Exercise Session 8

Computer Systems

Agenda

- Virtual Memory
  - Segmentation
  - Paging
  - TLB
- Demand Paging
  - Page Replacement

Terminology

- Physical address: address as seen by the memory unit
- Virtual or Logical address: address issued by the processor
  - Loads
  - Stores
  - Instruction fetches
  - Possible others (e.g. TLB fills)...

Memory management

1. Allocating physical addresses to applications
2. Managing the name translation of virtual addresses to physical addresses
3. Performing access control on memory access

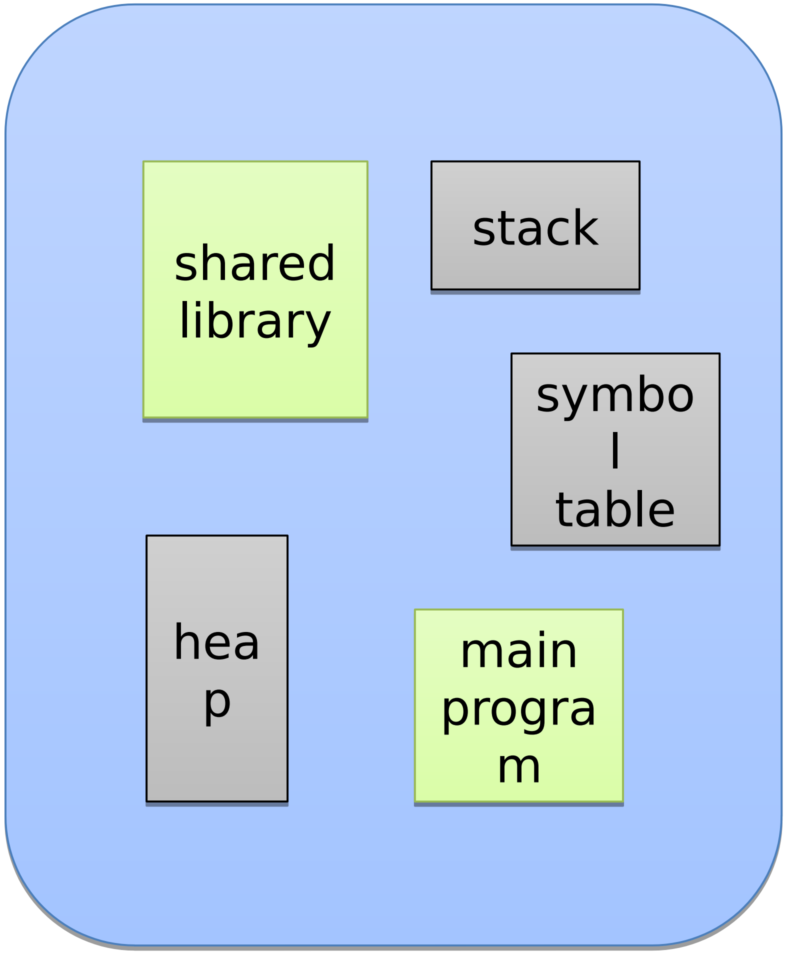
- Functions 2 & 3 usually involve the hardware Memory Management Unit (MMU)

Segmentation

Segmentation

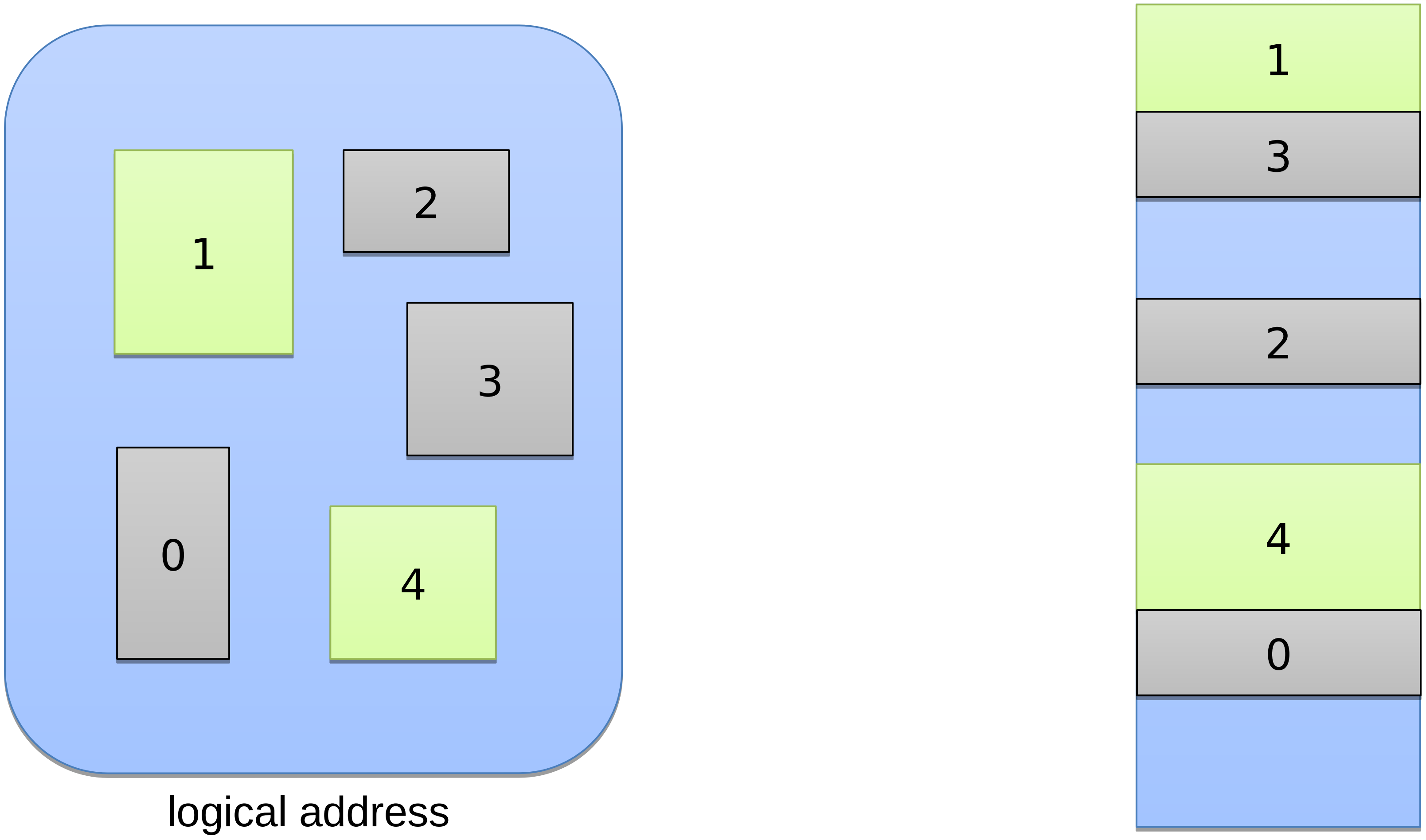
- Generalize base + limit:
  - Physical memory divided into *segments*
  - Logical address = (segment id, offset)
- Segment identifier supplied by:
  - Explicit instruction reference
  - Explicit processor segment register
  - Implicit process state

