



# (12) 发明专利申请

(10) 申请公布号 CN 114691240 A

(43) 申请公布日 2022.07.01

(21) 申请号 202210348686.8

(22) 申请日 2022.04.01

(71) 申请人 武汉深之度科技有限公司

地址 430206 湖北省武汉市东湖新技术开发区高新大道999号未来科技城B5栋11层(自贸区武汉片区)

(72) 发明人 聂鳌 沈寒

(74) 专利代理机构 北京瀚方律师事务所 11774

专利代理师 姜莹

(51) Int.Cl.

G06F 9/445 (2018.01)

G06F 8/41 (2018.01)

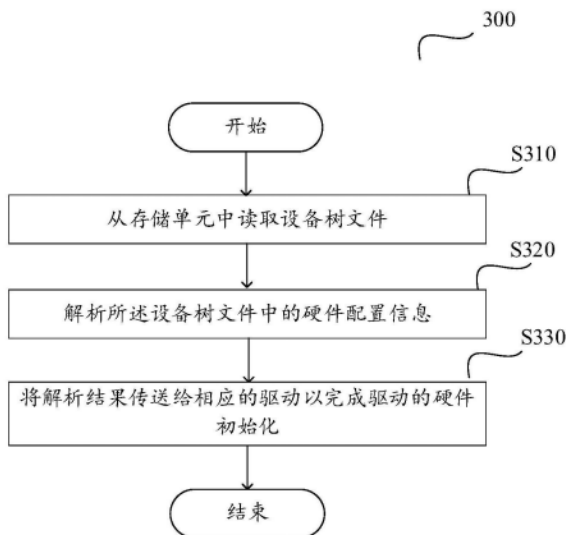
权利要求书2页 说明书10页 附图5页

## (54) 发明名称

驱动硬件配置信息加载方法、装置和系统，以及计算设备

## (57) 摘要

本发明公开了一种驱动硬件配置信息加载方法、装置和系统，以及计算设备和可读存储介质。其中，驱动硬件配置信息加载方法在计算设备中执行，包括动态驱动硬件配置信息加载步骤：从存储单元中读取设备树文件，所述设备树文件由开发人员制备，且用于描述开发板上所有驱动的硬件配置信息；解析所述设备树文件中的硬件配置信息；将解析结果传送给相应的驱动以完成驱动的硬件初始化。根据本发明实施方式，只需要制备设备树文件，并在加载驱动时读取设备树文件，就可以获知驱动的相关硬件配置信息。不需要将设备树文件编译成特定的文件，系统就可以直接识别。



1. 一种驱动硬件配置信息加载方法,在计算设备中执行,包括动态驱动硬件配置信息加载步骤:

从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;

解析所述设备树文件中的硬件配置信息;

将解析结果传送给相应的驱动以完成驱动的硬件初始化。

2. 一种驱动硬件配置信息加载方法,在计算设备中执行,包括:

判断设备资源是否大于等于阈值;

在设备资源小于阈值的情况下,判断采用静态驱动硬件配置信息加载方式,其中,将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、驱动的系统代码一起编译、下载到设备中;

在设备资源大于等于阈值的情况下,执行以下步骤:

从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;

解析所述设备树文件中的硬件配置信息;

将解析结果传送给相应的驱动以完成驱动的硬件初始化。

3. 如权利要求1或2所述的驱动硬件配置信息加载方法,从存储单元中读取设备树文件包括:从存储单元中读取修改后的设备树文件,所述修改后的设备树文件是根据驱动的硬件配置信息而修改的。

4. 如权利要求3所述的驱动硬件配置信息加载方法,还包括:识别驱动的硬件配置信息,且在识别出驱动的硬件配置信息发生变化的情况下,通知相关人员修改设备树文件。

5. 一种驱动硬件配置信息加载装置,包括:

读取模块,用于从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;

解析模块,用于解析所述设备树文件中的硬件配置信息;

传送模块,用于将解析结果传送给相应的驱动以完成驱动的硬件初始化。

6. 如权利要求5所述的驱动硬件配置信息加载装置,所述读取模块用于从存储单元中读取修改后的设备树文件,所述修改后的设备树文件是根据驱动的硬件配置信息而修改的。

7. 如权利要求6所述的驱动硬件配置信息加载装置,还包括:识别模块,用于识别驱动的硬件配置信息,且在识别出驱动的硬件配置信息发生变化的情况下,通知相关人员修改设备树文件。

8. 一种驱动硬件配置信息加载系统,包括:

判断装置,用于判断设备资源是否大于等于阈值,且在判断出设备资源小于阈值的情况下,触发静态设备树文件加装置,在判断出设备资源大于等于阈值的情况下,触发动态驱动硬件配置信息加载装置;

静态驱动硬件配置信息加载装置,用于将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、驱动的系统代码一起编译、下载到设备中;

动态驱动硬件配置信息加载装置,包括:

读取模块,用于从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动件的硬件配置信息;

解析模块,用于解析所述设备树文件中的硬件配置信息;

传送模块,用于将解析结果传送给相应的驱动件以完成驱动件的硬件初始化。

9. 一种计算设备,包括:

至少一个处理器和存储有程序指令的存储器;

当所述程序指令被所述处理器读取并执行时,使得所述计算设备执行如权利要求1-4中任一项所述的驱动件硬件配置信息加载方法。

10. 一种存储有程序指令的可读存储介质,当所述程序指令被计算设备读取并执行时,使得所述计算设备执行如权利要求1-4中任一项所述的驱动件硬件配置信息加载方法。

## 驱动硬件配置信息加载方法、装置和系统,以及计算设备

### 技术领域

[0001] 本发明涉及驱动加载的技术领域,尤其是一种驱动硬件配置信息加载方法、装置和系统,以及计算设备和可读存储介质。

### 背景技术

[0002] 随着物联网时代的到来,物联网使用的MCU芯片技术也蓬勃发展,所能提供的外设硬件资源也愈加丰富。MCU可以支持各种外设——其中相同外设提供多个相同的功能模块,以满足在实际使用中的各种需求。

[0003] 目前主要采用两种方式来加载外设。在第一种方式中,在驱动代码中配置硬件资源,然后将其进行编译并下载到设备中。当进行设备驱动开发时,需要对每个设备进行定制化驱动硬件信息的代码修改,即每个驱动的硬件配置信息是在驱动代码中直接指定后编译下载的。一旦编译后,生成的系统镜像外设配置就固定下来且无法改变。若外设硬件信息发生变化,就必须在驱动代码中修改对应配置,然后重新编译。

