Operating Systems

Lab. Class

Project #4

Project Plan

- 5 projects
  0. Install xv6
  1. System call
  2. Scheduling
  3. Memory
  4. Virtual Memory
  5. Concurrency

- Individual projects

Page Table Hardware in Xv6

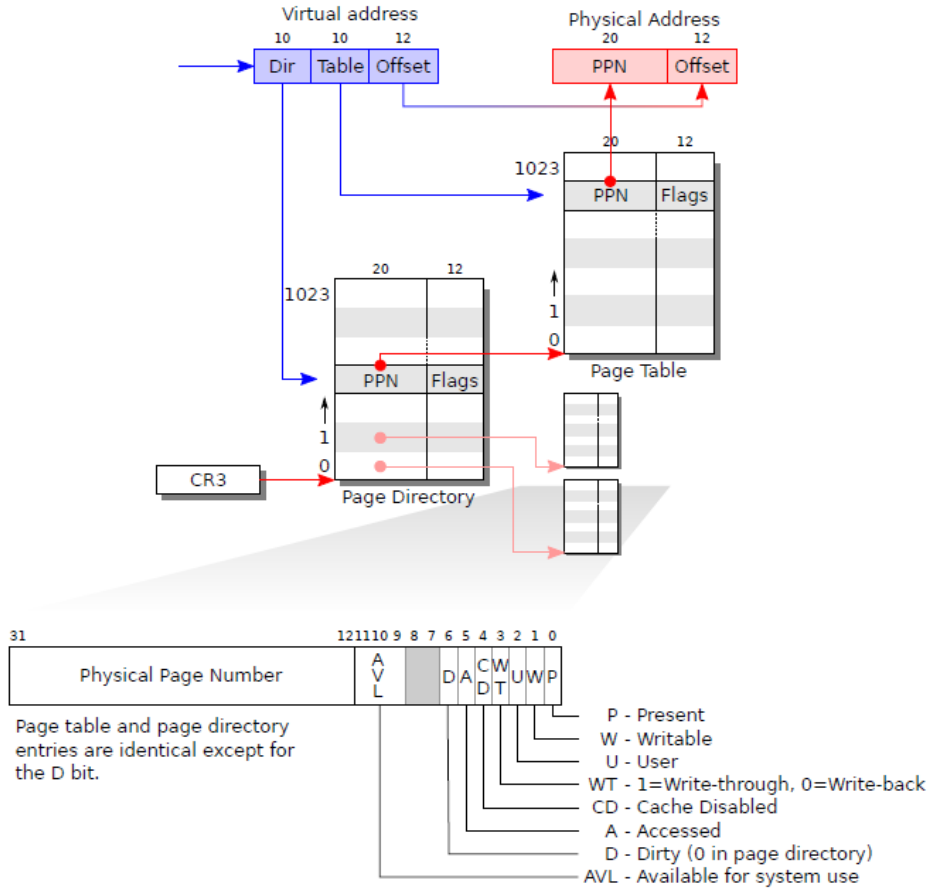
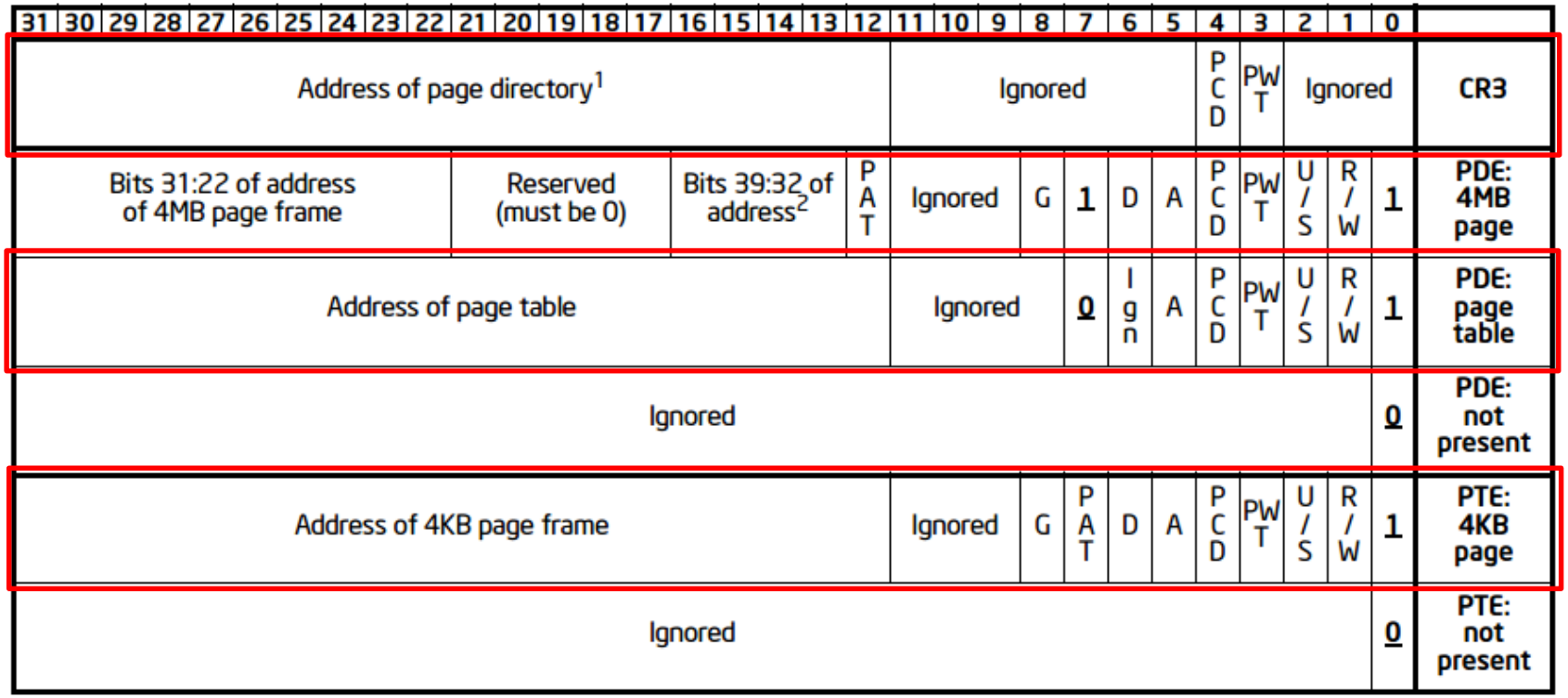


