Operating System

Project #2
16.10.10
Project Plan

• 5 projects
  – Install Xv6
  – System call + scheduling
  – Virtual memory (stack growth + COW)
  – Thread-support
  – Concurrency

• Single-handed project
Address Translation in Intel x86

Figure 3-1. Segmentation and Paging
## Formats of Paging Entries in Intel x86

<table>
<thead>
<tr>
<th>Address of page directory¹</th>
<th>Ignored</th>
<th>P</th>
<th>C</th>
<th>D</th>
<th>Ignored</th>
<th>P</th>
<th>C</th>
<th>W</th>
<th>T</th>
<th>Ignored</th>
<th>CR3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits 31:22 of address of 2MB page frame</td>
<td>Reserved (must be 0)</td>
<td>Bits 39:32 of address²</td>
<td>P</td>
<td>A</td>
<td>T</td>
<td>Ignored</td>
<td>G</td>
<td>1</td>
<td>D</td>
<td>A</td>
<td>CR3: 4MB page</td>
</tr>
<tr>
<td>Address of page table</td>
<td>Ignored</td>
<td>Q</td>
<td>I</td>
<td>gn</td>
<td>A</td>
<td>Ignored</td>
<td>P</td>
<td>C</td>
<td>W</td>
<td>T</td>
<td>Ignored</td>
</tr>
<tr>
<td>Ignored</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PDE: not present</td>
</tr>
<tr>
<td>Address of 4KB page frame</td>
<td>Ignored</td>
<td>G</td>
<td>P</td>
<td>A</td>
<td>D</td>
<td>A</td>
<td>CR3: 4KB page</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ignored</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PTE: not present</td>
</tr>
</tbody>
</table>

---

¹ Address of page directory
² Bits 39:32 of address

**Figure 4-4. Formats of CR3 and Paging-Structure Entries with 32-Bit Paging**
Page Fault in Intel x86

- CR2 stores linear address that caused page fault
- Processor triggers interrupt #14 (page fault)
Control Registers in Intel x86

Figure 2-7. Control Registers
Exception & Interrupt Handling in xv6

- Follows Intel x86 architecture

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

- Tip
  - Get page fault address with rcr2()
Virtual Address Space

Figure 2-2. Layout of a virtual address space and the physical address space.
Process User Stack in xv6

- 1 stack page & 1 guard page

Figure 2-3. Memory layout of a user process with its initial stack.
Stack Growth in xv6

- 4 stack page & 1 guard page
- Stack grows when current stack is full
- Stack pointer can move up to 32 bytes (pushal instruction)
  - trapframe->esp
- When stack pointer reaches guard page, stack overflow occurs and process is killed
Project #2 – Stack Growth

• Implement stack growth in xv6

• Submit a tar.gz file

• Send email to T.A
  – [SSE3044]Project#2-YOURID-YOURNAME
    • ex) [SSE3044]Project#2-2016710580-이규선
  – Email address: lgs0409@naver.com
  – Wrong title is not allowed

• Due date
  – 2016-10-30(Sun) PM 23:59
  – Get no point for late submission