**Virtual Memory**

4-14-2003

**Opening Discussion**

- What did we talk about last class?
- Have you seen anything interesting in the news?

**Adding Another Level**

- The idea we started discussing in this chapter was that we wanted the illusion of unlimited fast memory with the knowledge that we couldn't really have it.
- To the end we created a hierarchy where we put a cache (or two or three) between the processor and memory. When memory isn't big enough we go to another level, the disk.
Virtual Memory

- Virtual memory serves multiple roles in a computer. Not only does it allow us to use the disk as the largest (and slowest) level in our memory hierarchy, it also allows more flexibility in the way that programs work.
- Consider two programs running on a computer at once. Now think of the addresses in their compiled code. How can we ensure that they don't conflict.

More on Virtual Memory

- Virtual memory works a lot like a cache while also helping to translate each program working in its own address space to the physical addresses of real memory.
- The “blocks” of VM are typically called pages and when we try to access one that isn’t in memory it is a page fault.
- Relocation does the mapping of virtual addresses to physical addresses.

A Pictorial View
**Design for Low Fault Rates**
- Because the cost of a page fault can be millions of clock cycles, we want to make sure that the rate of page faults is very low.
  - Use large pages to take advantage of fast sequential reads (~64K).
  - Go fully associative.
  - Page faults handled in software so more sophisticated algorithms can be used.
  - Use write-back for write faults and keep a dirty bit.

**Page Tables**
- To keep track of what virtual pages are in what physical pages, we use a page table.
- The machine keeps a page table register to store the address of the beginning of the page table. Every possible virtual page has an entry in the page table.
- Each process gets its own page table. When we switch processes, we switch which page table we are using.

**Reading from a Page Table**
Translation-Lookaside Buffer (TLB)

A potential problem with this scheme is that when we go to an address in memory, we first have to go to memory to translate that address from a virtual address to a physical address. The TLB caches recently used translations.

Early MIPS TLB

Minute Essay

Assume a system with the following characteristics and calculate the CPI. What dominates performance?

- L1 cache miss rate of 5%.
- L2 cache miss rate of 1%. L2 latency of 10 cycles.
- Page fault rate of 0.001% with memory latency of 100 cycles.
- Page fault latency of 1,000,000 cycles.