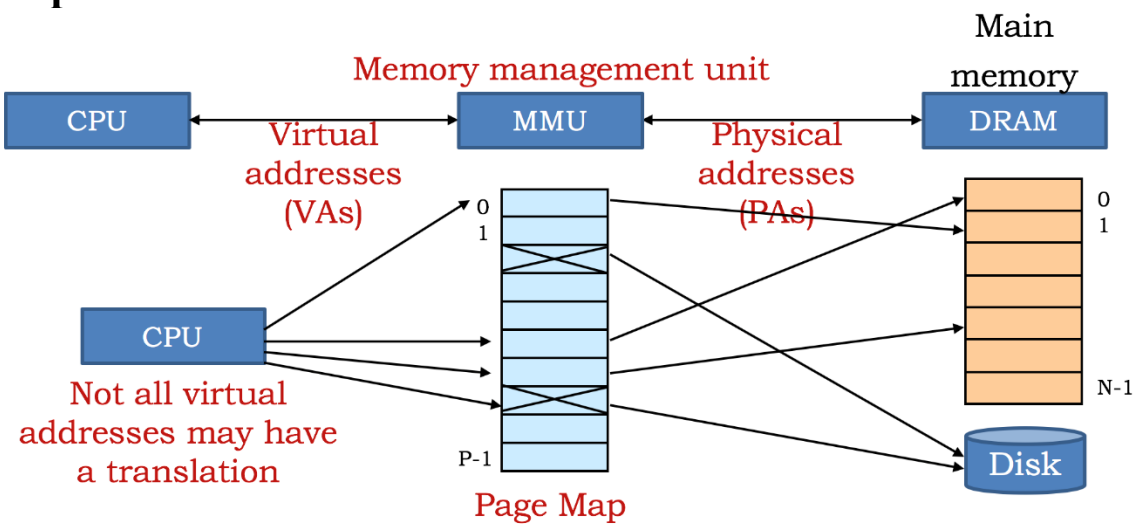
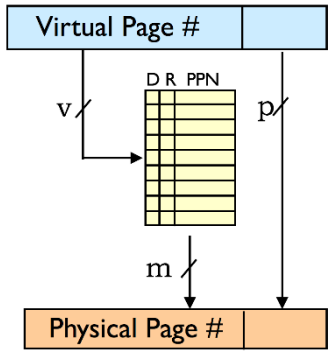
6.004 Recitation 18

