



(12) 发明专利

(10) 授权公告号 CN 101403982 B

(45) 授权公告日 2011. 07. 20

(21) 申请号 200810225648. 3

US 20030149716 A1, 2003. 08. 07, 说明书第

(22) 申请日 2008. 11. 03

2 页第 26 段, 第 3 页第 28 段.

CN 101231600 A, 2008. 07. 30, 权利要求 1.

(73) 专利权人 华为技术有限公司

地址 518129 广东省深圳市龙岗区坂田华为
总部办公楼

审查员 鲁洁

(72) 发明人 侯忠明

(74) 专利代理机构 北京三高永信知识产权代理
有限责任公司 11138

代理人 何文彬

(51) Int. Cl.

G06F 9/50 (2006. 01)

(56) 对比文件

CN 1834950 A, 2006. 09. 20, 全文.

CN 1119749 C, 2003. 08. 27, 全文.

CN 1906587 A, 2007. 01. 31, 全文.

CN 101166099 A, 2008. 04. 23, 权利要求

1-14, 说明书第 6 - 20 页, 附图 3 - 12.

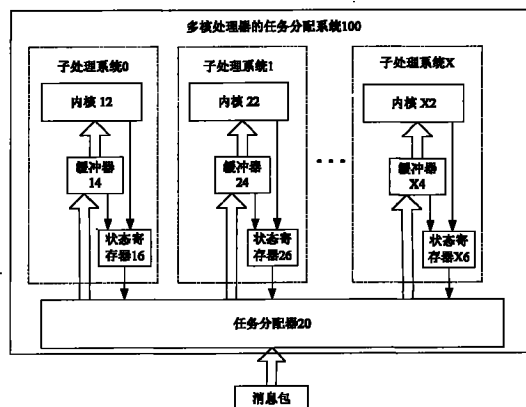
权利要求书 2 页 说明书 8 页 附图 4 页

(54) 发明名称

一种多核处理器的任务分配方法和系统

(57) 摘要

本发明公开了一种多核处理器的任务分配方法、系统及设备,属于通信技术领域。所述方法包括:任务分配器根据状态寄存器的状态判断处理器内核是否空闲;任务分配器通过传输通道向空闲的处理器内核传输消息包。所述系统包括:多个子处理系统,所述多个子处理系统分别包括任务分配器、状态寄存器及处理器内核。本发明提高了多核处理器处理任务的效率,使多核处理器在不降低对突发数据的处理能力及系统性能的前提下,实现低功耗。



1. 一种多核处理器的任务分配方法,其特征在于,所述方法包括:

任务分配器根据状态寄存器的状态判断处理器内核是否空闲,其中,空闲的处理器内核的状态包括等待状态、休眠状态或关闭状态;

当所述空闲的处理器内核的状态为等待状态时,所述任务分配器通过旁路向所述空闲的处理器内核传输消息包;

当所述空闲的处理器内核的状态为休眠或关闭状态时,所述任务分配器通过缓冲器向所述空闲的处理器内核传输所述消息包。

2. 根据权利要求1所述的多核处理器的任务分配方法,其特征在于,所述任务分配器根据状态寄存器的状态判断处理器内核是否空闲,包括:

当所述多核处理器中包括多个多线程处理器内核时,所述多个多线程处理器内核相应的状态寄存器根据所述多线程处理器内核空闲线程的数目向所述任务分配器发送一状态信息;

所述任务分配器根据所述多个多线程处理器内核所相应的状态寄存器返回的状态信息,判断所述多个多线程处理器内核的空闲程度。

3. 根据权利要求1所述的多核处理器的任务分配方法,其特征在于,所述方法还包括:

所述缓冲器将接收的所述任务分配器发送的消息包缓存;

当所述处理器内核为休眠或关闭状态时,唤醒所述休眠或关闭状态的处理器内核;

当所述休眠或关闭状态的处理器内核被唤醒后,所述缓冲器将缓存的消息包发送给被所述唤醒后的处理器内核。

4. 一种多核处理器的任务分配方法,用于多核处理器中,所述多核处理器包括多个子处理系统,其特征在于,所述方法包括:

通过状态寄存器读取所述多个子处理系统的状态信息,所述状态信息显示所述多个子处理系统的当前状态,所述当前状态包括等待、休眠或关闭;

向处于休眠或关闭状态的子处理系统的缓冲器发送消息包;唤醒所述处于休眠或关闭状态的子处理系统;所述缓冲器将所述消息包发送给所述子处理系统的处理器内核;

通过旁路向处于等待状态的子处理系统的处理器内核发送消息包。

5. 一种多核处理器的任务分配系统,其特征在于,所述系统,包括:任务分配器和子处理系统,所述子处理系统包括状态寄存器、处理器内核、路径控制器及传输通道,所述传输通道包括旁路和缓冲器,其中,

所述状态寄存器,用于向所述任务分配器发送所述子处理系统的状态信息,所述状态信息包括所述子处理系统的处理器内核是否处于空闲状态,所述空闲状态包括等待、休眠或关闭;

所述任务分配器,用于向所述子处理系统发送消息包;

所述路径控制器,用于接收所述消息包;当所述处理器内核处于等待状态时,选择所述旁路作为传输通道,并将所述消息包发送给所述旁路;当所述处理器内核处于休眠或关闭状态时,选择所述缓冲器作为传输通道,并将所述消息包发送给所述缓冲器;