User’s View of a Program



logical address

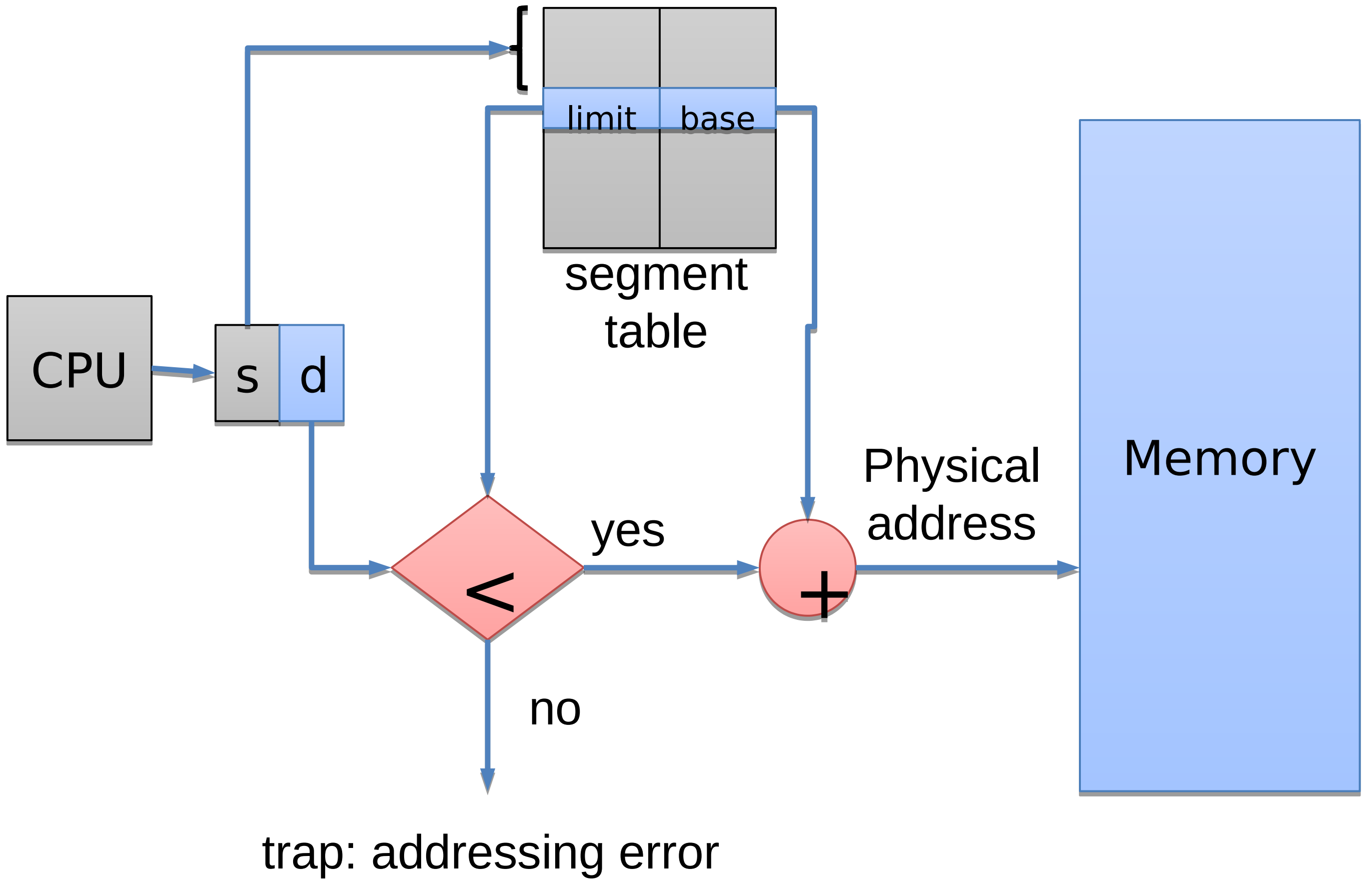
Segmentation reality



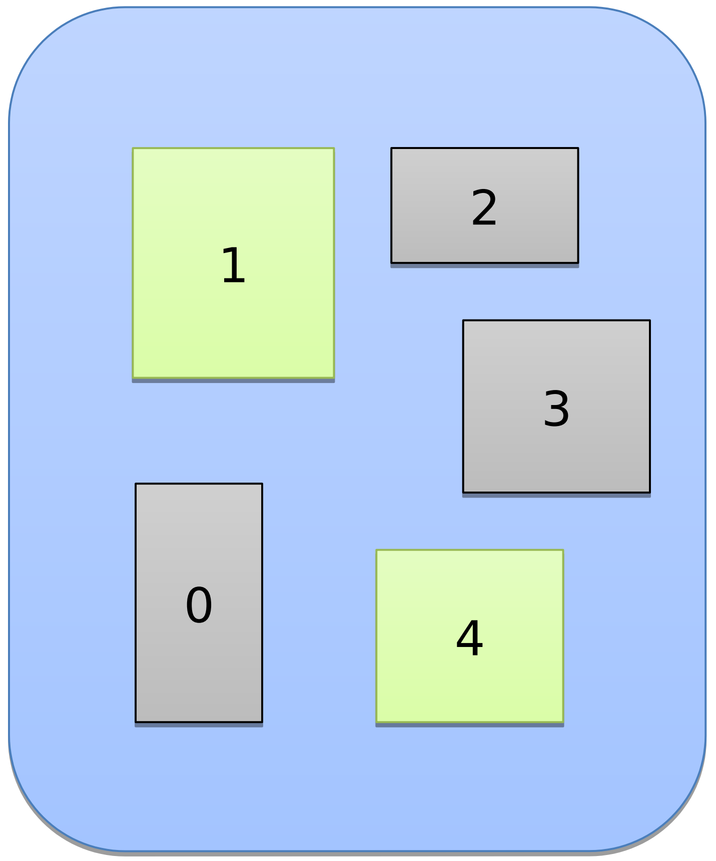
logical address

physical memory

Segmentation Hardware



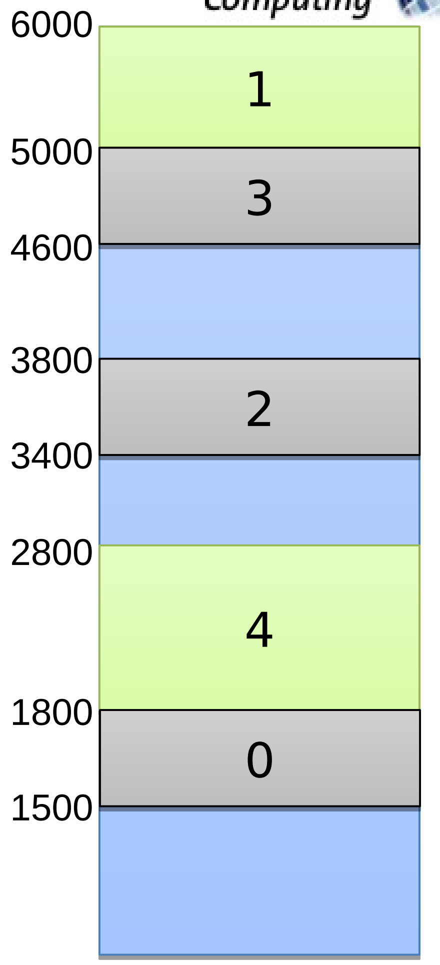
Segmentation reality



logical address

	limit	base
0	300	1500
1	1000	5000
2	400	3400
3	400	4600
4	1000	1800

segment table



physical memory

Segmentation Architecture

- **Segment table** – each entry has:
  - **base** – starting physical address of segment
  - **limit** – length of the segment
- **Segment-table base register (STBR)**
  - Current segment table location in memory
- **Segment-table length register (STLR)**
  - Current size of segment table