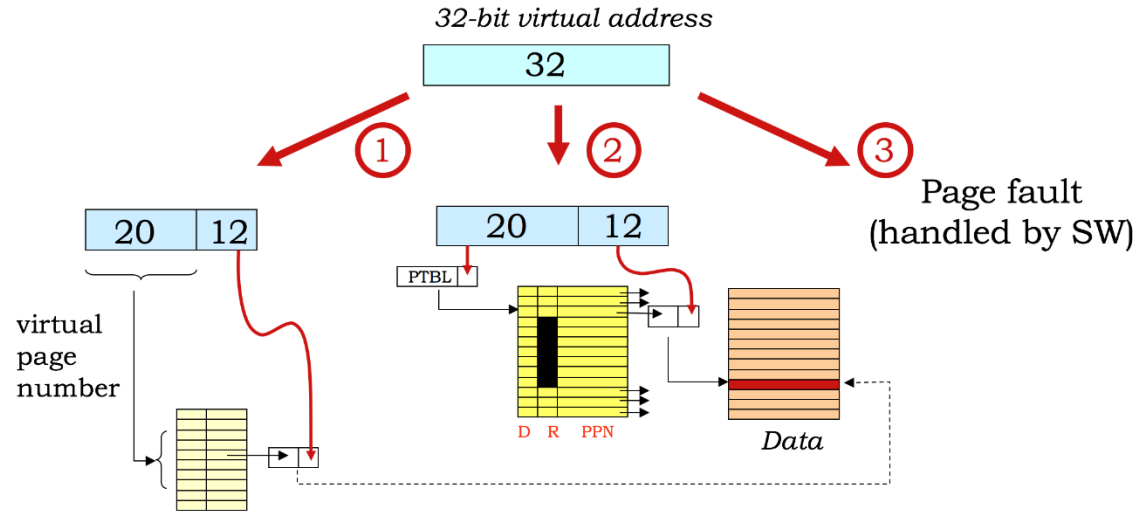
L18 – Virtual Memory

Conceptual Overview





(v + p)	bits in virtual address
(m + p)	bits in physical address
2^v	number of *virtual* pages
2^m	number of *physical* pages
2^p	bytes per physical page
2^{v+p}	bytes in virtual memory
2^{m+p}	bytes in physical memory
$(m+2)2^v$	bits in the page table



Look in TLB: VPN→PPN cache

Lecture Take-Home Problem

Suppose we have a memory system with the following characteristics and current values:

- Virtual Memory of size 2^{32} bytes
- Physical Memory of size 2^{24} bytes
- Page size is 2^{10} bytes
- 4-entry fully associative TLB

TLB

Tag | Data

VPN	R	D	PPN
0	0	0	3
6	1	1	2
1	1	1	9
3	0	0	5

Page Table

VPN	R	D	PPN
0	0	0	7
1	1	1	9
2	1	0	0
3	0	0	5
4	1	0	5
5	0	0	3
6	1	1	2
7	1	0	4
8	1	0	1
	…		

A. How many pages can be stored in physical memory at once?

2^{14}

B. How many entries are there in the page table?

2^{22}

C. How many bits per entry in the page table? (Assume each entry has PPN, resident bit, dirty bit)

16

D. How many pages does page table take?

2^{23} bytes = 2^{13} pages

E. What fraction of virtual memory can be resident?

$1/2^{8}$

F. What is the physical address for virtual address 0x1804? What components are involved in the translation? What about for 0x1080? And for 0x0FC?

[VPN = 6] 0x804

[VPN = 4] 0x1480

[VPN = 0] page fault

Practice, practice, practice!

Problem 1

Consider a virtual memory system that uses a single-level page table to translate virtual addresses into physical addresses. Each of the questions below asks you to consider what happens when **just ONE of the design parameters** (page size, virtual memory size, physical memory size) of the original system is changed. **Circle the correct answer.**

(A) If the physical memory size (in bytes) is **doubled**, the number of entries in the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

(B) If the page size (in bytes) is **halved**, the number of entries in the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

(C) If the virtual memory size (in bytes) is **doubled**, the number of bits in each entry of the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

(D) If the page size (in bytes) is **doubled**, the number of bits in each entry of the page table
(a) stays the same
(b) doubles
(c) is reduced by half
(d) increases by one
(e) decreases by one

Problem 2

A test program has been running on RISC-V with a 2^8 byte page size and has halted *just before* executing the following instructions.

```
lw a3, 0x3C8(x0)   FAULT
 sw a4, 0x440(x0)   HIT
 lw a5, 0x214(x0)   HIT
lw a6, 0x77C(x0)   FAULT
 sw a7, 0x5C8(x0)   HIT
```

Below is a table containing the first 8 locations of the page table at the time execution was halted, with the least-recently used and next-least-recently used pages marked. Assume all pages are being used. Execution resumes, and the above instructions are executed.

Which of the accesses memory locations will result in page table hits, and which will result in page faults? For every field that will change, cross it out, and then write the new value.

VPN	*D*	*R*	*PPN*
0	1	1	0x1
LRU -> 1	0	~~1~~ 0	0x0
2	1	1	0x6
3	0	~~0~~ 1	~~0x9~~ 0x0
4	~~0~~ 1	1	0x4
5	~~0~~ 1	1	0x2
Next LRU -> 6	~~1~~ 0	~~1~~ 0	0x7
7	0	~~0~~ 1	~~0x3~~ 0x7