[0004] 在第二种方式中,引入了设备树的概念,这是由于外部硬件种类众多,为了尽可能减少硬件资源配置引起驱动代码的修改。其中,将驱动中外设的硬件配置信息写入设备树文件,采用工具链将设备树文件编译成相关头文件,然后再和内核、驱动的系统代码一起编译下载。该第二种方式虽然解决了硬件信息发生变化导致驱动文件的修改问题,然而仍然需要对设备树文件进行重新编译和下载。也就是说,对硬件配置信息进行解析然后生成头文件是在编译之前完成的。每一次修改设备树文件,都必须重新编译后才能在设备上运行,以完成相应的驱动硬件初始化。

[0005] 上述两种方法都无法绕过设备资源信息必须参与编译这一环节,都需要重新编译代码、烧写到设备。如果设备数量较多,且使用的外设驱动不相同,就必须根据设备的实际使用外设模块修改驱动中对应的硬件信息,然后重新编译下载。

### 发明内容

[0006] 为此,本发明提供了一种驱动硬件配置信息加载方法、装置和系统,以及计算设备和可读存储介质,以力图解决或者至少缓解上面存在的至少一个问题。

[0007] 根据本发明一方面,提供一种驱动硬件配置信息加载方法,在计算设备中执行,包括动态驱动硬件配置信息加载步骤:从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;解析所述设备树文件中的硬件配置信息;将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0008] 根据本发明另一方面,提供一种驱动硬件配置信息加载方法,在计算设备中执行,包括:判断设备资源是否大于等于阈值;在设备资源小于阈值的情况下,判断采用静态驱动硬件配置信息加载方式,其中,将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、驱动的系统代码一起编译、下载到设备中;在设备资源大于等于阈值的情况下,执行以下步骤:从存储单元中读取设备树文件,所述设备树文件由开发人员

制备,且用于描述开发板上所有驱动的硬件配置信息;解析所述设备树文件中的硬件配置信息;将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0009] 根据本发明另一方面,提供一种驱动硬件配置信息加载装置,包括:读取模块,用于从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;解析模块,用于解析所述设备树文件中的硬件配置信息;传送模块,用于将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0010] 根据本发明另一方面,提供一种驱动硬件配置信息加载系统,包括:判断装置,用于判断设备资源是否大于等于阈值,且在判断出设备资源小于阈值的情况下,触发静态设备树文件加装置,在判断出设备资源大于等于阈值的情况下,触发动态驱动硬件配置信息加载装置;静态驱动硬件配置信息加载装置,用于将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、驱动的系统代码一起编译、下载到设备中;动态驱动硬件配置信息加载装置,包括:读取模块,用于从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;解析模块,用于解析所述设备树文件中的硬件配置信息;传送模块,用于将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0011] 根据本发明另一方面,提供一种计算设备,包括:至少一个处理器和存储有程序指令的存储器;当所述程序指令被所述处理器读取并执行时,使得所述计算设备执行上述驱动硬件配置信息加载方法。

[0012] 根据本发明另一方面,提供一种存储有程序指令的可读存储介质,当所述程序指令被计算设备读取并执行时,使得所述计算设备执行上述驱动硬件配置信息加载方法。

[0013] 在传统技术中,如果需要加载驱动,需要在驱动源代码中写入相关配置信息,比如用哪些管脚、哪些寄存器,然后将该驱动源代码与内核系统代码一起编译。根据本发明实施方式,只需要制备设备树文件,并在加载驱动时读取设备树文件,就可以获知驱动的相关硬件配置信息。不需要将设备树文件编译成特定的文件,系统就可以直接识别。

## 附图说明

[0014] 为了实现上述以及相关目的,本文结合下面的描述和附图来描述某些说明性方面,这些方面指示了可以实践本文所公开的原理的各种方式,并且所有方面及其等效方面旨在落入所要求保护的主题的范围内。通过结合附图阅读下面的详细描述,本公开的上述以及其它目的、特征和优势将变得更加明显。遍及本公开,相同的附图标记通常指代相同的部件或元素。

[0015] 图1示出了根据本发明实施方式的计算设备100的示意图;

[0016] 图2示出本发明实施方式的驱动硬件配置信息加载方法的整体架构图。

[0017] 图3示出根据本发明实施方式的驱动硬件配置信息加载方法的流程图。

[0018] 图4示出传统驱动硬件配置信息加载模式的示意性流程图。

[0019] 图5示出了包括静态驱动硬件配置信息加载方式、动态驱动硬件配置信息加载方式这两种加载方式的整体流程图。

[0020] 图6示出根据本发明实施方式的另一驱动硬件配置信息加载方法的流程图。

[0021] 图7示出根据本发明实施方式的驱动硬件配置信息加载装置的示意性框图。

[0022] 图8示出根据本发明实施方式的驱动硬件配置信息加载系统的示意性框图。

### 具体实施方式

[0023] 下面将参照附图更详细地描述本公开的示例性实施方式。虽然附图中显示了本公开的示例性实施方式,然而应当理解,本公开可以以各种形式实现而不应被这里阐述的实施方式所限制。相反,提供这些实施方式是为了能够更透彻地理解本公开,并且能够将本公开的范围完整地传达给本领域的技术人员。

[0024] 针对现有外设硬件信息改变时设备资源信息必须参与编译这一技术问题,本发明提供了一种驱动硬件配置信息加载方法和装置、计算设备和可读存储介质,其能够直接将硬件资源信息解析成设备节点信息,然后传入驱动硬件初始化模块,完成驱动的加载,且能够实时修改设备树文件中的硬件配置信息而不用重新编译系统,实现了灵活配置外设驱动。

[0025] 本发明的驱动硬件配置信息加载方法在计算设备中执行。计算设备可以是任意具有存储和计算能力的设备,其例如可以实现为服务器、工作站等,也可以实现为桌面计算机、笔记本计算机等个人配置的计算机,或者实现为手机、平板电脑、智能可穿戴设备、物联网设备等终端设备,但不限于此。

[0026] 图1示出了根据本发明一个实施方式的计算设备100的示意图。需要说明的是,图1所示的计算设备100仅为一个示例,在实践中,用于实施本发明的驱动硬件配置信息加载方法的计算设备可以是任意型号的设备,其硬件配置情况可以与图1所示的计算设备100相同,也可以与图1所示的计算设备100不同。实践中用于实施本发明的驱动硬件配置信息加载方法的计算设备可以对图1所示的计算设备100的硬件组件进行增加或删减,本发明对计算设备的具体硬件配置情况不做限制。

[0027] 如图1所示,在基本的配置102中,计算设备100典型地包括系统存储器106和一个或者多个处理器104。存储器总线108可以用于在处理器104和系统存储器106之间的通信。

