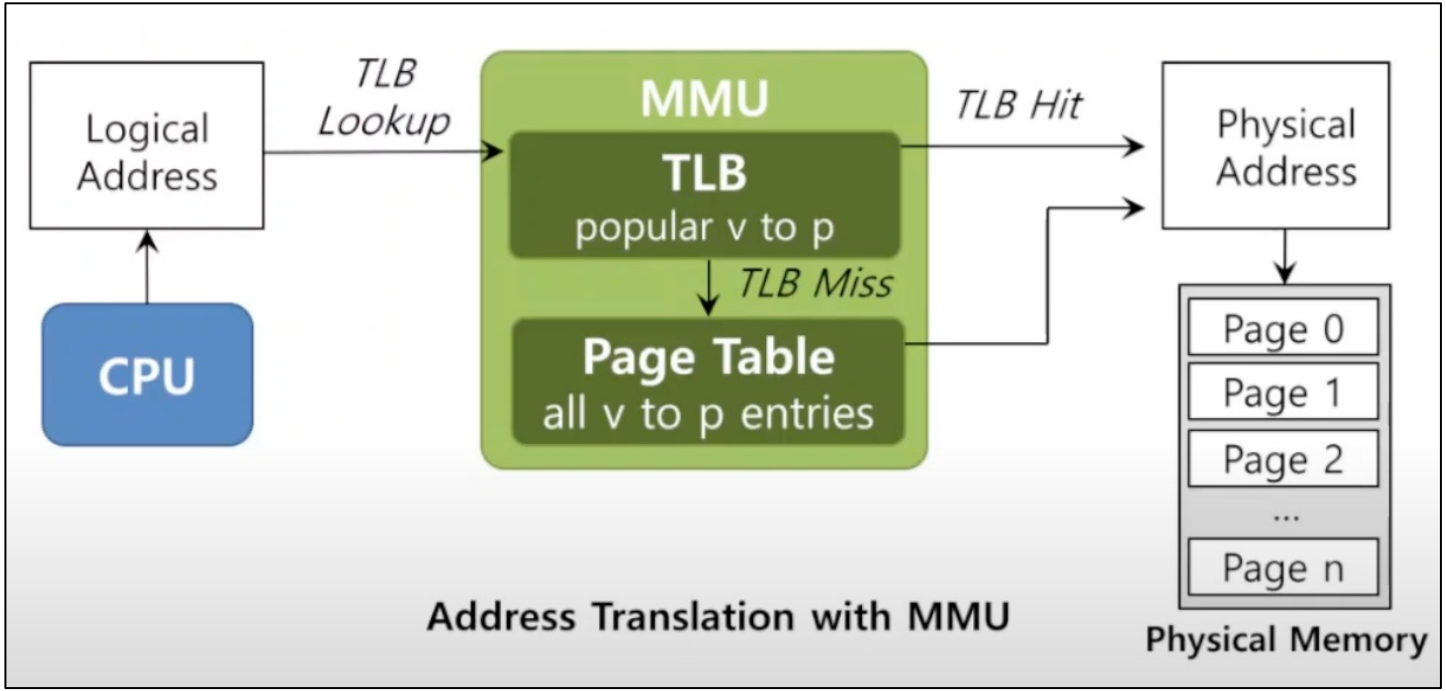
分页：快速地址转换

邰颖
南京大学
智能科学与技术学院

Translation-Lookaside Buffer，TLB

- 芯片内存管理单元（MMU）的一部分
- TLB本质上是一个硬件缓存，缓存常用或经常访问的内存区域，加速地址转换



TLB基本算法及其实现

```
1: VPN = (VirtualAddress & VPN_MASK ) >> SHIFT
2: (Success , TlbEntry) = TLB_Lookup(VPN)
3:    if(Success == True){ // TLB Hit
4:    if(CanAccess(TlbEntry.ProtectBit) == True ){
5:        offset = VirtualAddress & OFFSET_MASK
6:        PhysAddr = (TlbEntry.PFN << SHIFT) | Offset
7:        AccessMemory( PhysAddr )
8:    }else RaiseException(PROTECTION_ERROR)
```