segment number ***s*** is legal if ***s*** < **STLR**

Paging

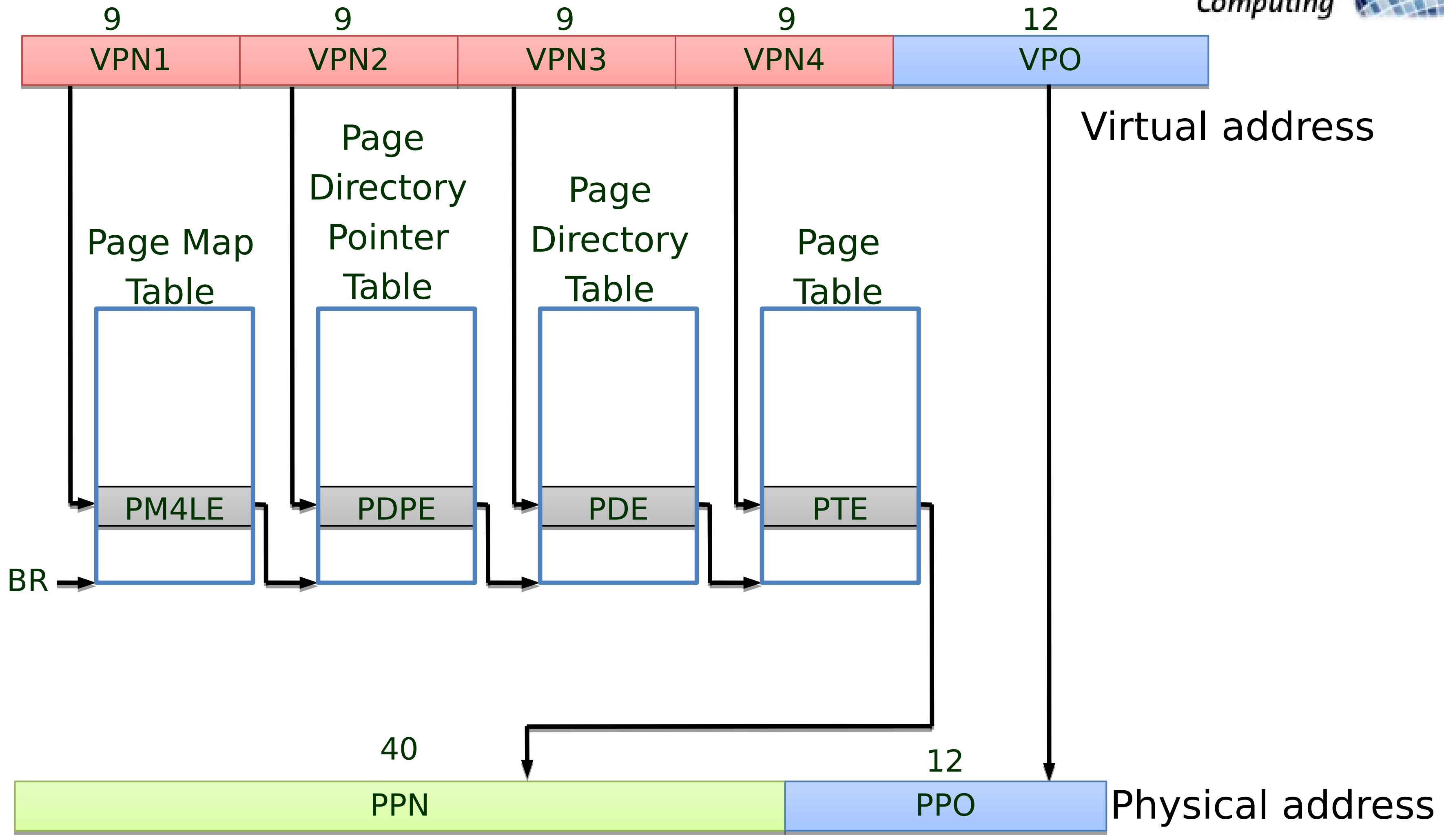
Paging

- Solves contiguous physical memory problem
  - Process can always fit if there is available free memory
- Divide physical memory into ***frames***
  - Size is power of two, e.g. 4096 bytes
- Divide logical memory into ***pages*** of the same size
- For a program of *n* pages in size:
  - Find and allocate *n* frames
  - Load program
  - Set up ***page table*** to translate logical pages to physical frames

Page table jargon

- Page tables maps VPNs to PFNs
  - Page table entry = PTE
- ***VPN*** = Virtual Page Number
  - Upper bits of virtual or logical address
- ***PFN*** = Page Frame Number
  - Upper bits of physical or logical address
- Same number of bits (usually).

