第10卷 第31期 2010年11月
1671—1815(2010)31-7810-06

科 学 技 术 与 工 程
Science Technology and Engineering

Vol. 10 No. 31 Nov. 2010

SPARC V8 处理器中存储管理单元的设计

肖建青 李红桥 张洵颖 龚龙庆*

(西安微电子技术研究所,西安 710054)

摘 要 存储管理单元 MMU(Memory Management Unit)是各种微处理器用以实现虚拟存储的关键部件。针对 SPARC V8 结构处理器的需求,提出一种 MMU 的设计方案。分析了虚拟地址到物理地址的映射关系,通过采用转换后备缓冲区 TLB(Translation Lookaside Buffer)加快了 SPARC V8 处理器在多任务处理时虚实地址的转换;以页式存储为依据,为页面的读、写、执行等访问提供了保护机制;并在异常发生时根据访问类型进行相应的异常处理。结论表明,该设计方案是可行的、有效的;并且可作为其它处理器 MMU 设计的基础。

关键词 存储管理单元 虚拟地址 物理地址 转换后备缓冲区 多任务

中图法分类号 TP311.52; **文献标志码** A

在计算机的发展历史上,出现了多任务和多用户系统。这种多任务系统建立在 CPU 时钟周期多元化的基础上,所需完成的任务分别进驻在主存储器中,从而可以获得较高的资源利用[1]。为了满足多任务系统发展的要求,实现虚拟存储的功能,计算机系统中产生了存储管理单元 MMU(Memory Management Unit)。存储管理单元是处理器支持虚拟存储器的核心部件[2],它的主要作用是进行虚拟地址到物理地址的转换,并且提供多任务间存储器访问的保护机制。

基于 SPARC V8 结构,设计了一种用于该处理器的存储管理单元,并采用 SOC(System On Chip)技术,将其集成到处理器核中,以提高系统的性能和可靠性。

1 MMU 的总体结构

SPARC V8 处理器采用虚拟地址 cache 的存储结构[3,4],即在指令流水时,CPU 发出的虚拟地址可以直接访问 cache,而不是每次都要等到 MMU 将虚拟地址转换成物理地址以后才能进行,这样显著地减少了访问 cache 所花费的时间,大大提高了系统的访存速率,其结构如图 1 所示。

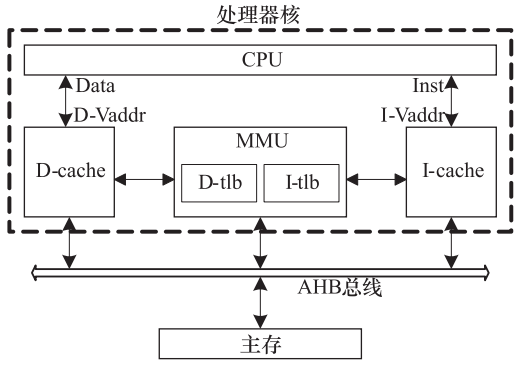

图 1 MMU 与处理器系统的结构图

可以看到,处理器核中集成了片内的 cache 和 MMU 等存储模块,片外则通过 AHB(Advanced High-Performance Bus)总线与主存相连。

该系统中 cache 的标记部分存储的是虚地址标记,当 CPU 访问 cache 时,将发出的虚拟地址与 cache 的虚地址标记进行比较,如果匹配,则访问命中,这时 CPU 直接从 cache 的存储器中获取所需的数据或指令,省去了 MMU 进行地址转换的时间。如果 cache 未命中,则就需要 MMU 将虚拟地址转换

