CS 305
Computer Architecture – Virtual Memory

A Real Problem

- What if you wanted to run a program that needs more memory than you have?

A Real Problem

- What if you wanted to run a program that needs more memory than you have?
  - You could store the whole program on disk, and use memory as a cache for the data on disk. This is one feature of virtual memory.
  - Before virtual memory, programmers had to manually manage loading "overlays" (chunks of instructions & data) off disk before they were used. This is an incredibly tedious, not to mention error-prone, process.

More Real Problems

- Running multiple programs at the same time brings up more problems.

1. Even if each program fits in memory, running 10 programs might not.
2. Multiple programs may want to store something at the same address.
3. How do we protect one program's data from being read or written by another program?

More Real Problems

- Running multiple programs at the same time brings up more problems.

1. Even if each program fits in memory, running 10 programs might not.
   - This is really the same problem as on the previous slide.

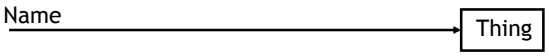
2. Multiple programs may want to store something at the same address.
   - I.e., what if both Program A and B want to use address 0x10000000 as the base of their stack?
   - It is impractical (if not impossible) to compile every pair of programs that could get executed together to use distinct sets of addresses.

3. How do we protect one program's data from being read or written by another program?

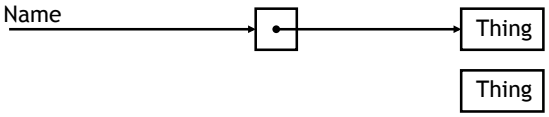
Indirection

- "Any problem in CS can be solved by adding a level of indirection"

- Without Indirection



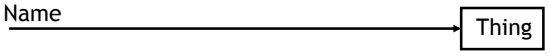
- With Indirection



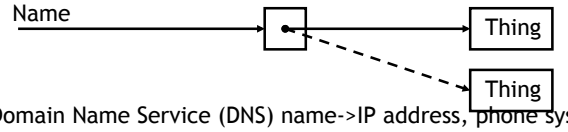
Indirection

- **Indirection**: Indirection is the ability to reference something using a name, reference, or container instead the value itself. A flexible mapping between a name and a thing allows changing the thing without notifying holders of the name.

- Without Indirection



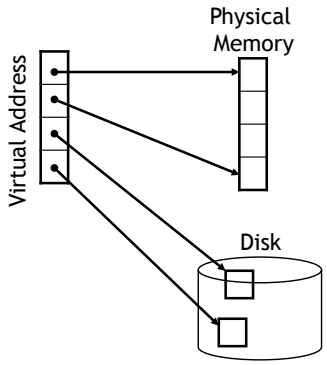
- With Indirection



- **Examples**:
  Pointers, Domain Name Service (DNS) name->IP address, phone system (e.g., cell phone number portability), snail mail (e.g., mail forwarding), 911 (routed to local office), DHCP, color maps, call centers that route calls to available operators, etc.

Virtual Memory

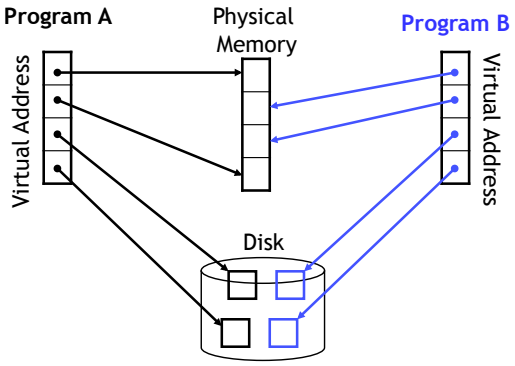
- We translate "virtual addresses" used by the program to "physical addresses" that represent places in the machine's "physical" memory.
  - The word "translate" denotes a level of indirection



A virtual address can be mapped to either physical memory or disk.

Virtual Memory

- Because different processes will have different mappings from virtual to physical addresses, two programs can freely use the same virtual address.
- By allocating distinct regions of physical memory to A and B, they are prevented from reading/writing each others data.



