[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 9/54 (2006.01)
H04L 29/06 (2006.01)

[12] 发明专利说明书

专利号 ZL 200610161981.3

[45] 授权公告日 2008 年 10 月 22 日

[11] 授权公告号 CN 100428171C

[22] 申请日 2006.12.8

[21] 申请号 200610161981.3

[73] 专利权人 杭州华三通信技术有限公司

地址 310053 浙江省杭州市高新技术产业开发区之江科技工业园六和路 310 号华为杭州生产基地

[72] 发明人 石 磊

[56] 参考文献

WO2004/001615A1 2003.12.31

CN1737780A 2006.2.22

Linux 中用户空间与内核空间的通信实现. 杨宇音，李志淮. 微机发展，第 15 卷第 5 期. 2005

一种高效的用户级通信协议的研究与实现. 李斌，辛海红，胡铭曾. 计算机工程，第 32 卷第 1 期. 2006

审查员 钟文芳

[74] 专利代理机构 北京同立钧成知识产权代理有限公司

代理人 刘 芳

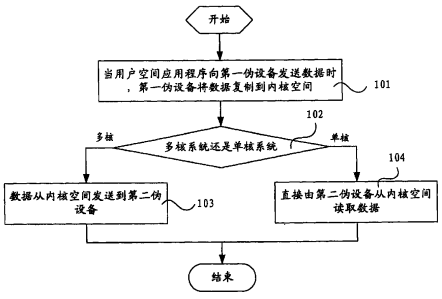
权利要求书 3 页 说明书 12 页 附图 3 页

[54] 发明名称

数据平面与控制平面之间的通讯方法

[57] 摘要

本发明涉及一种数据平面与控制平面之间的通讯方法，包括以下步骤：当运行在控制平面的操作系统的用户空间应用程序向运行在控制平面的操作系统中注册的第一伪设备发送数据时，第一伪设备将数据复制到运行在控制平面的操作系统的内核空间；如果数据平面运行于多核系统，则将数据从内核空间发送到所述多核系统中注册的第二伪设备；如果数据平面运行在单核系统，则直接由第二伪设备从内核空间读取数据。 在数据平面和控制平面分别注册了具有一致接口的伪设备，并将硬件环境的判断以及多核硬件时核间通信的操作封装起来，使上层业务模块的设计不需要考虑硬件环境，从而增强了上层业务模块的适应性和可扩展性。



1、一种数据平面与控制平面之间的通讯方法，其特征在于，包括以下步骤：

当运行在控制平面的操作系统的用户空间应用程序向运行在控制平面的操作系统中注册的第一伪设备发送数据时，所述第一伪设备将数据复制到所述运行在控制平面的操作系统的内核空间；

如果所述数据平面运行于多核系统，则将数据从所述内核空间发送到所述多核系统数据平面中注册的第二伪设备。

2、根据权利要求 1 所述的数据平面与控制平面之间的通讯方法，其特征在于，如果所述数据平面运行于单核系统，则直接由所述第二伪设备从所述内核空间读取数据。

3、根据权利要求 1 所述的数据平面与控制平面之间的通讯方法，其特征在于，当所述用户空间应用程序向第一伪设备发送的数据为操作指令时，所述将数据从所述内核空间发送到所述多核系统数据平面中注册的第二伪设备的操作具体为：

将所述第一伪设备的设备信息和该操作指令封装为消息，并通过核间高速通讯通道将所述消息从所述内核空间发送到所述多核系统；

所述多核系统解析出该消息中的第一伪设备的设备信息和操作指令，并根据该设备信息查找对应的第二伪设备，然后提供给该第二伪设备该操作指令以使该第二伪设备执行相应操作。

4、根据权利要求 1 所述的数据平面与控制平面之间的通讯方法，其特征在于，所述用户空间应用程序向第一伪设备发送的数据为用于配置数据平面的配置信息时，所述将数据从所述内核空间发送到所述多核系统数据平面中注册的第二伪设备的操作具体为：

调用核间高速通讯通道将该配置信息从所述内核空间发送到所述多核系统，所述多核系统根据该配置信息对第二伪设备进行配置操作。

5、根据权利要求 1 所述的数据平面与控制平面之间的通讯方法，其特征在于，所述将数据从所述内核空间发送到所述多核系统数据平面中注册的第二伪设备的操作具体为：第一伪设备判断所述数据是否低于预设的容量阈值，是则将数据封装为消息，并通过内核空间所在的 CPU 核经由核间高速通讯通路，发送到所述多核系统数据平面所在的 CPU 核；否则所述内核空间所在的 CPU 核将数据写入共享内存，并将数据的内存地址封装成消息，通过核间高速通讯通路发送到所述多核系统数据平面所在的 CPU 核，再由所述多核系统数据平面所在的 CPU 核根据所述消息中的内存地址将数据从共享内存中取出。

6、根据权利要求 2 所述的数据平面与控制平面之间的通讯方法，其特征在于，所述直接由所述第二伪设备从所述内核空间读取数据的操作具体为：

所述第二伪设备直接访问所述运行在控制平面的操作系统的内核空间的共享内存，并读取数据。

7、根据权利要求 1-6 任一所述的数据平面与控制平面之间的通讯方法，其特征在于，在注册第一伪设备和第二伪设备时，为所述第一伪设备和第二伪设备设置一致的操作接口。

