(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 108388498 B
(45)授权公告日 2020.09.08

(21)申请号 201810145805.3

(22)申请日 2018.02.12

(65)同一申请的已公布的文献号
申请公布号 CN 108388498 A

(43)申请公布日 2018.08.10

(73)专利权人 北京大学
地址 100871 北京市海淀区颐和园路5号

(72)发明人 孙广宇 张超 孟彤

(74)专利代理机构 北京万象新悦知识产权代理有限公司 11360
代理人 黄凤茹

(51)Int.Cl.
G06F 11/30(2006.01)
G06F 11/34(2006.01)

(56)对比文件
CN 103902462 A,2014.07.02
CN 105760624 A,2016.07.13
CN 104598310 A,2015.05.06
CN 103106131 A,2013.05.15
CN 107092493 A,2017.08.25
US 7992033 B2,2011.08.02
US 2016098200 A1,2016.04.07
孙广宇等.《面向非易失内存的结构和系统级设计与优化综述》.《华东师范大学"数据科学与工程"论坛内存计算数据管理主题报告会论文集》.2014,72-81页.

审查员 杨帆

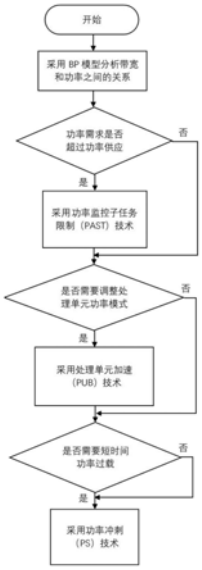
权利要求书3页 说明书8页 附图3页

(54)发明名称

内存中计算的功率建模方法及功率管理方法

(57)摘要

本发明公布了内存中计算的功率建模方法及功率管理方法，首先建立BP模型，采用每功率带宽BP表示内存中计算PIM中的带宽和功率之间的关系；功率管理方法包括功率监控子任务限制PAST、处理单元加速PUB和功率冲刺PS；当功率供应超过功率需求时，采用PAST管理PIM内功率消耗；当需要采用动态调整功率模式时采用PUB方法；当需要短时间的功率过载时，采用PS方法提高功率仲裁器的功率上限值；实现内存中计算的功率管理。采用本发明的BP模型得到的功率和实测相符，其中PAST方法能成功限制PIM的功率，PUB能成功提升芯片的性能；同时采用PAST、PUB和PS方法，能够产生更有效的能源系统，合理配置功率管理方案，进一步提升PIM的性能。



CN 108388498 B

1.一种基于内存中计算的功率管理方法,首先建立BP模型,采用每功率带宽BP表示内存中计算PIM中的带宽和功率之间的关系;所述功率管理方法包括功率监控子任务限制PAST、处理单元加速PUB和功率冲刺PS;当功率供应超过功率需求时,采用PAST对PIM内功率消耗进行管理;当需要采用动态调整功率模式时,采用PUB方法;当需要短时间的功率过载时,采用PS方法提高功率仲裁器的功率上限值;由此实现基于内存中计算的功率管理;

A)采用功率监控子任务限制PAST管理功率,针对PIM任务的功率需求可能超过功率供应限制进行功率管理;PAST方法采用的装置为两级功率仲裁系统,两级功率仲裁系统包含多个内存芯片和一个功率仲裁器L2;单个内存芯片内部包括网络接口、重排子任务队列、内存块即处理单元、功率仲裁器L1;功率仲裁器包括算术逻辑单元、数据选择器和计数器;在每个内存芯片内,由功率仲裁器L1控制内部的内存块;

采用PAST方法管理功率包括如下步骤:

A1)内存芯片内部部件从网络连接中获得请求,将任务划分成多个子任务,存储在子任务队列中,再对需求发送方进行应答;每一个子任务由一个存储器端的处理单元PU完成;

A2)在任何内存块的执行阶段之前,子任务队列通过使用ACQUIRE信号和需要的功率值P从功率仲裁器获得功率许可;

A3)子任务队列将一个子任务发射到一个内存块,该内存块新建一个子任务到队列的末尾;当有足够的功率运行一个新的子任务时,功率仲裁器发送一个START信号到该内存块使其开始执行;否则,该内存块被暂停,功率仲裁器将子任务对功率的需求放入子任务重排队列;直到具有足够的功率预算,内存块被激活;当整个任务都被内存块完成后,向功率仲裁器发送RELEASE信号,释放为那个内存块分配的功率;

B)采用处理单元加速方法PUB管理功率,将处理单元的功率模式按功率需求划分成多级功率模式;通过采用简单调度方法或优化调度方法为PIM内的多个处理单元分配功率模式,动态调整处理单元的功率模式以提升关键路径中子任务的性能;

B1)简单调度方法执行如下操作:

每次仅发布一个子任务;当子任务队列中没有需要在该处理单元上执行的子任务时,该处理单元处于低功率模式;一旦添加一个队列条目后,处理单元的功率模式由低功率模式升级为高功率模式;

功率仲裁器评估当前剩余功率与所需功率;从划分好的最高功率模式到最低功率模式进行扫描,如果有足够的空闲功率,处理单元以该功率模式开始执行;如果PU无法开始执行,功率仲裁器将当前正在运行的处理单元从高功率模式降低到低功率模式;如果PU仍然无法开始,暂停队列,等待足够的空闲功率;由此实现为PIM内的多个处理单元分配功率模式;足够的空闲功率指的是空闲功率值高于模式的功率;

B2)优化调度方法,基于子任务的有向无环图,以三状态有限状态机方式FSM运作;三状态为:READY、UPDATE和CHECK;具体执行如下操作:

首先将FSM初始化,置于READY状态;

当有子任务结束时,引发UPDATE状态,并更新图和当前可用功率的计数器,然后返回到READY状态;

当有更新时,状态转移到CHECK,再确定将要发布的子任务的功率模式;

