第 32 卷 第 16 期 计 算 机 工 程 2006 年 8 月
Vol.32 № 16 Computer Engineering August 2006
·软件技术与数据库· 文章编号：1000—3428(2006)16—0041—03 文献标识码：A 中图分类号：TP311.52

Windows CE 设备驱动程序开发

胡军辉，王友钊
(浙江大学仪器系，杭州 310027)

摘 要：Windows CE 操作系统标准设计的一个重要方面，就是原始设备制造商(OEMs)和独立硬件开发商(IHVs)可以自主开发设备驱动程序来支持他们的硬件，而不需要微软公司去另外开发设备驱动程序。该文介绍了 Windows CE 体系结构和开发 wince 设备驱动程序的过程，阐述了设备驱动程序模型、设备驱动结构和中断处理。Windows CE 操作系统支持 4 种设备驱动模型：本机设备驱动，流接口设备驱动，USB 设备驱动，NDIS 网络驱动。

关键词：嵌入式操作系统；Windows CE ；设备驱动；中断处理；开发过程

Device Driver Development for Embedded Windows CE

HU Junhui, WANG Youzhao
(Instrument Department, Zhejiang University, Hangzhou 310027)

【Abstract】A key aspect of the modular design of Windows CE is that original equipment manufacturers (OEMs) and independent hardware vendors (IHVs) can implement device drivers that support their own hardware without additional development from Microsoft. This paper introduces the system structure of Windows CE and the process of developing device drivers for Windows CE. It provides an overview of device-driver architecture, elaborates the Windows CE device-driver models and interrupt handle. Windows CE supports the following four driver models: native device driver， stream interface device driver, universal serial bus (USB) device driver, network driver interface specification (NDIS) driver.

【Key words】Embedded operation system; Windows CE; Device driver; Interrupt handle; Developing process

Windows CE 是一个抢先式多任务并具有强大通信能力的 Windows32 嵌入式操作系统，是微软专门为信息设备、移动应用、消费类电子产品、嵌入式应用等非 PC 领域而从头设计的战略性操作系统产品。目前，随着以 PDA、信息家电、机顶盒等为代表的嵌入式系统应用的广泛发展，嵌入式系统已经越来越走近普通人的生活，随之而来的就是对嵌入式软、硬件设计的广泛需要。嵌入式软件开发，特别是嵌入式软件的驱动开发，成为一个热点。

1 Windows CE 的系统体系结构

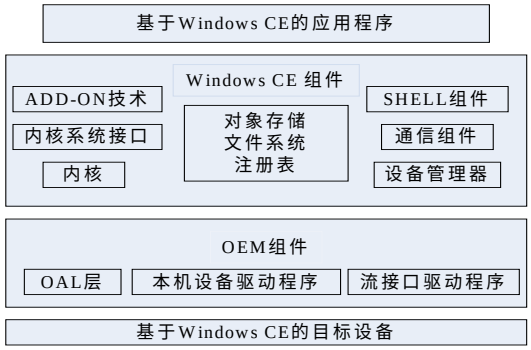

图 1 基于 Windows CE 的目标平台及其组件

Windows CE并不是Windows 桌面操作系统的一部分或缩减版本，而是微软全新设计开发的一种开放的、模块化的、可升级的抢先式多任务 32 位实时操作系统。一个基于Windows CE的平台主要由以下几部分组成：WindowsCE内核组件，OAL层和驱动程序。OAL层和驱动程序作为目标设备和WindowsCE内核组件之间的接口，将操作系统上层与硬件隔离，从而便于支持硬件扩展和即插即用功能[1]。图 1 详细描述了它们之间的关系。

2 Windows CE 的中断处理

就其中断处理而言，Windows CE 采用了一种独特的方法。它将中断处理分为两步：中断服务例程(ISR)和中断服务线程(IST)。具体来讲就是把每个硬件的设备中断请求(IRQ)和一个 ISR 联系起来，当一个中断发生并未被屏蔽时，内核调用该中断注册的 ISR。因为 ISR 运行于内核模式，所以应该被设计得尽可能的短，ISR 的基本职责是引导内核调度和启动合适的 IST。IST 在设备驱动程序软件模块中编写，它从硬件获取或向硬件发送数据和控制代码，并进一步处理设备中断。Windows CE 还支持中断嵌套，即在一个 ISR 运行时，内核并不关闭中断。当优先级比之高的中断发生时，内核保存当前执行的中断服务例程 ISR 的运行状态，挂起该 ISR，转而执行更高优先级中断的 ISR。等到高优先级中断的 ISR 执行完后，被挂起的低优先级中断的 ISR 才重新开始被 CPU 调度执行。

