Windows CE 环境下无线网卡的自动安装

杭州浙江大学信息与通信工程研究所(310027) 刘 丹 李式巨

摘 要：讨论了 Windows CE 环境下无线网卡的安装，并提出了一种在系统断电重启时自动安装无线网卡的解决方案。对 Windows CE 的系统定制、应用程序的开发及系统封装进行了介绍。

关键词：嵌入式系统 Windows CE 无线网卡 设备驱动

Windows CE 是为多种嵌入式系统和产品而设计的紧凑、高效、可升级的操作系统，并特别为有限的硬件资源设计了多线程、多任务和完全优先的操作系统环境。在无线通信领域有很大的应用前景。

本文重点讨论了 Windows CE 环境下无线网卡的安装，并提出了一种在系统断电重启时自动安装无线网卡的解决方案；还对 Windows CE 的系统定制、应用程序的开发及系统封装进行了介绍。由于 Windows CE 与 Windows 的同源和相似，决定了学习 Windows CE 的简易性，这样就缩短了应用者的开发周期。

1 Windows CE 简介

Windows CE 包括四大基本模块，它们提供了操作系统的关键特性，分别是：内核(Kernel)模块、对象存储(Object Store)模块、GWES(用户、应用程序和操作系统之间的图形用户界面)模块和通信(Communication)模块。图 1 为 Windows CE 的模块化结构图。

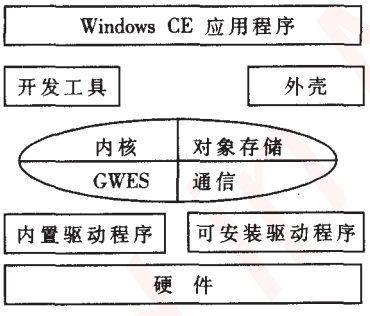

图 1 Windows CE 模块化结构图

操作系统定制工具有 Windows CE Platform Builder(简称 PB)，应用程序开发工具有嵌入式开发工具包 Embedded Visual Tools，包括 Embedded Visual C++(简称 EVC)和 Embedded Visual Basic(简称 EVB)。

2 特定硬件平台 Windows CE 操作系统的开发

有了具体的嵌入式硬件平台后，就可以为其开发满足特定功能需要的 Windows CE 系统。总之，可以分成三个步骤：操作系统的定制、特定功能应用程序模块的开发、功能模块封装入操作系统。

2.1 操作系统的初步定制

图 2 为在 PB 中定制 Windows CE 操作系统的一般流程[3]。

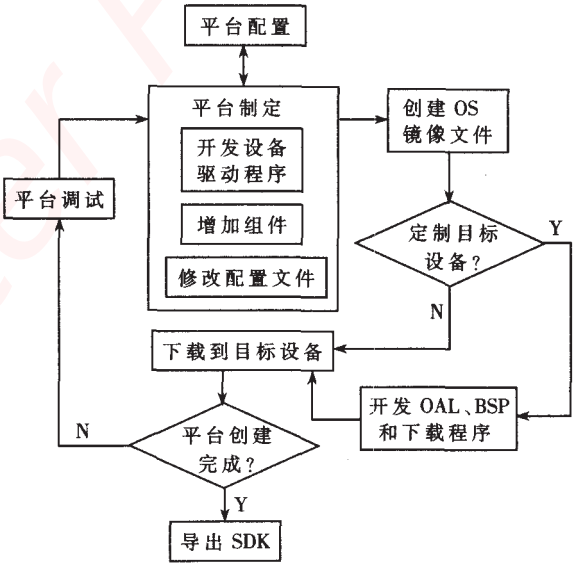

图 2 Windows CE 系统定制流程

首先，选择操作系统的基本配置，并且为特定的平台选择相应的微处理器和平台支持包 BSP (Board Support Packet)。PB 提供的 BSP 有 CEPC (基于 PC 机的 WinCE 硬件开发平台) 和 ODO (基于 Hitachi D9000 的 WinCE 硬件开发平台)。开发者可以选择其一或自己定制特定平台的 BSP。其次，制定平台，在此阶段可开发设备驱动，适当地裁剪、添加组件，如有必要还需对某些配置文件进行修改。然后，封装所需要的各功能模块，编译生成 OS 镜像文件。若设备为自己定制的目标设备，则需开发 OAL(OEM Adaptation Layer)、BSP 和 BootLoader。接着，把镜像文件下载到目标设备，进行调试；需要的话，进行重复配置、封装、下载及调试，直到满足要求，完成平台的创建。