当一个子任务在CHECK状态结束时,状态变回READY后转移到UPDATE;

由此实现为PIM内的多个处理单元分配功率模式；

3)采用功率冲刺方法PS管理功率，使得在短时间内提供过载的功率，再返回到欠载功率状态进行恢复；具体执行如下操作：

将处理单元的执行阶段划分为：正常执行阶段、冲刺阶段和恢复阶段；

在冲刺阶段，通过PAST和PUB方法提供更多电流，提高功率仲裁器的功率上限值，从而提升处理单元处理任务时的功率；

当冲刺阶段结束处于恢复阶段时，功率仲裁器向队列和正在运行的多个处理单元发送一个额外的PAUSE命令，减少功率消耗，使功率达到以往的功率上限值，通过降低功率恢复正常状态。

2.如权利要求1所述基于内存中计算的功率管理方法，其特征是，所述建立的BP模型表示为式1：

$$\mathrm{BP}=\frac{B}{DP+LP}=\frac{1}{\sqrt{C}e_r+r_we_S+e_C+\frac{1}{B}(CP_l+P_C)} \quad (式1)$$

式1中，BP为每功率带宽；B表示内存所使用的带宽；DP为动态功率；LP为泄漏功率；P＝DP+LP，表示PIM元件的总功率；$\sqrt{C}e_r+r_we_s+e_c$是将PIM所使用的动态能量正规化到一个比特；$\sqrt{C}e_r$部分表示用于到达目标单元的搜索路径的耗能；e_s表示完成几个内存之间的电阻切换或状态转换；e_c表示用于计算的能量；CP_l+P_C表示泄漏功率；内存的泄漏功率和容量相关，以CP_l表示，P_C代表内核以及内存控制的泄漏功率。

3.如权利要求1所述基于内存中计算的功率管理方法，其特征是，当内存芯片内部有n个处理单元时，最多同时执行n个子任务。

4.如权利要求1所述基于内存中计算的功率管理方法，其特征是，所述重排子任务队列扩展FIFO队列方法，包括更多的头指针和尾指针并向队列添加更多的选项，允许多个子任务同时进行；所述重排子任务队列的每个条目包含5栏：下标ID、子任务命令Command、依赖掩码Mask、功率说明Power及状态S；下标ID附加在每个条目上；子任务命令Command包含对相应处理单元或存储器banks的函数调用块；功率说明Power为在该处理单元内部完成计算和数据存取所需的预期功率；状态S表明子任务的状态是处于待定PD、已发布IS还是已完成CP。

5.如权利要求4所述基于内存中计算的功率管理方法，其特征是，所述重排子任务队列中，一旦一个子任务进入了队列，该子任务的初始状态是待定PD；仅有当满足条件时该子任务才会被发布；所述条件为：该子任务所有依赖的条目已经完成且能够满足功率需求；一旦队列从相应处理单元收到完成信号，状态即改变为已完成CP；队列头处的已完成条目会被撤走，让出空间给接下来的子任务；设定一个额外的计数器用于计算正在进行的子任务的数目；如果所有队列的头均充满待定PD或已发布IS状态的子任务，队列即停止。

6.如权利要求1所述基于内存中计算的功率管理方法，其特征是，将内存块处理单元PUB的功率模式按功率需求划分为两级功率模式，分别为活跃模式和加速模式；当子任务队列中没有需要在该处理单元上执行的子任务时，该处理单元处于活跃模式；在添加一个队列条目后，该处理单元从活跃模式升级为加速模式。

7.如权利要求1所述基于内存中计算的功率管理方法,其特征是,所述空闲功率为功率上限减去已使用功率。

内存中计算的功率建模方法及功率管理方法

技术领域

[0001] 本发明涉及内存中计算技术,具体涉及一种基于内存中计算的功率建模方法和功率管理方法。

背景技术

[0002] 内存中计算(Processing-in-Memory),简称为PIM,是一种将计算转移到内存的思想,目前被广泛应用于对数据敏感性应用进行加速。随着众多以数据为中心的应用,例如实时分析、图计算和神经网络算法等,对高速度、高带宽的数据检索需求日益强烈,PIM设计正逐渐成为学界、业界的研究热点。PIM相关的计算任务(例如单词计数,范围查找等等)通常都较为简单却包含了内存中大量甚至全部的数据。因此,可以将一个PIM任务划分并转移到多个内存处理器单元中(例如HMC存储库和RRAM交叉开关阵列),使其本地并行完成任务。这样可以更加有效地利用到bank级别甚至是cell级别的带宽,从而显著地提高数据处理速度。带宽利用度的提升带来了功率消耗成本的增长。以往的研究曾涉及到关于PIM设计的高功耗问题,例如内存的冷却系统需要重新设计来应对增加的功耗。这导致了额外的散热成本和设计复杂度,并影响了PIM的逻辑行为。另外,PIM中功率和数据处理吞吐量之间的关系还没有被很好地研究过。在没有一个合适的功率模型引导的情况下,PIM结构可能会被设计为不匹配的内存带宽和功率供应关系,以往的PIM设计在巅峰吞吐量时的功耗可能会超过功率供应。

发明内容

[0003] 为了克服上述现有技术的不足,本发明提供针对内存中计算PIM的功率模型(BP模型)和功率管理方法及相应的装置;通过本发明技术方案可以对内存中计算PIM架构中的功耗和带宽进行建模,得到PIM中功率和带宽之间的关系,可用于PIM功耗系统的早期设计,降低PIM对功耗的需求,提升处理单元的性能,并动态调整提升能量效率。

