Multi-level paging

Glenn Bruns
CSUMB

Lecture Objectives

After this lecture, you should be able to:

- Explain the reasons why multi-level paging is used
- Be able to manually perform address translation with multi-level paging

Recap: Problems with paging

- page tables are big
- page tables are slow

translation look-aside buffers address the speed problem

Problem: big page tables

With 32 bit address space and 4 KB page size, about 4 MB per page table

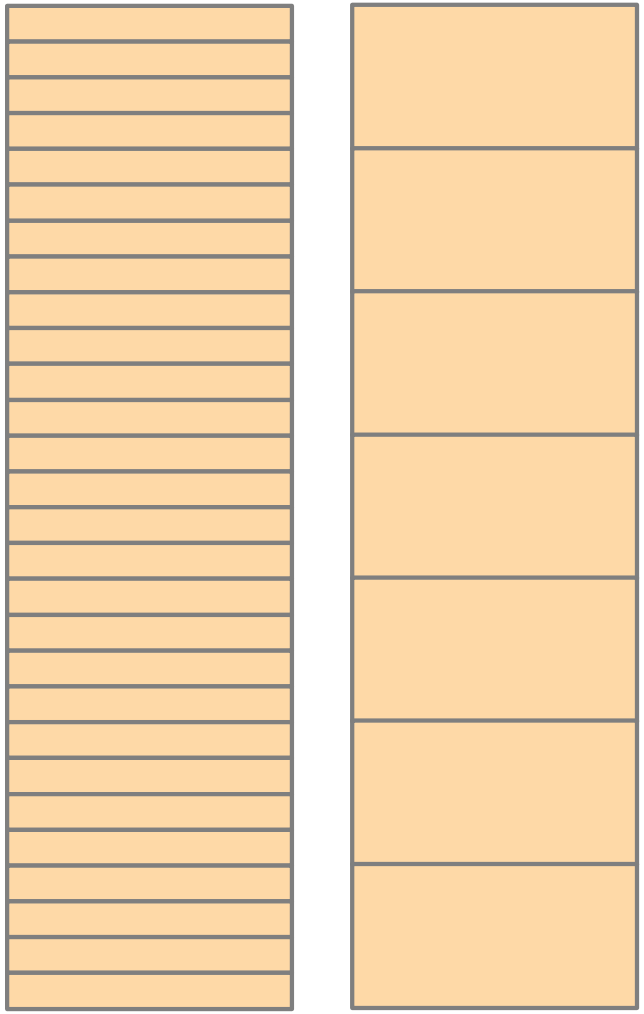
And one page table per process

A solution idea: make pages bigger

With a 32 bit address space, and a page size of 4 KB, a page table will have about a million entries.

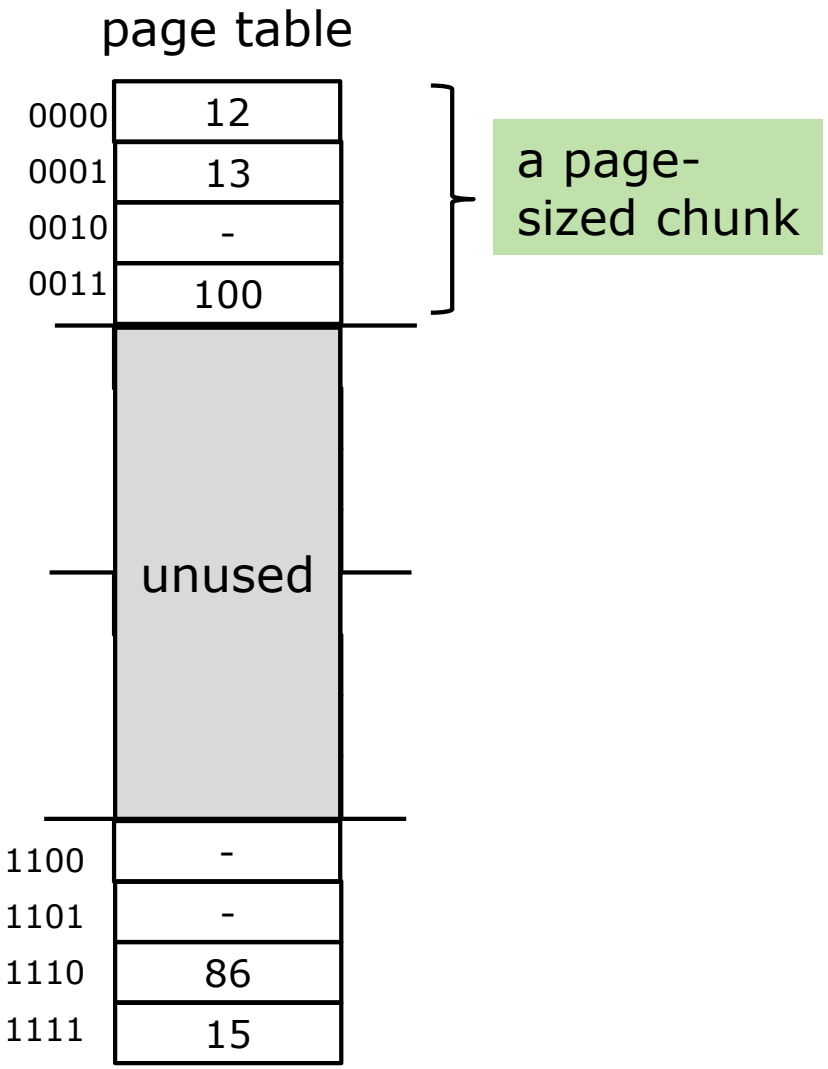
How many entries if page size changed to 64 KB?

