

**Lecture 37  
 VM III**

2004-11-24



**Lecturer PSOE Dan Garcia**  
[www.cs.berkeley.edu/~ddgarcia](http://www.cs.berkeley.edu/~ddgarcia)

This 230mph electric car accelerates faster than a Porsche 911 Turbo and goes 200 miles on a single charge! Wow!



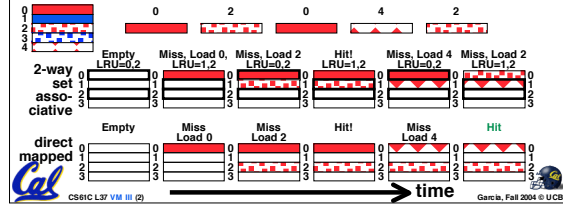
CS61C L37 VM III (1)

[eliica.com](http://eliica.com)

Garcia, Fall 2004 © UCB

**Peer Instruction Correction**

- A direct-mapped cache will never out-perform a 2-way set-associative cache of the same size.
  - I said "TRUE ... increased associativity!"
  - Right Answer "FALSE ... consider the following"
    - We have 4 byte cache, block size = 1 byte. Compare a 2-way set-associative cache (2 sets using LRU replacement) with a direct mapped cache (four rows).



**Peer Instruction**

1. Increasing at least one of {associativity, block size} always a win
2. Higher DRAM bandwidth translates to a lower miss rate
3. DRAM access time improves roughly as fast as density

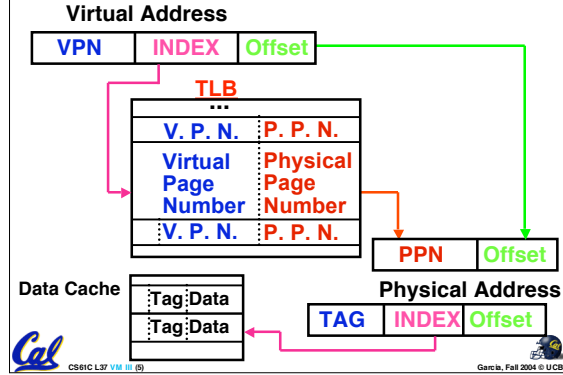
	A	B	C
1:	F	F	F
2:	F	F	T
3:	F	T	F
4:	F	T	T
5:	T	F	F
6:	T	F	T
7:	T	T	F
8:	T	T	T



CS61C L37 VM III (2)

Garcia, Fall 2004 © UCB

**Address Translation & 3 Concept tests**



CS61C L37 VM III (3)

Garcia, Fall 2004 © UCB

**Peer Instruction (1/3)**

◦ 40-bit virtual address, 16 KB page



◦ 36-bit physical address



◦ Number of bits in Virtual Page Number/ Page offset, Physical Page Number/Page offset?

- 1: 22/18 (VPN/PO), 22/14 (PPN/PO)
- 2: 24/16, 20/16
- 3: 26/14, 22/14
- 4: 26/14, 26/10
- 5: 28/12, 24/12



CS61C L37 VM III (4)

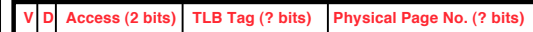
Garcia, Fall 2004 © UCB

**Peer Instruction (2/3): 40b VA, 36b PA**

◦ 2-way set-associ. TLB, 256 "slots", 40b VA:



◦ TLB Entry: Valid bit, Dirty bit, Access Control (say 2 bits), Virtual Page Number, Physical Page Number



◦ Number of bits in TLB Tag / Index / Entry?

- 1: 12 / 14 / 38 (TLB Tag / Index / Entry)
- 2: 14 / 12 / 40
- 3: 18 / 8 / 44
- 4: 18 / 8 / 58



CS61C L37 VM III (5)

Garcia, Fall 2004 © UCB

### Peer Instruction (3/3)

- 2-way set-assoc, 64KB data cache, 64B block



- Data Cache Entry: Valid bit, Dirty bit, Cache tag + ? bits of Data



- Number of bits in Data cache Tag / Index / Offset / Entry?