8、根据权利要求 7 所述的数据平面与控制平面之间的通讯方法，其特征在于，所述为第一伪设备和第二伪设备设置一致的操作接口的操作具体为：

为第一伪设备和第二伪设备设置一致的字符型设备操作接口，所述字符型设备操作包括打开伪设备、关闭伪设备、从伪设备中读取数据、向伪设备写入数据、向伪设备发送操作指令和从伪设备读取状态信息中的一种或多种操作。

9、根据权利要求 7 所述的数据平面与控制平面之间的通讯方法，其特征在于，所述为第一伪设备和第二伪设备设置一致的操作接口的操作具

体为：为所述第一伪设备和第二伪设备设置一致的网络操作接口。

10、根据权利要求 9 所述的数据平面与控制平面之间的通讯方法，其特征在于，在所述多核系统数据平面中注册第二伪设备时，为所述数据平面分配 IP 地址，并在所述运行在控制平面的操作系统中增加到数据平面的通讯路由表。

数据平面与控制平面之间的通讯方法

技术领域

本发明涉及一种数通设备中数据平面与控制平面之间的通讯方法，尤其是针对多核（Multi-Core）环境下上层应用程序在执行时在数据平面与控制平面之间的进行数据通讯的方法。

背景技术

数通设备（用于数据通讯的设备，例如路由器等）按照工作职能的不同一般可以在逻辑上分为数据平面(DATA PLANE)和控制平面(CONTROL PLANE)两部分，其中数据平面主要负责对网络报文的高速处理，而控制平面主要运行管理软件负责对整个系统的配置以及运行一些控制层的协议。

在一般的单核（Single-Core）系统中，数据平面和控制平面在物理上不是相分离的结构，它们共享一个 CPU，并交替获得 CPU 时间以运行在它们的程序。以采用 Linux 作为操作系统的数通设备为例，假设控制平面运行于 Linux 的用户空间，而数据平面运行在 Linux 的内核空间。由于 Linux 的用户空间和内核空间的内存是相互隔离的，因此运行在 Linux 控制平面的管理程序在与数据平面的程序交互时，就必须通过用户空间与内核空间的通讯实现。实际上，用户空间和内核空间在操作系统中只是被逻辑上隔离，它们可以共享一个内存空间，因此数据平面和控制平面一般采用共享内存的方式进行通讯。

多核 CPU 的出现为数据平面和控制平面的以更高的效率运行提供了很好的条件。在多核系统设计中，可以根据多核 CPU 中的每个核（Core）处理功能的不同分成两类，其中一类运行数据平面的操作系统和应用程序，另一类

运行控制平面的操作系统和应用程序。

按照这种分类方法，可以选择在多核 CPU 中的一个核上运行控制平面系统和程序，控制平面使用 Linux 操作系统，用于提供通用开放的管理配置平台；其他的核上运行数据平面系统和程序，数据平面上运行与应用相关的专用的系统和专用程序。数据平面系统和控制平面系统具有共用的物理内存。

这种分工明显可以带来数据处理效率的提高，但由于两个平面运行在不同的核上，逻辑上使用不同的内存空间，因此可以被看作是两台设备。这样的系统设计方法必然要考虑基于多核 CPU 的数据平面和控制平面的通讯方式。

现有的多核 CPU 提供了核间高速通讯通路 API（即由多核 CPU 硬件实现的核间高速通信通路），这种核间高速通讯通路的方式适合在核间进行少量数据的高速发送，是解决运行数据平面的核与运行控制平面的核之间的通讯的基础。但是对于核间大量数据的发送，则需要核间高速通讯通路结合共享内存的方式实现。

上述的技术方案虽然解决了多核 CPU 的环境下数据平面和控制平面间的通信问题，但底层的应用程序接口缺少封装，运行在数据平面或控制平面中的上层应用程序在通讯时需要直接对底层的 API 进行控制，在程序设计上就须考虑底层硬件的情况而设计相应的处理方式，难以实现硬件无关性，设计难度较大，不易移植。

发明内容

本发明的目的是提出一种数据平面与控制平面之间的通讯方法，能够使数据平面或控制平面的上层业务模块在设计时不必考虑底层的硬件差异，既适用于多核环境，也可以应用于单核环境。

为实现上述目的，本发明提供了一种数据平面与控制平面之间的通讯方法，包括以下步骤：

当运行在控制平面的操作系统的用户空间应用程序向运行在控制平面的操作系统中注册的第一伪设备发送数据时，所述第一伪设备将数据复制到所述运行在控制平面的操作系统的内核空间；

第一伪设备判断如果所述数据平面是否运行于多核系统硬件，是则将数据从所述内核空间发送到运行在数据平面的操作所述多核系统数据平面中注册的第二伪设备。

在上述技术方案中，如果所述数据平面运行于单核系统，否则直接由所述第二伪设备从所述内核空间读取数据。

伪设备将硬件底层的操作封装起来，能够根据单核硬件和多核硬件的硬件环境进行数据传送，上层应用程序调用伪设备在数据平面和控制平面之间进行通讯时，不需考虑底层的通讯实现方式，降低了应用程序设计上的难度，也实现了在单核硬件和多核硬件上的可移植性。

当所述用户空间应用程序向第一伪设备发送的数据为操作指令时，所述将数据从所述内核空间发送到运行在数据平面的操作系统中注册的第二伪设备的操作具体为：

将所述第一伪设备的设备信息和该操作指令封装为消息，并通过核间高速通讯通道将所述消息从所述内核空间发送到运行在数据平面的操作系统；

所述运行在数据平面的操作系统解析出该消息中的第一伪设备的设备信息和操作指令，并根据该设备信息查找对应的第二伪设备，然后提供给该第二伪设备该操作指令以使该第二伪设备执行相应操作。

