面向任务信息的层次化存储 DPM 策略

黄少珉[1,2]，戚隆宁[1]，杨　军[1]，胡　晨[1]
（1. 东南大学国家专用集成电路系统工程技术研究中心，南京　210096；2. 东南大学集成电路学院，南京　210096）

[摘要]　面向层次化存储的 DPM 策略利用数据缓冲区实现请求集中从而延长 PMC 的空闲时间，可获得比传统策略更好的节能效果。在此基础上提出基于任务信息（task information based，TIB）的层次化存储 DPM 策略。TIB 策略对任务的数据访问模式进一步细分，通过修改存储访问接口获知任务数据访问的模式，根据不同模式决定预读与替换算法，使策略具有更好的节能意识。

[关键词]　数据缓冲区；任务信息；DPM；预读策略
[中图分类号]　TP316　[文献标识码]　A　[文章编号]　1009－1742（2010）02－0083－07

1　前言

嵌入式系统功能与结构的日益复杂使得系统的功耗不断增长。过去低功耗研究工作将重点放在降低处理器功耗上。在面向影像、视频等的应用中，能耗的主要来源是对存储器的频繁访问[1,2]。文献[3]的研究表明，在具有硬盘的计算机系统中，硬盘消耗了超过 20 % 的总能量。闪存往往被认为是功耗较低的非易失存储器件，然而对其进行写操作的功耗要远高于读操作的功耗[4]。读写功耗的差异性使得在写操作所占比例较高的应用场合，闪存功耗并不能被忽视。因此，嵌入式系统中存储系统功耗在总功耗中所占比例较大，降低存储系统功耗对于降低系统整体功耗至关重要。

动态功耗管理（dynamic power management，DPM）是一种有效的系统级功耗优化技术，通过观察负载情况有选择性的将设备设置为低功耗模式以达到降低功耗的目的[5]。在面向存储系统的 DPM 研究中，利用存储系统层次结构中各层设备的差异，将较高功耗设备的服务请求向较低功耗设备转移，从而为较高功耗设备创造出更长的空闲时间，以使其有更多机会进入低功耗模式以节约能量，被称为面向层次化存储的 DPM。利用数据缓冲区与存储 Cache 将对硬盘等高功耗存储设备的服务请求集中，便是此类研究中的典型。文献[6]基于缓冲区动态分组（dynamic grouping of data buffer，DGDB）的策略考虑了多任务环境下的数据访问特性，解决了之前基于缓冲区的层次化存储 DPM 策略的固有问题。在此基础上，对系统任务的数据访问模式进行划分，并将分类信息引入策略，提出基于任务信息（task information based，TIB）的层次化存储 DPM 策略（以下简称 TIB 策略）。任务信息对缓冲区预读以及块替换算法产生影响，使 TIB 策略具有更好的节能效果。

2　TIB 策略的总体结构

2.1　TIB 与 DGDB

在 DGDB 策略中，任务只向策略模块发出请求而不提供任何信息，任务并不知道策略的存在。这是 DGDB 策略的一个优势，因为在实施该策略的系统中，应用程序无须为策略进行修改。与此同时，DGDB 策略在数据预读与缓冲区块替换上的处理也是相对粗糙的。由于无法了解不同任务数据访问的

[收稿日期]　2008－11－11；修回日期　2009－10－26
[基金项目]　国家自然科学基金资助项目（60676011）
[作者简介]　胡　晨（1967－），男，江苏南京市人，东南大学国家专用集成电器系统工程技术研究中心教授，研究方向为嵌入式系统；E－mail：hc@seu.edu.cn

模式,DGDB 模块对所有任务按照顺序访问的方式进行处理。软硬件可定制是嵌入式系统的重要特点之一。在嵌入式系统中对应用程序进行修改,通过自定义的接口函数可将任务信息引入 DPM 策略。TIB 策略将任务数据访问模式的信息引入,根据各任务的数据访问模式决定缓冲区的尺寸以及预读与替换算法。与 DGDB 策略相比,TIB 策略中的缓冲区尺寸与预读数据量根据任务定制,从而更加精确;替换算法对数据块未来的访问情况更为了解,具有更好的节能意识。

