PALM 掌上电脑在无线抄表系统中的应用

田海山 郝晓军 河北工业大学计算机学院 (300130)
张永革 丛汲泉 船舶工业总公司七零七研究所(300130)

Abstract

Palmtop Computer has drawn more and more attention of people for its efficiency and facility,so it has been used in every walk of life.The system make use of some technology such as serial communication,infrared communication,and database's mobile store to complete the palmtop computer's application in the system of wireless read water meter.

Keywords:palmtop computer,serial communication,infrared communication,mobile store,database,wireless read water meter

摘 要

掌上电脑以其高效、灵巧而受到越来越多的关注,并已经应用到各行各业。本系统利用串口通讯技术、红外通讯技术与数据库的移动存储技术实现了 PALM 在自来水公司的无线抄表系统中的应用。

关键词:掌上电脑(PDA),PALM,串口通信,红外通讯,移动存储、数据库,无线抄表

0 引言

目前,多数自来水公司主要采用人工入户抄表。为了避免人为操作误差、保证抄表的及时顺利进行,对现有抄表系统进行改造、实行无线抄表、移动存储、无纸办公是十分必要的。随着计算机技术和网络技术的发展, 无线抄表可以实现。本系统在设计上采用表头模块、系统硬件、应用软件三层的体系结构(图 1)。本文主要涉及抄表人员在工作中用到的抄表工具为 PALM 掌上电脑 (也有称之为 PDA-Personal Digital Assistant)在无线抄表系统中的应用。针对使用环境是流动性强,作业时间长,对掌上电脑的灵巧性及电力供应要求都严格的环境,所以我们采用 PALM 公司生产的 PALM IIIe。系统程序特点有:1) 实现了移动存储;2) 具有集成化、智能化、网络化的特点,可靠性高;3) 操作维护简便,工作相对独立;4) 模块化结构,中文视窗界面,人机对话简单易学;5) 系统可扩展性强,无连接操作。

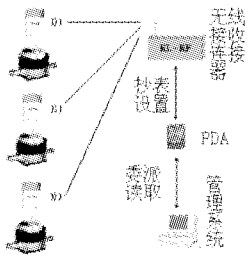

图 1 无线抄表系统结构

1 系统的组成

软件系统按功能模块化的方法进行设计,系统实现分三个层次,分别是数据的组织与管理层、系统控制管理层、与 RL、ET 及 PC 进行通信的通信层。通信层包括了通信链路的建立、数据链路层通信协议的实现、表示层的具体实现;数据库管理完成数据的建立、排序、检索与维护,实现了真正的移动存储;系统控制管理层负责程序事件的捕获、事件处理的分配、系统事件处理、菜单事件处理、应用事件派遣、窗体事件处理。通信层采用串口与红外两种通信方式,串口通信方式采用 RS232 串行总线技术,全双工,传送波特率为 57600bps。红外通信方式遵循 IRDA 的分层协议,最底层采用 SIR(Serial IR)协议,支持最高传输速率 115K, 通过 IrLAP (IR Link Access Protocol)与 IrLMP(IR Link Management Protocol)的协同工作,上层采用 OBEX(OBject EXchange),系统架构见图 2。开发工具采用了 Metrowerks 公司的 CodeWarrior 开发工具,编写了基于 Palm OS 3.1/3.5 平台的应用软件,具有控制、数据处理、人机交互、网络通讯等功能。另外,软件系统设计采用了冗余技术,便于扩展。