Figure 2-1. x86 page table hardware.

Formats of Paging Entries in Intel x86



31–12	11–9	8	7	6	5	4	3	2	1	0	
Address of page directory[1]	Ignored					PCD	PWT	Ignored			CR3
Bits 31:22 of address of 4MB page frame; Reserved (must be 0); Bits 39:32 of address[2]; PAT	Ignored	G	1	D	A	PCD	PWT	U/S	R/W	1	PDE: 4MB page
Address of page table	Ignored		0	Ign	A	PCD	PWT	U/S	R/W	1	PDE: page table
Ignored										0	PDE: not present
Address of 4KB page frame	Ignored	G	PAT	D	A	PCD	PWT	U/S	R/W	1	PTE: 4KB page
Ignored										0	PTE: not present

Figure 4-4. Formats of CR3 and Paging-Structure Entries with 32-Bit Paging

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

Control Registers in Intel x86

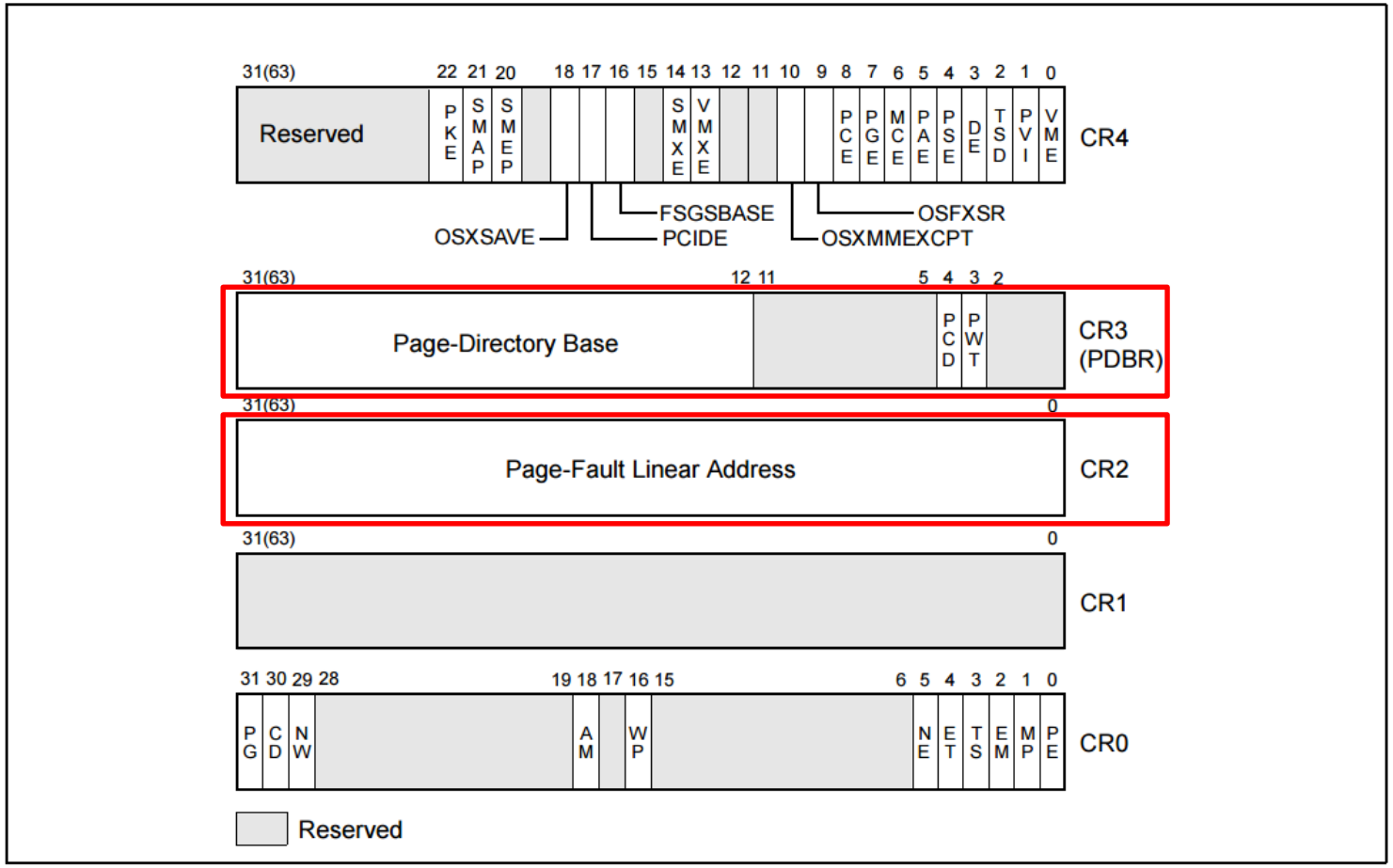


Figure 2-7. Control Registers

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

Page Fault Exception in Intel x86

- Conditions
  1. There is no translation for the linear address
  2. There is a translation for the linear address, but its access rights do not permit the access

- CR2 stores the linear address that caused a page fault

- Processor triggers interrupt 14 (page fault)

