(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 116166328 A
(43)申请公布日 2023.05.26

(21)申请号 202211701560.0

(22)申请日 2022.12.28

(71)申请人 苏州长风航空电子有限公司
地址 215151 江苏省苏州市高新区建林路379号

(72)发明人 程骥思 辛春明 赵振华 张锋

(74)专利代理机构 北京清大紫荆知识产权代理有限公司 11718
专利代理师 张梦龙

(51)Int.Cl.
G06F 9/4401(2018.01)

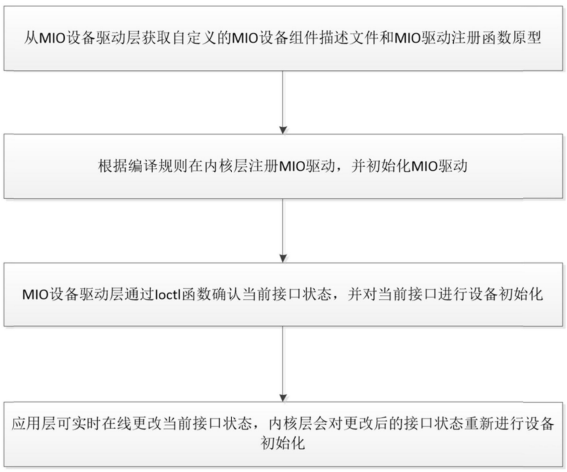
权利要求书1页 说明书4页 附图2页

(54)发明名称

一种VxWorks操作系统下驱动MIO设备的方法

(57)摘要

本申请提供了一种VxWorks操作系统下驱动MIO设备的方法,属于机载显示的技术领域,具体包括:从MIO设备驱动层获取自定的MIO设备组件描述文件和MIO驱动注册函数原型;根据编译规则在内核层注册MIO设备驱动,并初始化MIO设备驱动,以及配置MIO设备资源。应用层实时在线更改当前接口状态,内核层对更改后的接口重新进行设备初始化;MIO设备驱动层通过Ioctl函数确认当前接口状态,并对当前接口进行对应设备的初始化;接口对应的设备初始化完成后,以相应的接口与外界进行通信。通过本申请的处理方案,能够在线改变当前MIO的接口状态,提高了效率。



CN 116166328 A

1.一种VxWorks操作系统下驱动MIO设备的方法,其特征在于,包括:

步骤1,从MIO设备驱动层获取自定的MIO设备组件描述文件和MIO驱动注册函数原型;

步骤2,根据自定义的MIO设备组件描述文件和MIO驱动注册函数原型,形成MIO驱动的编译规则;

步骤3、根据编译规则在内核层提供的对下接口函数iosDrvInstal l和iOSDevAdd向内核层的I/O子系统注册MIO设备驱动,并初始化MIO设备驱动,以及配置MIO设备资源。

步骤4,应用层实时在线更改当前接口状态,内核层对更改后的接口重新进行设备初始化;

步骤5,MIO设备驱动层通过Ioctl函数确认当前接口状态,并对当前接口进行对应设备的初始化;

步骤6、接口对应的设备初始化完成后,以相应的接口与外界进行通信。

2.根据权利要求1所述的VxWorks操作系统下驱动MIO设备的方法,其特征在于,MIO设备驱动层包括MIO设备和MIO驱动单元;

在MIO驱动单元设置有用户自定义的MIO设备组件描述文件和MIO驱动注册函数原型;

MIO驱动注册函数原型中包括MIO驱动初始化函数和MIO驱动服务函数;

MIO驱动初始化函数包括设置MIO全局控制寄存器基址,读取MIO版本号,读取当前MIO功能状态。

3.根据权利要求2所述的VxWorks操作系统下驱动MIO设备的方法,其特征在于,编译规则以Makefile文件的形式存储在MIO驱动单。

4.根据权利要求3所述的VxWorks操作系统下驱动MIO设备的方法,其特征在于,根据编译规则编写MIO驱动注册函数代码并存储在MIO驱动单元。

5.根据权利要求1所述的VxWorks操作系统下驱动MIO设备的方法,其特征在于,所述步骤5中,Ioctl函数设置如下:针对读取的参数进行判定:若参数为0x0,则控制当前MIO功能为I2C,读取MIO功能状态确认为I2C功能后,进行I 2C设备的初始化;若参数为1,则控制当前MIO功能为UART,读取MIO功能状态确认为UART功能后,进行UART设备的初始化;若参数为2,则控制当前MIO功能为PWM,读取当前MIO功能为PWM后,进行PWM设备的初始化。

