Lab 2 Introduction
Don Porter
cs372h - Spring 2007

Setting up page tables
- Allocate and zero a page for pgdir
  - In i386_vm_init()
- Create entries for used physical addresses
  - See boot_map_segment()
  - Add page tables as needed
  - See boot_pgdir_walk()
- This is tricky:
  - Mapping linear -> physical while using virtual addresses
- Finally, set cr3 register to physical address of pgdir
  - And some cr0 bits also - see i386_vm_init

From segmentation to paging
- x86 cannot turn segmentation off!
  - Workaround: set segment offsets to zero
- How do we resolve virtual addresses in the transition?
  - Temporarily map pgdir[0] -> pgdir[PDX(kernbase)]
  - VA 0xF0000000 = LA 0x00000000
  - But LA 0x0 actually maps to something!
**Helper Functions**

- Macros in inc/mmu.h to access pte bits/addresses
- ROUNDUP() - can be used to page-align addresses

**Setting up memory layout**

- See inc/memlayout.h for diagram of virtual address space
- Magic of Paging:
  - Map same physical address to multiple virtual addresses with different permissions
  - Expose kernel data to users Read-only!
  - Protection enforced by hardware
- Question: Why can’t the kernel protect itself from buggy device drivers the same way
  - Check out “Mondrix” for research interest

**Page management**

- Array of struct pages mirrors physical memory
  - Stores reference count and linked list pointer
  - Why a reference count?
- Because they are arranged in contiguous memory, they can be used to calculate physical address

**Translation Lookaside Buffer**

- A hardware managed cache of page table entries
  - On a miss, hardware automatically walks the page tables
- Thus, when you change a page table entry, you must update TLB
  - Cannot edit entries, only drop them and force hardware to re-read them
- This includes adding an entry!
  - Caches invalid entries
More helpers

- inc/queue.h has macros for list manipulation
  - LIST_INIT(), LIST_INSERT_HEAD(), etc.
- kern/pmap.h has several macros for address translation
  - pa2page, page2kva, etc
  - KADDR, PADDR, etc

Lots of opportunity for error!

- Keep solutions as simple and clear as possible
- Make sure they follow descriptions exactly
- Pay close attention to provided test cases