Operating Systems Quiz 2 ECE344, Winter 2022

Duration: 1 hour

Examiner: D. Yuan

Instructions

Examination Aids: This is an open book exam.

All questions have been provided in this exam booklet. You need to provide your answers in Quercus.

If any of the questions appear unclear or ambiguous to you, then make any assumptions you need, state them and answer the question that way. If you believe there is an error, state what the error is, fix it, and respond as if fixed.

Be brief and specific as possible. Most questions should only need 1 sentence to answer. Time is tight, and you can't afford to write long answers. Marks will be deducted for incorrect statements in an answer.

You'll get 20% of the mark for each question where you leave the answer empty.

Work independently.

Question 1 (4 mark): TLB structure (Robin)
We discussed a number of bits a TLB entry could have, including:

- Virtual Page Number (VPN)
- Page Frame Number (PFN)
- Modified (dirty) bit
- Reference bit
- Valid bit
- Process ID bits
- Protection bits

For each of them, state whether it is "must have" (i.e., necessary for a correctly functional TLB) and which are "nice to have" (i.e., would ease the OS's optimization on performance, but not necessary for functional correctness)? Briefly explain why for each answer with 1 sentence.

VPN: must have. TLB is a fully associate cache, needs VPN as the tag for comparison.
PFN: must have. Otherwise it cannot know what is the physical address.
Modified bit: nice to have. Without it, the OS just needs to copy the page to disk on eviction.
Reference bit: nice to have. Only used to make a more informed replacement policy.
Valid bit: must have. Without it, the MMU doesn't know whether the mapping is valid.
PID bits: nice to have. Without it the OS can simply flush the TLB on context switch.
Protection bits: must have. Otherwise the MMU cannot detect an access violation.

Marking:

- 0.5 each question
- 0.5 bonus if all answers are correct.

Question 2 (9 mark): Translation (Rui: first 4, (e): Zhihao, (f): Jack)
We have a computer that has the following specifications:

- 12 bits virtual addresses (VA)
- Each page has 16 bytes
- The RAM has 256 bytes
- The machine uses software managed TLB (trap into the OS on a TLB fault)

Answer the following questions:

(a) (0.5 mark) What's the size of the virtual address space?

2^12 = 4KB.

(b) (0.5 mark) How many bits in the VA are used as the virtual page number (VPN)?

8 bits.

(c) (0.5 mark) How many page frames does the RAM have?

16.

(d) (0.5 mark) Now you're to implement an OS on this machine. You quickly concluded that a single-level page table is infeasible on this machine. Why?

A single level page table requires 2^8 = 256 entries. If each entry uses 1 byte, then it would use up the entire RAM just to store the page table.

(e) (5 mark) So you decide to use a two-level page table design, and divide the VPN by half: the higher half of the bits in the VPN are used to index the top-level page table, whereas the lower half is used to index the second-level. Each page table entry is stored in a byte, and a page table entry has the following format:

Bit	0	1	2 - 3	4 - 7
Meaning	Valid	Modified	Protection	Page Frame Number

In other words, the highest bit (bit 0) is the valid bit (1 means valid, 0 invalid); then a modified bit (1 means modified, 0 not), two protection bits (00: no access, 01: read only, 10: write only, 11: read and write), and 4 bits for the page frame number (PFN). **Both of the page table levels have the same entry format (and yes, it's correct, the top-level page table's entry also has this format).**

The content of the RAM is as follows:

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	ac	01	ff	fa	ff	b2	82	9a	4c	33	29	5a	b6	c8	e2	ff
1	00	00	b5	b3	b2	f3	f4	f6	ff	00	00	ff	ff	ff	ff	ff
2	ff	fb	63	00	42	00	00	32	00	55	00	32	71	00	fc	00
3	00	00	32	62	a3	84	d3	d5	78	a2	77	a2	87	fa	fb	ff
4	42	00	00	32	00	55	00	32	fa	fa	fa	fa	00	ff	ff	ff
5	a3	8d	82	d4	fa	ff	d3	fa	bb	00	40	43	00	27	ee	dd
6	d2	34	a1	32	53	00	ff	fa	f2	ff	fc	fa	ff	ff	32	ff
7	d3	d5	78	a2	77	53	00	00	ff	ff	ff	ff	ff	ff	ff	ff
8	d3	29	78	b3	dd	fa	cc	ff	ff	b0	fc	43	ff	fa	fb	ff
9	39	44	00	72	39	00	62	77	00	23	00	dd	ff	ff	ff	ff
a	00	00	00	00	ff	ba	b3	bb	b3	b0	b3	73	00	73	73	74
b	00	00	00	7a	7c	00	40	43	00	27	34	00	00	00	00	00
c	48	60	00	00	23	f4	00	18	29	3a	00	b5	b8	00	00	25
d	00	40	43	00	27	a1	32	53	00	00	ff	ff	08	83	a2	c2
e	d3	2a	d3	a2	32	53	00	ff	fa	f2	ff	ff	ff	ff	ff	ff
f	00	b0	39	44	00	72	39	00	62	77	00	23	00	56	33	00

Each entry is the content of a byte. There are 256 bytes. The (physical) address of any byte in the table would be 0xRowNumberColumnNumber. So for example, the address of the highlighted byte at row index 2, column index 0 is 0x20, which has the content 0xff.

A process makes the following sequence of memory access instructions. The physical address of the beginning of the top-level page table for this process is 0x20 (highlighted). For each instruction, explain the result of the instruction execution, including:

- The physical address after the translation
- If the instruction is a load, the content that's being read
- If **any** memory content gets modified by the instruction, specify the modified data and the address

The first operand of load/store is the virtual address, whereas the second operand is the destination register (for load) or the data to be written (for store). Assume each load/store is reading/writing a single byte.

(i) load 0x010, r1
Physical address: 0x00, content: 0xac, no change to memory state

(ii) store 0x014, 3
PA: 0x04, after the modification, the content at 0x04 becomes 3 (0.5 mark)

Also the byte at address 0xf1 is changed from 0xb0 to 0xf0 (to set the modified bit) (0.5 mark)

(iii) load 0xeb8, r2
Physical address: 0x58, content 0xbb, no change to memory state.

(iv) store 0xeca, 0
Physical address: 0x8a, content at PA0x8a after the modification: 0 (0.5 mark)
Also the byte at address 0xcc is changed from 0xb8 to 0xf8 (0.5 mark)

(v) store 0xe5c, 4
Physical address: 0x4c, content at PA 0x4c after the modification: 4 (1 mark)

Note: No change to the address 0xc5 (2nd level page table entry), because the modified bit is already set.

(f) (2 mark) Now you decide to change the page table format in your OS to, say, 3 levels. Can you make the change without changing the hardware (i.e., TLB structure)?

Yes, because it's software managed TLB. The TLB format has nothing to do with the OS page table structure.

Marking: 1 mark if the student does not mention or hint at the use of software-managed TLB.

Question 3 (4 mark): Replacement (2 mark each) (Rishi, Haiqi)

(a) Can you describe a scenario where the Not Recently Used (NRU) algorithm will pick a page that is not the Least Recently Used?

If the last accesses to page X and Y occurs in the same epoch, then both pages will have the same "age" counter value in NRU, so any could be chosen as the victim page under NRU.

(b) Can you describe a scenario where the LRU clock algorithm will pick a page that is not the same as LRU?

Say page X is ordered before Y in the logical clock, but the last access of X is later than Y. During the clock scan, the reference bits for both pages are reset. And since then none of them get accessed again. The next time the clock algorithm will choose page X to evict before Y, even when X's last accessed was later than Y.

Question 4 (3 mark): **Null pointer dereference.** (Kia, Ding)
When a process dereferences a NULL pointer (address 0x0), the process will receive a segmentation fault from the OS. Please describe every step taken by the x86 computer system (both the OS and hardware) that implement this feature.

For each step, specify whether it's done by the MMU (hardware), the OS, or the process.

The key is to not map the page at virtual page number 0. (**1.5 mark**)
1: MMU looks up the VPN on a load/store, which is 0
2. TLB miss (b/c it's not mapped)
3: MMU looks up the page table
4. Page fault, trap into the OS
5: OS finds out the address is 0x0, delivers the segmentation fault to the process

Marking:

- If only describing the general translation procedure, without knowing that page number 0 is not mapped, at most 1.5 marks (any mistake madein the translation procedure -.5 mark)