6.根据权利要求1所述的VxWorks操作系统下驱动MIO设备的方法,其特征在于,所述步骤4中,应用层上通过在应用程序设置ioctl函数或输入命令,以在线改变当前MIO的接口状态。

一种VxWorks操作系统下驱动MIO设备的方法

技术领域

[0001] 本申请涉及机载显示的领域,尤其是涉及一种VxWorks操作系统下驱动MIO设备的方法。

背景技术

[0002] 飞腾X100是一款主CPU配套芯片,其主要功能是实现GPU系统和PCIE、USB、SATA等高速接口扩展,同时实现整机系统的上下电控制等功能。

[0003] 飞腾2000/4平台是一款面向桌面应用的高性能通用4核处理器。每2个核构成1个处理器核簇(Cluster),并共享L2 Cache。处理器核通过片内高速互联网络及相关控制器与存储系统、I/O系统相连。存储系统包含Cache子系统和DDR,I/O子系统包括PCIE、高速I/O子系统、千兆以太网GMAC和低速I/O系统。

[0004] VxWorks是美国Wind River公司开发的嵌入式实时操作系统,具有高性能、可高度裁剪等特点,能够支持多种微处理器,如PowerPC、X86、ARM、MIPS等。它以其良好的可靠性和卓越的实时性能够被广泛的应用在通信、军事、航空、航天等高精尖技术及对实时要求极高的领域中。

[0005] MIO接口英文名称为MULTUSE IO,即为多功能接口IO。它仅通过两个GPIO口便可拥有三种功能,分别为I2C、PWM、UART功能。并且三个控制器拥有相同的配置空间。正是因为三者拥有相同的配置空间,那么他们无法同时被初始

[0006] 化,这就导致了每次需要使用另一功能时,必须修改程序,重新烧写系统。这5样操作复杂,并且效率不高,容易出错。

发明内容

[0007] 有鉴于此,本申请提供一种VxWorks操作系统下驱动MIO设备的方法,解决了现有技术中的问题,能够在线改变当前MIO的接口状态,提高了效率。

[0008] 本申请提供的一种VxWorks操作系统下驱动MIO设备的方法采用如下的技0术方案:

[0009] 一种VxWorks操作系统下驱动MIO设备的方法,其特征在于,包括:

[0010] 步骤1,从MIO设备驱动层获取自定的MIO设备组件描述文件和MIO驱动注册函数原型;

[0011] 步骤2,根据自定义的MIO设备组件描述文件和MIO驱动注册函数原型,5形成MIO驱动的编译规则;

[0012] 步骤3、根据编译规则在内核层提供的对下接口函数iosDrvInstall和iOSDevAdd向内核层的I/O子系统注册MIO设备驱动,并初始化MIO设备驱动,以及配置MIO设备资源。

[0013] 步骤4,应用层实时在线更改当前接口状态,内核层对更改后的接口重新进行设备初始化;

[0014] 步骤5,MIO设备驱动层通过Ioctl函数确认当前接口状态,并对当前接口进行对应

设备的初始化；

[0015] 步骤6、接口对应的设备初始化完成后，以相应的接口与外界进行通信。

[0016] 可选的，MIO设备驱动层包括MIO设备和MIO驱动单元；

[0017] 在MIO驱动单元设置有用户自定义的MIO设备组件描述文件和MIO驱动注册函数原型；

[0018] MIO驱动注册函数原型中包括MIO驱动初始化函数和MIO驱动服务函数；

[0019] MIO驱动初始化函数包括设置MIO全局控制寄存器基址，读取MIO版本号，读取当前MIO功能状态。

[0020] 可选的，编译规则以Makefile文件的形式存储在MIO驱动单。

[0021] 可选的，根据编译规则编写MIO驱动注册函数代码并存储在MIO驱动单元。

[0022] 可选的，所述步骤5中，Ioctl函数设置如下：针对读取的参数进行判定：若参数为0x0，则控制当前MIO功能为I2C，读取MIO功能状态确认为I2C功能后，进行I2C设备的初始化；若参数为1，则控制当前MIO功能为UART，读取MIO功能状态确认为UART功能后，进行UART设备的初始化；若参数为2，则控制当前MIO功能为PWM，读取当前MIO功能为PWM后，进行PWM设备的初始化。

