

## 【信息与计算机】

# 基于 VC ++ 的飞机电源控制器 试验台软件设计\*

刘 晔, 杨新伟, 张 璐, 惠培智, 王 斌, 师 丹, 孙培培

(西安交通大学 电气工程学院, 西安 710049)

**摘要:** 飞机电源控制器试验台用于对几种某型飞机电源控制器和控制盒在手动或自动方式下进行性能检测, 系统人机对话简单, 操作使用方便, 具有数据库管理和打印测试结果功能. 试验台软件系统采用 VC ++ 6.0 编程, 系统按照规定的测试流程进行试验, 同时将测试结果进行数据库存储, 还可实现数据报表的打印, 试验过程中需要与其他设备进行通信.

**关键词:** 电源控制器; 串口通信; 数据库; 打印

**中图分类号:** TP311

**文献标识码:** A

**文章编号:** 1006-0707(2010)02-0111-03

电源控制器(PCU)在飞机供电系统(EPS)的控制保护中居于核心地位, PCU 负责对飞机供电系统进行监控和保护, 并将供电系统的工作状态信息和 PCU 本身的工作状态信息传递给非航空电子处理机(NAMP). 电源控制器和电源控制盒是否正常工作关乎整个飞机供电系统的性能. 传统的试验台采用完全手动方式, 操作复杂, 效率低下; 本系统首次采用计算机控制的全自动方式设计, 用于对几种飞机电源控制器和控制盒的控制功能、监控功能和技术状态进行性能检查和试验. 软件系统采用 VC ++ 6.0 编写, 具有友好的人机交互界面和方便实用的测试流程, 设计中充分考虑到易用性, 即使测试人员对其功能不了解, 也能在人性化的引导下完成整个测试过程.

## 1 测试系统总体设计<sup>[1-3]</sup>

测试系统总体软件流程如图 1 所示. 系统首先进入登录界面, 输入密码登录成功后进行自测试, 即对系统本身的硬件、各种电源、继电器和各种接口进行检查, 以保证实验台没有故障, 可以工作正常; 条件满足后即可选择所要测试的产品然后进入该产品主测试界面. 如果测试条件具备则按照系统测试流程顺序执行, 执行过程中将试验结果

实时显示在主测试界面上, 同时将所需数据存入数据库以供打印分析.

## 2 数据库管理设计<sup>[4]</sup>

数据库采用微软的 Access 数据库, Access 数据库是目

前应用广泛的数据库之一. 软件使用 MFC 自带的 ODBC 类实现数据库管理设计, 可以方便的创建 Access 数据库, 新建数据表, 设计数据表的结构. 设计的数据库管理界面如图 2 所示. 数据显示区可以将数据库结构和内容显示出来, 显示项目包括: 编号、测试日期、产品型号、产品序号、测试人、试验编号以及所有测试参数等, 数据库操作有数据查询浏览、添加、修改、删除; 左下角数据查询可以通过修理日期、件号或飞机架次来查找相应相关的数据记录; 右下角数据管理可以删除一条或者所有记录.

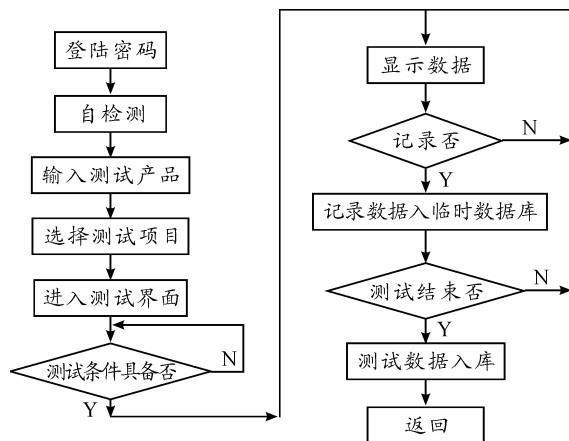


图 1 系统总体软件流程

当按产品序号查询时, 按如下代码实现.

```

CString m_SearchObject2 = "产品序号"; SearchCondi-
tions = m_SearchObject2 + " = " + sqlsearch2 + " "; //生成
查询语句
  
