Slide 1

Administrivia

- Homework 1 graded. Most people did well. More grades coming soon, plus an average so far and a letter grade estimate.
- (Review minute essay from last time.)

Slide 2

Paging — Recap

- Recall basic ideas of paging:
  - Divide address spaces into pages, memory into page frames; allocate memory page (frame) by page (frame).
  - Use page tables (one per process) to keep track of things.
  - Use MMU to translate program (virtual) addresses into memory locations — using page table for current process. Generate "page fault" interrupt if impossible.
- Notice that we get memory protection for free; can also get memory sharing. Related issue — might be nice to have "read-only" bit in page table.
- Still some issues to address — performance, large tables, how to use this for virtual memory.

Slide 3

Performance / Large Address Spaces

- Even with good choice of page size, serious performance implications — page table can still be big, and every memory reference involves page-table access — how to make this feasible/fast?
- Consider several options — compare access time, cost, context-switch time:
  - Keep page table for current process in registers.
  - Keep whole page table in main memory, pointed to by special register.
  - Use multilevel page tables. (More about this later.)
  - Use inverted page tables (one entry per page frame). (More about this later.)
- If page tables are in memory, performance improves with "translation lookaside buffer" (TLB) — special-purpose cache.

Slide 4

Large Address Spaces

- Clearly page tables can be big. How to make this feasible?
- One approach — multilevel page tables. Figure on p. 208.
- Another approach — inverted page tables (one entry per page frame). Figure on p. 214.

Slide 5

Paging and Virtual Memory

- Idea — if we don't have room for all pages of all processes in main memory, keep some on disk ("pretend we have more memory than we really do").
- Or a simpler view: All address spaces live in secondary memory / swap space / backing store, and we "page in" as needed (demand paging).
- Consider an example . . .

Slide 6

Page Tables, Revisited

- What do we need for each entry in a page table?
  - Page frame number.
  - Present/absent bit (was valid/invalid).
  - Protection bit(s).
  - "Modified since last page-in?" bit.
  - "Referenced recently?" bit.
  - "Okay to cache?" bit.
- Goal is to keep this somewhat minimal — mostly data the MMU needs.

  If present/absent bit says "absent", two cases — error and "page not in memory right now" — MMU should generate "page fault" interrupt, let page fault interrupt handler decide.

Slide 7

Minute Essay

- None — sign in.