Project 5. Page Fault Handler

2019.5.15 (Wed.)

TAs
김종석(ks77sj@gmail.com) /
최동규(gmj03003@gmail.com)
Project Plan

- Total 7 projects
  0) Starting xv6 operating system (5%)
  1) System call (10%)
  2) Thread (20%)
  3) Synchronization (15%)
  4) Scheduling 1(15%)
  5) Page fault handler (15%)
  6) Copy on Write (20%)
What is VM (Virtual memory)?
Address Translation in Intel x86

- Why is the Offset index of PA 20?
  - PAGESIZE 4096 bytes
- Why is the PPN index of PA 20?
  - Memory(4G) / PAGESIZE
A virtual address has a three-part structure as follows:

<table>
<thead>
<tr>
<th>Directory Index</th>
<th>Page Table Index</th>
<th>Offset within Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDX(va)</td>
<td>PTX(va)</td>
<td></td>
</tr>
</tbody>
</table>

// page directory index
#define PDX(va) (((uint)(va) >> PDXSHIFT) & 0x3FF)

// page table index
#define PTX(va) (((uint)(va) >> PTXSHIFT) & 0x3FF)

// construct virtual address from indexes and offset
#define PGADDR(d, t, o) (((uint)(d) << PDXSHIFT | (t) << PTXSHIFT | (o))

// Page directory and page table constants.
#define NFDETRIES 1024  // # directory entries per page directory
#define NFPTENTRIES 1624 // # PTEs per page table
#define PGSIZE 4096    // bytes mapped by a page
#define PGSIZELEN 12   // log2(PGSIZE)
#define PTXSHIFT 12    // offset of PTX in a linear address
#define PDXSHIFT 22    // offset of PDX in a linear address
#define PGROUNDUP(sz) (((sz)+PGSIZE-1) & -(PGSIZE-1))
#define PGROUNDOWN(a) (((a)) & -(PGSIZE-1))

// Page table/directory entry flags.
#define PTE_P 0x001  // Present
#define PTE_W 0x002  // Writeable
#define PTE_U 0x004  // User
#define PTE_PWT 0x008 // Write-Through
#define PTE_PCD 0x010 // Cache-Disable
#define PTE_A 0x020  // Accessed
#define PTE_D 0x040  // Dirty
#define PTE_PS 0x060 // Page Size
#define PTE_MZ 0x100 // Bits must be zero

// Address in page table or page directory entry
#define PTE_ADDR(pte) ((uint)(pte) & ~0xFFF)
#define PTE_FLAGS(pte) ((uint)(pte) & ~0xFFF)
Page Fault Exception in Intel x86

• Conditions
  • There is no translation for the linear address
  • 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 handler in xv6

traps.h

// x86 trap and interrupt constants.

// Processor-defined:
#define T_DIVIDE 0 // divide error
#define T_DEBUG 1 // debug exception
#define T_NMI 2 // non-maskable interrupt
#define T_BRKPT 3 // breakpoint
#define T_OFLOW 4 // overflow
#define TBOUND 5 // bounds check
#define T_ILLOP 6 // illegal opcode
#define T_DEVICE 7 // device not available
#define T_DIVFLT 8 // double fault

// #define T_COPROC 9 // reserved (not used since 486)
#define T_SSE 10 // invalid task switch segment
#define T_SYSENTER 11 // segment not present
#define T_SYSEXIT 12 // stack exception
#define T_GPF 13 // general protection fault
#define T_PGFLT 14 // page fault

// #define T_SYSCALL 15 // reserved
#define T_FPREL 16 // floating point error
#define T_ALIGN 17 // alignment check
#define T_MCHK 18 // machine check
#define T_SSIMPLE 19 // SIMD floating point error

// These are arbitrarily chosen, but with care not to overlap
// processor defined exceptions or interrupt vectors.
#define T_SYSCALL 64 // system call
#define T_DEFAULT 500 // catchall
#define T_IRQ 32 // IRQ 0 corresponds to int T_IRQ
#define IRQ_TIMER 0
#define IRQ_KBD 1
#define IRQ_COM1 4
#define IRQ_IDE 14
#define IRQ_SDRM 19
#define IRQ_SPURIOUS 31

// trap.c

if (page fault occurs, “trapno” of trapframe automatically filled with T_PGFLT and call trap function in trap.c

- You have to make your “own” page fault handler

- Currently, implemented as below...
  - rcr2() -> page fault address

// In user space, assume process misbehaved.
cprintf("pid %s: trap %d err %d on cpu %d \n",
  myproc()->pid, myproc()->name, tf->trapno,
  tf->err, cpuid(), tf->eip, rcr2());

myproc()->killed = 1;
User Address Space in xv6

- 1 stack page & 1 guard page

Figure 2-3. Memory layout of a user process with its initial stack.
Project 5. Stack growth

- Initial size of stack
  - Prepare 1 page initially
  - Can be grow up to 4 pages

- Growth of stack
  - tf->esp can move upto 32bytes
  - New page should be allocated to this process if current stack is full

- When stack pointer reaches guard page or a process accesses invalid address, kill that process
Project 5. Stack growth

• Print out “Invalid access” when approaching abnormal address
• Print out “Allocate page” if page-fault handler is executed normally
Template Code

• wget http://csl.skku.edu/uploads/SWE3004S19/xv6-skku.tar.gz

• Modifications
  • halt system call
  • Halt xv6 program
  • make tarball
    • Compress your source codes into one .tar.gz file for submission
    • You should enter your ID & project no. on Makefile
  • CPUS=1
Test Case

• Erase `-Werror` in Makefile

---

```makefile
CC = $(TOOLPREFIX)gcc
AS = $(TOOLPREFIX)gas
LD = $(TOOLPREFIX)ld
OBJCOPY = $(TOOLPREFIX)objcopy
OBJDUMP = $(TOOLPREFIX)objdump
CFLAGS = -fno-pic -static -fno-builtin -fno-strict-aliasing -O2 -Wall -MD -g -m32 -Werror -fno-omit-frame-pointer
#CFLAGS = -fno-pic -static -fno-builtin -fno-strict-aliasing -fvar-tracking -fvar-tracking-assignments -O0 -g -Wall -MD -gdebug-2 -m32 -Werror -fno-omit-frame-pointer
CFLAGS += $(shell $(CC) -fno-stack-protector -E -x c /dev/null >/dev/null 2>&1 && echo -fno-stack-protector)
ASFLAGS = -m32 -gdebug-2 -Wa,divide
# FreeBSD ld wants `elf_386_fbsd`
LDFLAGS += -m $(shell $(LD) -V | grep elf_386 2>/dev/null | head -n 1)
```

• Correct Result

```
Allocate page
Allocate page
Allocate page
Invalid access
```
Submission

• You need to submit a document.

• Just write how you implemented your code.

• You can use English or Korean.
Submission

• Send your code file (xv6-project-5-studentID.tar.gz) and document file to ks77sj@gmail.com

• Please send a mail with title including [SWE3004-P5]
  • Ex) [SWE3004-P5] 2014111111-project5

• PLEASE DO NOT COPY
  • YOU WILL GET F GRADE IF YOU COPIED

• Due date: 5/22(Wed.), 23:59:59 PM
  • Delays are allowed only one week from the deadline. And there will be up to -40% penalty.
Questions

• If you have questions, please email to TA
  • You can't ask questions on deadline day

• You can also visit #85533. Please email TA before visiting