(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 116028122 A
(43)申请公布日 2023.04.28

(21)申请号 202211507058.6

(22)申请日 2022.11.29

(71)申请人 龙芯中科技术股份有限公司
地址 100095 北京市海淀区中关村环保科技示范园龙芯产业园2号楼

(72)发明人 陈劲宏 王建忠

(74)专利代理机构 北京同立钧成知识产权代理有限公司 11205
专利代理师 屈蓓 刘芳

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

权利要求书2页 说明书11页 附图3页

(54)发明名称

基于处理器的设备处理方法及设备

(57)摘要

本发明提供一种基于处理器的设备处理方法及设备。该方法包括:处理器获取设备树,设备树以分别挂载于设备控制器的各设备对象为节点,分别存储各设备对象各自对应的属性信息;处理器对设备树进行解析,分别得到设备对象各自对应的标识属性;处理器根据标识属性在驱动库中分别进行匹配,得到设备对象各自对应的设备驱动;处理器根据设备驱动分别对设备对象进行初始化。本发明实施例可以将设备对象的属性信息添加到设备树中,并通过设备树将设备对象的属性信息传输给处理器,以使处理器获取设备对象的设备驱动,以对其进行初始化。

CN 116028122 A

1.一种基于处理器的设备处理方法,其特征在于,所述方法包括:

所述处理器获取设备树;所述设备树以分别挂载于设备控制器的各设备对象为节点,分别存储各设备对象各自对应的属性信息;

所述处理器对所述设备树进行解析,分别得到所述设备对象各自对应的标识属性;

所述处理器根据所述标识属性在驱动库中分别进行匹配,得到所述设备对象各自对应的设备驱动;

所述处理器根据所述设备驱动分别对所述设备对象进行初始化。

2.根据权利要求1所述的方法,其特征在于,所述设备树以所述处理器为根节点,并以所述设备对象为所述根节点的子节点的存储格式,分别存储所述设备对象的属性信息。

3.根据权利要求2所述的方法,其特征在于,所述处理器对所述设备树进行解析,分别得到所述设备对象各自对应的标识属性,包括:

所述处理器在操作系统启动时,通过所述操作系统的内核代码对所述设备树进行解析,并遍历所述设备树的叶节点,分别得到所述设备对象对应的标识属性。

4.根据权利要求1所述的方法,其特征在于,所述处理器根据所述设备驱动分别对所述设备对象进行初始化,包括:

所述处理器根据所述设备驱动分别调用所述设备对象的初始化函数,注册所述设备对象对应的属性信息。

5.根据权利要求1至4任一项所述的方法,其特征在于,所述设备控制器位于所在设备的PCI总线上,所述方法还包括:

所述处理器从所述PCI总线的配置信息中获取所述设备控制器的属性信息;

所述处理器将所述设备控制器的属性信息注册后,通过PCI总线访问所述设备控制器。

6.根据权利要求1至4任一项所述的方法,其特征在于,还包括:

所述处理器根据所述设备对象的属性信息对所述设备对象进行访问,以调用所述设备对象执行既定的触发事件。

7.根据权利要求1所述的方法,其特征在于,所述设备对象包括:闪存,所述闪存通过SPI接口挂载在所述设备控制器下。

8.一种基于处理器的设备处理装置,其特征在于,包括:

设备树获取模块,用于获取设备树;所述设备树以分别挂载于设备控制器的各设备对象为节点,分别存储各设备对象各自对应的属性信息;

设备树解析模块,用于对所述设备树进行解析,分别得到所述设备对象各自对应的标识属性;

设备驱动获取模块,用于根据所述标识属性在驱动库中分别进行匹配,得到所述设备对象各自对应的设备驱动;

第一初始化模块,用于根据所述设备驱动分别对所述设备对象进行初始化。

9.一种计算设备,其特征在于,包括:至少一个处理器和存储器;

所述存储器存储计算机执行指令;

所述至少一个处理器执行所述存储器存储的计算机执行指令,使得所述计算设备实现如权利要求1至7任一项所述的方法。

10.一种计算机可读存储介质,其特征在于,所述计算机可读存储介质中存储有计算机

执行指令，当所述计算机执行指令被计算设备执行时用于实现如权利要求1至7任一项所述的方法。

11.一种计算机程序，其特征在于，所述计算机程序用于实现如权利要求1至7任一项所述的方法。

基于处理器的设备处理方法及设备

技术领域

[0001] 本发明涉及计算机技术领域，尤其涉及一种基于处理器的设备处理方法及设备。

背景技术

[0002] 在计算机领域中，处理器可以运行软件代码以实现数据处理过程。这里的软件代码可以包括操作系统和操作系统支持的应用软件。

[0003] 现有技术中，计算机可以包括通过总线结构相互连接的处理器和桥片，桥片可以通过设备控制器接入一个或多个外部的设备对象。由于处理器和设备对象之间通过桥片建立连接，处理器可以通过输入的设备对象的属性信息和桥片，对接入的设备对象进行访问。