2010 年 8 月 3 日收到

第一作者简介:肖建青(1985—),男,湖北人,硕士研究生,研究方向:嵌入式计算机应用。E-mail: xjq_career@126.com。

* 通信作者简介:龚龙庆(1962—),男,陕西人,硕士,研究员,硕士生导师,研究方向:空间计算机设计。

成物理地址,送回给 cache 的控制器后再由物理地址去访问主存,MMU 的工作原理如图 2 所示。

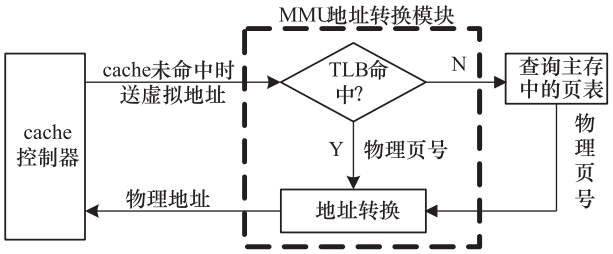

图 2　MMU 的工作原理

MMU 模块中主要包括 TLB(Translation Lookaside Buffer)表项及其相应的控制电路,其中 TLB 表项是一个类似于 cache 的存储模块,它保存着虚实地址的转换信息。对应于处理器中独立的I-cache和D-cache,其 MMU 也分别设置了 I-tlb 和 D-tlb,两者在结构和转换原理上完全一样,使得指令和数据的地址转换可以独立进行,这样提供了更大的转换带宽,进而提高了处理器的性能和处理速率。

另外,MMU 还提供了存储器保护功能,确保一个进程不能读/写其它进程的地址空间。如果当前访问违背了存储保护的原则,或者虚拟地址到物理地址的转换不成功,都会产生相应的异常信号去告知 CPU,并通过操作系统调用异常处理程序来进行处理。

2　MMU 的地址映射策略

SPARC V8 处理器的 MMU 把来自 CPU 的32 位虚拟地址转换成 36 位的物理地址,从而提供了 32 位总线到 64GB 物理空间的存储器映射。MMU 采用了页式存储管理,它把主存储器、虚拟存储器和磁盘存储器都分成固定大小(4K)的块,称为页。因此,虚拟地址和物理地址具有共同的页偏移量,只要找到了物理页号就可以形成物理地址。实际上,虚拟地址到物理地址的转换,就是从虚地址标志映射到物理页号的过程,它按照图 3 所示的五个步骤进行。

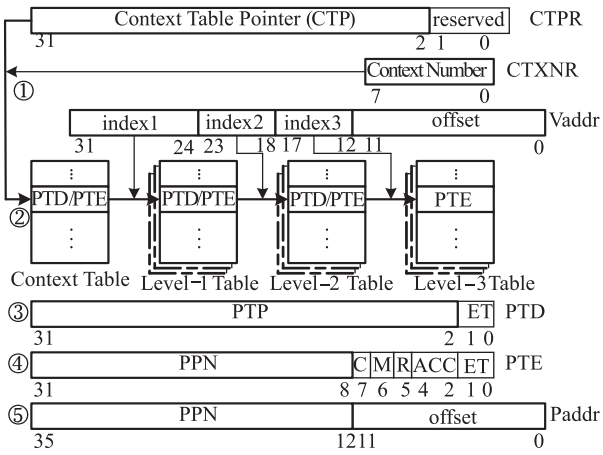

图 3　虚拟地址和物理地址的映射关系

主存中的三级页表用来存储全部的地址转换信息,而第一级页表的根指针则存储在上下文表中,通过页表中页表项的最低两位 ET 来判断是否要继续查找页表。可见,MMU 可以实现不同级数的页表访问,最少 1 级,最多 4 级[5]。页表中的存储项如果为 PTD(Page Table Descriptors),则它包含了指向下一级页表的指针(PTP);如果为 PTE(Page Table Entry),则它包含了物理地址的页号(PPN)。虚拟地址 Vaddr 中的三个虚拟地址标志 index1、index2 和 index3 分别为查找相应级别的页表提供了索引。