2.2 服务请求的数据访问模式

笔者将服务请求源的数据访问模式分为 3 种:a. 顺序访问;b. 循环访问;c. 随机访问。图 1 对 3 种模式的访问模型进行了说明。顺序访问指对数据按照存储地址增长的方向进行连续访问,访问从某一地址开始以一定速度连续进行,图 1(a)说明了该模型。顺序访问模型在嵌入式系统中的实例有视频播放、文件复制等。循环访问指对存储设备中某一段数据内容进行反复访问:每次访问仍是顺序进行,但访问至数据段尾部时回到首部重新开始下一次访问,图 1(b)说明了该模型。循环访问模型在嵌入式系统中的实例如音频文件的循环播放,对某数据段的循环读写等。随机访问指每次访问都从存储空间或某数据段随机选择一个地址开始,且连续访问的长度也为随机量。图 1(c)说明了随机访问模型:阴影部分表示 4 次随机访问所覆盖的连续地址范围,$T_1 \sim T_4$ 为 4 次访问的时间段,每次访问的起始地址与长度为随机的,并满足一定分布,各随机访问的区域可以存在交叠,未在图中表示。随机访问在嵌入式系统中的实例有图片浏览、资源动态加载以及语音识别[7]等。

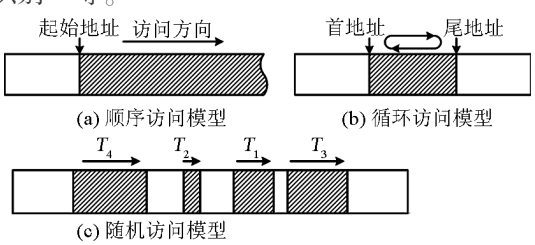

图 1 服务请求的数据访问模式

Fig. 1 Data access mode of service request

2.3 TIB 策略的结构

图 2 说明了 TIB 策略的结构,TIB 模块为其核心模块。运行时的多任务通过数据访问接口向 TIB 模块发出数据请求,并向其传递任务信息;TIB 模块根据任务信息对数据进行预读与替换;PM 模块为功耗管理模块,接收 TIB 模块的数据请求命令并转发至设备(图 2 中用 D 表示的模块),并根据设备空闲情况向设备发出功耗模式转换的命令。任务在启动时按照图 3 所示的任务信息数据结构向 TIB 模块注册信息。TIB 模块根据此信息为任务分配缓冲区,决定预读与替换算法。在任务运行时,若该信息发生变化,任务也可向 TIB 模块传递更新后的信息。任务运行结束时通知 TIB 模块,TIB 模块随即收回为该任务分配的缓冲区资源。在任务信息数据结构中,当 TYPE 为循环访问类型时,BEGIN_ADDR 与 END_ADDR 才有效。TIB 策略支持具有多种低功耗模式的设备,当每次预读结束时,PM 使数据存储设备进入策略所指定的低功耗模式。

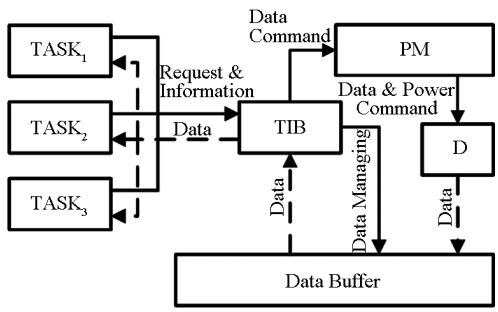

图 2 TIB 策略框架

Fig. 2 Framework of TIB

```
STRUCT  TASK_INFO
{
    TYPE;          // 任务数据访问模式所属类型
    RATE;          // 任务数据访问的速度
    BEGIN_ADDR;    // 循环访问的起始地址
    END_ADDR;      // 循环访问的结束地址
}
```

图 3 任务住处的数据结构

Fig. 3 Data structure of task information

3 缓冲区设计与预读策略

TIB 策略的数据缓冲区采用分块结构,缓冲区的块大小固定(设为 C_b)。TIB 策略根据不同任务数据访问模式的类型决定为其分配的缓冲区块数,并在数据缺失时从缺失处地址开始对数据进行预读。缓冲区的总块数为各任务缓冲区块数之和。