[0028] 取决于期望的配置,处理器104可以是任何类型的处理,包括但不限于:微处理器( $\mu$ P)、微控制器( $\mu$ C)、数字信息处理器(DSP)或者它们的任何组合。处理器104可以包括诸如一级高速缓存110和二级高速缓存112之类的一个或者多个级别的高速缓存、处理器核心114和寄存器116。示例的处理器核心114可以包括运算逻辑单元(ALU)、浮点数单元(FPU)、数字信号处理核心(DSP核心)或者它们的任何组合。示例的存储器控制器118可以与处理器104一起使用,或者在一些实现中,存储器控制器118可以是处理器104的一个内部部分。

[0029] 取决于期望的配置,系统存储器106可以是任意类型的存储器,包括但不限于:易失性存储器(诸如RAM)、非易失性存储器(诸如ROM、闪存等)或者它们的任何组合。计算设备中的物理内存通常指的是易失性存储器RAM,磁盘中的数据需要加载至物理内存中才能够被处理器104读取。系统存储器106可以包括操作系统120、一个或者多个应用122以及程序数据124。在一些实施方式中,应用122可以布置为在操作系统上由一个或多个处理器104利用程序数据124执行指令。操作系统120例如可以是Linux、Windows等,其包括用于处理基本系统服务以及执行依赖于硬件的任务的程序指令。应用122包括用于实现各种用户期望的功能的程序指令,应用122例如可以是浏览器、即时通讯软件、软件开发工具(例如集成开发环境IDE、编译器等)等,但不限于此。当应用122被安装到计算设备100中时,可以向操作系

统120添加驱动模块。

[0030] 在计算设备100启动运行时,处理器104会从存储器106中读取操作系统120的程序指令并执行。应用122运行在操作系统120之上,利用操作系统120以及底层硬件提供的接口来实现各种用户期望的功能。当用户启动应用122时,应用122会加载至存储器106中,处理器104从存储器106中读取并执行应用122的程序指令。

[0031] 计算设备100还包括储存设备132,储存设备132包括可移除储存器136和不可移除储存器138,可移除储存器136和不可移除储存器138均与储存接口总线134连接。

[0032] 计算设备100还可以包括有助于从各种接口设备(例如,输出设备142、外设接口144和通信设备146)到基本配置102经由总线/接口控制器130的通信的接口总线140。示例的输出设备142包括图形处理单元148和音频处理单元150。它们可以被配置为有助于经由一个或者多个A/V端口152与诸如显示器153或者扬声器之类的各种外部设备进行通信。示例外设接口144可以包括串行接口控制器154和并行接口控制器156,它们可以被配置为有助于经由一个或者多个I/O端口158和诸如输入设备(例如,键盘、鼠标、笔、语音输入设备、触摸输入设备)或者其他外设(例如打印机、扫描仪等)之类的外部设备进行通信。示例的通信设备146可以包括网络控制器160,其可以被布置为便于经由一个或者多个通信端口164与一个或者多个其他计算设备162通过网络通信链路的通信。

[0033] 网络通信链路可以是通信介质的一个示例。通信介质通常可以体现为在诸如载波或者其他传输机制之类的调制数据信号中的计算机可读指令、数据结构、程序模块,并且可以包括任何信息递送介质。“调制数据信号”可以这样的信号,它的数据集中的一个或者多个或者它的改变可以在信号中编码信息的方式进行。作为非限制性的示例,通信介质可以包括诸如有线网络或者专线网络之类的有线介质,以及诸如声音、射频(RF)、微波、红外(IR)或者其他无线介质在内的各种无线介质。这里使用的术语计算机可读介质可以包括存储介质和通信介质二者。

[0034] 在根据本发明的计算设备100中,应用122包括用于执行本发明的驱动硬件配置信息加载方法300的指令和其他应用126,驱动硬件配置信息加载方法300的指令可以指示处理器104执行本发明的驱动硬件配置信息加载方法300,以直接将硬件资源信息解析成设备节点信息,然后传入驱动硬件初始化模块,完成驱动的加载。

[0035] 图2示出本发明实施方式方法的整体架构图。首先由开发人员根据设备开发板的具体设计情况准备设备树文件。设备树文件也称为DTS文件(Device Tree Source),是用于描述开发板上驱动的硬件配置信息的文件。开发板上所有的驱动比如SPI驱动、flash驱动、SD卡驱动的硬件配置信息全部写入该文件中。然后,将设备树文件存放到设备内或外的存储单元中。下面给出了一部分设备树文件的示例:

```

Pinctrl: pin-controller@40020000{
    Compatible="st, stm32-pinctrl1";
    #address-cells=<1>;
    #size-cells=<1>;
    Reg=<0X40020000 0X2000>;

```

```

[0036] gpioa:gpio@40020000{
    Compatible="st, stm32-gpio";
    Gpio-controller;
    @gpio-cells=<2>;
    Reg=<0x40020000 0x400>;
    Clocks=<&rcc 1 0x00000001>;
    Label="GPIOA";
};

```

```

[0037] gpiob:gpio@40020400{
    Compatible="st, stm32-gpio";
    Gpio-controller;
    @gpio-cells=<2>;
    Reg=<0x40020400 0x400>;
    Clocks=<&rcc 1 0x00000002>;
    Label="GPIOB";

```

```

}

```

[0038] 根据本发明实施方式的设计,内核添加了驱动硬件配置信息加载装置,如图2中的粗线框所示。驱动硬件配置信息加载方法300由该驱动硬件配置信息加载装置来执行。内核、驱动的系统代码编译完成后被下载、烧写到设备中。在设备上电、系统启动后,内核要加载驱动时需要读取驱动相关的寄存器、端口等,比如对于GPIO驱动,需要知道驱动的是GPIO



的哪个端口(1-7中的哪一个)。此时驱动硬件配置信息加载装置就会读取存储单元中的设备树文件,解析设备树文件中的硬件资源信息,从设备树文件中解析出驱动能识别的硬件信息,即上述例子中的具体端口号,并传送给驱动以完成硬件初始化。

[0039] 下面参考图3描述上述驱动硬件配置信息加载装置执行的驱动硬件配置信息加载方法的具体过程。图3示出了根据本发明一个实施方式的驱动硬件配置信息加载方法300的流程图。如图3所示,方法300包括步骤:

[0040] S310,从存储单元中读取设备树文件。

[0041] S320,解析所述设备树文件中的硬件配置信息。

[0042] S330,将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0043] 以以上所示的设备树文件举例说明。比如解析设备树文件中的硬件配置信息gpioa,解析出的信息存放在gpio的描述结构体中,该结构体包含信息如下:

[0044] 此外设寄存器起始地址:0x40020000

[0045] 此外设寄存器大小:0x400

[0046] 此外设的时钟总线名称:RCC

[0047] 此外设的时钟总线号:1

[0048] 此外设的时钟总线比特位:0x00000001

[0049] 此外设的驱动名称:GPIOA

[0050] 将该信息发送给驱动后,驱动就能根据硬件信息的描述完成初始化。

[0051] 在传统技术中,如果需要加载驱动,需要在驱动源代码中写入相关配置信息,比如用哪些管脚、哪些寄存器,然后将该驱动源代码与内核系统代码一起编译。然而,根据本发明实施方式,只需要制备设备树文件,并在加载驱动时读取设备树文件,就可以获知驱动的相关硬件配置信息。不需要将设备树文件编译成特定的文件,系统就可以直接识别。

[0052] 另外,一旦外设调整,可以直接修改设备树文件(设备树文件是ASCII码,打开可以正常查看和修改),以匹配调整后的外设。如果同一驱动的所采用管脚变化,只需要在设备树文件中对应地修改,再把修改后的设备树文件置入存储单元中即可。然而在传统方法中,由于传统的设备外设硬件资源信息要么直接写入驱动配置文件中,要么使用设备树生成头文件跟系统代码一起编译,一旦编译后驱动的硬件资源信息就无法再进行更改。当设备需要启用跟编译驱动代码设置的外设模块不同的模块时,或者设备需要修改为采用其他外设时,就必须从驱动源码中修改或者重新用设备树生成头文件,然后再次编译、下载。当设备繁多且外设配置不尽相同时,就必须对每个设备进行定制化修改和编译。或者在设备部署地,需要临时修改设备的外设模块时,在没有编译环境和烧写工具的情况下,无法完成此操作。

[0053] 比如,拿以上设备树文件的示例来举例说明。在上述设备树文件中,gpio是个管脚的驱动,其中gpioa、gpiob以及未示出的其他gpio管脚的所有配置几乎相同,除了某个寄存器地址不同。比如有的设备需要gpioa,有的设备需要gpiob。根据本发明实施方式,如果只需要gpioa,就在设备树文件中只写gpioa的配置,如果只需要gpiob,就在设备树文件中只写gpiob的配置。如果设备不再需要gpiob功能模块,可以直接在设备树文件中将gpiob的内容删除。然而传统静态加载方式需要从系统代码中删除,传统动态加载方法需要将设备树中gpiob部分删除后,重新编译设备树文件,即每次修改后都需要重新编译。然而把所有代

码都按照传统技术中教导的那样重新编译一遍,会浪费计算资源。根据本发明实施方式,省去了重新编译的步骤环节。

[0054] 因此,方法300中的步骤“从存储单元中读取设备树文件”可以包括:从存储单元中读取修改后的设备树文件,所述修改后的设备树文件是根据驱动硬件配置信息而修改的。

[0055] 该步骤是通过图2所示的增设的驱动硬件配置信息加载装置实现的。当设备的外设硬件配置信息发生变化时,只需要修改设备树文件即可。在设备启动后,会通过驱动硬件配置信息加载装置解析到新配置的硬件信息,然后传送给驱动模块完成驱动的加载。这样就避免了硬件信息改变时,必须修改代码和重新编译的过程。

[0056] 当前实时操作系统一般加载的image镜像中已经包含设备外设驱动功能模块,而本发明实施方式的方法则是将设备外设驱动功能模块从image镜像中独立出来,直接加载未编译的设备树文件,从而使得能够直接修改设备树文件以灵活定制外设功能,而不用重新编译整个系统。

[0057] 方法300还可以包括:识别驱动的硬件配置信息,且在识别出驱动的硬件配置信息发生变化的情况下,通知相关人员修改设备树文件。

[0058] 驱动硬件配置信息加载装置可以定时获取设备树文件,或者响应于工作人员的触发来获取设备树文件,或者可以设置为在设备重启后,驱动硬件配置信息加载装置获取到更新的设备树文件,从而可以读取到修改后的设备树文件,且根据修改后文件中的硬件配置信息来加载驱动。

[0059] 另外,由于设备本身内存较小或者由于其他程序的运行导致设备内存不足时,如果希望以更节省空间的方式执行驱动的加载,可以选择采用传统驱动硬件配置信息加载方式。其中,内存可以是设备内部的内存,也可以是内部内存和外挂RAM空间之和。

[0060] 即首先判断设备资源的大小是否满足一定条件,在设备资源小于一定阈值(阈值可以根据动态加载所需的空间决定)时,即设备性能匮乏、资源有限,可以配置使用传统驱动硬件配置信息加载模式,即将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、驱动的系统代码一起编译、下载。该传统驱动硬件配置信息加载模式如图4所示。

[0061] 在大于等于该阈值时,判断采用以上参考图3所述的动态加载步骤。

[0062] 图5示出了包括静态驱动硬件配置信息加载、动态驱动硬件配置信息加载方式这两种加载方式的整体架构图。可以通过两种方式配置硬件资源信息:针对存储资源较为丰富、性能较快的硬件平台,可以直接从文件系统加载设备树文件,解析其中的硬件配置信息,并完成硬件驱动的初始化和设置工作;针对资源较为匮乏、性能较弱的硬件平台,可以使用外部工具链解析设备树文件,并自动生成对应的C语言头文件,与内核一起编译后下载固件到硬件,然后完成硬件驱动的初始化和设置工作。

[0063] 因此,根据本发明实施方式,提供一种驱动硬件配置信息加载方法。如图6所示,该方法包括以下步骤:

[0064] S610,判断设备资源是否大于等于阈值;

[0065] S620,在设备资源小于阈值的情况下,判断采用静态驱动硬件配置信息加载方式,其中,将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、

驱动的系统代码一起编译、下载到设备中；

[0066] S630,在设备资源大于等于阈值的情况下,执行以下步骤:

[0067] S6310,从存储单元中读取设备树文件;

[0068] S6320,解析所述设备树文件中的硬件配置信息;

[0069] S6330,将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0070] 步骤S6310~S6330的执行过程与以上图3的步骤S310~S330相同,在此不再予以赘述。