[0004] 本发明基于内存中计算PIM的功率建模涉及到:带宽使用、内存容量和内存类型之间的关系。本发明采用术语“每功率带宽”(Bandwidth per Power,简称BP)来描述带宽和功率之间接近线性的关系,用B/P来计算,带宽B表示内存所使用的带宽,分母为功率P,表示PIM元件的总功率,包括动态功率(dynamic power,简称DP)和泄漏功率(leakage power,简称LP)。动态功率是指用来在取回的数据上进行计算以及存取数据所使用的功率,包含激活cell、驱动字线以及放大感应等等。泄漏功率是用来保存数据所消耗的功率,包含了刷新功率、保证解码器和计算逻辑活跃的功率、以及寄生电路所泄露的功率。通过在一个时间周期内有多少读或写的位来测量带宽。因为核心可能位于内存片内,核心使用的带宽可能高于芯片的IO(Input/Output,输入/输出)带宽,且小于整合的cell级别的带宽。理论上的最大带宽是当所有的cells都并行不停地访问时的cell级别带宽。

[0005] 本发明提供的技术方案是:

[0006] 一种内存中计算的功率建模方法,采用每功率带宽(BP)表示带宽和功率之间的关

系;建立BP模型;具体建模公式如式1:

[0007] $$\mathrm{BP}=\frac{B}{DP+LP}=\frac{1}{\sqrt{C}e_r+r_we_S+e_C+\frac{1}{B}(CP_l+P_C)} \quad (式\ 1)$$

[0008] 式1中,BP为每功率带宽;B表示内存所使用的带宽;DP为动态功率;LP为泄漏功率;P＝DP+LP,表示PIM元件的总功率;$\sqrt{C}e_r+r_we_s+e_c$是将PIM所使用的动态能量正规化到一个比特;如果所有的数据存取为读(或写),参数值写比率(r_w)是0(或者1),这个值在0和1之间变动。$\sqrt{C}e_r$部分表示用于到达目标单元的搜索路径的耗能,因此和容量相关。e_s表示完成几个内存之间的电阻切换或状态转换,因此和容量无关。用于计算的能量用e_c来表示。泄漏功率使用CP_l+P_C来表示。内存的泄漏功率和容量相关,以CP_l表示,P_C代表内核以及内存控制的泄漏功率。

[0009] 基于上述PIM的功率模型(BP模型),本发明提供基于内存中计算的功率管理方法及硬件装置,包括:功率监控子任务限制(PAST)、处理单元加速(PUB)和功率冲刺(PS);首先利用BP模型进行计算得到针对不同内存的带宽和功率之间的关系,当功率供应超过功率需求时,采用PAST技术来对PIM内功率消耗进行管理;当需要采用动态调整功率模式时,采用PUB技术,可提升关键路径子任务的性能;当需要短时间的功率过载时,采用PS技术来提高功率仲裁器的功率上限值。

[0010] A)功率监控子任务限制(Power-Aware Subtask Throttling,简称PAST)方法,用于解决PIM任务的功率需求可能超过功率供应限制的问题。

[0011] PAST设计中,单个内存芯片内部包含一个网络接口、一个重排子任务队列、一个内存块(即处理单元)和一个L1功率仲裁器;PAST方法可采用两级功率仲裁系统或更多级功率仲裁系统。两级功率仲裁系统包含多个内存芯片和一个L2功率仲裁器。其中,功率仲裁器L1或L2均包含一个算术逻辑单元,一个数据选择器和一个计数器。

[0012] 采用PAST方法管理功率,具体包括如下步骤:

[0013] A1)芯片内部PAST部件(请求到达PAST的网络接口)从网络连接中获得请求,将任务划分成多个子任务,存储在子任务队列中,再对需求发送方进行应答;

[0014] 一个子任务由仅有一个存储器端的处理单元(PU)完成;如果整个内存有多个(n个)PU,则会同时最多有n个子任务在执行;

[0015] A2)在任何内存块(为PIM设计中的处理单元)的执行阶段之前,子任务队列使用一个ACQUIRE信号和需要的功率值(P)从功率仲裁器获得功率许可;

[0016] A3)子任务队列将一个子任务发射到一个内存块,该内存块也新建一个子任务到队列的末尾;如果有足够的功率来运行一个新的子任务,则功率仲裁器L1发送一个START信号到内存块使其开始执行;否则,这个内存块被暂停;然后功率仲裁器将子任务对功率的需求放入子任务重排队列;直到具有足够的功率预算,处理单元(内存块)才会被激活。在整个任务都被内存块完成后,会向功率仲裁器发送一个RELEASE信号来释放为那个内存块分配的功率。

[0017] B)处理单元加速(Processing Unit Boost,简称PUB),通过动态调整处理单元的功率模式,提升关键路径中子任务的性能;

[0018] 本发明将处理单元的功率模式按功率需求划分成多级功率模式。例如，可将处理单元的功率模式划分为两级：活跃模式和加速模式。其中，活跃模式的功率需求较低，加速模式的功率需求较高，之后将以这种两级功率模式划分为例进行说明。基于这种多级功率模式划分方法，可利用PIM设计的特点，将PUB当作一种动态电压频率调整(DVFS)设计。PUB的目标是给PIM内的多个PU分配功率模式，设计的关键是调度算法。本发明提出两种调度算法，分别为简单调度算法和优化调度算法。

[0019] B1)通过简单的调度算法给PIM内的多个PU分配功率模式；

[0020] 简单的算法设计为功率仲裁器每次仅发布一个子任务：如果子任务队列中没有需要在该处理单元上执行的子任务(标注着这个处理单元的标号)，这意味着它不会被使用，单元会处于活跃模式。一旦添加一个队列条目后，相关PU的功率模式会升级(处理单元PU从活跃模式升级为加速模式)。然后功率仲裁器评估当前剩余功率与所需功率。从最高功率模式到最低模式进行扫描，如果空闲功率值(功率上限减去已使用功率)高于扫描到的模式的功率需求值(处理单元PU的功率需求值)，PU会以这个功率模式开始执行。如果PU无法开始执行，功率仲裁器则将当前正在运行的PU从高功率模式降低到低功率模式(如划分为两级功率模式，从高功率模式降低到低功率模式即从加速模式到活跃模式)。如果PU仍然无法开始，队列会暂停以等待足够的空闲功率。