当任务 T_i 为顺序访问时,缓冲区尺寸 Q_i 的计

算方法与 DGDB 策略相同[6]，所需缓冲区块数为

$$M_i = \left\lceil \frac{\sqrt{\frac{e_m R_i \beta_i}{P_S}\left(1 - \frac{R_i \beta_i}{\alpha}\right)}}{C_b} \right\rceil + 1 \quad (1)$$

式(1)中[6]：R_i 为 T_i 运行时间与所有任务运行总时间的比率；β_i 为 T_i 单独运行时消耗数据的速度；e_m 为进入低功耗模式 S_m 的转换能耗；P_S 为缓冲区单位尺寸的静态功耗；α 为从设备读取数据至缓冲区的速度。当 T_i 为循环访问模式时，循环访问数据段中的数据在相当长的一段时间内将被任务反复访问，因此将数据段的内容全部预读进缓冲区可大幅减少对存储设备的访问。此时所需缓冲区块数为

$$M_i = \left\lceil \frac{A_E - A_B}{C_b} \right\rceil + 1 \quad (2)$$

式(2)中：A_B 为循环访问数据段的起始地址；A_E 为循环访问数据段的结束地址。T_i 为随机访问模式时，请求并非如同顺序访问模式一样连续产生，每次请求的数据量 L 为离散型随机变量。此时，缓冲区尺寸的确定应从概率统计的角度出发。随机访问模式下的缓冲区尺寸应使每次数据请求过程中缓冲区与设备的总能耗期望最低。设缓冲区尺寸为 Q，即每次预读的数量为 Q，则每次请求所产生的预读次数为

$$N_{PR} = \begin{cases} \frac{L}{Q}, \frac{L}{Q} = \left\lceil \frac{L}{Q} \right\rceil \\ \left\lceil \frac{L}{Q} \right\rceil + 1, \frac{L}{Q} \neq \left\lceil \frac{L}{Q} \right\rceil \end{cases} \quad (3)$$

设预读结束后使设备进入模式 S_j，则每次请求过程中设备的能耗为[6]

$$E_D = N_{PR} \frac{Q}{\alpha} P_R + N_{PR} e_j + \left(\frac{L}{R_i \beta_i} - N_{PR} \frac{Q}{\alpha} \right) P_j \quad (4)$$

式(4)中：P_R 为从设备读取数据时设备的平均功耗；e_j 为进入低功耗模式 S_j 的转换能耗；P_j 为进入低功耗模式 S_j 后设备的平均功耗。式(4)中，设备的能耗由三部分组成：第一项为读取数据所耗费的总能量；第二项为进入低功耗模式 S_j 所耗费的总转换能量；第三项为空闲时所耗费的能量。每次请求过程中缓冲区的能耗为[6]

$$E_{\text{Buffer}} = P_S Q \frac{L}{R_i \beta_i} + E_{\text{Read}} L + E_{\text{Write}} N_{PR} Q \quad (5)$$

式(5)中：E_{Read} 为从缓冲区读取单位数据量的能耗；E_{Write} 为向缓冲区写入单位数据量的能耗。式(5)中，第一项为缓冲区静态能耗；第二项为任务读取数据的能耗；第三项为从设备预取数据后向缓冲区写入的能耗。每次请求的总能耗为式(4)与式(5)之和，若将 N_{PR} 视为 L 的函数 $N_{PR}(L)$，则有总能耗为

$$\begin{aligned} E_{\text{Total}} &= E_D + E_{\text{Buffer}} = N_{PR}(L) \frac{Q}{\alpha} P_R + N_{PR}(L) e_j + \left(\frac{L}{R_i \beta_i} - N_{PR}(L) \frac{Q}{\alpha} \right) P_j \\ &+ P_S Q \frac{L}{R_i \beta_i} + E_{\text{Read}} L + E_{\text{Write}} N_{PR}(L) Q \\ &= N_{PR}(L) \left(\frac{Q}{\alpha} P_R + e_j + E_{\text{Write}} Q - \frac{Q}{\alpha} P_j \right) + L \left(\frac{P_j + P_S Q}{R_i \beta_i} + E_{\text{Read}} \right) \end{aligned} \quad (6)$$