- **第1行** 提取虚拟页面号(VPN)。
- **第2行** 查询TLB是否命中。
- **第5至8行** 从相关的TLB条目中提取页面帧号，并形成所需的物理地址，访问内存。

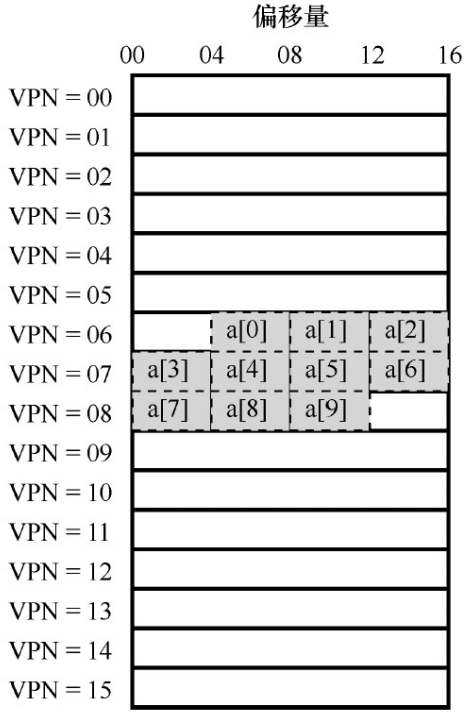
TLB基本算法及其实现（续）

```
11:     }else{ //TLB Miss
12:         PTEAddr = PTBR + (VPN * sizeof(PTE))
13:         PTE = AccessMemory(PTEAddr)
14:         (…)
15:
16:         TLB_Insert( VPN , PTE.PFN , PTE.ProtectBits)
17:         RetryInstruction()
18:     }
19:}
```

- **第11-12行：**硬件访问页表以查找虚拟地址到物理地址的映射。
- **第16行：**更新TLB，将映射信息插入TLB，并重新执行指令。

例子：访问数组 - TLB 如何提高其性能

- 假设：8位虚拟地址空间，16字节的页面大小
- 因此4位VPN和4位OFFSET，数组a从虚拟地址100（01100100）开始加载




```
0: int sum = 0;
1: for(i = 0; i < 10; i++) {
2:     sum += a[i];
3: }
```

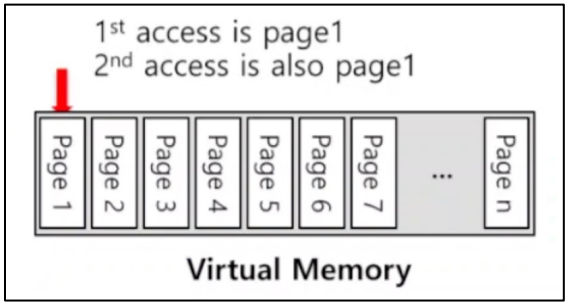
TLB因空间局部性提高了性能，命中情况：

miss, hit, hit,
miss, hit, hit, hit,
miss, hit, hit

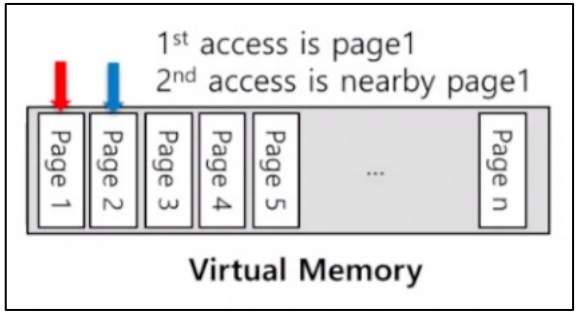
3次未命中和7次命中，TLB命中率为70%！

局部性 (Locality)