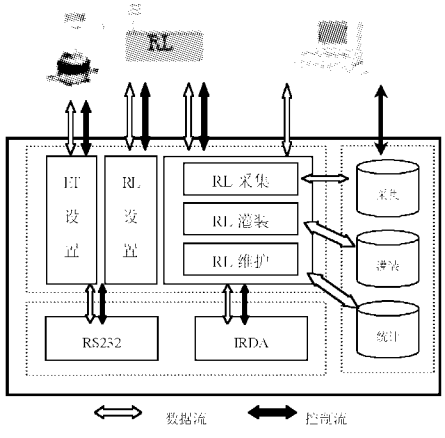

图 2 系统通信结构

2 系统的功能

1) ET 设置:ET 初始参数设置,ET 自检,发射测

试包,开、关发射振荡器。

2) RL 设置:RL 现场设置,ET 浏览及设置,RA 浏览及设置,读取 RA 接收包计数,读取 RA 接收 ET 统计信息,在线帮助。

3) RL 数据交互:RL 数据初始灌装,RL 数据采集,RL 数据上传,RL 维护,数据库维护。

3 Palm OS 串行传输

Palm OS 串行端口支持每秒 300 到 115200 字节的串行传输。Palm OS 在软件方面有以下几个层面共同组成它的串行传输系统。每个层面都是建立在下一层面之上并且添加一些功能。以下的层面组成了 Palm OS 串行传输堆栈,如图 3。

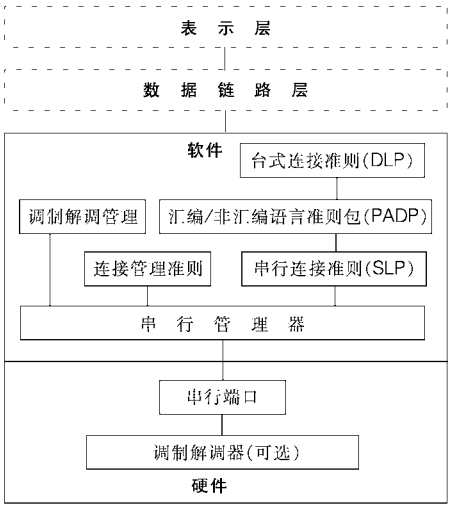

图 3 组成 Palm OS 串行传输栈的层面

1) 串行管理器:在最底层,串行管理器直接控制 RS-232 信号和硬件串行端口。这个层面支持字节平面的输入和输出,使它对用户应用程序来说更灵活。

2) 调制解调管理器:直接建立在串行管理器上,调制解调管理器为拨号和控制提供一个较小的 API,可以处理直接或通过一个掌上调制解调器电缆上的串行端口。

3) 串行连接协议(SLP):也建立在串行管理器上,这个协议为数据包提供一个有效的发送和接收系统,包括 CRC-16 的检查。Palm OS 还提供一个 API 让应用程序通过串行连接管理器(SLM)来调用 SLP,SLP 是一个类似 socket 结构的应用,支持远距离高度和远距离程序调用(RPC)

4) 汇编和非汇编协议包(PADP):建立在串行连接协议的基础上,PADP 为台式连接协议提供缓存数据传输能力,如以下所示。PADP 完全是内部层面的,应用程序不能使用它。

5) 台式电脑连接协议(DLP):建立在 PADP 层面上,可以实现远程使用各种 Palm OS 子系统,包括数据存储。HotSync 技术使用 DLP 进行同步操作安装和支持数据库。如果用户不能通过 Palm OS 应用程序使用 DLP 的功能时,通过为台式电脑写一个 HotSync 导管间接使用它。

6) 连接管理协议(CMP):是直接建立在串行管理器上,是系统的另一个协议,可以用外部传输软件来调整。

·通信链路的建立

```
//重置串行端口线段误差状态
Comm_ClearError ();
// 返回一个已经打开的库的参考号
errError = SysLibFind( "Serial Library", & serRefNum );
if (errError)    return    false;
//获得一个串行端口并用指定的波特率和缺省设置将其打开
errError = SerOpen( serRefNum, 0/ * Port,must be 0 * /,
SerialBaud/ * Baud * / );
Comm_ClearError ();
```

·通信链路的撤消

```
Comm_ClearError ();
//替换当前的接收队列,恢复初始缓存为 0
SerSetReceiveBuffer( serRefNum, 0, 0 );
// 释放一个串行端口
SerClose( serRefNum );
```