[0023] 可选的，所述步骤4中，应用层上通过在应用程序设置ioctl函数或输入命令，以在线改变当前MIO的接口状态。

[0024] 综上所述，本申请包括以下有益技术效果：

[0025] 本申请在VxWorks下建立自定义的MIO设备描述文件，进而编写MIO驱动的编译规则，接着注册MIO驱动以及初始化MIO驱动，然后设计MIO驱动中Ioctl函数，最后应用层上可利用ioctl函数控制MIO接口当前的状态，从而达到实时切换接口的目的。

附图说明

[0026] 为了更清楚地说明本申请实施例的技术方案，下面将对实施例中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其它的附图。

[0027] 图1为本申请Vxworks下MIO驱动设备实现方法的流程图；

[0028] 图2为本申请VxWorks下MIO驱动设备具体设计流程图。

具体实施方式

[0029] 下面结合附图对本申请实施例进行详细描述。

[0030] 以下通过特定的具体实例说明本申请的实施方式，本领域技术人员可由本说明书所揭露的内容轻易地了解本申请的其他优点与功效。显然，所描述的实施例仅仅是本申请一部分实施例，而不是全部的实施例。本申请还可以通过另外不同的具体实施方式加以实施或应用，本说明书中的各项细节也可以基于不同观点与应用，在没有背离本申请的精神下进行各种修饰或改变。需说明的是，在不冲突的情况下，以下实施例及实施例中的特征可以相互组合。基于本申请中的实施例，本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例，都属于本申请保护的范围。

[0031] 要说明的是，下文描述在所附权利要求书的范围内的实施例的各种方面。应显而

易见，本文中所描述的方面可体现于广泛多种形式中，且本文中所描述的任何特定结构及/或功能仅为说明性的。基于本申请，所属领域的技术人员应了解，本文中所描述的一个方面可与任何其它方面独立地实施，且可以各种方式组合这些方面中的两者或两者以上。举例来说，可使用本文中所阐述的任何数目个方面来实施设备及/或实践方法。另外，可使用除了本文中所阐述的方面中的一或多者之外的其它结构及/或功能性实施此设备及/或实践此方法。

[0032] 还需要说明的是，以下实施例中所提供的图示仅以示意方式说明本申请的基本构想，图式中仅显示与本申请中有关的组件而非按照实际实施时的组件数目、形状及尺寸绘制，其实际实施时各组件的型态、数量及比例可为一种随意的改变，且其组件布局型态也可能更为复杂。

[0033] 另外，在以下描述中，提供具体细节是为了便于透彻理解实例。然而，所属领域的技术人员将理解，可在没有这些特定细节的情况下实践所述方面。

[0034] 本申请实施例提供一种VxWorks操作系统下驱动MIO设备的方法。

[0035] 如图1和图2所示，一种VxWorks操作系统下驱动MIO设备的方法，包括：

[0036] 步骤1，从MIO设备驱动层获取自定的MIO设备组件描述文件和MIO驱动注册函数原型；MIO设备驱动层包括MIO设备和MIO驱动单元；在MIO驱动单元设置有用户自定义的MIO设备组件描述文件和MIO驱动注册函数原型；MIO驱动注册函数原型中包括MIO驱动初始化函数(如xxxDrv，xxxDevCreate)和MIO驱动服务函数(如xxxOpen、xxxRead、xxxWrite等)；MIO驱动初始化函数包括设置MIO全局控制寄存器基址，读取MIO版本号，读取当前MIO功能状态。

[0037] 步骤2，根据自定义的MIO设备组件描述文件和MIO驱动注册函数原型，形成MIO驱动的编译规则；编译规则以Makefile文件的形式存储在MIO驱动单。

[0038] 步骤3、根据编译规则编写MIO驱动注册函数代码并存储在MIO驱动单元，根据编译规则在内核层(IOSLib)提供的对下接口函数iosDrvInstall和iOSDevAdd向内核层的I/O子系统注册MIO设备驱动，并初始化MIO设备驱动，以及配置MIO设备资源。

[0039] 步骤4，应用层实时在线更改当前接口状态，内核层对更改后的接口重新进行设备初始化。应用层上通过在应用程序设置ioctl函数或输入命令，以在线改变当前MIO的接口状态。

[0040] 步骤5，MIO设备驱动层通过Ioctl函数确认当前接口状态，并对当前接口进行对应设备的初始化。

[0041] 具体的，Ioctl函数设置如下：针对读取的参数进行判定：若参数为0x0，则控制当前MIO功能为I2C，读取MIO功能状态确认为I2C功能后，进行I2C设备的初始化；若参数为1，则控制当前MIO功能为UART，读取MIO功能状态确认为UART功能后，进行UART设备的初始化；若参数为2，则控制当前MIO功能为PWM，读取当前MIO功能为PWM后，进行PWM设备的初始化。

