18-447 Lecture 22: Virtual Memory: Survey of Modern Systems

James C. Hoe
Dept of ECE, CMU
April 15, 2009

Announcements: Spring Carnival!!!
Final Thursday, May 7 5:30-8:30p.m Room TBA
Two Guest Lectures next Mon and Wed (not on final)
L23: multicore cache-coherence by Nikos Hardavellas
L24: advanced multicore design by Prof. Onur Mutlu

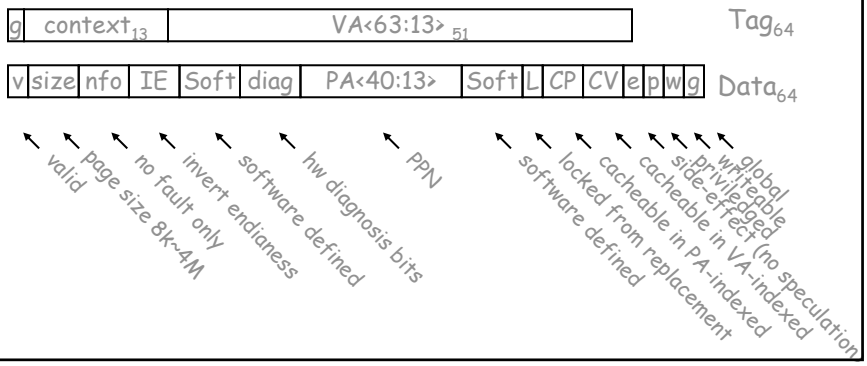
Handouts: Assigned Reading "Virtual memory in contemporary microprocessors." B. L Jacob and T. N. Mudge. IEEE Micro, July/August 1998

Case Studies

B. Jacob and T. Mudge, *Virtual Memory in Contemporary Processors*, IEEE Micro, vol. 18, no. 4, 1998.

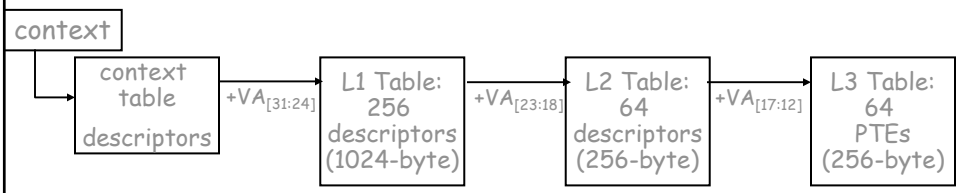
SPARC V9

- 64-bit Virtual Address
  - an implementation can choose not to map the high-order bits (must be sign-extended from the highest mapped bit)
  - e.g. UltraSPARC 1 maps only the lower 44 bits
- physical address space size set by implementation
- 64 entry fully associative I-TLB and D-TLB



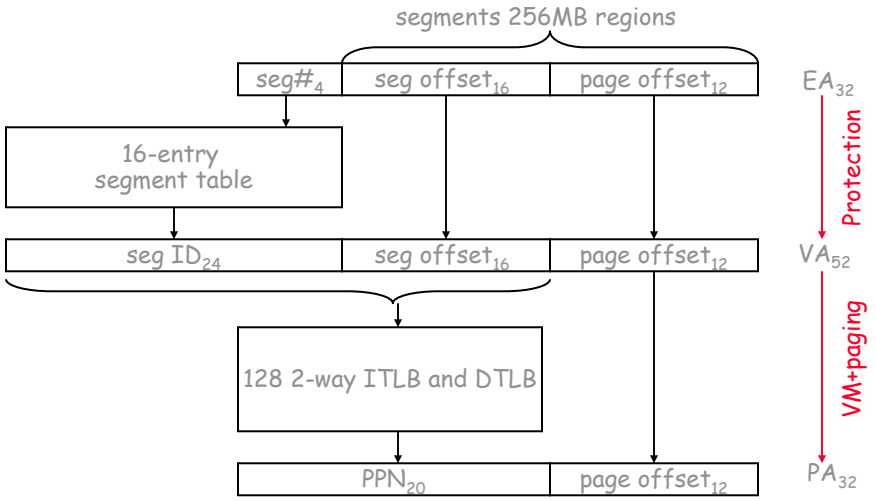
TLB Miss Handling

- SPARC V8 (32-bit) defines a 3-level hierarchical page table for HW MMU page-table walk



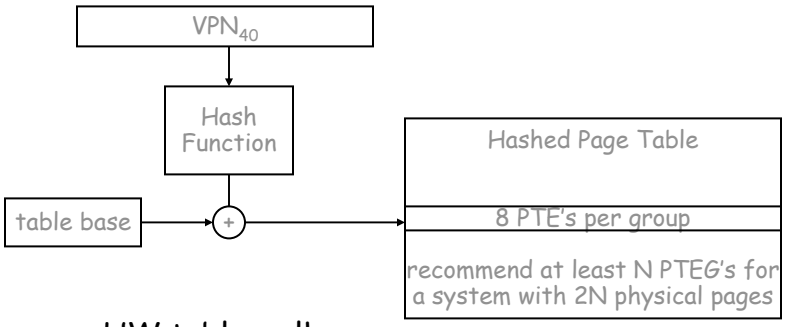
- SPARC V9 (64-bit) defines Translation Storage Buffer
  - a software managed, direct-mapped cache of PTEs (think hashed page table)
  - HW assisted address generation on a TLB miss, e.g., for 8-k pages {$TSBbase_{63:21}$, $Logic(TSBbase_{20:13}, VA_{32:22}, size, split?)$, $VA_{21:13}$,0000}
  - TLB miss handler (SW) search TSB. If TSB misses, a slower TSB-miss handler takes over

IBM PowerPC (32-bit)



64-bit PowerPC = 64-bit EA -> 80-bit VA → 64-bit PA. How many segments in EA?