最后导出相应的 SDK(Software Development Kit)软件开发工具包，运行后加载到 EVC 中，使得可以进行特定

硬件平台上的应用程序开发。SDK 包含程序库、头文件、示例程序源代码和库函数使用文档,同时还包括编程指导和 API 参考以及设备驱动工具包(DDK)。

2.2 特定功能应用程序模块的开发

开发特定功能的应用程序模块就是在 EVC 中编写应用程序,满足系统功能扩充的需要。重点要做以下工作:

(1)为特定平台选择相应的微处理器,如 WCE THUMB 或 WCE x86em 等。

(2)编写应用程序。方法上 EVC 与 PC 机上运行的 VC 差别不大。两者的区别主要有:前者的 API 是后者 API 的子集,对不兼容的函数要做一些修改或者寻求可替代的函数;前者为 Unicode 环境,所有字符都是两个字节,是一种世界范围的编码标准,有助于开发国际化软件,而后者为 ANSI 美国标准,每个字符一个字节,因此必要时两者间要作数据类型转换;另外在用户界面编程和内存管理等方面也有些差异。

(3)对程序进行编译和调试。编译时一定要选择特定的硬件平台。调试时可用微软提供的工具 Microsoft Activesync 建立 PC 机与目标机的连接,把系统镜像从 PC 机下载到目标机上进行调试;另外 EVC 中带有模拟器,可以在 PC 机上模拟目标平台上的大部分功能。

2.3 功能模块封装入系统

OEM 开发者通常把必要的应用程序和操作系统封装在一起发布给使用者,因此要把应用程序打包封装入初步制定的操作系统中。需要做的工作主要有两个:

(1)把编译好的可执行文件拷贝到 Windows CE 系统相应的文件夹中。

(2)在 PB 中修改相应的系统配置文件。PB 所提供的配置文件包括四种文件类型:.bib,说明需要打包进镜像文件的 Windows CE 文件;.dat,文件系统、目录和文件分配表描述;.db,Windows CE 对象存储数据库的描述;.reg,系统注册表。在开发过程中最常用到的配置文件有:Platform.bib、Platform.reg、Platform.dat、Config.bib。Platform.bib 定义打包到 OS 镜像文件时所需要的文件(files)和模块(modules);Platform.reg 定义目标平台冷启动时所加载的注册表键值;Platform.dat 定义目标平台冷启动时所加载的系统文件、目录和链接等;Config.bib 定义可用的物理地址,并进行一些属性设置。

做好以上两步以后,对操作系统重新编译、下载、调试,最终得到功能完整的系统镜像。

3 开发实例——Windows CE 环境下无线网卡的自动安装

3.1 项目介绍

项目要求实现一个独立的嵌入式无线通信模块,通过现有的 IEEE802.11b 无线网卡接入无线局域网进行通信。系统采用 Samsung 公司的 S3C2410 芯片开发嵌入式系统硬件平台,需要在 Windows CE 环境下驱动 PCMCIA 无线网卡。

3.2 无线网卡安装的问题和解决方案

Windows CE 操作系统支持两种类型的设备驱动程序:内置式设备驱动程序和可安装设备驱动程序。当目标机与 PC 机相连,且目标机有显示屏和键盘时,对于可安装的设备驱动程序,其安装步骤为:建立目标机与 PC 机的连接,将设备驱动程序的.dll 文件复制到 Windows CE 的 Windows 目录下;当系统检测到设备时,显示屏中会出现相应的对话框,用键盘输入驱动的名称即可。

