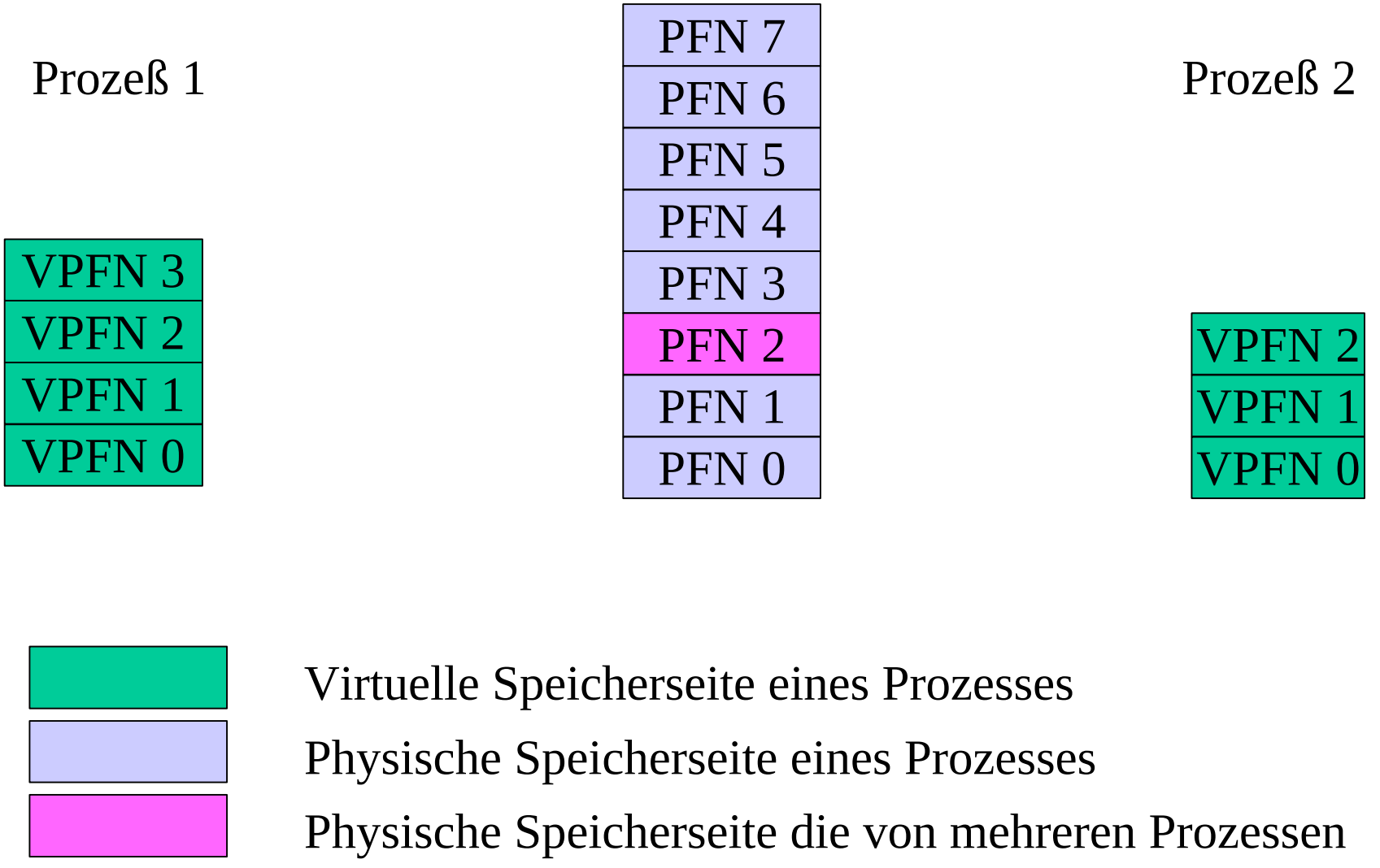
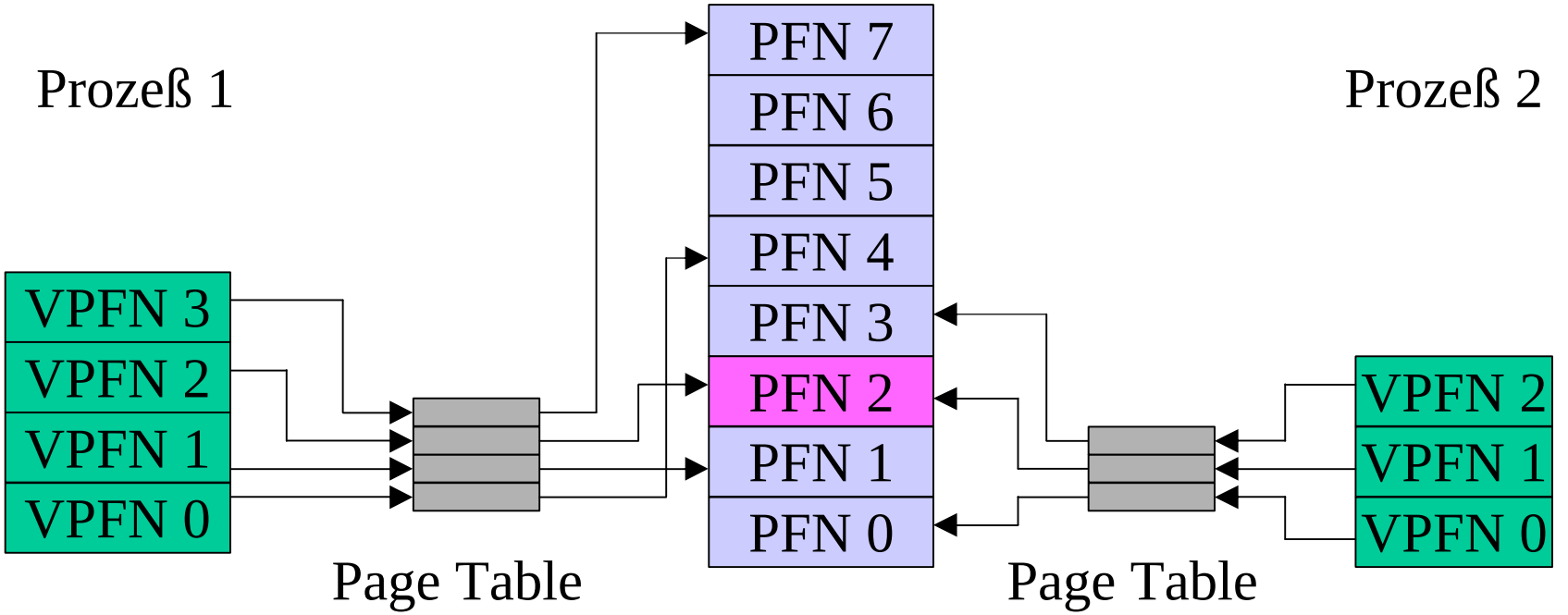
Linux Paging, Caching und Swapping

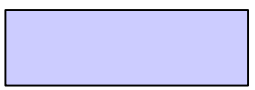
Inhalte

- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

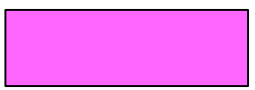

Prozeß 1
Prozeß 2
PFN 7
PFN 6
PFN 5
PFN 4
PFN 3
PFN 2
PFN 1
PFN 0
VPFN 3
VPFN 2
VPFN 1
VPFN 0
VPFN 2
VPFN 1
VPFN 0
Virtuelle Speicherseite eines Prozesses
Physische Speicherseite eines Prozesses
Physische Speicherseite die von mehreren Prozessen gemeinsam genutzt wird



Virtuelle Speicherseite eines Prozesses



Physische Speicherseite eines Prozesses



Physische Speicherseite die von mehreren Prozessen gemeinsam genutzt wird

Page Table

Page-Größe bei ix86 Systemen beträgt 4 Kilobyte, also 0x1000 B.

Virtuelle Adresse
0x00484456

Physische Adresse
0x00089456

Prozessor extrahiert
Offset und VPFN

Prozessor generiert
physische Adresse
und greift darauf zu

Prozessor greift auf
Page Table 0x484 zu

VPFN 0x484

Page Offset 0x456

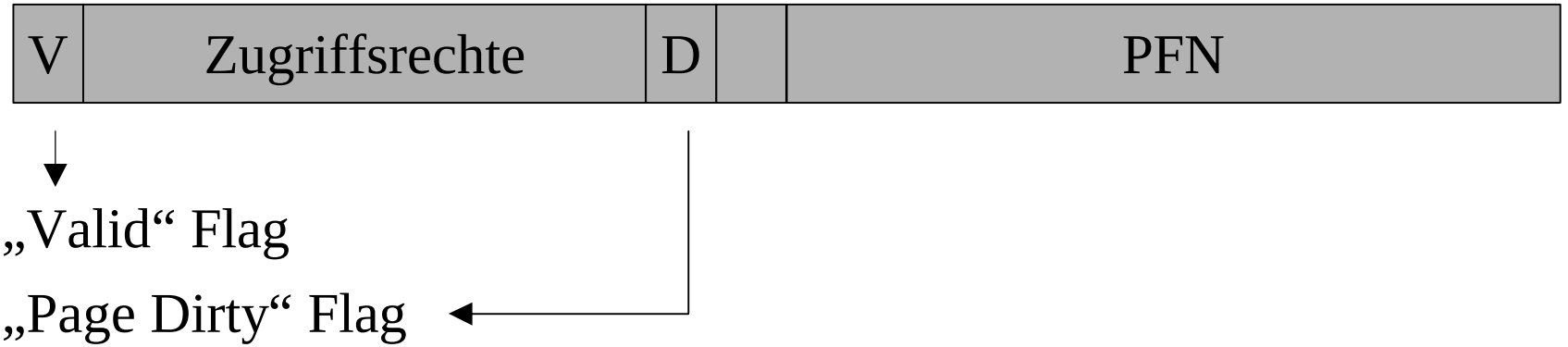
PFN 0x89

Page Offset 0x456

Inhalt

- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

Ein einzelner Page Table Eintrag eines Prozesses



Ein einzelner Page Table Eintrag eines Prozesses

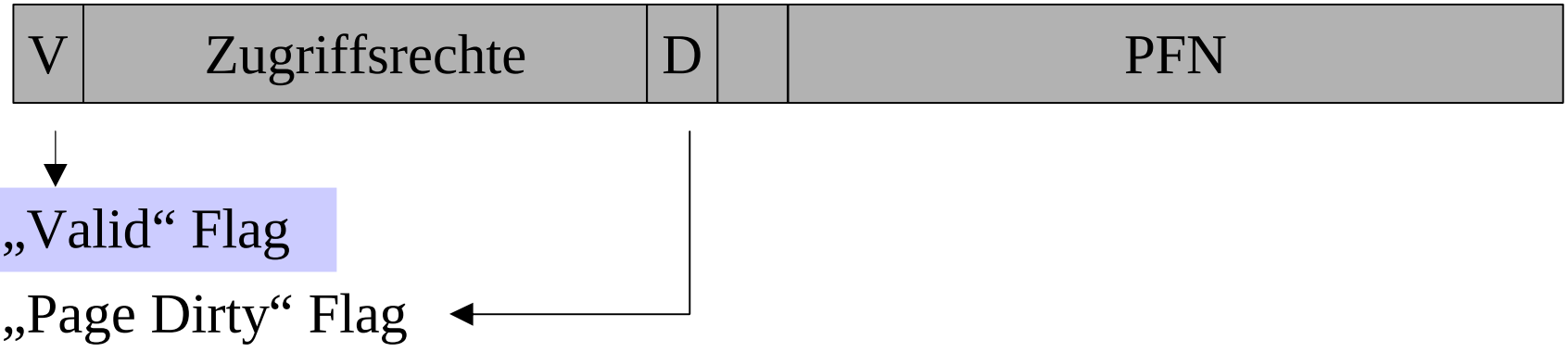
„Valid“ Flag

„Page Dirty“ Flag

Valid gesetzt & PFN gesetzt → Seite vorhanden

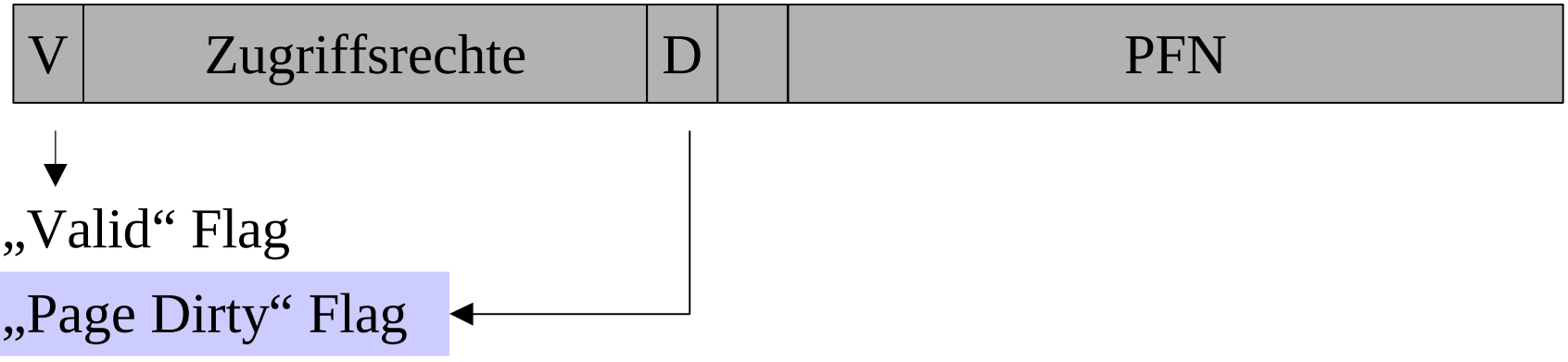
Valid gelöscht & PFN gelöscht → Seite ungültig (Page fault)

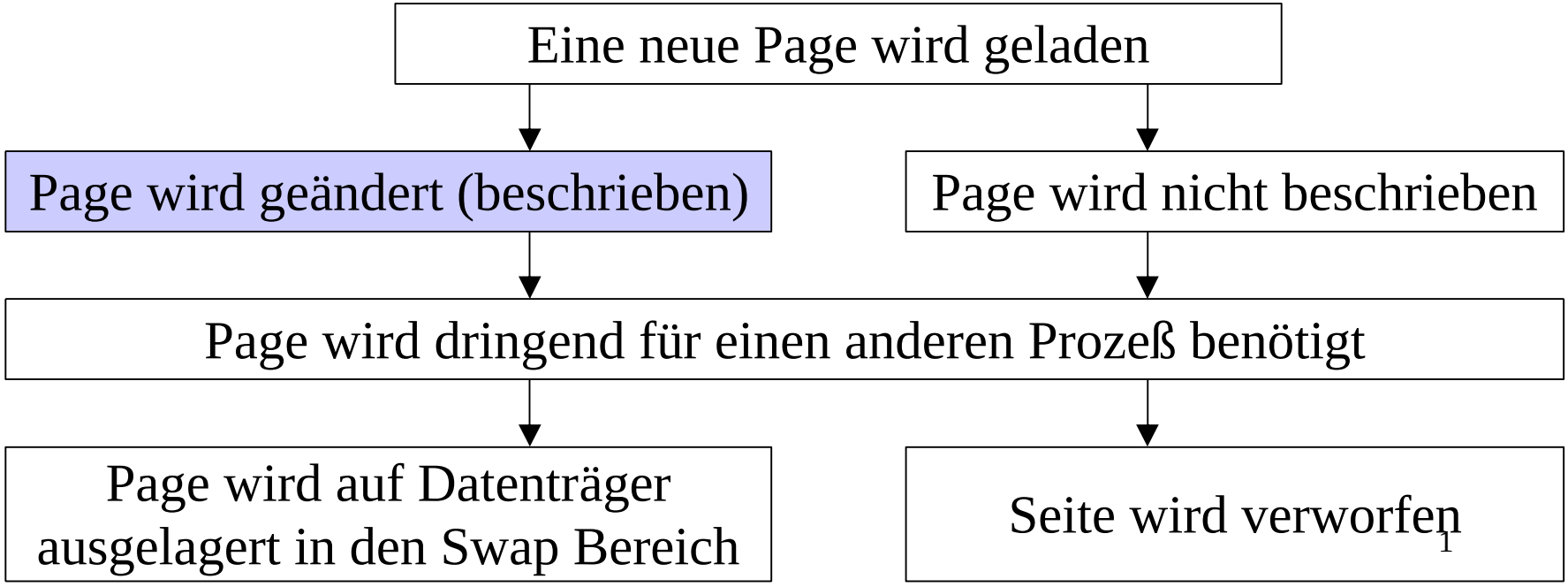
Ein einzelner Page Table Eintrag eines Prozesses