Virtual to Physical Memory

- Each process has its own virtual address space, which may be larger than the physical address space (namely size of RAM).
- The concept of a virtual address space that is bound to a separate physical address space is central to proper memory management
- Virtual address – generated by the CPU
- Physical address – address seen by the memory controller

Caching revisited

- Once the translation infrastructure is in place, the problem boils down to caching.
  - We want the size of disk, but the performance of memory.
- The design of virtual memory systems is really motivated by the high cost of accessing disk.
  - While memory latency is ~100 times that of cache, disk latency is ~100,000 times that of memory.
    - i.e., the miss penalty is a real whopper.
- Hence, we try to minimize the miss rate:
  - VM "pages" are much larger than cache blocks. Why?
  - A fully associative policy is used.
    - With approximate LRU
- Should a write-through or write-back policy be used?

Finding the right page

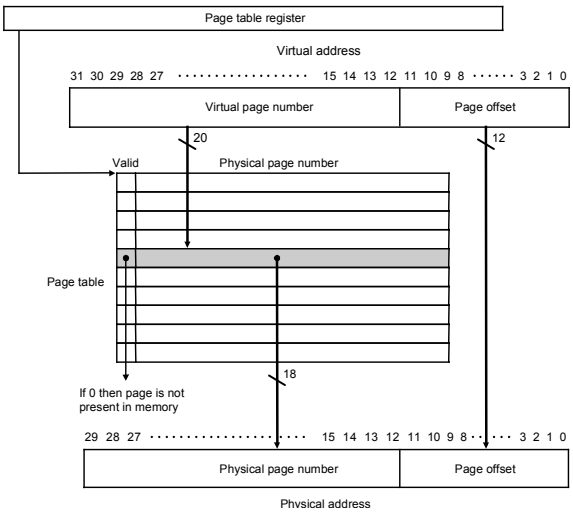
- If it is fully associative, how do we find the right page without scanning all of memory?

Finding the right page

- If it is fully associative, how do we find the right page without scanning all of memory?
  - Use an index, just like you would for a book.

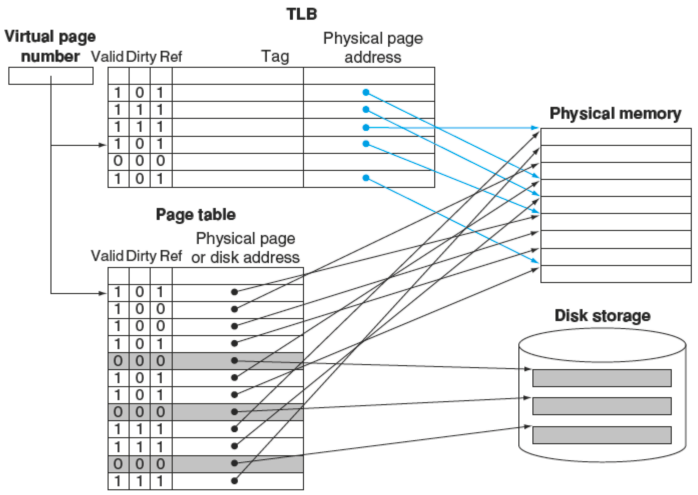
- Our index happens to be called the page table:
  - Each process has a separate page table
    - A "page table register" points to the current process's page table
  - The page table is indexed with the virtual page number (VPN)
    - The VPN is all of the bits that aren't part of the page offset.
  - Each entry contains a valid bit, and a physical page number (PPN)
    - The PPN is concatenated with the page offset to get the physical address
  - No tag is needed because the index is the full VPN.

Page Table picture



Caching Translations for speed

- Virtual to Physical translations are cached in a Translation Lookaside Buffer (TLB).

