FH Bielefeld
University of
Applied Sciences

Vorlesung

Betriebssysteme

Teil 8

Speicherverwaltung (2)

FH Bielefeld
University of
Applied Sciences

Inhalt der heutigen Vorlesung

- Virtueller Speicher
- Seitenersetzungsstrategie

- Übungsaufgabe

FH Bielefeld
University of
Applied Sciences

Virtueller Speicher: Überblick

- Idee des **virtuellen Speichers** (*virtual memory*, Fotheringham, 1961):
  - Ist ein **Programm größer** als der zur Verfügung stehende Speicher, dann wird nur der gerade benötigte Teil im Speicher gehalten.
- Wichtige Fragen:
  - Welche Teile werden gerade benötigt?
  - Welche Teile können ausgelagert werden?

    -> **Auslagerungs- und Einlagerungsstrategien**.
- **Zweistufiges Adressierungsschema**:
  - Die von den Programmen benutzten virtuellen Adressen werden von der *Memory Management Unit* (MMU) in physikalische Adressen umgewandelt und
  - dann erst an den Speicher gegeben.
- Wichtigstes Verfahren: **Paging**

FH Bielefeld
University of
Applied Sciences

Virtueller Speicher: Paging

Prinzip des Paging:

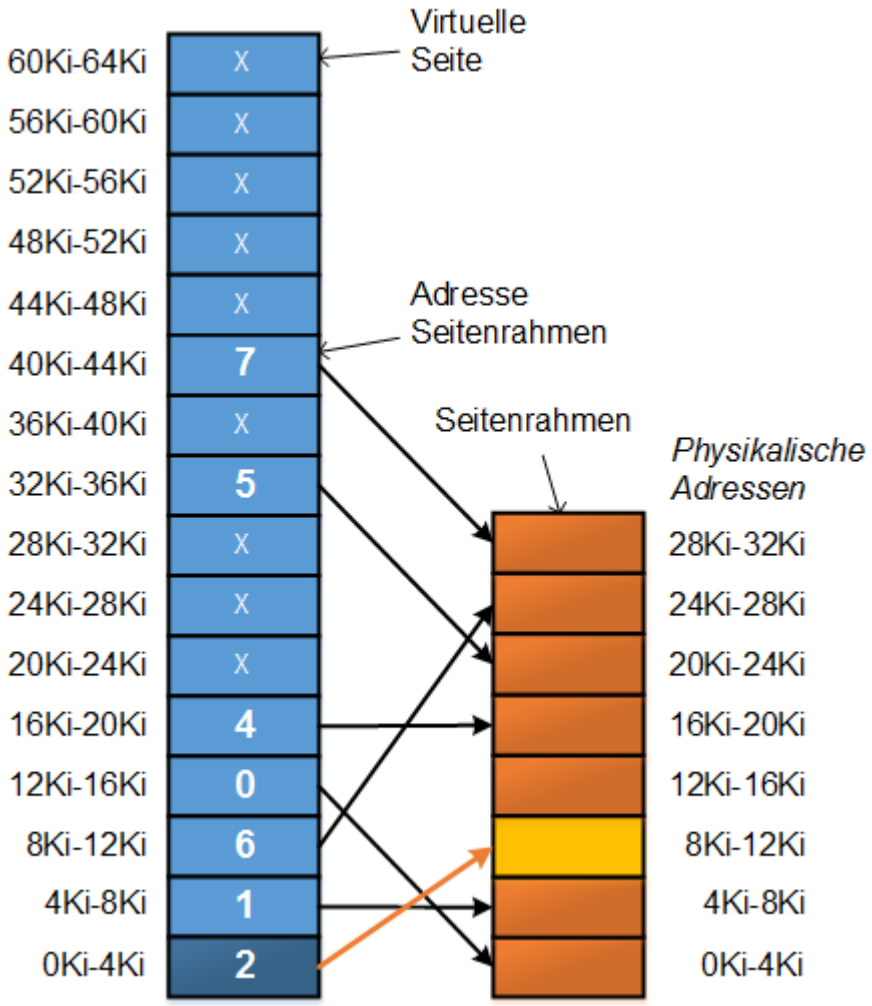
- Der virtuelle Adressraum ist in **Seiten** (*pages*) aufgeteilt.
- Der physikalische Speicher ist in **Seitenrahmen** / **Seitenkacheln** (*page frames*) aufgeteilt.
- Seiten und Seitenrahmen sind immer gleich groß!
- Die virtuelle Adresse wird in
  - eine **Seitennummer** (*page number*) und
  - eine **Adresse** innerhalb der Seite (*page offset*) aufgeteilt.
- Die Seitennummer adressiert einen Seitenrahmen über eine **Seitentabelle** (*page table*).
- Seiten, die nicht im Speicher gehalten werden können, werden auf Platte (Hintergrundspeicher) **auslagert**.