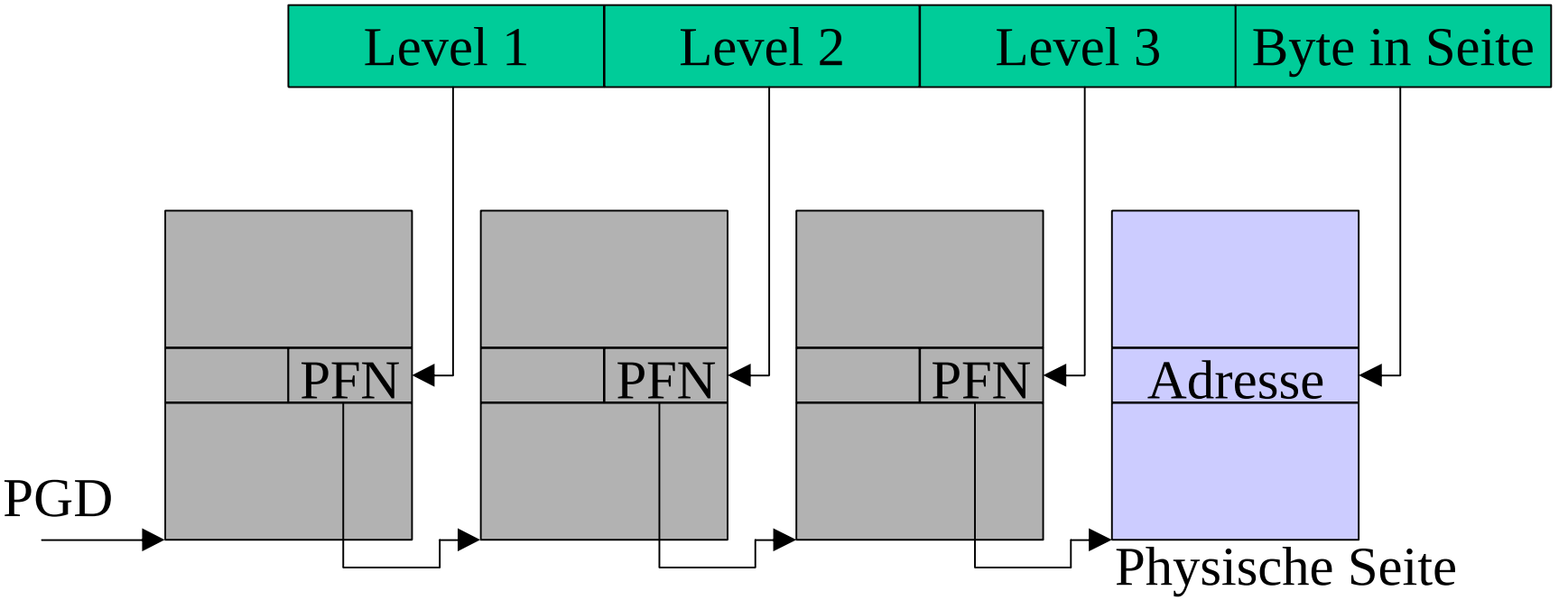


Valid gesetzt	&	PFN gesetzt	→	Seite vorhanden
Valid gelöscht	&	PFN gelöscht	→	Seite ungültig (Page fault)
Valid gesetzt	&	PFN gelöscht	→	Page fault: Demand Paging
Valid gelöscht	&	PFN gesetzt	→	Page fault: Seite im Swap

Ein einzelner Page Table Eintrag eines Prozesses





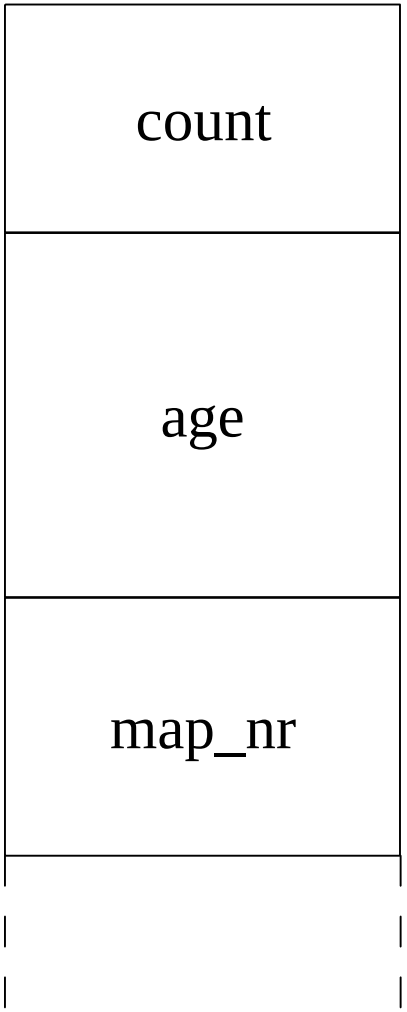


PGD: „Page Global Directory“.Vom Kernel initialisiert. Weist auf den ersten Page Table Level.

Inhalt

- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

„mem_map_t“ oder auch „Page“ Struktur:

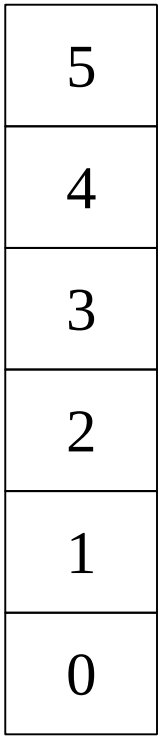


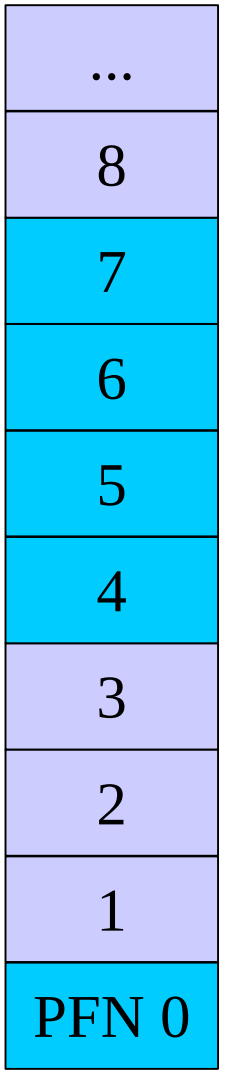
Anzahl der Benutzer, die die Seite benutzen

Alter der Seite. Entscheidet, ob die Seite verworfen bzw. ausgelagert wird, wenn wieder Speicherplatz benötigt wird.

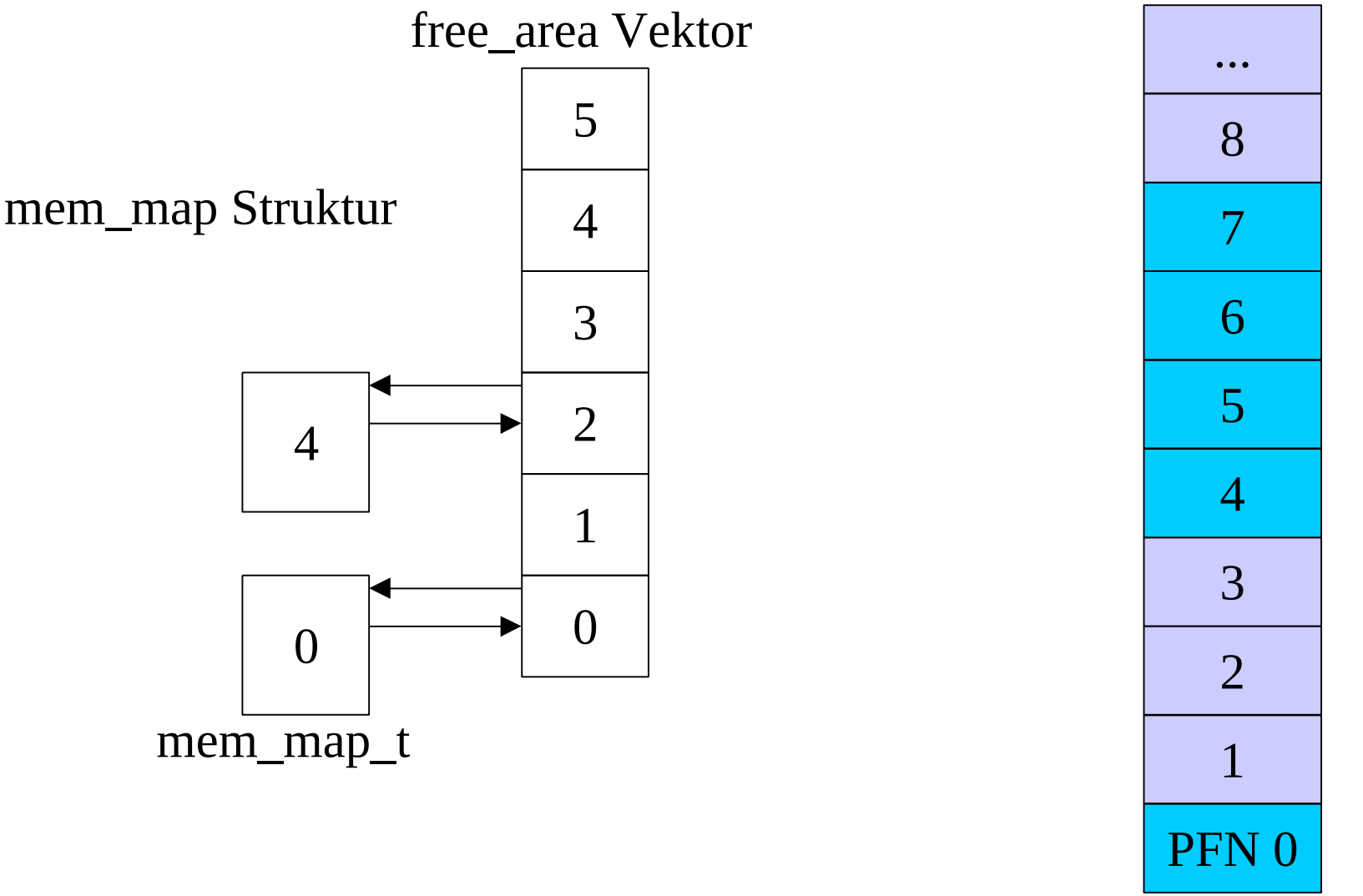
Physische PFN

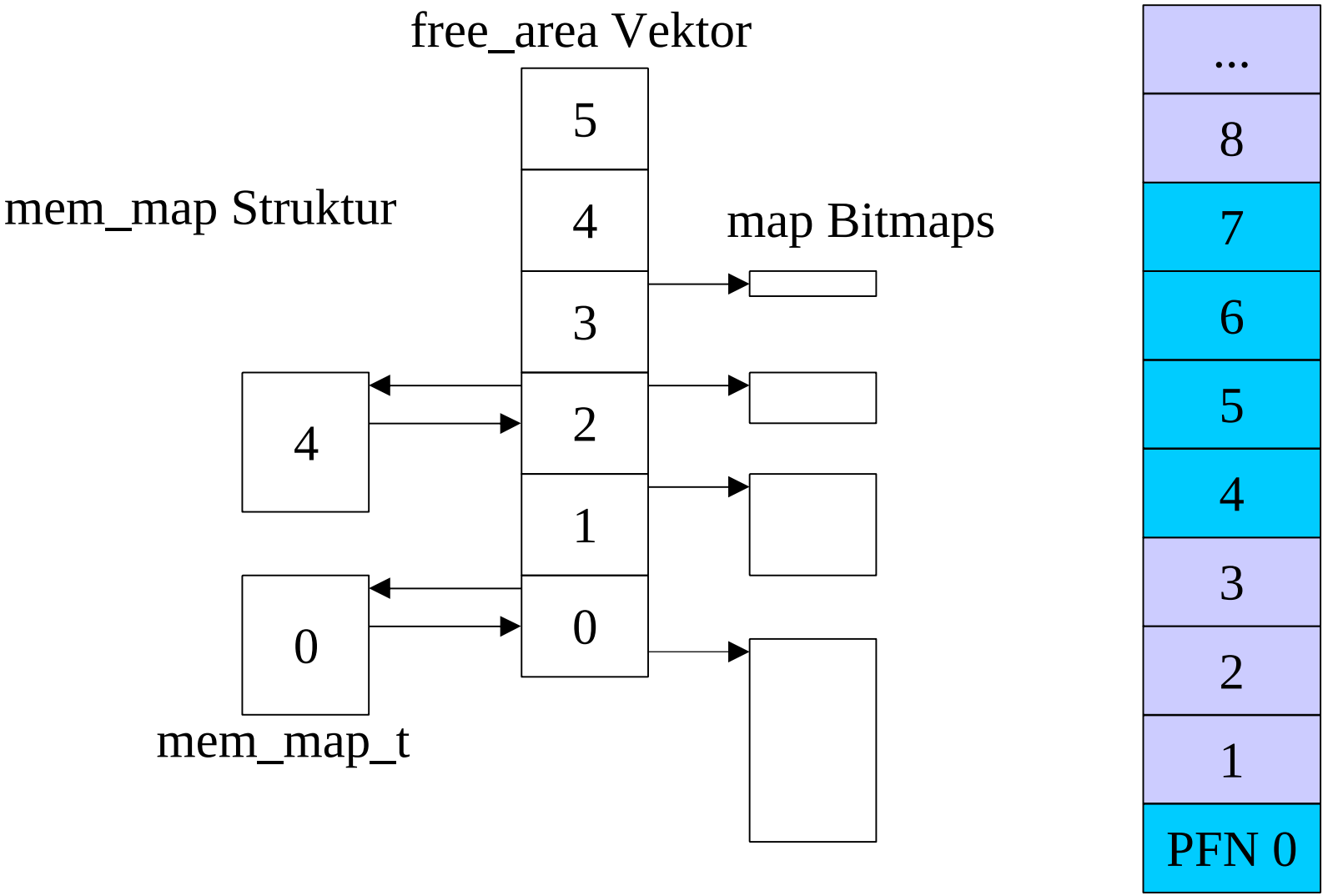
free_area Vektor





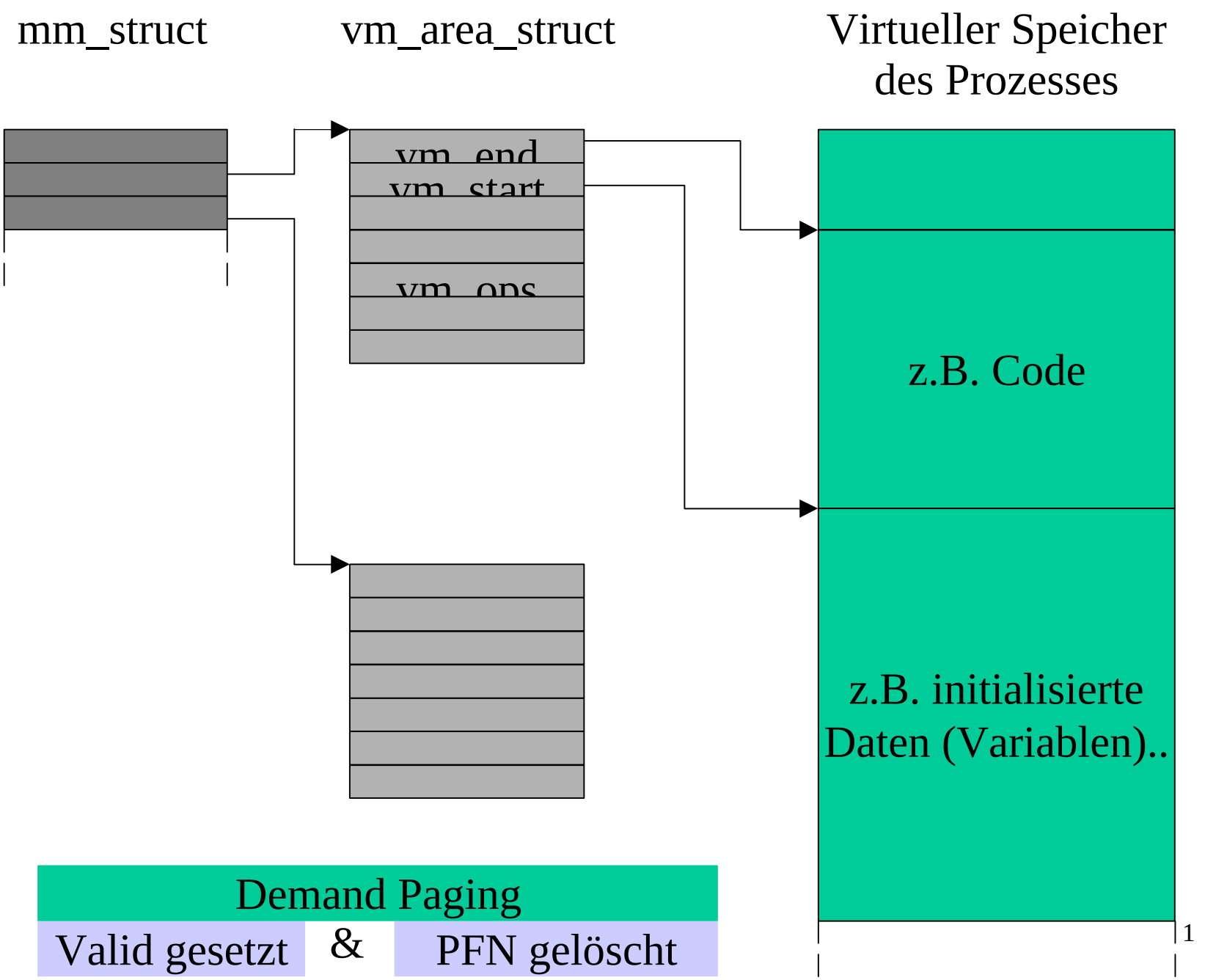
Freie Speicherseite im Hauptspeicher (nicht alloziert)


free_area Vektor
mem_map Struktur
5
4
3
2
1
0
4
0
mem_map_t
...
8
7
6
5
4
3
2
1
PFN 0
Freie Speicherseite im Hauptspeicher (nicht alloziert)


free_area Vektor
mem_map Struktur
map Bitmaps
5
4
3
2
1
0
4
0
mem_map_t
...
8
7
6
5
4
3
2
1
PFN 0
Freie Speicherseite im Hauptspeicher (nicht alloziert)