[0042] 步骤6、接口对应的设备初始化完成后，以相应的接口(I2C、UART、PWM三者之一))与外界进行通信。

[0043] 本申请在VxWorks下建立自定义的MIO设备描述文件，进而编写MIO驱动的编译规则，接着注册MIO驱动以及初始化MIO驱动，然后设计MIO驱动中Ioctl函数，最后应用层上可利用ioctl函数控制MIO接口当前的状态，从而达到实时切换接口的目的。

[0044] 以上所述，仅为本申请的具体实施方式，但本申请的保护范围并不局限于此，任何熟悉本技术领域的技术人员在本申请揭露的技术范围内，可轻易想到的变化或替换，都应涵盖在本申请的保护范围之内。因此，本申请的保护范围应以权利要求的保护范围为准。

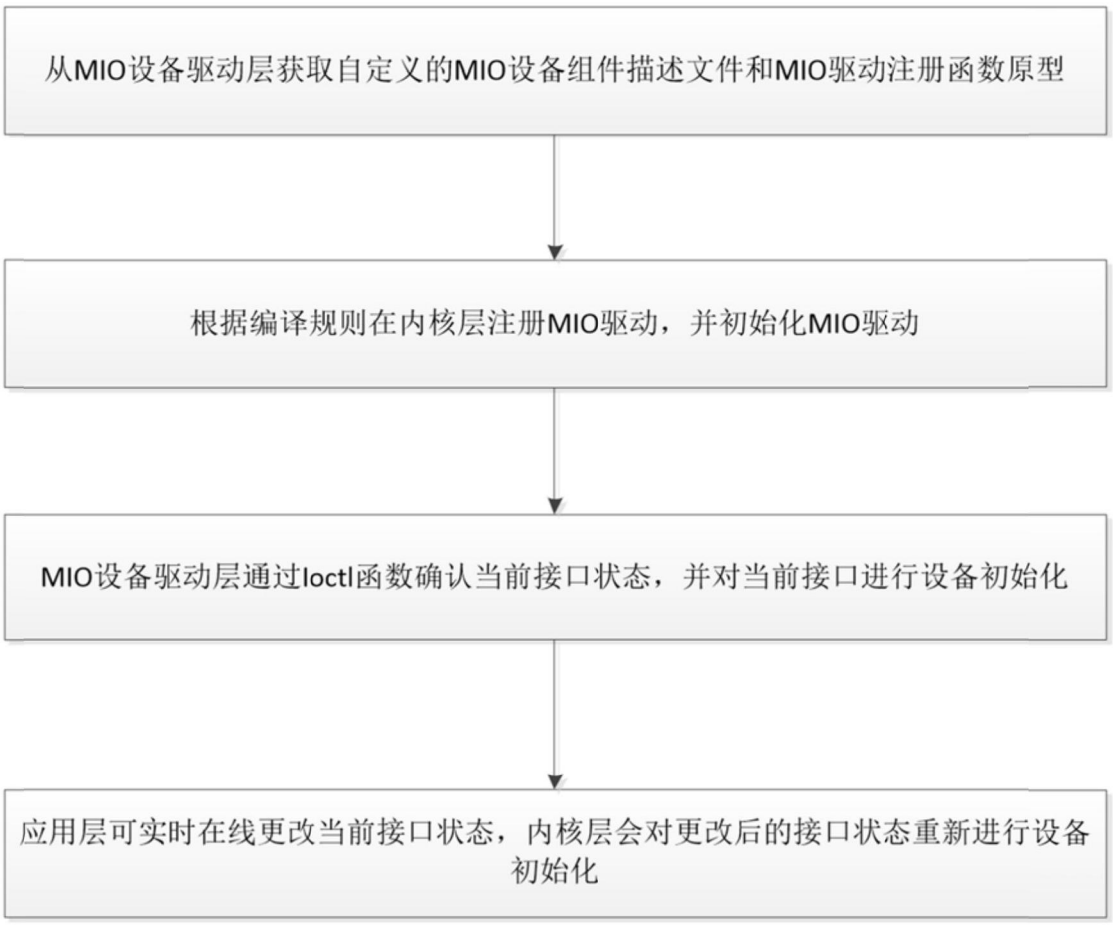

从MIO设备驱动层获取自定义的MIO设备组件描述文件和MIO驱动注册函数原型
根据编译规则在内核层注册MIO驱动，并初始化MIO驱动
MIO设备驱动层通过Ioctl函数确认当前接口状态，并对当前接口进行设备初始化
应用层可实时在线更改当前接口状态，内核层会对更改后的接口状态重新进行设备初始化

