CS 31 Homework 9: Paging and Virtual Memory

Due: Tuesday, Dec. 3, 2024

Names:

Question 1

For each of the following x86-64 instructions, indicate whether the instruction **could** cause a page fault, whether it **could** cause a cache miss, and whether it **could** cause the dirty bit in the cache to be set to 1. HINT: Think of whether a particular instruction **could** result in a read or write from main memory, the cache or the registers.

a. `movq $7, %rcx`

Page fault? YES or NO	Cache miss? YES or NO	Dirty bit? YES or NO

b. `movq $7, (%rdx)`

Page fault? YES or NO	Cache miss? YES or NO	Dirty bit? YES or NO

c. `movq (%rax), %rbx`

Page fault? YES or NO	Cache miss? YES or NO	Dirty bit? YES or NO

d. `addq %rbx, -8(%rax)`

Page fault? YES or NO	Cache miss? YES or NO	Dirty bit? YES or NO

Question 2

Assume a system with the following architecture:

- **8-bit** virtual addresses, **16-byte** page size, **4** pages of RAM

a. How many bytes of data can a single process store in:

(a) physical memory (b) virtual memory

b. Divide each virtual address listed below into its `page number` and `page offset`.

`1 0 0 0 1 0 1 0` `1 0 0 0 1 1 1 1` `1 0 1 0 1 0 1 0`

c. On the Page Table diagram (next page), show the history of the results of the following memory operations on RAM and the Page Table. Assume FIFO (first-in-first-out) replacement. Within each box, time progresses downward, so the first address loaded appears at the top and subsequent changes are written below. To the right of the table, label each change with number of the operation that caused it. Annotate each operation below with *hit* or *page fault* to indicate whether the data was found in RAM. Don't forget to update the valid bits, especially when a page is kicked out!

(a) `read 0 0 0 1 1 0 1 0`
(b) `write 0 0 0 1 1 0 1 1`
(c) `read 1 1 1 1 1 0 0 0`
(d) `read 1 1 1 1 1 0 1 0`
(e) `read 0 1 1 0 1 0 0 0`
(f) `write 0 0 0 0 1 0 0 1`
(g) `read 0 0 0 0 0 0 0 0`
(h) `read 0 1 0 1 0 1 1 1`
(i) `write 0 0 0 1 1 0 1 0`
(j) `read 0 0 0 1 1 1 0 1`

Use the Page Table (on the next page) and the map of RAM (below) to help keep track of virtual memory.

RAM

frame #	page #
00	
01	
10	
11	

Page Table

index	V	frame	notes
0	0		
1	0		
...	...	...	
5	0		
6	0		
...	...	...	
15	0		