由 2.2 中随机访问模型，并根据随机变量函数的期望计算公式[8]得总能耗的期望为

$$\begin{aligned} &E(E_{\text{Total}}) = \\ &\left(\frac{Q}{\alpha} P_R + e_j + E_{\text{Write}} Q - \frac{Q}{\alpha} P_j \right) E(N_{PR}(L)) + \left(\frac{P_j + P_S Q}{R_i \beta_i} + E_{\text{Read}} \right) E(L) \\ &= \left(\frac{Q}{\alpha} P_R + e_j + E_{\text{Write}} Q - \frac{Q}{\alpha} P_j \right) \sum_{k=1}^{L_{\max}} [N_{PR}(L_k) q_k] + \left(\frac{P_j + P_S Q}{R_i \beta_i} + E_{\text{Read}} \right) \sum_{k=1}^{L_{\max}} (L_k q_k) \end{aligned} \quad (7)$$

此时以 Q 为变量，观察 Q 对总能耗期望的影响。由式(3)可知：

$$\begin{aligned} &\lim_{Q \to +\infty} N_{PR} = 1 \\ &\lim_{Q \to 0} N_{PR} = +\infty \end{aligned} \quad (8)$$

则当 Q 趋向于正无穷时有：

$$\begin{aligned} &\lim_{Q \to +\infty} E(E_{\text{Total}}) = \\ &\lim_{Q \to +\infty} \left(\frac{Q}{\alpha} P_R + e_j + E_{\text{Write}} Q - \frac{Q}{\alpha} P_j \right) \sum_{k=1}^{L_{\max}} [N_{PR}(L_k) q_k] \\ &+ \lim_{Q \to +\infty} \left(\frac{P_j + P_S Q}{R_i \beta_i} + E_{\text{Read}} \right) \sum_{k=1}^{L_{\max}} (L_k q_k) \\ &= \sum_{k=1}^{L_{\max}} q_k \lim_{Q \to +\infty} \left(\frac{Q}{\alpha} P_R + e_j + E_{\text{Write}} Q - \frac{Q}{\alpha} P_j \right) + \lim_{Q \to +\infty} \left(\frac{P_j + P_S Q}{R_i \beta_i} + E_{\text{Read}} \right) \sum_{k=1}^{L_{\max}} (L_k q_k) \\ &= (+\infty) + (+\infty) = +\infty \end{aligned} \quad (9)$$

当 Q 趋向于 0 时有：

$$\begin{aligned} &\lim_{Q \to 0} E(E_{\text{Total}}) = \\ &\lim_{Q \to 0} \left(\frac{Q}{\alpha} P_R + e_j + E_{\text{Write}} Q - \frac{Q}{\alpha} P_j \right) \sum_{k=1}^{L_{\max}} [N_{PR}(L_k) q_k] \end{aligned}$$

$$+\lim_{Q\to 0}\left\{\frac{P_j+P_S Q}{R_i\beta_i}+E_{\mathrm{Read}}\right\}\sum_{k=1}^{L_{\max}}(L_k q_k)$$

$$=e_j\lim_{Q\to 0}\sum_{k=1}^{L_{\max}}[N_{PR}(L_k)q_k]+$$

$$\left\{\frac{P_j}{R_i\beta_i}+E_{\mathrm{Read}}\right\}\sum_{k=1}^{L_{\max}}(L_k q_k)=+\infty \tag{10}$$

则总能量期望在 Q 值的$(0,+\infty)$上可获得最小值。在 TIB 策略中,缓冲区块大小固定为 C_b,则 Q 只可能为 C_b 的整数倍。至此,在 TIB 策略中,考虑具有多种低功耗模式的 PMC,任务随机访问模式下缓冲区块数的确定方法为:设允许最大缓冲区块数为 $M_{\max}$,令 $Q=M\times C_b(1\leqslant M\leqslant M_{\max})$,并将 Q 代入式(7),可得在预读后 PMC 进入低功耗模式 S_j 时使式(7)取得最小值的 M 值,设这些 M 值分别为 $M_j(1\leqslant j\leqslant N)$,所对应的最低能耗期望为 E_j,则任务 T_i 的随机访问模式下最优化缓冲区块数为

$$M_i=M_m,$$
$$(1\leqslant M_m\leqslant M_{\max},E_m=\min(E_j,1\leqslant j\leqslant N)) \tag{11}$$