Windows CE 的中断处理涉及到很多组件。异常处理器是处理中断的主要组件之一。通常系统启动时在异常处理器中注册各种 ISR，当有中断发生时，处理器将控制权移交给内核中的异常处理器。然后异常处理器调用该中断所对应的 ISR，该 ISR 的作用就是将这个中断“翻译”成一个逻辑中断标识符，并将其传递回内核。内核激活一个与该逻辑中断对应的事件对象，从而使 IST 得以被调度执行。图 2 详细描述

作者简介：胡军辉(1980—)，男，硕士生，主研方向：嵌入式系统；王友钊，教授

收稿日期：2005-10-08 **E-mail：**hujunhui1980@163.com

了它们之间的关系。

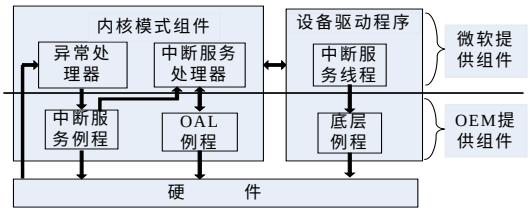

图 2 中断处理组件

3 设备驱动程序结构

Windows CE 设备驱动程序在结构上可分为单体结构和分层结构。也就是说设备驱动程序可以直接对设备所进行的操作来实现软件接口，也可以将软件接口和针对设备所进行的操作分散到两个层上。单体结构的设备驱动程序直接访问设备，把跟硬件设备相关的函数，也就是 DDI(Device Driver Interface)函数直接提供给操作系统。分层结构的设备驱动程序在结构上分为 MDD(model device driver)层和 PDD (platform-dependent driver)层，MDD 层是提供给操作系统调用的，MDD 层通过 DDSI(Device Driver Service Provider Interface)函数来调用 PDD 层，而 PDD 层是跟硬件设备相关的。

MDD 模型设备驱动程序执行下列任务:连接 PDD 层并定义它希望调用的函数；把不同的函数集提供给操作系统；处理像中断处理这样的复杂任务；与 GWES 模块和内核通信。依赖平台的 PDD 层与 MDD 和硬件都有接口，这就意味着必须适合目标平台的需要。PDD 有针对具体硬件的函数组成，而这些函数与 MDD 层相对应。但这种对应不是直接的一一对应。MDD 函数实现对应的任务，而 MDD 则通过使用这些任务来实现其目标。因为 PDD 层是硬件相关的，因此必须生成一个设置好的 PDD 并输出到平台硬件。微软为各种各样的内部设备提供了几个样本 PDD 层。

由微软提供的许多驱动程序实例都采用了分层结构，这样可以减少开发人员在将这些实例移植到别的新设备上的工作量。也可以不用 MDD 和 PDD 层，把自己的设备驱动程序实现为单片驱动程序。例如，如果性能是一个至关重要的因素，使用单片驱动程序就比使用分层的驱动程序好，因为单片驱动程序避免了与 MDD 和 PDD 层之间的函数调用有关的重叠过程。

如果硬件的性能恰好与 MDD 层中的函数所执行的任务相匹配，也可以选择实现单片驱动程序。在这种情况下，实现单片驱动程序可能要比实现分层的驱动程序简单且高效。但不管是实现单片驱动程序还是分层的驱动程序，都可以基于源代码实现样本分层的驱动程序。如果设备被直接映射到内存，设备驱动程序可以直接访问它们，否则设备驱动程序必须通过下一层设备驱动程序来访问它们的设备。

4 设备驱动程序模型

设备驱动程序是将操作系统和设备连接起来，使得操作系统能够识别设备，并为应用程序提供服务。Windows CE 支持广泛的基于各种 CE 平台的设备驱动程序，也提供一些用于驱动开发的模型，其中包括来自其他操作系统的驱动程序模型。因为有这些多变的驱动程序模型，使 Windows CE 可以适应大部分的内部和外围设备。目前，Windows CE 提供了 4 种设备模型，其中两种是专用于 WindowsCE 模型，另外两种外部模型来自其他的操作系统，进行汇总如图 3 所示。

