

VxWorks开发人员指南丛书

WIND RIVER

VxWorks网络程序员指南

[美] Wind River 著

王金刚 宫霄霖 熊辉 译

6.2
2



清华大学出版社

VxWorks 开发人员指南丛书

VxWorks 网络程序员指南

[美] Wind River 著

王金刚 宫霄霖 熊 辉 译

清华大学出版社

北 京

内 容 简 介

本书是《VxWorks 开发人员指南丛书》的网络程序员开发指南分册, 根据 Wind River 公司的技术文档 VxWorks Network Programmer's Guide 翻译而成。

主要包括: VxWorks 网络编程概论、VxWorks 网络栈、数据链路层组件、网络栈配置、网络配置协议、动态路由协议、VxWorks 下的套接字、远程访问应用、DNS 和 SNTP、添加网络接口驱动程序、添加网络服务等, 从基本理论到各层协议的编程开发, 详细讲述基于 VxWorks 操作系统网络应用程序的开发。

本书语言通畅、条理清晰、内容详细, 主要针对从事以 VxWorks 操作系统为基础内核的嵌入式系统开发人员, 作为了解 VxWorks 并且将其应用到项目中的指导手册。

版权所有, 翻印必究。

本书封面贴有清华大学出版社激光防伪标签, 无标签者不得销售。

图书在版编目(CIP)数据

VxWorks 网络程序员指南/(美国)风河公司著; 王金刚, 宫霄霖, 熊辉译.

—北京: 清华大学出版社, 2003

(VxWorks 开发人员指南丛书)

ISBN 7-302-06823-2

I. V… II. ①风… ②王… ③宫… ④熊… III. 实时操作系统, VxWorks-指南 IV. TP316.2-62

中国版本图书馆 CIP 数据核字(2003)第 077554 号

出 版 者: 清华大学出版社

<http://www.tup.com.cn>

社 总 机: 010-62770175

地 址: 北京清华大学学研大厦

邮 编: 100084

客户服务: 010-62776969

组稿编辑: 曾 刚

文稿编辑: 庾 永

封面设计: 秦 铭

版式设计: 张红英

印 刷 者: 清华大学印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 185×260 印张: 17.25 字数: 390 千字

版 次: 2003 年 9 月第 1 版 2003 年 9 月第 1 次印刷

书 号: ISBN 7-302-06823-2/TP·5070

印 数: 1~5000

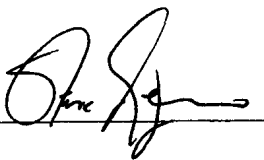
定 价: 24.00 元

授权声明

"Wind River Systems, Inc. does not endorse this publication, nor has Wind River reviewed its contents for accuracy. This publication is the sole creation of Tsing Hua University Press, which bears sole responsible for its contents. Wind River has granted the Tsing Hua University Press a limited right to use the marks VXWORKS, TORNADO, VXSIM, CROSS WIND, WIND SH, WINDVIEW, WIND and WIND RIVER for identification purposes only. This grant does not indicate that Wind River has endorsed the contents of the book. VXWORKS, TORNADO, VXSIM, CROSS WIND, WIND SH, WINDVIEW, WIND and WIND RIVER are trademarks and/or registered trademarks of Wind River Systems, Inc. and are used with permission in this publication."

Wind River Systems, Inc. 授权清华大学出版社编译出版本技术文档，由清华大学出版社完全负责该技术文档的编译、编辑、审校、出版；此出版物的中文版权为清华大学出版社单独拥有；清华大学出版社对此出版物内容完全负责。Wind River Systems, Inc. 已授权清华大学出版社在此出版物中使用其注册商标：VXWORKS、TORNADO、VXSIM、CROSS WIND、WIND SH、WINDVIEW、WIND 和 WIND RIVER。以上授权说明清华大学出版社承认并保证此出版物的内容。VXWORKS、TORNADO、VXSIM、CROSS WIND、WIND SH、WINDVIEW、WIND 和 WIND RIVER 等 Wind River Systems, Inc. 的商标和（或）注册商标在此出版物中的使用已得到了 Wind River Systems, Inc. 的许可。

WIND RIVER SYSTEMS, INC.