[0021] B2)通过优化调度算法给PIM内的多个PU分配功率模式；

[0022] 优化PUB基于子任务的有向无环图，是一种针对功率仲裁器的贪心算法。该算法以三状态有限状态机方式(FSM)运作：READY，UPDATE和CHECK。初始化算法将FSM置于READY状态。如果有子任务结束，会引发UPDATE状态，并更新图和当前可用功率的计数器，然后返回到READY状态。如果有更新，状态会转移到CHECK，然后决定将要发布的子任务的功率模式。如果一个子任务在CHECK状态结束，状态变回READY后会转移到UPDATE。

[0023] 3)功率冲刺(Power Sprinting)，在短时间内提供过载的功率，然后返回到欠载功率状态来进行恢复。

[0024] PS将处理单元的执行阶段划分为：正常执行阶段、冲刺阶段和恢复阶段。PS在冲刺阶段通过PAST和PUB方法，以提供更多电流的方式，提高功率仲裁器的功率上限值，从而提升处理单元处理任务时的功率。当冲刺阶段结束处于恢复阶段时，功率仲裁器向队列和正在运行的多个PU发送一个额外的PAUSE命令，减少功率消耗使其达到以往的功率上限值，通过降低功率来恢复正常状态。

[0025] 与现有技术相比，本发明的有益效果是：

[0026] 本发明提供了一种针对内存中计算的功率模型和功率管理技术，通过架构级别的仿真，针对不同内存类型，对内存中计算的功率建模并实施功率管理。具体实施中对本发明技术方案的性能和获得的功率提升进行了评估。具体实施表明，采用本发明技术方案的功率模型得到的功率和实测相符，其中PAST方法能成功限制PIM的功率，PUB能成功提升芯片的性能；同时采用PAST的硬件模型，PUB功率模式调度方法，和PS功率冲刺方法，能够产生一个更加有效的能源系统，能够合理配置功率管理方案，可进一步提升PIM的性能。

附图说明

[0027] 图1为本发明方法的流程框图。

[0028] 图2为本发明的收集到的数据点与BP模型预测的动态功率和泄漏功率之间的对比示意图；

[0029] 图中，(a1)、(a2)、(a3)分别表示PCM的动态读功率、动态写功率和泄漏功率与容量之间的关系；(b1)、(b2)、(b3)分别表示STTRAM的动态读功率、动态写功率和泄漏功率与容量之间的关系；(c1)、(c2)、(c3)分别表示RRAM的动态读功率、动态写功率和泄露功率与容量之间的关系；(d1)、(d2)、(d3)分别表示DRAM的动态读功率、动态写功率和泄漏功率与容量之间的关系。

[0030] 图3为本发明实施例中基于功率监控子任务限制(PAST)过程的硬件设计结构框图；

[0031] 其中，(a)表示二级仲裁器结构，(b)表示芯片内部的交互，(c)表示重排子任务队列，(d)表示功率仲裁器的实现；①为ACQUIRE信号；②为START信号；③为RELEASE信号；ISSUE为将任务发射到一个内存块；NEW为内存块新建一个子任务到队列的末尾。

[0032] 图4为本发明实施例中处理单元加速(PUB)示意图；

[0033] 其中，(a)为子任务的有向无环图(灰色节点表示处于关键路径)；(b)为执行过程中的功率消耗。

[0034] 图5为本发明实施例的功率冲刺示意图，表示功率需求和供应之间的关系，并标记了内存冲刺的主要阶段；

[0035] 其中，t_S为执行时间中的冲刺时间；t_R为执行时间中的恢复时间；t_N为执行时间中的正常执行时间。

[0036] 图6为本发明实施例结合PAST、PUB和PS的实验结果图；

[0037] 其中，MA、TS、AW、TF、PR、BF分别为采用矩阵加法、树搜索、数组游走、平均青年追随者、网页排序、贝尔曼-福特算法；对于10W，15W和20W的功率上限以及4W和8W的功率冲刺，每个HMC立方体的正规化加速比。

具体实施方式

[0038] 下面结合附图，通过实施例对本发明做进一步说明。

[0039] 本发明首先对内存中计算(PIM)中功率和带宽之间的关系进行了建模，提出了BP模型，并基于BP模型提供了功率管理方法及其硬件装置。

[0040] 图1为本发明方法的流程框图。根据本方法提出的BP模型，针对不同内存，对带宽和功率之间的关系进行建模和分析。然后根据分析结果，如果功率供应超过功率需求，则采用功率监控子任务限制(PAST)技术来对PIM内功率消耗进行管理；如果需要动态调整功率模式，则采用处理单元加速(PUB)技术，来提升关键路径子任务的性能；如果需要短时间的功率过载，则采用功率冲刺(PS)技术来提高功率仲裁器的功率上限值。

[0041] 表1不同内存类型对应的参数的值和定义

[0042]

内存类型	PCM	STT-RAM	RRAM	3D DRAM
$e_r(J/b)$	8.15×10^{-17}	2.10×10^{-16}	1.17×10^{-16}	5.90×10^{-17}
$e_S(J/b)$	1.13×10^{-10}	5.27×10^{-13}	7.52×10^{-13}	2.03×10^{-14}
$P_l(W/b)$	3.80×10^{-12}	2.79×10^{-12}	1.20×10^{-11}	3.94×10^{-11}

