More Caches & Virtual Memory

Scott Beamer (cs61c-tb)

Cache Misses

- *Compulsory Misses:* When this is the first time the missed address has been accessed, so it has never been in the cache
- *Conflict Misses:* When the missed address was in the cache, but was evicted because another address mapped to the same location (can't happen in fully-associative)
- *Capacity Misses:* When the missed address was evicted because the cache isn't big enough (can't happen in direct mapped)

Associative Caches

- Give more flexibility in choosing location to put blocks in cache
- This is done to reduce conflict misses
- This is done by using more tag comparators. Every possible row where a block could be now needs a comparator (direct mapped had only 1 comparator because each block could only map to 1 row)
- *Set Associative:* Like direct mapped, but instead of mapping each block to a single row, it is now mapped to a group of rows (a set).
- *Fully Associative:* Any block can go to any row
- Generalizing this you can see that a direct mapped cache is 1-Way Set Associative, and a fully associative cache is N-Way Set Associative where N is the number of rows.
- Associativity can reduce conflict misses, but this comes at the cost of extra comparators which makes it slower, more expensive, and infeasible for larger sizes.

Replacement Policies

- Needed for caches with some associativity (not direct mapped) to determine which row to use
- Least Recently Used (LRU): Evict the row that was last used the longest time ago
- Random: Pick a row at random

Cache Problem

- From Lab12 - These are the access times from the Wazcog Mark IV computer for the same cache test program.
- What is the cache size in bytes?
- What is the block size in bytes?
- What is the allocation policy and the associativity?

Size/Stride	4	8	16	32	64	128	256
64	109	126	122	121	101		
128	120	118	116	121	117	105	
256	119	116	102	117	113	113	110
512	466	966	949	949	983	308	311
1024	458	949	983	966	949	983	307
2048	449	992	954	966	979	979	979

Cache Details

- *Average Memory Access Time (AMAT):* How long it takes to get something out of the cache, including the the weighted time for misses. It generally is computed like
  AMAT = hit time + (miss time)(miss rate)
- *Write Through:* A cache where very write is also written to the level above
- *Write Back:* A cache that writes to the higher level only if it is taking that block out. It uses a dirty bit to see if the block has been modified and needs to be copied.
- Cache performance is often improved by adding another level or levels of larger but slower caches

Reasons for Virtual Memory

- *Size:* VM can give the illusion to a program that it has much more memory than there is physical memory by spilling onto disk if needed.
- *Entire Address Space:* VM gives each program the illusion that it is the only program running and that it has control of the entire virtual address space.
- *Security:* If done correctly by OS, it is impossible for one program to mess with another.

How Virtual Memory Works

- OS adds a level of indirection to every memory access by a program
- Each program requests a *virtual address* that is mapped to a *physical address* by VM
- *Pages:* The unit of transfer for VM (like block for caches)
- Each address breaks into an offset (for within the page) and page number

Physical Address	
Physical Page Number (PPN)	Page Offset

Virtual Address	
Virtual Page Number (VPN)	Page Offset

- *Page Table:* A map for each program than translates VPN's to PPN's and stores relevant information such as: valid, dirty, and if the page is in memory or disk.
- *Translation Look-Aside Buffer (TLB):* Hardware to improve VM performance. It stores part of the Page Table (like a cache) for quicker access. It is placed in between the CPU and caches instead of in between the caches and memory. Why?

VM Problems

Virtual Address Bits	Physical Address Bits	Page Size	VPN Bits	PPN Bits	Bits per row of PT (4 extra bits)
32	32	16KB			
32	26			13	
	32		21		21
		32KB	25		25
64			48		28