By: 

Name, title *STEVE KENNEDY*
GROUP VICE PRESIDENT

Date *3-4-03*



译 序

VxWorks 是美国风河公司 (Wind River System 公司, 即 WRS 公司) 研制的一个高性能实时操作系统。WRS 公司组建于 1981 年, 是一个专门从事实时操作系统开发与生产的软件公司, 该公司在实时操作系统领域被世界公认为是最具有领导作用的公司。

VxWorks 是一个运行在目标机上的高性能、可裁减的嵌入式实时操作系统。它以其良好的可靠性和卓越的实时性被广泛地应用在通信、军事、航空、航天等高精尖技术及实时性要求极高的领域中。目前, VxWorks 由于具有开放式结构并且支持工业标准, 使开发者只需做最少的工作即可设计出有效的适合于不同用户要求的产品, 因而越来越受到消费电子行业的青睐。

本文主要讲述了标准的 VxWorks 网络栈。VxWorks 网络栈是 VxWorks 内部的一个与 BSD 4.4 兼容的实时 TCP/IP 堆栈, 它从基于 BSD4.3 的堆栈升级而来, 增加了完全的路由支持以及 Internet 的一些新特性, 使得 VxWorks 的网络性能更加优越。

本书布局如下。第 1 章对全书的内容作了简要的介绍。第 2 章主要对 VxWorks 支持的协议和工具作了简单的介绍。第 3 章到第 9 章分别对各个网络单元进行了介绍。第 3 章讨论了对数据链路层配置的一般要求和网络驱动程序。第 4 章介绍了 TCP/IP 协议包, 侧重强调了 TCP/IP 协议包的组成, 例如 TCP/IP 协议本身, 以及 UDP (用户数据报协议)、ARP (地址解析协议)、IGMP (网际组播传输协议) 和无编号接口的 IP 协议等的使用。第 5 章主要讨论了 VxWorks 中, DHCP、BOOTP 以及 SNMP 协议的配置。第 6 章讨论了路由信息协议 (RIP)。第 7 章讨论了 VxWorks 下的套接字工具。第 8 章讨论了对基于网络连接的资源的远程访问应用。第 9 章介绍了如何在 VxWorks 中配置及使用 DNS (域名服务系统) 和 SNTP (简单网络定时协议)。第 10 章和第 11 章分别对 VxWorks 下如何利用 MUX 添加新的网络接口驱动程序和添加新的网络服务应用程序进行了介绍, 并给出了相关的函数。

本书由王金刚、宫霄霖和熊辉编译, 参加翻译工作的人员还有高伟、苏淇、丁大尉、姜平、杨锡勋等。由于 VxWorks 嵌入式系统在国内的应用还处于起始阶段, 相关的书籍和材料有限, 再加上时间仓促以及编译人员水平有限, 不足之处在所难免, 希望读者提出宝贵意见。

译者

2003.4.30

目 录

第 1 章 概览	1
1.1 引言	1
1.2 章节概览	2
1.3 UML 语言注解	7
1.4 推荐参考文献	9
第 2 章 VxWorks 网络栈	10
2.1 引言	10
2.2 支持的协议和工具	10
2.3 设置与网络任务相关的任务优先级	12
第 3 章 数据链路层组件	14
3.1 引言	14
3.2 以太网驱动支持	14
3.2.1 BSD 数据报过滤器 (BPF)	14
3.2.2 附加的滤波器语法	15
3.3 共享内存的底板网络驱动	16
3.3.1 底板的共享内存区域	16
3.3.2 处理器间中断	20
3.3.3 顺序编址	21
3.3.4 共享内存网络的配置	24
3.4 用户接口	31
第 4 章 网络栈的配置	32
4.1 引言	32
4.2 配置设置摘要	32
4.3 创建时配置网络栈	34
4.3.1 网络协议的可检测性	35
4.3.2 配置 ARP、IP、TCP、UDP、IGMP 和 ICMP 协议	36
4.3.3 网络存储池配置	38
4.3.4 测试网络连接	43
4.3.5 支持多个网络接口驱动程序	44