所述传输通道,用于将所述路径控制器发送的所述消息包传送给所述处理器内核;

其中,当所述处理器内核处于休眠或关闭状态时,所述路径控制器将所述消息包发送给所述缓冲器缓存,并唤醒所述处理器内核,当所述处理器内核被唤醒后,所述缓冲器将所

述消息包发送给所述处理器内核。

一种多核处理器的任务分配方法和系统

技术领域

[0001] 本发明涉及通信技术领域,特别涉及一种多核处理器的任务分配方法和系统。

背景技术

[0002] 在高性能处理器设计中,靠单纯提高主频实现处理器性能提升已经遇到瓶颈,目前主流的趋势是使用处理器多内核设计,例如 PC(Personal Computer,个人计算机)通用处理器的双核以及网络处理器的几十个处理器内核的设计。在现实中,处理器的任务在时间域上往往是不均衡的,因此对处理能力的需求也有所不同。例如,网络繁忙时,可能需要网络处理器的全部处理能力,而网络较空闲时,可能只需要等效一两个处理器内核的处理能力就足够了,这种处理能力上的裕量,为降低处理器功耗提供了可能的空间。关于低功耗设计(LPD, Low Power Design),目前在器件级上已有一些比较成熟的方法,比如停止闲置器件的时钟信号或者切断该部分的电压。

[0003] 现有技术中,当处理器空闲时,保存现场,然后整个处理器进入休眠状态。当有任务要处理时,恢复现场,唤醒处理器进行处理。

[0004] 在实现本发明的过程中,发明人发现:现有技术中由于整个处理器进入休眠,此时无法处理任何实时数据。而当有任务需要处理时只能先唤醒处理器,然后才发送任务信息,这样处理任务的效率就受到了影响。

发明内容

[0005] 为了提高多核处理器处理任务的效率,使多核处理器在不降低对突发数据的处理能力及系统性能的前提下,实现低功耗,本发明实施例提供了一种多核处理器的任务分配方法、系统及设备,所述技术方案如下:

[0006] 一方面,提供了一种多核处理器的任务分配方法,所述方法包括:

[0007] 任务分配器根据状态寄存器的状态判断处理器内核是否空闲,其中,空闲的处理器内核的状态包括等待状态、休眠状态或关闭状态;

[0008] 当所述空闲的处理器内核的状态为等待状态时,所述任务分配器通过旁路向所述空闲的处理器内核传输消息包;

[0009] 当所述空闲的处理器内核的状态为休眠或关闭状态时,所述任务分配器通过缓冲器向所述空闲的处理器内核传输所述消息包。

[0010] 另一方面,还提供了一种多核处理器的任务分配方法,用于多核处理器中,所述多核处理器包括多个子处理系统,所述方法包括:

[0011] 通过状态寄存器读取所述多个子处理系统的状态信息,所述状态信息显示所述多个子处理系统的当前状态,所述当前状态包括等待、休眠或关闭;

[0012] 向处于休眠或关闭状态的子处理系统的缓冲器发送消息包;唤醒所述处于休眠或关闭状态的子处理系统;所述缓冲器将所述消息包发送给所述子处理系统的处理器内核;

[0013] 通过旁路向处于等待状态的子处理系统的处理器内核发送消息包。

[0014] 再一方面,还提供了一种多核处理器的任务分配系统,所述系统,包括:任务分配器和子处理系统,所述子处理系统包括状态寄存器、处理器内核、路径控制器及传输通道,所述传输通道包括旁路和缓冲器,其中,

[0015] 所述状态寄存器,用于向所述任务分配器发送所述子处理系统的状态信息,所述状态信息包括所述子处理系统的处理器内核是否处于空闲状态,所述空闲状态包括等待、休眠或关闭;

[0016] 所述任务分配器,用于向所述子处理系统发送消息包;

[0017] 所述路径控制器,用于接收所述消息包;当所述处理器内核处于等待状态时,选择所述旁路作为传输通道,并将所述消息包发送给所述旁路;当所述处理器内核处于休眠或关闭状态时,选择所述缓冲器作为传输通道,并将所述消息包发送给所述缓冲器;

[0018] 所述传输通道,用于将所述路径控制器发送的所述消息包传送给所述处理器内核;

[0019] 其中,当所述处理器内核处于休眠或关闭状态时,所述路径控制器将所述消息包发送给所述缓冲器缓存,并唤醒所述处理器内核,当所述处理器内核被唤醒后,所述缓冲器将所述消息包发送给所述处理器内核。

[0020] 本发明实施例提供的技术方案的有益效果是:

[0021] 可以根据应用场景和数据流量灵活配置,提高了多核处理器处理任务的效率,在不降低系统性能的前提下实现了低功耗;屏蔽了处理器内核的开关状态,从任务分配器来看,所有的处理器内核都是可用的,确保了在降低功耗的同时,不降低对突发数据的处理能力。

附图说明

[0022] 为了更清楚地说明本发明实施例或现有技术中的技术方案,下面将对实施例或现有技术描述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发明的一些实施例,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他的附图。

[0023] 图1是本发明实施例1提供的多核处理器的任务分配系统示意图;

[0024] 图2是本发明实施例1提供的多核处理器的任务分配系统的另一种示意图;

[0025] 图3是本发明实施例2提供的多核处理器的任务分配方法流程图;

[0026] 图4是本发明实施例3提供的多核处理器的任务分配方法流程图;

[0027] 图5是本发明实施例4提供的路径控制器结构示意图;