```

\* 收稿日期: 2009-12-28

作者简介: 刘晔(1963—), 男, 博士, 副教授, 主要从事测控技术与自动化、光纤传感技术、智能信息融合和航空电源技术等方面的研究.

```

m_Dataset73 -> m_strFilter = SearchConditions; // 查询
m_Dataset73 -> Open();
RecCount = m_Dataset73 -> GetRecordCount();
m_73display. DeleteAllItems();
m_Dataset73 -> MoveFirst();
CString str;
int hangj = liei = 0;
while( m_Dataset73 -> IsEOF() = 0) // 显示查询得
到的记录
{
    m_73display. InsertItem( hangj, "" );
    for( liei = 0; liei < int( m_Dataset73 -> m_nFields );
    liei ++ )
    {
        m_Dataset73 -> GetFieldValue( ( short) liei, str );
        m_73display. SetItemText( hangj, liei, str );
    }
    hangj ++ ;
    m_Dataset73 -> MoveNext();
}
m_Dataset73 -> Close();
删除所有记录, 实现代码:
if( m_Dataset73 -> IsOpen() )
m_Dataset73 -> Close();
m_Dataset73 -> MoveFirst();
while( m_Dataset73 -> IsEOF() = 0 )
{
    m_Dataset73 -> Delete();
    m_Dataset73 -> MoveNext();
}
m_Dataset73 -> Requery();
m_73display. DeleteAllItems();
m_Dataset73 -> Close();

```



图2 数据库管理界面

### 3 试验报告打印功能<sup>[5-7]</sup>

打印是 Microsoft Windows 程序设计中较难的一部分, 而 MFC 基本类库应用程序框架大大简化了打印的工作, 而且还提供了打印预览的功能机制, 正如显示有一个显示设备描述表一样, 打印机也要有一个打印设备描述表。当使

用图形函数时, MFC 使用统一代码在显示设备描述表上和打印机设备表上绘画, 这种两重性是通过 CView::OnDraw() 函数实现的, 主框架将创建打印机打印设备描述表并在打印完成后删除它, 调用 StartDoc 和 EndDoc 来开始和结束打印工作, 还为每一页都用 StartPage 和 EndPage 将 GDI 调用包括起来。

在文档/视图应用程序中实现打印功能的关键是一组 CView 虚函数, 主框架要在打印处理的不同阶段调用它们。主要 CView 打印函数如表 1 所示。

表 1 主要 CView 打印覆盖函数说明

函数	说明
OnPreparePrinting	在设置打印参数时调用。覆盖时可以用来调用 OnPreparePrinting 并给主框架提供打印页数(如果知道)和其他与打印有关的信息。
OnBeginPrinting	在打印开始前调用。覆盖时可以用来分配字体和其他打印需要的资源。
OnPrepareDC	在每一页被打前调用。覆盖时可以用来指定视口原点位置并在 OnDraw 打印下页之前设置剪贴区域。
OnPrint	在每一页被打之前调用。覆盖时可以用来打印页眉、页脚以及其它没有用 OnDraw 函数或不依赖 OnDraw 打印的页元素。
OnEndPrinting	在打印结束后调用。覆盖时可以用来解放在 OnBeginPrinting 中的分配资源。

框架还默认实现了打印预览功能, 并使得在应用程序中使用这一功能变得简单。打印预览将显示文档的一页或两页在打印时将出现的图像。这种实现方法还提供了一些控制, 如用于打印被显示的页, 移动带下一页或者前一页, 在实现一页和显示两页之间的切换, 缩放显示以便以不同的大小察看显示内容, 以及关闭显示等。要实现打印预览不是直接在一个设备上绘制图像, 框架必须用屏幕模拟打印机。为了做到这一点, MFC 实现了 CPreviewDC 类与 CPreviewView 类的联合使用。所有的 CDC 对象都有 2 个设备表。在 PreviewDC 对象中, 第 1 个设备描述表示的是被模拟的打印机, 第 2 个表示的是输出实际在其上显示的屏幕。

设计的打印预览如图 3 所示。

试验台实现了从数据库中取出所需数据然后打印在报表相应位置, 在 OnDraw 函数中添加代码如下:

```

m_Data71. m_strFilter = m_SearchObject;
// 选出所需数据
m_Data71. Open();
m_Data71. MoveFirst();

```

```
ppDC -> TextOut(1750, 1425, m_Data71. m_canshu1);
m_Data71. Close();
```

打印预览时可以上下翻页,实现代码如下。

```
CPrintFrame71 * pf = (CPrintFrame71 *); AfxGetMainWnd();
pvpagenum --;
if(pvpagenum < 0)
pvpagenum = 6; //超过最大页数时回到第一页
new CPrintFrame71(pf -> m_pOldWnd);
pf -> DestroyWindow();
CPrintFrame71 * pf1 = (CPrintFrame71 *); AfxGetMainWnd();
pf1 -> m_pOldWnd -> ShowWindow(SW_HIDE);
```