4.4	TCP/IP 概述	45
4.5	配置 IP 到链路层的接口	46
4.5.1	绑定 IP 到 MUX (链路层)	46
4.5.2	给接口分配 IP 地址和网络掩码	47
4.5.3	配置 IP 广播地址	55
4.6	VxWorks 下的 IGMP	56
4.6.1	包含 IGMPv2	56
4.6.2	IGMPv2 APIs	56
4.7	手动编辑路由表	58
4.8	透明子网的代理 ARP	63
4.8.1	代理 ARP 协议概述	64
4.8.2	路由和代理 ARP 服务器	64
4.8.3	代理 ARP 和广播数据报	65
4.8.4	代理 ARP 配置	66
4.9	使用无编号接口	71
4.10	网络字节顺序	73
4.11	给 IP 地址配置主机名	73
第 5 章	网络配置协议	76
5.1	引言	76
5.2	BOOTP 自举引导协议	76
5.3	DHCP 动态主机配置协议	79
5.3.1	在镜像中包括 DHCP 组件	80
5.3.2	配置 DHCP 客户机	81
5.3.3	配置 DHCP 服务器	81
5.3.4	配置支持 DHCP 的中继代理	87
5.3.5	应用程序中的 DHCP	88
5.4	DHCP、BOOTP 以及网络初始化的引导参数	89
5.5	SNMP 简单网络管理协议	91
第 6 章	动态路由协议	93
6.1	引言	93
6.2	路由信息协议 RIP	93
6.2.1	VxWorks RIP 协议调试函数	94
6.2.2	RIP 协议配置	95
6.2.3	生成 RIP 不支持接口设备清单	96

第 7 章 VxWorks 套接字	98
7.1 引言	98
7.2 BSD 套接字	98
7.2.1 VxWorks 规范的套接字的依存性	99
7.2.2 数据报套接字 (UDP)	99
7.2.3 流套接字 (基于 TCP)	111
7.3 ZBUF 套接字	118
7.3.1 ZBUF 套接字和保护域	119
7.3.2 ZBUF 函数调用和发送数据缓存区数据	119
7.3.3 ZBUF 套接字数据结构的操作	119
7.3.4 ZBUF 套接字调用函数	127
第 8 章 远程访问应用	133
8.1 引言	133
8.2 RSH、FTP 和 netDrv	133
8.2.1 RSH	134
8.2.2 FTP	135
8.2.3 使用 netDrv	135
8.3 NFS 和 nfsDrv	137
8.3.1 VxWorks NFS 客户机	138
8.3.2 VxWorks NFS 服务器	140
8.4 TFTP	142
8.4.1 主机 TFTP 服务器	142
8.4.2 VxWorks TFTP 服务器	142
8.4.3 VxWorks TFTP 客户机	143
8.5 RPC 远程过程调用	143
8.6 rlogin	143
8.7 telnet	144
第 9 章 DNS 和 SNTP	146
9.1 引言	146
9.2 域名系统 DNS	146
9.2.1 域名	146
9.2.2 VxWorks 解析器	147
9.3 SNTP: 一个简单的定时协议	148
9.3.1 SNTP 客户机的使用	148
9.3.2 SNTP 服务器的使用	149