[0004] 然而，在一些兼容性较差的操作系统中，例如，Linux操作系统，由于处理器无法获取到设备对象的属性信息，从而导致无法对设备对象进行初始化。

发明内容

[0005] 本发明提供一种基于处理器的设备处理方法及设备，用以在兼容性较差的操作系统中实现设备对象的初始化。

[0006] 第一方面，本发明提供一种基于处理器的设备处理方法，包括：

[0007] 所述处理器获取设备树；所述设备树以分别挂载于设备控制器的各设备对象为节点，分别存储各设备对象各自对应的属性信息；

[0008] 所述处理器对所述设备树进行解析，分别得到所述设备对象各自对应的标识属性；

[0009] 所述处理器对所述设备树进行解析，分别得到所述设备对象各自对应的标识属性。

[0010] 可选地，所述设备树以所述处理器为根节点，并以所述设备对象为所述根节点的子节点的存储格式，分别存储所述设备对象的属性信息。

[0011] 可选地，所述处理器对所述设备树进行解析，分别得到所述设备对象各自对应的标识属性，包括：

[0012] 所述处理器在操作系统启动时，通过所述操作系统的内核代码对所述设备树进行解析，并遍历所述设备树的叶节点，分别得到所述设备对象对应的标识属性。

[0013] 可选地，所述处理器根据所述设备驱动分别对所述设备对象进行初始化，包括：

[0014] 所述处理器根据所述设备驱动分别调用所述设备对象的初始化函数，注册所述设备对象对应的属性信息。

[0015] 可选地，所述设备控制器位于所在设备的PCI总线上，所述方法还包括：

[0016] 所述处理器从所述PCI总线的配置信息中获取所述设备控制器的属性信息；

[0017] 所述处理器将所述设备控制器的属性信息注册后，通过PCI总线访问所述设备控制器。

[0018] 可选地，还包括：

[0019] 所述处理器根据所述设备对象的属性信息对所述设备对象进行访问，以调用所述设备对象执行既定的触发事件。

[0020] 可选地，所述设备对象包括：闪存，所述闪存通过SPI接口挂载在所述设备控制器下。

[0021] 第二方面，本发明提供一种基于处理器的设备处理装置，包括：

[0022] 设备树获取模块，用于获取设备树；所述设备树以分别挂载于设备控制器的各设备对象为节点，分别存储各设备对象各自对应的属性信息；

[0023] 设备树解析模块，用于对所述设备树进行解析，分别得到所述设备对象各自对应的标识属性；

[0024] 设备驱动获取模块，用于根据所述标识属性在驱动库中分别进行匹配，得到所述设备对象各自对应的设备驱动；

[0025] 第一初始化模块，用于根据所述设备驱动分别对所述设备对象进行初始化。

[0026] 可选地，所述设备树以所述处理器为根节点，并以所述设备对象为所述根节点的子节点的存储格式，分别存储所述设备对象的属性信息。

[0027] 可选地，所述设备树解析模块还用于：

[0028] 在操作系统启动时，通过所述操作系统的内核代码对所述设备树进行解析，并遍历所述设备树的叶节点，分别得到所述设备对象对应的标识属性。

[0029] 可选地，所述第一初始化模块还用于：

[0030] 根据所述设备驱动分别调用所述设备对象的初始化函数，注册所述设备对象对应的属性信息。

[0031] 可选地，所述设备控制器位于所在设备的PCI总线上，所述装置还包括：

[0032] 第二初始化模块，用于从所述PCI总线的配置信息中获取所述设备控制器的属性信息；

[0033] 第一访问模块，用于将所述设备控制器的属性信息注册后，通过PCI总线访问所述设备控制器。

[0034] 可选地，所述处理器还包括：

[0035] 第二访问模块，用于根据所述设备对象的属性信息对所述设备对象进行访问，以调用所述设备对象执行既定的触发事件。

[0036] 可选地，所述设备对象包括：闪存，所述闪存通过SPI接口挂载在所述设备控制器下。

[0037] 第三方面，本发明实施例还提供了一种计算设备，包括：至少一个处理器和存储器；

[0038] 所述存储器存储计算机执行指令；

[0039] 所述至少一个处理器执行所述存储器存储的计算机执行指令，使得所述计算设备实现如第一方面所述的方法。

[0040] 第四方面，本发明实施例还提供了一种计算机可读存储介质，所述计算机可读存储介质中存储有计算机执行指令，当所述计算机执行指令被计算设备执行时用于实现第一方面所述的方法。

[0041] 第五方面，本发明实施例还提供了一种计算机程序，该计算机程序用于实现上述

第一方面的方法。

[0042] 本发明提供的基于处理器的设备处理方法及设备，该方法包括：处理器获取设备树，设备树以分别挂载于设备控制器的各设备对象为节点，分别存储各设备对象各自对应的属性信息；处理器对设备树进行解析，分别得到设备对象各自对应的标识属性；处理器根据标识属性在驱动库中分别进行匹配，得到设备对象各自对应的设备驱动；处理器根据设备驱动分别对设备对象进行初始化。本发明实施例可以将设备对象的属性信息添加到设备树中，并通过设备树将设备对象的属性信息传输给处理器，以使处理器获取设备对象的设备驱动，以对其进行初始化。

附图说明

[0043] 此处的附图被并入说明书中并构成本说明书的一部分，示出了符合本发明的实施例，并与说明书一起用于解释本发明的原理。

[0044] 图1是本发明实施例提供的一种计算设备的结构示意图；

[0045] 图2是本发明实施例提供的一种基于处理器的设备处理方法的步骤流程图；

[0046] 图3和图4是本发明实施例提供的两种设备树的结构示意图；

[0047] 图5为本发明实施例提供的一种基于处理器的设备处理装置的结构框图；

[0048] 图6为本发明实施例提供的另一种计算设备的结构框图。

[0049] 通过上述附图，已示出本发明明确的实施例，后文中将有更详细的描述。这些附图和文字描述并不是为了通过任何方式限制本发明构思的范围，而是通过参考特定实施例为本领域技术人员说明本发明的概念。

具体实施方式

[0050] 这里将详细地对示例性实施例进行说明，其示例表示在附图中。下面的描述涉及附图时，除非另有表示，不同附图中的相同数字表示相同或相似的要素。以下示例性实施例中所描述的实施方式并不代表与本发明相一致的所有实施方式。相反，它们仅是与如所附权利要求书中所详述的、本发明的一些方面相一致的装置和方法的例子。

[0051] 下面这几个具体的实施例可以相互结合，对于相同或相似的概念或过程可能在某些实施例中不再赘述。下面将结合附图，对本发明的实施例进行描述。

[0052] 本发明实施例可以用于计算设备中，计算设备是具有数据处理功能的任意设备。图1是本发明实施例提供的一种计算设备的结构示意图。参照图1所示，计算设备可以包括内存、处理器、桥片和设备对象。处理器和桥片之间可以通过总线连接，不同的处理器支持的总线不同，例如，当处理器为LOONGARCH架构处理器且桥片为7A桥片时，总线可以为HT(hyper transport，高速传输端对端总线技术)总线。