[0028] 图6是本发明实施例5提供的缓冲器结构示意图。

具体实施方式

[0029] 下面将结合本发明实施例中的附图,对本发明实施例中的技术方案进行清楚、完整地描述,显然,所描述的实施例仅是本发明一部分实施例,而不是全部的实施例。基于本发明中的实施例,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,都属于本发明保护的范围。

[0030] 为了提高多核处理器处理任务的效率,使多核处理器在不降低对突发数据的处理

能力及系统性能的前提下,实现低功耗,本发明实施例提供了一种多核处理器的任务分配方法、系统及设备,详见如下各实施例,对本发明实施例所提供的技术方案进行描述:

[0031] 实施例 1

[0032] 请参照图 1,本发明实施例提供了一种多核处理器任务分配系统 100,所述多核处理器任务分配系统 100,包括任务分配器 20,及多个子处理系统 0 至 X。

[0033] 下面以子处理系统 0 为例进行说明,该子处理系统 0 包括一处理器内核 12、一缓冲器 14、及一状态寄存器 16。其中,缓冲器 14 的输入端和输出端分别与任务分配器 20 和处理器内核 12 相通,用于自任务分配器 20 处接收任务信息,并将该任务信息发送给处理器内核 12。状态寄存器 16 与缓冲器 14 及处理器内核 12 相通,用于将处理器内核 12 和缓冲器 14 的状态,反馈给任务分配器 20。

[0034] 而在实际应用中,所述任务分配器 20 根据该多核处理器任务分配系统中的各个状态寄存器所反映的判断结果,向多个子处理系统 0 至 X 分发任务。

[0035] 在实际应用中,状态寄存器 16 保持监控处理器内核 12 和缓冲器 14 的状态。在本发明实施例中,当所述处理器内核 12 正在处理任务、或缓冲器 14 中保存有任务信息时,状态寄存器 16 向任务分配器 20 反馈其所属的子处理系统处于忙碌状态。而当处理器内核 12 处于等待、休眠、或关闭状态,且缓冲器 14 中并未保存有任务信息时,则状态寄存器 16 向任务分配器 20 反馈其所属的子处理系统处于空闲状态。

[0036] 在实际应用中,状态寄存器 16 向任务分配器 20 反馈的信息中可包括第一状态位,用于表示其所属的子处理系统是否处于空闲状态。

[0037] 进一步地,所述状态寄存器 16 向任务分配器 20 反馈的信息中还可包括第二状态位,用于表示其所属的子处理系统的当前状态,即在空闲状态下所处的具体的某一状态情况,如等待、休眠、或关闭等。其中,当需要分配任务时,处于等待状态的子处理系统的优先级高于处于休眠状态的子处理系统,处于休眠状态的子处理系统的优先级高于处于关闭状态的子处理系统,从而当有多个子处理系统处于空闲状态时,任务分配器 20 可将任务优先分配给处于等待状态的子处理系统,避免了唤醒处于休眠、关闭状态的子处理系统所带来的能源的消耗以及效率的降低。

[0038] 更进一步地,当子处理系统的处理器内核 12 为多线程处理器内核时,所述状态寄存器 16 向任务分配器 20 反馈的信息还可包括第三状态位,用于表示所述处理器内核 12 的空闲程度,比如有几条线程空闲,而任务分配器 20 在分配任务的时候会优先向空闲程度高的处理器内核 12 分配任务。易于理解的是,当由于降低能耗的需要而让部分子处理系统休眠或关闭时,即使其状态寄存器中显示其有较高的空闲程度,只要其他的子处理系统的处理器内核 12 中仍有空闲线程,就不会唤醒处于休眠或关闭状态的子处理系统,从而降低了该系统的工作功耗。

[0039] 在本发明实施例中,所述缓冲器 14 可以采用 FIFO(First In First Out,先入先出缓存队列)形式实现,在其它可选择的实施例中,其也可以采用其它形式的缓冲器,且并不会影响到本发明实施例的实施。

[0040] 本发明实施例的多核处理器任务分配系统 100 通过状态寄存器 16 来判断子处理系统的状态,如果有任务需要进行分配,而处于活动状态的子处理系统均繁忙时。任务分配器 20 根据处于休眠或关闭状态的子处理系统的状态寄存器 16 了解到这些休眠或关闭状态

的子处理系统可以处理任务,于是任务分配器 20 可先将任务存入缓冲器 14 中,并唤醒处于休眠或关闭状态的子处理系统的内处理器内核 12,从而提高了任务处理的效率。

[0041] 在本发明实施例中,可由缓冲器 14 来唤醒其相应的处理器内核 12。

[0042] 本发明实施例充分利用多核处理器结构上的特点,利用缓冲器和状态寄存器,在处理器内核处于休眠或关闭的状态时,任务分配器也可以向其分配任务,以使得当处理器内核被唤醒后可以及时进行处理,提高了任务处理的效率。