[0071] 根据本发明实施方式,可以在实时操作系统的微控制器平台上同时提供静态加载和动态加载硬件配置信息的方法,在实践中动态加载和静态加载使用同一套API接口。根据设备资源是否丰富选择对应的动态或者静态加载硬件信息的方式。在设备资源匮乏时,选择静态加载方式可以很好地满足设备的功能需求,做到小而精。在设备资源丰富时,则选择动态加载的方式。如图2所示,增设完整自研的驱动硬件配置信息加载装置,该装置实现将硬件信息配置功能推迟到编译之后,动态将信息传递给驱动加载模块。在内核启动后,读取设备树文件,直接将硬件资源信息解析成设备节点信息,然后传入驱动硬件初始化模块,完成驱动的加载。且记载硬件信息的设备树文件可以实时修改而不用重新编译系统,使得能够灵活配置外设驱动且不需要重新编译系统。

[0072] 根据本发明实施方式,还提供一种驱动硬件配置信息加载装置。图7示出该驱动硬件配置信息加载装置的示意性框图。如图7所示,该装置包括:

[0073] 读取模块710,用于从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;

[0074] 解析模块720,用于解析所述设备树文件中的硬件配置信息;

[0075] 传送模块730,用于将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0076] 其中,读取模块710可以从存储单元中读取修改后的设备树文件,所述修改后的设备树文件是根据驱动的硬件配置信息而修改的。

[0077] 装置还可以包括识别模块,用于识别驱动的硬件配置信息,且在识别出驱动的硬件配置信息发生变化的情况下,通知相关人员修改设备树文件。

[0078] 根据本发明实施方式,还提供一种驱动硬件配置信息加载系统。如图8所示,该系统包括:

[0079] 判断装置810,用于判断设备资源是否大于等于阈值,且在判断出设备资源小于阈值的情况下,触发静态设备树文件加装置,在判断出设备资源大于等于阈值的情况下,触发动态驱动硬件配置信息加载装置;

[0080] 静态驱动硬件配置信息加载装置820,用于将设备树文件中的硬件配置信息通过工具链编译成设备头文件,然后再将其与内核、驱动的系统代码一起编译、下载到设备中;

[0081] 动态驱动硬件配置信息加载装置830,包括:

[0082] 读取模块8310,用于从存储单元中读取设备树文件,所述设备树文件由开发人员制备,且用于描述开发板上所有驱动的硬件配置信息;

[0083] 解析模块8320,用于解析所述设备树文件中的硬件配置信息;

[0084] 传送模块8330,用于将解析结果传送给相应的驱动以完成驱动的硬件初始化。

[0085] 读取模块8310、解析模块8320、传送模块8330与参考图7描述的读取模块710、解析

模块720、传送模块730功能相同。

[0086] 这里描述的各种技术可结合硬件或软件,或者它们的组合一起实现。从而,本发明的方法和设备,或者本发明的方法和设备的某些方面或部分可采取嵌入有形媒介,例如可移动硬盘、U盘、软盘、CD-ROM或者其它任意机器可读的存储介质中的程序代码(即指令)的形式,其中当程序被载入诸如计算机之类的机器,并被所述机器执行时,所述机器变成实践本发明的设备。

[0087] 在程序代码在可编程计算机上执行的情况下,计算设备一般包括处理器、处理器可读的存储介质(包括易失性和非易失性存储器和/或存储元件),至少一个输入装置,和至少一个输出装置。其中,存储器被配置用于存储程序代码;处理器被配置用于根据该存储器中存储的所述程序代码中的指令,执行本发明的驱动硬件配置信息加载方法。

[0088] 以示例而非限制的方式,可读介质包括可读存储介质和通信介质。可读存储介质存储诸如计算机可读指令、数据结构、程序模块或其它数据等信息。通信介质一般以诸如载波或其它传输机制等已调制数据信号来体现计算机可读指令、数据结构、程序模块或其它数据,并且包括任何信息传递介质。以上的任一种的组合也包括在可读介质的范围之内。

[0089] 在此处所提供的说明书中,算法和显示不与任何特定计算机、虚拟系统或者其它设备固有相关。各种通用系统也可以与本发明的示例一起使用。根据上面的描述,构造这类系统所要求的结构是显而易见的。此外,本发明也不针对任何特定编程语言。应当明白,可以利用各种编程语言实现在此描述的本发明的内容,并且上面对特定语言所做的描述是为了披露本发明的优选实施方式。

[0090] 在此处所提供的说明书中,说明了大量具体细节。然而,能够理解,本发明的实施方式可以在没有这些具体细节的情况下被实践。在一些实例中,并未详细示出公知的方法、结构和技术,以便不模糊对本说明书的理解。

[0091] 类似地,应当理解,为了精简本公开并帮助理解各个发明方面中的一个或多个,在上面对本发明的示例性实施方式的描述中,本发明的各个特征有时被一起分组到单个实施方式、图、或者对其的描述中。然而,并不应将该公开的方法解释成反映如下意图:即所要求保护的本发明要求比在每个权利要求中所明确记载的特征更多特征。本领域那些技术人员应当理解在本文所公开的示例中的设备的模块或单元或组件可以布置在如该实施方式中所描述的设备中,或者可替换地可以定位在与该示例中的设备不同的一个或多个设备中。前述示例中的模块可以组合为一个模块或者此外可以分成多个子模块。

[0092] 本领域那些技术人员可以理解,可以对实施方式中的设备中的模块进行自适应性地改变并且把它们设置在与该实施方式不同的一个或多个设备中。可以把实施方式中的模块或单元或组件组合成一个模块或单元或组件,以及此外可以把它分成多个子模块或子单元或子组件。除了这样的特征和/或过程或者单元中的至少一些是相互排斥之外,可以采用任何组合对本说明书(包括伴随的权利要求、摘要和附图)中公开的所有特征以及如此公开的任何方法或者设备的所有过程或单元进行组合。除非另外明确陈述,本说明书(包括伴随的权利要求、摘要和附图)中公开的每个特征可以由提供相同、等同或相似目的的替代特征来代替。

[0093] 此外,本领域的技术人员能够理解,尽管在此所述的一些实施方式包括其它实施方式中所包括的某些特征而不是其它特征,但是不同实施方式的特征的组合意味着处于本

发明的范围之内并且形成不同的实施方式。此外,所述实施方式中的一些在此被描述成可以由计算机系统的处理器或者由执行所述功能的其它装置实施的方法或方法元素的组合。因此,具有用于实施所述方法或方法元素的必要指令的处理器形成用于实施该方法或方法元素的装置。此外,装置实施方式的在此所述的元素是如下装置的例子:该装置用于实施由为了实施该发明的目的的元素所执行的功能。

[0094] 如在此所使用的那样,除非另行规定,使用序数词“第一”、“第二”、“第三”等等来描述普通对象仅仅表示涉及类似对象的不同实例,并且并不意图暗示这样被描述的对象必须具有时间上、空间上、排序方面或者以任意其它方式的给定顺序。