FH Bielefeld
University of
Applied Sciences

Virtueller Speicher: Beispiel Paging

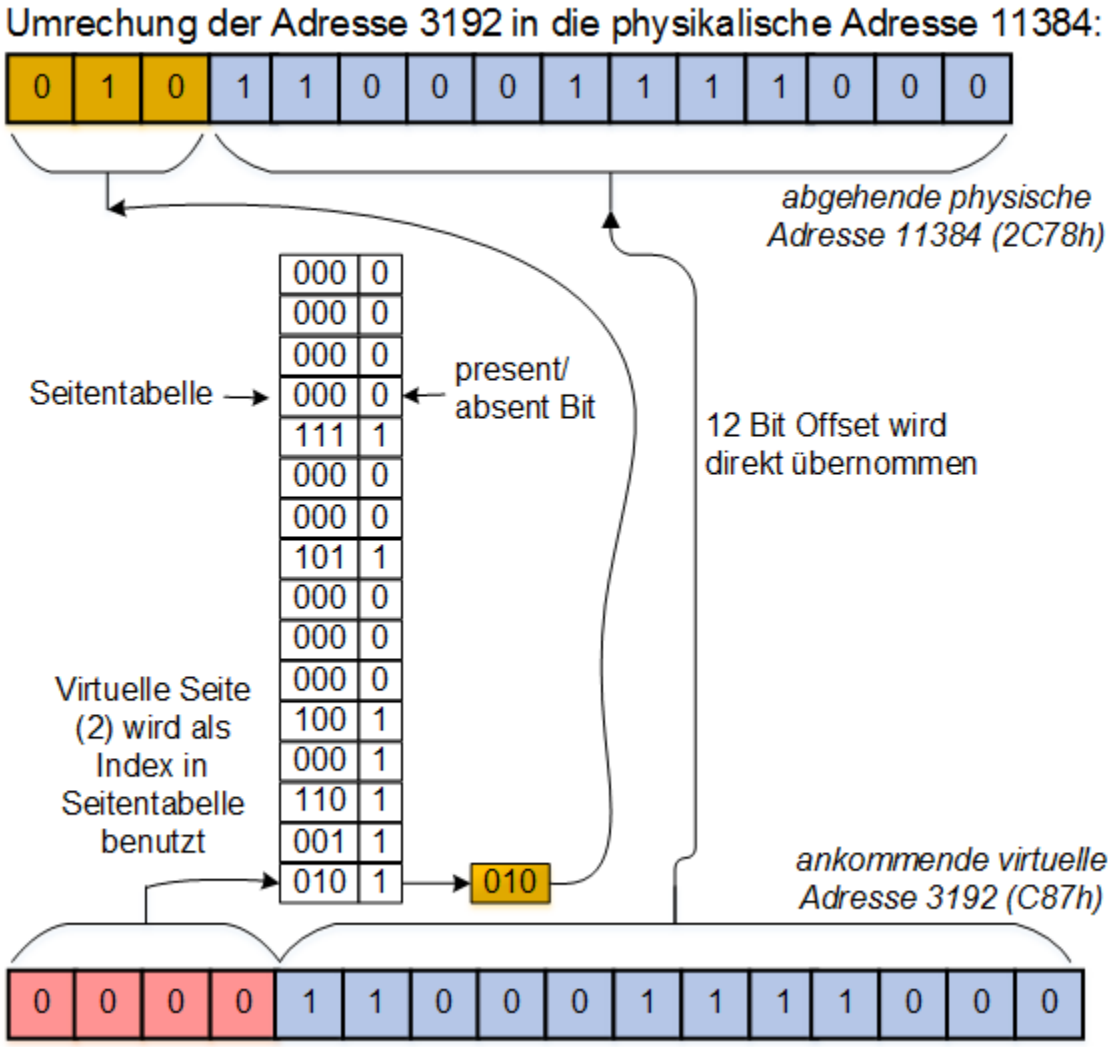
Beispiel für Paging:

- Ein System habe
  - 64 KiByte virtuellen Speicher (16 Bit Adressen)
  - aber nur 32 KiByte RAM.
  - Die Seitengröße betrage 4 KiByte.
- Der virtuelle Adressraum wird auf die physikalischen Adressen abgebildet.
- Aus dem Befehl: **MOV REG, 3192** im virtuellen Adressraum wird hier: **MOV REG, 11384** auf dem physikalischen Bus.



FH Bielefeld
University of
Applied Sciences

Virtueller Speicher: Umrechnung der Adressen in MMU



FH Bielefeld
University of
Applied Sciences

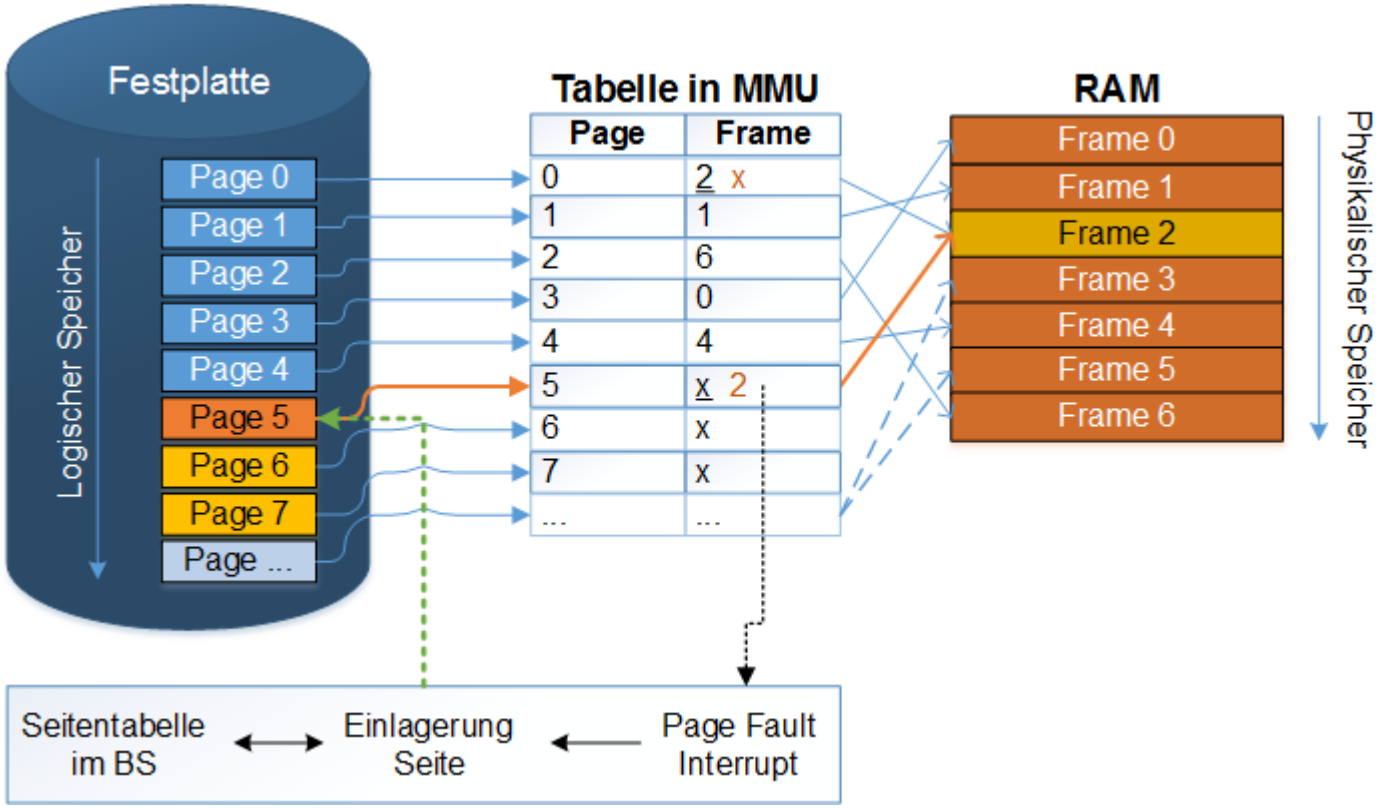
Virtueller Speicher: Seitenfehler

Was passiert bei einem Zugriff auf eine **nicht geladene Seite**?

- Beispiel: Befehl **MOV REG, 22870**
- Die MMU stellt fest, dass die virtuelle Seite (5 = 0101) nicht geladen ist und löst eine **Unterbrechung** aus →**Seitenfehler** (*page fault*). Der aufrufende **Prozess** wird **blockiert**.
- Das Betriebssystem sucht einen **freien Seitenrahmen** aus.
  - Ist kein Seitenrahmen frei, wird ein benutzter Seitenrahmen gewählt.
  - Wurde dieser modifiziert, wird er auf die Platte zurückgespeichert.
- **Seite** wird von der Platte **geladen** und in den **Seitenrahmen** geschrieben.
