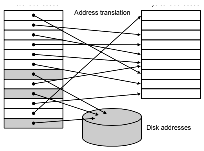
Virtual Memory

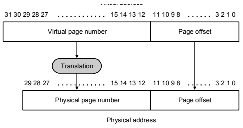
- Main memory can act as a cache for the secondary storage (disk)



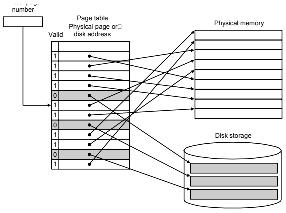
- Advantages:
  - illusion of having more physical memory
  - program relocation
  - protection

Pages: virtual memory blocks

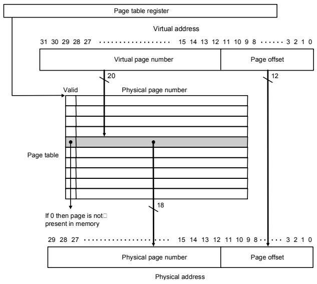
- Page faults: the data is not in memory, retrieve it from disk
  - huge miss penalty, thus pages should be fairly large (e.g., 4KB)
  - reducing page faults is important (LRU is worth the price)
  - can handle the faults in software instead of hardware
  - using write-through is too expensive so we use writeback



Page Tables

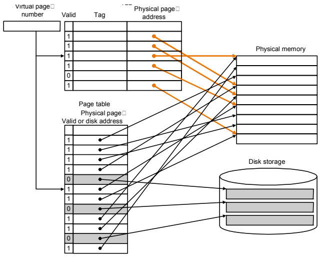


Page Tables

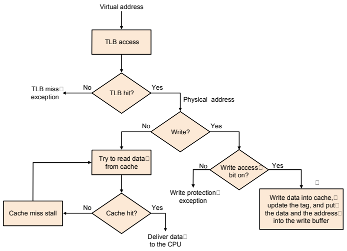


Making Address Translation Fast

- A cache for address translations: translation lookaside buffer



TLBs and Caches



Replacement Policies

- Replacement Policies in Multi-way Set Associative caches
  - Random: Replace any line arbitrarily
  - Least Recently Used (LRU): Find the least recently used line to replace
  - Keep Most Recently Used (MRU): Keep the last used line in the set and replace any other randomly
- LRU performs the best
- MRU does equally well

LRU Scheme

- We explain LRU with an example of a 4-way set associative cache
- Associate a 2-bit counter with each line (log k bit for k-way cache)
- Initially all lines are invalid
- For a miss bring a new line in an invalid line, make it valid, set its counter to zero, increment all other counters
  - If no invalid line, replace the line with counter value = 3, set its counter to zero, increment all other counters
- For a hit, set the accessed line's counter to zero and increment counters of those lines whose values is smaller than the accessed line
- Try this algorithm for an examples where lines read are 0, 64, 128, 64, 192, 256, 128, 0, 256, 192, 64…
  - There are 64 lines in each cache and it is 4-way set associative

Reading or Writing a Memory word

- Check the address in TLB
- If not there, get the physical translation and also store the entry in TLB
  - Penalty 40-50 cycles
- If page itself is not present, page fault occurs
  - Read the page, update page table and TLB
  - Penalty 100's of thousands cycles
- Once physical address is there If there, perform read or write in cache
- If cache miss
  - Read the line in cache for read
  - May need to replace a dirty or clean line
    - Penalty 20-40 cycles
  - For Write read the line if write allocate, else write around
- If cache hit read or write in cache
  - Also write in main memory if write through

