Virtual Memory

Discussion 13

Zhanrui ZHANG

Why we need Virtual Memory

Adding Disks to Hierarchy (Use memory efficiently)

Simplifying Memory for Apps

Protection Between Processes

Virtual Memory

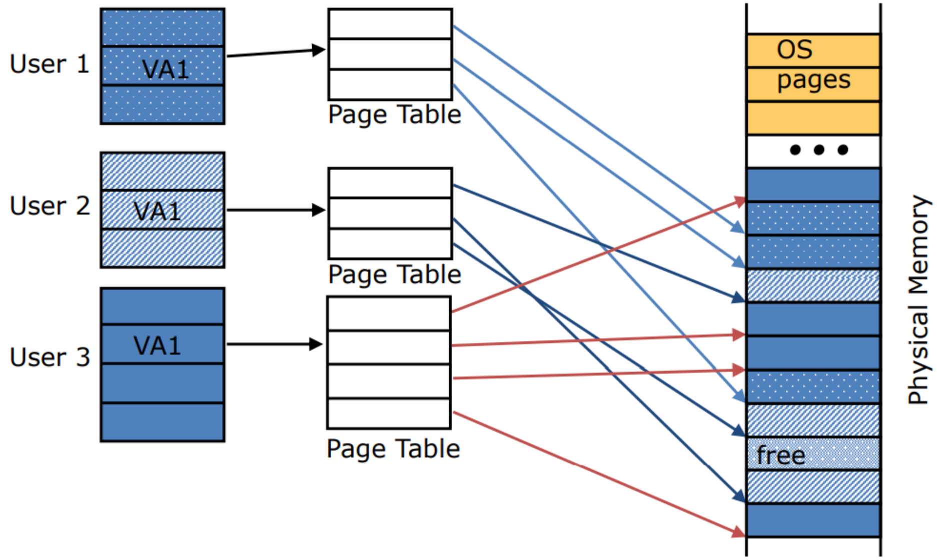
- Memory is split up to small equal sized sections called pages (page frame)

- Each program may occupy multiple pages, which are not necessarily contiguous in physical memory.

- A page table records where the different pages of a program are located in physical memory.

- Unused pages may be paged out to a swap space on disk to make room for others

Page Table



Virtual Page Number	Offset

Physical Page Number	Offset

Offset represent the address in one page, so it should always have the same bits

Page table resides in the main memory

Page Table Entry

Valid	Dirty	Permission Bits	PPN
— Page entry (VPN: 0) —			
— Page entry (VPN: 1) —			

PPN: Physical Page Number (May be empty)

Analogy: page table as a long array

index : virtual page number

content: page table entry

Page Fault

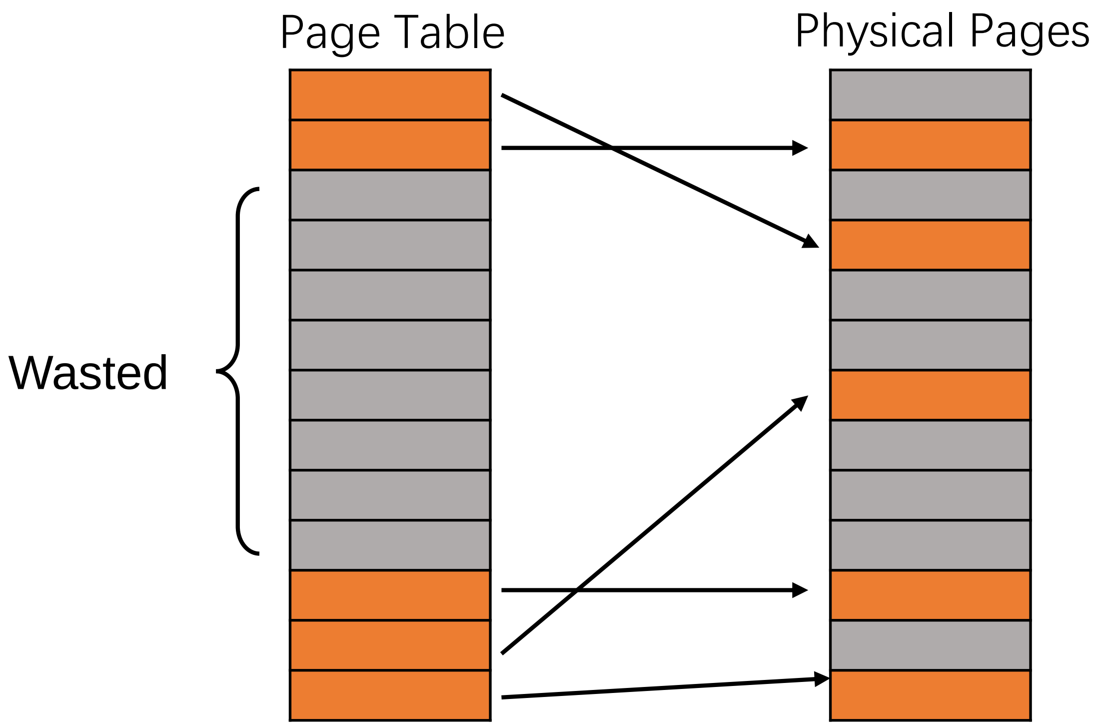
Virtual Page asked is not in memory
1. Allocate a new page in memory.
2. The page do exist, but it is on the disk.