如果控制平面与数据平面交互的数据为用于配置数据平面的配置信息，所述将数据从所述内核空间发送到所述多核系统数据平面中注册的第二伪设备的操作具体为：

调用核间高速通讯通道将该配置信息从所述内核空间发送到所述多核系统，所述多核系统根据该配置信息对第二伪设备进行配置操作。

除了对多核硬件和单核硬件的封装外，还可以在控制平面和数据平面之间数据的传输方式上进行封装，所述将数据从所述内核空间发送到所述多核系统数据平面中注册的第二伪设备的操作具体为：第一伪设备判断所述数据是否低于预设的容量阈值，是则将数据封装为消息，并通过内核空间所在的 CPU 核经由核间高速通讯通路，发送到所述多核系统数据平面所在的 CPU 核；否则将所述内核空间所在的 CPU 核将数据写入共享内存，并将数据的内存地址封装成消息，通过核间高速通讯通路发送到所述多核系统数据平面所在的 CPU 核，再由所述多核系统所在的 CPU 核根据所述消息中的内存地址将数据从共享内存中取出。

核间高速通讯通路适合于核间的少量数据的高速传输，当核间需要传输的数据量较大时，则需要结合核间高速通讯通路和共享内存来实现核间的数据传输。在数据平面和控制平面各自的操作系统初始化时，可将共享内存区映射到同一个物理地址的内存块上，这个内存块相对于各自的操作系统是独立的，这时两个平面的系统都可以对共享内存进行访问。在通讯时，作为发送方的 CPU 核将需要交互的数据写入共享内存中，并将内存地址封装在核间高速通讯通道的消息中，并将该消息发到作为目的方的 CPU 核上，作为目的方的 CPU 核对接收到的消息进行解析后，按照消息中的物理地址从共享内存中将数据取出。

进一步地，当数据平面运行在单核环境时，用户空间和内核空间都处于同一个 CPU 核，因此第二伪设备可以直接访问所述运行在控制平面的操作系统的内核空间的共享内存，并读取数据。

在注册第一伪设备和第二伪设备时，为所述第一伪设备和第二伪设备设置一致的操作接口，能够极大的简化上层程序的设计难度。

为第一伪设备和第二伪设备设置操作接口时，可以设置一致的字符型设备操作接口，所述字符型设备操作包括打开伪设备、关闭伪设备、从伪设备中读取数据、向伪设备写入数据、向伪设备发送操作指令和从伪

设备读取状态信息中的一种或多种操作。

也可以设置一致的网络操作接口，在运行在数据平面的操作系统中注册第二伪设备时，为所述数据平面分配 IP 地址，并在所述运行在控制平面的操作系统中增加到数据平面的通讯路由表。采用的这种网络设备方式，可以使第二伪设备以及在第二伪设备通信时采用的共享内存和核间高速通讯通道封装成一个网络设备，从而使控制平面和数据平面的交互视为两台网络设备之间的交互，这种交互利用 TCP/IP 协议，并通过 Socket（套接字）的方式来实现通讯。由于 TCP/IP 协议的普及性，因此采用这种方式的可扩展性和适应性都较强。

基于上述技术方案，本发明具有以下优点：

1、本发明在数据平面和控制平面采用伪设备进行数据传输，由伪设备将硬件环境的判断以及具体到多核硬件的核间通信的操作封装起来，使上层业务模块的设计不需要考虑硬件环境，从而增强了上层业务模块的适应性和可扩展性。

2、本发明针对操作系统中适用的设备管理方式来设计伪设备的管理方式，使其易于兼容，在设计时可以采用字符设备或块设备管理方式，也可以采用网络管理方式，因此也具有较大的灵活性。

3、本发明在不同硬件环境下都为上层应用程序提供了统一的操作接口，极大的简化了上层应用程序的设计。

下面通过附图和实施例，对本发明的技术方案做进一步的详细描述。

附图说明

图 1 为本发明应用于单核环境下的结构示意图。

图 2 为本发明应用于多核环境下的结构示意图。

图 3 为本发明数据平面与控制平面的通讯方法的一实施例的流程示意图。

图 4 为本发明数据平面与控制平面的通讯方法的另一实施例的流程示意图。

图 5 为本发明数据平面与控制平面的通讯方法的又一实施例的流程示意图。

具体实施方式

本发明的原理是利用在操作系统中注册的伪设备(pseudo-device)来完成数据平面和控制平面的数据通讯，并由伪设备封装对硬件环境和/或数据传输方式的适配过程，伪设备的接口采用统一的接口，使上层应用程序在设计时得到极大的简化，且不必考虑其硬件环境所造成的影响。

如图 1 所示，为本发明应用于单核环境下的结构示意图。在单核 CPU 中运行操作系统，该操作系统包括用户空间和内核空间，用户空间和内核空间即表示操作系统的不同运行态，其中数据平面运行于内核空间，控制平面运行于用户空间。在用户空间中运行有控制平面应用程序 1，可以对数据平面中的数据平面应用程序 2 进行操作，数据平面应用程序 2 与控制平面应用程序 1 进行数据交互，在系统启动时，会在用户空间和内核空间分别注册伪设备 3 和伪设备 4，伪设备 3 和伪设备 4 的接口定义需设置一致，伪设备 3 和伪设备 4 之间的数据交互通过共享内存的方式来实现。