- **时间局部性 (Temporal Locality)**
  一个最近被访问的指令或数据项，未来很可能会被再次访问。



- **空间局部性 (Spatial Locality)**
  如果一个程序访问了地址x的内存，那么它很可能很快会访问接近x的内存。



谁来处理TLB未命中？

•**硬件完全处理TLB未命中（CISC架构）**

- 硬件必须确切知道页表在内存中的位置
- 硬件会“遍历”页表，找到正确的页表项并**提取所需的转换信息，更新并重新**执行指令
- **硬件管理的TLB**

谁来处理TLB未命中？（续）

- **RISC架构也被称为“软件管理的TLB”**
  - •在TLB未命中时，硬件会引发异常（即Trap处理程序）
    - •**Trap处理程序**是操作系统中的一段代码，专门用于处理TLB未命中的情况

TLB 控制流算法（操作系统处理）

```
1:      VPN = (VirtualAddress & VPN_MASK) >> SHIFT
2:      (Success, TlbEntry) = TLB_Lookup(VPN)
3:      if (Success == True) // TLB Hit
4:              if (CanAccess(TlbEntry.ProtectBits) == True)
5:                      Offset = VirtualAddress & OFFSET_MASK
6:                      PhysAddr = (TlbEntry.PFN << SHIFT) | Offset
7:                      Register = AccessMemory(PhysAddr)
8:              else
9:                      RaiseException(PROTECTION_FAULT)
10:     else // TLB Miss
11:             RaiseException(TLB_MISS)
```

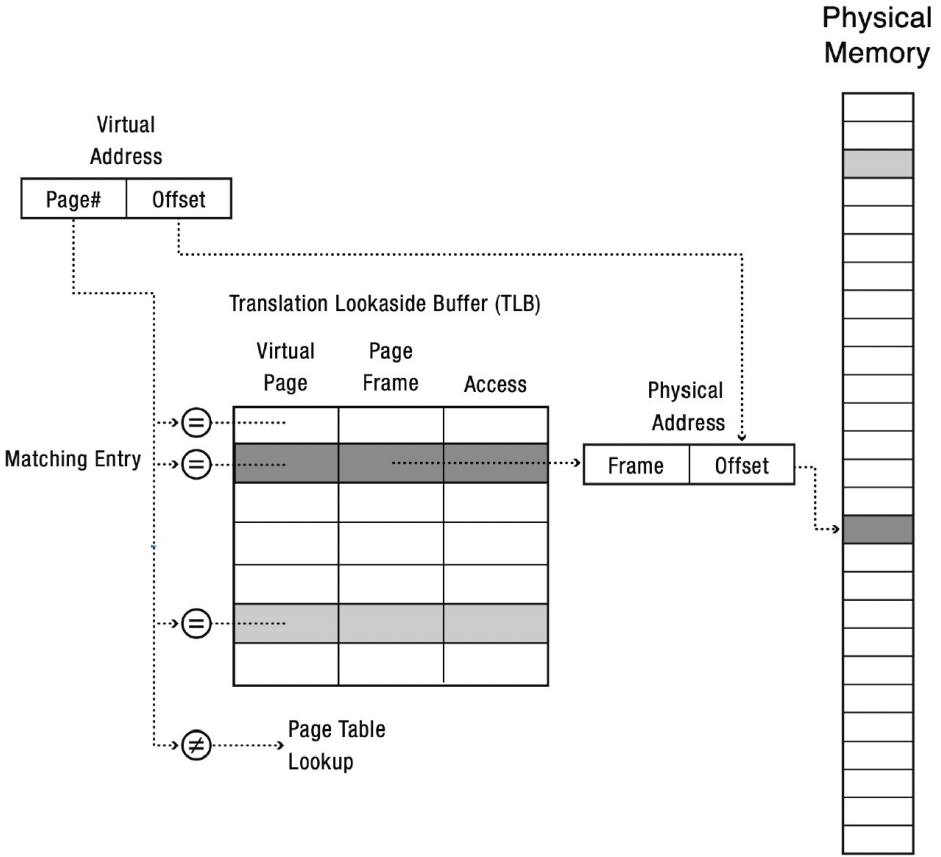
TLB项（TLB Entry）中的内容

- **TLB 是全相联的（fully associative）**

  - 一个典型的TLB可能包含 32、64 或 128 个条目
  - 硬件会**并行**搜索整个TLB以找到所需的映射
  - 其他位信息：有效位（valid bits）、保护位（protection bits）、地址空间标识符（address-space identifier）、脏位（dirty bit）