[0053] 上述桥片是用作连接桥梁的处理芯片，其中可以设置有一个或多个设备控制器，设备控制器可以设置于桥片的内部总线上。这里的内部总线可以为外设部件互连标准(peripheral component interconnect，PCI)总线，这里的设备对象可以通过串行外设接口(serial peripheral interface，SPI)挂载在设备控制器下。此时，设备控制器也可以称为SPI控制器，设备对象也可以称为SPI设备。

[0054] 设备对象可以理解为设备控制器的下属设备，桥片的地址空间中存在上述设备控

制器对应的地址和设备对象对应的地址，从而上述处理器可以通过访问桥片的地址空间，对设备控制器及其下挂载的设备对象进行访问。具体可以包括：首先，处理器从内存中加载软件代码；然后，处理器可以运行该软件代码，以通过PCI总线对设备控制器进行访问，以及，通过设备控制器对设备对象进行访问。

[0055] 本发明实施例中的上述处理器可以为CPU(central processing unit，中央处理单元)，本发明实施例的处理器可以为微处理器，处理器用于执行计算机指令以实现对数据的处理。上述设备对象可以是挂载在设备控制器上的任意外部设备，设备对象可以为一个或多个，例如，设备对象可以为闪存(Flash)，用于进行数据存储、备份和固化参数等，设备对象还可以为网卡、声卡等设备。其中，网卡设备用于对网络通信过程中所需要的网络数据进行处理，以实现网络通信，网络通信包括向网络发送数据请求，以及接收网络发送的响应数据。声卡设备用对音频数据进行处理，以实现音频播放，音频播放可以包括外放麦克风播放或耳机播放。

[0056] 在一些兼容性较差的操作系统中，例如，Linux操作系统，处理器通过PCI总线的配置信息获取设备控制器的属性信息，以初始化设备控制器。但是由于PCI总线的配置信息中不存在设备控制器下挂载的设备对象的属性信息，从而无法实现对设备对象的初始化。

[0057] 本发明实施例可以将设备对象的属性信息添加到设备树中，并通过设备树将设备对象的属性信息传输给处理器，以使处理器访问设备对象，并对其进行初始化。如此，实现了对设备对象的初始化。

[0058] 下面以具体地实施例对本发明的技术方案以及本发明的技术方案如何解决上述技术问题进行详细说明。下面这几个具体的实施例可以相互结合，对于相同或相似的概念或过程可能在某些实施例中不再赘述。下面将结合附图，对本发明的实施例进行描述。

[0059] 图2为本发明实施例提供的一种基于处理器的设备处理方法的步骤流程图，应用于图1所示的计算设备中。参照图2所示，该方法包括以下步骤。

[0060] S201：处理器获取设备树；设备树以分别挂载于设备控制器的各设备对象为节点，分别存储各设备对象各自对应的属性信息。

[0061] 其中，设备树用于存储计算设备中各硬件设备的属性信息，并且以分级的形式表示不同硬件设备之间的关系。设备树中处于同一级的节点对应的硬件设备是独立的，而设备树中的子节点对应的硬件设备需要父节点对应的硬件设备进行访问。例如，本公开实施例的设备对象的访问需要通过处理器执行，从而处理器对应的节点是是设备对象对应的节点的父节点。

[0062] 需要说明的是，上述设备树可以包括多级节点，可以将根节点称为第一级节点，根节点的子节点称为第二级节点，依次向下递增，直至叶子节点。上述第一级节点中可以存储处理器的属性信息，而设备控制器的属性信息和设备对象的属性信息可以存储在设备树的第N级节点中，N大于1。

[0063] 上述设备控制器的属性信息和设备对象的属性信息可以位于不同级节点中或同一级节点中。图3和图4是本公开实施例提供的两种设备树的结构示意图。在图3中，设备控制器的属性信息和设备对象的属性信息位于不同级节点中，在图4中，设备控制器的属性信息和设备对象的属性信息位于同一级节点中。其中，图3所示的设备树中设备控制器和设备对象对应父子节点，需要通过设备控制器的节点访问设备对象的节点，从而适用于需要同

时对设备控制器的节点和设备对象的节点进行访问的场景。而图4所示的设备树中设备控制器和设备对象对应兄弟节点，可以适用于单独对设备控制器的节点进行访问的场景，或，单独对设备对象的节点进行访问的场景。

[0064] 参照图3所示，设备树包括N1至N6共六个节点。其中，N1是第一级节点，用于存储处理器的属性信息。N2和N3为第二级节点，N2用于存储设备控制器DC1的属性信息，N3用于存储设备控制器DC2的属性信息。N4是第三级节点，用于存储挂载在设备控制器DC1下的设备对象DO1的属性信息。N5是第三级节点，用于存储挂载在设备控制器DC2下的设备对象DO2的属性信息。N6是第三级节点，用于存储挂载在设备控制器DC2下的设备对象DO3的属性信息。

[0065] 从上述图3中可以看出，设备控制器的属性信息位于第二级节点中，而设备对象的属性信息位于第三级节点中。而设备对象所对应的节点与其所在的设备控制器所对应的节点是父子关系。

[0066] 参照图4所示，设备树包括N1至N6共六个节点，并且，N1是第一级节点，N2、N3、N4、N5和N6均为是第二级节点。与图3所示的N1至N6节点一样，N1用于存储处理器的属性信息，N2用于存储设备控制器DC1的属性信息，N3用于存储设备控制器DC2的属性信息，N4用于存储挂载在设备控制器DC1下的设备对象DO1的属性信息，N5用于存储挂载在设备控制器DC2下的设备对象DO2的属性信息，N6用于存储挂载在设备控制器DC2下的设备对象DO3的属性信息。

[0067] 从上述图4中可以看出，设备控制器的属性信息和设备对象的属性信息位于同一级节点中，并且设备控制器所对应的节点与其所在的设备控制器所对应的节点是兄弟关系。