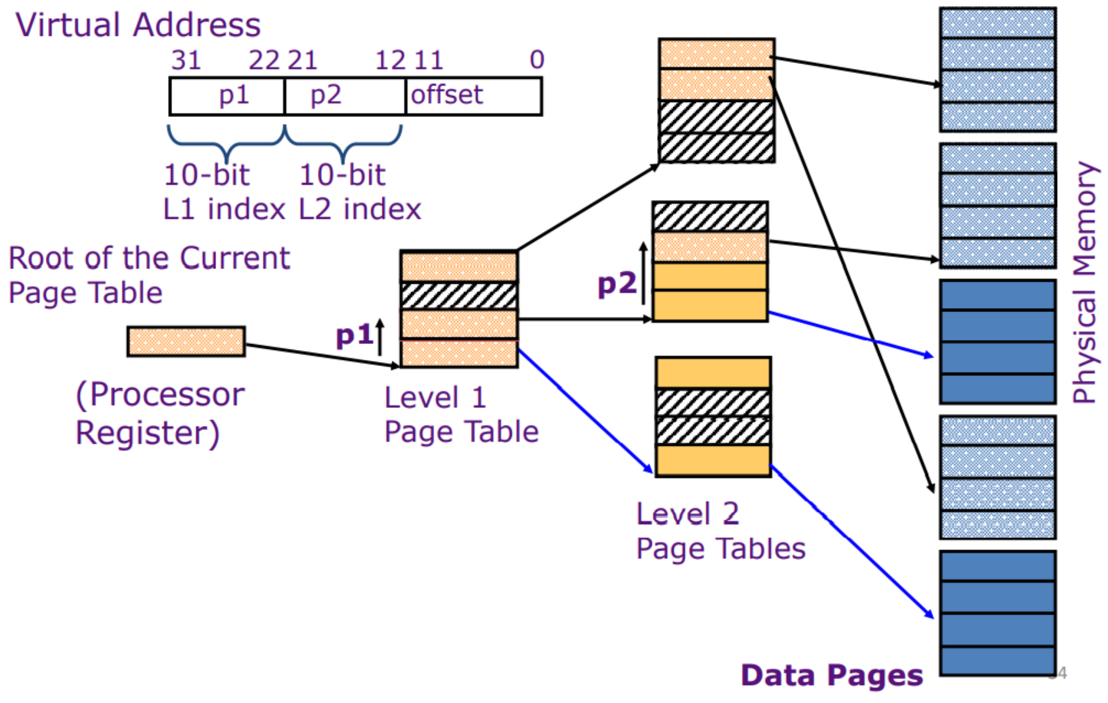
1. Allocate a new page in memory.
2. If no free space in memory, you need to evict one page first and swap the page back into memory.