图 3 中,CTPR 和 CTXNR 分别是 MMU 的上下文表指针寄存器和上下文寄存器,CTP 是指向上下文表(Context Table)的首地址;Context Number 是上下文号,作为查上下文表的偏移量。在查找过程中,首先由 CTPR 和 CTXNR 的内容去查上下文表,得到三级页表的入口地址,判断该地址是 PTD 还是 PTE。如果是 PTE 则不需要再查页表,这样每页有 4GB 的寻址空间;否则继续查找页表。如果在 Level-1 Table 中找到 PTE,则每页有 16MB 的寻址空间;如果在 Level-2 Table 中找到 PTE,则每页有 256KB 的寻址空间;如果在 Level-3 Table 中找到 PTE,则每页有 4KB 的寻址空间;如果找不到 PTE 的话,则提示错误。找到 PTE 后,由 PTE 中的 PPN 加上虚拟地址 Vaddr 的 offset 偏移量,就得到了物理地址 Paddr。

3 TLB 的设计实现

从主存中查页表来形成物理地址的过程可见，每一个虚拟地址到物理地址的转换，至少要一次访存(在上下文表中就找到了 PTE)，最多要四次访存(在第三级页表中找到 PTE)，其地址的转换速率太慢。为了加快虚拟地址到物理地址的转换，MMU 中专门设有页描述符 cache，也叫 TLB(Translation Lookaside Buffer)，它里面高速缓存有地址转换所需的页表项信息。这样，每次地址转换时，只需要将虚拟地址和 TLB 中的虚地址标志进行比较，若匹配成功，就可以直接利用与虚标志相对应的页表项来形成物理地址，大大加快了地址转换的速率。TLB 的结构如图 4 所示。

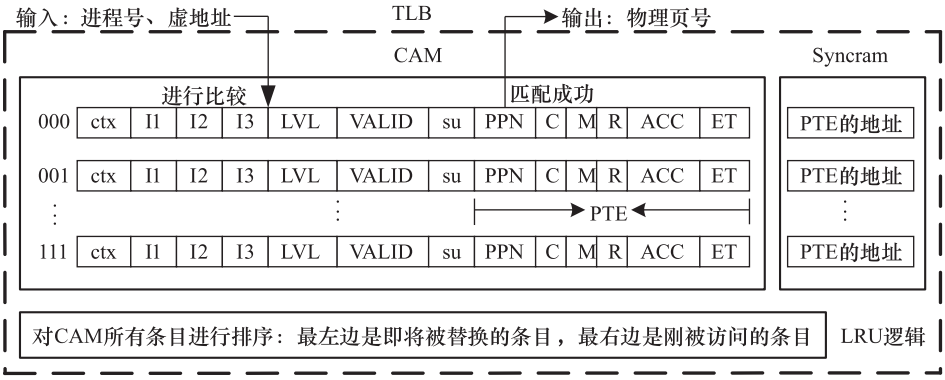

图 4 具有 8 个条目的 TLB 的结构图

TLB 实际上是专门用于地址转换的 cache，其结构包含相联存储器 CAM、TLB 条目的替换逻辑和写回主存的同步存储器 Syncram 三部分。MMU 的 TLB 最多可以配置 64 个全相联的 CAM 条目[6]，如果条目数太多，会影响相联比较的速度；如果条目数太少，又会降低 TLB 的命中率，所以选择合适的 CAM 条目数也非常关键。

3.1 TLB 的匹配逻辑

判断 TLB 是否命中是通过相联存储器 CAM(content-addressable memory)来实现的。CAM 是按内容寻址的存储器，它的基本原理是：把存储单元所存内容的某一部分作为检索项(即关键字项)，去检索该存储器，并将存储器中与该检索项符合的存储单元内容进行读出或写入。

在图 4 中，TLB 有 8 个 CAM 条目项，编号从“000”到“111”。每个条目项中存有进程号 ctx、虚地址的三个索引(I1、I2 和 I3)、页表级别 LVL、页表项的有效位 VALID、超级用户位 su 和对应的 PTE 信息。

在 TLB 进行虚实地址的转换时，将输入的用户进程号、虚地址同时与所有 CAM 条目中的进程标志、虚地址标志进行比较，并按照每个条目的 LVL 进行匹配。若匹配成功且 VALID 为 1，则 TLB 命中，此时将对应 CAM 条目中的 PPN 取出，与虚地址部分拼接来形成物理地址，拼接的规则如下。

