送指令
      pSendData = NULL ;
    }
    }
```

3.3 异常处理

异常处理是本系统至关重要的一部分，当程序出现异常情况时能够及时的处理，并给出报警和正确的提示信息。

我们将程序异常分为两大类：一是系统异常，它包括本身系统程序的问题、数据库连接、串口通讯、日志文件是否打开等异常情况。二是报告异常，它包括报告不全、未收到报告、接受数据错误、校验错误、状态异常等。

为了保证程序安全可靠的执行，我们采取以下几种处理方式：

(1) 记录和显示日志文件。实时的显示系统所做的各项动作，为系统管理员提供良好的监视环境，同时可调阅系统发生异常历史记录，为编程人员完善系统提供依据。

(2) 实时报警。以声音和异常信息显示为系统管理员提供当前系统所发生的异常情况。

(3) 实时异常处理。是指系统对某些异常(主要是报告异常)具有一定的处理能力，例如，当出现'未收到报告'异常时，系统会自动再发送一次采样命令等等。

4 结束语

vc++软件. 它是基于 Windows 操作系统。有丰富的人机界面，实时性强，再加上功能强大的 SQLServer2000 数据库，能方便的将采集的数据实时的提供给客户端显示。利用 Vc++软件作为计算机上位机编程软件，通过 PLC 与上位机通讯. 再通过网络将采样数据发送给客户机. 并给予图形显示，构成发酵罐集中控制系统。该系统不仅可以使实验室工作人员同时控制多台发酵罐，达到缩短实验周期的目的，同时通过客户端的实时图形曲线的分析和对发酵罐的实时命令控制，使实验的过程具有可控性，改掉了以往采样数据单一，实验控制性差的问题，提高了实验结果正确率。获得实验室工作人员的好评。

参考文献

1 西门子 S7—200 用户指南 fZ1

2 Kruglinski David J. Visual c++技术内幕【第四版)[M]，北京：清华大学出版社. 1999.

3 Wendy B., Michacl B., UML with Rational Rose 从入门到精通[M]，北京：电子工业出版社. 2000.