IBM PowerPC Hashed Page Table



- HW table walk
  - VPN hashes into a PTE group (PTEG) of 8
  - 8 PTEs searched sequentially for tag match
  - if not found in first PTE group search a second PTE group
  - if not found in the 2nd PTE group, trap to software handler
- Hashed table structure also used for EA to VA mapping in 64-bit implementations

MIPS R10K

- 64-bit virtual address
  - top 2 bits set kernel/supervisor/user mode
  - additional bits set cache and translation behavior
  - bit 61-40 not translate at all (holes in the VA??)
- 8-bit ASID (address space ID) distinguishes between processes
- 40-bit physical address
- Translation -
  "64"-bit VA and 8-bit ASID → 40-bit PA

MIPS TLB

- 64-entry fully associative unified TLB
  - paired: each entry maps 2 consecutive VPNs to 2 different PPNs
  - software managed
    - 7-instruction page table walk in the best case
    - TLB Write Random: chooses a random entry for TLB replacement
    - OS can exclude some number of TLB entry (low range) to be excluded from the random selection, to hold translations that cannot miss or should not miss
- TLB entry

VPN_{20}	$ASID_6$	0_6
PPN_{20}	ndvg	0_8

R2000

  - N: noncacheable
  - D: dirty (actually a write-enable bit)
  - V: valid
  - G: global entry, i.e., ignore ASID matching

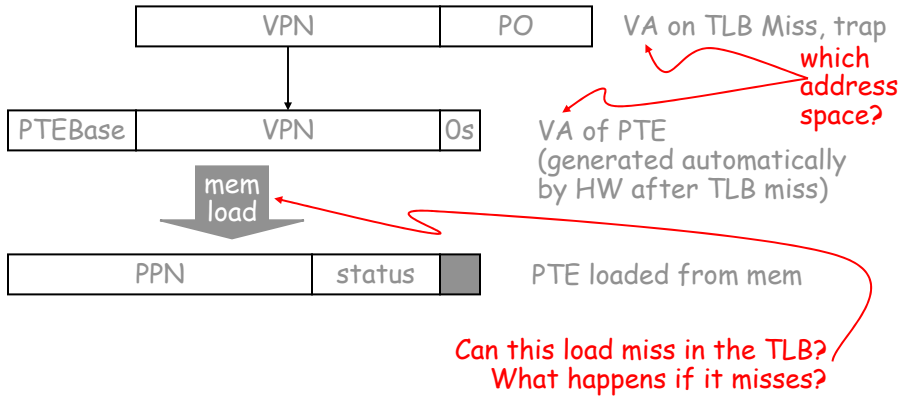
MIPS Bottom-Up Hierarchical Table

- Page table organization is not part of the ISA
- Reference design optimized for software TLB miss handling

- Bottom-Up Table
  - start with a basic 2-level hierarchical table (32-bit case)
  - map all of the L2 tables (empty or not) linearly in the mapped kseg
  - VPN is the index into this linear table in VA

A linear translation table that scales with VA size,
Is this okay?

Bottom-Up Table Walk



Notice translation also eats up TLB entries!

UserTLB Miss Handling


```
mfc0 k0,tlbcxt      # move the contents of TLB
                    #      context register into k0
mfc0 k1,epc         # move PC of faulting load
                    #      instruction into k1
lw k0,0(k0)         # load thru address that was
                    #      inTLB context register
mtc0 k0,entry_lo    # move the loaded value
                    #      into the EntryLo register
tlbwr               # write entry into the TLB
                    #      at a random slot number
j k1                # jump to PC of faulting
                    #      load instruction to retry
rfe                 # RESTORE FROM
                    #      EXCEPTION
```


HP PA-RISC: PID and AID

- 2-level translation:

  64-bit EA → 96-bit VA (global)→ 64-bit PA

- Variable sized segmented EA to VA
- A different twist on protection
  - everyone else: limit what can be named by a process
    - In PowerPC, OS controls what VA can be reached by a process by controlling what's in the segment registers
  - HP-RISC: rights-based access control
    - User controls segment registers, i.e., user can generate any VA it wants
    - Each virtual page has an access ID (not related to ownership by a processes) assigned by the OS
    - Each process has 8 active protection IDs in special HW registers controlled by the OS
    - A process can only access a page if it has the key (PID) that fits the lock (AID)

Intel x86

- Intended for two-level address translation with segments for naming and protection (similar to PPC)
  - user private 48-bit effective address
    16-bit segment number (implicit) + 32-bit segment offset
    each addr register has a corresponding segment register
  - a global 32-bit virtual address
    20-bit page number + 12-bit page offset
  - an implementation defined paged physical address space
    What is strange about this?
- 32-bit VA too small to be shared by multiple processes
- No major OS today uses segmentation features
  - code, data, stack segments always mapped to 0~(2^{32}-1)
  - time multiplex VA between processes for naming and protection
  - set MMU to use a different table on context switch
  - must flush TLB on context switch because TLB entries are not defined to have ASIDs