第 10 章 添加新的网络接口驱动程序	151
10.1 引言	151
10.1.1 MUX 和 OSI 网络模型	151
10.1.2 协议层与 MUX 的接口	152
10.1.3 数据链路层与 MUX 接口	153
10.1.4 END 和 NPT 驱动程序的差异	154
10.1.5 管理网络驱动程序和网络服务的存储区	155
10.1.6 在驱动程序中支持发散/聚集 (Scatter-Gather) 方式数据传送	155
10.1.7 早期的 NPT 驱动程序中的连接层头文件分配	155
10.1.8 缓冲区地址对齐	156
10.2 END 驱动程序实现	158
10.2.1 END 操作	158
10.2.2 The END 驱动程序和 MUX 接口	161
10.3 NPT 驱动程序的实现	170
10.3.1 NPT 操作	170
10.3.2 The NPT 驱动程序和 MUX 接口	173
10.4 移植 BSD 驱动程序到 MUX	181
10.4.1 移除引用设备号	182
10.4.2 生成一个 END 对象来表示设备	182
10.4.3 实现标准的 END 或 NPT 程序进入配给函数	182
10.5 支持多网络接口的驱动程序	184
10.5.1 为多个驱动程序 配置 VxWorks	184
10.5.2 在运行时启动额外的驱动程序	184
10.6 避免内存泄漏	185
 第 11 章 添加新的网络服务	 186
11.1 引言	186
11.2 写网络服务子层	186
11.2.1 接口初始化	186
11.2.2 数据结构和资源	187
11.2.3 子层函数 (sublayer Routines)	188
11.3 与 MUX 对接	189
11.3.1 使用 muxTkBind() 注册的服务函数	190
11.3.2 使用 muxBind() 注册服务函数	191
11.4 在服务中添加套接字接口	193
11.4.1 实现一个套接字后端	194
11.4.2 在套接字后端中使能 Zbuf 支持	196

11.4.3 套接字函数的实现.....	196
附录 A netBuflib 的使用	207
附录 B MUX/NPT 函数以及数据结构.....	224
附录 C 捆绑的 PPP、SLIP 和 CSLIP 协议.....	246

第 1 章 概 览

1.1 引 言

本书描述了基于 BSD4.4 TCP/IP 协议版本的标准的 VxWorks 网络栈。从第 2 章到第 9 章，描述了如何配置 VxWorks，以添加或删除某一特定的网络协议或 utility 实用程序；同时，还描述了如何设置与所提供的协议有关的参数。如果应用设计使用到 VxWorks 集成了的协议和驱动程序，则只需阅读这些章节。

本书的其他章节与附录适用于那些需要自己添加新的数据链路层驱动程序或新的网络服务程序的开发者。这一部分的内容来源于《VxWorks Network Protocol Toolkit Programmer's Guide》；由于该书是绑定在标准的 VxWorks 堆栈中的，因此本书也将其包括在内。



注意：本书尽管描述了如何编写与 MUX 的接口代码，却不是一本关于如何写网络协议或数据链路层驱动程序的向导。

保护域

在 Tornado 3.0 版本中，保护域特性被引入到 VxWorks 中。这使得可能生成受保护的系统资源分组，如任务、代码模块、信号量、消息队列，以及物理的或虚拟的内存页面。保护域之间的资源可以共享，但是这只有在受到相当严格控制的环境下才被允许。这种控制借用内存管理单元（MMU）的使用得以加强，从而能使得你的 VxWorks 系统的敏感部分免受错误代码和其他误操作的影响。

在 Tornado 2.0.2 和最近的 Tornado 2 系列版本中的 VxWorks 并不支持保护域。然而，Tornado 2.0.2 版本和 Tornado 3.1 版本中的网络协议栈工具是一样的；因此，本书对使用两种不同版本的 Tornado / VxWorks 的网络编程的使用者都适合。但是，使用支持保护域的 VxWorks 系统的网络编程者应该注意，直接访问某些网络协议栈将因其内核保护域作用而受到限制。对于这样一些受限制的应用，本书将会注明其细节。当然，如果使用的是 VxWorks（non-AE）版本，则可以忽略这些注意事项。

1.2 章节概览

在第 2 章对网络协议栈的概览后，从第 3 章到第 9 章分别描述了各个单一的网络协议单元。其余的章节则集中阐述了在现有的网络协议栈的基础上如何添加新的驱动程序和新的网络应用程序。以下是章节名称：

- 第 2 章 VxWorks 网络栈；
 - 第 3 章 数据链路层组件；
 - 第 4 章 网络栈的配置；
 - 第 5 章 网络配置协议；
 - 第 6 章 动态路由协议；
 - 第 7 章 VxWorks 下的套接字；
 - 第 8 章 远程访问应用；
 - 第 9 章 DNS 和 SNTP；
 - 第 10 章 添加新的网络接口驱动程序；
 - 第 11 章 添加新的网络服务。
- 本书还包含了一个目录，方便在使用时查找。