[0043] 请参照图 2,本发明实施例还提供另一种多核处理器任务分配系统 200。与本发明实施例提供的多核处理器任务分配系统 100 的相同之处省略说明,本实施例的多核处理器任务分配系统 200 与前一实施例中所述的多核处理器任务分配系统的区别之处在于:每个子处理系统还包括旁路 (Bypass) 213 及路径控制器 215,其中,路径控制器 215 与任务分配器 220 相通,并接受任务分配器 220 发送的任务信息,路径控制器 215 根据状态寄存器 216 中储存的状态信息,选择任务信息的传输通道。具体地,当待接收任务的处理器内核 212 在处理数据时,即该处理器内核 212 处于忙碌状态时,路径控制器 215 选择缓冲器 214 作为任务信息发送的路径;当处理器内核 212 处于等待状态时,路径控制器 215 选择旁路 213 作为任务信息发送的路径,该旁路 213 可以直接将接收的任务消息发送给处于等待状态的处理器内核 212,以加快任务信息的处理速度;当处理器内核 212 处于休眠或关闭状态时,路径控制器 215 将任务信息通过缓冲器 214 发送,该缓冲器 214 接收并缓存任务消息的同时,唤醒处于休眠或关闭状态的处理器内核 212,当该处理器内核 212 被唤醒后,缓冲器 214 将缓存的任务消息发送给处理器内核 212。

[0044] 本发明实施例提供的系统,通过设置状态寄存器以及传输通道,可以根据应用场景和数据流量灵活配置,提高了多核处理器处理任务的效率,在不降低系统性能的前提下实现了低功耗;屏蔽了处理器内核的开关状态,从任务分配器来看,所有的处理器内核都是可用的,确保了在降低功耗的同时,不降低对突发数据的处理能力。

[0045] 实施例 2

[0046] 本发明实施例提供了一种多核处理器的任务分配方法,该方法包括:

[0047] 首先,任务分配器根据状态寄存器的状态判断处理器内核是否空闲;

[0048] 然后,通过传输通道向空闲的处理器内核传输消息包。

[0049] 下面将结合附图对本发明实施例提供的多核处理器的任务分配方法进行说明:本发明实施例提供了一种多核处理器的任务分配方法,且以任务分配器和处理器内核之间设置的传输通道为缓冲器为例进行说明,参见图 3,方法包括:

[0050] 301:任务分配器根据每个处理器内核对应的状态寄存器的忙闲状态查找空闲的处理器内核,然后把消息包通过缓冲器发送给空闲的处理器内核处理。如果该空闲的处理器内核处于休眠或关闭状态,则执行 302;如果该空闲的处理器内核处于等待状态,则执行 304。

[0051] 具体地,对于多线程处理器内核,则要根据内部线程的状态设置状态寄存器,以 8 线程处理器内核为例,一个处理器内核可以同时处理最多 8 个任务,以权重 0-8 来表示处理器内核忙闲状态,若 8 个线程全部为空闲,状态寄存器的第三状态位为 8,若有 3 个被占用,5 个空闲,则状态寄存器的第三状态位为 5。为了避免资源竞争,任务分配器通常优先把任务发给状态寄存器空闲程度高的处理器内核,而多线程处理器内核只有内部全部线程都空

闲时才能关闭处理器内核。任务分配器在活动内核中分配任务时,此时,为了在效率和功耗上取得一个平衡点,可以根据处理器内核的处理能力设定,当活动处理器内核中空闲线程比率高于关闭阈值则关掉一个处理器内核,此内核设为非活动状态,待内部全部线程运行完后,自动停止工作,在所有活动处理器内核中仍然执行空闲程度高的处理器内核优先分配算法。

[0052] 302:处理器内核处于休眠或关闭状态,任务分配器发给处理器内核的消息包暂存在缓冲器中,同时唤醒处理器内核,并将状态寄存器设为忙碌状态。

[0053] 其中,当处理器内核处于休眠状态时,整个系统的功耗降低且处于休眠状态的处理器内核可以很快地被唤醒,缓冲器的缓存压力较小。当处理器内核处于不活动状态时,还可以通过关掉单个处理器内核的电源来关闭处理器内核,这样做可以更为省电,但唤醒时所需要的切换时间相对较长,需要的缓存更大,具体采用哪种方式实现低功耗视应用场合而定。

[0054] 303:处理器内核被唤醒后,对暂存在缓冲器中的消息包处理完毕后,状态寄存器复位为闲状态,然后结束。

[0055] 其中,当状态寄存器复位为空闲状态后,从任务分配器来看,该处理器内核又处于可分配任务状态。

[0056] 304:处理器内核处于等待状态,缓冲器接收任务分配器发来的消息包,并将接收的消息包直接传输给处理器内核处理,同时将状态寄存器设为忙碌状态;

[0057] 其中,由于处理器内核处于等待状态,可以对通过缓冲器传输的消息包直接进行处理,不需要等待。此时,该处理器内核可处理已在接收的消息包,但不接收新消息包。

[0058] 305:当处理器内核处理完消息包后,恢复等待状态,将状态寄存器复位为空闲状态,然后结束。

[0059] 其中,当状态寄存器复位为空闲状态后,从任务分配器来看,该处理器内核又处于可分配任务状态。

[0060] 本发明实施例中多核处理器通过状态寄存器的忙闲状态进行动态开关,屏蔽了处理器内核的开关状态对任务分配器在分配任务时的影响。即使处理器内核处于关闭状态,但从任务分配器来看,该处于关闭状态的处理器内核仍是可用的,且所有的处理器内核都是可用的,确保了在降低功耗的同时,不降低对突发数据的处理能力。

[0061] 实施例 3

[0062] 本发明实施例还提供了一种多核处理器的任务分配的方法,且以任务分配器和处理器内核之间设置的传输通道包括旁路和缓冲器为例,进行说明,参见图 4,方法包括:

[0063] 401:任务分配器根据每个处理器内核对应的状态寄存器所显示的忙闲状态,查找空闲的处理器内核,然后把接收到的消息包发送给空闲的处理器内核处理。

[0064] 其中,当处理器内核为单线程内核时,设置其状态寄存器的空闲或忙碌状态的定义,可以根据具体的需要实现,例如,根据该单线程处理器内核是否正在处理数据,从而定义该单线程内核的忙闲状态,如果该处理器内核正在处理任务,或缓冲器中保存有任务时,则为忙碌状态;反之,当处理器内核并未在处理任务,而缓冲器中亦未有任务时,则为空闲状态,更具体的,所述空闲状态包括处理器内核等待处理任务、休眠和关闭这三种情况。当任务分配器获取到多个内核都处于空闲状态时,可以随机任选出一个空闲内核,也可以根

据预设的优化选择算法,从中选择一个空闲内核,本发明实施例不限制具体的选择方式和方法。

[0065] 其中,当处理器内核为多线程处理器内核时,则要根据内部线程的状态设置状态寄存器,以 8 线程处理器内核为例,即一个处理器内核可以同时处理最多 8 个任务,以权重 0-8 来表示处理器内核忙闲状态,若 8 个线程全部为空闲,状态寄存器的第三状态位为 8,若有 3 个被占用,5 个空闲,则状态寄存器的第三状态位为 5。为了避免资源竞争,任务分配器通常优先把任务发给状态寄存器空闲程度高的处理器内核,而多线程处理器内核只有内部全部线程都空闲时才能关闭处理器内核或使处理器内核休眠。根据多线程处理器内核的状态将多线程处理器内核分为活动与非活动两种状态,任务分配器只在活动内核中分配任务,此时为了在效率和功耗上取得一个平衡点,可以根据处理器内核的处理能力设定,当处于活动的处理器内核中的空闲线程高于关闭阈值则关掉一个处理器内核,此内核设为非活动状态,待内部全部线程运行完后,自动停止工作;在所有工作的处理器内核中仍然在执行任务且空闲程度高的处理器内核优先分配任务。即当多核处理器中的处理器内核为多线程时,则根据多线程中的空闲线程数目,设定空闲程度,每个多线程处理器内核的状态寄存器根据所述空闲程度设定状态值,相应地,任务分配器根据所述空闲程度,优先选择活动内核中空闲线程数目最大的多线程处理器内核进行分配。

[0066] 其中,针对处理器的内核状态进行如下说明:

[0067] (1) 对于单线程内核而言,默认所有内核都可由任务分配器进行任务分配,内核为降低功耗而关闭与否对分配器是透明的。因此这种情况下只有空闲状态(包括等待、休眠、关机)和忙碌状态。

[0068] (2) 对于多线程内核而言,其管理较为复杂,当一个内核中的空闲线程高于关闭阈值则关掉一个处理器内核,此内核设为非活动状态,待内部全部线程运行完后,自动停止工作,只有所有线程都空闲,才能关闭该内核。因此如果所有内核仍由任务分配器分配任务,就可能造成所有内核都工作但每个内核中平均工作线程较少,从降低功耗的角度看是不合理的。应该存在一种算法能够保证工作的内核数较少而每核内平均工作的线程数较多,同时要关掉某个内核时也需要有机制保证其内部全部线程不再分配任务。因此多线程内核有活动和非活动状态的区别,而上述“空闲”、“忙”等状态只在活动内核中有效,活动内核内部关闭与否对分配器也是透明的,任务分配器只在活动内核中分配。而对于非活动的内核也可以看成一直是“忙”状态,实际上不参与工作,任务分配器也不会向其分配。

[0069] 402:与该空闲的处理器内核相对应的路径控制器收到任务分配器发送的消息包后,根据该处理器内核的空闲状态选择一个通道传输消息包。如果处理器内核处于休眠或关闭状态,则执行 203,如果处理器内核处于等待状态,则执行 205。

[0070] 其中,如前文所述,处理器内核的空闲状态,根据空闲的时间长短可为等待状态、休眠或关闭状态,当该处理器内核不处理数据时,则由忙碌状态转为空闲状态,具体地,处理器内核首先会转为等待状态,而当该等待状态的持续时间达到设定时间值时,该处理器内核由等待状态切换到休眠或关闭状态,从而减少功耗。

[0071] 403:当处理器内核处于休眠或关闭状态,选择使用缓冲器传输消息包,传输的消息包由缓冲器缓存,同时唤醒处理器内核,状态寄存器设为忙碌状态。

[0072] 其中,处理器内核空闲时可以采用关掉时钟的方式来实现低功耗,该方式切换唤

醒速度快；也可以采用分别关掉单个处理器内核的电源来实现低功耗，该方式更省电，节省功耗，但切换唤醒时间相对较长，需要的缓存更大，例如对于功耗要求较高时，则可以采用分别关掉单个处理器内核的电源的方式，而对于对切换唤醒时间要求较高时，则可以采用上述第一种关掉时钟的方式。其中，具体采用哪种方式实现低功耗依据具体的应用场合而定。

[0073] 404：处理器内核被唤醒后，对暂存在缓冲器中的消息包处理完后，把输入通道切换到旁路，状态寄存器复位为空闲状态，然后结束。

[0074] 其中，当状态寄存器复位为空闲状态后，从任务分配器来看，该处理器内核又处于可分配任务状态。

[0075] 405：当处理器内核处于等待状态，选择使用旁路传输消息包，旁路开始接收任务分配器发送的消息包，状态寄存器设为忙碌状态，当前处理器内核可处理已在接收的消息包，但不接收新消息包。