[0068] 需要说明的是，对于图3中的设备树，处理器需要先获取设备控制器的属性信息，然后再根据设备控制器的属性信息获取到设备对象的属性信息。具体地，首先，处理器解析设备树中的第二级节点，得到所有设备控制器对应的标识属性，该标识属性中包括设备控制器的属性信息；然后，将设备控制器的标识属性在设备驱动库进行匹配，得到设备控制器的设备驱动；最后，通过设备控制器的设备驱动，从设备树中获取设备控制器下挂载的设备对象的属性信息。

[0069] 而对于图4中的设备树，处理器可以直接获取设备对象的属性信息，实现了跨过设备控制器直接获取设备对象的属性信息的目标。对于图4所示的设备树，如果处理器不需要从设备树中获取设备控制器的属性信息，那么可以从图4所示的设备树中删除设备控制器对应的节点N2和N3，使设备树的第二级节点均用于存储设备对象的属性信息。如此，可以保证处理器在不获取设备控制器的属性信息的情况下，也可以直接获取设备对象的属性信息，实现对设备对象的初始化。

[0070] 当然，上述设备树可以以文件的形式存储，可以称为设备树文件，例如，.dts文件，是一种通过ASCII码描述设备树的文本文件，位于操作系统的/arch/arm/boot/dts目录中。

[0071] 操作系统在生成设备树之后，还可以根据设备控制器和/或设备对象的删除、新增更新设备树，操作系统可以通过驱动程序获取的信息建立设备树。

[0072] 在操作系统的启动过程中，先生成设备树再根据设备树执行处理器、设备控制器以及设备对象的初始化。

[0073] 在图3所示的设备树的生成过程中，首先，操作系统内核中的扫描进程可以扫描硬

件设备，得到处理器的属性信息、设备控制器的属性信息以及设备对象的属性信息。然后，操作系统内核根据处理器的属性信息生成一个节点N1，并将该节点N1作为设备树的根节点。再然后，操作系统内核根据每个设备控制器的属性信息生成一个节点，例如，两个设备控制器可以得到节点N2和N3，并将该N2和N3作为根节点的两个子节点。最后，操作系统内核针对每个设备控制器生成其子节点，例如，对于设备控制器DC1，生成挂载在设备控制器DC1下的设备对象DO1的节点N4，并作为N2的子节点。对于设备控制器DC2，生成挂载在设备控制器DC2下的设备对象DO2和DO3的节点N5和N6，并作为N3的两个子节点，这样就生成了如图3所示的设备树。

[0074] 对于图3所示的设备树，在将一个新的设备对象添加到一个设备控制器下时，设备控制器可以检测到该设备对象，并从设备对象的配置信息中获取该设备对象的属性信息，以将该设备对象的属性信息添加到设备树的第三级节点中。

[0075] 对于图3所示的设备树，在将一个设备对象从设备控制器下拔出时，设备控制器可以检测到拔出操作，并从设备树的第三级节点中删除该设备对象对应的节点。例如，在将设备对象DO1拔出时，设备控制器DC1可以从设备树中删除节点N4。

[0076] 在图4所示的设备树的生成过程中，首先，操作系统内核中的扫描进程可以扫描硬件设备，得到处理器的属性信息、设备控制器的属性信息以及设备对象的属性信息。然后，操作系统内核根据处理器的属性信息生成一个节点N1，并将该节点N1作为设备树的根节点。再然后，操作系统内核根据每个设备控制器的属性信息生成一个节点，例如，两个设备控制器可以得到节点N2和N3，并将该N2和N3作为根节点的第一个子节点和第二个子节点。最后，操作系统内核根据设备对象的属性信息生成一个节点，例如，三个设备对象可以得到三个节点N4、N5和N6，作为根节点的第三至第五个子节点，如此可以得到图4所示的设备树。

[0077] 对于图4所示的设备树，在将一个新的设备对象添加到一个设备控制器下时，设备控制器可以检测到该设备对象，并从设备对象的配置信息中获取该设备对象的属性信息，以将该设备对象的属性信息添加到图4所示的设备树中。具体地，首先，设备控制器将设备对象的属性信息生成一个节点；然后，设备控制器将该节点作为处理器的一个新的子节点添加到图4所示的设备树中。

[0078] 对于图4所示的设备树，在将一个设备对象从设备控制器下拔出时，设备控制器可以检测到拔出操作，并从设备树的第二级节点中删除该设备对象对应的节点。例如，在将设备对象DO1拔出时，设备控制器DC1可以从图4所示的设备树中删除节点N4。

[0079] 在得到上述图3或图4所示的设备树之后，可以遍历上述设备树，以获取目标设备控制器的属性信息或设备对象的属性信息。

[0080] 对于图3所示的设备树，在获取目标设备控制器的属性信息时对应的遍历过程是：从设备树中按照顺序每次获取一个第二级节点(例如N2)，以将该第二级节点N2中的属性信息与目标设备控制器的标识进行匹配，如果匹配成功，则将当前第二级节点N2中的属性信息作为目标设备控制器的属性信息。否则，继续获取下一个第二级节点N3中的属性信息，以再次和目标设备控制器的标识进行匹配，直至匹配成功或最后一个第二级节点N3匹配结束。对于图3所示的设备树，在获取设备对象的属性信息时，可以先确定设备对象的目标设备控制器，以将目标设备控制器对应节点(例如N3)的子节点(例如N5)中存储的属性信息与设备对象的标识进行匹配。如果匹配成功，则将目标设备控制器对应节点N3的当前子节点

N5中的属性信息作为设备对象的属性信息。否则，继续获取该目标设备控制器对应节点的下一个子节点N6中的属性信息，以再次和目标设备控制器的标识进行匹配，直至匹配成功或目标设备控制器对应节点N3的最后一个子节点N6匹配结束。

[0081] 对于图4所示的设备树，获取目标设备控制器的属性信息或设备对象的属性信息的过程类似，以设备对象为例，对应的遍历过程是：从设备树中按照顺序每次获取一个第二级节点(例如N2)，以将该第二级节点N2中的属性信息与设备对象的标识进行匹配，如果匹配成功，则将当前第二级节点N2中的属性信息作为设备对象的属性信息。否则，继续获取下一个第二级节点N3中的属性信息，以再次和设备对象的标识进行匹配，直至匹配成功或最后一个第二级节点N6匹配结束。