1. 数据链路层组件

第 3 章数据链路层组件讨论了对数据链路层配置的一般要求和网络驱动程序。这些网络驱动程序用于处理在网络硬件设备，如以太网卡、共享内存底板等之上的通信规范。同时，这些驱动程序还是网络协议栈的基础。本章还讨论了 Berkeley 包滤波器 (BPF)。



注意：本章前面描述的 VxWorks 集成的点对点协议 (PPP)、串行数据接口协议 (SLIP) 和可压缩串行数据接口协议 (CSLIP) 的工具现在已被认为不利于将来的应用并且将在下一个版本的 Tornado 中被删除。考虑到这种变化，集成的 PPP、SLIP 和 CSLIP 协议的配置信息将被移到综合介绍 PPP、SLIP 和 CSLIP 的附录 C。要了解更多的关于这些功能的变更，请联系您本地的风河公司 (Wind River) 的客户经理。如果需要点对点协议 (PPP) 的解决方案，请垂询客户经理关于 WindNet PPP。WindNet PPP 是基于可扩展的远程访问框架上的可靠的且易于管理的点对点协议 (PPP) 的解决方案。

2. VxWorks 下的 TCP/IP 协议

第 4 章网络栈的配置介绍了 TCP/IP 协议包。本章还侧重强调了 TCP/IP 协议包的组成，例如 TCP/IP 协议本身，以及 UDP (用户数据报协议)、ARP (地址解析协议)、IGMP (因特网多点传输协议) 和无编号接口的 IP 协议等的使用。

VxWorks 使用 MUX (多路复用器) 接口来处理与数据链路层的通信。使用 MUX 的目的在于将数据链路层和网络层分离。这样在现有网络协议基础上添加新的网络驱动程序就变得容易多了。同时,也更易于在 VxWorks 数据链路层之上运行可变化的协议。关于 MUX 的详细内容请参见第 10 章添加新的网络接口驱动程序。

在本章中,对 IP、TCP 和 UDP 的讨论只是一个概览,而不是这些协议的详细参考,它讲述了这些协议在 VxWorks 下应该如何进行配置。这些协议的详细参考请参见一些已出版的著作(详见 1.4 节推荐参考文献)。

本章对 ARP 和代理 ARP 协议的讨论更为详细。ARP 提供了 IP 地址与相应的媒介地址的动态对应。在 VxWorks 中,利用 ARP 协议生成了一个代理 ARP 协议,使得不同的网络看起来像是处在一个逻辑网络中。这种协议取代了原有的利用子网概念对共享资源网络的访问。这样做的好处在于保持了原始的 IP 地址,在处理无编号接口的点对点时也是如此(参见 4.9 节使用无编号接口)。

IGMP 协议保证了 TCP/IP 多次传送的数据报的正确送达。标准的 VxWorks 协议栈支持了面向主机方和面向路由器方的 IGMP V2 版本的协议,其技术规范是 RFC 2236。详情请参见 4.6 节 VxWorks 下的 IGMP。

3. 网络配置协议

第 5 章网络配置协议讨论了如下的协议配置:

- DHCP——动态主机配置协议;
- BOOTP——自举引导协议;
- SNMP——简单网络管理协议。

VxWorks 可以利用 DHCP 或 BOOTP 建立并保持网络配置信息。在启动时, DHCP 和 BOOTP 都能给客户机端分配 IP 地址以及相关信息。然而, BOOTP 分配的 IP 地址将是永久性的。DHCP 对 BOOTP 进行了扩展以支持 IP 地址的借用。如果需要,这种借用也可以是永久性的,同时也可以合理地设定借用的时间长度,从而满足客户机对 IP 地址的借用只需一定时长的要求。

当借用时间到期,被借用的 IP 地址将可以被其他客户机使用。如果借用 IP 地址的客户机在借用时间到期后还需要一个 IP 地址,则需要对借用重新协商以获得 IP 地址。由于这种协商的需要, DHCP 在系统运行时都处于活跃状态,而 BOOTP 则不是。