[0076] 其中，当处理器内核处于等待状态，选择使用旁路传输消息包，将消息包直接传送给等待状态的处理器内核，实现了处于等待状态的处理器内核对旁路传输消息包的直接处理，此时消息包的处理过程不需要等待，因此，减少传输消息包的功耗和时延。

[0077] 406：当处理器内核处理旁路传输完消息包后，进入等待状态，状态寄存器复位为空闲状态，然后结束。

[0078] 其中，当状态寄存器复位为空闲状态后，从任务分配器来看，该处理器内核又处于可分配任务状态。

[0079] 本发明实施例中多核处理器通过状态寄存器的忙闲状态进行动态开关，任务分配器根据状态寄存器的状态分配内核任务，根据选择的传输通道（缓冲器或旁路）实现对消息包的传输，即使多核处理器处于关闭状态，从任务分配器来看，该关闭的多核处理器仍是可分配任务的状态，确保了在为降低功耗而关闭空闲的处理器内核的同时，又不会降低对突发数据的处理能力。

[0080] 实施例 4

[0081] 参见图 5，本发明实施例提供了一种路径控制器，该路径控制器，包括：

[0082] 接收模块 501，用于接收任务分配器发送的消息包；

[0083] 选择模块 502，用于选择传输通道传输消息包。

[0084] 其中，当传输通道包括旁路和缓冲器时，选择模块 502 具体包括：

[0085] 第一选择单元，用于当空闲的处理器内核的状态为等待状态时，选择旁路作为传输通道，并将接收的消息包发送给旁路；

[0086] 第二选择单元，用于当空闲的处理器内核的状态为休眠或关闭状态时，选择缓冲器作为传输通道，并将接收的消息包发送给缓冲器。

[0087] 本发明实施例提供的路径控制器，可以实现对传输通道的选择，进而根据选择的传输通道实现对消息包的传输，从而提高了多核处理器处理任务的效率，使多核处理器在不降低对突发数据的处理能力及系统性能的前提下，实现低功耗。

[0088] 实施例 5

[0089] 参见图 6，本发明实施例提供了一种缓冲器，该所述缓冲器，包括：

[0090] 接收模块 601，用于接收路径控制器传输的消息包；

[0091] 缓存模块 602,用于将接收模块接收到的消息包缓存;

[0092] 唤醒模块 603,用于唤醒休眠或关闭的处理器内核;

[0093] 传输模块 604,用于当唤醒模块唤醒处理器内核后,将缓冲模块缓存的消息包传输给被唤醒的处理器内核。

[0094] 进一步地,当处理器内核处于空闲状态,则传输模块 604 还用于将接收模块接收的消息包直接发送给处于空闲状态的处理器内核。

[0095] 本发明实施例提供的缓冲器,当处理器内核为休眠或关闭状态时,对接收的消息包缓存的同时,并唤醒该处理器内核,当唤醒处理器内核后,将缓存的消息包发送,从而提高了多核处理器处理任务的效率,使多核处理器在不降低对突发数据的处理能力及系统性能的前提下,实现低功耗。

[0096] 通过以上的实施方式的描述,本领域的技术人员可以清楚地了解到本发明可以通过硬件实现,也可以借助软件加必要的通用硬件平台的方式来实现。基于这样的理解,本发明的技术方案可以以软件产品的形式体现出来,该软件产品可以存储在一个非易失性存储介质(可以是 CD-ROM, U 盘,移动硬盘等)中,包括若干指令用以使得一台计算机设备(可以是个人计算机,服务器,或者网络设备等)执行本发明各个实施例所述的方法。