[0043] 本发明针对不同内存类型，使用从以往校验模拟工具和文献中采集的数据来对模型进行校验：从NVsim收集了自旋扭矩转换磁存储器(spin torque transfer random access memory，简称STTRAM)，相变化内存(phase change memory，简称PCM)，以及阻变存储器(resistance random access memory，简称RRAM)的数据；从cacti-3DD收集了3D堆叠动态随机存储器(3D stacked dynamic random memory，简称3D DRAM)的数据。不同内存类型对应的参数的值和定义如表1所示。内存所使用的功率由其动态能量和泄漏功率进行验证。本发明提出的模型与收集到的数据结果契合。

[0044] 图2展示了在PCM、STTRAM、RRAM和RRAM四中不同内存类型上，模型预测的动态读功率、动态写功率和泄露功率与收集到的数据点之间的关系。可以看出，预测值和实测值之间的差距很小，说明本发明提出的BP模型对PIM中带宽和功率的关系进行了很好的描述，具有很强的应用性。

[0045] 首先，根据功率供应和功率需求之间的关系，可以采用功率监控子任务(PAST)技术，来降低PIM对功率的需求，PAST的具体硬件设计图如图3所示，该PAST部件从网络连接中获得请求，将任务划分成多个子任务，然后将它们存储在子任务队列中，再对需求发送方进行应答。一个子任务由仅有一个存储器端的处理单元(PU)完成。如果整个内存有多个(n个)PU，则会同时最多有n个子任务在执行。在任何内存块(为PIM设计中的处理单元)的执行阶段之前，子任务队列需要使用一个ACQUIRE(图3中的①)信号和需要的功率值(P)从功率仲裁器获得功率许可。队列将一个子任务发射(ISSUE)到一个内存块，该内存块也新建(NEW)一个子任务到队列的末尾。如果有足够的功率来运行一个新的子任务，会有一个START(图3中的②)信号发送到内存块然后使其开始执行。否则，这个内存块会被暂停。然后功率仲裁器将这个需求放入一个队列。直到具有足够的功率预算，单元都不会被激活。在整个任务都被内存块完成后，会向功率仲裁器发送一个RELEASE(图3中的③)信号来释放为那个内存块分配的功率。

[0046] 具体结合硬件设计，PAST的实现方式如下：

[0047] 芯片内部PAST部件从网络连接中获得请求，将任务划分成多个子任务，然后将它们存储在子任务队列中，再对需求发送方进行应答。图3(b)展示了一个芯片内部PAST部件的结构和组件之间交互。

[0048] 如图3(a)所示，本发明基于PAST提出了一个用于具有多个内存芯片的PIM设计的两级功率仲裁系统。芯片由网络进行连接，包含了单个内存的PAST部件，并添加了一个共享的二级仲裁器(图3中的L2)。在每个芯片内，一个功率仲裁器L1采用PAST中的方法来进行任务划分和功率管理，控制其内存块的执行。因此，二级功率仲裁系统中的每个芯片都采用PAST方法进行功率管理和任务划分。这种两级设计增加了功率仲裁系统的可扩展性，也使得在内存芯片之间能实现可调整功率再分配。具体实施中，也可通过使用比两级更多的级

数，使可扩展性进一步得到提升。此两级仲裁系统和两级缓存系统的工作方式相似。L2仲裁器保存内存的总功率预算，L1仲裁器仅保存它自己的芯片的功率值。L1的功率预算可以通过从L2获取(或释放)一部分功率而增加(或减少)。所有的L1预算之和等于L2的预算。每一个芯片的子任务首先会查询本地仲裁器(L1)来获取功率。如果具有足够的功率，仲裁器会对多个PU返回一个START作为应答。否则，L1仲裁器会查询L2仲裁器。在L1和L2之间交换的功率预算的粒度是P的几倍。

[0049] 如图3(c)所示，基于PAST本发明还提出了重排子任务队列，以支持子任务互相依赖。重排队列是对现有的FIFO队列方法的一种延伸，它扩展了更多的头和尾指针并向队列添加了更多的选项，允许多个子任务同时进行。队列的每个条目包含5栏：下标(ID)、子任务命令(Command)、依赖掩码(Mask)、功率说明(Power)以及状态(S)。下标附加在每个条目上；子任务命令包含了对相应PU(或存储器banks)的函数调用块；功率说明是在该PU内部完成计算和数据存取所需的预期功率；状态表明子任务是处于待定(PD)、已发布(IS)还是已完成(CP)。一旦一个子任务进入了队列，它的初始状态是待定(PD)。仅有当以下两个条件满足时它才会被发布：(1)它所有依赖的条目已经完成，(2)能够满足功率需求。一旦队列从相应PU收到了完成信号，状态就改变为已完成(CP)。队列头处的已完成条目会被撤走，为接下来的子任务让出空间。和现有的FIFO实现相同，一个额外的计数器用于计算正在进行的子任务的数目。如果所有队列的头都充满了待定(PD)或已发布(IS)状态的子任务，队列会停止来保证公平。

[0050] 仲裁器在硬件中通过一个简单的整数算术逻辑单元、一个寄存器和几个数据选择器实现。如图3(d)中所示。计数器用于记录这个功率仲裁器控制的目前可用的功率。从计数器的值减去给定的功率值。如果结果是正值，会发送一个START信号，计数器中的功率值得到更新。

[0051] 然后，根据是否需要对处理单元进行动态调整，决定是否采用处理单元加速(PUB)技术，来提升处理单元的性能。PUB举例如图4所示。

[0052] PIM的任务被划分为7个子任务(A-G)，子任务之间的箭头表示依赖关系：C指向A表示A应该在C之前完成。在本例子中每个处理单元具有两种处理模式：活跃和加速。加速模式下的功率消耗大概是活跃模式下的2倍，活跃模式下的延时大概是加速模式下的1.5倍。在本例子中，功率上界(P_{max})是3，活跃模式的处理单元功率正规化为1。