[0095] 尽管根据有限数量的实施方式描述了本发明,但是受益于上面的描述,本技术领域的技术人员明白,在由此描述的本发明的范围内,可以设想其它实施方式。此外,应当注意,本说明书中使用的语言主要是为了可读性和教导的目的而选择的,而不是为了解释或者限定本发明的主题而选择的。

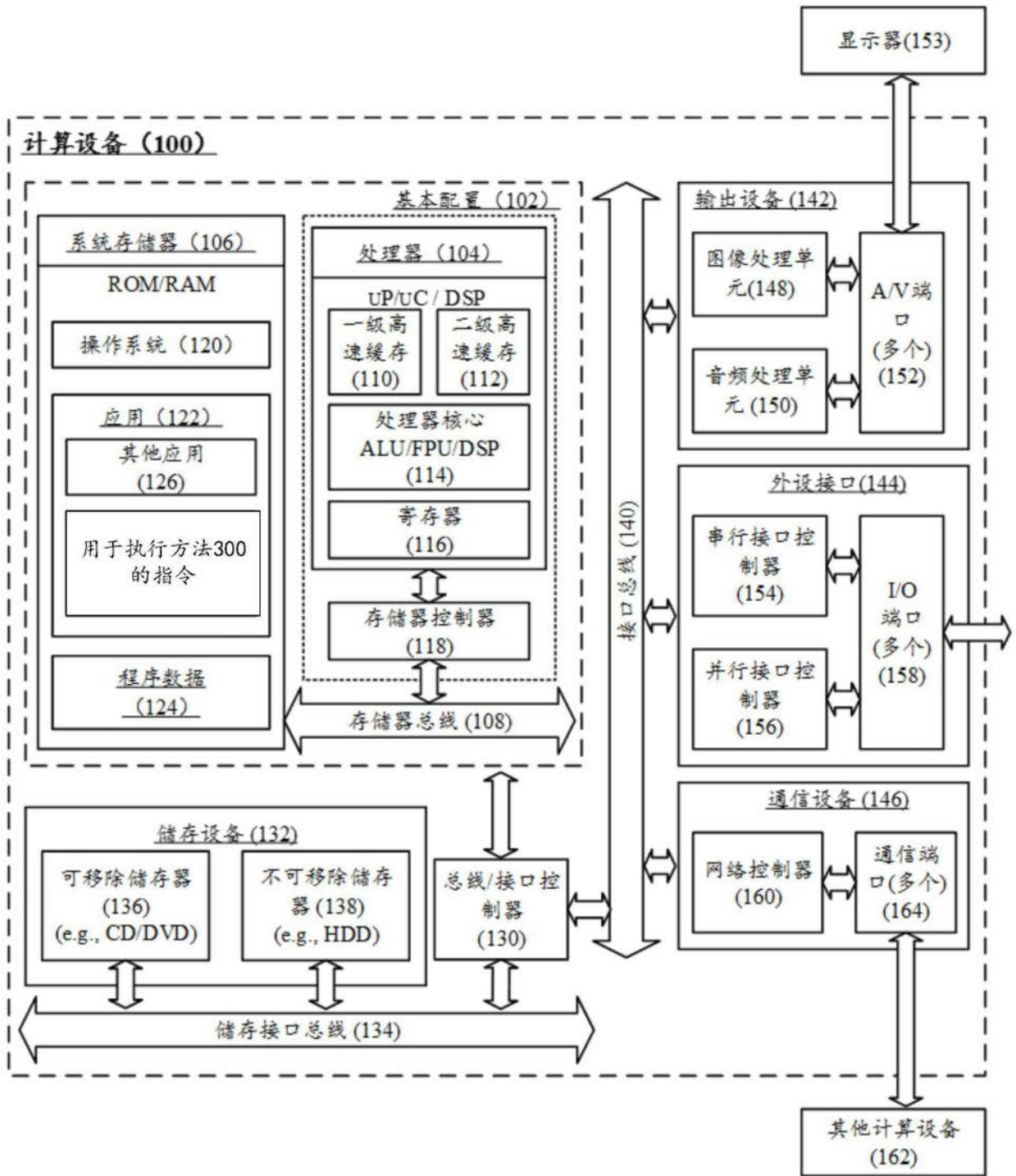


