COMP3231/9201/3891/9283 Operating Systems 2021/T1

UNSW

Administration
- Notices
- Course Outline
- UNSW Timetable
- Consultations
- Survey Results!!

Work
- Lectures
- Tutorials

Support
- Ed Forums
- Wiki

Assignments
- Submission Guide
- Assignment 0 Warm-up

Resources
OS/161
- General
- Man Pages
- Sys161 Pages

C coding
- Info Sheet

Debugging
- Learn Debugging

General
- "Hardware" Guide
- R3000 Reference Manual
- Intro. to Prog. Threads

Previous years
- 2020 T2
- 2020 T1
- 2019 T1
- 2018 S1
- 2017 S1
- 2016 S1
- 2015 S1
- 2014 S1
- 2013 S1
- 2012 S1
- 2011 S1
- 2010 S1
- 2009 S1
- 2008 S1
- 2007 S1
- 2006 S1
- 2005 S2
- 2005 S1
- 2004 S2
- 2004 S1

Tutorial Week 9

Questions

Assignment 3 Exercises

By now you should be familiar enough with navigating the kernel source that you can find the answers to the questions below by yourself (Hint: use a source code browser or the *grep* utility). You may also find the MIPS r3000 reference useful.

1. What is the difference between the different MIPS address space segments? What is the use of each segment?
2. What functions exist to help you manage the TLB? Describe their use. (Hint: look in `kern/arch/mips/include/tlb.h`)
3. What macros are used to convert from a physical address to a kernel virtual address?
4. What address should the initial user stack pointer be?
5. What are the `entryhi` and `entrylo` co-processor registers? Describe their contents.
6. What do the `as_*` functions do? Why do we need `as_prepare_load()` and `as_complete_load()`?
7. What does `vm_fault()` do? When is it called?
8. Assuming a 2-level hierarchical page table (4k pages), show for the following virtual addresses:
    1. The page number and offset;
    2. the translated address (after any page allocation); and
    3. the contents of the page table after the TLB miss.

Staff
- Kevin Elphinstone (LiC)
- TBD (Admin)

Grievances
- Student Reps

The page table is initially empty, with no L2 pages. You may assume that the allocator returns frames in order, so that the first frame allocated is frame 1, then frames 2, 3, 4, etc.

- 0x100008
- 0x101008
- 0x1000f0
- 0x41000
- 0x41b00
- 0x410000

C bitwise operations refresher

Given the following constants from the source code:

```
#define TLBHI_VPAGE   0xfffff000
#define TLBLO_PPAGE   0xfffff000
#define TLBLO_NOCACHE 0x00000800
#define TLBLO_DIRTY   0x00000400
#define TLBLO_VALID   0x00000200
```

And a 32-bit page table entry and 32-bit virtual address is defined as

```
uint32_t pte;
vaddr_t vaddr;
```

9. What C expressions would you use to set or reset the valid bit in a page table entry?
10. What C expression would you use to test if the valid bit is set?
11. How would you extract the 12-bit offset (bits 0 – 11) from the virtual address?
12. How would you convert the 10 most significant bits (22-31) of a virtual address into an index?
13. How would you convert the next 10 most significant bits (12-21) into an index?
14. How would you round down a virtual address to the base of the page?

Page last modified: 4:49pm on Saturday, 23rd of May, 2020

Print Version