[0082] 可以看出，上述设备树中存储有各种硬件设备的属性信息，不同硬件设备具有不同特征，从而对应不同的属性信息。例如，处理器的属性信息可以包括但不限于：频率、处理器缓存大小、指令集类型、功耗、处理器数量等。频率可以包括但不限于：主频、倍频、总线类型、总线频率。

[0083] 设备控制器的属性信息包括设备控制器的地址空间、设备控制器的标识、设备控制器下挂载的设备对象数量等。

[0084] 上述设备对象的属性信息是与访问策略相关联的信息，在属性信息不同时，可以对应不同的访问策略。这里的属性信息可以包括但不限于：兼容性(compatible)、总线编号(bus_num)、串行外设接口的时钟信息，其中，时钟信息可以包括时钟极性(SPI-CPOL)、时钟相位(SPI-CPHA)和时钟是否为高电平片选信号(SPI-CS-HIGH)、SPI三线(SPI-3WIRE)。当上述设备对象为存储设备时，还可以包括存储设备中寄存器的地址。

[0085] 其中，兼容性的属性值是一个字符串列表，用于将目标外部设备和驱动程序的信息绑定起来，以从字符串列表中选取目标外部设备所要使用的驱动程序。驱动程序的信息可以对应有驱动程序的供应方、驱动程序的名称、驱动程序的版本等。可以理解的是，处理器通过驱动程序访问目标外部设备，从而目标外部设备所兼容的驱动程序会影响对目标外部设备的访问，也就是说，处理器与目标外部设备的兼容性相关。

[0086] 总线编号是表示目标外部设备挂载的设备控制器所在的总线的编号。处理器需要通过设备控制器所在的总线和设备控制器访问目标外部设备，从而在设备控制器所在的总线不同时，对目标外部设备的访问策略也可能不同，也就是说，访问策略与总线编号相关。

[0087] 寄存器的地址包括起始地址和长度，这样就可以根据起始地址和长度确定结束地址，结束地址为起始地址和长度之和，从而起始地址和长度即可用于表示寄存器的地址范围。处理器在访问目标外部设备时可以将一些数据暂时存储在对应的寄存器中，不同目标外部设备对应的寄存器的地址可以不同。从而对目标外部设备的访问策略与目标外部设备对应的寄存器的地址相关。

[0088] 在一些实施方式中，设备树可以以设备控制器的属性信息存储格式存储设备对象的属性信息。如此，处理器可以复用已有的设备树解析算法，不需要额外开发解析代码，即可从设备树中解析得到设备对象的标识属性，降低了代码复杂度和开发成本。

[0089] S202：处理器对设备树进行解析，分别得到设备对象各自对应的标识属性。

[0090] 其中，处理器可以按照设备树的预设结构对设备树进行解析，得到设备对象的标识属性，也就是说，在确定结构之后，即可确定设备树中的每个节点存储的信息，从而可以

实现解析，以从设备树中提取到属性信息，用标识属性表示。这里的标识属性可以理解为软件代码中的对象，例如，Linux操作系统中的platform device对象。

[0091] 本发明实施例可以复用操作系统中对设备树的处理逻辑实现，可以降低开发成本，降低代码复杂度。

[0092] 具体地，处理器在启动计算设备的操作系统的过程中，可以通过操作系统的内核代码对设备树进行解析，并遍历设备树的叶节点，分别得到设备对象对应的标识属性，并执行S203和S204得到对设备对象进行初始化。可以看出，本发明实施例在启动操作系统时即可获取到设备对象的属性信息，以为操作系统在后续运行过程中访问设备对象做好准备，可以提高访问设备对象的效率。并且，通过内核代码解析设备树可以更好的保证设备树的安全性。

[0093] 其中，遍历设备树的叶节点可以理解为逐个获取设备树中的各个叶节点，每个叶节点可以得到一个设备对象的标识属性。

[0094] S203：处理器根据标识属性在驱动库中分别进行匹配，得到设备对象各自对应的设备驱动。

[0095] 其中，设备驱动可以理解为驱动函数，用于对设备对象进行驱动，其中可以包括对设备对象进行驱动的各种函数。例如，Linux操作系统中的platform driver函数。一个设备对象的设备驱动可以位于一个或多个驱动文件中，所有设备对象的驱动文件可以存储在一个设备对象驱动目录中。同样地，还可以设置控制器驱动目录，以存储所有设备控制器的驱动文件。

[0096] 设备对象的标识属性和设备驱动是按照一定映射规则建立的对应关系，设备驱动名称可以为驱动文件的名称，设备驱动名称中可以和设备对象的标识属性名称中均包括设备对象名称，例如，设备驱动名称可以为：name_driver，标识属性可以为name_device，其中，name为设备对象名称。从而，在将设备对象的标识属性在驱动库中匹配时，判断驱动库中的每个驱动文件的名称中是否和设备对象的标识属性名称中包括同一个设备对象名称，如果包括，则确定匹配成功，否则匹配失败。本发明实施例对设备对象的标识属性和设备驱动之间的映射规则不加限制。

[0097] 现有技术中，由于设备树中不存在目标外部设备的属性信息，从而驱动程序中也不存在目标外部设备的设备对象和对该目标外部设备的设备驱动。而本发明实施例在设备树中添加目标外部设备的属性信息之后，需要添加设备对象和设备驱动，并建立和设备对象之间的映射关系。

[0098] S204：处理器根据设备驱动分别对设备对象进行初始化。

[0099] 其中，上述设备驱动中可以包括设备对象的初始化函数，以使处理器调用设备驱动中的该初始化函数，在操作系统内核的设备注册列表中注册设备对象对应的属性信息。这里的初始化函数与操作系统相关，例如，Linux操作系统中的probe函数。如此，可以在初始化目标设备对象的时候注册设备对象的属性信息，以供后续访问设备对象时使用，可以保证设备对象的正常访问。