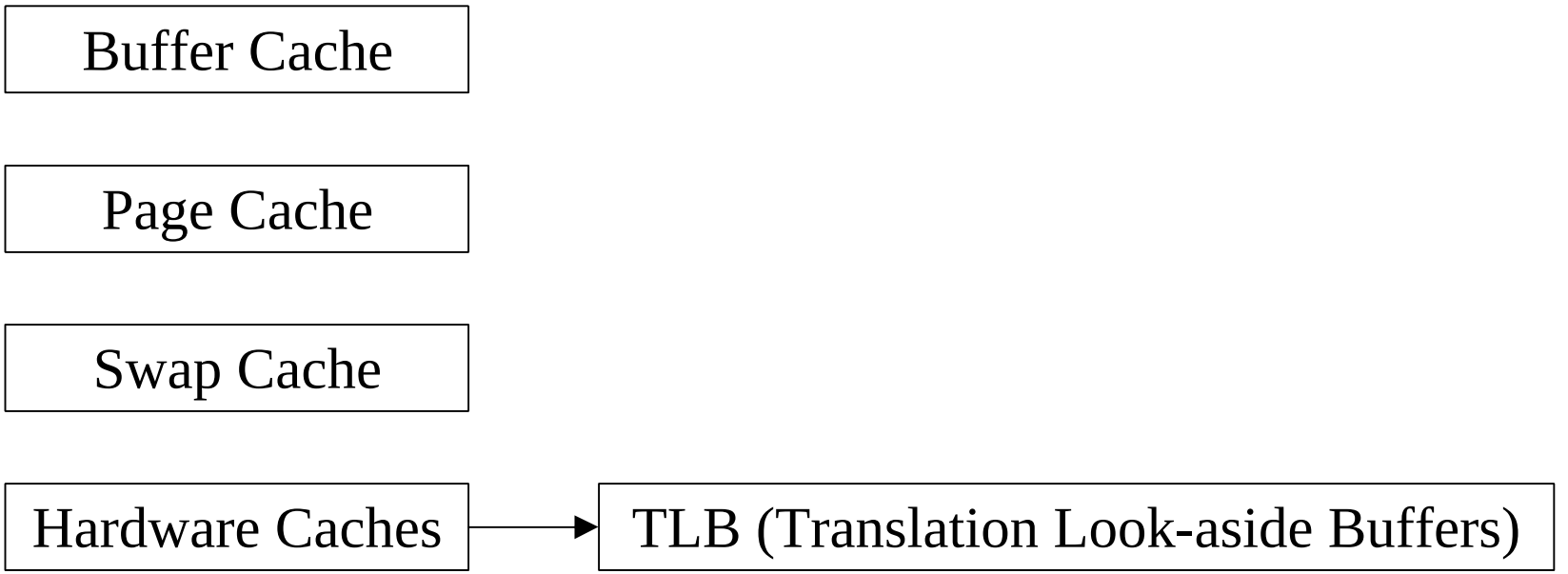
Inhalt

- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping und Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

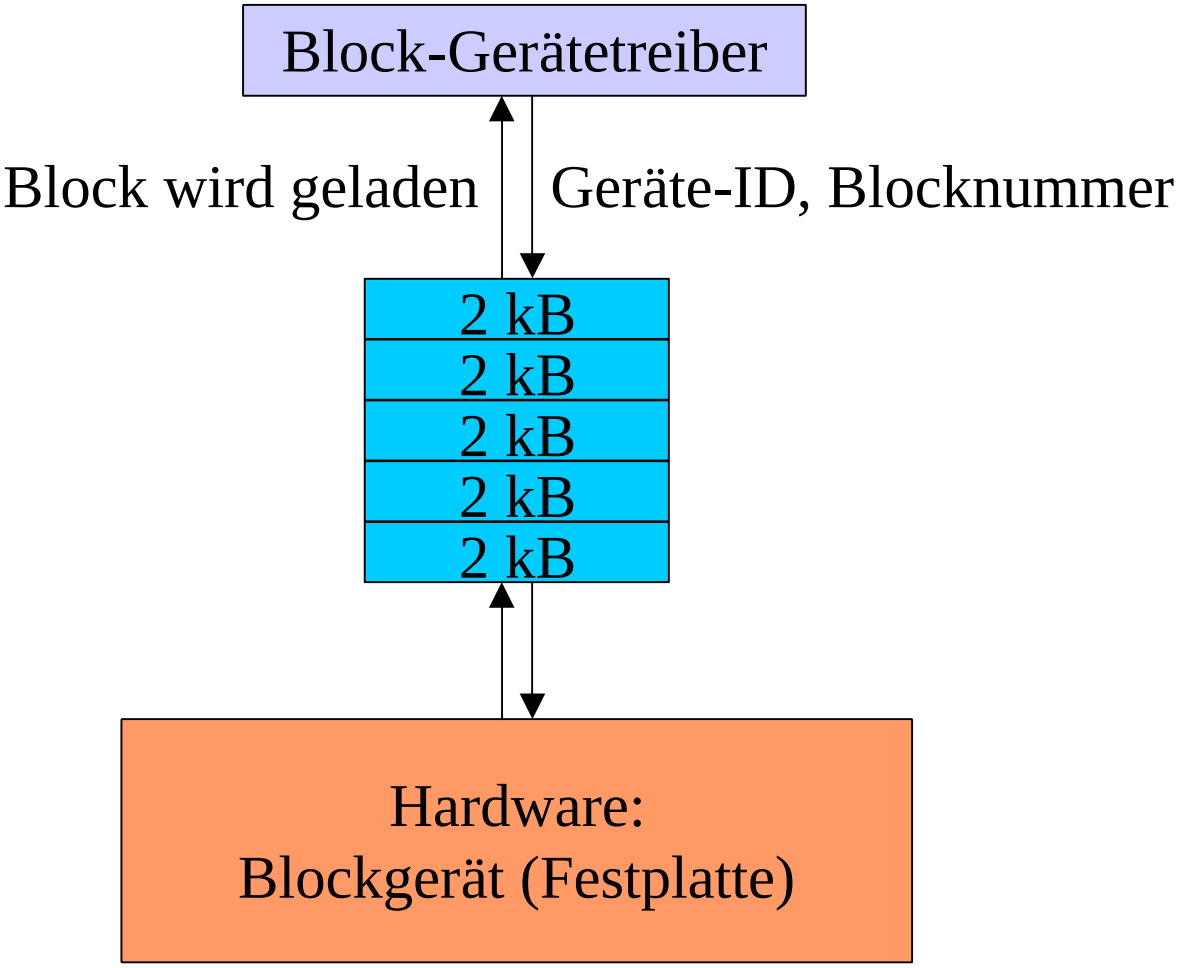

mm_struct
vm_area_struct
Virtueller Speicher des Prozesses
vm_end
vm_start
vm_ops
z.B. Code
z.B. initialisierte Daten (Variablen)..
Demand Paging
Valid gesetzt
&
PFN gelöscht

Inhalt

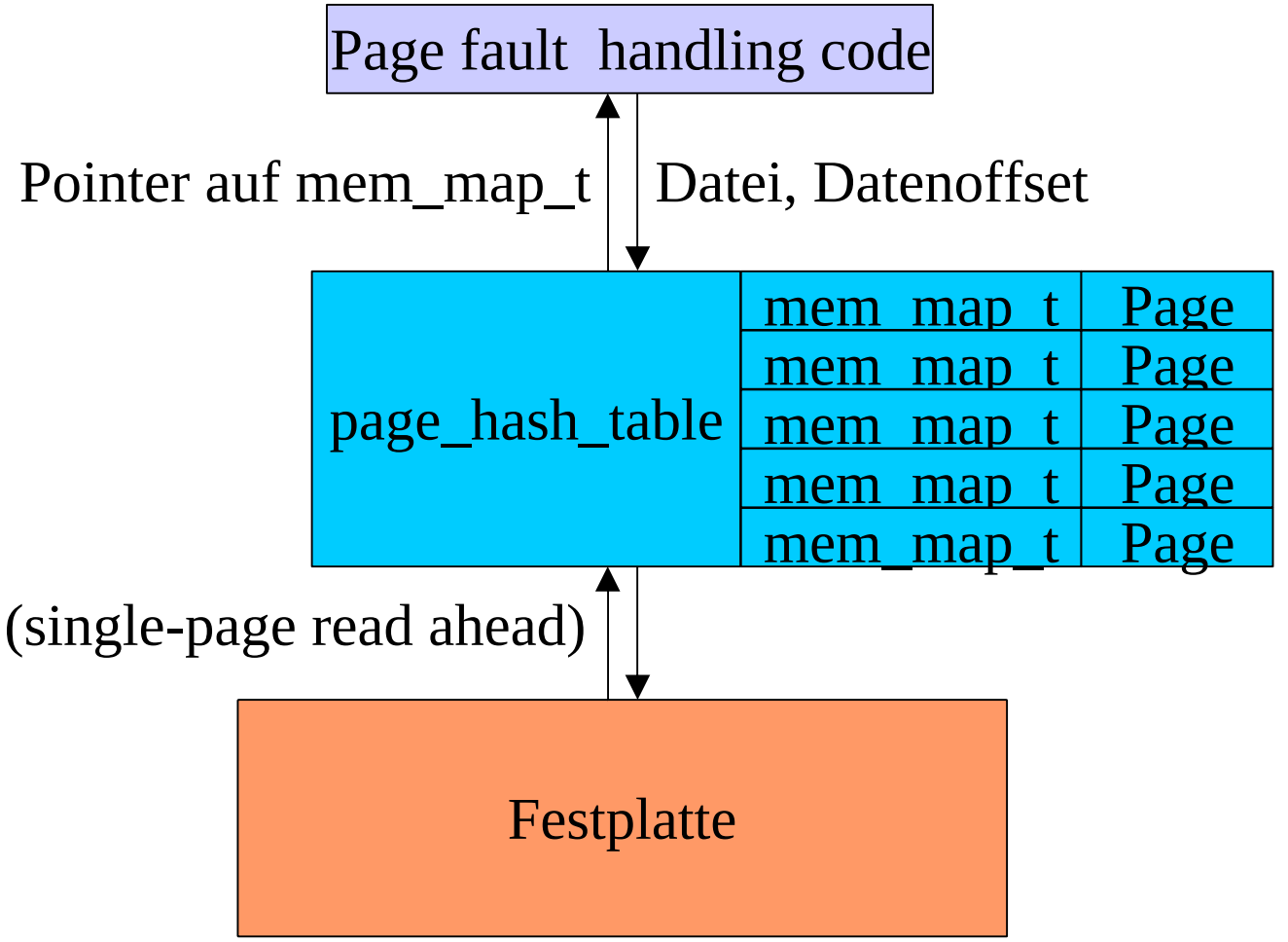
- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten


Buffer Cache
Page Cache
Swap Cache
Hardware Caches
TLB (Translation Look-aside Buffers)

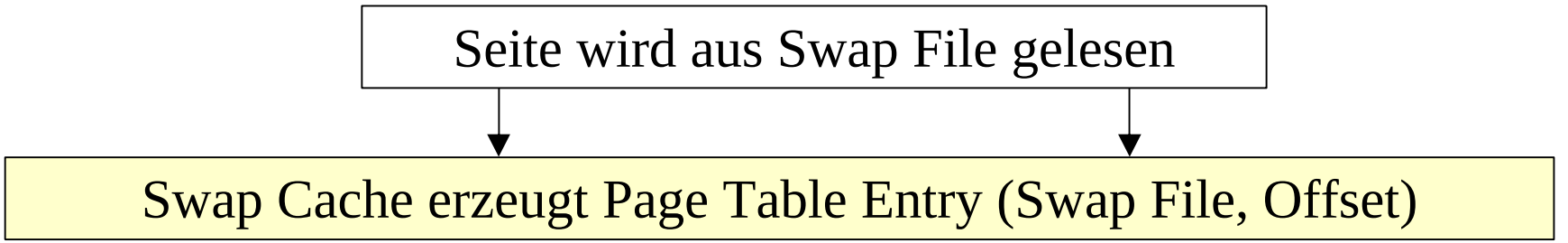
Buffer Cache



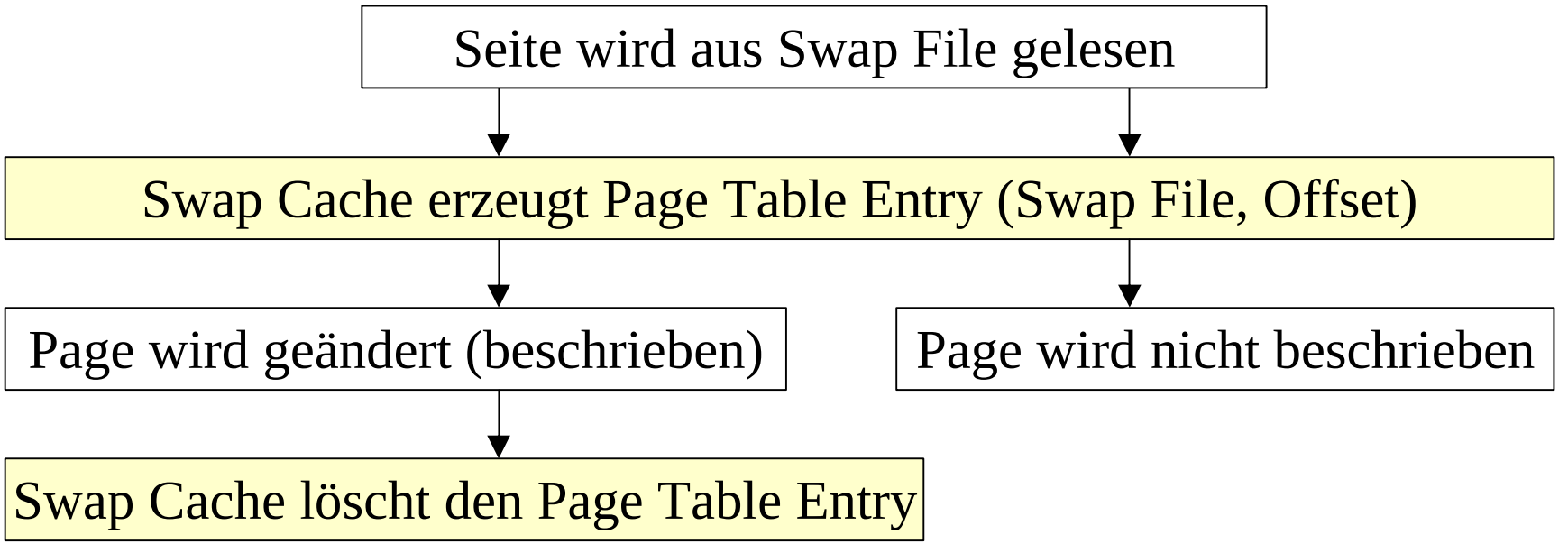
Page Cache


Page fault handling code
Pointer auf mem_map_t
Datei, Datenoffset
page_hash_table
mem_map_t
Page
mem_map_t
Page
mem_map_t
Page
mem_map_t
Page
mem_map_t
Page
(single-page read ahead)
Festplatte