请求数据量 L 的随机分布可通过直方图法[9]进行在线统计,样本值通过滑动窗口获得以避免过多历史数据对统计的影响,使策略具有自适应 L 非稳态分布的能力。

4 替换算法

现有替换算法大都为提高性能而设计,多依赖历史的块访问信息决定替换规则。最近最久未使用算法(least recently used,LRU)以及最不经常使用算法(least frequently used,LFU)为两种常用且经典的替换算法,许多替换算法都是基于这两种算法的变形[10]。LRU 算法在发生缺失时将最近一段时间内未使用时间最长的块替换掉,算法依据的是访问局部性原理。LRU 算法仅根据最后访问的时间来决定从缓冲区中替换的块,而无法区分频繁访问的数据块与较少访问的数据块。例如,某一数据块已被访问了 10 次,而访问时间上在其之后的另一个块只被访问了 1 次。按照 LRU,前一个数据块会被优先替换,而无视该块比只存在一次访问的另一块更受欢迎这一事实。LFU 算法可以弥补这一不足。LFU 选择近期访问次数最少的块优先替换,有利于数据的总体优化使用。

在文章所讨论的数据访问环境下,LRU 与 LFU 的问题在于无法预知将来数据访问的信息,造成将要被访问的数据块被换出的情况发生,使 PMC 不得不再次工作,减小了空闲时间的同时也增加了能耗,图 4 说明了这种情况。在访问序列中,实线框内的访问由顺序访问模式的任务(任务 A)产生,虚线框内的由随机访问模式的任务(任务 B)产生。访问序列与缓冲区中的标号用以表示存储设备中的数据块,N 表示该块中无数据。首先任务 A 将数据块 1~4读入缓冲区,接下来任务 B 将数据块22~24 读入缓冲区,接下来的 3 次访问 PMC 保持空闲。在第 6 次访问中,数据块 25 不在缓冲区中,任务 B 预读数据块 25~27,LRU 与 LFU 替换算法此时都首先选择未访问过的数据块 3,4 进行替换。由于任务 A 为顺序访问模式,这两个数据块将来必将被访问,因此在第 7 次访问中,任务 A 不得不将数据块 3~6 读入缓冲区,此时 LRU 或 LFU 替换算法又将尚未使用的数据块 26,27 替换。可见,由于缺乏对未来数据访问信息的了解,LRU 与 LFU 替换算法造成了数据的重复读取,浪费了设备的能量。

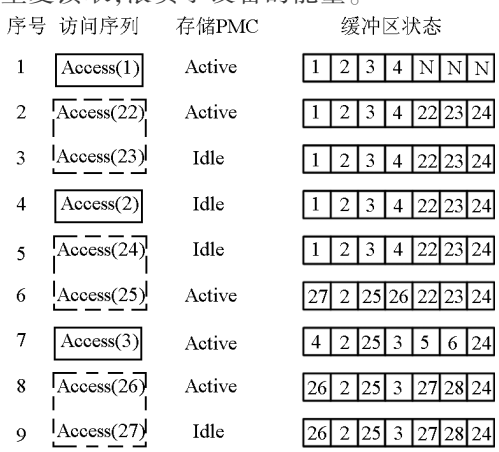

图 4 多任务混合数据访问下 LRU 与 LFU 替换算法对设备空闲时间的影响

Fig. 4 The impact caused by LRU and LFU onidle time of devices under multi - task mode

DGDB 策略的替换算法中,已访问的(BLK_USED)缓冲区块的替换优先级高于未访问的(BLK_UNUSE)缓冲区块,替换算法将首先选择 BLK_USED 的缓冲区块进行替换。对缓冲区块按照是否已访问进行划分实际上是一种对未来访问情况进行预测的思想,因此 DGDB 的替换算法从一定程度上克服了 LRU 与 LFU 算法的缺陷。然而,DGDB 的替换算法仍有通过任务信息进一步优化的必要,图 5 对此进行了说明。在访问序列中,实线框内的访问

由循环访问模式的任务(任务 A)产生,虚线框内的由随机访问模式的任务(任务 B)产生。任务 A 的循环访问数据段包括数据块 1 ~4。首先任务 A 将数据块 1 ~4 读入缓冲区,接着任务 B 将数据块 22 ~24读入缓冲区,之后的 4 次访问设备保持空闲。在第 8 次访问中发生缺失,DGDB 根据其替换算法选择已访问的数据块 1 ~3 进行替换。由于任务 A 为循环访问模式,其后仍然需要访问数据块 1 ~4,在第 9 次访问中不得不重新读入所需数据。究其原因,DGDB 策略的替换算法并不了解任务的访问模式,因此无法根据未来的数据访问做出有利于节能的替换。