problem: “internal fragmentation”



Another solution Idea

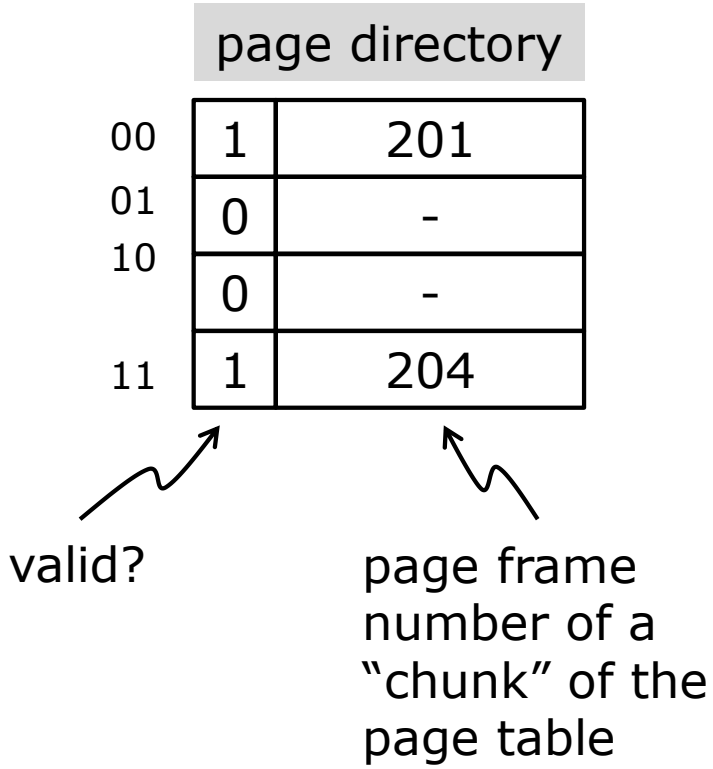
- insight: processes usually use only a small part of their address space
- Most of a processes' page table is unused
- Break page table into page-sized chunks, use only some of these chunks



Page directory

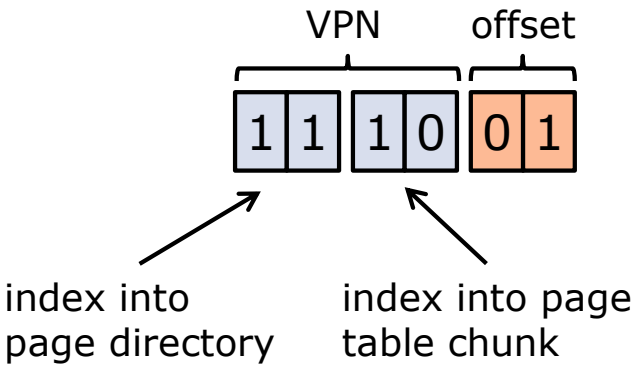
A new page directory shows, for each group of pages, either:

- pointer to a page table "chunk", or
- "not valid"



Good news: we only store the part of the page table we need
Bad news: page translation is more complicated

VPN now has 2 parts



page directory

00	1	201
01	0	-
10	0	-
11	1	204

page table

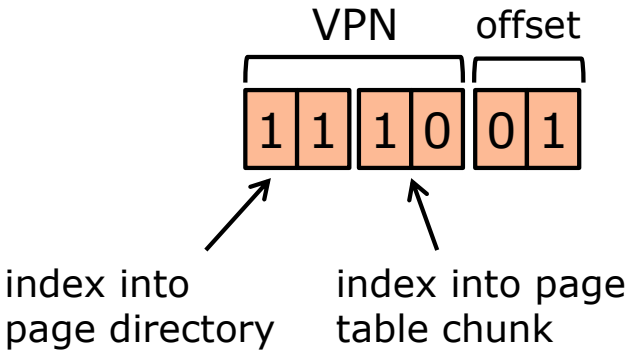
00	12
01	13
10	-
11	100

PFN 201

00	-
01	-
10	88
11	15

PFN 204

Address translation



page directory

00	1	201
01	0	-
10	0	-
11	1	204

page table

00	12
01	13
10	-
11	100

PFN 201

00	-
01	-
10	88
11	15

PFN 204

1. Where is page directory?

   Get page directory base address from a register

2. Which page table chunk?

   Add page directory index to get page directory entry; get PFN of page table chunk

3. Which page?

   If valid, add page table index to PFN of page table; get PFN of virtual page

Exercise

For each virtual address below, what is the physical address?