尽管 SNMP 协议也能提供网络配置信息,但它与 DHCP 和 BOOTP 明显的不同之处在于,它不是在系统启动时建立的。你可以建立一个网络管理站(NMS),通过它能够远程配置、监控和控制网络连接的设备。因此, SNMP 协议是一个具有特殊含义的网络配置协议。

4. 动态路由协议

第 6 章动态路由协议讨论了路由信息协议(RIP)。RIP 保存了小型内部网络的路由信息。VxWorks 提供的 RIP 协议服务器端是基于 BSD 4.4 版本的路由程序。它支持三种模式

的操作: RIP 协议版本 1、具有组播功能的 RIP 协议版本 2, 以及具有广播功能的 RIP 协议版本 3。RIP 协议工具还支持一个协议排除列表, 当列表中的接口设备在可达时可以使用该功能将接口设备排除在 RIP 协议之外。

5. VxWorks 下的套接字

第 7 章 VxWorks 下的套接字讨论了 VxWorks 下的套接字工具。利用套接字的应用能够在处理某一底板上、单一 CPU、以太网或任何网络连接之间的通信。套接字通信功能可以处理任何的 VxWorks 任务和主机系统进程间的通信。VxWorks 支持基于 TCP 和 UDP 的标准 BSD 套接字接口。利用这些套接字可以实现以下功能:

- 进程间通信;
- 访问 IP 的组播功能;
- 预览和修改路由表。

VxWorks 除了支持标准的 BSD 套接字外还支持 ZBUF 套接字。ZBUF 套接字是基于数据抽象上的另一种套接字调用, 我们称之为 ZBUF 套接字, 即零复制缓冲套接字。利用 ZBUF 套接字, 可以在一个 VxWorks 目标机上不同的软件模块之间共享数据缓存或数据缓存的不同部分。由于 ZBUF 套接字的这一功能, 在做数据复制时将无时间消耗。虽然 ZBUF 套接字接口是标准的 WRS 规范, 但只是对运行在 VxWorks 目标机上的应用才是有效的。而其他情况下的套接字终端连接可以使用标准的 BSD 套接字。



注意: VxWorks 同时也支持一种注册机制, 利用这种机制可以为已经移植到 VxWorks 中的协议添加新的套接字后端 (BackEnd)。详见第 11 章添加新的网络服务。

6. 远程访问应用

第 8 章远程访问应用讨论了对基于网络连接的资源的远程访问应用。VxWorks 支持以下的应用:

- RPC (远程过程调用, 应用于分布式处理);
- RSH (远程命令解释程序, 应用于远程命令的执行);
- FTP (文件传输协议, 应用于远程文件的访问);
- NFS (网络文件系统, 应用于远程文件访问);
- TFTP (小文件传输协议, 应用于远程文件访问);
- rlogin (应用于远程登录);
- telnet (应用于远程登录)。

7. 其他的网络应用

第 9 章 DNS 和 SNTP 介绍了如何在 VxWorks 中配置及使用 DNS (域名服务系统) 和 SNTP (简单网络定时协议)。

DNS (域名服务系统) 是一个分布式的数据库, 多数的 TCP/IP 协议下的应用利用它在

主机名与 IP 地址之间进行转换。DNS（域名服务系统）使用的是一种客户/服务器的结构。客户端被称作解析器（resolver），服务器端被称作命名服务器（name server）。VxWorks 在函数库 `resolvLib` 中提供了分解器客户端的功能函数。关于域名服务系统的详细内容，请参考 RFC-1034 和 RFC-1035。

SNTP（简单网络定时协议）的应用使用了 SNTP 客户端和 SNTP 服务器端。利用 SNTP 客户端功能，目标系统可以通过一个或多个远程资源提供的时钟来维持系统内部时钟的准确性。借助 SNTP 服务器，目标系统则可以给其他系统提供时钟信息。

8. 使用 MUX 添加新的网络接口驱动程序

第 10 章添加新的网络接口驱动程序描述了如何利用 MUX 添加新的网络接口驱动程序。使用 MUX（多路转换器）的目的在于提供一个接口，一方面使得网络服务应用程序免于受到特殊的网络驱动程序的影响，反之亦然，网络驱动程序也将不受特殊的网络应用程序的影响。