·数据的发送

```
//计算 CRC 校验码
CRCCode=Crc16CalcBlock(SendStrings,StrLen((const_char
 * )psContent),0);
 Packet(SendStrings, setData);
Comm_ClearError();
//通过串口发送数据
ulSended = SerSend(serRefNum, psTarget, ulLength, &er-
rError);
if (errError)    {// 握手超时
    return    false;
    }
Comm_ClearError();
//一直等待直到串口发送缓冲为空.
if (ulSended != ulLength)
    errError= SerSendWait(serRefNum,lTimeoutTicks);
Comm_ClearError();
//等待直到接收队列中收到指定字节的数据
errError= SerReceiveWait(serRefNum,1,
SysTicksPerSecond() * 2);
Comm_ClearError();
```

·数据的接收

```
//返回当前的接收队列中的字节数
errError =
SerReceiveCheck(serRefNum,numBytesP);
//返回指定长度的数据, 当线段误差或超时错误发生时返回一
个错误代码
```

```
ulReceived = SerReceive (serRefNum, psTarget, ulLength,
lTimeoutTicks, &errError);
//解开数据包
UnPack(recData);
```

·表示层的实现：主要处理数据在通信过程中数据的表示与转化。

4 控制管理

先来理解 Palm 的程序结构。Palm OS 最先执行的代码是 Pilot Main 例程，结构如下所示：

```
DWord PilotMai(Word cmd,Ptr cmdPBP,Word launchFlags)
{
    return StarterPilotMain(cmd, cmdPBP, launchFlags);
}
```

第一个参数 cmd 是一条启动代码，这条代码告诉应用程序如何启动；第二个参数 cmdPBP 是个指向与启动代码有关的结构的指针；第三个参数提供与启动有关的额外信息。

```
static DWord StarterPilotMain (Word cmd,  Ptr cmdPBP,
Word launchFlags)
{
    Err error;
//检查 ROM 文件的版本是否符合要求
    errorRomVersionCompatible(ourMinVersion,launchFlags);
    if (error) return (error);
    switch (cmd)
        {
        case sysAppLaunchCmdNormalLaunch:
            error = AppStart();
            if (error)
                return error;
            FrmGotoForm(MainForm);
            AppEventLoop();
            AppStop();
            break;
        default:
            break;
        }
    return 0;
}
```

·应用程序的启动

```
static Err AppStart(void)
{
    //打开通讯端口
//打开数据库
.....
}
```

很多复杂的程序都在 AppStart 例程中执行数据库的初始化和用户参数的检索，这些行为要先于应用程序的其他行为而发生。

·应用程序的关闭

当 AppEventLoop 例程存在，并且应用程序即将关闭时，AppStop 例程就开始了。

```
static void AppStop(void)
{
   StarterPreferenceType prefs;
   Err          err;
    CommPort_Close();
     err = DmCloseDatabase(RLDB);
  .....
    //为保存参数设置保存状态标志，This data
    // 这些数据在热同步时进行备份
    PrefSetAppPreferences (appFileCreator,  appPrefID,
appPrefVersionNum,
          &prefs, sizeof (prefs), true);
}
```

在一般情况下，该例程的代码应该完成以下任务：在程序退出执行这行关闭数据库、保护用户参数，以及对其他相关任务的执行。

·事件的处理

在 StarterPilotMain 调用 AppStart 之后，执行权就移交给 AppEventLoop 了。

```
static void AppEventLoop(void)
{
EventType event;
Uint error;
do {
        //得到下一个有效事件
EvtGetEvent(&event, evtWaitForever);
if (! SysHandleEvent(&event))
if (! MenuHandleEvent(0, &event, &error))
if (! AppHandleEvent(&event))
FrmDispatchEvent(&event);
} while (event.eType != appStopEvent);
}
```

事件循环的作用是处理应用程序接收到的事件。所接收到的事件进入事件循环，在循环中由 AppEventLoop 例程来一个个地处理事件。AppEventLoop 例程通过函数 EvtGetEvent 来从事件队列中捕获事件，然后将这些事件分派给事件处理程序。