图1

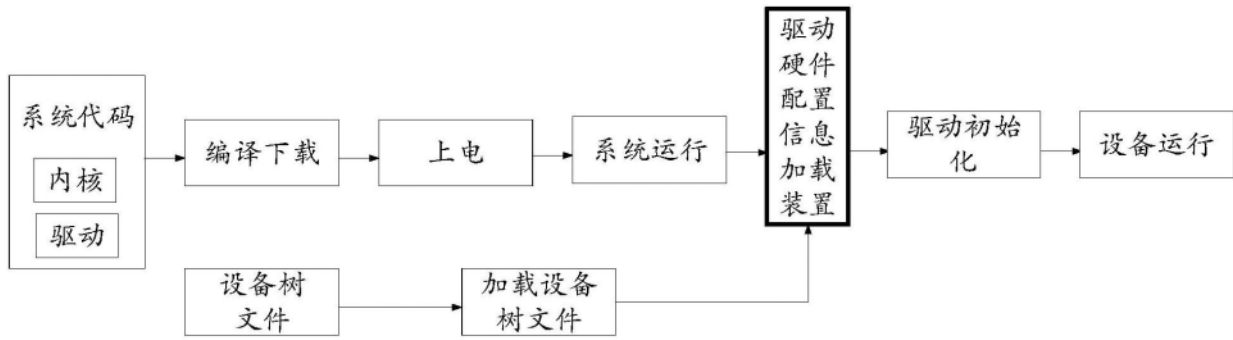


图2

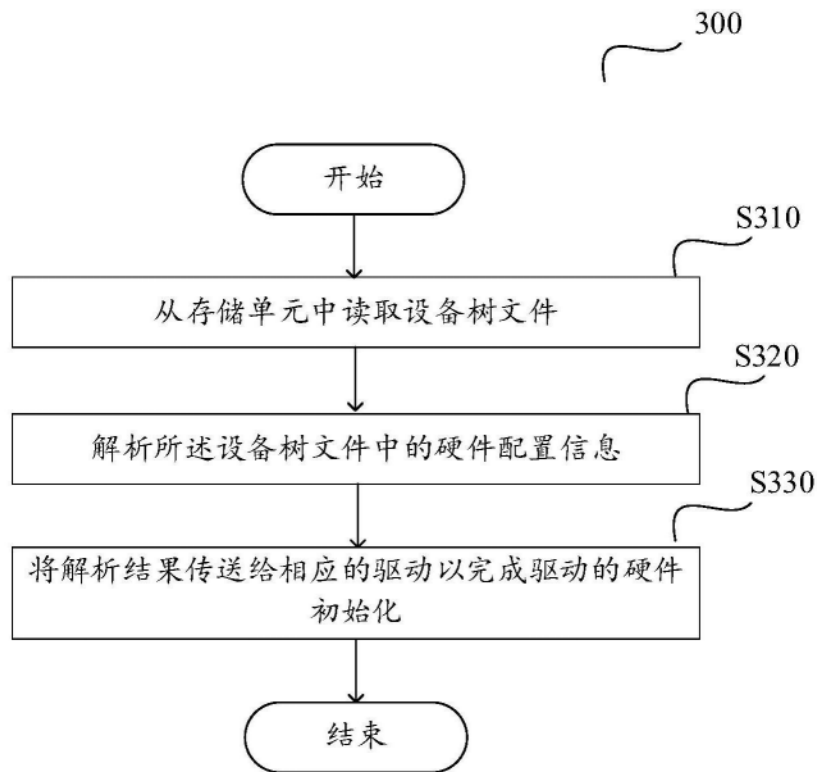


图3

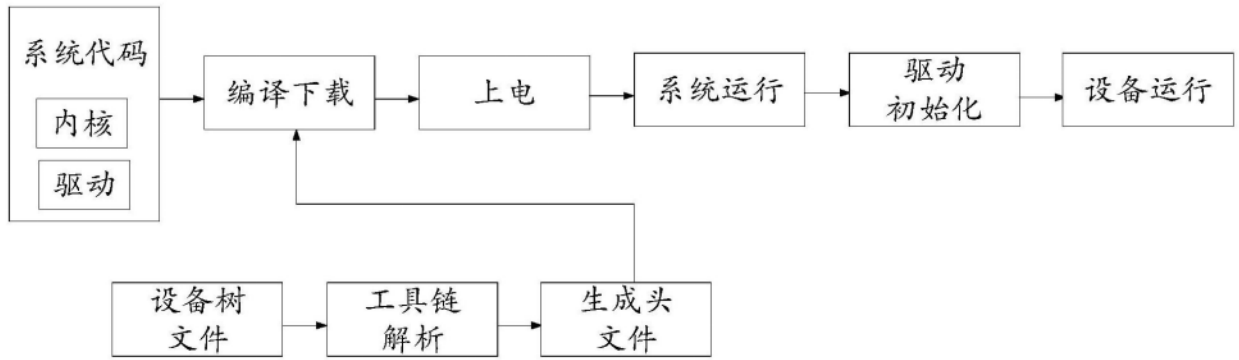


图4

