9/9 Questions Answered

Saved at 1:40 PM

Check-in Quiz 03 Virtual Memory

Q1
3 Points

Consider a system with the following configuration:

- 64 bit address space (one address is 64 bits)
- 16 GiB physical memory
- 16 bit (2 bytes) addressability
- 4 KiB page size

Note that we have an addressability of 16 bits (2 bytes), in the VM lectures, we usually worked with 8-bit addresability. This means that each address corresponded to 16 bits of memory, meaning that instead of each address corresponding to a single byte, it instead corresponds to two bytes.

As a reminder:
1KiB = 2^10 bytes
1GiB = 2^30 bytes
(2^10 is 2 to the power of 10 or 1024)

Q1.1
1 Point

How many bits are needed to represent the page offset of an address? (e.g. how many bits does it to take to represent the number of possible addresses that correspond to a single page)

Please answer with a number only:

11

Explanation

Correct! A page is 4 KiB which is 2^2 * 2^ 10 bytes = 2^12 bytes. Since 2 bytes correspond to one address, there are only 2^11 addresses that go into a page, and thus we need 11 bits to specify where in a page we want to go (note that due to how addressability works, we can't specify a specific byte per address, instead we get an address to a specific pair of bytes).

Save Answer   Last saved on **Oct 09 at 1:39 PM**

Q1.2
1 Point

How many bits are needed to represent the virtual page number of an address?

Please answer with a number only:

53

Explanation

Correct! There are 2^64 addresses, and each 2^11 addresses go to a page. This means we can divide memory into 2^64 / 2^11 = 2^53 pages.

Save Answer   Last saved on **Oct 09 at 1:39 PM**

Q1.3
1 Point

How many bits does it take to represent a physical page?

Please answer with a number only:

22

Explanation

Correct! First thing to do is calculate the number of bytes in physical memory. 16 GiB = 2^4 * 2^30 bytes = 2^34 bytes. Then dividing by page size we get 2^34 / 2^12 = 2^22 physical page frames.

Save Answer Last saved on **Oct 09 at 1:39 PM**

Q2 LRU

2 Points

Imagine we are working with a system that has 4 physical pages and that we are u sing LRU as the replacement policy.

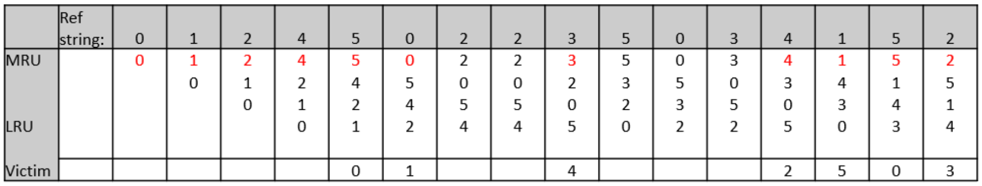
For the following reference string, what is the number of page faults?

0 1 2 4 5 0 2 2 3 5 0 3 4 1 5 2

11

Explanation

We get 11 page faults:



	Ref string:	0	1	2	4	5	0	2	2	3	5	0	3	4	1	5	2
MRU		0	1	2	4	5	0	2	2	3	5	0	3	4	1	5	2
			0	1	2	4	5	0	0	2	3	5	0	3	4	1	5
				0	1	2	4	5	5	0	2	3	5	0	3	4	1
LRU					0	1	2	4	4	5	0	2	2	5	0	3	4
Victim						0	1			4				2	5	0	3

Save Answer Last saved on **Oct 09 at 1:39 PM**

Q3 Page Tables

5 Points

For this problem, let us work with an architecture that supports

- 16-bit addresses
- 1-byte addressability
- pages that are 1 KiB
- 4 physical page frames

Assume that on this architecture, we access the following virtual pages:

- 0x36 (11 01 10)
- 0x1B (01 10 11)
- 0x0A (00 10 10)
- 0x0C (00 11 00)
- 0x1A (01 10 10)
- 0x0B (00 10 11)

Assume that physical memory starts empty.

As a reminder:
1KiB = 2^10 bytes
(2^10 is 2 to the power of 10 or 1024)

Q3.1 Array
1 Point

How many entries would exist in the page table if it was a "big array" page table

64

Explanation

Correct! There is one PTE for every possible virtual page

Save Answer

Last saved on **Oct 09 at 1:40 PM**

Q3.2 Inverted
1 Point

How many entries would exist if it used an inverted page table?

4

Explanation

Correct! there is one PTE per physical page

Save Answer Last saved on **Oct 09 at 1:40 PM**

Q3.3 Multi-level
1 Point

What if the architecture used a 3-level page table that had 2 bits to index into each level of the page table. (Note in lecture we had a 4-level page table with 9 bits to index into each level)

(Repeating the for convenience)
we access the following virtual pages:

- 0x36 (11 01 10)
- 0x1B (01 10 11)
- 0x0A (00 10 10)
- 0x0C (00 11 00)
- 0x1A (01 10 10)
- 0x0B (00 10 11)

How many Page Table Entries would exist in the page table (after accessing the addresses listed above)
(You can also think of this as how many valid entries are there in the bottom level of the page table).

6

Explanation

Correct! there will be one PTE per page that is accessed

Save Answer Last saved on **Oct 09 at 1:40 PM**

Q3.4 Multi Level Page Table: Top Level
1 Point

After accessing the specified addresses, how many entries would be non-`NULL` in the top level of the page table? (Same page table as the one described in 3.3)

3

Explanation

Correct! there will be three entries. We can look at the top two bits of each of the virtual pages. We see that there is 00, 01, and 11. Those are three different indexes into the top level page table.

Save Answer Last saved on **Oct 09 at 1:40 PM**

Q3.5 Multi Level Page Table: Mid Level
1 Point

After accessing the specified addresses, how many entries would be non-`NULL` in the second-highest level of the page table? (Same page table as the one described in 3.3)

4

Explanation

Correct! We can look at how many unique combinations of the upper 4 bits of the virtual page number to get this. We get 0010, 0011, 0110, 1101. Note that we care about these 4 bits instead of just two when deciding this specific metric. There are only three unique 2-bit indexes into the mid level page table (01, 10, 11), but some of these (like page 0x0A and 0x1A) are indexes into different tables in the second level.

Save Answer Last saved on **Oct 09 at 1:40 PM**

Save All Answers

Submit & View Submission ›