x86-64 Paging



Systems@ETH Zürich

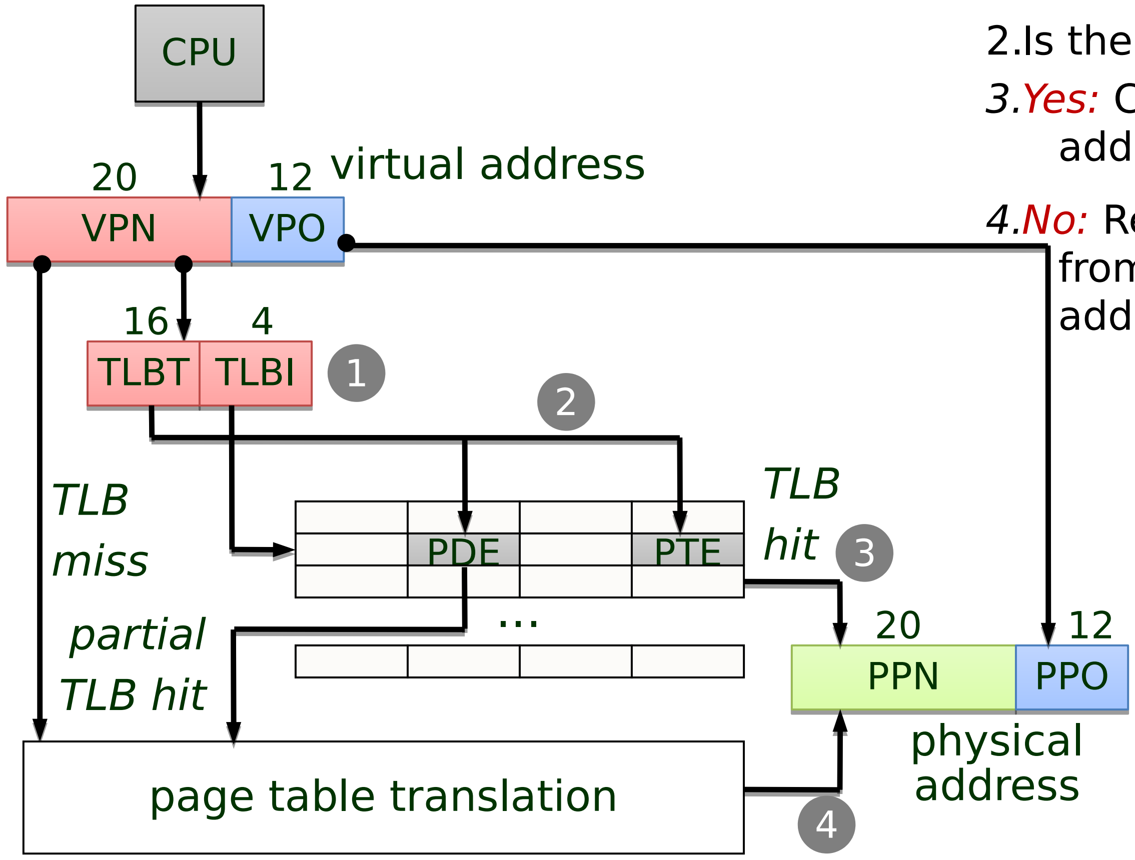
Distributed Computing

Translation Lookaside Buffer (TLB)

Problem: performance

- Every logical memory access needs two physical memory accesses
  - Load page table entry
  - Load desired location
- Performance ⇒ half as fast as with no translation
  - Solution: cache page table entries

Translating with the P6 TLB



1. Partition VPN into TLBT and TLBI.
2. Is the PTE for VPN cached in set TLBI?
3. *Yes:* Check permissions, build physical address
4. *No:* Read PTE (and PDE if not cached) from memory and build physical address

Caches and Paging

Cache Management

- Virtual Index, Virtual Tag

  +No need to wait for translation

  -Might get the wrong data if not flushed

- Physical Index, Physical Tag

  +Easy to manage

  -Need to wait for address translation

Cache Management

- Virtual Index, Physical Tag

  +Address lookup can happen in parallel with cache lookup

  -Either limited size or overhead

- Physical Index, Virtual Tag

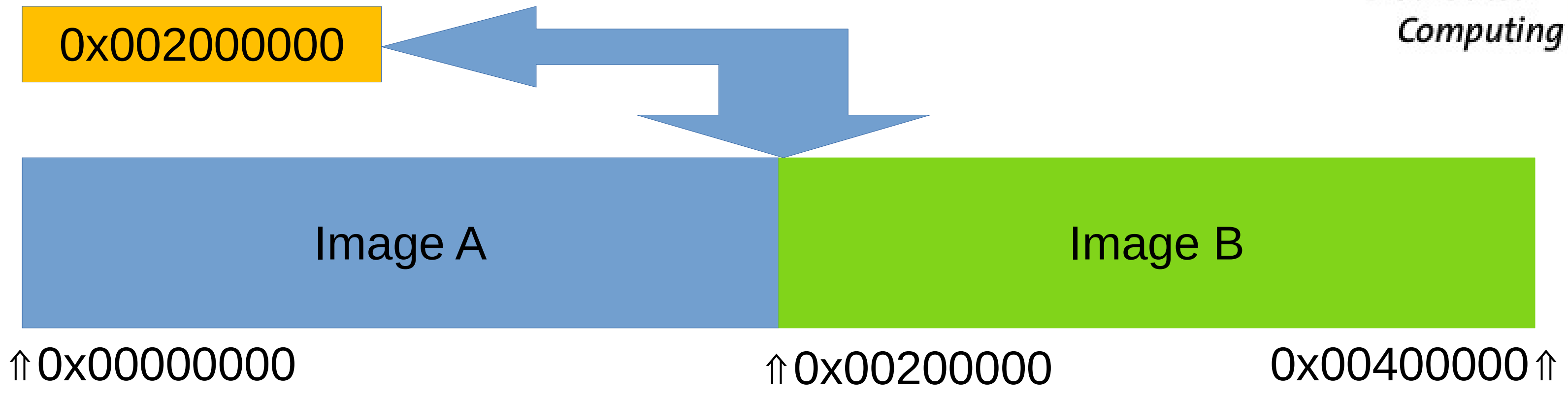
  No

Virtual Memory really does make your life easier!

Example: ESP8266

- Small Wifi-enabled microcontroller
- Typically 4MB flash storage for code
- Used in embedded applications #IoT
- Wouldn’t it be nice if we could update our smart flower pots without disassembling them?

OTA updates



Idea: Run from one partition and update the other one
This works great!
Until someone tries to use the PC to address something…
Solution: Link two images and never flash the wrong one!

Vollbild

LUX Fäustel 1.250 g Classic

☆☆☆☆☆ (0)

- Gewicht: 1'250 g
- Glasfasergriff
- Nach DIN gefertigt

Alle Artikelinfos

Menge 1 **Fr. 12.90**

Preis inkl. gesetzl. MWST. 7.7%

Reservieren & im Markt selbst abholen

Abholbereit ab nächstem Öffnungstag 11 Uhr im OBI Markt Volketswil

Im OBI Markt **Volketswil**
weniger als 3 Artikel vorrätig

Verfügbarkeit in anderem Markt prüfen

Artikel vergleichen Artikel merken

Artikelbeschreibung Bewertungen (0)

Artikelbeschreibung

Art.Nr. 3439098

☆☆☆☆☆ (0) Bewertungen lesen

Der LUX-TOOLS Fäustel 1.250 g hat einen geschmiedeten Kopf und ist mit einem stabilen, gummierten Griff aus Glasfaser verklebt.

Gewicht: 1.250 g
Material des Kopfes: Werkzeugstahl.
Nach DIN 1193 und DIN 6473.