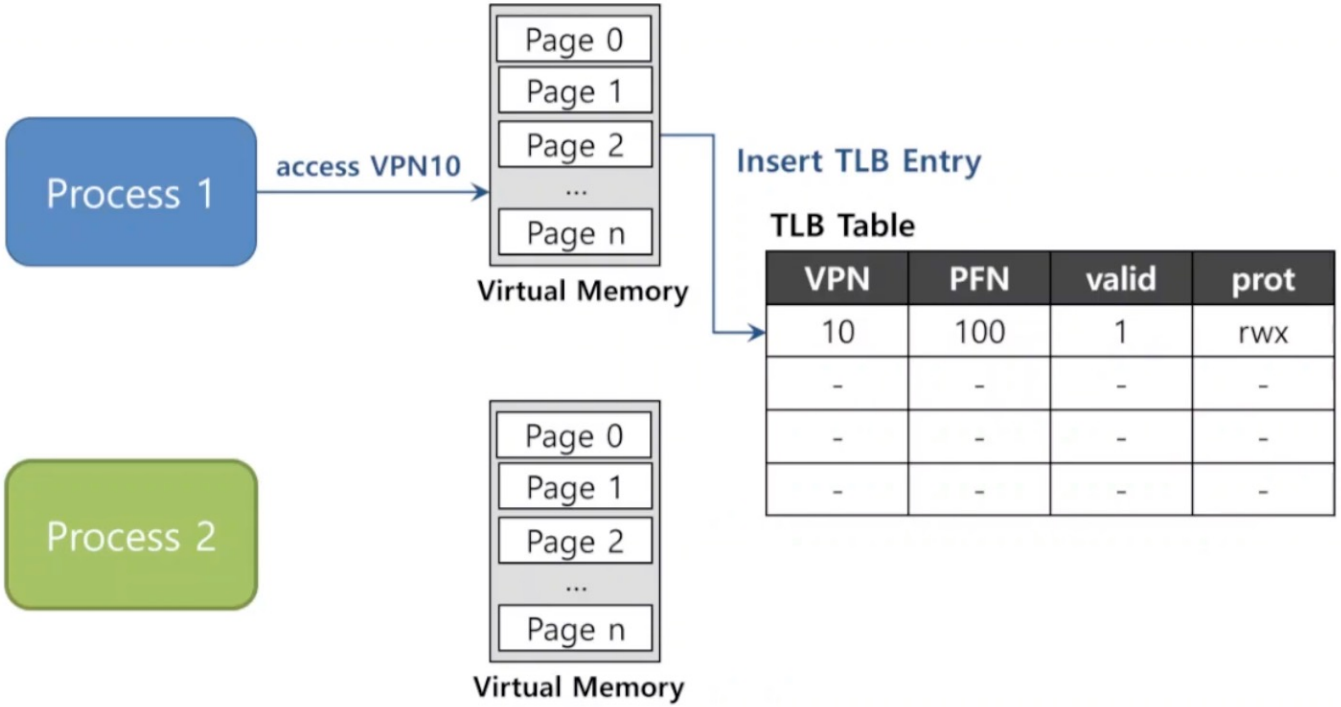
Typical TLB entry looks like this

示例：从虚拟地址到物理内存



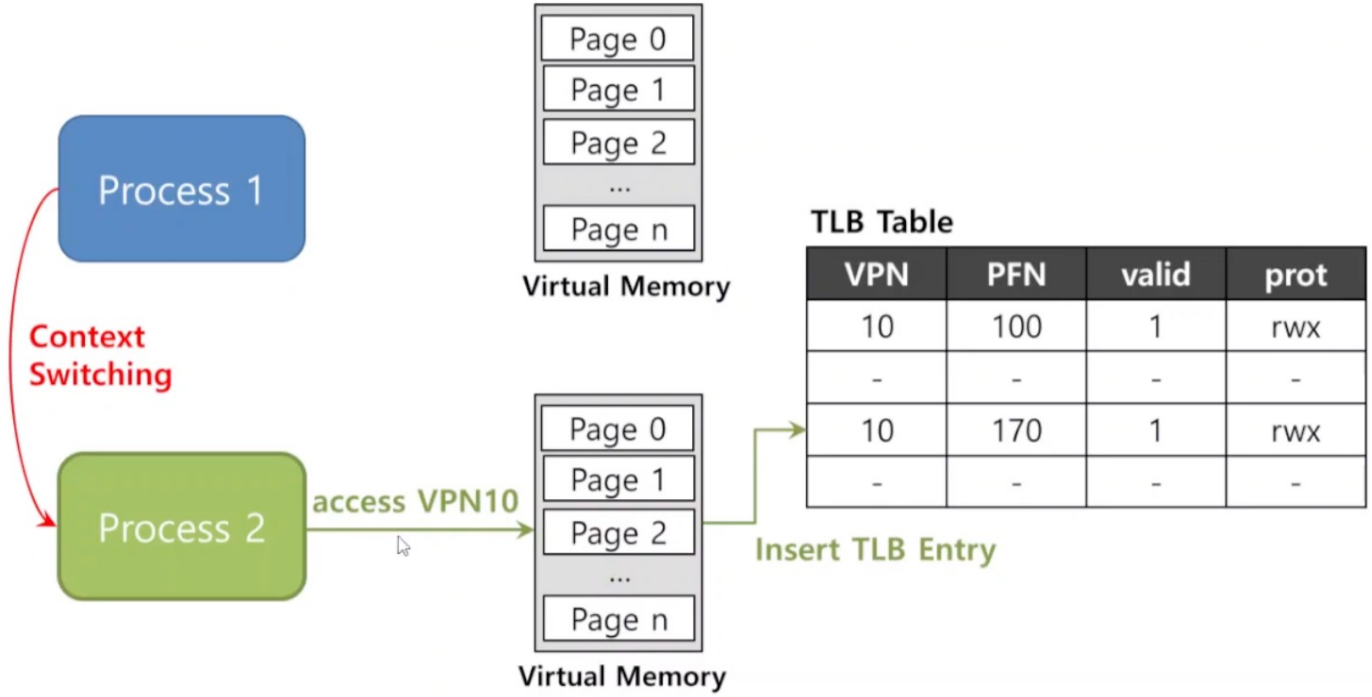
TLB问题1：上下文切换（Context Switching）

- **进程1**访问**虚拟页号VPN10**，并将映射关系插入TLB表：



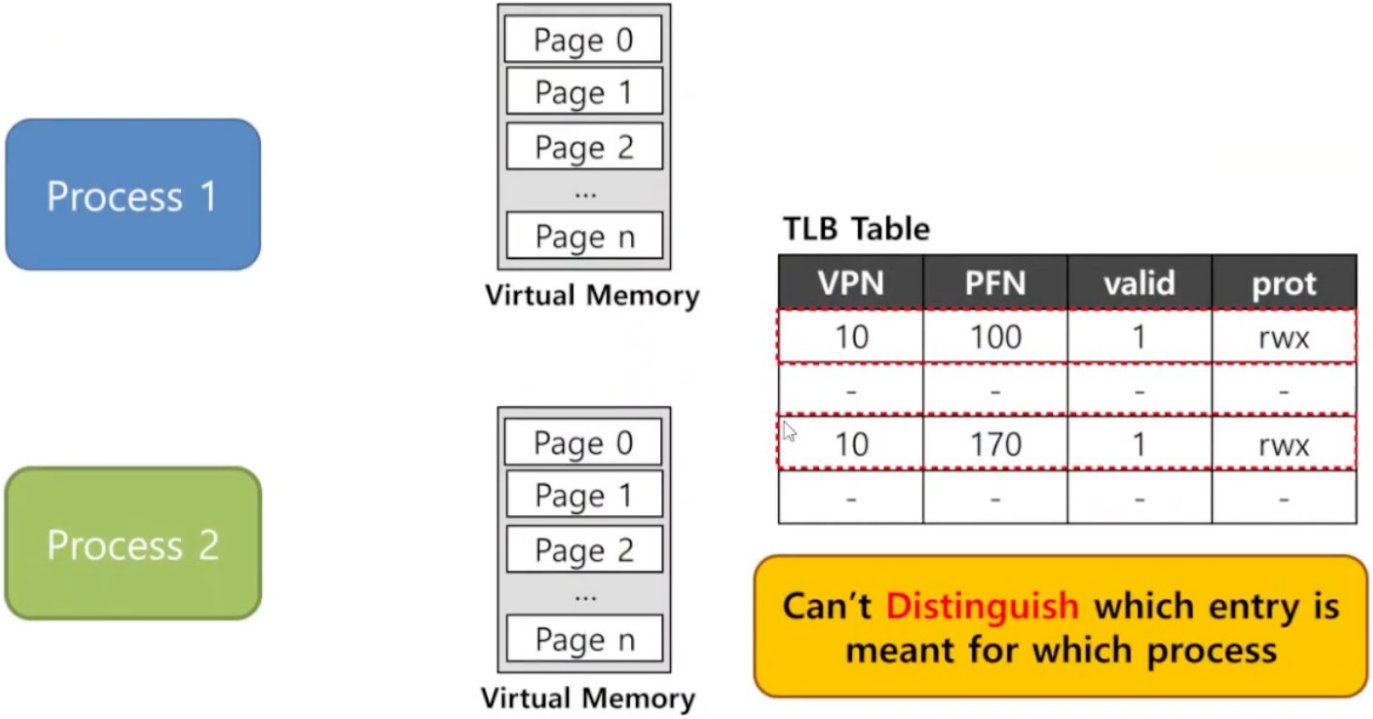