(1) 当 LVL = PAGE 时，表示该条目的 PTE 来自第三级页表，此时需要 ctx、I1、I2、I3 都匹配。生成的物理地址由该条目的 PPN 与虚拟地址的页偏移量拼接而成，即 PA[35:0] = PPN[31:8] & VA[11:0]。

(2) 当 LVL = SEGMENT 时，表示该条目的 PTE 来自第二级页表，此时需要 ctx、I1、I2 都匹配。生成的物理地址由该条目中 PPN 的高 18 位与虚拟地址的 I3 和页偏移量拼接而成，即 PA[35:0] = PPN[31:14]&VA[17:0]。

(3) 当 LVL = REGION 时，表示该条目的 PTE 来自第一级页表，此时需要 ctx、I1 都匹配。生成的物理地址由该条目中 PPN 的高 12 位与虚拟地址的

I2、I3 和页偏移量拼接而成,即 PA[35:0] = PPN[31:20]&VA[23:0]。

(4) 当 LVL = CTX 时,表示该条目的 PTE 来自上下文表,此时只需要 ctx 匹配即可。生成的物理地址由该条目中 PPN 的高 4 位与整个 32 位的虚拟地址拼接而成,即 PA[35:0] = PPN[31:28]&VA[31:0]。

3.2　TLB 的替换策略

如果虚地址标志匹配不成功或者页表项的 VALID 为 0,则 TLB 未命中,表示本次地址转换所需的信息不在 TLB 中。这时需要到主存中查相应级别的页表,找到对应的 PTE 并装入 TLB,然而如果此时 TLB 中没有空余的条目,就必须将某些条目替换掉。这里用到的替换算法有 LRU 和随机法两种,一般选用 LRU 算法。

随机法是从编号 0 到编号 7(假设 TLB 中一共有 8 个条目)依次加 1 循环计数,把当前的随机数(用三位逻辑向量表示)作为被替换的条目地址。这种算法简单,易于实现,但没有利用条目使用情况的历史信息,反映不了程序的局部性,其命中率低,因此很少采用。

LRU(最近最少被使用)算法则是根据各条目的使用情况,将所有条目的编号按照从左到右的顺序排列,最左边表示最近最少被使用的,最右边表示刚被访问的,这样每次要进行替换时,就把最左边的条目替换掉。其过程如图 5 所示。

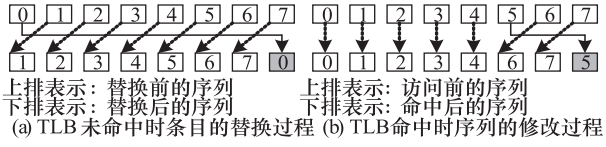

图 5　TLB 条目序列的更新示意图

图 5 中(a)表示 TLB 未命中时条目的替换过程。此时,最左边的 0 号条目被来自主存中的新条目替换掉,并将该新装入的条目放在最右边,而其它的条目依次向左移动。在新序列中,条目 1 移到了最左边,成了最近最少被访问的、下次最先被替换的条目。

如果 TLB 命中的话,也是要修改 LRU 序列的,因为访问命中了某个条目后,表示该条目最近刚被使用过,应该把它放在最右边,其过程如图 5 中(b)所示。由图可知,当访问命中条目 5 后将其移到了最右边,它左边的条目位置不变,而它右边的条目依次向左移动。

3.3　TLB 的写回

与 cache 的写回特性相似,MMU 中的指令 TLB 不能被 CPU 更改,所以在这里只需要讨论数据 TLB 被修改后所实现写回主存的操作。

如前所述,每次在 TLB 未命中时,都要去主存中查找所需的 PTE。查找结束后,将该 PTE 及相关的虚地址标志信息写到要被替换的那个 CAM 条目里;同时还将该 PTE 的地址写到 Syncram 里,如图 4 所示,所以 TLB 中的 Syncram 预先记录了所有 CAM 条目中 PTE 所对应的主存地址。