目前，MUX 支持两种网络接口驱动模式：END（增强型网络驱动）和 NPT（网络协议包驱动）驱动接口。END 驱动程序是面向数据帧格式的驱动接口，它利用 MUX 来实现数据帧之间的交换。现在标准的 VxWorks 网络协议栈中附送的驱动方式都是 END 型的。NPT 接口形式是面向数据报格式的驱动接口，它利用 MUX 来实现数据报之间的交换。

9. 加载网络设备

`routine muxDevLoad()` 实现将网络设备加载到系统。该 function 返回一个 COOKIE 号用以区分设备，并且在以后的所有其他的调用中都用这个 COOKIE 号来识别该设备。系统自动地为每一个在系统设备表中定义的接口加载函数 `muxDevLoad()`。有关如何建立设备表的详细内容请参见第 10 章添加 NPT 驱动程序到 VxWorks 或添加 END 驱动程序到 VxWorks 的相关内容。

10. 启动网络设备

在网络设备被成功加载后，VxWorks 调用函数 `muxDevStart()` 来启动设备。请参考第 10 章装载驱动程序的相关内容。

11. 使用 MUX 添加新的网络服务应用程序

第 11 章添加新的网络服务讲述了如何在 VxWorks 中利用 MUX 添加新的网络服务。本书定义了一种网络服务，它可以作为 OSI 网络模型的网络层与传输层之间的工具。在 VxWorks 中，网络服务与数据链路层的通信是利用 MUX 接口进行的。给 VxWorks 移植一种新的网络服务在很大程度上就是利用 MUX 接口移植数据链路层的访问代码。

12. 为新的网络服务注册一个套接字后端

VxWorks 套接字接口提供了函数 `sockLibAdd()` 以方便应用程序访问网络服务。这种注册功能使得在为网络服务添加套接字后端（Back End）时得以简化。在设计标准的套接字

接口时，已考虑到了为每一个协议层工具支持不同的套接字后端。因此，添加套接字库函数并不会干扰套接字后端进行其他网络服务。采取分层的结构使得这成为可能（见图 1-1）。