项目中要为现有的无线网卡安装驱动程序。因为 Windows CE 系统运行时是基于 RAM 存储的,ROM 相当于只读硬盘,一旦系统断电或冷启动后,RAM 中的信息就会丢失,尤其是一些注册表的信息,所以要对 RAM 持续供电。但由于本系统硬件电源不能保证持续供电,安装了无线网卡的驱动程序后,信息存储在 RAM 中,系统断电或冷启动后,相关的信息会丢失。而且,目标机在实际使用中无显示屏和键盘。因此,断电重启时需要自动重新安装无线网卡驱动程序。

解决方案为:从系统持久存储 SM 卡中读取网卡驱动程序 xi825.dll 和 TCP/IP 属性配置文件 config.txt,并按配置文件设置注册表键值,然后为无线网卡安装驱动程序。若需要根据具体应用环境更新 TCP/IP 属性值,可用新的配置文件覆盖 SM 卡中的旧文件,冷启动后,重新设置属性值,再安装网卡驱动即可。

用 EVC 编写应用程序完成自动安装无线网卡驱动的功能,图 3 为应用程序的大体流程。

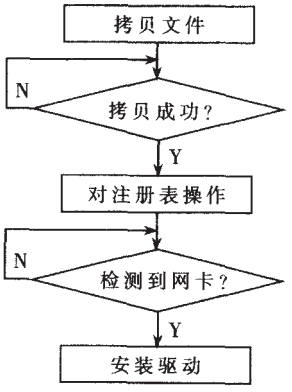

图 3 应用程序编写流程

首先,把两个文件从 SM 卡的 storage card 文件夹拷贝到 Windows CE 系统的 Windows 目录下;拷贝成功后,按 config.txt 的内容对注册表进行操作,设置具体环境下的 TCP/IP 属性值;系统检测到无线网卡后,在弹出名为"Unidentified PCCard Adapter" 的对话框中程序自动输入网卡驱动程序名,完成无线网卡的自动安装。

3.3 几个具体问题的解决

下面,再讨论一下本方案中几个具体问题的解决办法。包括对注册表的操作、自动安装功能的实现、程序流

程的总体控制以及将程序打包封装入操作系统的作法。

3.3.1 对注册表的操作

Windows CE 中有一系列 API 函数可以对注册表进行操作，完成打开关闭注册表、读取或修改键值等功能。另外，Windows CE 注册表中值的类型为 Unicode；而配置文件 config.txt 是在 PC 机上产生的，类型为 ANSI。要按 config.txt 中读取的值设定注册表值，就要进行数据类型的转换，可以用函数 MultiByteToWideChar( )实现[4]。关键代码如下：

```
RegOpenKeyEx(HKEY_LOCAL_MACHINE,        //根键
             TEXT("Comm\\xi8251\\Parms\\TcpIP"),
                                        //打开根键下的子键
             0,                         //预留值，必设为 0
             0,                         //不支持此项，必设为 0
             &hKey                      //最终打开键的句柄指针
             );
RegSetValueEx( hKey,                    //对键操作的句柄
             TEXT("IpAddress"),         //键中的数据项名
             0,                         //预留值，必设为 0
             REG_SZ,                    //数据项中值的类型
             (CONST BYTE*)((LPCTSTR)regData),
                                        //存有数据项值的缓冲区
             dwDataSize                 //值的字节数
             ) ;                        //对"IpAddress"数据项的
                                        值作修改，即改变 IP 值
```

3.3.2 自动安装功能的实现

用 FindWindow( )函数判断对话框窗口的出现；用 keybd_event( )函数模拟键盘输入。关键代码如下：

```
TCHAR g_szTitle[80]=TEXT("Unidentified PCCard Adapter");
                                        //指定对话框的标题
HWND  hWnd=::FindWindow(NULL, g_szTitle) ;
//判断此标题名的窗口是否已出现，不论其是否为前台
                                        窗口
if (hWnd! =NULL)                        //若窗口已出现
{::SetForegroundWindow(hWnd);           //将此窗口设置为前
                                        台窗口
  keybd_event(0x58,0,0,0);              //按下 x 键
  keybd_event(0x58,0,KEYEVENTF_KEYUP,0); //抬起 x 键
//完成了按下和抬起 x 键的两个动作，就模拟了键盘输
入字符 x，
  ……//用同样的方法输入 'i825.dll'
keybd_event(0x0d,0,0,0);
keybd_event(0x0d,0,KEYEVENTF_KEYUP,0); //最后输入
                                        确认键
}
```