对于多核环境，如图 2 所示，本发明应用于多核环境下的结构示意图。控制平面运行在控制平面 CPU 上，在控制平面上运行着操作系统，数据平面运行于数据平面 CPU 上，该数据平面 CPU 是区别于控制平面 CPU 的其它核，在数据平面中运行着专用的操作系统及程序，例如一些专用嵌入式系统。在控制平面中运行有控制平面应用程序 1，能够对数据平面上的数据平面应用程序 2 进行操作，数据平面应用程序 2 与控制平面应用程序 1 进行数据交互。在系统启动时，在控制平面中注册伪设备 3'，在数据平面中注册伪设备 4'，与单核环境一样，其接口定义可以与伪设备 3'相一致。

本发明的基本过程如图 3 所示，为本发明数据平面与控制平面的通讯方法的一实施例的流程示意图，包括以下步骤：

步骤 101，当运行在控制平面的操作系统的用户空间应用程序向运行在控制平面的操作系统中注册的第一伪设备发送数据时，所述第一伪设备将数据复制到所述运行在控制平面的操作系统的内核空间；

步骤 102，如果数据平面运行于多核系统，则执行步骤 103；如果运行于单核系统，则执行步骤 104；

步骤 103，将数据从内核空间发送到所述多核系统中注册的第二伪设备，并结束操作；

步骤 104，直接由所述第二伪设备从所述内核空间读取数据。

伪设备在设计时可针对于多核系统或单核系统通过宏来实现不同的程序构架，当对于多核系统或单核系统对伪设备程序进行编译时，选择多核系统或单核系统相对应的宏执行编译操作。编译后的伪设备程序能够为多核系统或单核系统提供上层应用的接口，使上层的应用程序能够不必针对多核系统或单核系统实现对应的程序框架，从而简化了程序设计。

在数通设备启动时，会进行注册伪设备的操作，在注册过程中可为第二伪设备设置与第一伪设备一致的操作接口。这样可以降低上层应用程序的设计难度，使上层应用程序具有良好的可移植性。在内核空间与数据平面进行数据交互时，还需要对硬件环境进行判断以选择不同的处理过程。

当数通设备为多核硬件环境时，伪设备在传输数据时可以利用核间高速通讯通路进行传输，请参看图 4，为本发明数据平面与控制平面的通讯方法的另一实施例的流程示意图，包括以下步骤：

步骤 101，当运行在控制平面的操作系统的用户空间应用程序向运行在控制平面的操作系统中注册的第一伪设备发送数据时，所述第一伪设备将数据复制到所述运行在控制平面的操作系统的内核空间；

步骤 102，如果所述数据平面运行于多核系统，则执行步骤 103a；如

果数据平面运行于单核系统，则执行步骤 104';

步骤 103a，将第一伪设备的设备信息和操作指令封装为消息，并将所述消息从所述内核空间发送到运行在数据平面的操作系统;

步骤 103b，解析出该消息中的第一伪设备的设备信息和操作指令，查找对应于该设备信息的第二伪设备，并根据该操作指令进行操作，然后结束操作;

步骤 104'，第二伪设备直接访问所述运行在控制平面的操作系统的内核空间的共享内存，并读取数据。

由于数据平面中已经注册了第二伪设备的设备信息，在接收到消息后，根据解析出的第一伪设备的设备信息（可以为伪设备名、设备参数等）可以相应的查找出第二伪设备，并由第二伪设备根据操作指令执行相应的操作。在单核环境下，由于控制平面和数据平面实际处于同一种操作系统的不同运行态，因此第二伪设备可以直接读取数据，如果数据为某种操作指令，则执行相应的操作。

运行在控制平面的多种操作系统下都可以使用伪设备，如 FreeBSD、Solaris 和 Linux 等，下面以 Linux 为例，提供伪设备的接口定义的一个实例，这里以 C 语言作实例说明。

Linux 中通过 register_chrdev(unsigned int major, const char *name, struct file_operations *fops)和 unregister_chrdev(unsigned int major, const char *name)定义伪设备的注册和卸载接口，其中 const char *name 为伪设备的注册名，是 Linux 用户态程序访问该设备时需要指定的名称，然后可以根据需要选择操作接口，例如选择字符型设备操作接口和网络操作接口。

先以字符型设备操作接口为例，字符型设备操作可以包括：打开伪设备（open）、关闭伪设备（close）、从伪设备中读取数据（read）、向伪设备写入数据（write）、向伪设备发送操作指令和从伪设备读取状态

信息（ioctl）中的一种或多种操作。其中在对伪设备进行操作前，需要先执行打开伪设备的操作。

这种字符型设备操作接口在伪设备采用字符设备方式或块设备方式时使用，而这两种设备管理方式都与 Linux 中的设备管理方式是一致的。

本发明还提供了伪设备的另一种设备管理方式：网络设备方式。其采用的操作接口为网络操作接口，与字符型设备操作接口相比，除了打开、关闭等操作外，需要通过套接字进行数据的交互传输。在运行在数据平面的操作系统中注册第二伪设备时，需要为数据平面分配 IP 地址，并在运行在控制平面的操作系统中增加到数据平面的通讯路由表。

下面再具体说明字符型设备操作接口中向伪设备发送操作指令和从伪设备读取状态信息（Ioctl）的接口，这个接口可以对设备进行配置，完成控制平面和数据平面的指令和数据交互。