第一个从事件循环值接收到事件的事件处理器是 SysHandleEvent，它使得 OS 能够处理一些重要的系统事件。如果 SysHandleEvent 完整地处理所接收的事件，它就返回 true。

如果系统对某个事件的处理不感兴趣，MenuHandleEvent 就获得了处理该事件的机会。如果 MenuHan-

dleEvent 完整地处理所接收的事件，它就返回 true。

·菜单事件处理

```
static Boolean MainFormDoCommand(Word command)
{
    Boolean handled = false;
    switch (command)
        {
        case MainOptionsMenuItem1001:
.....
            handled = true;
            break;
.....
default:
    break;
        }
    return handled;
}
```

经过上面两层的筛选后剩下来的事件可能就是应用程序自身所感兴趣的事件了，由 AppHandleEvent 来处理，它的唯一目的是处理 frmLoadEvent 事件。它载入并激活视窗资源，除此还设置一个回调函数来作为当前活动视窗的事件处理器。如果 AppHandleEvent 完整地处理所接收的事件，它就返回 true。

·准备视窗

```
static Boolean AppHandleEvent( EventPtr eventP)
{
//变量定义
if (eventP->eType == frmLoadEvent)
{
// 调用窗体资源
formId = eventP->data.frmLoad.formID;
FrmSetActiveForm(frmP);
//设置回调函数来作为当前活动视窗的事件处理程序。
switch (formId)
{
case MainForm:
FrmSetEventHandler(frmP, MainFormHandleEvent);
break;
.....
    }
return true;
}
return false;
}
```

·处理视窗事件

FrmDispatchEvent 函数将应用程序中没有处理的那些事件传给 FrmHandleEvent 函数，该函数的功能是让系统对这些事件进行缺省处理，无返回值。

5 数据库管理

作为一个便携式信息存储器和显示设备，运行 Palm OS 的掌上电脑必须有能力存储大量的各种数据并且能允许用户迅速地调用这些数据。因为在 Palm OS 中所有的数据存储都发生在 RAM 中，所以在 Palm OS 中创建和维护永久性数据的空间的方式有别于台式电脑中的空间方式。

使用 Palm OS 的 API 功能，可以创建、删除一个数据库，也可以打开和关闭它，这很像台式电脑处理文件的方式。Palm OS 的数据库与台式电脑的文件之间最大的不同在于数据的读写。

·创建数据库：创建一个新的数据库的函数是 DdmCreateDatabase。

一般来说，应用程序应该在 AppStart 程序中检验其数据库是否存在。如果数据库不存在，AppStart 会提示 DMCreateDatabase 创建一个新的数据库。本系统在 AppStart 中创建了三个数据库：灌装数据库、采集数据库、统计数据库。

·数据库的打开：用 DmOpenDatabaseByTypeCreator 函数，可以按照所给的数据库类型和创建者 ID 打开一个数据库，如果成功返回一个参数表示打开了数据库。

·数据库的关闭：应用程序结束时，用 DmCloseDatabase 来关闭。应用程序的 AppStop 函数是紧接在程序中任何清除操作后面调用 DmCloseDatabse 的。

·删除数据库：要删除一个数据库及其所有记录从掌上电脑删除，调用 DmDeleteDatabase 函数。

6 结束语

此系统运行稳定，操作简单，实现了真正的不入户抄表。已经被天津、河北、山东、内蒙等 30 多家自来水公司采用，公司的工作效率大大提高。一名抄表员可以在一分钟内抄到 125 家住户的水表流量。下一步的工作将实施无连接方案，全面采用 IRDA 的快速红外(FIR)传输协议，其传输速率达到 4M/S。

参考文献

1 Lonnon R.Foster.Palm OS Programming Bible.IDG Books Worldwide, Inc.

2 Robert Mykland.PALM OS PROGRAMMING from the GROUD UP.McGraw-Hill Companies,Inc.

3 Palm OS SDK Reference.3Com Corporation,1999

4 Palm OS Programmer's Companion.3Com Corporation 1999

5 高传善.数据通信与计算机网络.高等教育出版社