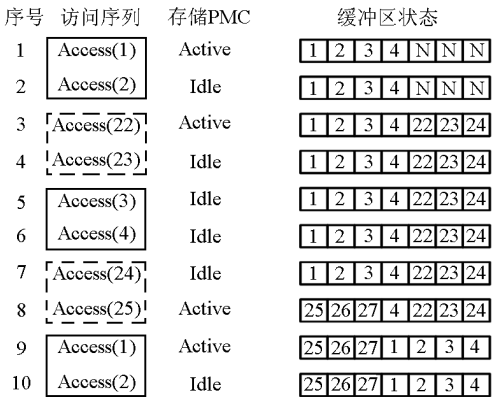

图 5　多任务混合数据访问下 DGDB 策略替换算法对设备空闲时间的影响

Fig. 5　The impact caused by replacement algorithm of DGDB on idle time of devices under multi – task mode

TIB 策略引入数据访问模式的信息正是为了避免以上现象造成的能耗损失。TIB 策略替换算法的思想为根据任务数据访问模式的不同采用不同的替换策略:a. 当任务为顺序访问模式时,预读的数据段中已访问的数据块将来不会再被访问,因此将数据块分为 USED 与 UNUSED 两种状态,前者表示数据块已被访问,后者表示尚未访问;b. 当任务为循环访问模式时,预读的数据段中所有的块都将被多次访问,将这些数据块设为 LOOPUSE 状态;c. 当任务为随机访问模式时,将预读的数据段中所有的块设为 RANDOMUSE 状态。TIB 模块根据任务所传递的访问模式信息,按照上述规则对缓冲区块的状态进行设置,在所需数据缺失时,按照表 1 的规则进行替换。USED 状态的缓冲区块将来不会再被访问,因此优先替换;UNUSED 与 LOOPUSE 状态的缓冲区块将来必被访问,因此不可替换;RANDOMUSE 状态的缓冲区块属于随机访问模式的任务,LFU 替换算法优先替换访问次数较少的块,实际上是按照块访问的概率进行排序,访问概率越高的块越不可能被替换,因此将 LFU 替换算法应用于随机访问模式能使数据块以最大的概率被重用。TIB 策略的替换算法根据任务的访问模式对数据块进行替换,能最大程度减小对设备的访问请求,从而延长空闲时间并减小访问能耗,克服了 LRU,LFU 以及 DGDB 策略替换算法的缺陷。

表 1　TIB 替换算法的规则

Table 1　Rules of replacement algorithm of TIB

缓冲区块状态	替换规则
USED	优先替换
UNUSED	不可替换
LOOPUSE	不可替换
RANDOMUSE	LFU 替换

5　仿真实验

为了对策略的效果进行评估比较,笔者以微硬盘及 SDRAM 缓冲区为例,对 TIB 策略与 DGDB 策略进行基于事件的仿真。在仿真实验中还实现了 TIB 策略的两种变形,即分别用 LRU 与 LFU 替换算法代替 TIB 策略中基于任务信息的替换算法,以比较 TIB 策略替换算法与 LRU 及 LFU 的差异。仿真采用平均功耗(average power,AP)作为指标,AP 为微硬盘与缓冲区的总平均功耗。

实验以 HITACHI Travelstar C4K60 系列微硬盘[11]为例。该微硬盘共有 3 种模式,Active Idle,Unload 与 Standby 模式,分别标记为 S_0,S_1,S_2 模式,其中 S_0 为正常工作模式,S_1,S_2 为低功耗模式,模式相关参数如表 2 所示。SDRAM 缓冲区的相关功耗参数则采用文献[12]中的实验值,硬盘数据读写与 SDRAM 数据操作的相关功耗参数如表 3 所示。评估所用策略如表 4 所示。多任务环境采用多请求源模型描述,设置三个不同的任务,如表 5 所示。顺序访问与循环访问模式的任务服务请求模型较为简单,通过人工合成的方式生成;而随机访问模型较为复杂,因此随机访问序列采自东南大学国家专用集成电路系统工程技术研究中心开发的 PMP(portable media player)系统平台中对微硬盘的请求。各任务运行时间与总时间的比例 R_i 即表征了各种访问模式在总访问序列中所占比例。表 6 说明了用于仿真的 10 个访问序列构成,总运行时间都为 3 600 s。