当用户空间的应用程序向第一伪设备发送的数据为用于配置数据平面的配置信息时，可以利用 Ioctl 接口进行操作，首先第一伪设备将该配置信息复制到运行在控制平面的操作系统的内核空间；如果数据平面运行于多核系统，则调用核间高速通讯通道将该配置信息进行封装后，从内核空间发送到多核系统，多核系统根据该配置信息对第二伪设备进行配置操作；如果数据平面运行在单核系统，则第二伪设备直接从所述内核空间读取该配置信息，并进行配置操作。在配置后可以根据需要选择是否向用户空间返回信息。

对于多核系统来说，运行在控制平面的操作系统将 Ioctl 操作指令封装到指定的消息格式中，然后调用多核的核间高速通道发送到数据平面所在核进行处理。处理完成后，数据平面所在核将返回值封装在消息中，发回控制平面所在核，然后按照回应消息的内容将结果返回用户空间应用程序。

以下用 C 语言给出消息的格式，本发明提供的消息格式只为示例，并

不作为对消息格式的限定。

```
struct ioctl_msg
{
    unsigned short tunnel_id; /*接收应答的消息队列 ID*/
    unsigned long addr;          /*ioctl 命令内容地址指针*/
    char dev_name[DEV_NAME_LEN_MAX];    /*伪设备名*/
    unsigned int  ioctl_cmd;             /*ioctl 命令号*/
    int ret;                    /*ioctl 执行结果返回值*/
};
```

以上实例都是以 C 语言实现的，但这并不限制本发明的实现语言为 C 语言，其他能够实现伪设备的高级语言或者汇编语言都应在本发明的覆盖之内。

下面给出基于上述消息格式的通讯过程，第一伪设备先从命令号中取出 Ioctl 下发的缓冲区的长度，并申请一块共享内存，当配置下发命令时，从用户空间将 ioctl 下发的缓冲区内容复制到共享内存中；然后将 ioctl 命令号、伪设备名和共享内存的物理地址封装到消息格式中，并发送消息；在运行在数据平面的操作系统查找到第二伪设备后，由第二伪设备将应答消息接收队列封装到消息格式中，并从应答接收队列接收应答消息，然后从消息中取出 ioctl 返回码。当 ioctl 命令为读取配置，则从消息中取出 Ioctl 输出缓冲区的地址并将共享内存中的内容写回用户空间，然后释放共享内存，并将返回码返回。

如果使运行在数据平面的上层业务模块在与控制平面内的控制平面应用程序在数据交互时不需考虑硬件的影响，则这种上层业务模块就可以运行在专用的嵌入式系统中，以嵌入式系统为例，在系统启动时需要注册一个与运行在控制平面的操作系统中伪设备接口一致的伪设备，依旧以 C 的实现方式举例，如下：

除了注册、卸载接口的定义外，还需要实现对多核系统的嵌入式系统中伪设备的管理，保存伪设备注册信息。在嵌入式系统接收到消息后进行解析，根据解析后消息中的伪设备的设备名查找注册过的伪设备，并执行该伪设备注册过的操作。在嵌入式系统处理完成 rpc_ioctl 命令后，需要将返回值和输出的缓冲区的地址封装到上述消息格式中，通过消息环将消息发送回多核系统控制平面所在核。以上描述了伪设备封装了多核环境和单核环境的硬件影响，对于多核环境下消息传递的方式还可以进一步的封装，如图 5 所示，为本发明数据平面与控制平面的通讯方法的另一实施例的流程示意图，包括以下步骤：

步骤 101，当运行在控制平面的操作系统的用户空间应用程序向运行在控制平面的操作系统中注册的第一伪设备发送数据时，所述第一伪设备将数据复制到所述运行在控制平面的操作系统的内核空间；

步骤 102，第一伪设备判断所述数据平面是否运行于多核硬件，如果运行于多核硬件，则执行步骤 1031，如果运行单核硬件，则执行步骤 104'；

步骤 1031，第一伪设备判断被传输的数据是否低于预设的容量阈值，是则执行步骤 1035，否则执行步骤 1032；

步骤 1032，内核空间所在的 CPU 核将数据写入共享内存；

步骤 1033，将数据的内存地址封装成消息，通过核间高速通讯通路发送到运行在数据平面的操作系统所在的 CPU 核；

步骤 1034，由运行在数据平面的操作系统所在的 CPU 核根据所述消息中的内存地址将数据从共享内存中取出，并结束操作；

步骤 1035，将数据封装为消息，并通过内核空间所在的 CPU 核经由核间高速通讯通路，发送到运行在数据平面的操作系统所在的 CPU 核，并结束操作；

步骤 104'，第二伪设备直接访问所述运行在控制平面的操作系统的内核空间的共享内存，并读取数据。

本实施例提供了控制平面与数据平面之间数据传输的进一步封装，即采用核间高速通讯通道和共享内存的结合方式，其中核间高速通讯通道中利用消息来传输数据，在消息中通常需要封装伪设备的识别标志。如本实施例所述，在核间高速通讯通道传输的消息只能够运送较少容量的数据，如果需要传输较大量的数据，就需要借助于共享内存的方式，这时消息的作用便是传送数据所在的物理地址。

从上述具体流程可以看出，上层业务模块和应用程序在设计时只需利用统一的程序接口进行设计即可，不需要考虑复杂的硬件环境，对硬件环境和通讯方式的适配都由伪设备来完成，从用户设计和使用的角度都屏蔽了硬件的影响，因此降低了设计难度。由于各种伪设备都可采用统一的接口，上层业务模块也因此具备了良好的可扩展性和可移植性。