Hierarchical Page Table

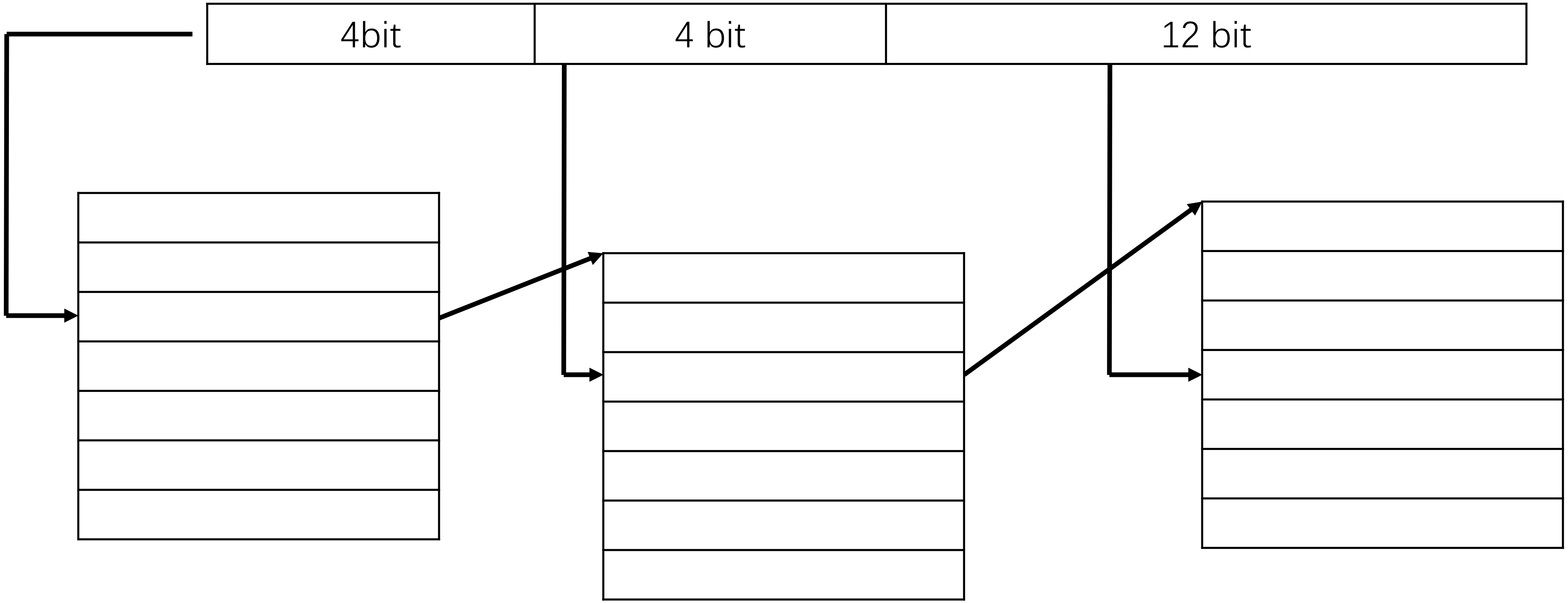
Page table resides in memory so of course it takes space.

If we have many pages, then storing those page table entries are quite expensive.





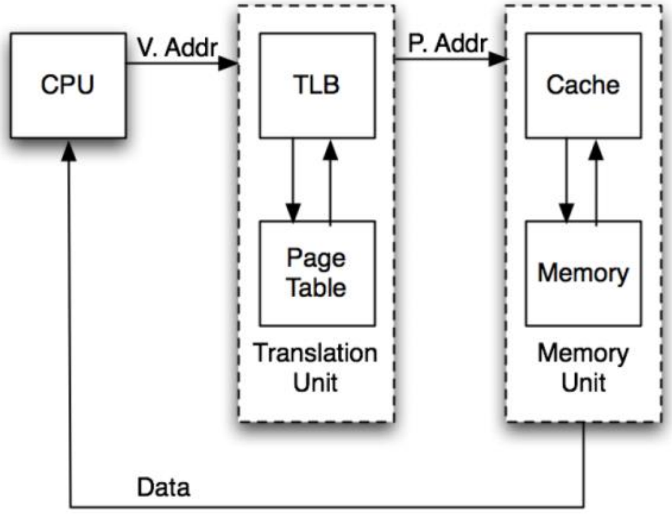
Hierarchical Page Table



Translation lookaside buffer

Cache for page translation

TLB miss is not Page fault!



TLB Valid	Tag (VPN)	Page Table Entry		
		Page Dirty	Permission Bits	PPN
— TLB entry —				
— TLB entry —				

For Exercise

Basic:

- Calculate offset based on page size
- Calculate virtual/physical page number.

Medium:

- Calculate the size of page table (one level)
- Understand the process of address translation in page table.
- LRU (or other) algorithm in TLB or page table.

Hard:

- Problem about multi-level page table.

Exercise

Consider a machine with a physical memory of 4 GB, a page size of 4 KB, and a page table entry size of 8 bytes. How many levels of page tables would be required to map a 48-bit virtual address space if every page table fits into a single page? And how many bits are there for each level of page table index in virtual address?

Consider a machine with a physical memory of 4 GB, a page size of 4 KB, and a page table entry size of 8 bytes. How many levels of page tables would be required to map a 48-bit virtual address space if every page table fits into a single page? And how many bits are there for each level of page table index in virtual address?

4KB page	12 bits offset
4KB page, 8 Byte PTE	2^9 PTEs in one page
4KB page, 48bit Virtual Address	2^(48-12)=2^36 Virtual pages

36 / 9 = 4 levels

THANK YOU