Page Fault Error Code in Intel x86

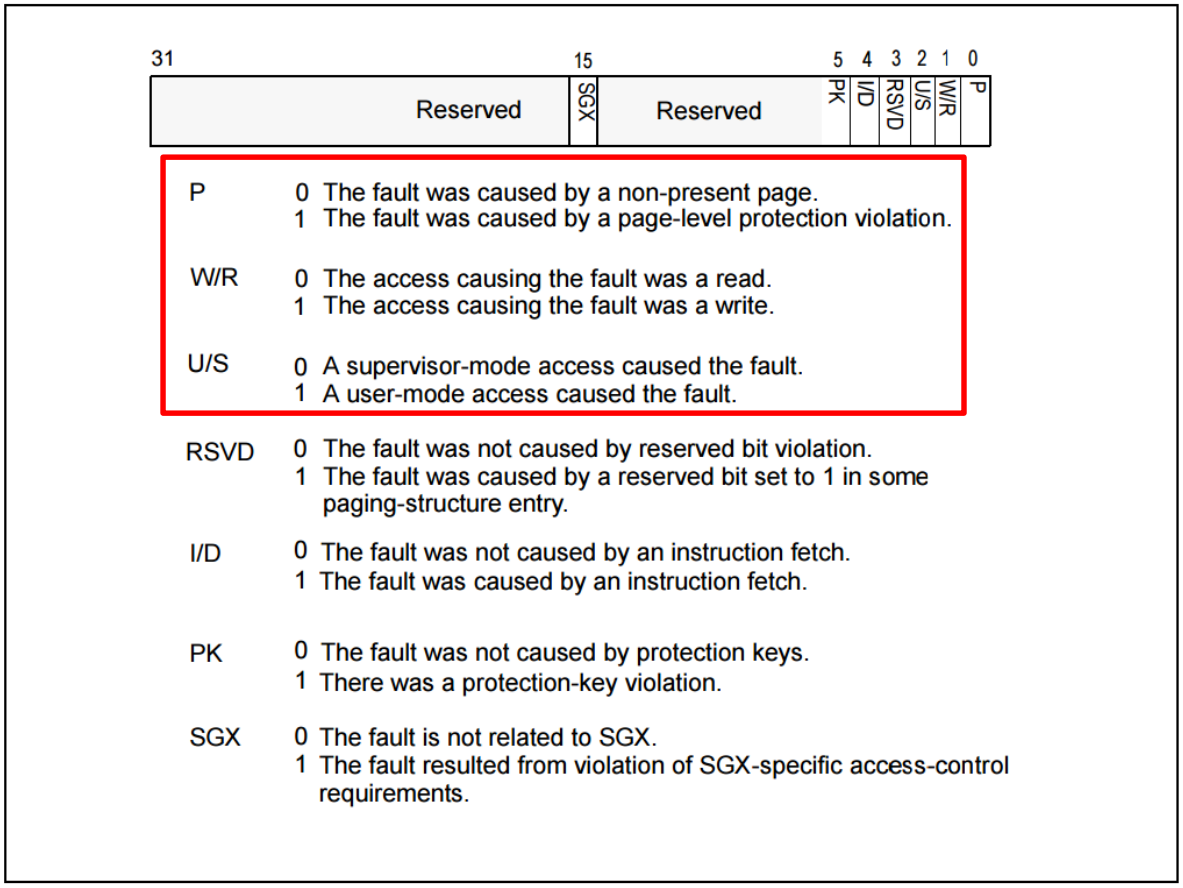


Figure 4-12. Page-Fault Error Code

Intel® 64 and IA-32 Architectures Software Developer's Manual V3

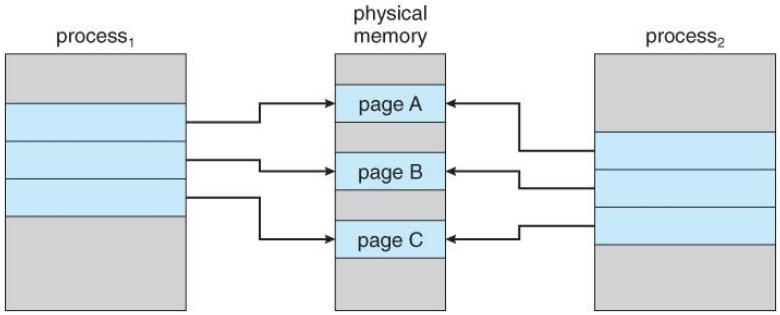
Exception & Interrupt Handling in Xv6

- Follows Intel x86 architecture

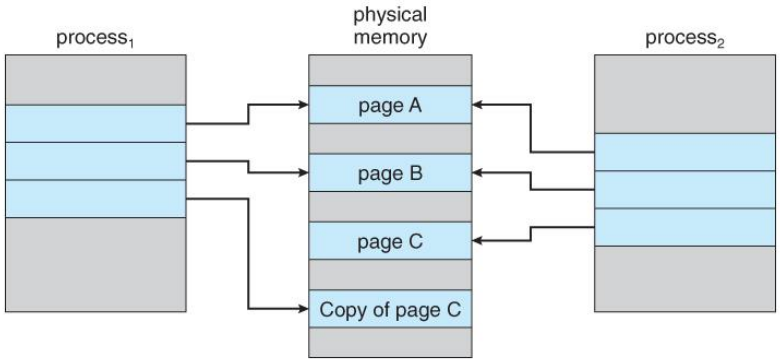
- Procedures
  1. Assign certain interrupt to interrupt descriptor table (IDT)
  2. All interrupts jump to alltraps() and build trap frame
  3. Handle each interrupt depending on its trap number

Copy-on-Write

- When a process forks
  - Create shared mappings to the same page frames in physical page
  - Shared pages are protected as read-only
- When data is written to shared pages
  - Protection fault is generated
  - OS allocates new space in physical memory and directs the write to it
- Reference counter for physical pages is needed



Before process 1 modifies page C



After process 1 modifies page C

https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

PA #4 - Copy-on-Write

- Implement copy-on-write on xv6

- Implementation details
  - Setting up page-fault handler
  - Implementing page-fault handler
  - Modify copyuvm() in vm.c from copy version to duplicate version
  - Managing pages using reference counter

- References
  - Reading cr2 register & loading cr3 register are implemented in x86.h
    - rcr2(), lcr3()
  - Xv6 commentary

PA #4 Template Code

- `git clone https://github.com/jinsoox/xv6-skku.git -b pa4`
- Modifications
  - freemem() system call
    - Return the number of free pages in kmem.freelist

Project Submission Procedure

- http://sys.skku.edu
  - Submit a tarball file made from "**make tarball**"

- Due date
  - 2017-05-14 23:59

- Since 6nd submission, -5% penalty of the project score
  - Up to 5 submissions are free to accept
- Every one day delay, -25% penalty of the project score
  - You can use up to 5 *slip* days