Technische Daten

Produktmerkmale

Art:	Fäustel
Typ:	Sonstige

Masse und Gewicht

Ähnliche Produkte

LUX Sappie 600 g Comfort

☆☆☆☆☆

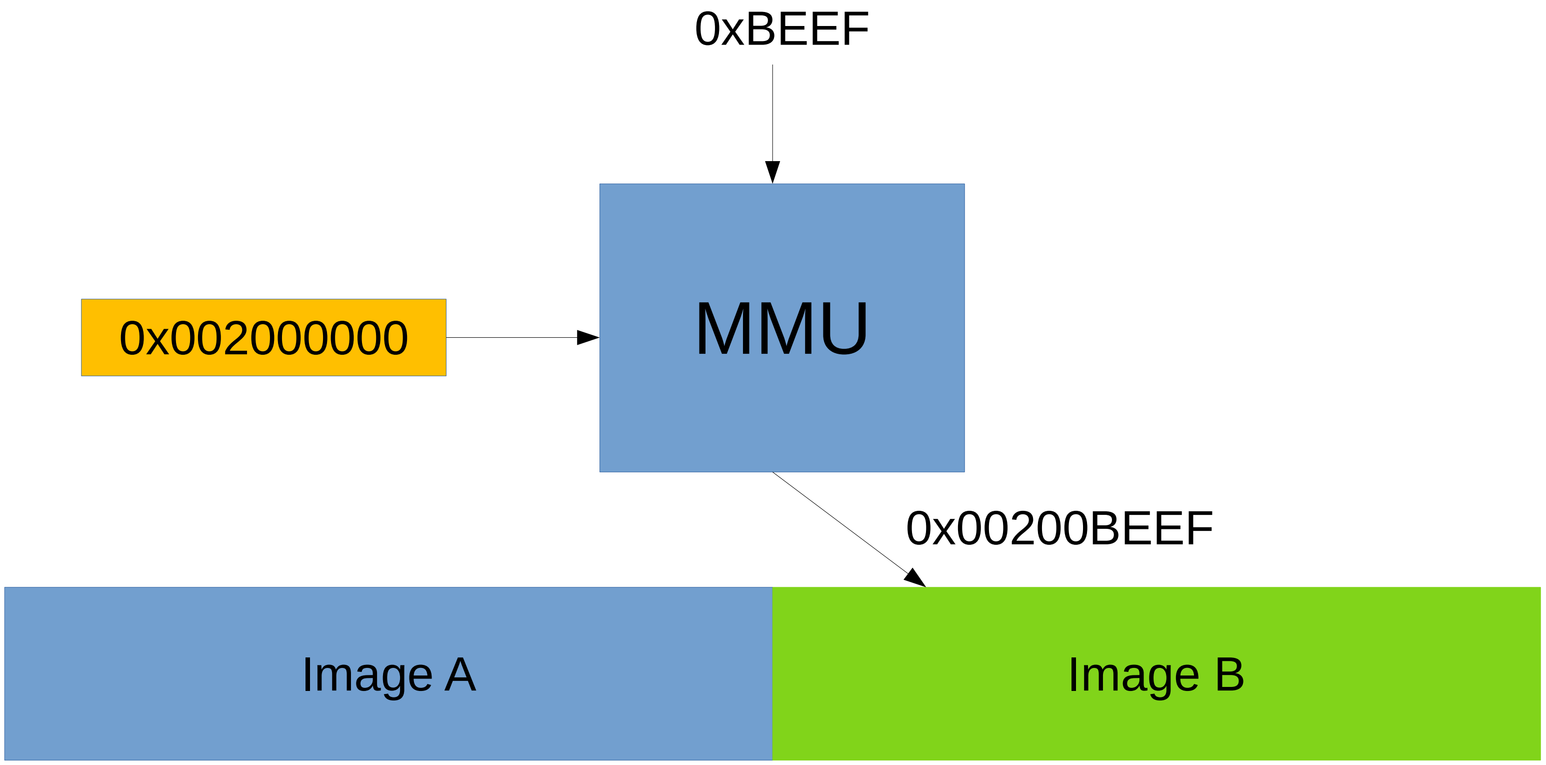
Fr. 44.-

LUX Latthammer Ganzstahl 770 g Comfort

☆☆☆☆☆

Fr. 43.-

Better Solution: Use an MMU!



But Wait! There’s More!

Systems@ETH Zürich

Distributed Computing

- Protection features can be abused for fun and profit
- E.g. Copy-On-Write:
- Share a page between two processes
- Make it read-only
- On a write, create a copy of that page and write there