TLB问题1：上下文切换（Context Switching）

- 上下文切换发生，**进程2**开始运行并访问**虚拟页号VPN10**。操作系统将**进程2**的TLB条目插入到TLB表中。



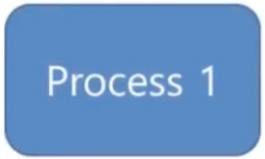
TLB问题1：上下文切换（Context Switching）

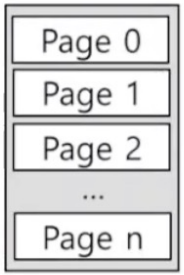
- **问题：**不能区分哪个条目属于哪个进程。

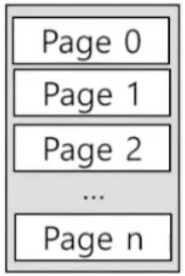


解决方案

- 在TLB中提供地址空间标识符（ASID）字段







TLB Table

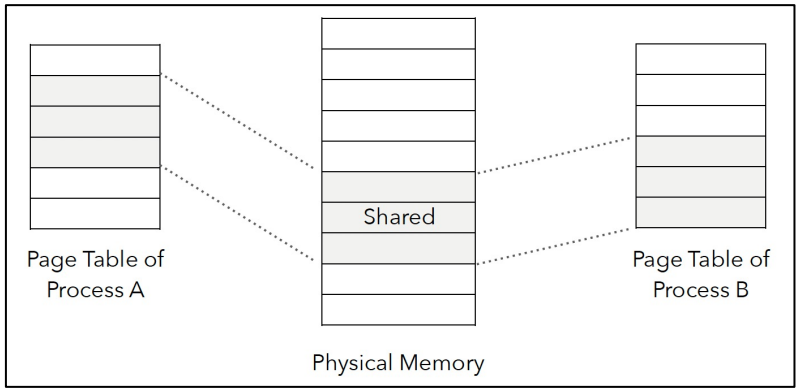
VPN	PFN	valid	prot	ASID
10	100	1	rwx	1
-	-	-	-	-
10	170	1	rwx	2
-	-	-	-	-