[0053] 我们设计的算法首先找到两个空闲节点(无父节点的节点)，将具有更多子节点的节点(B)进行升级(从活跃模式升级到加速模式)。因为功率上限是3，A只能更新到活跃模式。此时A和B分配到的功率模式为[活跃，加速]。当B完成后，剩余功率变为2，另外两个节点(D和E)空闲，若将这两个节点设置为活跃模式后，就没有剩余功率，因此将他们设置为[活跃，活跃]。在D和E执行结束后，只有节点F空闲，因此将其功率模式设置为[加速]。当C结束后，无法找到空闲节点，因此等到F结束后，G才会以最高模式，即加速模式发布。G完成后，PIM的执行结束。

[0054] 最后，为了能灵活进行内存功率的分配，可以采用功率冲刺(PS)来进行短时间的功率过载和之后的欠载，从而达到更好的能量效率。

[0055] 图5展示了功率冲刺的三个主要阶段：正常，冲刺和恢复。t_N，t_S，t_R用于代表这些主要阶段消耗的时间。最小的恢复时间(t_R)是用于恢复冲刺消耗的额外功率资源的时间和额

外热量的散热时间两者的最大值。在恢复阶段后，内存返回到正常阶段，这时它已准备好下一次冲刺。在有限功率供应下，功率需求更好地得到了满足。

[0056] 限制功率冲刺能力的关键因素是封装的热电容。以往的工作使用块状金属或相变材料来存储热量，并使用超级电容器来存储能量。热量由这些材料进行存储，并最终通过散热器排出。对于一个4Gb的HMC而言，我们在封装内部附加了一块金属，并在封装旁边放置了一个独立超级电容器。一块1mm宽的铜(3.45J/cm^{3}K)被展成227mm^2，并使用了一个1F的超级电容器。充电延迟被设置为和散热时间相同。我们假设冲刺和恢复的效率都是90％。对于一个具有1s的冲刺持续时间(t_S)和10s的恢复持续时间(t_R)的额外4W冲刺功率而言，在冲刺阶段增加的温度是5.1C，并且必须将0.49W的功率分配用于在恢复阶段对超级电容器进行充电。

[0057] 图6中展示了同时采用三种方法的实现结果。

[0058] 我们在SMCSim(Smart Memory Cube Simutor，智能内存块模拟器，一种基于gem5的高层次模拟环境)上搭建了本发明的评价系统，这是一个完整的系统PIM平台：它采用gem5，DRAMSim2和ModelSim来实现闭环仿真。缓存的功率消耗通过McPat进行评估。HMC功率通过Micron SDRAM功率计算器和CACTI-3DD进行收集，并参照发布的HMC数据进行缩放。从模拟器收集到的数据用于计算性能，带宽和功率消耗。

[0059] 表2 PIM平台架构的参数

[0060]

组件	配置
主处理器	4 CMPs, 4 out-of-order cores/socket, 4GHz, 4-issue, 128-ROB, 32 KB private I/D cache, 2MB shared L2 per socket
内存	32 cubes, 8 4Gb DRAM layers, 16 vault/cube, 4 links/cube DDR3-1600 10-10-10, 11pJ/b, 2.6W standby/cube, 1W/link
HMC核心	16 1-issue in-order cores/cube, 1GHz, 32KB L1/core (60mW dynamic + 20mW static)/core
RRAM	16GB, 8 ranks, 8 4Gb-die/rank, 256×256 cells/array
	DDR3-1066, 7-7-7, R:0.293pJ/b, W:0.865pJ/b, L:92.9mW/die

[0061] 通过使用我们的功率管理技术，基于HMC和基于RAM的PIMs都能够获得进步。表2展示了架构的参数，其中还列出了时间和功率参数。标注着“Memory”和“HMC核心”的行代表着基于HMC的系统，标注着“RRAM”的行代表基于RRAM的PIM系统。基线系统仅使用“内存”行，不包含附加的“HMC核心”。

[0062] 我们对综合评估采用了不同的基准点。我们使用和以往工作相似的基准点。为了评估基于HMC的PIM设计，我们选择了和大数据分析以及图计算领域相似的基准点。它们包含了矩阵加法(MA)，树搜索(TS)，数组游走(AW)，平均青年追随者(TF)，网页排序(PR)，以及贝尔曼-福特算法(TF)。为了评估基于RRAM的设计，我们选取了几个通用应用以及几个神经网络设计。从Axbench选取的通用基准点包含了金融分析(blackscholes)，动画(inversek2j)，3D游戏(jmeint)，图像压缩(jpeg)，以及图像边缘检测(sobel)。神经网络基准点包含了使用MNIST数据集的两个CNN设计和三个MLP设计，以及以ImageNet出名的VGG-D。PIM内核保持着和这些以往工作相似的配置。

[0063] 将PAST、PUB和PS结合起来会产生一个更能源有效的系统。正如图6所示。通过冲刺

得到的额外的4W或8W功率通过使用PAST和PUB提供给HMC PIM设计。“None”表示针对没有功率供应容量保证的原始PIM系统的加速比。接下来的几栏表示使用不同功率管理配置达到的加速比:例如,“10+PS4”表示以10W作为基础功率上限以及具有额外的4W功率冲刺容量。结果显示即使当功率上限很低会损失性能,可以通过应用功率冲刺实现性能的提升。平均上,10W基础功率和8W的PS可以达到4.09倍的加速比(比原始3.78倍的加速比要高)。使用20W的功率上限和8W的冲刺功率可以使性能获得进一步的提升。总而言之,如果能合理配置我们的功率管理技术,现有的PIM设计可以进一步提升性能。

[0064] 最后需要注意的是,公布实施方式的目的在于帮助进一步理解本发明,但是本领域的技术人员可以理解:在不脱离本发明及所附的权利要求的精神和范围内,各种替换和修改都是可能的。因此,本发明不应局限于实施例所公开的内容,本发明要求保护的范围以权利要求书界定的范围为准。