表 2　微硬盘功耗模式参数

Table 2　Parameters of all the power modes of micro harddisk

模式	平均功耗/W	转换能耗/J	损益平衡时间(T_{BE})/s
S0	0.511 8	—	—
S1	0.352 6	0.485 3	3.05
S2	0.079 6	2.006 1	5.57

表 3　数据操作的功耗参数

Table 3　Power parameters of data operations

微硬盘		SDRAM 缓冲区	
参数名称	参数值	参数名称	参数值
数据读取速度	4.508 MB/s	静态功耗	0.012 W/MB
数据写入速度	4.322 MB/s	读数据耗费的能量	0.003 J/MB
数据读取平均功耗	1.0 W	写数据耗费的能量	0.003 J/MB
数据写入平均功耗	1.1 W	—	—

表 4　仿真实验所用策略

Table 4　Policies for simulations

策略名称	策略描述
DGDB	5.2 节中的 DGDB 策略
TIB – LRU	采用 LRU 替换算法的 TIB 策略
TIB – LFU	采用 LFU 替换算法的 TIB 策略
TIB	TIB 策略

表 5　仿真所用任务设置

Table 5　Tasks settings for simulations

任务	访问速度/(MB·s^{-1})	访问模式
T1	0.53	顺序访问
T2	0.47	循环访问
T3	0.42	随机访问

表 6　仿真所用的各访问序列

Table 6　Data access series for simulations

序列编号	序列构成			序列编号	序列构成		
	R_1	R_2	R_3		R_1	R_2	R_3
1	0.80	0.10	0.10	6	0.20	0.60	0.20
2	0.70	0.15	0.15	7	0.10	0.10	0.80
3	0.60	0.20	0.20	8	0.15	0.15	0.70
4	0.10	0.80	0.10	9	0.20	0.20	0.60
5	0.15	0.70	0.15	10	0.30	0.35	0.35

将各策略 AP 值与无策略时微硬盘功耗的比值绘制成图 6 所示柱状图。在 10 组访问序列下，TIB 策略都获得了最低的平均功耗。访问序列 1 ~3 中，顺序访问所占比例较高，DGDB 策略与 TIB 策略具有相近的功耗(TIB 优于 DGDB 1 % ~3 %)。这是因为在顺序访问下，DGDB 的替换算法对未访问与已访问的数据块进行了区分，起到了与 TIB 策略替换算法近似的效果，而 LRU 与 LFU 替换算法的效果较差，与之前的分析一致。访问序列 4 ~6 中，循环访问所占比例较高，此时 TIB 策略的优势十分明显(功耗优于 DGDB 10 % ~17 %，与无策略时相比最大降低 46.9 %)。这是由于 TIB 策略根据任务信息将循环访问的数据段预读进缓冲区并保证其不被换出，而其他策略的替换算法都无法做到这一点。因此，当循环访问所占比例较高时，TIB 策略能大大减少对微硬盘的访问，节约了较多能量。访问序列 7 ~9 中，随机访问所占比例较大，TIB 策略针对随机访问模式预读与替换规则保证其继续保持最好的节能效果与性能(功耗优于 TIB – LFU 2 % ~4 %)。此时 TIB – LFU 策略的效果优于 DGDB，这点与之前分析较为一致：LFU 替换算法按照访问概率进行替换选择，在随机访问中具有其与生俱来的优势。在访问序列 10 中，各访问模式所占比例大体相当，此时 TIB 策略针对不同访问模式的预读与替换算法保证了其仍能获得最好的节能效果(功耗优于 DGDB 约 2 %)。