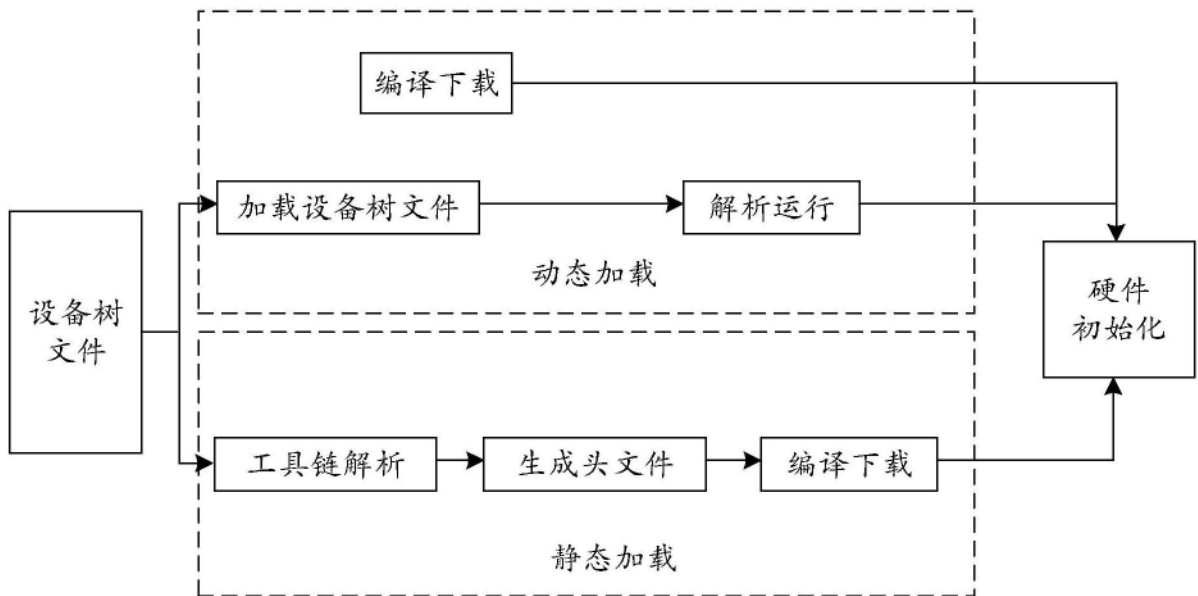


图5



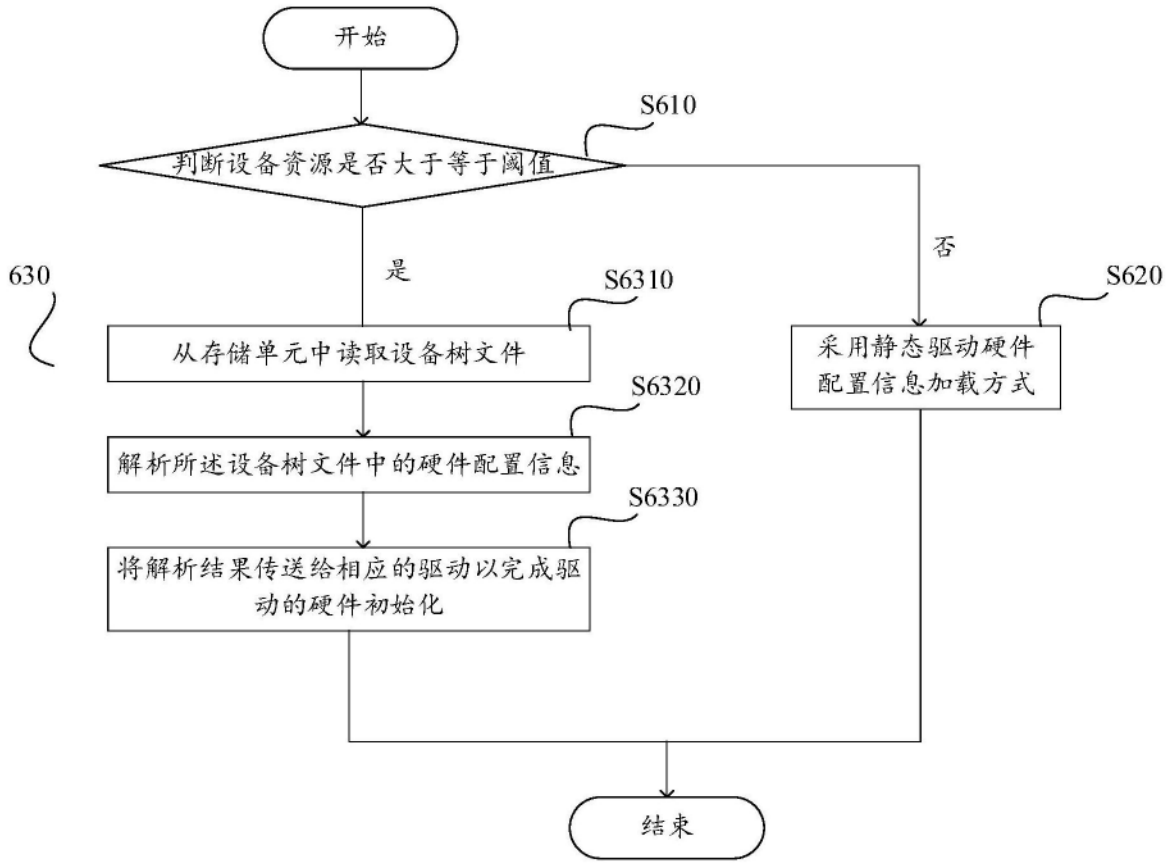


图6

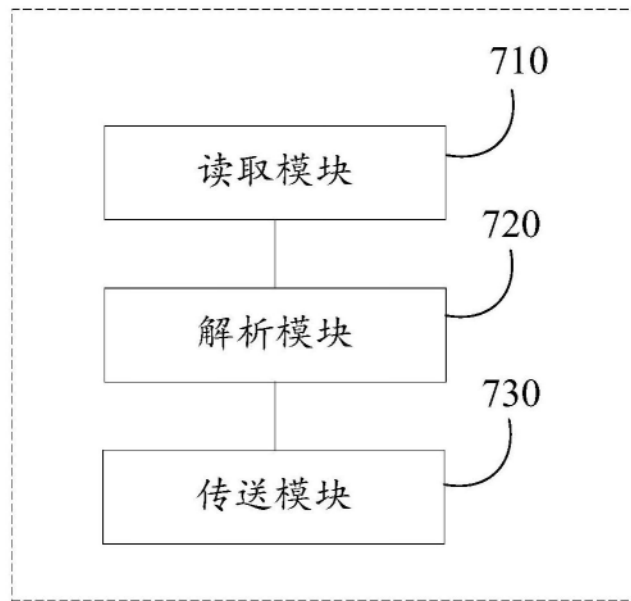


图7

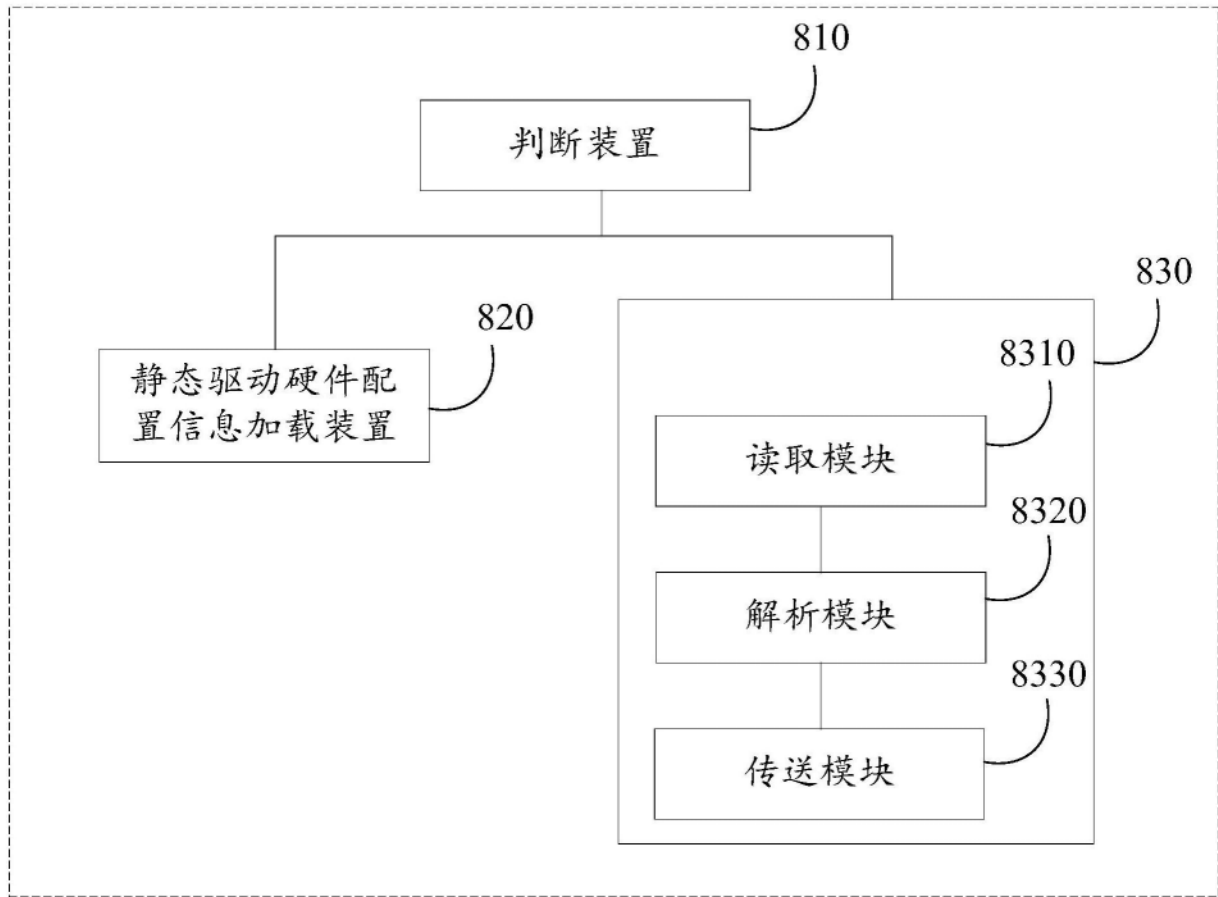


图8