图1

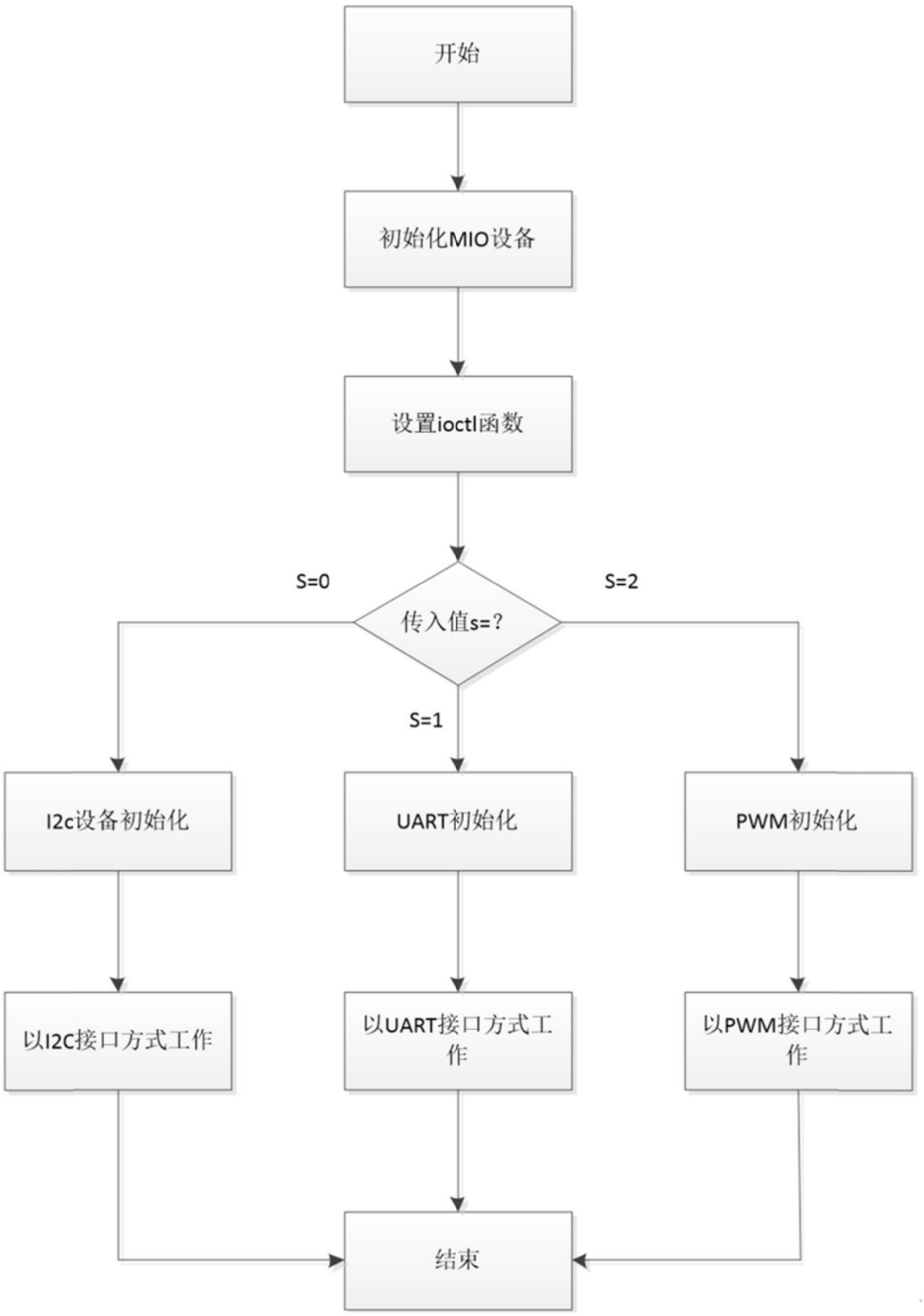

开始
初始化MIO设备
设置ioctl函数
传入值s=?
S=0
S=1
S=2
I2c设备初始化
UART初始化
PWM初始化
以I2C接口方式工作
以UART接口方式工作
以PWM接口方式工作
结束

图2