A Big Example

- Instruction Frequency: LW(20%), SW(10%), R(50%), BR(15%), J(5%)
- Branch Penalty: 3 cycles on 20% mis-predictions = 15*0.20*3 = 9 cycles
- Data Cache 1: Miss rate 10% (of load/store), write back, write around, 50% dirty replacement, penalty for reading or writing a line 20 cycles
  - Load penalty = 20*0.10*0.50*20 + 20*0.10*0.50*(20+20) = 60 cycles
  - Store Penalty = 0 (because of write around, otherwise will be 30)
- Data Cache 2: Miss rate 5% (of load/store), write back, write allocation, 50% dirty replacement, penalty for reading or writing a line 100 cycles
  - Load penalty = 20*0.05*0.5*100 + 20*0.05*0.5*(100+200) = 150 cycles
  - Store Penalty = 10*0.05*0.5*100 + 10*0.05*0.5*(100+100) = 75 cycles
- TLB: Miss Rate 2% (of load/store), Miss Penalty 100 cycles
  - Total Penalty = (20+10)*0.02*100 = 60 cycles
- Page faults: 0.01% (of load/store), Penalty 300,000 cycles
  - Total Penalty = (20+10)*0.0001*300,000 = 900 cycles
- Total Time = 100+9+60+150+75+60+900 = 1354 cycles, or CPI=13.54
- Notice that miss rates can be spacified per instruction or per load/store

Misses and Replacement Policies

- 3 C Misses
  - Compulsory: Miss will have to occur on first read (or write)
  - Capacity: A line is replaced and then brought back
  - Conflict: a miss occurs as some other line is occupying that line
- Example Suppose we read line a first time (no line is in cache), then read line b that replaces line a, and then read line a again
- The first and second misses are compulsory, second miss is also capacity and conflict, and the third miss is capacity (and also conflict)
- The terminology can be confusing here
  - The first read is always classified as compulsory
  - The replacement and read back is conflict if there was place in cache elsewhere but you had to bring it at that place due to mapping
  - If there was no place at all then it is capacity miss (like cache is full in a fully associative cache)

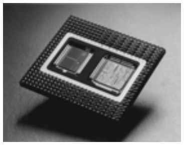
Virtual Memory: Other Translation Schemes

- In a single-level translation
  - 32 bit virtual address
  - 4KB Page size (12 bit address in each page)
  - Leaves 20-bit page address => 1 Million Pages =>4MB for Table
- One alternate is to only have a limited size page table with Hi and Lo Checks
  - But program use many addresses segments
- Alternate is to have a two level page table
- Divide page addresses in two parts of 10 bits each
  - There are 1K tables of 1K entries each (total is still 1M entries)
  - Most significant 10 bits points to a table (with 1K entries, each 4 bytes long, a total of 4KB that fits in a page) that contains the address of that part of table
  - Least significant 10 bits are used to access a particular entry in the selected table
- We only need to keep the first table (pointing to real tables) and some of the second level tables in memory

Modern Systems

- Very complicated memory systems:

Characteristic	Intel Pentium Pro	PowerPC 604
Virtual address	32 bits	52 bits
Physical address	32 bits	32 bits
Page size	4 KB, 4 MB	4 KB, selectable, and 256 MB
TLB organization	A TLB for instructions and a TLB for data Both four-way set associative Pseudo-LRU replacement Instruction TLB: 32 entries Data TLB: 64 entries TLB misses handled in hardware	A TLB for instructions and a TLB for data Both two-way set associative LRU replacement Instruction TLB: 128 entries Data TLB: 128 entries TLB misses handled in hardware



Characteristic	Intel Pentium Pro	PowerPC 604
Cache organization	Split instruction and data caches	Split intruction and data caches
Cache size	8 KB each for instructions/data	16 KB each for instructions/data
Cache associativity	Four-way set associative	Four-way set associative
Replacement	Approximated LRU replacement	LRU replacement
Block size	32 bytes	32 bytes
Write policy	Write-back	Write-back or write-through

Some Issues

- Processor speeds continue to increase very fast
  — much faster than either DRAM or disk access times
- Design challenge: dealing with this growing disparity
- Trends:
  - synchronous SRAMs (provide a burst of data)
  - redesign DRAM chips to provide higher bandwidth or processing
  - restructure code to increase locality
  - use prefetching (make cache visible to ISA)