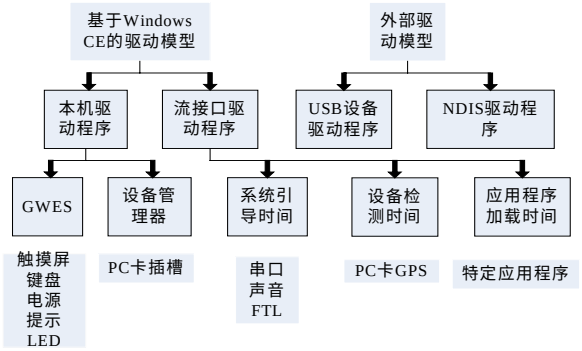

图 3 驱动程序模型

4.1 本机设备驱动程序

要把 Windows CE 移植到目标平台上，必须为在平台上已建立的设备提供驱动程序。一些类型的设备，如键盘、显示器和 PC 卡插槽等对操作系统都有一个自定义接口。因为这些接口是专门用于 Windows CE 的，所以这类驱动程序叫本机驱动程序。一般来说，只有 OEM 开发商对设备驱动程序感兴趣，他们建立基于 Windows CE 的平台。而独立硬件销售商只开发附加的硬件驱动程序。Windows CE 平台生成器提供本机设备驱动程序的样本，可考虑把本机设备驱动程序样本应用到自己的平台上，而不需要再从头开发自己的本机驱动程序。

一般来说，原始设备制造商都把本机设备驱动程序与图形、视窗和事件子系统(GWES)连接起来。

4.2 开发流接口驱动程序

流接口驱动程序是动态连接库[2]，是由一个叫做设备管理程序的特殊应用程序加载、管理和卸载，与具有单独目的接口的内部驱动程序相比，所有流接口驱动程序使用同一个接口并调用同一个函数集流接口函数。对于每个流接口驱动程序来说，其所要求的入口点用来实现标准文件I/O 函数和电源管理函数，这些函数由Windows CE 操作系统的内核使用。其所要求的入口点为: XXX-close，XXX-Deinit，XXX-Init，XXX-IOControl，XXX-Open，XXX-PowerDown，XXX-Power Up，XXX-Read，XXX-Seek，XXX-Write。当生成一个 DLL 后，就用设备文件名前缀替换入口点名字中的 XXX。

流接口驱动程序是为了连接到基于 Windows CE 的平台的外围设备而设计的。这些外围设备包括调制解调器、打印机、数字相机和 PC 卡。所有这些外围设备都必须通过外部连接器如串行口或 PC 卡插槽连接。因此，外围设备的驱动程序就像桌面计算机的打印机驱动程序一样，都当作用户模式的程序来运行，这些程序使用内置硬件的服务来控制它们的设备。流接口驱动程序的主要任务是把外设的使用传递给应用程序，这是通过把设备表示为文件系统的一个特殊文件实现的。流接口驱动程序一般是由独立硬件制造商(IHV)为外围备设设计的，但定制基于 Windows CE 的平台的 OEM 们却有时编写一些内部设备的流接口驱动程序。另外，尽管流接口驱动程序通常是由设备管理程序加载和卸载的，但有时应用程序也执行加载和卸载的任务。

流接口驱动从设备管理器和通过文件系统调用的应用程序来接受命令，驱动将这些命令翻译成它所控制设备的适当操作的所有信息。所有的流接口驱动，无论它管理的是内置式设备还是可安装式设备，是系统引导加载还是动态加载，它们和其它的系统组件都有类似的交互过程。

4.3 USB 驱动程序

USB 系统由计算机、一个或多个 USB 设备和物理总线组成。主机又分两层：包含 USB 设备驱动程序的软件层和主机控制器硬件层，也称适配层。主机的主要责任就是控制对 USB 设备的双向数据传输。USB 设备是使用数据格式规则与主机进行通信的外围设备。物理总线是一组 USB 电缆用来将控制器和外围设备连接起来。

USB 拓扑结构为树状总线结构，其结构如图 4。

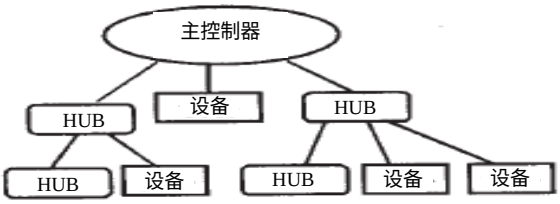

图 4 USB 拓扑结构

HUB 只有一个通往 USB 上层的接口，中继站最多有 7 个连接外围设备或其他中继站的下层接口。通过 HUB 连接到一起，最多可以有 127 个设备可以连接到主计算机上。

USB 系统有 4 种数据传输类型。

(1)控制传输：它是双向传输，被 USB 系统软件用来进行查询、配置和给 USB 设备发送通用的命令。

(2)同步传输：它提供了确定的间隔时间，被用于时间严格并具有容错性的流数据传输，或用于要求恒定的数据传输率的即时应用中。

(3)中断传输：它主要用于定时查询设备是否有中断数据需要传送。

(4)批量传输：它适用于需要大批量的传送和接收数据的设备，同时在没有带宽和间隔时间要求的情况下，要求保证传输。

USB 系统软件有两层组成：较高的 USB 设备驱动程序层和较低的由 Windows CE 实现的 USB 函数层。USB 设备驱动程序层使用 USB 函数来建立与其所控制设备的连接并对这些设备进行配置和通信。较低的 USB 函数层执行一些内部关联的任务：管理所有 USB 设备驱动程序和主机的内建 USB 根集成器之间的通信；在适当的时间加载和卸载 USB 设备驱动程序；进行从数据到 USB 协议框架和打包格式的双向转换；通过建立与所有 USB 设备上的通用端点的通信，执行一般的配置和与状态相关的任务。