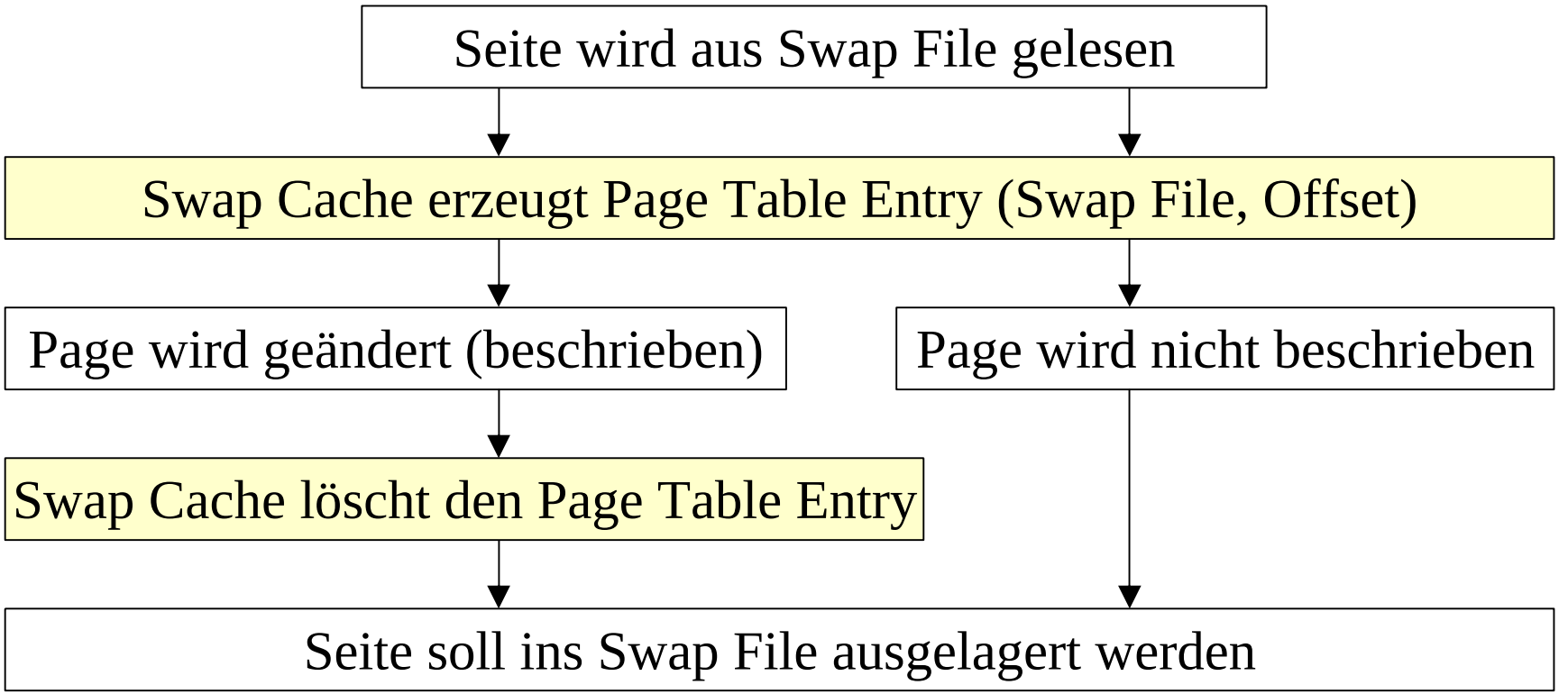
Swap Cache

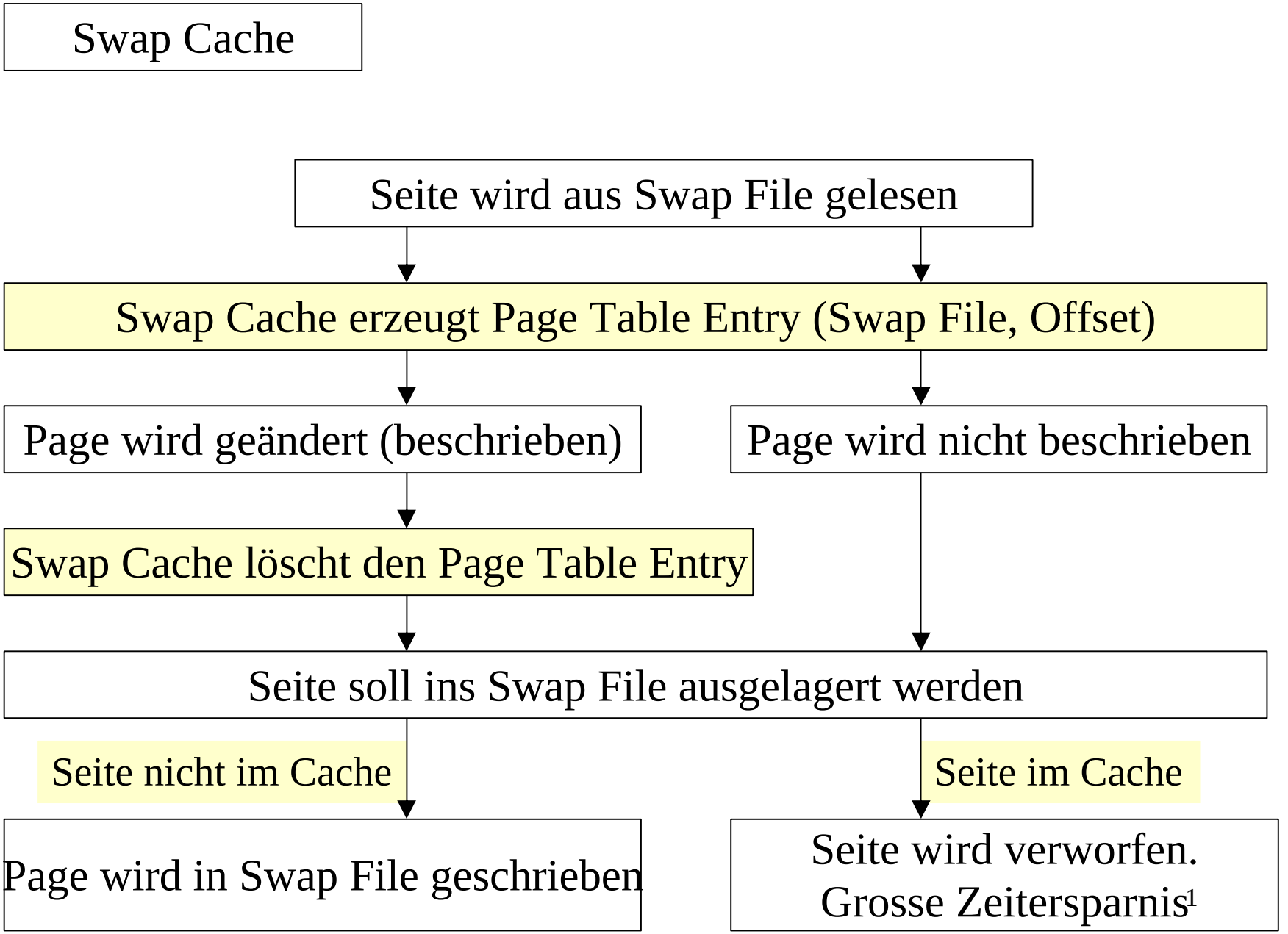

Seite wird aus Swap File gelesen
Swap Cache erzeugt Page Table Entry (Swap File, Offset)

Swap Cache

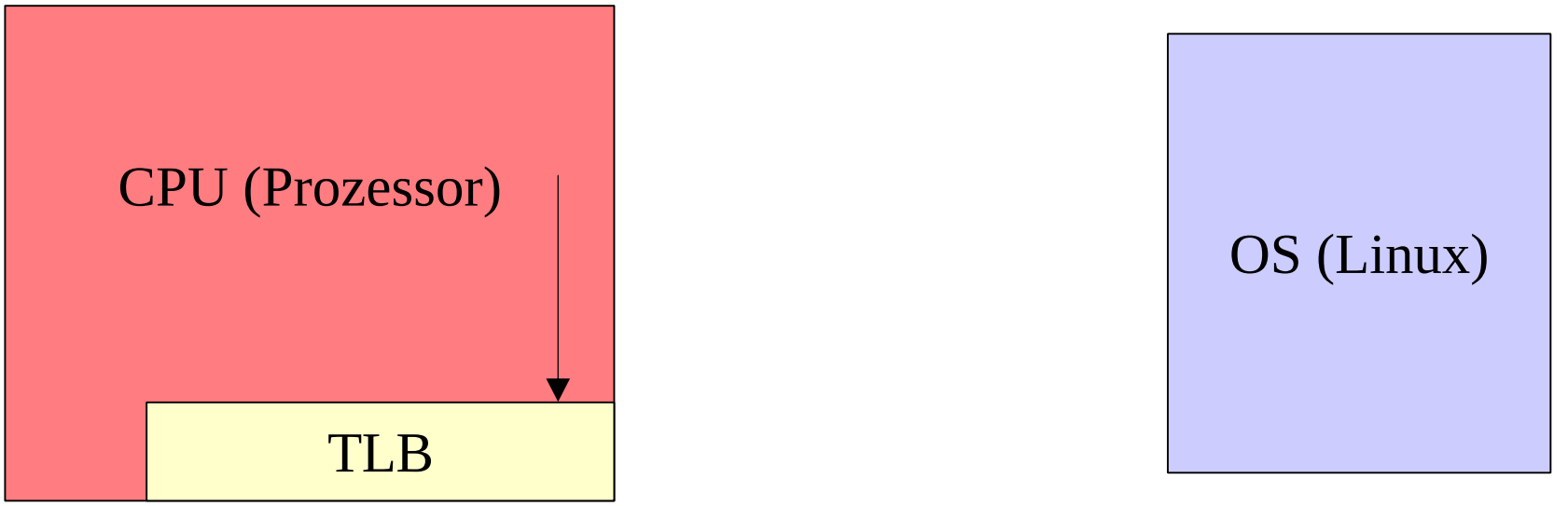

Seite wird aus Swap File gelesen
Swap Cache erzeugt Page Table Entry (Swap File, Offset)
Page wird geändert (beschrieben)
Page wird nicht beschrieben
Swap Cache löscht den Page Table Entry

Swap Cache

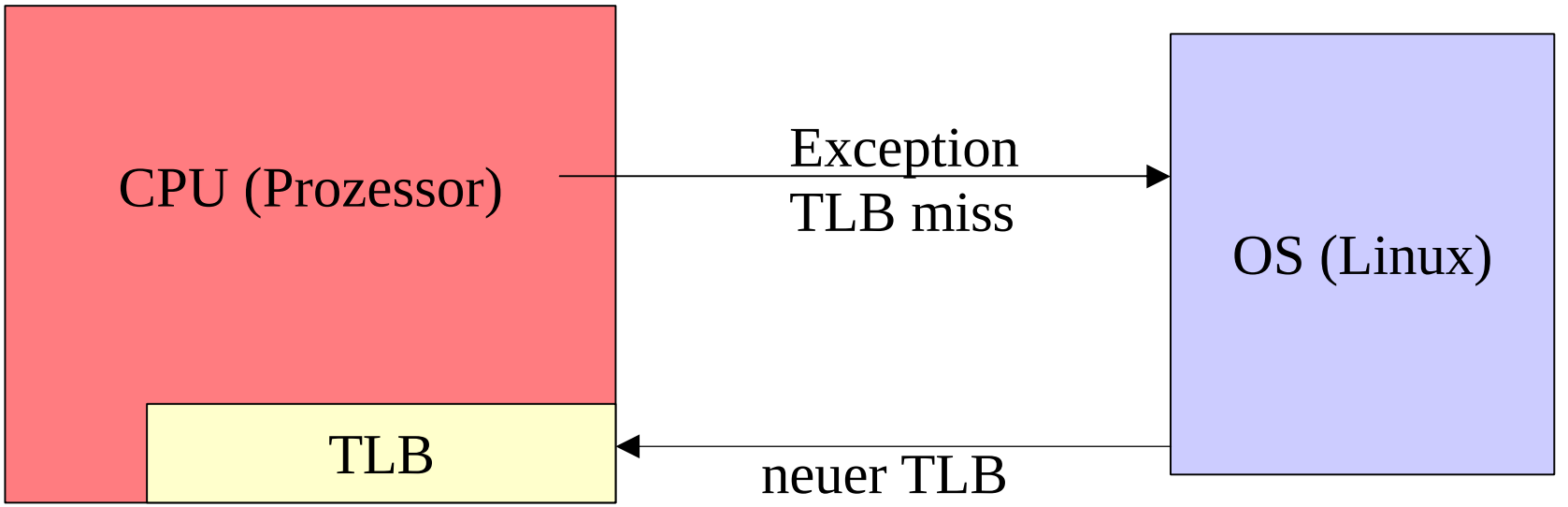

Seite wird aus Swap File gelesen
Swap Cache erzeugt Page Table Entry (Swap File, Offset)
Page wird geändert (beschrieben)
Page wird nicht beschrieben
Swap Cache löscht den Page Table Entry
Seite soll ins Swap File ausgelagert werden


Swap Cache
Seite wird aus Swap File gelesen
Swap Cache erzeugt Page Table Entry (Swap File, Offset)
Page wird geändert (beschrieben)
Page wird nicht beschrieben
Swap Cache löscht den Page Table Entry
Seite soll ins Swap File ausgelagert werden
Seite nicht im Cache
Seite im Cache
Page wird in Swap File geschrieben
Seite wird verworfen.
Grosse Zeitersparnis[1]

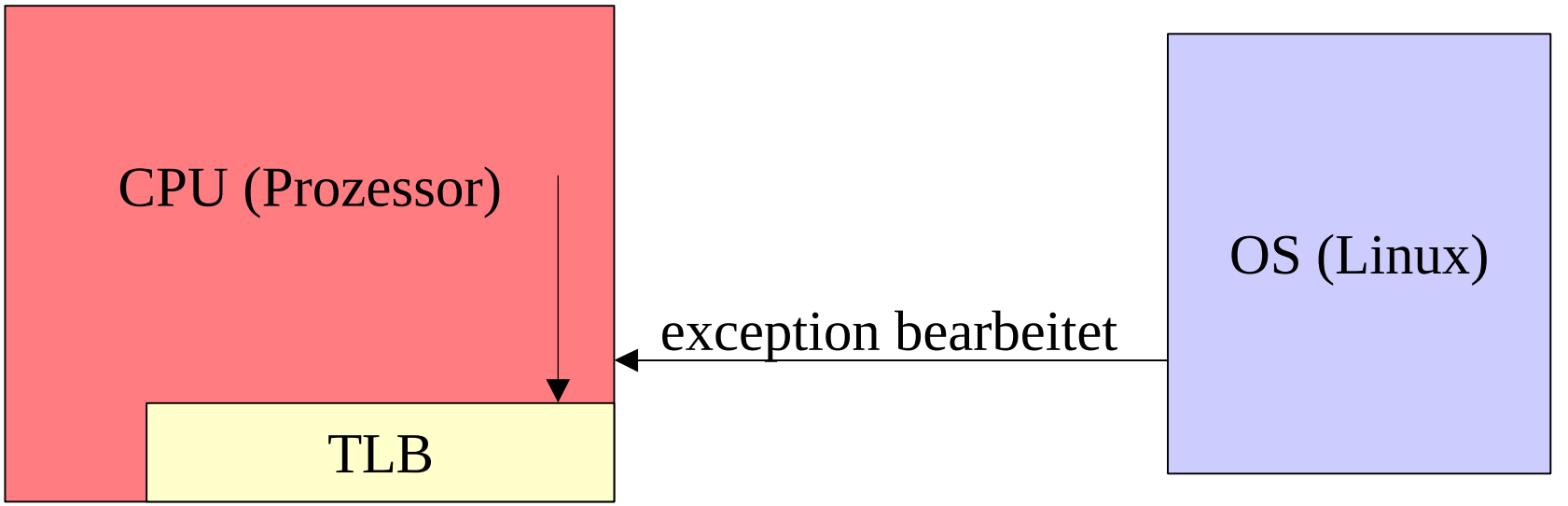
Hardware Caches → TLB (Translation Look-aside Buffers)



Hardware Caches → TLB (Translation Look-aside Buffers)



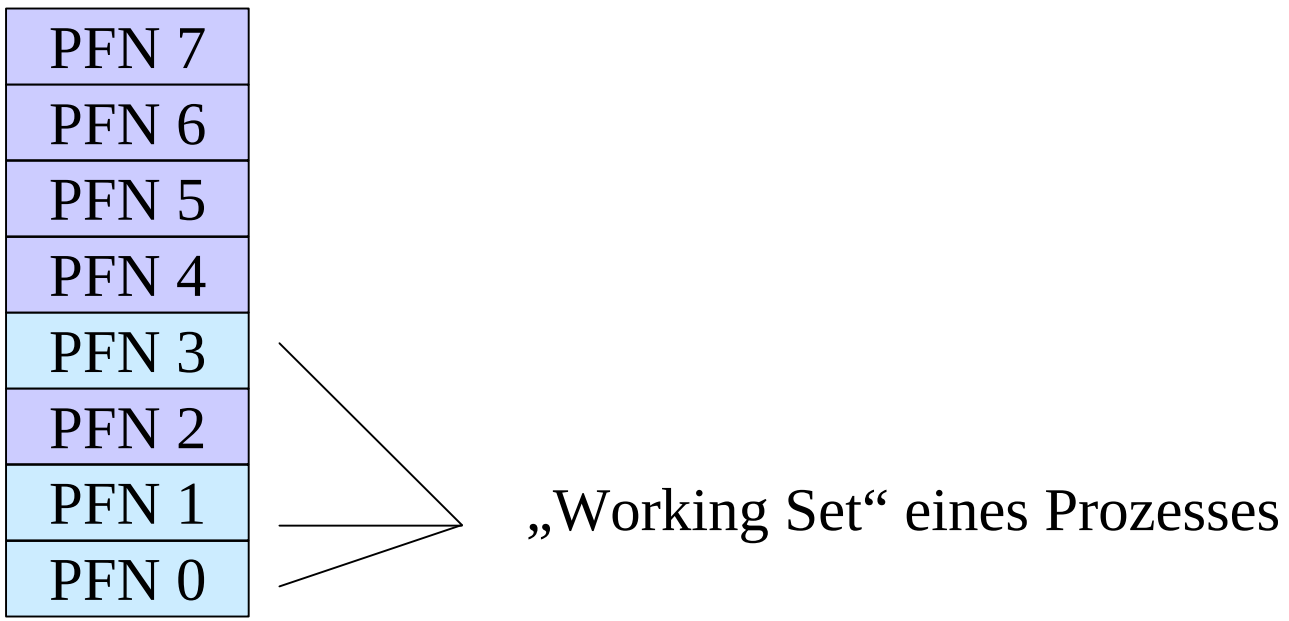
Hardware Caches → TLB (Translation Look-aside Buffers)