- **Seitentabelle** wird **aktualisiert**.
- Der **Befehlszähler** des aufrufenden Prozesses wird **zurückgesetzt** (der letzte Befehl muss wiederholt werden) und der Prozess wird wieder in den ‚bereit' Zustand versetzt.

FH Bielefeld
University of
Applied Sciences

Virtueller Speicher: Arbeitsweise Paging



- Einlagern / Auslagern erfolgt unabhängig von Prozesszugehörigkeit
- Einlagern, falls Seite (hier: Seite 5) von einem Prozess referenziert wird
- Auslagern (Seite 0 in Rahmen 2), falls Rahmen für eine andere Seite benötigt wird

FH Bielefeld
University of
Applied Sciences

Seitenersetzung: Strategien

- Die **Seitenersetzungsstrategie** bestimmt, welcher belegte Seitenrahmen bei einem Seitenfehler aus dem Speicher entfernt wird, damit eine neue Seite eingelagert werden kann.
- Ziel aller Strategien: **möglichst wenig Transfers** von Seiten zwischen Platte und Speicher (Effizienz).
- Ähnliche Probleme existieren in anderen Bereichen der Informatik, z.B.
  - Cache-Speicher für Datenzugriffe in Datenbanken oder auf Prozessorebene,
  - Zwischenspeichern von WWW-Seiten auf einem Web-Server.
  - Lösungen sind also auf andere Gebiete übertragbar.