最后应当说明的是：以上实施例仅用以说明本发明的技术方案而非对其限制；尽管参照较佳实施例对本发明进行了详细的说明，所属领域的普通技术人员应当理解：依然可以对本发明的具体实施方式进行修改或者对部分技术特征进行等同替换；而不脱离本发明技术方案的精神，其均应涵盖在本发明请求保护的技术方案范围当中。

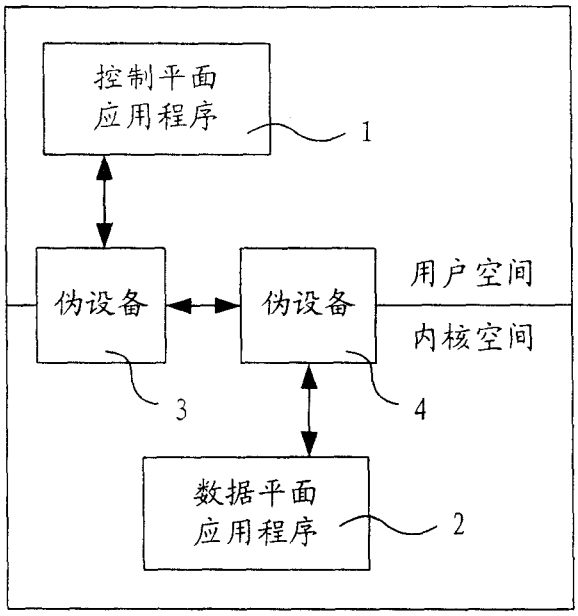

控制平面
应用程序
1
伪设备
伪设备
用户空间
内核空间
3
4
数据平面
应用程序
2

图 1

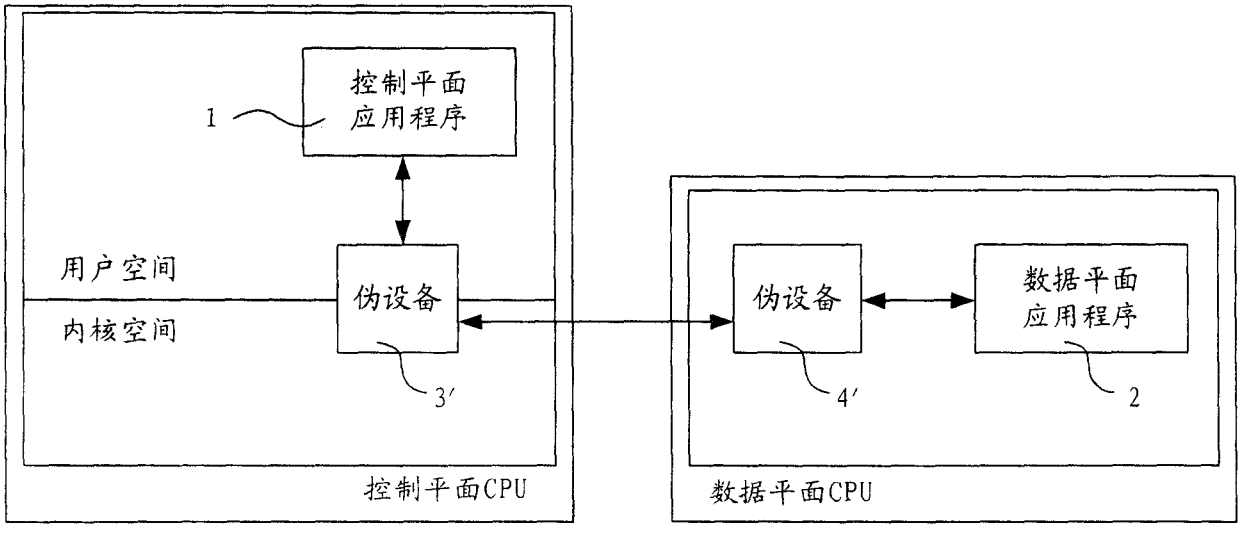

控制平面
应用程序
1
用户空间
内核空间
伪设备
3′
控制平面CPU
伪设备
数据平面
应用程序
4′
2
数据平面CPU

图 2

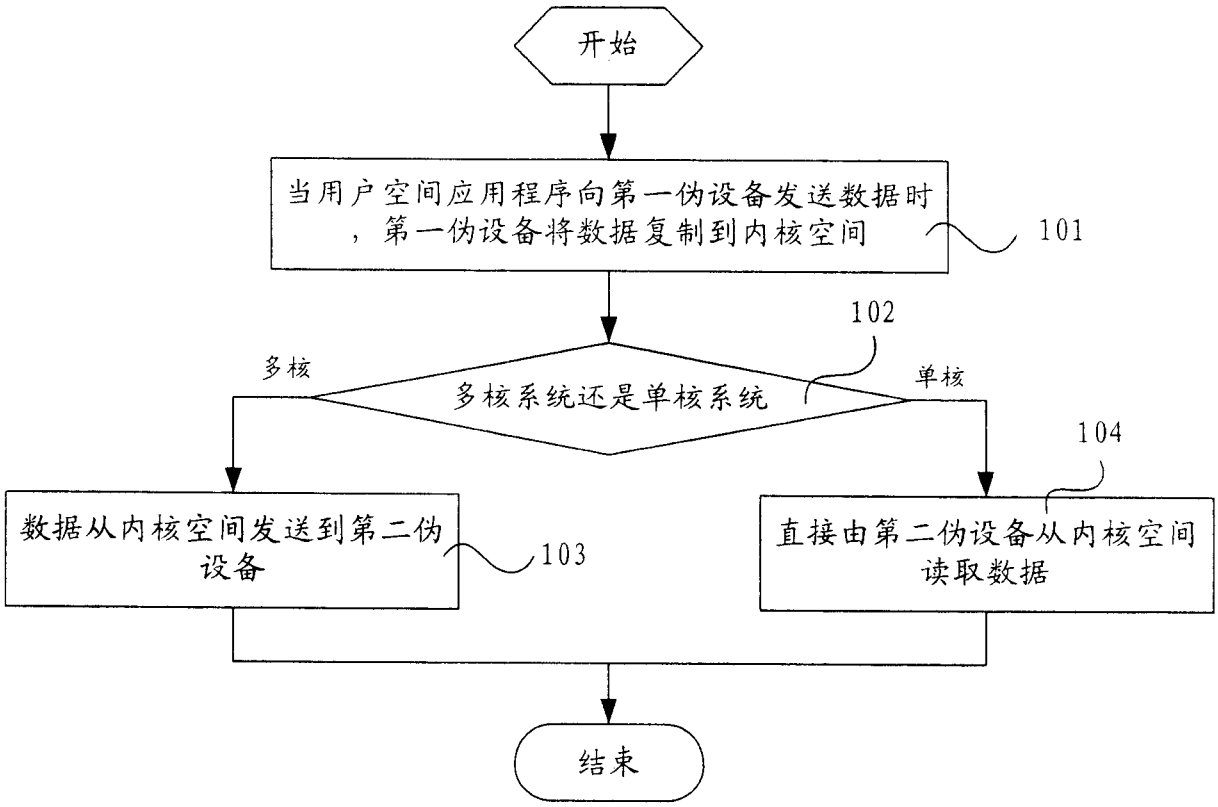

开始
当用户空间应用程序向第一伪设备发送数据时，第一伪设备将数据复制到内核空间
101
102
多核系统还是单核系统
多核
单核
数据从内核空间发送到第二伪设备
103
104
直接由第二伪设备从内核空间读取数据
结束