Systems@ETH Zürich

Distributed Computing

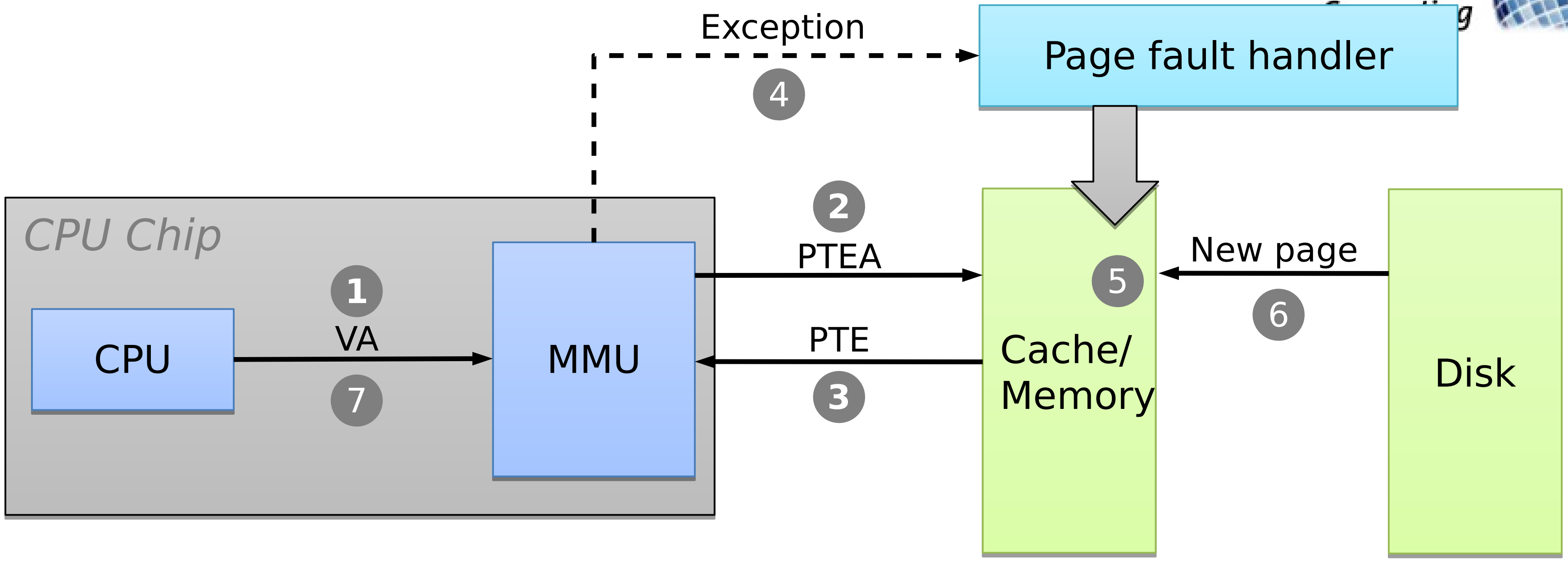
Demand Paging

Demand Paging

- Bring a page into memory only when it is needed
  - Less I/O needed
  - Less memory needed
  - Faster response
  - More users

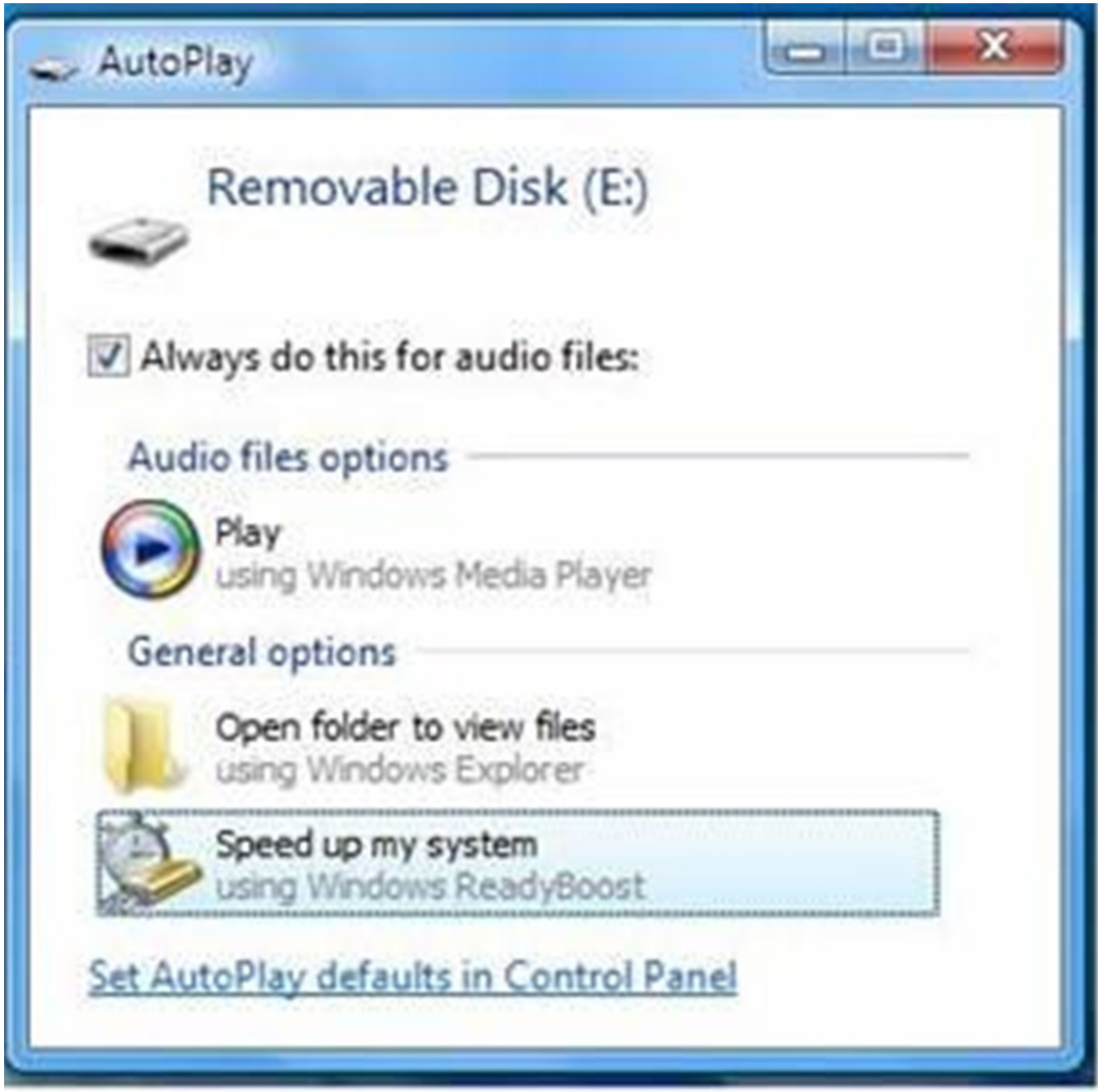
- Turns RAM into a *cache* for processes on *disk*!