另一个案例

- 两个进程共享一个物理页帧：
  - **进程1**与**进程2**共享物理页帧101
  - **进程1**将该物理页帧映射到其虚拟地址空间的第10页
  - **进程2**将该物理页帧映射到其虚拟地址空间的第50页

页面共享是有用的，因为它减少了使用的物理页框数量。

VPN	PFN	valid	prot	ASID
10	101	1	rwx	1
-	-	-	-	-
50	101	1	rwx	2
-	-	-	-	-



TLB问题2：替换策略（TLB Replacement Policy）

- 目标是最小化TLB未命中率（minimize TLB miss rate）

- LRU（最近最少使用，Least Recently Used）
  - 驱逐最近未被使用的条目
  - 利用内存引用流中的局部性（locality）

Reference Row

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1

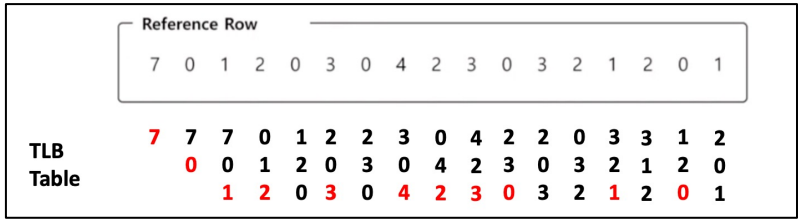
TLB Table

7	7	7	0	1	2	2	3	0	4	2	2	0	3	3	1	2
	0	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0
		1	2	0	3	0	4	2	3	0	3	2	1	2	0	1

TLB未命中总数：11

TLB问题2：替换策略（续）

TLB未命中总数：11



步骤	访问页面	TLB缓存情况（顶部为最近使用，底部为最久未使用）	是否命中	被替换页面（若miss）
1	7	7	miss	无（TLB初始为空）
2	0	0, 7	miss	无（TLB未满）
3	1	1, 0, 7	miss	无（TLB未满）
4	2	2, 1, 0	miss	7
5	0	0, 2, 1	hit	无
6	3	3, 0, 2	miss	1
7	0	0, 3, 2	hit	无
8	4	4, 0, 3	miss	2
9	2	2, 4, 0	miss	3
10	3	3, 2, 4	miss	0
11	0	0, 3, 2	miss	4
12	3	3, 0, 2	hit	无
13	2	2, 3, 0	hit	无
14	1	1, 2, 3	miss	0
15	2	2, 1, 3	hit	无
16	0	0, 2, 1	miss	3
17	1	1, 0, 2	hit	无

...

一个实际的TLB条目（MIPS R4000示例）

- 所有64位的TLB（现代化系统MIPS R4000示例，采用软件管理TLB）

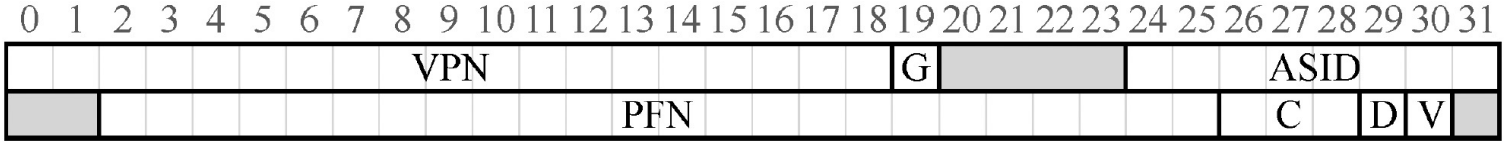


图 19.4　MIPS 的 TLB 项

Flag	Content
19-bit VPN	The rest reserved for the kernel.
24-bit PFN	Systems can support with up to 64GB of main memory($2^{24} * 4KB$ pages).
Global bit(G)	Used for pages that are globally-shared among processes.
ASID	OS can use to distinguish between address spaces.
Coherence bit(C)	determine how a page is cached by the hardware.
Dirty bit(D)	marking when the page has been written.
Valid bit(V)	tells the hardware if there is a valid translation present in the entry.

小结

我们介绍了TLB的基本算法及其实现，支持快速地址转换

我们介绍了TLB面临的2个问题