正确输入后，可以看到无线网卡的显示灯开始闪烁，说明网卡已经安装成功，等待进行无线通信。

3.3.3 程序流程的总体控制

为了保证程序流程的顺序，整个程序中还需要有一个总体监视控制的管理员。这种监控功能，可以通过发送接收特定消息给主控函数的方法，也可以采用等待特定事件对象的方法，还可以采用开定时器查询的方法等。鉴于方法简单，而且对系统资源占用并不大，这里选用了开定时器查询的方法，并设置了标志位 flag 区分不同阶段工作，进行相应的操作。下面列出部分关键代码：

```
flag=0;                         //最初标志位设为 0，即先拷贝
                                所需要文件
m_nTimer =SetTimer(1,2000,NULL);    //开启定时器
KillTimer(m_nTimer);            //时间到，先关闭定时器，进行
                                相应判断和操作。
```

3.3.4 应用程序封装入系统

上文已经讲了将应用程序封装入操作系统的一般方法，分为应用程序的拷贝和修改系统配置文件两步。此项目最后要把开发的应用程序 monitor.exe 打包封装入操作系统。这里重点介绍一下系统中几个配置文件的具体修改情况。

(1)在 Platform.bib 文件中的 Files 部分加入以下代码

```
;  Name          Path                    Memory   Type
;  ……            ……                      ……       ……
   monitor.exe   $(_FLATRELEASEDIR)      NK       U
                 \ monitor.exe
   monitor.lnk   $(_FLATRELEASEDIR)      NK       U
                 \ monitor.lnk
```

其中，monitor.lnk 为 monitor.exe 的链接程序(快捷方式)。制作方法很简单，可以通过一个 ASCII 码编辑器编写，格式为：#20\windows\monitor.exe。

此配置文件中的代码表示，将 monitor.exe 和 monitor.lnk 加载到名为 NK 的内存区中(NK 的定义在 config.bib 文件中的 memory 部分完成)，文件的属性为 U(非压缩文件)，这样就完成了把自己所开发的应用程序及其链接程序封装进操作系统的“声明”。

(2)在 Platform.dat 文件中加入以下代码：

```
Directory ("\Windows\StartUp"): -File ("monitor.lnk","\windows\monitor.lnk")
```

由于 Platform.bib 做过声明，这两个文件在操作系统启动后会映射到 \Windows 目录下，这是默认的目录。所以此处的代码表示在 Windows CE 启动时会自动运行 monitor.exe。

做完了以上工作，再对操作系统进行编译，把应用程序封装到了 Windows CE 操作系统中，而且系统启动后会运行此应用程序，完成自动安装无线网卡的功能。

参考文献

1 微软公司.Microsoft Windows CE 程序员指南. 北京：北京大学出版社，2000

2 微软公司.Microsoft Windows CE 开发人员指南系列.北京：北京希望电子出版社，1999

3 Microsoft. Windows CE Platform Builder 3.0 Library.

4 Microsoft. Embedded Visual C++ 3.0 Library.

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用

VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用
14. 基于 MPC8270 的 VxWorks BSP 的移植
15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现
17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发

Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通

12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核
15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究
20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings

Windows CE：

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]
16. 基于 WinCE 的 BootLoader 研究

PowerPC：

1. Freescale MPC8536 开发板原理图

2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计
7. 用 PowerPC860 实现 FPGA 配置
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的μC-OSⅡ及 LwIP 协议栈的移植与实现

Hardware：

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用

6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究
13. 基于 UEFI 固件的恶意代码防范技术研究