- 1: 12 / 9 / 14 / 87 (Tag/Index/Offset/Entry)
- 2: 20 / 10 / 6 / 86
- 3: 20 / 10 / 6 / 534
- 4: 21 / 9 / 6 / 87
- 5: 21 / 9 / 6 / 535



CS61C L37 VM III (10)

Garcia, Fall 2004 © UCB

### 4 Qs for any Memory Hierarchy

- Q1: Where can a block be placed?
  - One place (direct mapped)
  - A few places (set associative)
  - Any place (fully associative)
- Q2: How is a block found?
  - Indexing (as in a direct-mapped cache)
  - Limited search (as in a set-associative cache)
  - Full search (as in a fully associative cache)
  - Separate lookup table (as in a page table)
- Q3: Which block is replaced on a miss?
  - Least recently used (LRU)
  - Random
- Q4: How are writes handled?
  - Write through (Level never inconsistent w/ lower)
  - Write back (Could be "dirty", must have dirty bit)

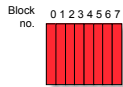


CS61C L37 VM III (12)

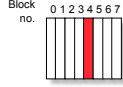
Garcia, Fall 2004 © UCB

### Q1: Where block placed in upper level?

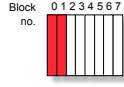
- Block 12 placed in 8 block cache:
  - Fully associative
  - Direct mapped
  - 2-way set associative
    - Set Associative Mapping = Block # Mod # of Sets



Fully associative:  
block 12 can go  
anywhere



Direct mapped:  
block 12 can go  
only into block 4  
(12 mod 8)



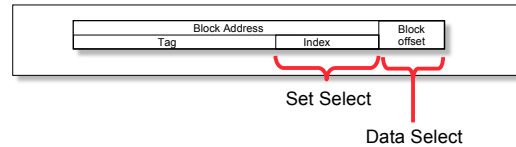
Set Set Set Set  
0 1 2 3  
Set associative:  
block 12 can go  
anywhere in set 0  
(12 mod 4)



CS61C L37 VM III (13)

Garcia, Fall 2004 © UCB

### Q2: How is a block found in upper level?



- Direct indexing (using index and block offset), tag compares, or combination
- Increasing associativity shrinks index, expands tag



CS61C L37 VM III (14)

Garcia, Fall 2004 © UCB

### Q3: Which block replaced on a miss?

- Easy for Direct Mapped
- Set Associative or Fully Associative:
  - Random
  - LRU (Least Recently Used)

Miss Rates Associativity:	2-way		4-way		8-way	
	LRU	Ran	LRU	Ran	LRU	Ran
Size						
16 KB	5.2%	5.7%	4.7%	5.3%	4.4%	5.0%
64 KB	1.9%	2.0%	1.5%	1.7%	1.4%	1.5%
256 KB	1.15%	1.17%	1.13%	1.13%	1.12%	1.12%



CS61C L37 VM III (15)

Garcia, Fall 2004 © UCB

### Q4: What to do on a write hit?

- Write-through
  - update the word in cache block and corresponding word in memory
- Write-back
  - update word in cache block
  - allow memory word to be "stale"
  - => add 'dirty' bit to each line indicating that memory be updated when block is replaced
  - => OS flushes cache before I/O !!!
- Performance trade-offs?
  - WT: read misses cannot result in writes
  - WB: no writes of repeated writes



CS61C L37 VM III (16)

Garcia, Fall 2004 © UCB

### Three Advantages of Virtual Memory

#### 1) Translation:

- Program can be given consistent view of memory, even though physical memory is scrambled
- Makes multiple processes reasonable
- Only the most important part of program (“**Working Set**”) must be in physical memory
- Contiguous structures (like stacks) use only as much physical memory as necessary yet still grow later



CS61C L37 VM III (17)