如果 CPU 修改了数据 TLB 中的某个条目,相应的 M 位会变为 1。在实现时,使用两个信号 NEEDSYNC 和 twNEEDSYNC 来分别表示在 TLB 命中和未命中的情况下,是否有某个条目的 PTE 被修改并需要写回到主存中。如果有,则将该 PTE 作为要写的数据、将 Syncram 中存的 PTE 地址作为访存地址通过 AHB 总线去更新主存,这样保证了 TLB 和主存页表中 PTE 信息的一致性。

3.4　快速 TLB 操作

以上设计思想是虚拟地址在访问 cache 未命中以后才去访问 MMU 中 TLB 的,其实还可以选择一种快速的 TLB 操作,即每次让虚拟地址同时访问 cache 标志和 TLB。这样在 cache 缺失时,该操作比原方法要节省 2 个时钟周期,进一步提高了系统的吞吐量,然而它需要在某种程度上增加硬件面积。所以,应针对处理器不同的性能需求,来综合考虑选择哪一种 TLB 操作。

4　存储保护与异常处理

4.1　存储保护

MMU 除了实现虚拟地址到物理地址的转换外,还可对页面的读、写、执行和用户权限做出相应的

保护。SPARC V8 处理器的存储保护功能[7]是通过上下文寄存器和 PTE 中设置标志位来完成的。

上下文寄存器里的上下文号(Context Number,即进程号)表明了当前的地址空间是属于该用户进程的,所以此时此刻其它进程都不能进行存储器的读/写操作。上下文号在任意两个进程间都建立起了保护墙,同时它也方便了进程间任务的切换。另外,在页表项 PTE 中设置了访问权限位 ACC 来表明当前的访问是超级用户还是一般用户、是读操作还是写操作、是指令空间还是数据空间,如表 1 所示。

表 1 访问权限的规则

ACC	一般用户访问 (ASI = 0x8 或 ASI = 0xA)	超级用户访问 (ASI = 0x9 或 ASI = 0xB)
0	只读	只读
1	读/写	读/写
2	读/执行	读/执行
3	读/写/执行	读/写/执行
4	只执行	只执行
5	只读	读/写
6	不能访问	读/执行
7	不能访问	读/写/执行

这里使用了地址空间标识符 ASI。其中,ASI = 0x8 和 ASI = 0xA 分别表示一般用户的指令空间和数据空间;ASI = 0x9 和 ASI = 0xB 分别表示超级用户的指令空间和数据空间。表中的“读/写”表示对数据的读写操作,“执行”表示指令的执行操作。从表可知,当 ACC = 6 或者 7 时只允许超级用户的访问,此时一般用户没有访问权限。

4.2 异常处理

如果当前的访问与相应的存储保护标志相匹配,那么转换出来的物理地址是有效的,可以用它去访问存储器;如果当前的访问违反了保护规则,那么将导致访问错误,转换出来的物理地址就无效,这时 MMU 会把错误类型和发生访问错误的地址分别记录在故障状态寄存器和故障地址寄存器中。

如图 6 所示,故障地址寄存器中存储的是 32 位的故障地址;故障状态寄存器中各字段所表示的含义说明如下:

图 6 故障状态/地址存储器

OW—从上次读故障状态寄存器以来,重复写入同一种故障的标志位;

FAV—故障地址的有效位;

FT—当前发生的故障类型;

AT—引起故障的访问类型;

L—引起故障的条目所在页表的级别;

EBE—外部总线错误;

reserved—保留位,其值必须为零。

当异常发生时,MMU 根据标志信息确定属于何种异常,然后把故障信息输出给 CPU,最终转到相应的异常处理程序进行处理[8]。在测试与验证中,我们总结了一些可能引起异常的情况,如表 2 所示,同时还列出了 MMU 对这些异常情况的处理方式。