FH Bielefeld
University of
Applied Sciences

Seitenersetzung: Strategien

Referenz-string	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
optimal Rahmen 0	7	7	7	2	2	2	2	2	2	2	2	2	2	2	2	2	2	7	7	7
optimal Rahmen 1		0	0	0	0	0	0	4	4	4	0	0	0	0	0	0	0	0	0	0
optimal Rahmen 2			1	1	1	3	3	3	3	3	3	3	3	1	1	1	1	1	1	1
FIFO Rahmen 0	7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
FIFO Rahmen 1		0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
FIFO Rahmen 2			1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1
LRU Rahmen 0	7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
LRU Rahmen 1		0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
LRU Rahmen 2			1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7

optimal: 9 Ersetzungen
FIFO: 15 Ersetzungen
LRU: 12 Ersetzungen

FH Bielefeld
University of
Applied Sciences

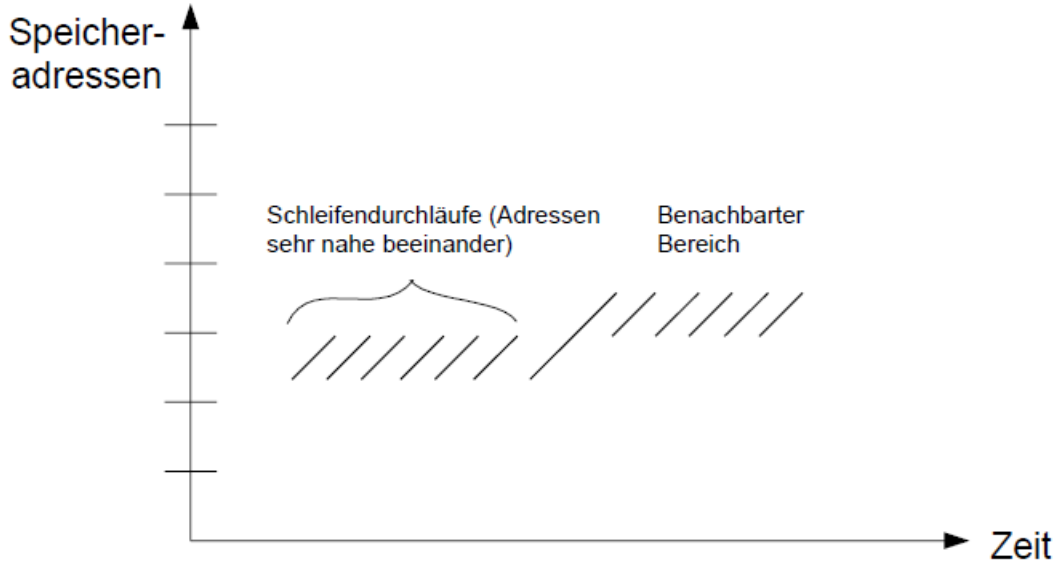
Lokalitätsprinzip

- Prozesse weisen zeitliche und räumliche Lokalität auf:
  - **zeitlich**: kürzlich angesprochene Adresse wird in naher Zukunft wieder angesprochen.
  - Gründe: Schleifen, Unterprogramme, Stacks, Zählvariable

FH Bielefeld
University of
Applied Sciences

Lokalitätsprinzip

- **räumlich**: Adressen in der Nachbarschaft kürzlich angesprochener Adressen werden mit größerer Wahrscheinlichkeit angesprochen als weiter entfernte.
- Gründe: Durchlaufen von Feldern, sequentieller Code-Zugriff



FH Bielefeld
University of
Applied Sciences

Seitenersetzung: LRU Strategie

Least-Recently-Used (LRU) Algorithmus:

- LRU ist eine gute Annäherung an die optimale Strategie.
- Annahme:
  - Die Seite, die in der **Vergangenheit** häufig benutzt wurde, wird **auch in Zukunft** häufig benutzt.
  - **Lokalität** der Ausführung.
- Algorithmus: Bei einem Seitenfehler wird die **am längsten unbenutzte Seite ausgelagert**.
- Implementierung ist **schwierig** und ineffizient, da bei jedem Speicherzugriff die *Seitenliste neu sortiert* werden muss.