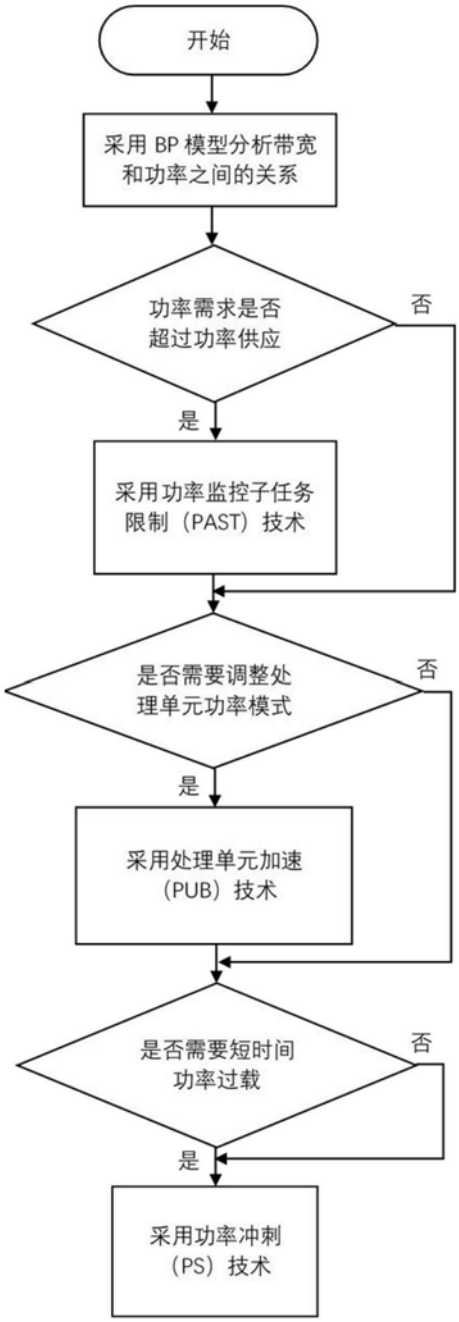

开始
采用 BP 模型分析带宽和功率之间的关系
功率需求是否超过功率供应
否
是
采用功率监控子任务限制（PAST）技术
是否需要调整处理单元功率模式
否
是
采用处理单元加速（PUB）技术
是否需要短时间功率过载
否
是
采用功率冲刺（PS）技术

图1

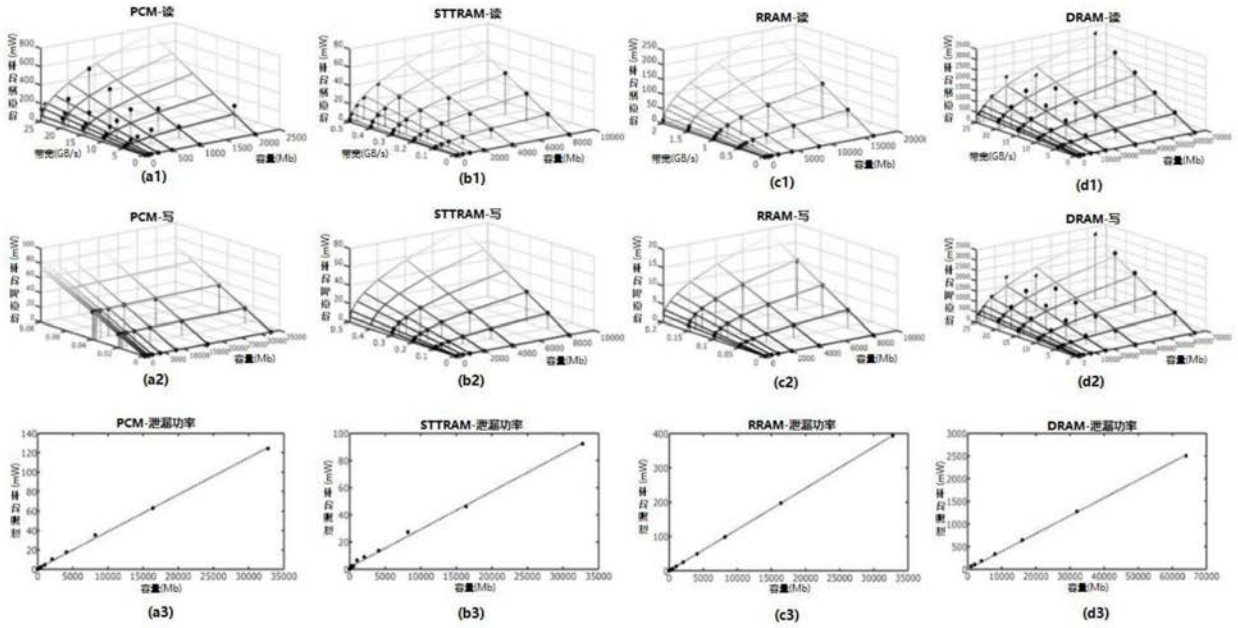

PCM-读
STTRAM-读
RRAM-读
DRAM-读
(a1)
(b1)
(c1)
(d1)
PCM-写
STTRAM-写
RRAM-写
DRAM-写
(a2)
(b2)
(c2)
(d2)
PCM-泄漏功率
STTRAM-泄漏功率
RRAM-泄漏功率
DRAM-泄漏功率
(a3)
(b3)
(c3)
(d3)