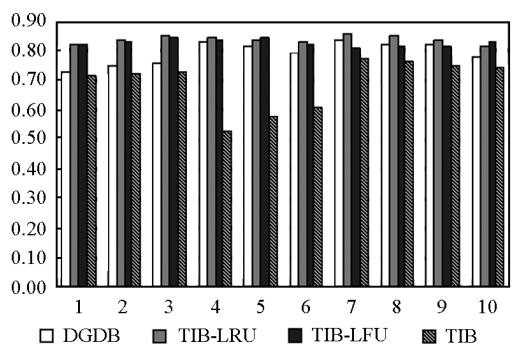

图 6　各策略 AP 值与无策略时微硬盘平均功耗的比值

Fig. 6　Ratio of AP value of each policy to average power of harddisk without policy

6　结语

在嵌入式多任务环境下，任务对存储系统部件的数据访问情况较为复杂。将服务请求源的数据访问模式分为 3 种：顺序访问、循环访问、随机访问。

在文献[6]中 DGDB 策略的基础上,将任务数据访问模式的信息引入缓冲区预读与替换算法,提出基于任务信息(task information based,TIB)的层次化存储 DPM 策略,使策略具有更好的节能意识。仿真结果表明,与 DGDB 策略相比,TIB 策略可进一步降低设备平均功耗,TIB 策略中的替换算法也比传统的 LRU 与 LFU 替换算法具有更好的节能效果

参考文献

[1] Catthoor F, Franssen F, Wuytack S, et al. Global communication and memory optimizing transformations for low power systems[A]. Proceedings of the International Workshop on Low Power Design[C]. La Jolla, CA, USA, 1994

[2] Catthoor F, Suytack S, Greef E, et al. Custom Memory Management Methodology: Exploration of Memory Organisation for Embedded Multimedia System Design[M]. Boston: Kluwer Academic Publishers, 1998

[3] Lu Y-H, De Micheli G. Adaptive hard disk power management on personal computers[A]. Proc. of the IEEE Great Lakes Symp. on VLSI[C]. Ann Arbor, MI, USA, 1999

[4] Chanik Park, Jeong-Uk Kang, Seon-Yeong Park, et al. Energy-aware demand paging on NAND flash-based embedded storages[A]. Proceedings of the International Symposium on Low Power Electronics and Design[C]. USA: New York, 2004

[5] Luca Benini; Alessandro Bogliolo; Giovanni De Micheli. A Survey of design techniques for system-Level dynamic power management[J]. IEEE Transactions on Very Large-Scale Integration Systems, 2000, 8(3): 299-316

[6] 黄少珉,张 哲,胡 晨,等. 基于数据缓冲区动态分组的功耗管理策略[J]. 电路与系统学报,2007, 12(1):26-32

[7] Athanasios E. Papathanasiou, Michael L. Scott. Energy efficient prefetching and caching[A]. Proceedings of the 2004 USENIX Annual Technical Conference[C]. Berkeley, CA, USA. 2004

[8] 盛 骤,谢式千,潘承毅. 概率论与数理统计[M]. 北京:高等教育出版社,1989

[9] Sandy Irani, Sandeep Shukla, Rajesh Gupta. Online strategies for dynamic power management in systems with multiple power-saving states[J]. ACM Transactions on Embedded Computing Systems, 2003, 2(3): 325-346

[10] Zhu Qingbo, Shankar Asim, Zhou Yuanyuan. PB-LRU: A self-tuning power aware storage cache replacement algorithm for conserving disk energy[A]. Proceedings of the 18th Annual International Conference on Supercomputing[C]. New York, NY, USA, 2004

[11] Hitachi. Hard Disk Drive Specifications, Travelstar C4K60[S]. 2003

[12] Cai Le, Yung-Hsiang Lu. Dynamic power management using data buffers[A]. Design, Automation and Test in Europe Conference and Exhibition[C]. Paris, France, 2004

Storage hierarchy oriented DPM policy based on task information

Huang Shaomin[1,2], Qi Longning[1], Yang Jun[1], Hu Chen[1]

(1. Nation ASIC System Engineering Research Center, Southeast University, Nanjing 210096, China;
2. School of Integrated Circuit, Southeast University, Nanjing 210096, China)

[**Abstract**] Storage hierarchy oriented DPM, which uses buffer to prolong idle time, can achieve lower power than traditional DPM policies. The paper proposes task information based (TIB) policy for storage hierarchy oriented DPM. TIB subdivides the data access mode of tasks and introduces them into policy by modifying access interface to make prefetching and replacement algorithm more energy aware.

[**Key words**] data buffer; task information; DPM; prefetching policy