## 4 通过计算机串口与设备通信

系统中需要 400Hz/115V 电源,采用 AC400 系列电源。该电源通过工控机串口来控制,通信协议在电源说明书中。所以应在 VC++ 中通过串口根据提供的通信协议来控制电源,使之输出正确的电压。

程序采用 VC++ 自带的串口控件 CMSComm 类。

AC400 系列电源通信协议的通信波特率:57 600 b/s,故串口初始化程序如下:

```
m_ctrlComm. SetCommPort(1); //选择 com1
if(! m_ctrlComm. GetPortOpen())
m_ctrlComm. SetPortOpen(TRUE); //打开串口
else
AfxMessageBox("cannot open serial port");
m_ctrlComm. SetSettings("57600,n,8,1");
//波特率 57 600,无校验,8 个数据位,1 个停止位
m_ctrlComm. SetInputMode(1); //表示以二进制方式
检取数据
m_ctrlComm. SetRThreshold(1);
m_ctrlComm. SetInputLen(0); //设置当前接收区数据
长度为 0
m_ctrlComm. GetInput(); //先预读缓冲区以清除残留
数据
```

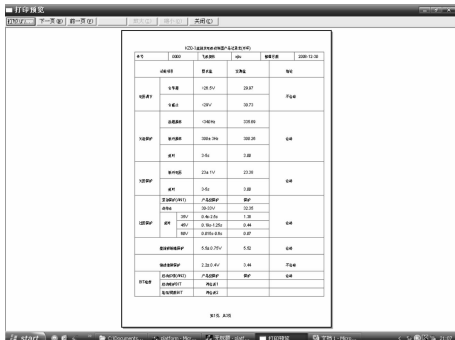


图3 打印预览

+ 地址码 + 帧长 + 功能码 + 数据信息 + 校验和。

其中,帧头、帧长、功能码、校验和用十六进制数表示,地址码、数据信息用组合 BCD 码表示。

校验和为所有帧的总和,用一个字节表示,和的溢出部分自动舍去。

校验和 = 帧头 + 地址码 + 帧长 + 功能码 + 数据信息。

帧长 = 功能码字节数 + 数据信息字节数 + 校验和字节数。

专门编写函数实现信息帧编码,具体代码如下。

```
change. Format("%0X", S_head);
m_strTXData += change; //帧头
m_strTXData += " ";
m_strTXData += "0";
change. Format("%0X", S_addr);
m_strTXData += change; //地址码
sum = S_head + S_addr + S_length + 0xA4 + 0xA5 + S_data1 + S_data2 + S_data3 + S_data4 + S_data5 + S_data6;
S_jiao = (int)(sum%256);
if(S_jiao < 10)
m_strTXData += "0";
change. Format("%0X", S_jiao);
m_strTXData += change; //校验码
根据三相电源通信协议,启动信息帧为:3A 01 0C A0
A1 01 40 00 11 50 11 50 11 50 EC
启动代码设计如下:
m_strTXData. Empty();
m_strTXData = "3A 01 0C A0 A1 01 40 00 11 50 11 50
11 50 EC";
CByteArray hexdata;
String2Hex(m_strTXData, hexdata);
m_ctrlComm. SetOutput(COleVariant(hexdata)); //发送
十六进制数据
同理,停机代码设计如下:
m_strTXData. Empty();
m_strTXData = "3A 01 03 A2 A3 83";
CByteArray hexdata;
m_ctrlComm. SetOutput(COleVariant(hexdata)); //发送
十六进制数据
```

实际应用中成功的实现了调压功能,如图 4 所示。

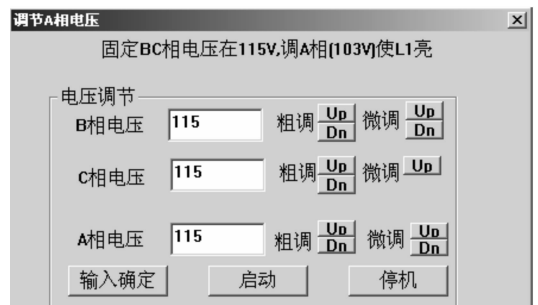


图4 通过串口与设备通信

AC400 系列电源通信协议的信息帧格式为:帧头(3A)

(下转第 125 页)