[0097] 以上所述仅为本发明的较佳实施例,并不用以限制本发明,凡在本发明的精神和原则之内,所作的任何修改、等同替换、改进等,均应包含在本发明的保护范围之内。

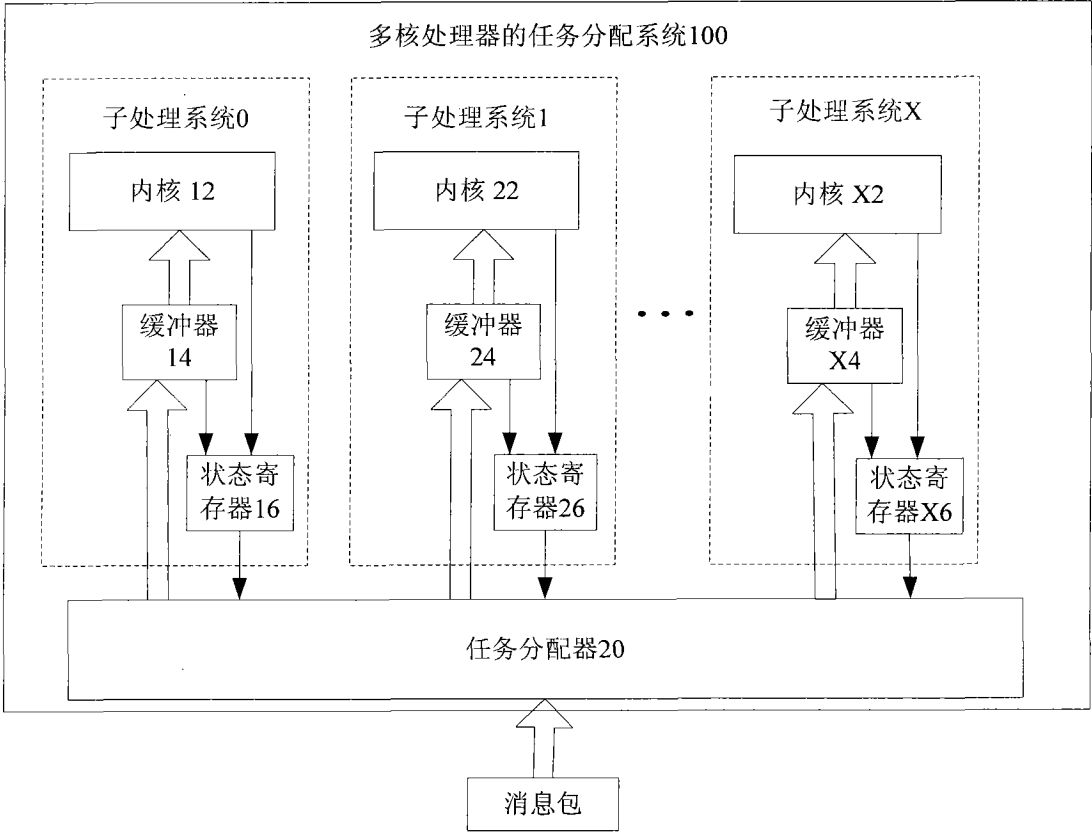


图 1

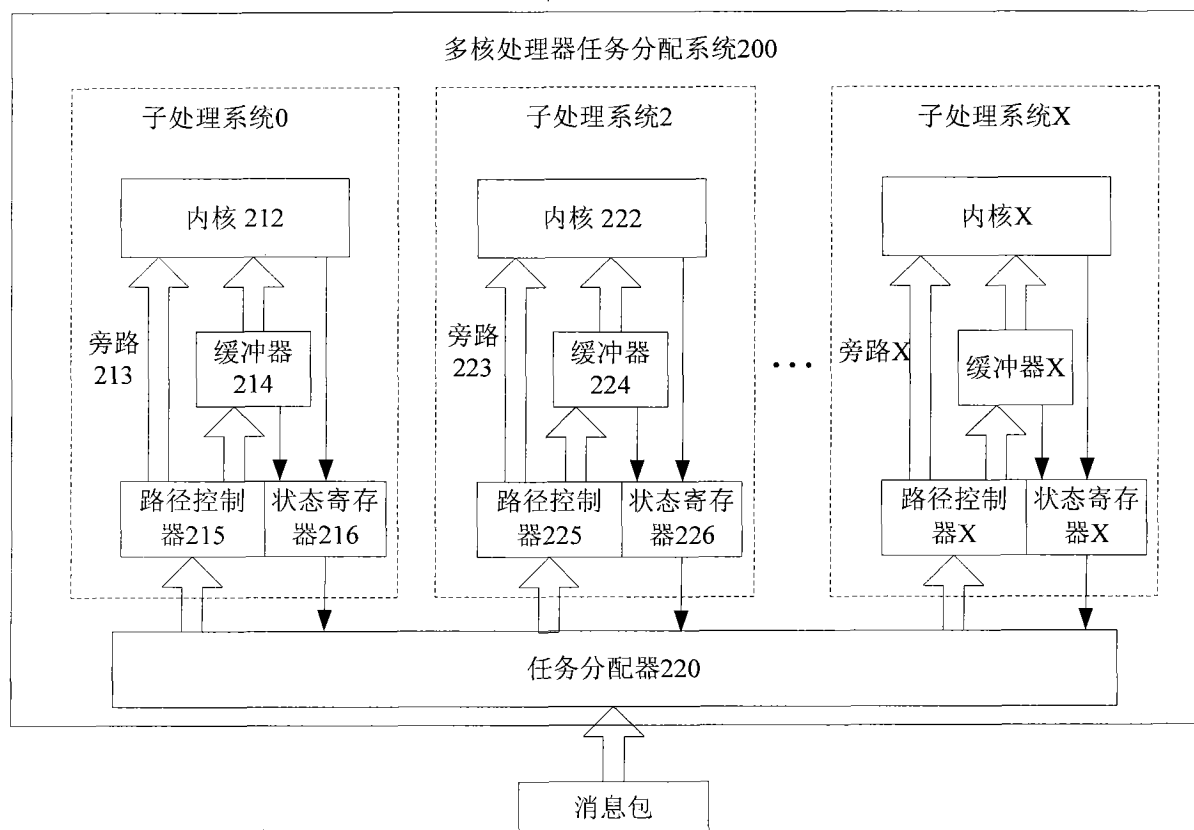


图 2