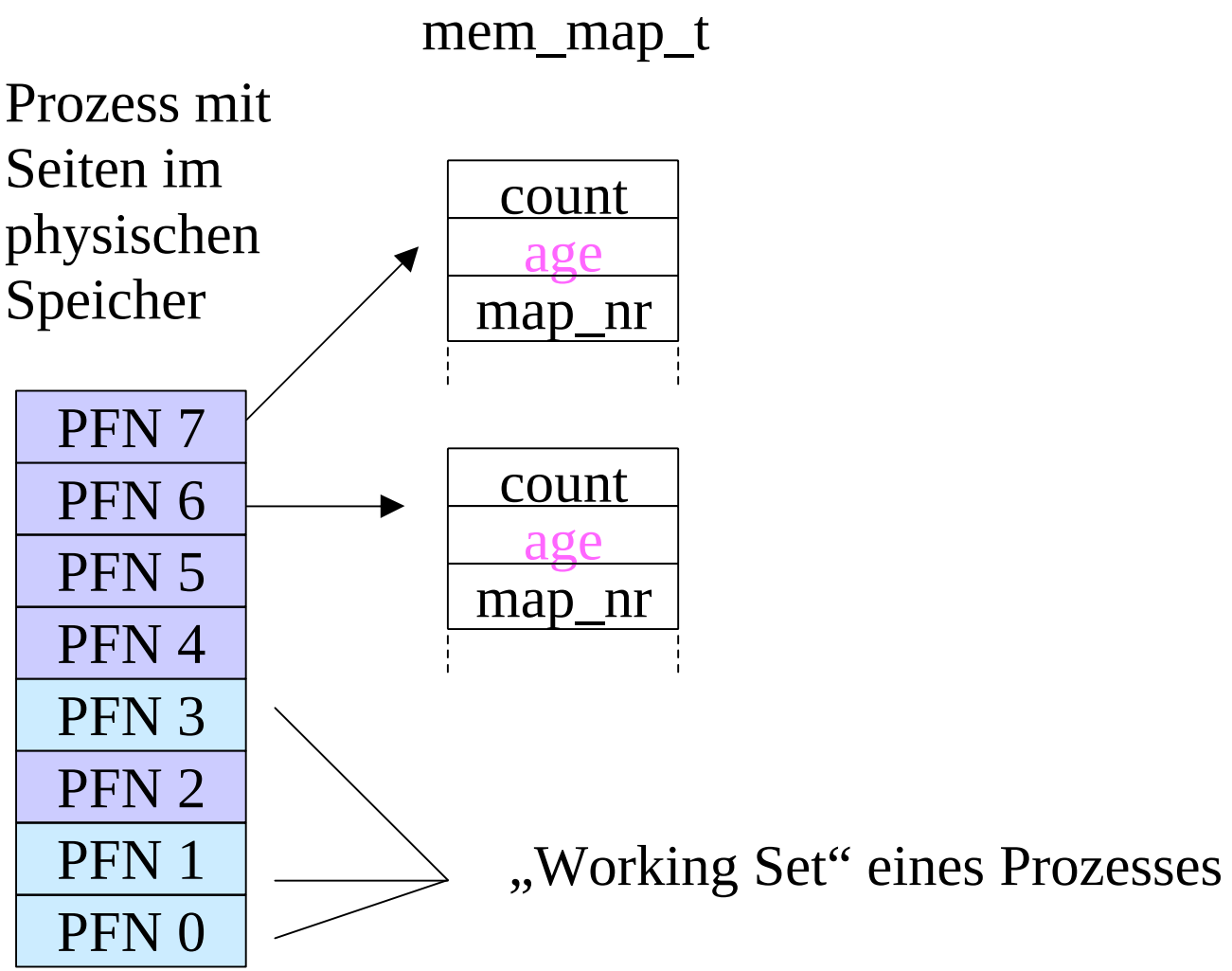


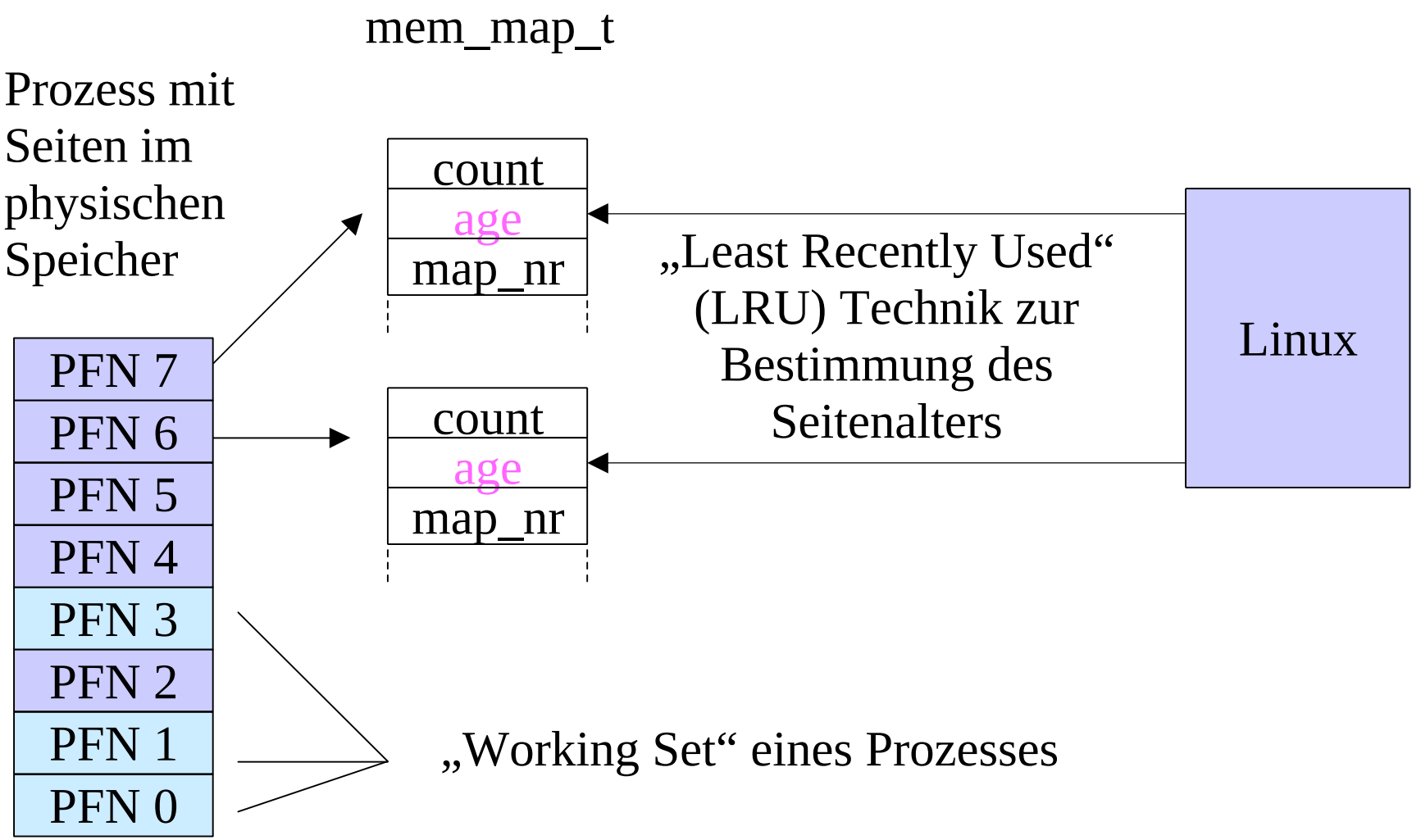
Inhalt

- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

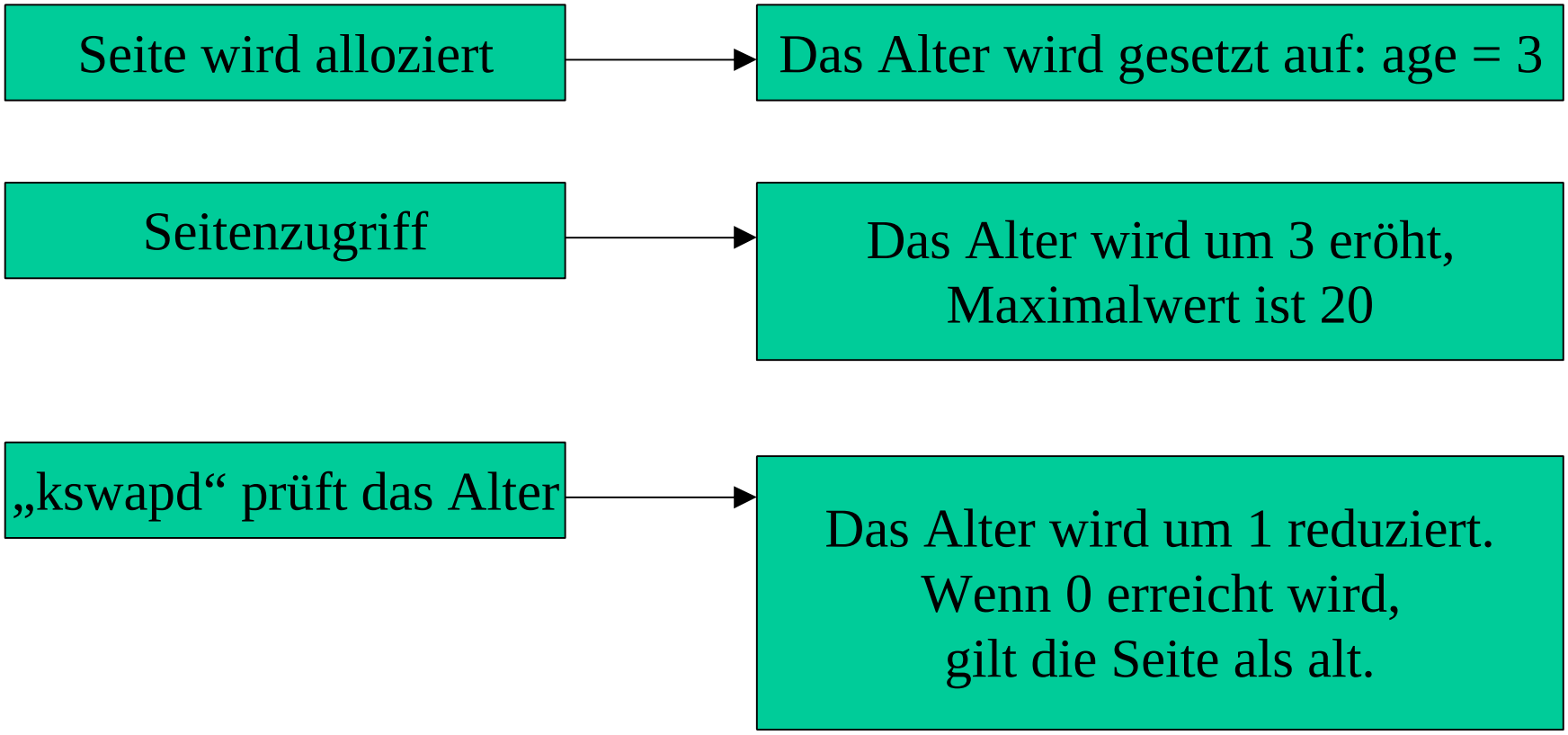
Prozess mit Seiten im physischen Speicher




mem_map_t
Prozess mit Seiten im physischen Speicher
count
age
map_nr
count
age
map_nr
PFN 7
PFN 6
PFN 5
PFN 4
PFN 3
PFN 2
PFN 1
PFN 0
„Working Set“ eines Prozesses

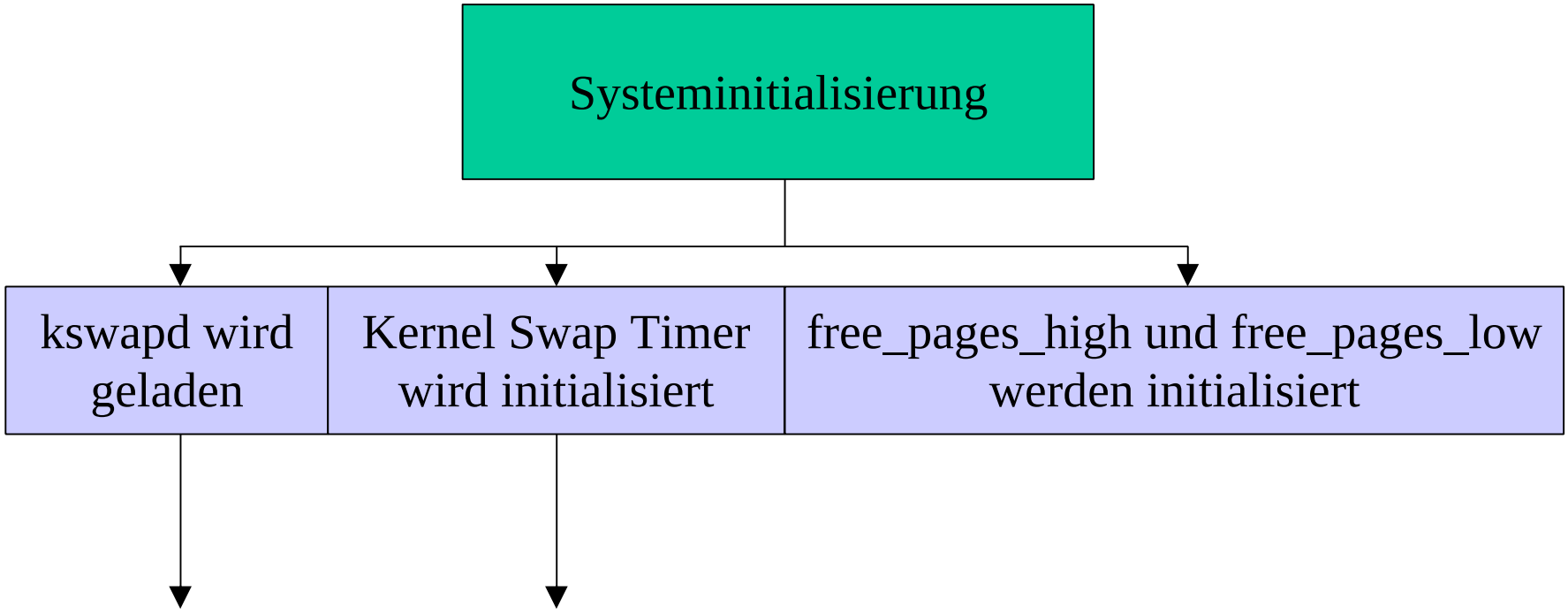

mem_map_t
Prozess mit Seiten im physischen Speicher
count
age
map_nr
count
age
map_nr
„Least Recently Used“ (LRU) Technik zur Bestimmung des Seitenalters
Linux
PFN 7
PFN 6
PFN 5
PFN 4
PFN 3
PFN 2
PFN 1
PFN 0
„Working Set“ eines Prozesses

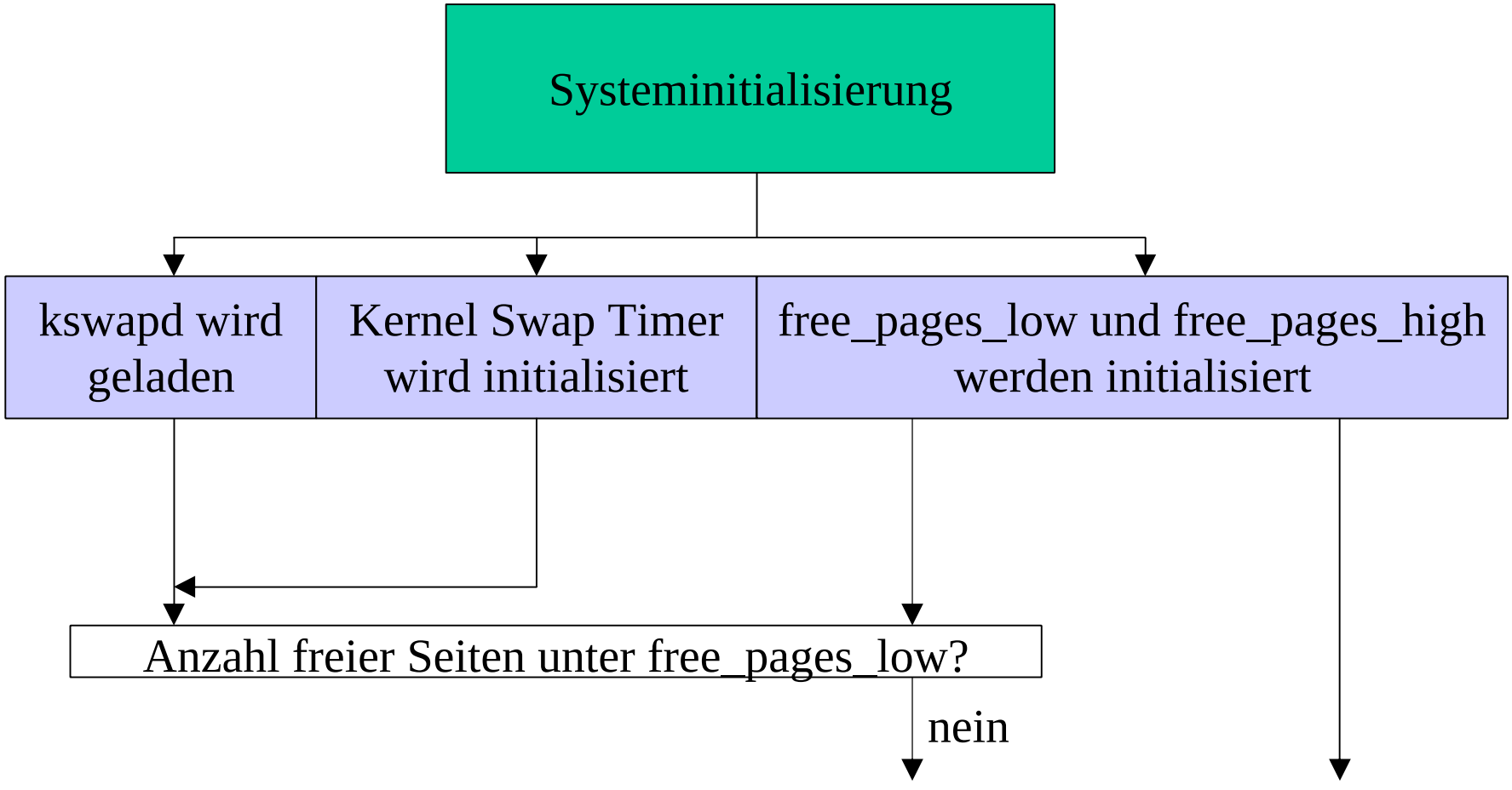
LRU im Detail:

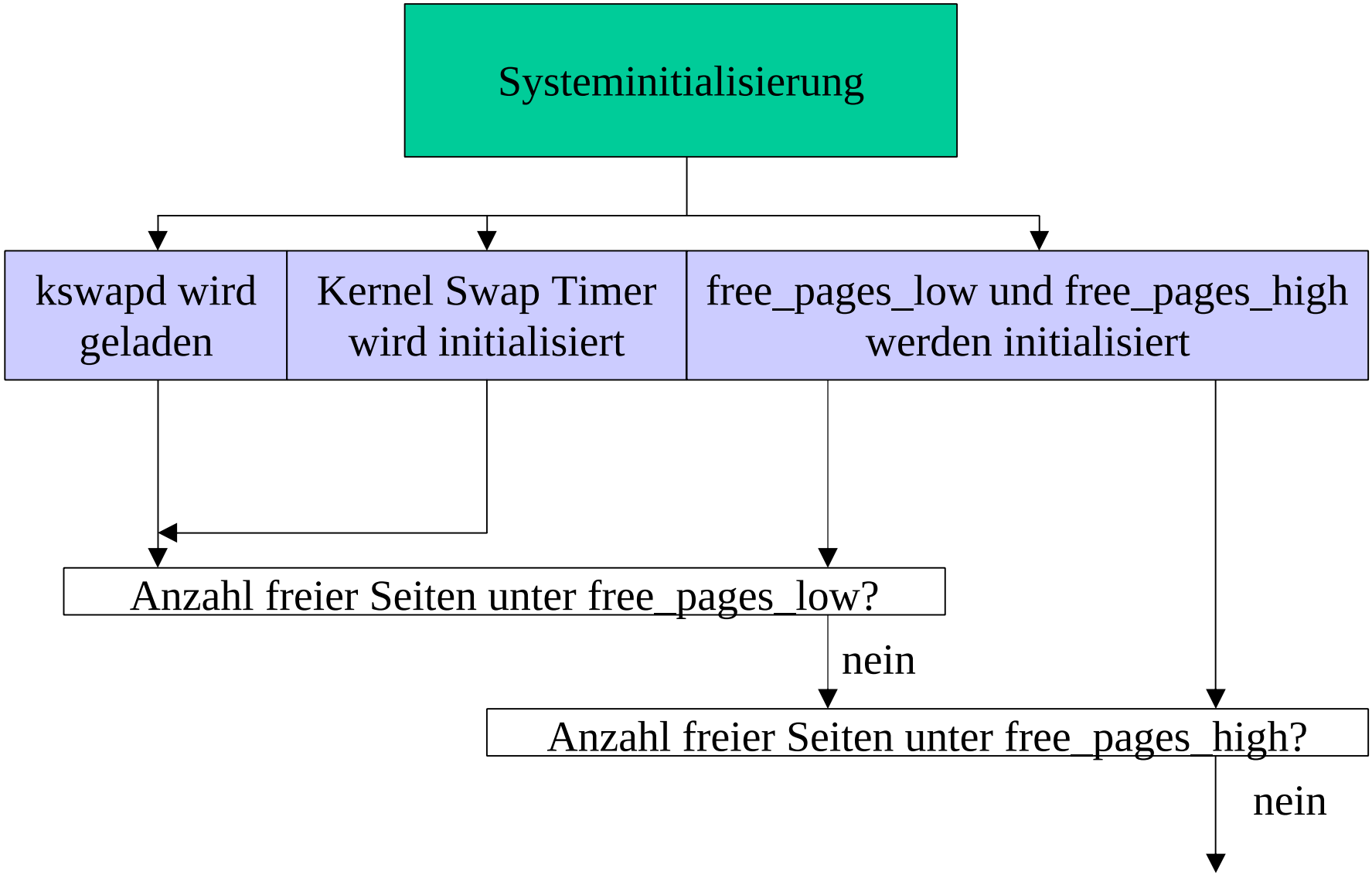


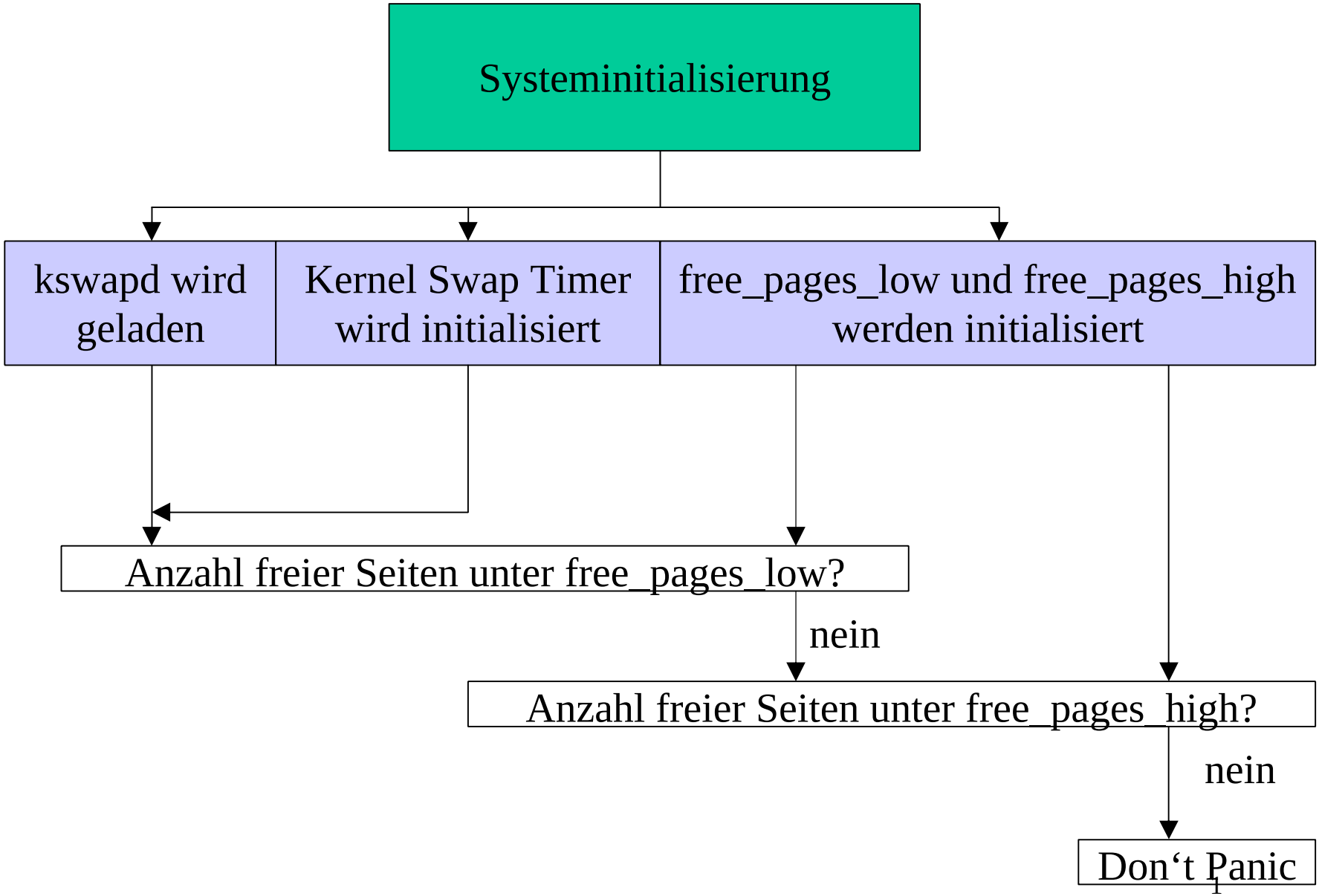
Inhalt

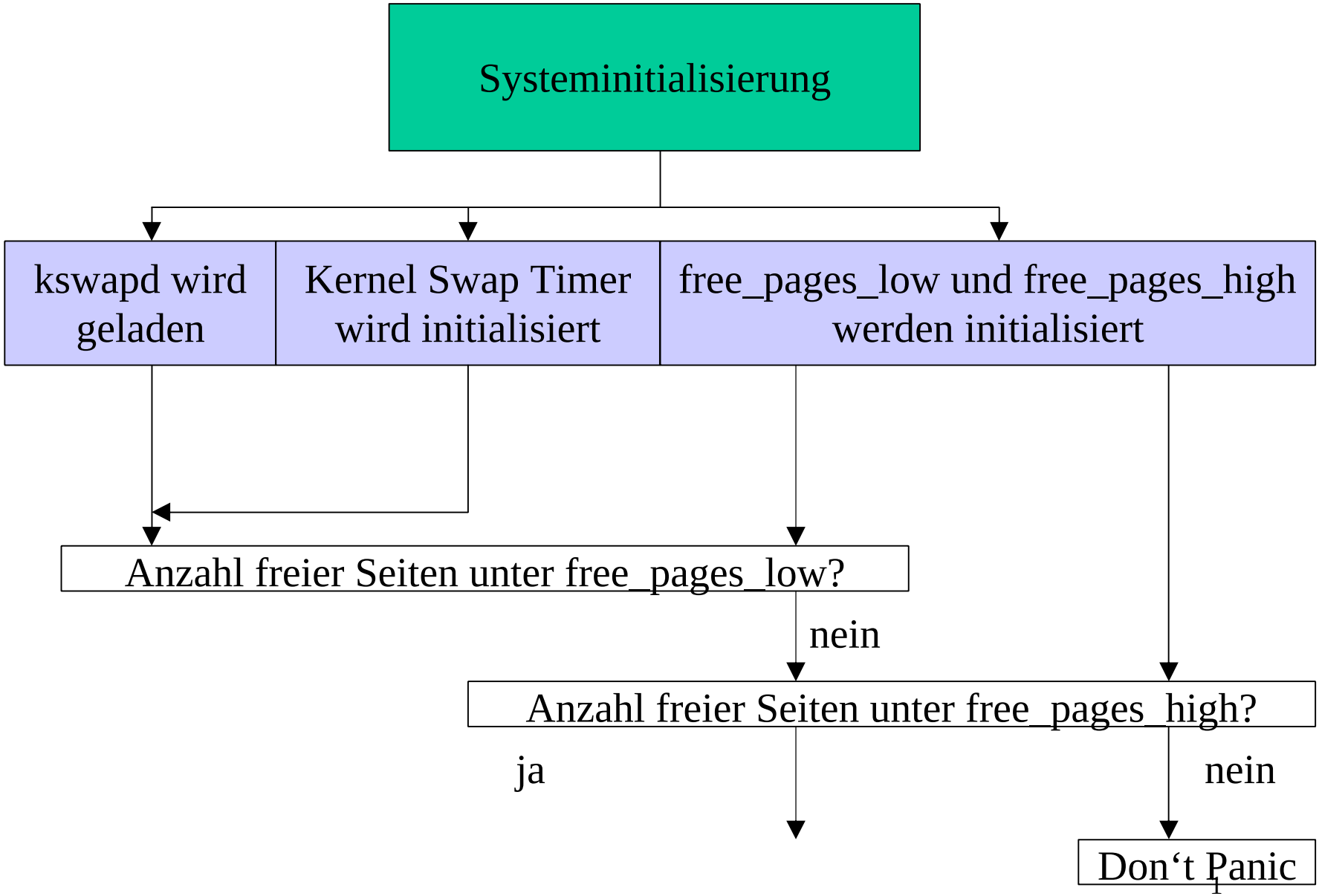
- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

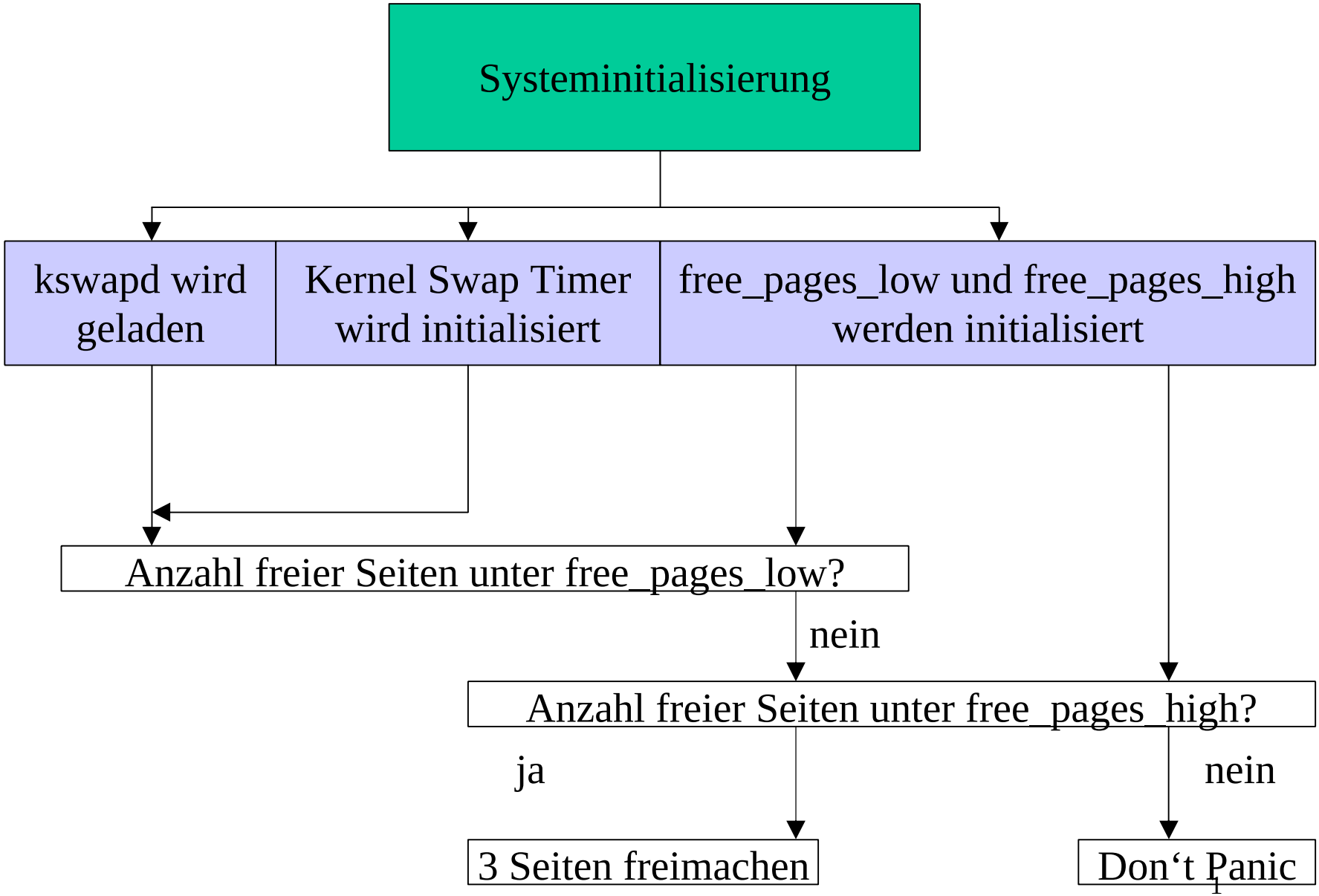

Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_high und free_pages_low werden initialisiert