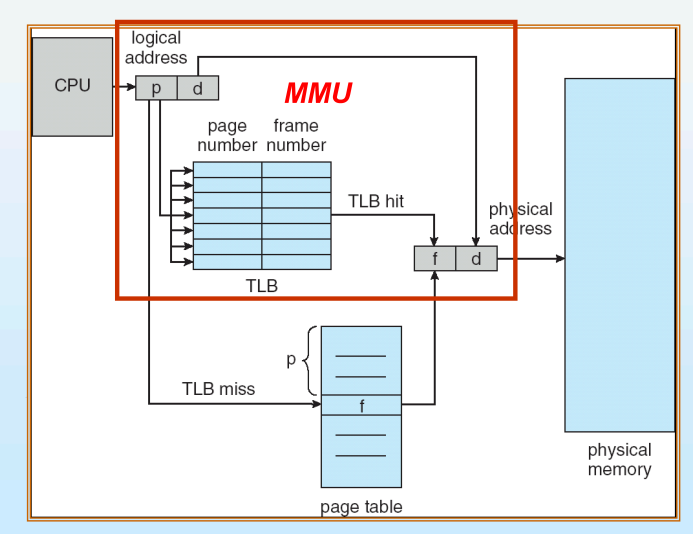


15

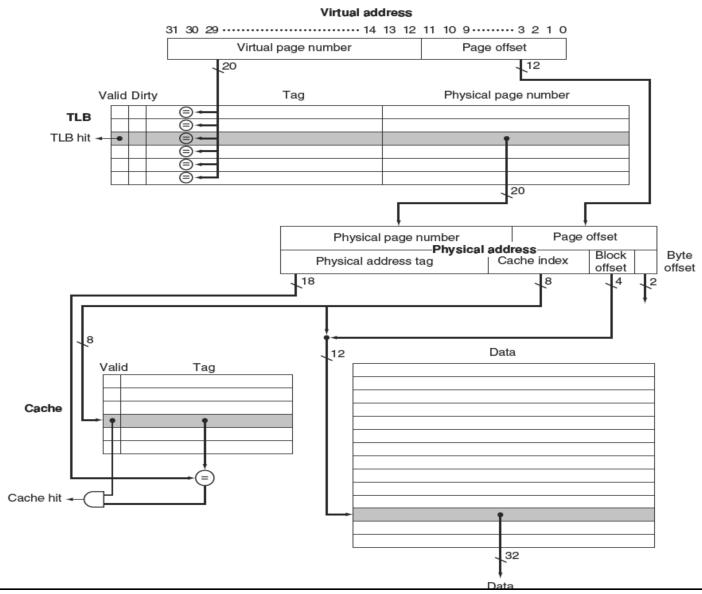
What about a TLB miss?

- If we miss in the TLB, we need to "walk the page table"
  - In MIPS, an exception is raised and software fills the TLB
    - MIPS has "TLB_write" instructions
  - In x86, a "hardware page table walker" fills the TLB
- What if the page is not in memory?
  - This situation is called a **page fault**.
  - The operating system will have to request the page from disk.
  - It will need to select a page to replace.
    - The O/S tries to approximate LRU
    - The replaced page will need to be written back if dirty.