[0100] 在注册上述设备对象的属性信息之后，处理器可以根据设备对象的属性信息对设备对象进行访问，以调用设备对象执行既定的触发事件。具体地，处理器可以调用上述S203的设备驱动访问设备对象。

[0101] 其中,触发事件包括但不限于:对设备对象的写入、读取等事件。

[0102] 可选地,上述设备控制器位于桥片中的PCI总线上,此时,处理器可以从PCI总线的配置信息中获取设备控制器的属性信息;然后,处理器将设备控制器的属性信息注册后通过PCI总线访问设备控制器。

[0103] 其中,PCI总线的配置信息中可以包括桥片中所有设备控制器的属性信息。处理器可以调用设备控制器的初始化函数,以将从PCI总线的配置信息中获取的设备控制器的属性信息,注册到操作系统内核中的设备列表中。这样,在需要访问设备控制器时,处理器调用操作系统中的访问接口,以根据注册的设备控制器的属性信息向PCI总线发送访问请求,实现对设备控制器的访问。

[0104] 可以看出,设备控制器的属性信息不需要从设备树中获取,而是可以从配置信息中获取到,可以降低初始化设备控制器的复杂度,提高初始化设备控制器的效率。

[0105] 本发明实施例的一种应用场景为,操作系统为Linux操作系统,设备对象包括:闪存Flash,闪存通过SPI接口挂载在7A桥片的设备控制器下。因此,设备控制器可以称为SPI设备控制器,闪存称为SPI闪存。

[0106] 基于上述场景,在操作系统启动的过程中,PCI扫描进程可以扫描得到Flash的属性信息和处理器的属性信息,以将处理器的属性信息存储到设备树的根节点,以及将每个flash的属性信息存储到一个第二级节点中,这样就得到了一个设备树。在得到设备树之后,操作系统的内核代码可以遍历该设备树的第二级节点,得到flash的标识属性,标识属性名称为SPI flash_device,以将标识属性名称SPI flash_device与驱动库中的设备驱动名称进行匹配,以获取到设备驱动SPI flash_driver。处理器可以根据SPI_flash_driver进入probe函数中以将flash的属性信息注册到PCI总线上。到此就完成了对flash的初始化过程。

[0107] 在需要访问flash时,处理器会发送给设备控制器要访问的目标flash的标识,设备控制器可以在PCI总线上存在该目标flash的属性信息时,完成对该flash的访问过程,也就实现了对flash的写入或读取过程。

[0108] 综上所述,本发明可以为7A桥片挂载多个闪存进行数据存储、固化参数等功能提供支持,使处理器适应更多存储方案。

[0109] 综上所述,本发明实施例的处理器中运行有设备控制器的驱动程序,设备控制器的驱动程序用于对设备控制器进行访问控制。该设备控制器的驱动程序可以通过PCI总线获取设备控制器的属性信息,并不需要通过设备树传输设备控制器的属性信息。但PCI总线的配置信息中不存在目标外部设备的属性信息。在这种场景下,由于设备控制器不使用设备树,因此无法通过设备控制器的驱动程序进入初始化Probe函数,以获取目标外部设备的属性信息。本发明实施例可以将目标外部设备的属性信息写入设备树中,并编写对应的设备驱动platform driver函数,以与设备对象建立映射关系。通过这种方式,可以进入初始化probe函数,以获取到目标外部设备的属性信息。如此,可以使处理器在Linux操作系统下,跨过设备控制器获取到目标外部设备的属性信息,以实现对目标外部设备的访问。

[0110] 图5为本发明实施例提供的一种基于处理器的设备处理装置的结构框图。参照图5所示,基于处理器的设备处理装置400包括以下模块。

[0111] 设备树获取模块401,用于获取设备树;所述设备树以分别挂载于设备控制器的各

设备对象为节点，分别存储各设备对象各自对应的属性信息。

[0112] 设备树解析模块402，用于对所述设备树进行解析，分别得到所述设备对象各自对应的标识属性。

[0113] 设备驱动获取模块403，用于根据所述标识属性在驱动库中分别进行匹配，得到所述设备对象各自对应的设备驱动。

[0114] 第一初始化模块404，用于根据所述设备驱动分别对所述设备对象进行初始化。

[0115] 可选地，所述设备树以所述处理器为根节点，并以所述设备对象为所述根节点的子节点的存储格式，分别存储所述设备对象的属性信息。

[0116] 可选地，所述设备树解析模块402还用于：

[0117] 在操作系统启动时，通过所述操作系统的内核代码对所述设备树进行解析，并遍历所述设备树的叶节点，分别得到所述设备对象对应的标识属性。

[0118] 可选地，所述第一初始化模块404还用于：

[0119] 根据所述设备驱动分别调用所述设备对象的初始化函数，注册所述设备对象对应的属性信息。

[0120] 可选地，所述设备控制器位于所在设备的PCI总线上，所述装置还包括：

[0121] 第二初始化模块，用于从所述PCI总线的配置信息中获取所述设备控制器的属性信息。

[0122] 第一访问模块，用于将所述设备控制器的属性信息注册后，通过PCI总线访问所述设备控制器。

[0123] 可选地，所述装置还包括：

[0124] 第二访问模块，用于根据所述设备对象的属性信息对所述设备对象进行访问，以调用所述设备对象执行既定的触发事件。

[0125] 可选地，所述设备对象包括：闪存，所述闪存通过SPI接口挂载在所述设备控制器下。

[0126] 图6为本发明实施例示例性示出的一种计算设备600的结构框图。该计算设备600包括存储器602和至少一个处理器601。

[0127] 其中，存储器602存储计算机执行指令。

[0128] 至少一个处理器601执行存储器602存储的计算机执行指令，使得计算设备600实现前述图2中的方法。

[0129] 此外，该计算设备还可以包括接收器603和发送器604，接收器603用于接收从其余装置或设备的信息，并转发给处理器601，发送器604用于将信息发送到其余装置或设备。

[0130] 上述计算设备是图2所示的方法对应的装置实施例，具体可以参照图2所示的方法实施例的详细说明，在此不再赘述。

[0131] 本发明实施例还提供了一种计算机可读存储介质，所述计算机可读存储介质中存储有计算机执行指令，当所述计算机执行指令被计算设备执行时用于实现如图2所示的方法。