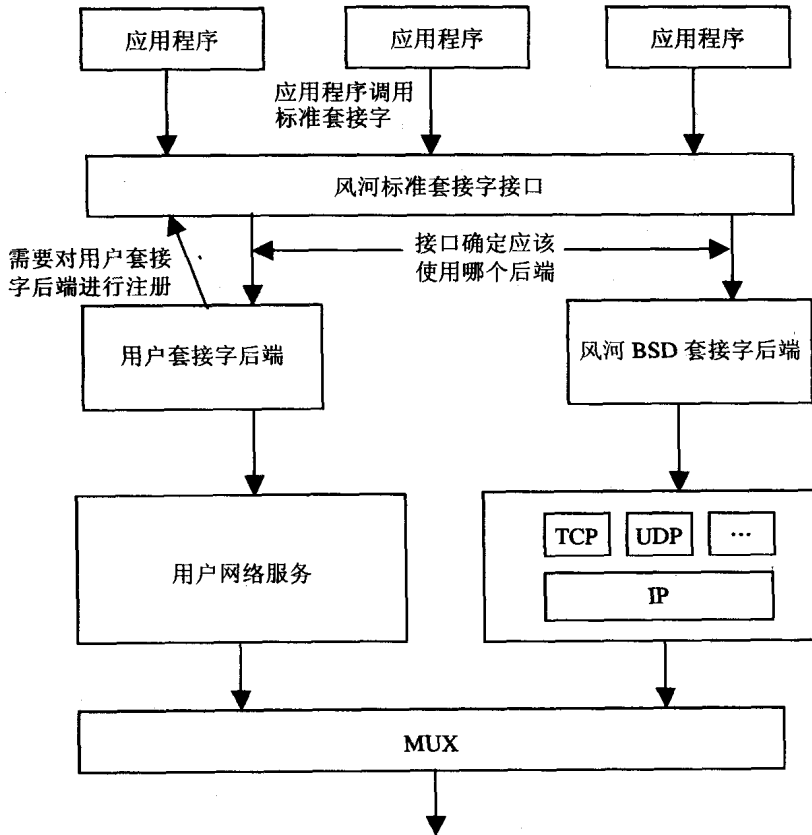


图 1-1 标准的套接字接口

由于添加一个新的网络服务可能会改变地址映射的要求，因此，VxWorks 网络栈也包含了一个为地址映射服务的注册机制。

13. 地址映射函数的注册服务

在使用 MUX 功能配置地址映射服务时，诸如地址解析和组播的地址映射，可以采用函数 `muxAddrResFuncAdd()` 和 `muxMCastAddrAdd()`。注册服务中的地址映射功能将每一成对的网络服务类型和网络驱动类型进行耦合。在必要时，网络服务将通过 MUX 获取到这些用于注册的函数。参见附录 B 中 `muxAddrResFuncAdd()` 和 `muxMCastAddrAdd()`。

14. 初始化网络服务

一般地，网络服务会提供一个函数将该服务分配给一个网络接口。这个函数使用 `muxTkBind()` 将网络服务与某一个网络驱动接口绑定在一起。在执行完这一函数之后，网络服务就可以通过相关的设备来收发数据报了。参见附录 B 中 `muxTkBind()`。

注意：通过设定配置参数 `MUX_MAX_BINDS` 可以改变 `MUX` 允许的最大的可绑定事件数。

15. 存储器管理和网络堆栈

附录 A 中网络缓冲库 (`netBufLib`) 的使用提供了关于如何利用 `netBufLib` 生成和管理存储池。目前情况下, 默认的 `VxWorks` 协议栈包含了两个存储池。关于这些存储池的管理请参见 4.3.3 节网络存储池配置。但是, 如果需要为一个新的网络驱动或协议生成新的存储池, 可以阅读附录 A 网络缓冲库 (`netBufLib`) 的使用。

1.3 UML 语言注解

本书的原理框图使用了 UML (统一建模语言) 语言注解方式。这部分是有关 UML 语言表述方式的一个概览。

图 1-2 中描述了如何用 UML 语言符号、类的数据成员和功能函数来表示一个类。在 `VxWorks` 协议栈中, 类是被当成数据结构来实现的。

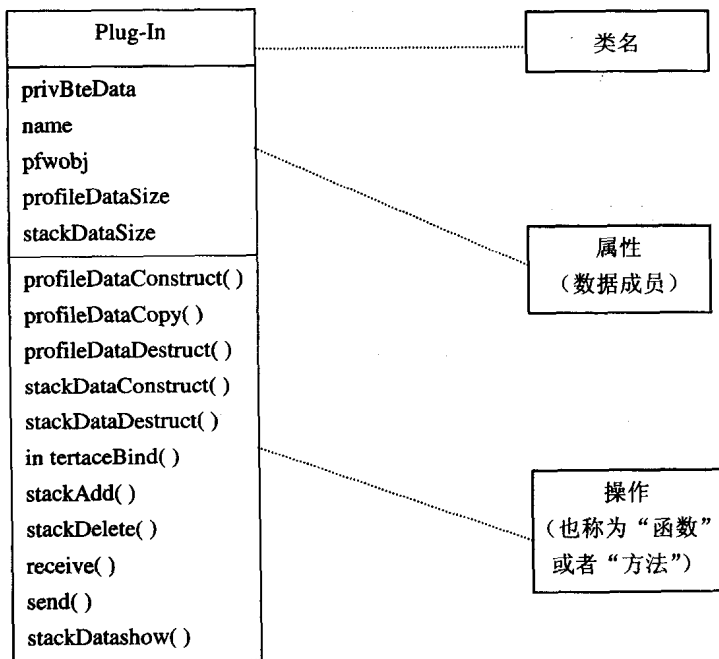


图 1-2 UML 注解的类

图 1-3 描述了使用 UML 语言符号表示的类的继承性。在 `VxWorks` 协议栈中, 类的继承性是通过使子类的数据结构中的第一个数据成员是父类的数据结构来实现的。在 UML