0	1	1	0	0	1

1	1	0	0	0	1

0	0	0	0	1	1

page directory

00	1	201
01	0	-
10	0	-
11	1	204

page table

00	12
01	13
10	-
11	100

PFN 201

00	-
01	-
10	88
11	15

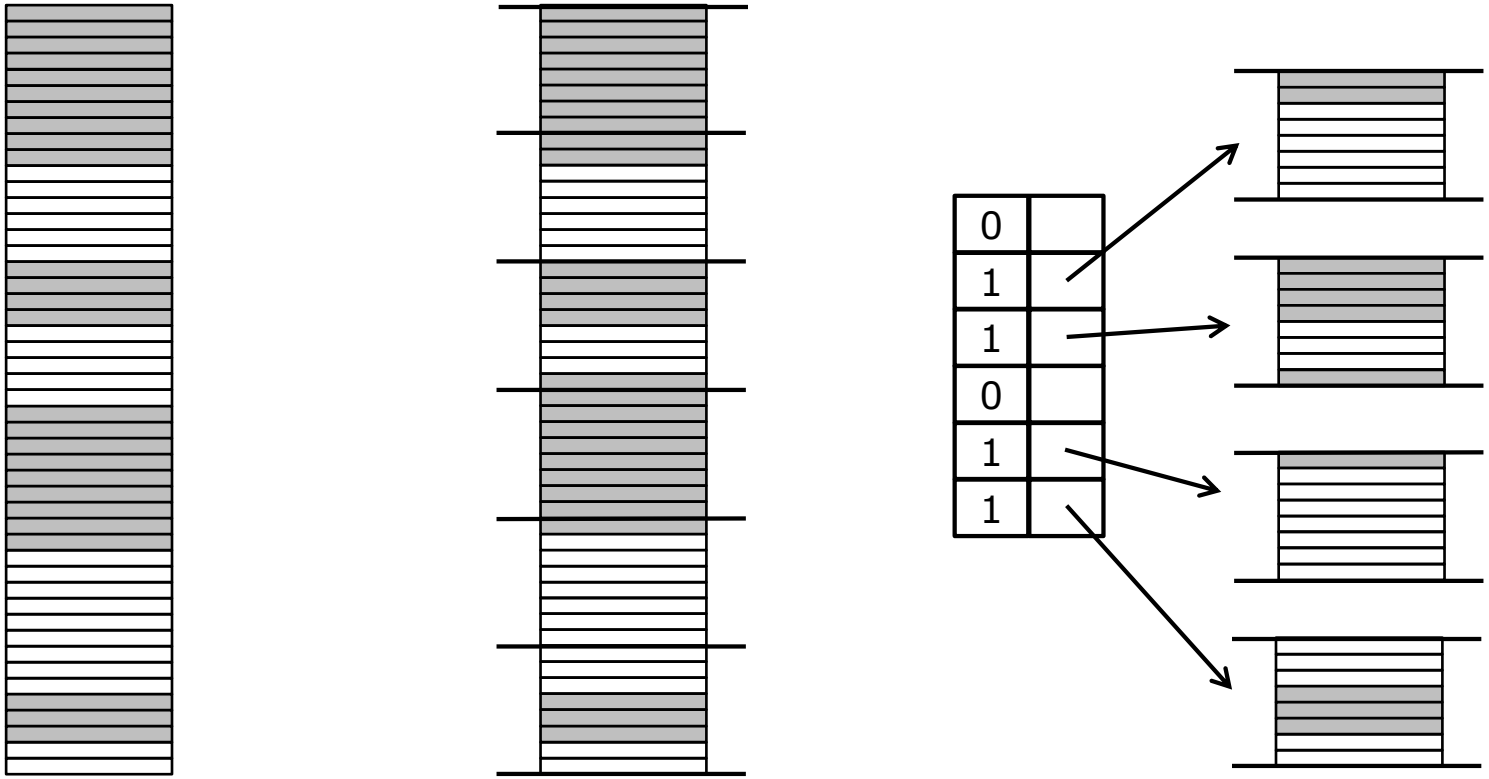
PFN 204

How much space spacing?

Before: simple page table

Look at page-sized chunks

After: unused chunks not stored

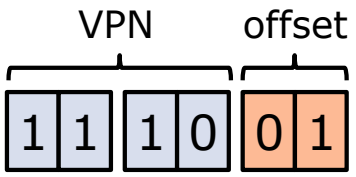


You save 1 page for every unused chunk, but add page directory

Address translation, including TLB

Use of TLB is the same as in the case of simple paging:

1. lookup the VPN part of the virtual address in the TLB
2. if a TLB hit:
   - form physical address from PFN in TLB entry
   - read value from physical address
3. if a TLB miss:
   - use the multi-level paging scheme to get a PFN value
   - stick the VPN, PFN pair in the TLB and retry instruction



see text for gory detail

Summary

- We need a page table for each process
- But most processes use only a small portion of their virtual address space
- Multi-level page tables avoid much of this waste