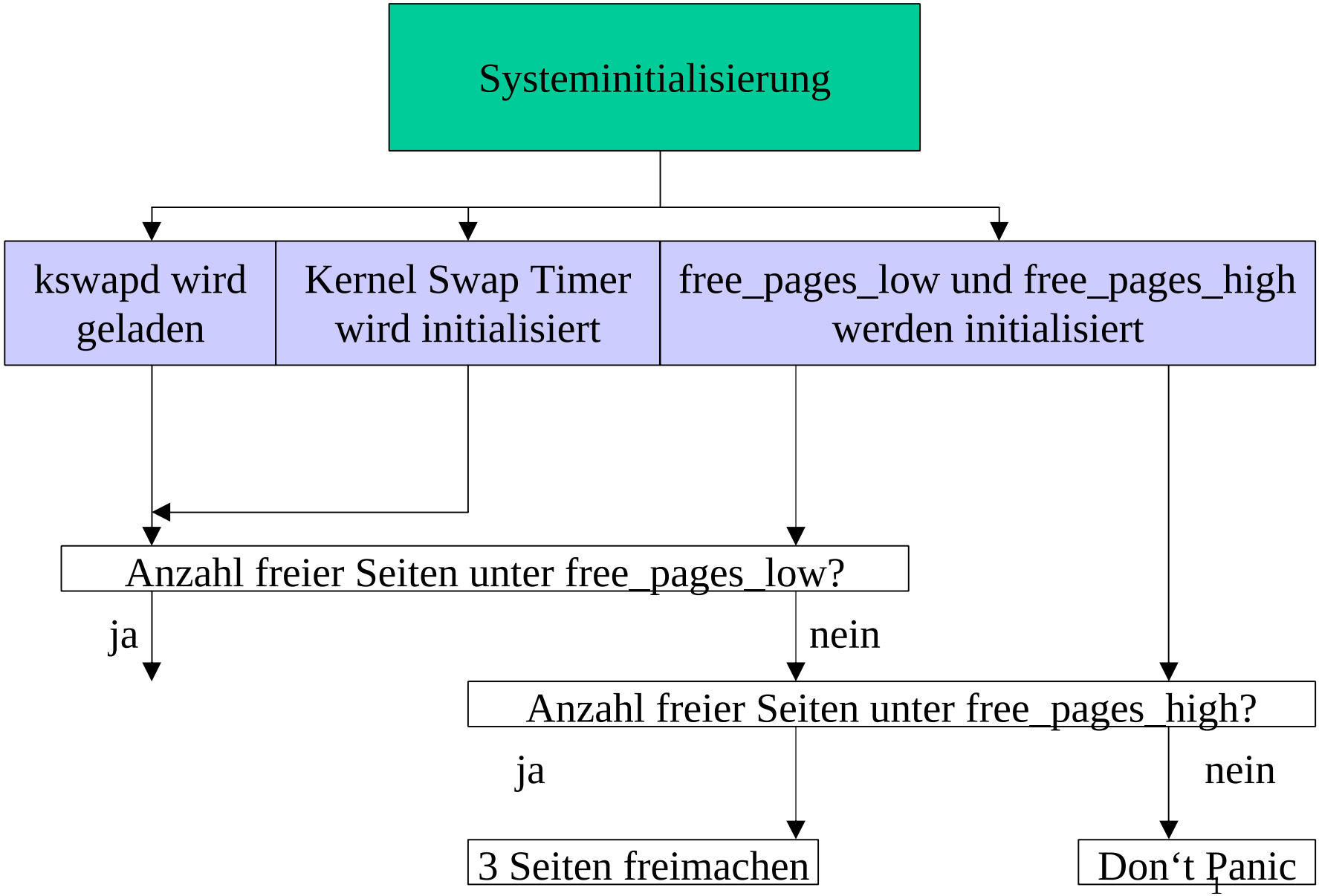

Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
nein

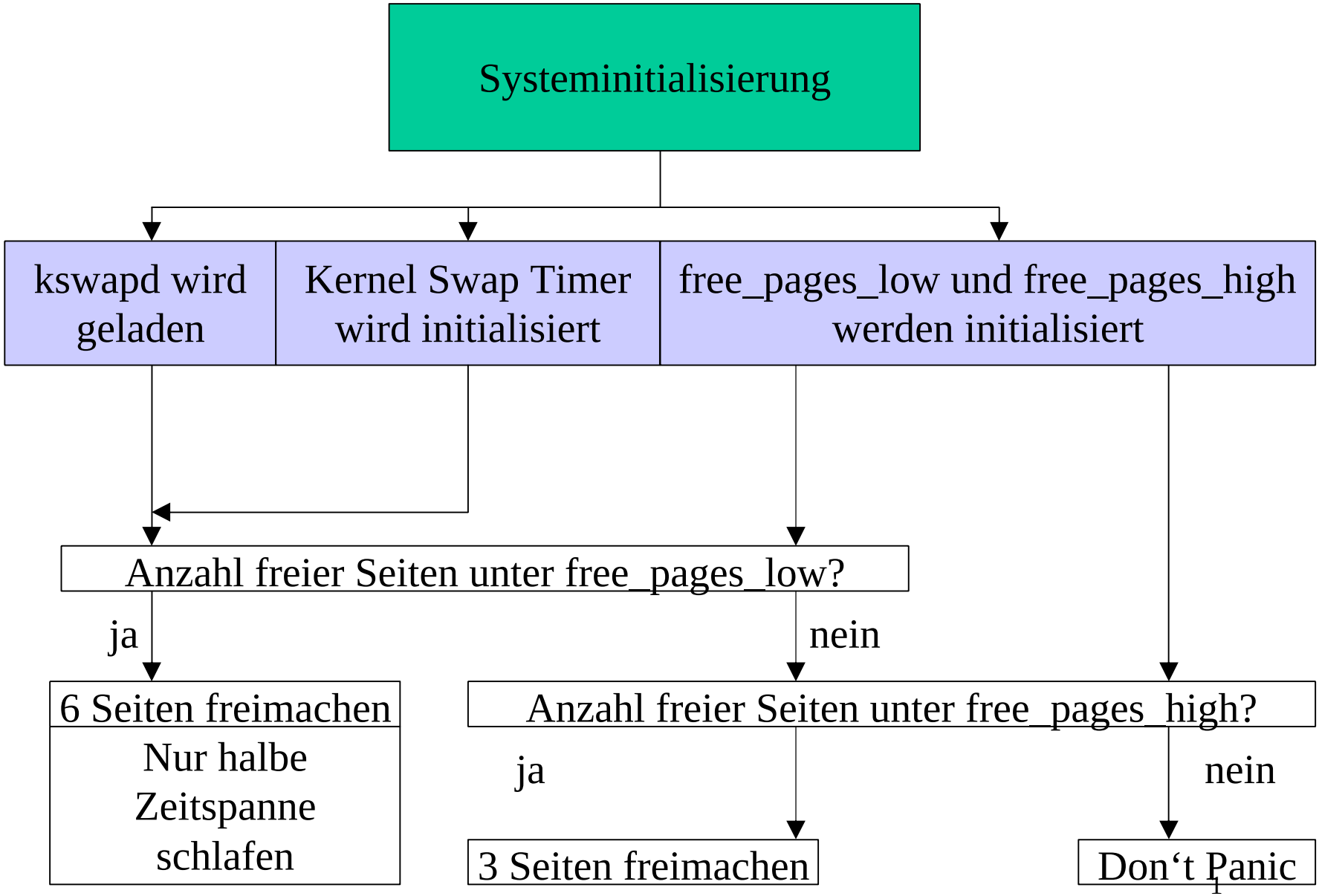

Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
nein
Anzahl freier Seiten unter free_pages_high?
nein

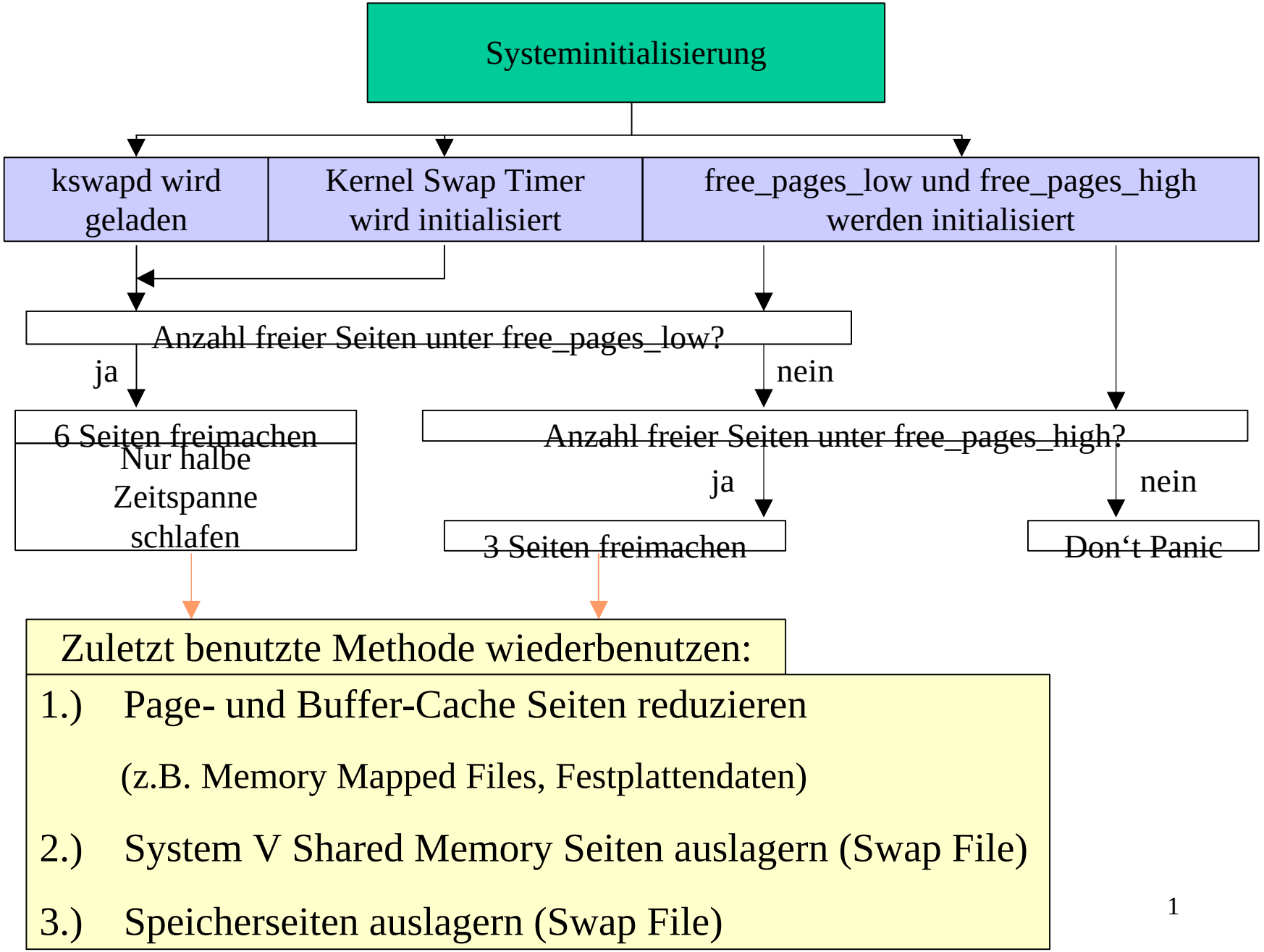

Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
nein
Anzahl freier Seiten unter free_pages_high?
nein
Don‘t Panic


Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
nein
Anzahl freier Seiten unter free_pages_high?
ja
nein
Don‘t Panic


Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
nein
Anzahl freier Seiten unter free_pages_high?
ja
nein
3 Seiten freimachen
Don‘t Panic


Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
ja
nein
Anzahl freier Seiten unter free_pages_high?
ja
nein
3 Seiten freimachen
Don‘t Panic


Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
ja
nein
6 Seiten freimachen
Nur halbe Zeitspanne schlafen
Anzahl freier Seiten unter free_pages_high?
ja
nein
3 Seiten freimachen
Don‘t Panic


Systeminitialisierung
kswapd wird geladen
Kernel Swap Timer wird initialisiert
free_pages_low und free_pages_high werden initialisiert
Anzahl freier Seiten unter free_pages_low?
ja
nein
6 Seiten freimachen
Nur halbe Zeitspanne schlafen
Anzahl freier Seiten unter free_pages_high?
ja
nein
3 Seiten freimachen
Don‘t Panic
Zuletzt benutzte Methode wiederbenutzen:
1.) Page- und Buffer-Cache Seiten reduzieren
(z.B. Memory Mapped Files, Festplattendaten)
2.) System V Shared Memory Seiten auslagern (Swap File)
3.) Speicherseiten auslagern (Swap File)

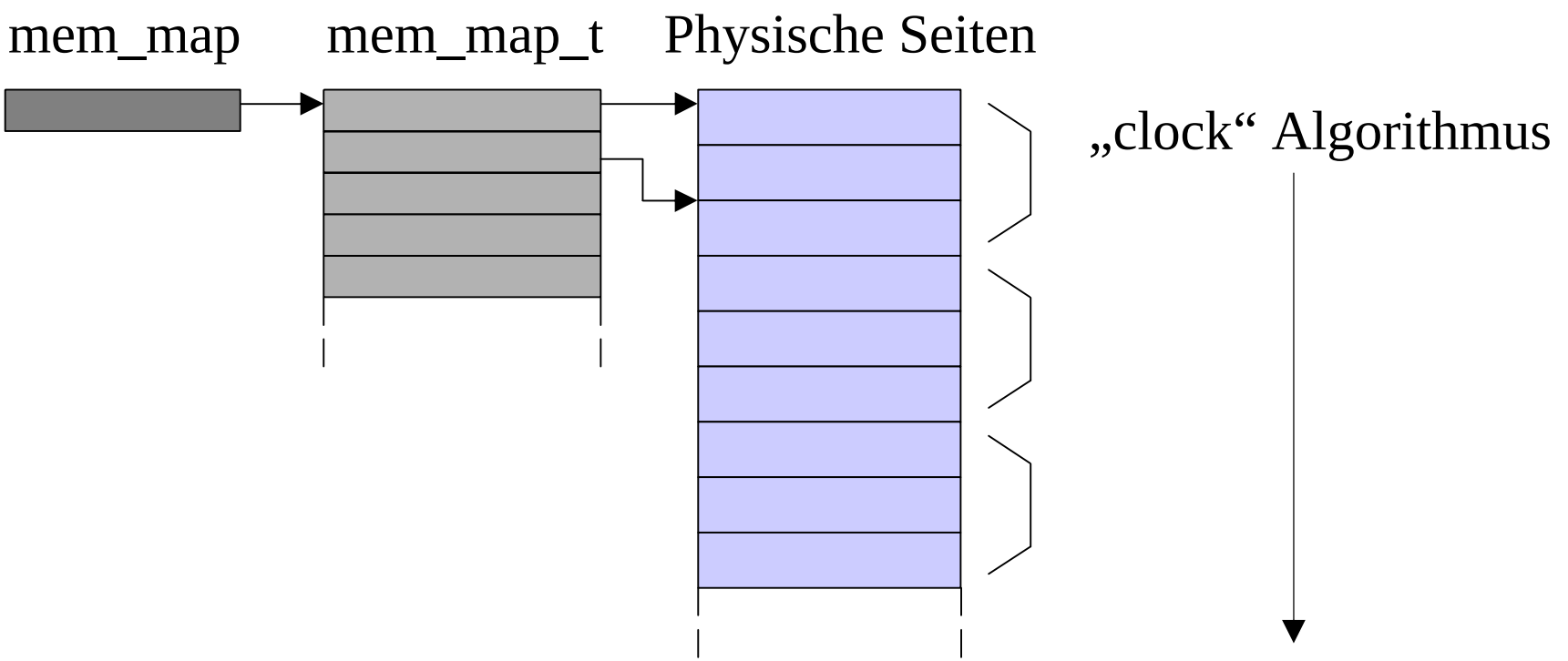
Inhalt

- Paging
  - Das Virtuelle Speichermodell
  - Die Page Table im Detail
  - Page Allocation und Page Deallocation
  - Memory Mapping & Demand Paging
- Caching
  - Die verschiedenen Caches
- Swapping
  - Welche Pages eines Prozesses werden ausgelagert
  - Der Kernel Swap Demon (kswapd)
  - Freimachen von Speicherseiten

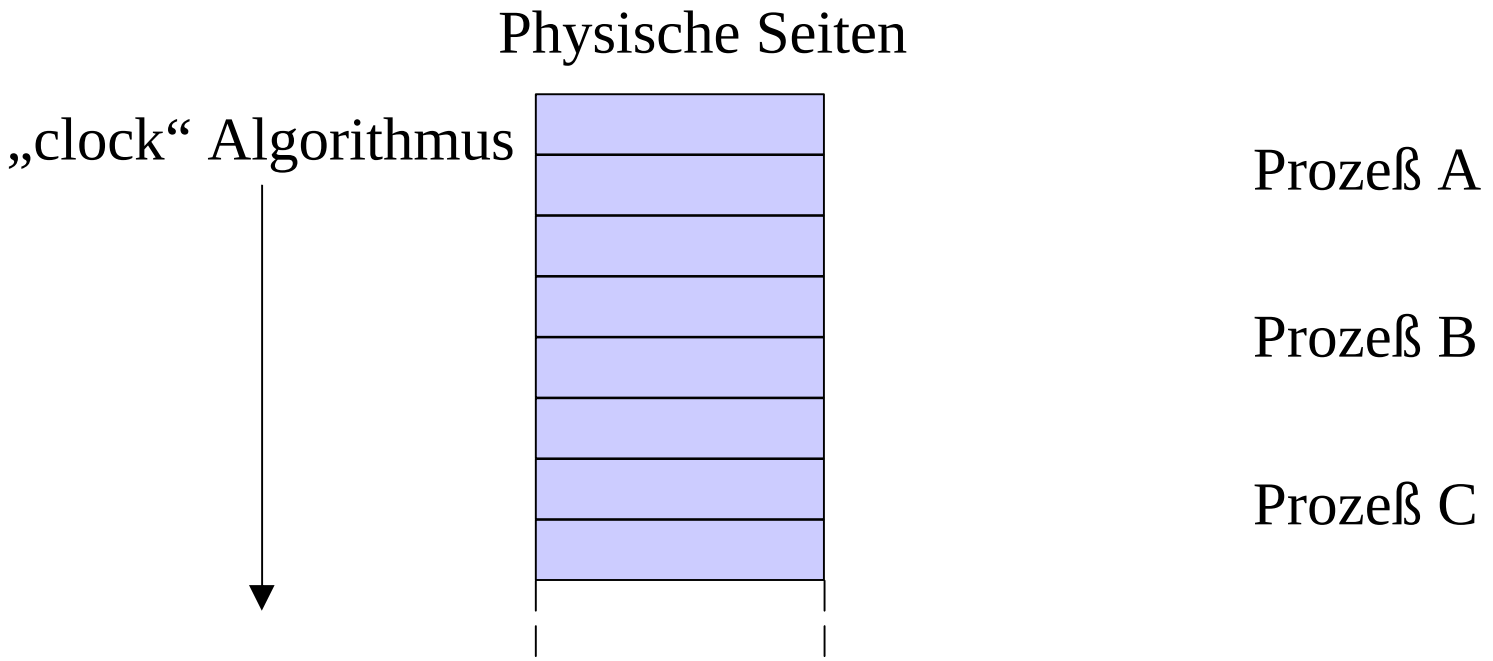
1.) Page- und Buffer-Cache Seiten reduzieren

(z.B. Memory Mapped Files, Festplattendaten)

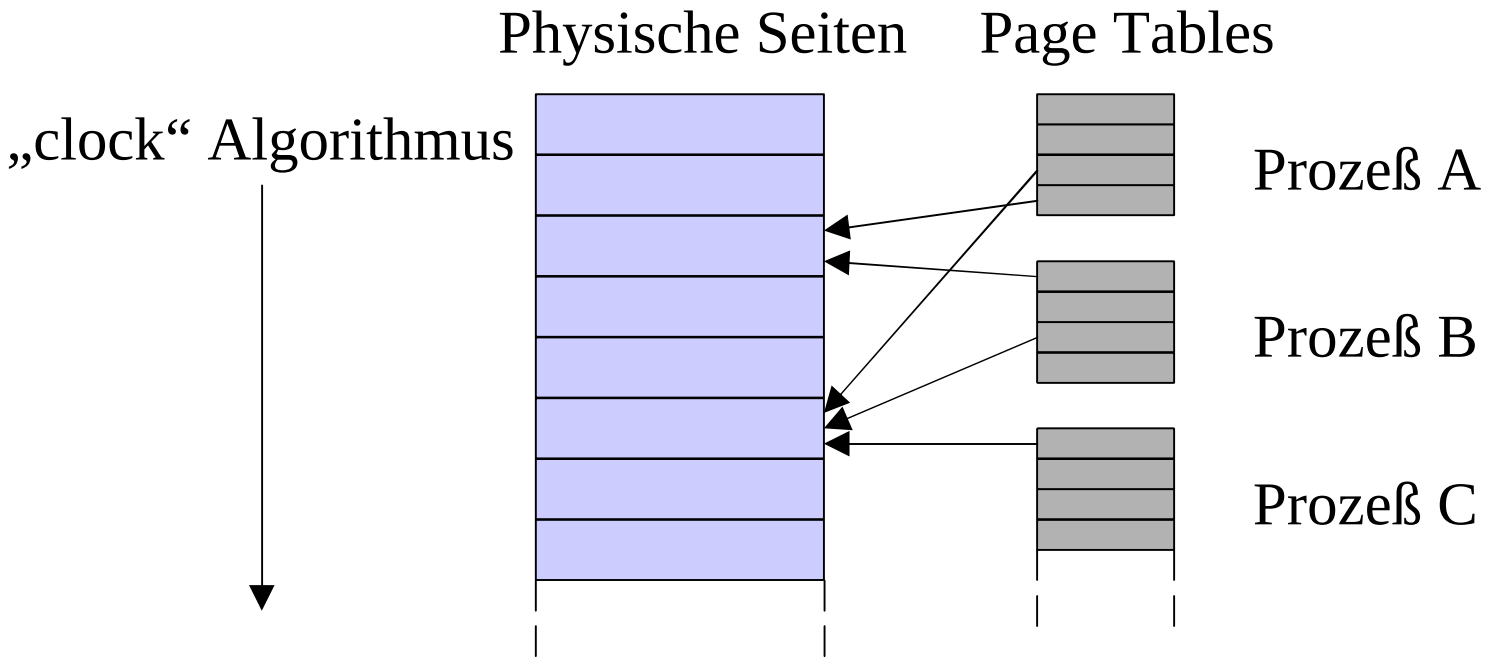
Kein Swapping (Datenträgerzugriff) nötig!