Garcia, Fall 2004 © UCB

### Three Advantages of Virtual Memory

#### 2) Protection:

- Different processes protected from each other
- Different pages can be given special behavior
  - (Read Only, Invisible to user programs, etc).
- Kernel data protected from User programs
- Very important for protection from malicious programs ⇒ Far more “viruses” under Microsoft Windows
- Special Mode in processor (“Kernel mode”) allows processor to change page table/TLB

#### 3) Sharing:

- Can map same physical page to multiple users (“Shared memory”)



CS61C L37 VM III (18)



Garcia, Fall 2004 © UCB

### Why Translation Lookaside Buffer (TLB)?

- Paging is most popular implementation of virtual memory (vs. base/bounds)
- Every paged virtual memory access must be checked against Entry of Page Table in memory to provide protection
- Cache of Page Table Entries (TLB) makes address translation possible without memory access in common case to make fast



CS61C L37 VM III (19)



Garcia, Fall 2004 © UCB

### Bonus slide: Virtual Memory Overview (1/4)

- User program view of memory:
  - Contiguous
  - Start from some set address
  - Infinitely large
  - Is the only running program
- Reality:
  - Non-contiguous
  - Start wherever available memory is
  - Finite size
  - Many programs running at a time



CS61C L37 VM III (20)



Garcia, Fall 2004 © UCB

### Bonus slide: Virtual Memory Overview (2/4)

- Virtual memory provides:
  - illusion of contiguous memory
  - all programs starting at same set address
  - illusion of ~ infinite memory ( $2^{32}$  or  $2^{64}$  bytes)
  - protection



CS61C L37 VM III (21)



Garcia, Fall 2004 © UCB

### Bonus slide: Virtual Memory Overview (3/4)

- Implementation:
  - Divide memory into “chunks” (pages)
  - Operating system controls page table that maps virtual addresses into physical addresses
  - Think of memory as a cache for disk
  - TLB is a cache for the page table



CS61C L37 VM III (22)



Garcia, Fall 2004 © UCB

### Bonus slide: Virtual Memory Overview (4/4)

#### Let's say we're fetching some data:

- Check TLB (input: VPN, output: PPN)
  - hit: fetch translation
  - miss: check page table (in memory)
    - Page table hit: fetch translation
    - Page table miss: page fault, fetch page from disk to memory, return translation to TLB
- Check cache (input: PPN, output: data)
  - hit: return value
  - miss: fetch value from memory



CS61C L37 VM III (23)

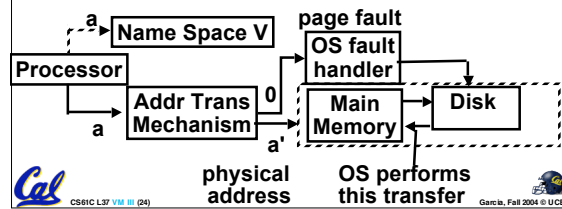


Garcia, Fall 2004 © UCB

### Address Map, Mathematically

$V = \{0, 1, \dots, n - 1\}$  virtual address space ( $n > m$ )  
 $M = \{0, 1, \dots, m - 1\}$  physical address space  
MAP:  $V \rightarrow M \cup \{\emptyset\}$  address mapping function

MAP( $a$ ) =  $a'$  if data at virtual address  $a$  is present in physical address  $a'$  and  $a'$  in  $M$   
=  $\emptyset$  if data at virtual address  $a$  is not present in  $M$



CS61C L37 VM III (24)



Garcia, Fall 2004 © UCB

### And in Conclusion...

- Virtual memory to Physical Memory Translation too slow?
  - Add a cache of Virtual to Physical Address Translations, called a **TLB**
- Spatial Locality means Working Set of Pages is all that must be in memory for process to run fairly well
- Virtual Memory allows protected sharing of memory between processes with less swapping to disk



CS61C L37 VM III (25)



Garcia, Fall 2004 © UCB