The different parts of the virtual memory system

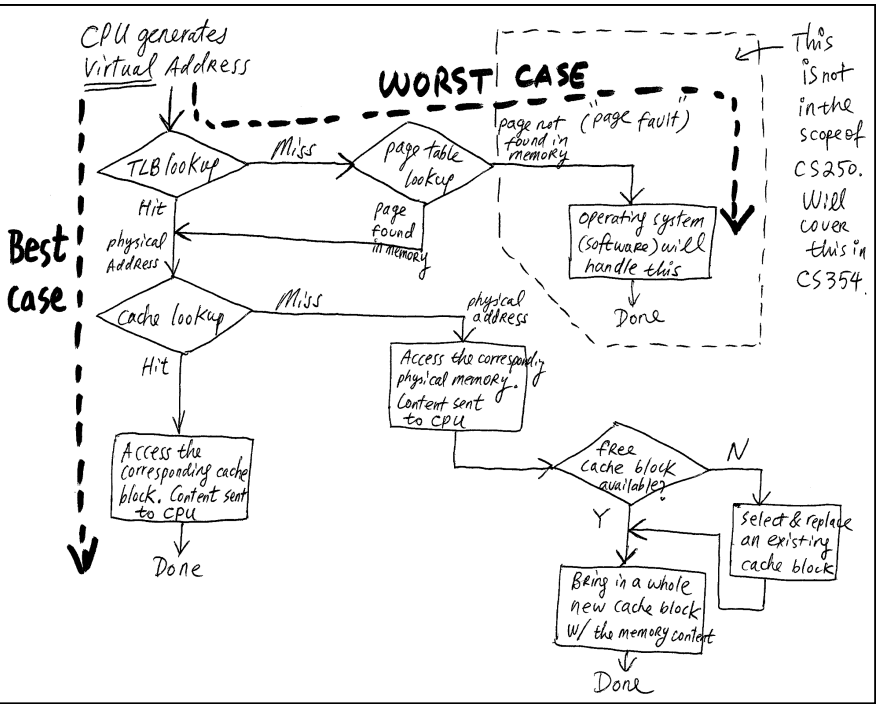


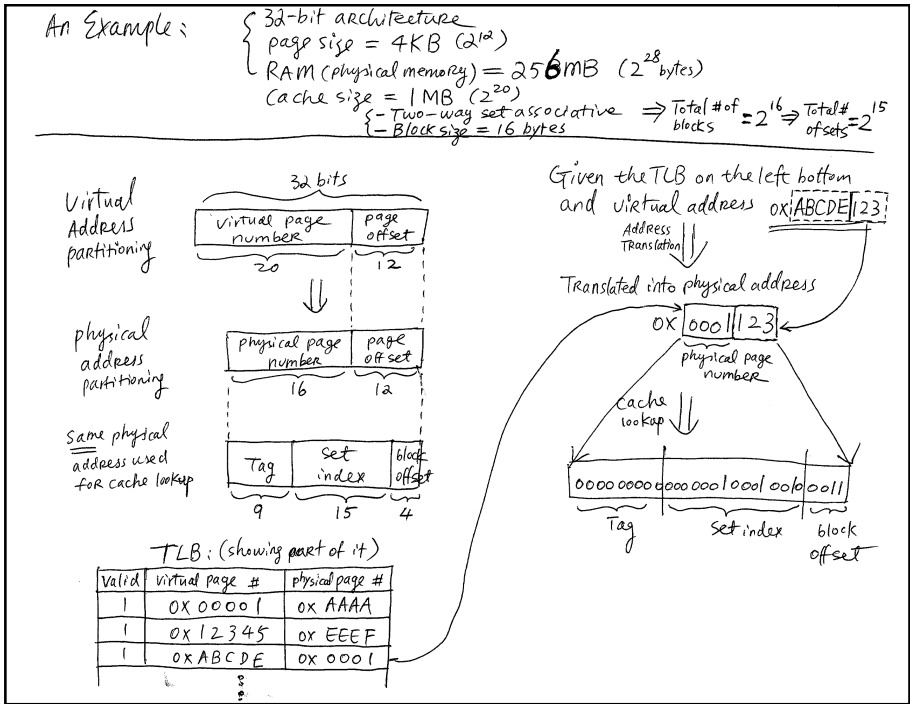
Going from a Virtual Address to a Data Item



Possible combinations

TLB	Page table	Cache	Possible? If so, under what circumstance?
Hit	Hit	Miss	Possible, although the page table is never really checked if TLB hits.
Miss	Hit	Hit	TLB misses, but entry found in page table; after retry, data is found in cache.
Miss	Hit	Miss	TLB misses, but entry found in page table; after retry, data misses in cache.
Miss	Miss	Miss	TLB misses and is followed by a page fault; after retry, data must miss in cache.
Hit	Miss	Miss	Impossible: cannot have a translation in TLB if page is not present in memory.
Hit	Miss	Hit	Impossible: cannot have a translation in TLB if page is not present in memory.
Miss	Miss	Hit	Impossible: data cannot be allowed in cache if the page is not in memory.





Summary

- Virtual memory is pure manna from heaven:
  - It means that we don't have to manage our own memory.
  - It allows different programs to use the same (virtual) addresses.
  - It provides protect between different processes.
  - It allows controlled sharing between processes (albeit somewhat inflexibly).
- The key technique is **indirection**:
  - Yet another classic CS trick you've seen in this class.
  - Many problems can be solved with indirection.
- Caching made a few cameo appearances, too:
  - Virtual memory enables using physical memory as a cache for disk.
  - We used caching (in the form of the Translation Lookaside Buffer) to make Virtual Memory's indirection fast.