Address translation and sharing using page tables

- 386 uses 2-level mapping structure
- one page directory page, with 1024 page directory entries (PDEs)
- up to 1024 page table pages, each with 1024 page table entries (PTEs)
- so la has 10 bits of directory index, 10 bits table index, 12 bits offset
- %cr3 register holds physical address of current page directory
- puzzle: what do PDE read/write and user/supervisor flags mean?

- now, access memory twice more for every memory access: really expensive!
- optimization: CPU's TLB caches vpn => ppn mappings
- if you change any part of the page table, you must flush the TLB!
  - by re-loading %cr3 (flushes everything)
  - by executing `invlpg va`

  Is TLB write through? Is it write back? If not, what is it?

- turn on paging by setting CR0_PG bit of %cr0

- Here's how the MMU translates an la to a pa:

```
uint
translate (uint la, bool user, bool write)
{
  uint pde;
  pde = read_mem (%CR3 + 4*(la >> 22));
  access (pde, user, write);
  pte = read_mem ( (pde & 0xfffff000) + 4*((la >> 12) & 0x3ff));
  access (pte, user, write);
  return (pte & 0xfffff000) + (la & 0xfff);
}

// check protection. pxe is a pte or pde.
// user is true if CPL==3
void
access (uint pxe, bool user, bool write)
{
  if (!(pxe & PG_P)
     => page fault -- page not present
  if (!(pxe & PG_U) && user)
     => page fault -- not access for user

  if (write && !(pxe & PG_W)) {
    if (user)
       => page fault -- not writable
    if (%CR0 & CR0_WP)
       => page fault -- not writable
  }
}
```

Can we use paging to limit what memory an app can read/write?

- user can't modify cr3 (requires privilege)
- is that enough?
- could user modify page tables? after all, they are in memory.

Who stores what?

- cr3: physical address
- GDTR: linear address
- GDT descriptor: linear address
- IDTR: linear address
- IDT descriptor: virtual address

Page tables vs. Segmentation

- Good: Pages are easy to allocate (keep a list of available pages and just allocate the first available).
- Good: Pages are easy to swap as everything is same size and pages are usually same size as disk blocks.
- Bad: Page tables can become very large (need one entry for each page-sized unit of virtual memory).