FH Bielefeld
University of
Applied Sciences

Seitenersetzung: NRU Strategie

- Der *Not-Recently-Used* (NRU) Algorithmus wertet aus, ob eine Seite gelesen oder verändert wurde. Dafür wird das *Referenced* Bit (R-Bit) und das *Modified* Bit (M-Bit) ausgewertet (Bits sind als Zusatzinfo in Seitentabelle enthalten).
- Die **Seiten** werden in folgende **Klassen** eingeteilt:
  - Klasse 0: nicht referenziert, nicht modifiziert
  - Klasse 1: nicht referenziert, modifiziert
  - Klasse 2: referenziert, nicht modifiziert
  - Klasse 3: referenziert, modifiziert
- Aus der niedrigsten nicht-leeren Klasse wird eine *zufällige* Seite verdrängt.
- Hinweis: Klasse 1 entsteht durch **periodisches Löschen der R-Bits** durch Timer-Unterbrechungen (daher: *not recently used*...)

Seitenersetzung: Second Chance Strategie

- Variante des FIFO-Algorithmus, die verhindert, dass häufig benutzte Seiten ausgelagert wird.
- Algorithmus: **R-Bit** der ältesten Seite wird via **Timer** geprüft:
  - R-Bit nicht gesetzt, dann wird die Seite ausgelagert.
  - R-Bit gesetzt, dann wird das Bit gelöscht und die Seite an den Anfang der FIFO-Liste verlagert. Die nächste Seite in der FIFO-Liste wird geprüft.
- Ist bei allen Seiten das R-Bit gesetzt, degeneriert der Algorithmus zum FIFO-Algorithmus, da die älteste Seite mit gelöschtem R-Bit durchgeschoben wird.
- Eine Variante von *Second Chance* mit einem **Ring-Puffer** statt einer FIFO-Liste wird ***Clock-Algorithmus*** genannt.
  - Ring-Puffer Verwaltung ist effizienter als Umhängen in Listen

Übungsaufgabe

Ein Speicher mit 6 Seitenrahmen sei entsprechend folgender Tabelle belegt.

Rahmen	Seite	Ladezeit	letzter Zugriff	Reference-Bit	Dirty-Bit
0	8	30	40	1	0
1	3	10	10	1	1
2	7	20	25	0	0
3	2	60	60	0	1
4	6	0	15	1	0
5	0	50	55	1	0

Geben Sie den Inhalt der 6 Seitenrahmen inklusive Ladezeit, letztem Zugriff, Reference- und Dirty-Bit nach der Folge von Schreibzugriffen auf die Seiten 2, 4, 6, 7, 0 für

a) LRU

b) FIFO Second Chance (Wenn eine Seite eine 2. Chance bekommt, wird ihre Ladezeit auf den aktuellen Wert gesetzt.)

an. (Schreiben Sie Zwischenschritte auf, wenn Sie sich Ihres Ergebnisses nicht sicher sind.)

Hinweis: Verwenden Sie am besten Zehnerschritte (80,90,100,..) für die Ladezeiten der neuen Seiten und unterteilen Sie diese dann in Einerschritte im Falle von FIFO Second Chance.

FH Bielefeld
University of
Applied Sciences

Lösung

a) Least recently Used:

	Seite	Ladezeit	l.Z.	r-Bit	d-Bit
0	8	30	40	1	0
1	4	90	90	1	1
2	7	20	110	1	1
3	2	60	80	1	1
4	6	0	100	1	1
5	0	50	120	1	1

b) Ladezeit und r-Bit sind ausschlaggebend.

	Seite	Ladezeit	l.Z.	r-Bit	d-Bit
0	8	110	40	0	0
1	7	114	114	1	1
2	4	92	92	1	1
3	2	112	80	0	1
4	6	113	100	0	1
5	0	111	120	1	1

FH Bielefeld
University of
Applied Sciences

Übungsaufgabe

FH Bielefeld
University of
Applied Sciences

Vorlesung

Vielen Dank für Ihre Aufmerksamkeit

Dozent

Prof. Dr.-Ing.
Martin Hoffmann
martin.hoffmann@fh-bielefeld.de