图2

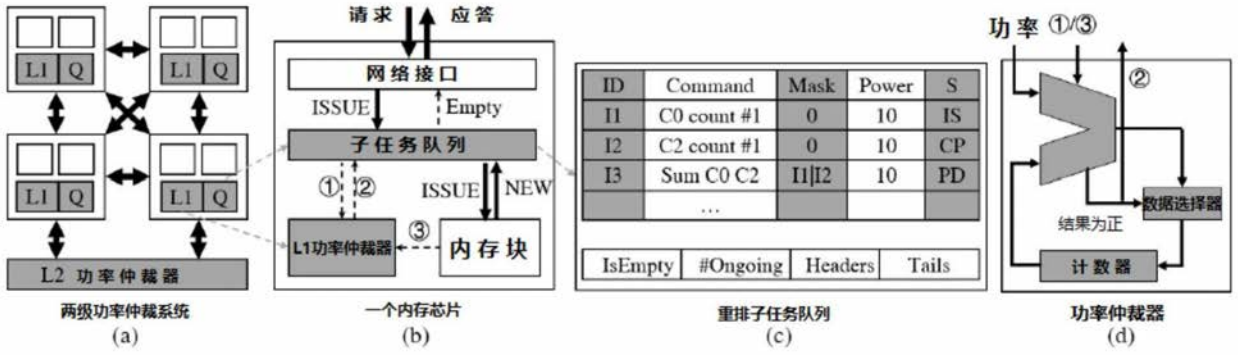

L1
Q
L2 功率仲裁器
两级功率仲裁系统
(a)
请求
应答
网络接口
ISSUE
Empty
子任务队列
①
②
ISSUE
NEW
L1功率仲裁器
③
内存块
一个内存芯片
(b)
ID
Command
Mask
Power
S
I1
C0 count #1
0
10
IS
I2
C2 count #1
0
10
CP
I3
Sum C0 C2
I1|I2
10
PD
IsEmpty
#Ongoing
Headers
Tails
重排子任务队列
(c)
功率 ①/③
②
数据选择器
结果为正
计数器
功率仲裁器
(d)

图3

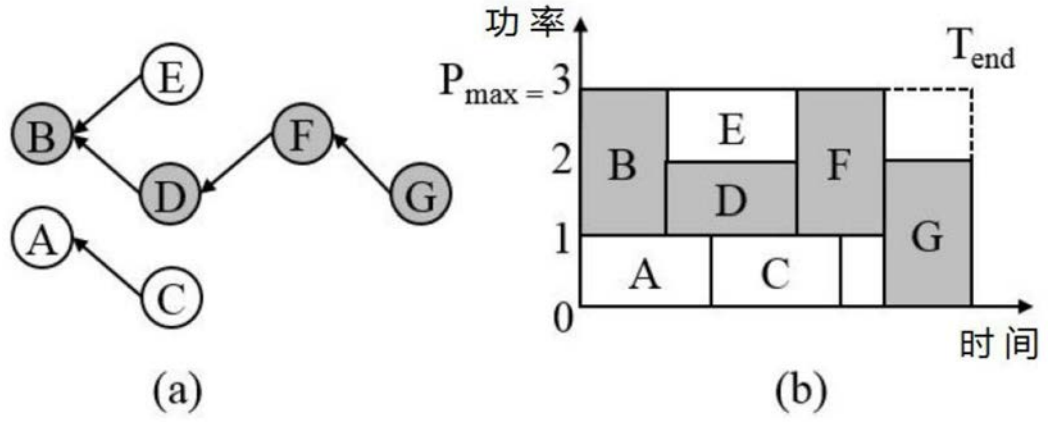

E
B
F
D
G
A
C
(a)
功率
T_{end}
$P_{max}=3$
2
1
0
B
E
F
D
G
A
C
时间
(b)

图4

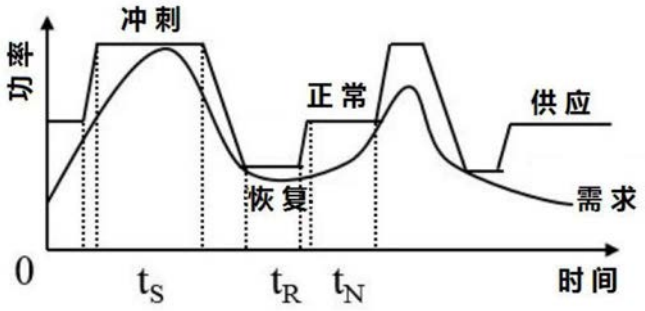

冲刺
功率
正常
供应
恢复
需求
0
t_S
t_R
t_N
时间

图5

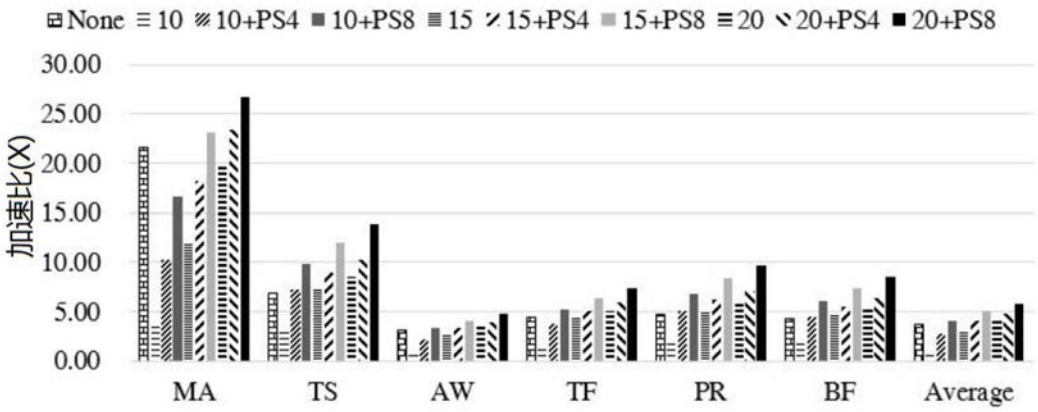

None
10
10+PS4
10+PS8
15
15+PS4
15+PS8
20
20+PS4
20+PS8
30.00
25.00
20.00
15.00
10.00
5.00
0.00
加速比(X)
MA
TS
AW
TF
PR
BF
Average

图6