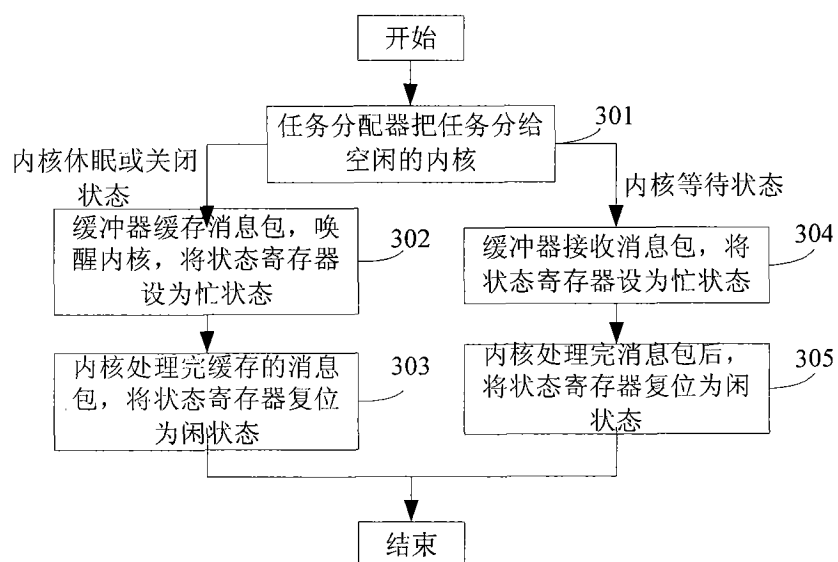


图 3

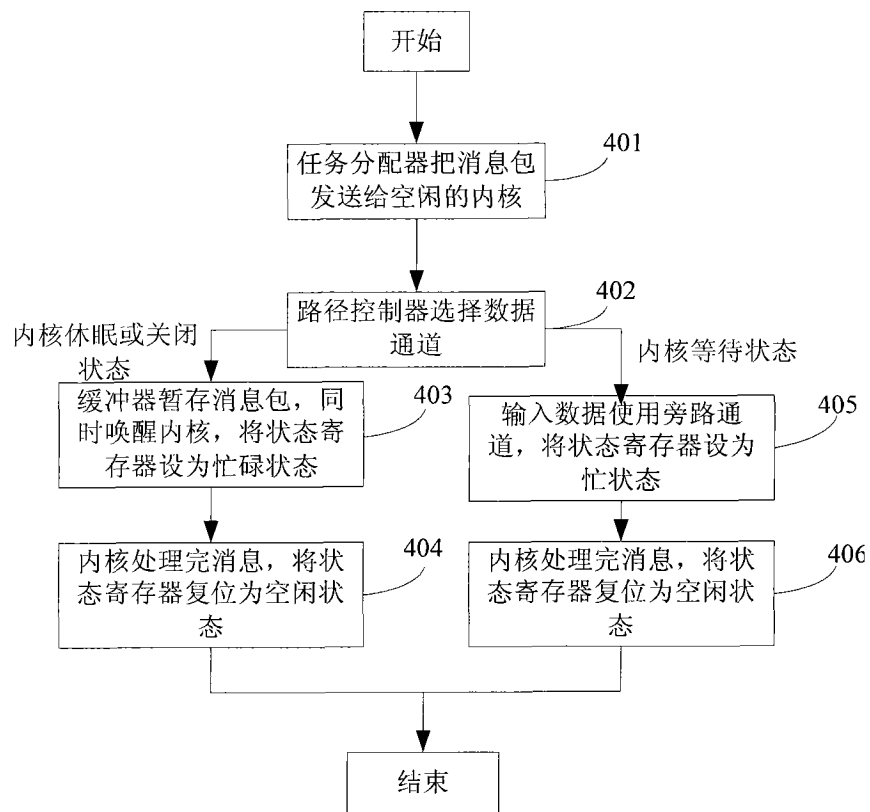


图 4

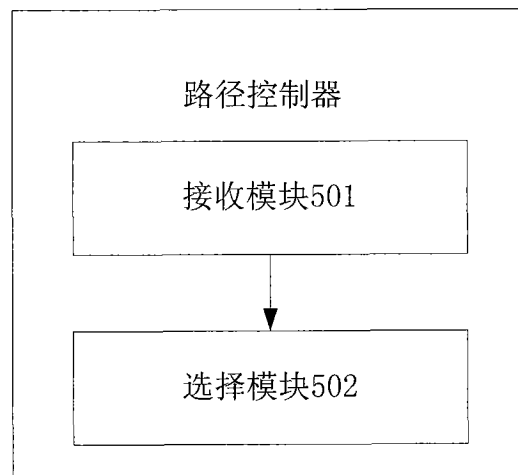


图 5

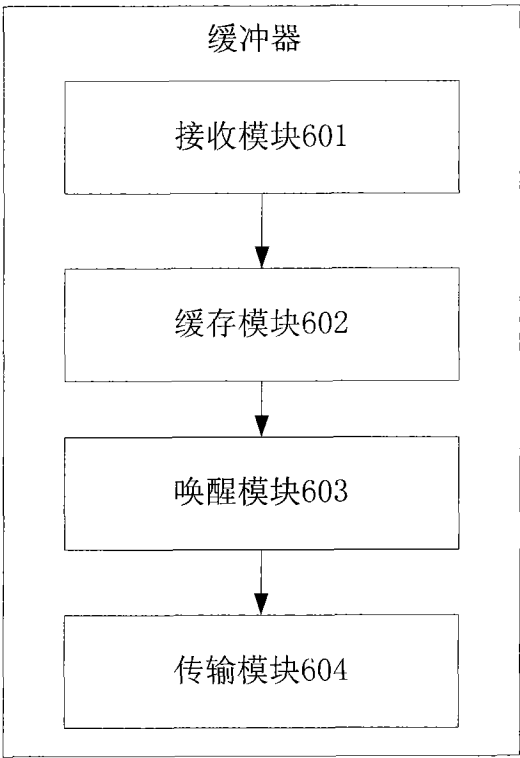


图 6