5 结论和展望

在实现多用户和多任务技术的计算机系统中,通过 MMU 把虚拟地址转换成物理地址来支持虚拟存储的管理,是当今微处理器设计的普遍趋势。本文基于 SPARC V8 体系结构的处理器,提出了一种存储管理单元的设计方案。在处理器访问 cache 未命中的情况下,可通过 MMU 生成的物理地址去访问主存以获取其所需的数据或指令,同时采用了 TLB 技术为 MMU 的虚实地址转换做了加速处理;在不同任务之间进行任务切换时,为页表的访问提供了存储保护;如果当前的访问违背了所规定的保护机制,MMU 能够根据发生故障的类型进行相应的异常处理,提高了系统的容错性和灵活性。

表 2　MMU 的异常说明

引起异常的条件	异常描述	异常处理
TLB miss	TLB 中没有匹配的页表项	从主存中调入 MMU
Store = 1 且 PTE. C = 0	对 DTLB 条目进行存储,但它不可缓存	Store 操作无效
PTE. V = 0	匹配的页表项无效	无效地址错误:置 FT = 1
ACC = 1 且 isid = id_icache	当前试图访问受保护的指令空间	保护错误:置 FT = 2
ACC = 4 且 isid = id_dcache	当前试图访问受保护的数据空间	保护错误:置 FT = 2
ACC = 2 且 read = 0	当前试图进行受保护的写操作	保护错误:置 FT = 2
ACC = 6 或 7 且 su = 0	只允许超级用户操作,当前却是一般用户	特权违背错误:置 FT = 3
Level-3 Table 中存储的不是 PTE	查找主存中的页表失败	转换错误:置 FT = 4 从辅存中调入主存
timeout 或 parity error	不查页表访问存储器时的外部总线错误	访问总线错误:置 FT = 5
ahbi. hresp = error ∣ split	存储器的响应与 MMU 的请求不相符	存储器异常:置 FT = 6

此外,进一步的研究工作主要是通过量化分析为 TLB 中 CAM 条目的数量选择最佳值,以寻求高命中率、高地址转换速率和低功耗的平衡点,使得在满足处理器应用需求的前提下,达到优化处理器性能的目的。

参　考　文　献

1　夏　晓,陆明达. 32 位嵌入式微处理器中存储管理单元结构的研究与设计. 上海:同济大学硕士论文,2004

2　Patterson D A, Hennessy J L. Computer Architecture: A Quantitative Approach, Third Edition. San Francisco: Morgan Kaufman Publish, 2002

3　施　蕾,刘　波,周　凯. 基于 SPARC V8 结构处理器的计算机系统设计. 空间控制技术与应用,2008;34(3):46—50

4　胡越明. 计算机系统结构. 北京:北京航空航天大学出版社,2007;10

5　吴志雄,郑灵翔,周剑扬. SPARC V8 处理器断电调试的设计与实现研究. 电子技术应用,2008;34(8):38—44

6　Gaisler J, Catovic E, Isomaki M, *et*, *al*. GRLIB IP Core User's Manual, Version 1.0.20, February,2009

7　SPARC International Inc. The SPARC Architecture Manual, Version 8, 1992

8　屈文新,樊晓桠. "龙腾" R2 微处理器存储管理单元的设计与实现. 西北工业大学学报,2007;25(1):137—140

Design of Memory Management Unit in SPARC V8 Processor

XIAO Jian-qing, LI Hong-qiao, ZHANG Xun-ying, GONG Long-qing*

(Xi'an Microelectronics Technique Institute, Xi'an 710054, P. R. China)

[**Abstract**]　Memory management unit is an important component of microprocessor which implements virtual memory. To satisfy the need of processor based on SPARC V8 architecture, a design of MMU is proposed. The mapping from any virtual address to a physical address is analyzed, the address translation using TLB to realize Multitask in SPARC V8 processor is speeded up. Through paged virtual memory, this design provided a protection mechanism for page-level read, write and execution, and dealt with corresponding exceptional events according to the access type when some exception occurred. Conclusions show that the design is feasible and effective, and it can be used as a basis for the design of MMU in other processor.

[**Key words**]　MMU　virtual address　physical address　TLB　multitask