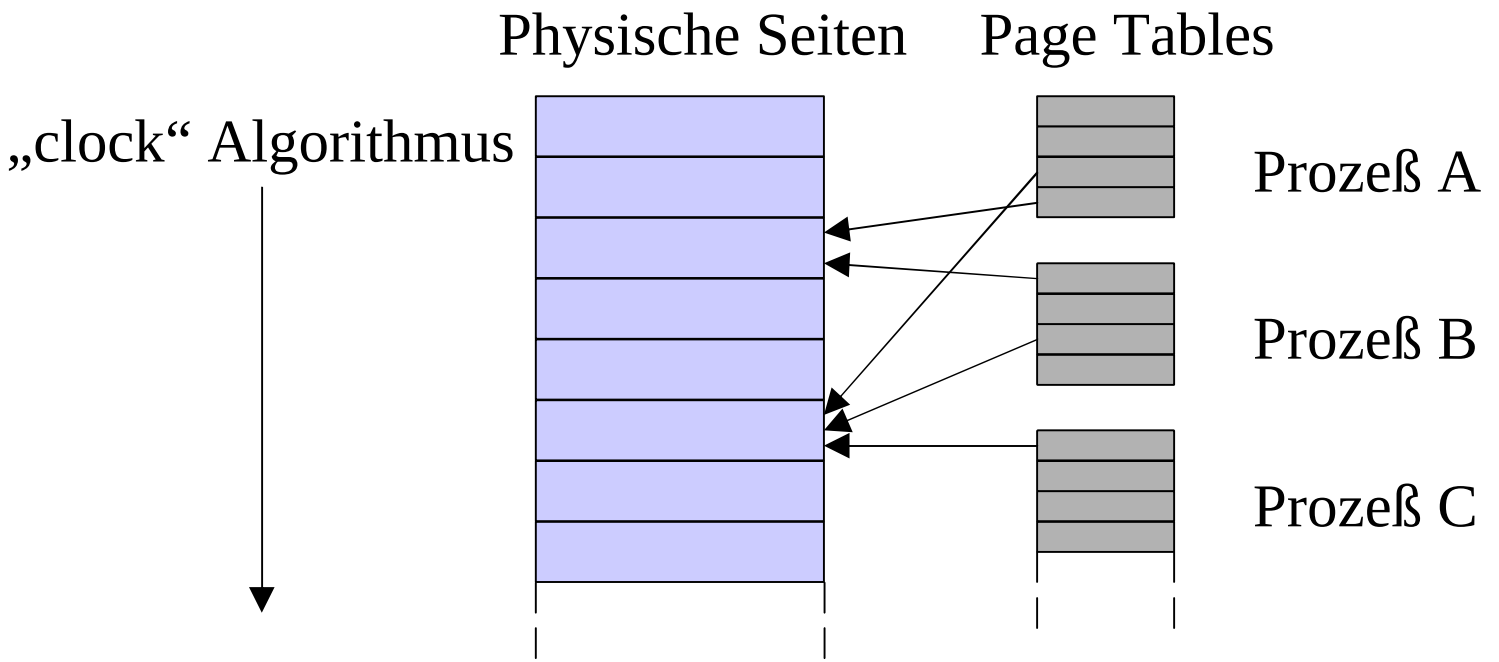
2.) System V Shared Memory Seiten auslagern (Swap File)



2.) System V Shared Memory Seiten auslagern (Swap File)



2.) System V Shared Memory Seiten auslagern (Swap File)



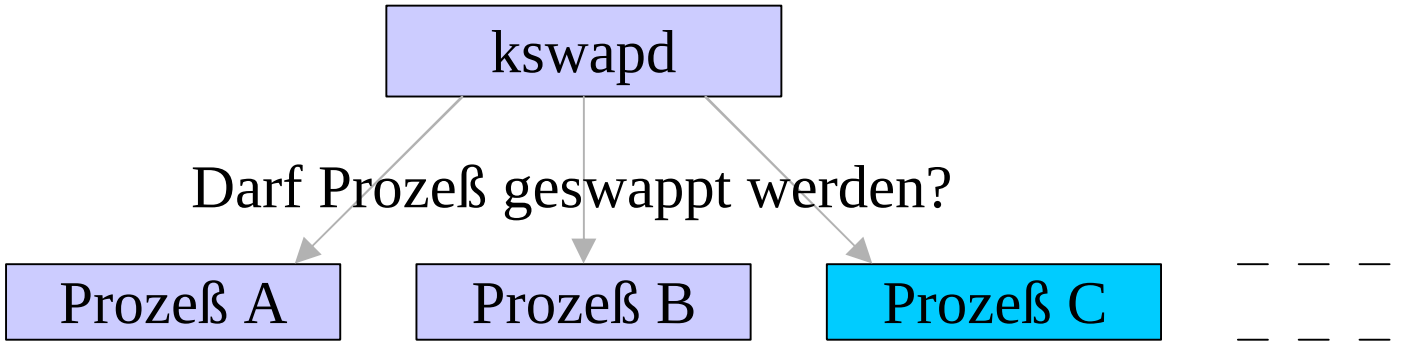
Page Table Eintrag „invalid“ setzen

PFN auf Swapposition setzen (Datei, Offset)

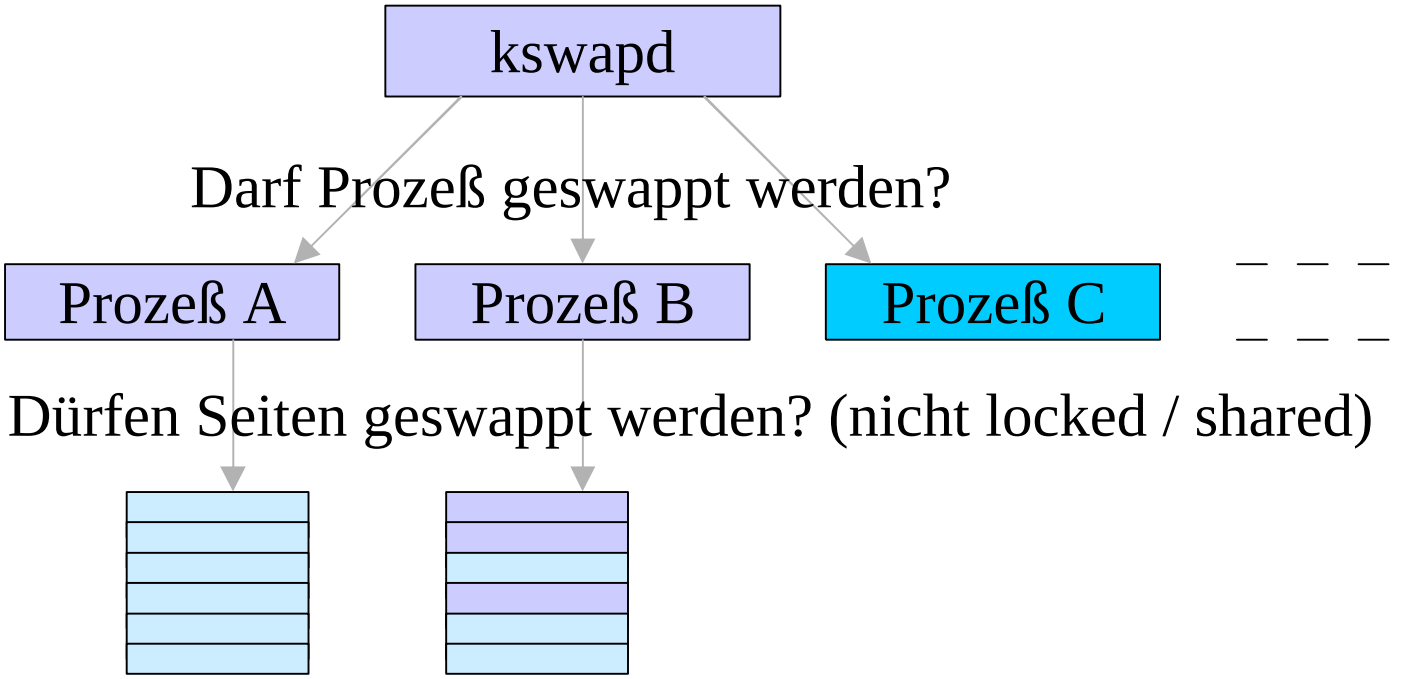
User Count für die Seite um 1 reduzieren (mem_map_t)

Wenn User Count = 0, dann Seite „swappen“

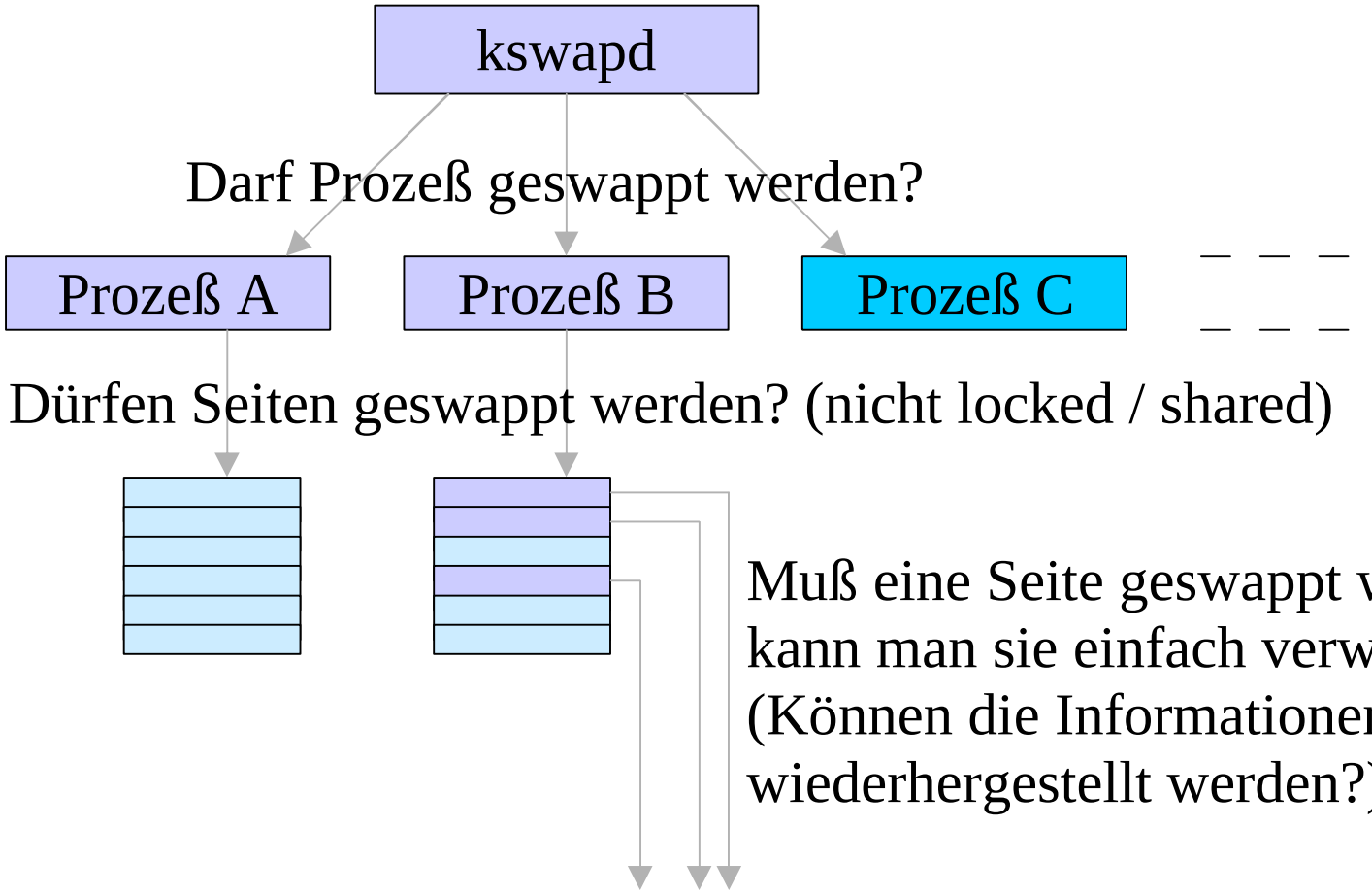
3.) Speicherseiten auslagern (Swap File)



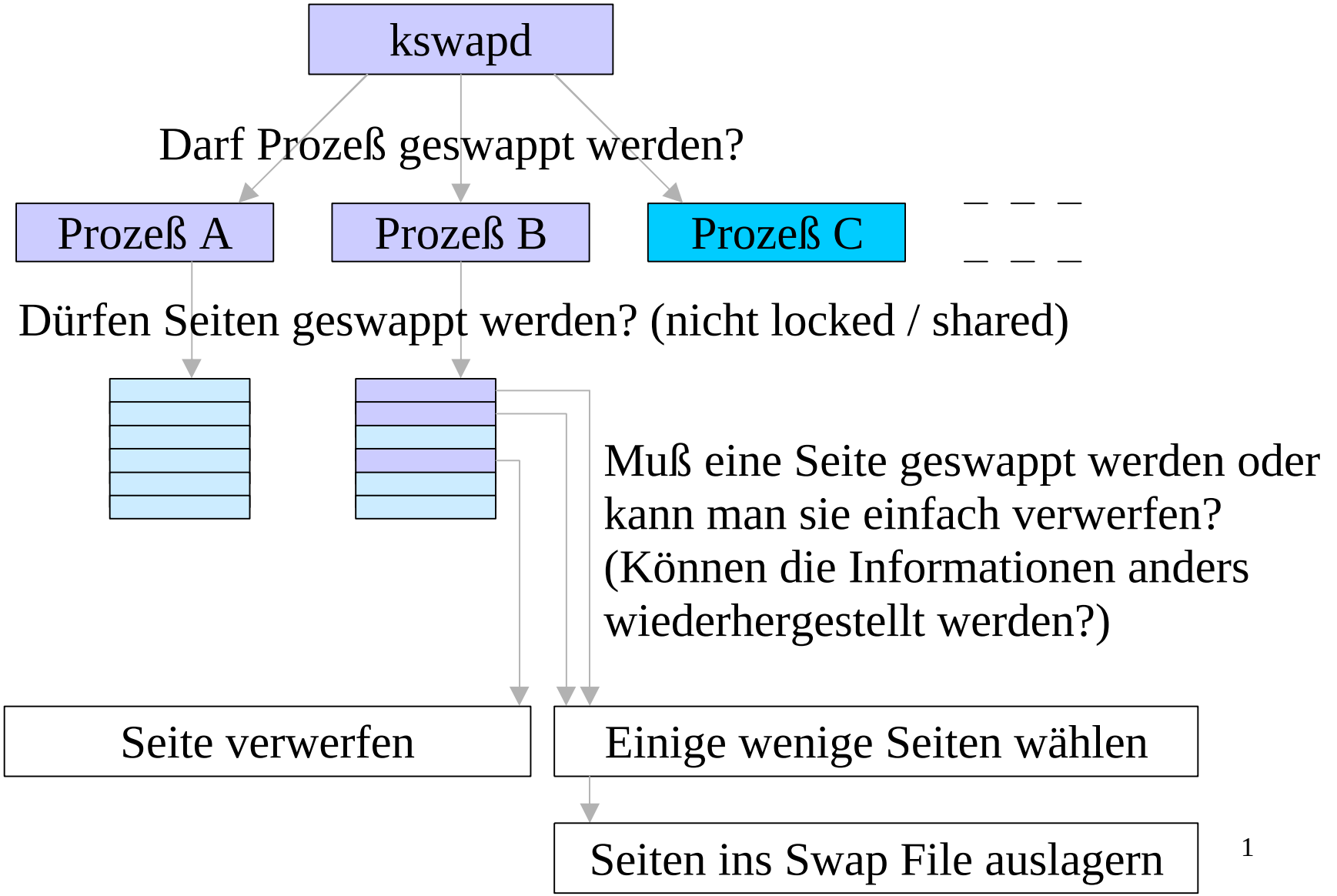
3.) Speicherseiten auslagern (Swap File)



3.) Speicherseiten auslagern (Swap File)



3.) Speicherseiten auslagern (Swap File)



Ende

„Linux Paging, Caching und Swapping“