[0132] 本发明实施例还提供了一种计算机程序，计算机程序用于实现前述图2所示的方法。

[0133] 以上描述仅为本发明的较佳实施例以及对所运用技术原理的说明。本领域技术人

员应当理解，本发明中所涉及的公开范围，并不限于上述技术特征的特定组合而成的技术方案，同时也应涵盖在不脱离上述公开构思的情况下，由上述技术特征或其等同特征进行任意组合而形成的其它技术方案。例如上述特征与本发明中公开的(但不限于)具有类似功能的技术特征进行互相替换而形成的技术方案。

[0134] 此外，虽然采用特定次序描绘了各操作，但是这不应当理解为要求这些操作以所示出的特定次序或以顺序次序执行来执行。在一定环境下，多任务和并行处理可能是有利的。同样地，虽然在上面论述中包含了若干具体实现细节，但是这些不应当被解释为对本发明的范围的限制。在单独的实施例的上下文中描述的某些特征还可以组合地实现在单个实施例中。相反地，在单个实施例的上下文中描述的各种特征也可以单独地或以任何合适的子组合的方式实现在多个实施例中。

[0135] 尽管已经采用特定于结构特征和/或方法逻辑动作的语言描述了本主题，但是应当理解所附权利要求书中所限定的主题未必局限于上面描述的特定特征或动作。相反，上面所描述的特定特征和动作仅仅是实现权利要求书的示例形式。

[0136] 本申请中说明书和权利要求书及上述附图中的术语“第一”、“第二”、“第三”等是用于区别类似或同类的对象或实体，而不必然意味着限定特定的顺序或先后次序，除非另外注明(Unless otherwise indicated)。应该理解这样使用的用语在适当情况下可以互换，例如能够根据本申请实施例图示或描述中给出那些以外的顺序实施。

[0137] 此外，术语“包括”和“具有”以及他们的任何变形，意图在于覆盖但不排他的包含，例如，包含了一系列组件的产品或设备不必限于清楚地列出的那些组件，而是可包括没有清楚地列出的或对于这些产品或设备固有的其它组件。

[0138] 本申请中使用的术语“模块”，是指任何已知或后来开发的硬件、软件、固件、人工智能、模糊逻辑或硬件或/和软件代码的组合，能够执行与该元件相关的功能。

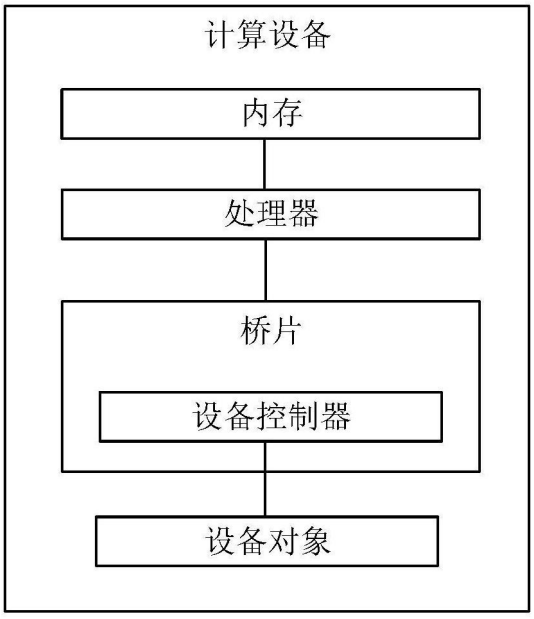
计算设备
内存
处理器
桥片
设备控制器
设备对象

图1

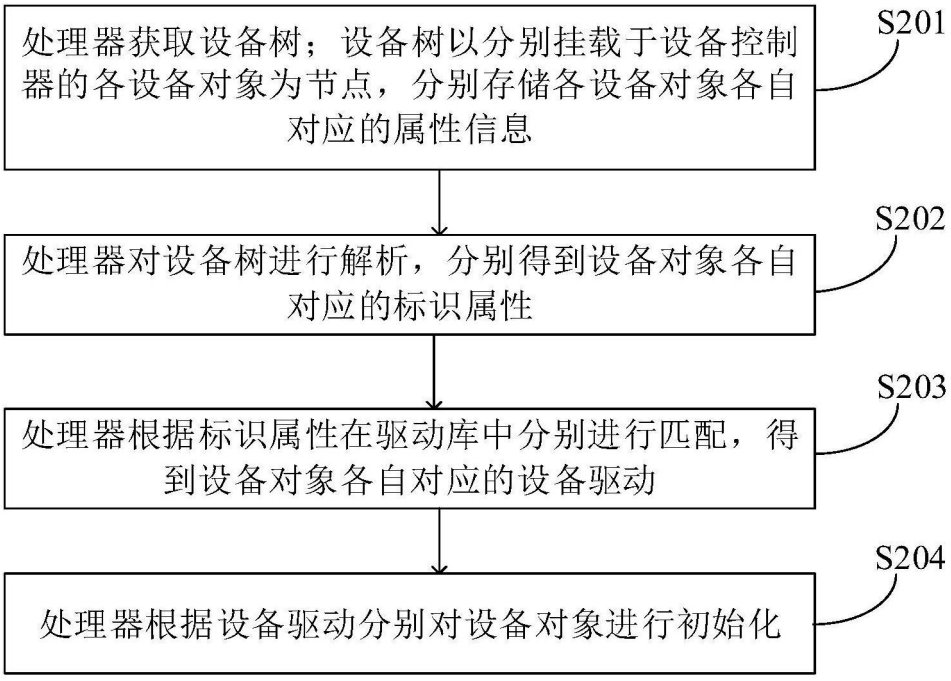
处理器获取设备树；设备树以分别挂载于设备控制器的各设备对象为节点，分别存储各设备对象各自对应的属性信息
S201
处理器对设备树进行解析，分别得到设备对象各自对应的标识属性
S202
处理器根据标识属性在驱动库中分别进行匹配，得到设备对象各自对应的设备驱动
S203
处理器根据设备驱动分别对设备对象进行初始化
S204