较低的 USB 函数层又由两部分组成：较高的通用串行总线驱动程序(USBD)模块和主控制器驱动程序(HCD)。

图 5 说明了与主机的 USB 硬件和外围设备相对应的软件的各个层。

图 5 USB 硬件和外围设备对应的软件关系

Windows CE 没有提供 USB 设备使用的标准机制，但是编写 USBD 可供采用的方法有：(1)使用流接口函数；(2)使用现有的 Windows CE 应用程序编程接口(API)；(3)创建用户指定的 API。

4.4 NDIS 网络驱动程序

网络驱动程序规范(NDIS)是Windows CE支持网络连接的一种方法。NDIS提供了两个抽象层，用来把网络驱动程序与协议栈，例如TCP/IP和红外数据协会(IRDA)相连，或者与网络适配器，例如以太网卡相连。NDIS给网络驱动程序的编写者提供了两组应用程序接口(API)：一组是用于网络协议栈的，另一组是用于网络接口卡的[3]。

Windows CE2.0 和以后版本实现了在 Windows NT 上使用的 NDIS 4.0 模型的一个子集，使得 OEM 厂商和独立硬件厂商能够把已有的 Window NT 上的网络驱动程序移植到 Windows CE 上。完整的 NDIS 支持多种类型的网络驱动程序，但是 Windows CE 和以后版本只支持小端口卡的驱动程序，不支持统一的全部的 NIC 驱动程序。此外，以太网卡和 IRDA 是 Windows CE 所支持的唯一 NDIS 介质类型。

图 6 说明了在 Windows CE 中协议栈、NDIS、NIC 和小程序端口卡驱动之间的关系。

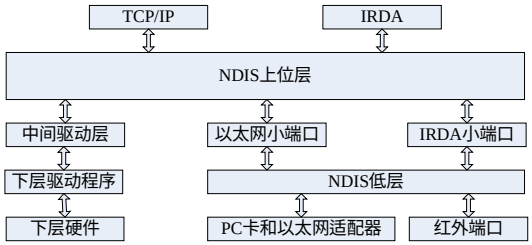

图 6 网络驱动程序

对于小端口卡驱动程序，Windows CE 与 Windows NT 具有很大程度的源代码兼容性。Windows CE 与 Windows NT 支持统一的 NDIS 应用程序接口。小端口卡驱动程序是一个复杂的软件，因此，微软推荐采用小端口卡的样本驱动程序进行改造或从其他操作系统移植现成的驱动程序，而不要从底层开发驱动程序。Windows CE 包括下面这些小端口卡的驱动程序：Proxim 无线以太网卡；FastIR；NE2000 兼容的网络适配器；IrSIR 红外串口中性小端口卡；Xircom CE2 以太网卡。

5 设备驱动程序开发过程

Windows CE 设备驱动程序的开发遵循以下步骤：

(1)确定设备驱动程序和设备之间的接口；

(2)确定设备驱动程序需要提供的 API 接口；

(3)确定设备驱动程序所要使用的其他接口；

(4)确定操作系统或应用程序所要使用的函数和设备所能实现的功能之间的映射关系；

(5)确定跟设备驱动程序相关的结构上的因素，例如，哪些函数有运行时间和内存使用上的限制；

(6)确定是否有可以从其他操作系统移植过来的或早期版本的源代码；

(7)完成剩余的源代码；

(8)测试和调试设备驱动程序。

6 结束语

文中简单介绍了 Windows CE 操作系统的体系结构和中断处理，详细阐述了 Windows CE 操作系统驱动程序的开发模型和体系结构。

希望文中的基于嵌入式操作系统 Windows CE 的驱动程序开发模型能够帮助读者开发出更多的嵌入式系统产品。

参考文献

1 Microsoft . 希望译. Microsoft Windows CE Device Driver Kit[M] . 北京: 北京希望电子出版社, 2000.

2 Goggin T A. 尤 滔, 张 平, 周晓权译. Windows CE 高级开发指南[M] . 北京: 电子工业出版社, 2001.

3 Microsoft . 希望译. Microsoft Windows CE Communications Guide[M]. 北京: 北京希望电子出版社, 2000.