15-213: Introduction to Computer Systems
Written Assignment #8

This written homework covers virtual memory.

Directions

Complete the question(s) on the following pages with single paragraph answers. These questions are not meant to be particularly long! Once you are done, submit this assignment on Canvas.

Below is an example question and answer.

Q: Please describe benefits of two's-complement signed integers versus other approaches.

A: Other representations of signed integers (ones-complement and sign-and-magnitude) have two representations of zero (+0 and -0), which makes testing for a zero result more difficult. Also, addition and subtraction of two's complement signed numbers are done exactly the same as addition and subtraction of unsigned numbers (with wraparound on overflow), which means a CPU can use the same hardware and machine instructions for both.

Grading

Each assignment will be graded in two parts:

1. Does this work indicate any effort? (e.g. it's not copied from a homework for another class or from the book)
2. Three peers will provide short, constructive feedback.

Due Date

This assignment is due on March 29th, 2023 by 11:59 pm Pittsburgh time (currently UTC-4). Remember to convert this time to the timezone you currently reside in.

Question #1

Virtual addressing is one of the most innovative ideas in computer science.

1) How does virtual memory keep address spaces between processes separate
2) How can it share information between the address space of different processes?
3) Name one example of where having shared memory between processes is useful.

1) Virtual memory keeps address spaces separate by ensuring that each process has its own page table that maps virtual addresses to physical addresses. This is done inside the operating system.
2) Multiple processes can access the same virtual address simultaneously--this is possible because the virtual address within each process would map to a different physical page in physical memory. The OS can share information between address spaces by mapping virtual pages in each of the page tables to the same physical page of memory.
3) This is most useful for code libraries--if multiple processes use the same code library, any process that needs the library can map its shared library segment to a series of shared read-only pages. This way, only one copy of a shared library module needs to exist between all running processes.

Question #2

1) Discuss one advantage of having more levels in a multi-level page table, and one advantage of having fewer levels in a multi-level page table.
2) What is a TLB, and how would using a TLB make multi-level page table lookup more efficient?

- **More Levels**: Since virtual addresses can contain data corresponding to multiple page tables at once, more levels would imply having more bits in the address available for virtual memory. In addition, each entry or "node" of the table will also have a smaller size, which means that overall, there will be less memory used on the whole page table.
- **Fewer Levels**: Fewer levels on a page table means a faster address translation time and fewer memory accesses (decreased time cost). The operating system code/interface may also be simpler, and intermediate entries on the page table would not take up as much space in the cache.
- A Translation Lookaside Buffer (TLB) is a set-associative cache that has the page table entries for some subset of pages at different levels, which helps to speed up address translation. It's especially useful for programs with good spatial and temporal locality; cached entries are likely to stay in the TLB for a while because pages are so large that many different addresses can map to the same page at once.

Question #3

Given a VM system with the following constraints:

- The memory is byte addressable
- Virtual addresses are **12 bits** wide
- Physical addresses are **10 bits** wide
- The page size is 64 bytes

A section of the page table is given below:

Virtual Page Number	Physical Page Number	Valid
00	-	0
01	2	1
02	-	0
03	7	0
04	4	1
05	-	0
06	5	1

Answer the following questions based on the inputs above:

a. How many bits are needed to encode the virtual page offset?
What is the relationship between the virtual and physical page offsets?

b. For the given 2 addresses, answer the following questions:

1. 0x18A
2. 0x19C

i. What is the virtual page offset
ii. What is the virtual page number
iii. Will this translation result in a page hit or a page fault? If it is a page hit, what is the equivalent physical page number?
iv. If the translation in (iii)resulted in a page hit, what is the complete physical address?

Ans:
A).
Page size = 64 bytes = 2^6
Out of the 12 bits for the virtual address, 6 bits are for the Virtual Page Offset VPO.

The VPO & the PPO are the exact same value (6 bits each).

B).
Given Virtual address – 12 bits

0x18A = 000110001010

1.
001010 – VPO (6 bits)

Virtual Page Offset - A

2.
000110 – VPN (12-6=6 bits)

Virtual Page number - 6

3.
It is a Page hit.(Valid VPN & valid bit set)

Physical page number - 5

4.

Since it was a hit, final Physical address – 0101001010(0x14A)

C).
Given Virtual address – 12 bits

0x19C = 000110011100

1.
011100 – VPO (6 bits)

Virtual Page Offset – 1C

2.
000110 – VPN (12-6=6 bits)

Virtual Page number - 6

3.
It is a Page hit.(Valid VPN & valid bit set)

Physical page number – 5

4. Since it was a hit, final Physical address – 0101011100(0x15C)