图 3

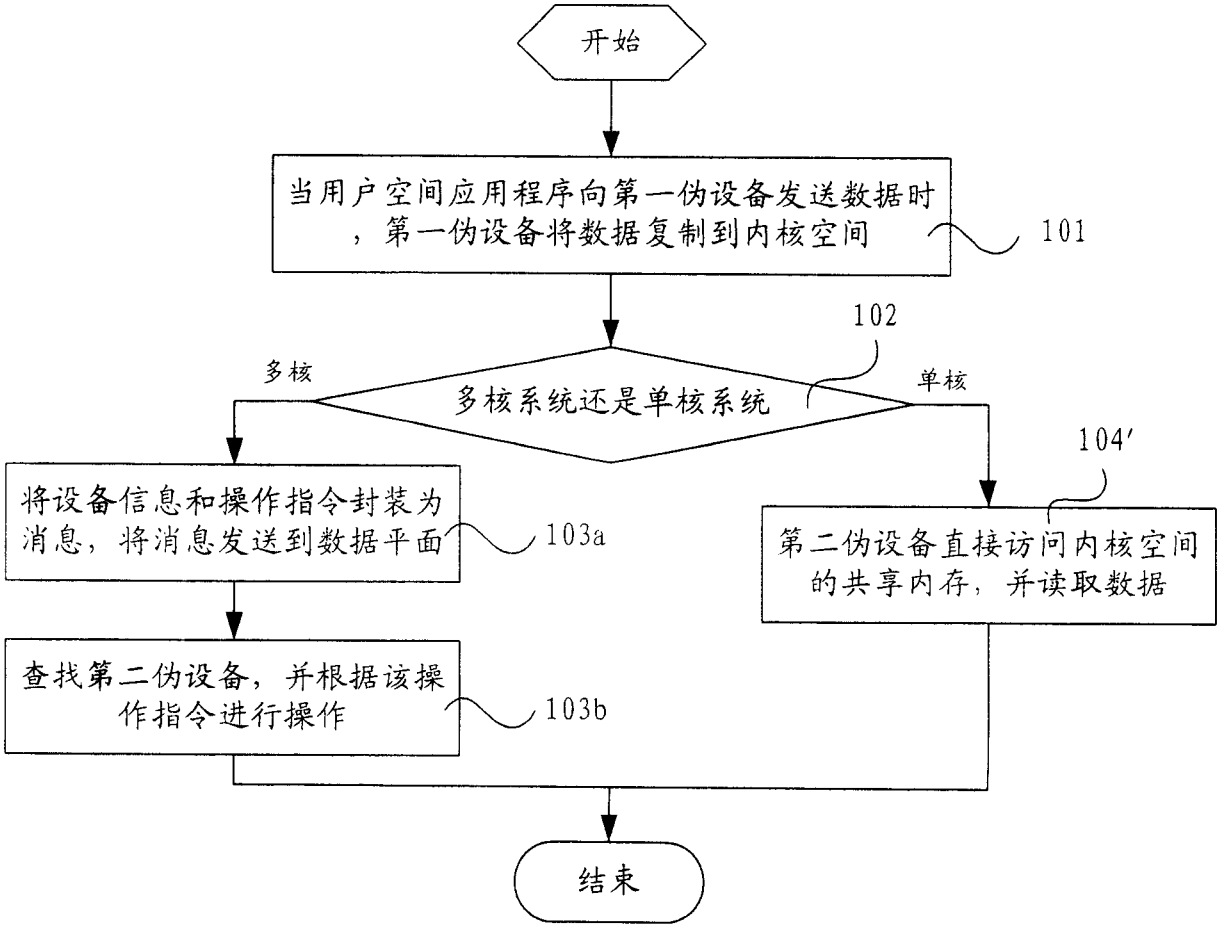

开始
当用户空间应用程序向第一伪设备发送数据时，第一伪设备将数据复制到内核空间
101
102
多核系统还是单核系统
多核
单核
将设备信息和操作指令封装为消息，将消息发送到数据平面
103a
查找第二伪设备，并根据该操作指令进行操作
103b
104′
第二伪设备直接访问内核空间的共享内存，并读取数据
结束

图 4

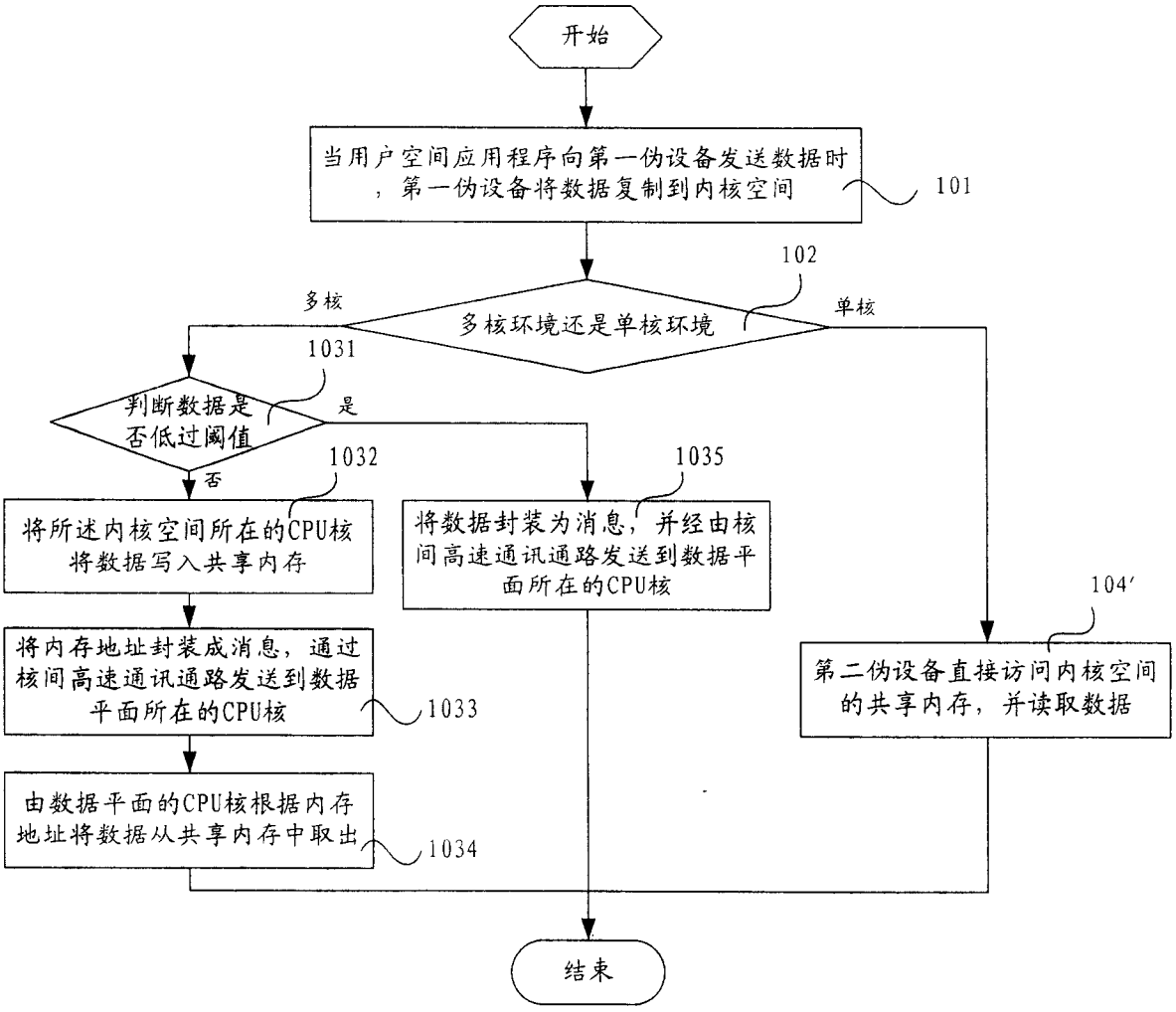

开始
当用户空间应用程序向第一伪设备发送数据时，第一伪设备将数据复制到内核空间
101
102
多核环境还是单核环境
多核
单核
1031
判断数据是否低过阈值
是
否
1032
将所述内核空间所在的CPU核将数据写入共享内存
1035
将数据封装为消息，并经由核间高速通讯通路发送到数据平面所在的CPU核
104'
第二伪设备直接访问内核空间的共享内存，并读取数据
将内存地址封装成消息，通过核间高速通讯通路发送到数据平面所在的CPU核
1033
由数据平面的CPU核根据内存地址将数据从共享内存中取出
1034
结束

图 5