图2

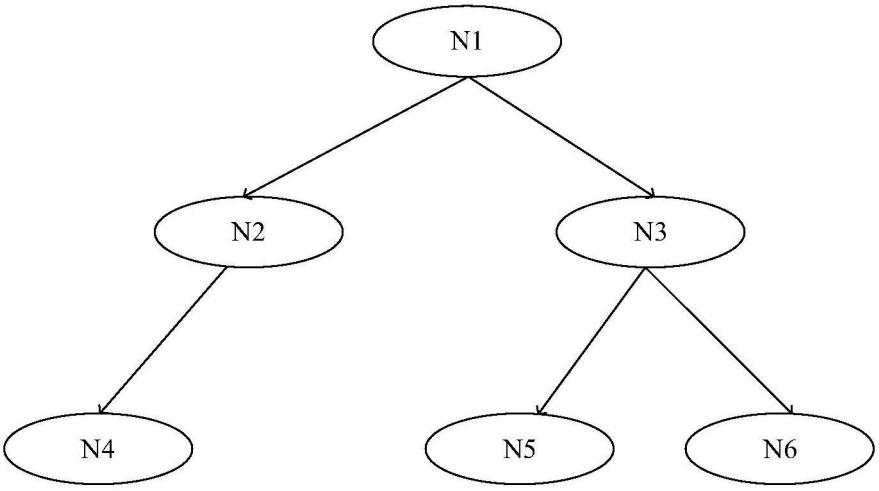

图3

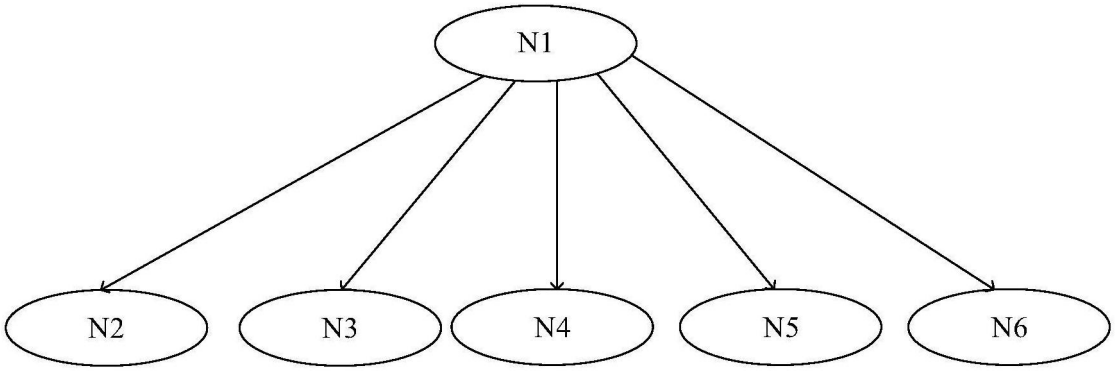

图4

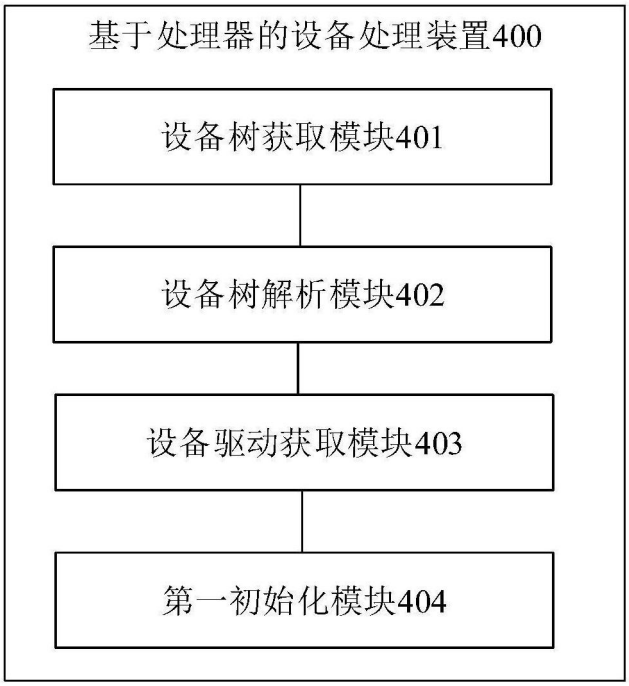
基于处理器的设备处理装置400
设备树获取模块401
设备树解析模块402
设备驱动获取模块403
第一初始化模块404

图5

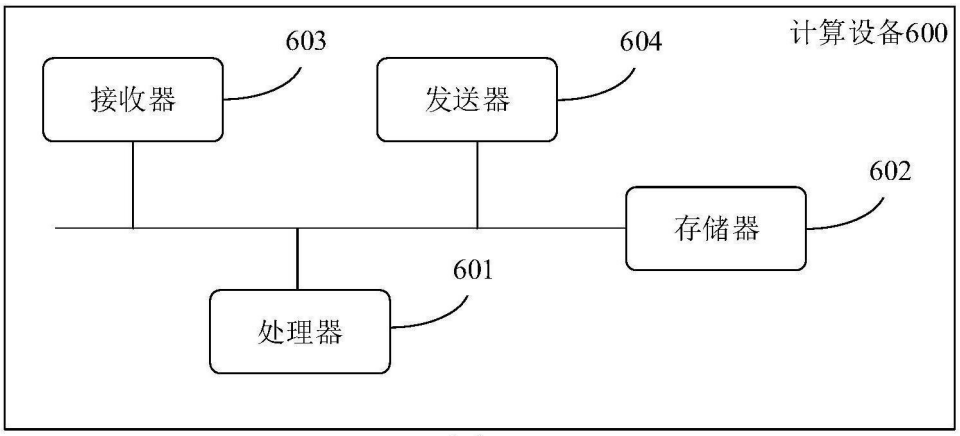
计算设备600
603
接收器
604
发送器
602
存储器
601
处理器

图6