Recall: handling a page fault



1) Processor sends virtual address to MMU
2-3) MMU fetches PTE from page table in memory
4) Valid bit is zero, so MMU triggers page fault exception
5) Handler finds a frame to use for missing page
6) Handler pages in new page and updates PTE in memory
7) Handler returns to original process, restarting faulting instruction

Anyone remember this?



AutoPlay

Removable Disk (E:)

[✓] Always do this for audio files:

Audio files options

Play
using Windows Media Player

General options

Open folder to view files
using Windows Explorer

Speed up my system
using Windows ReadyBoost

Set AutoPlay defaults in Control Panel

Anyone remember THIS?

Page Replacement

What happens if there is no free frame?

- *Page replacement* – find “little used” resident page to discard or write to disk
  - “victim page”
  - algorithm
  - performance – want an algorithm which will result in minimum number of page faults
- Same page may be brought into memory several times

FIFO (First-In-First-Out) page replacement

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

3 page frames:

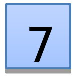
FIFO (First-In-First-Out) page replacement

Systems@ETH Zürich

Distributed Computing

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

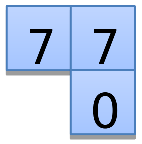
3 page frames:



FIFO (First-In-First-Out) page replacement

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

3 page frames:



FIFO (First-In-First-Out) page replacement

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

3 page frames:

7	7	7
	0	0
		1

FIFO (First-In-First-Out) page replacement

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

3 page frames:

7	7	7	2
	0	0	0
		1	1

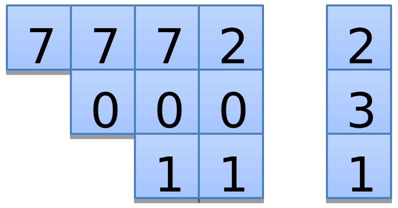
FIFO (First-In-First-Out) page replacement

Systems@ETH zürich

Distributed Computing

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

3 page frames:



FIFO (First-In-First-Out) page replacement

reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

3 page frames:

7	7	7	2		2	2	4	4	4	0
	0	0	0		3	3	3	2	2	2
		1	1		1	0	0	0	3	3

FIFO (First-In-First-Out) page replacement

reference string:	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
3 page frames:	7	7	7	2		2	2	4	4	4	0			0	0					
		0	0	0		3	3	3	2	2	2			1	1					
			1	1		1	0	0	0	3	3			3	2					

FIFO (First-In-First-Out) page replacement

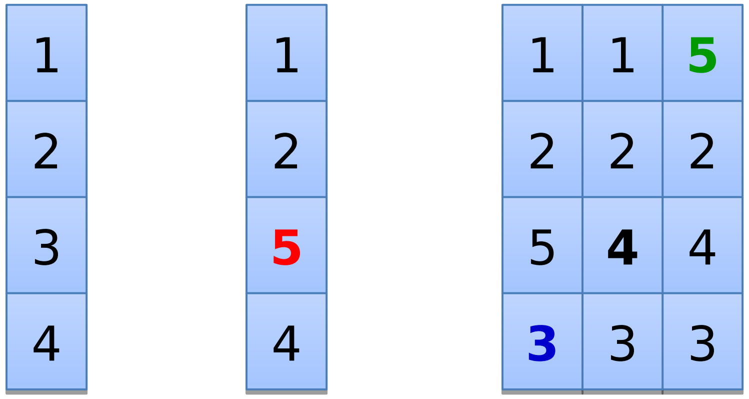
reference string:	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
3 page frames:	7	7	7	2		2	2	4	4	4	0			0	0			7	7	7
		0	0	0		3	3	3	2	2	2			1	1			1	0	0
			1	1		1	0	0	0	3	3			3	2			2	2	1

Here, 15 page faults.

Least Recently Used (LRU) algorithm

- Reference string: 1 2 3 4 1 2 **5** 1 2 **3** **4** **5**

  4 frames



- Counter implementation
  - Every page entry has a counter; every time page is referenced through this entry, copy the clock into the counter
  - When a page needs to be changed, look at the counters to determine which are to change

LRU page replacement

reference string:	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
3 page frames:	7	7	7	2		2		4	4	4	0			1		1		1		
		0	0	0		0		0	0	3	3			3		0		0		
			1	1		3		3	2	2	2			2		2		7		

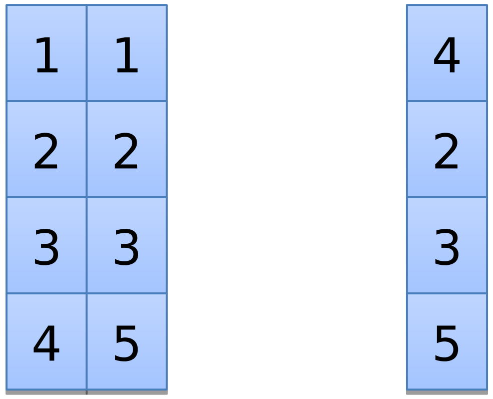
Here, 12 page faults.

Optimal algorithm

Replace page that will *not be used* for longest period of time

4 frames example:

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



⇒ 6 page faults

How do you know this? – you can't!

Used for measuring how well your algorithm performs