

## Virtuelle Speicherverwaltung

Software-Prinzipien und Hardware-Unterstützung der virtuellen Speicherverwaltung

### Grundprinzip und Zusammenhang mit dem Betriebssystem

#### Definition des Begriffs "Betriebssystem":

z. B. nach dem Deutschen Institut für Normung (DIN 44300):

Ein Betriebssystem umfasst die Programme eines digitalen Rechensystems, die zusammen mit den Eigenschaften der Rechanlage die Grundlage der möglichen Betriebsarten des digitalen Rechensystems bilden und insbesondere die Abwicklung von Programmen steuern und überwachen.

➔ Ziel eines Betriebssystems ist es, die semantische Lücke zwischen Anwendung (wünscht möglichst hohe Operationen) und Hardware (bietet elementare Operationen) zu verkleinern.

### Spezielle Aufgaben von Betriebssystemen

- Betriebsmittelverwaltung
- Auftragsverwaltung (Prozeßverwaltung)
- Speicherverwaltung

#### □ Betriebsmittelverwaltung (*resource management*)

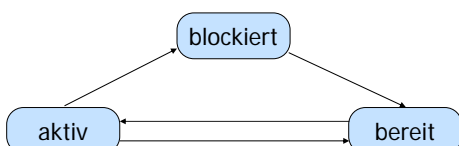
Verwaltet alle peripheren Betriebsmittel des Rechensystems, wie z. B. Festplatte, Floppy Disk, CDROM, Drucker, ...

**Ziel:** einfache, effektive und konfliktfreie Nutzung der Betriebsmittel durch die verschiedenen Prozesse.

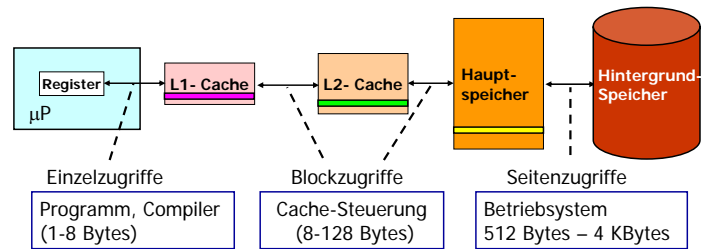
### Spezielle Aufgaben von Betriebssystemen

Verwaltung dieser Zustände und Übergänge (aufgrund äußerer oder innerer Ereignisse) sind wesentliche Aufgaben der Auftragsverwaltung.

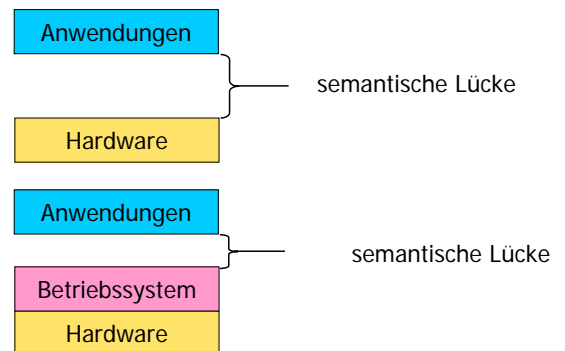
**Ziel:** möglichst effiziente und scheinbar parallele Bearbeitung mehrerer Aufgaben



Daten werden nur zwischen aufeinanderfolgenden Ebenen der Speicherhierarchie kopiert.



### Betriebssystem verkleinert die semantische Lücke



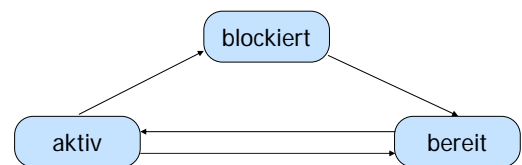
Betriebssystem erweitert die Fähigkeiten der Hardware.

### Spezielle Aufgaben von Betriebssystemen

#### □ Auftragsverwaltung (*task management*)

Verwaltung der Prozesse (*tasks*)

Ein Prozeß kann im Laufe seiner Bearbeitung im wesentlichen folgende unterscheidbaren Zustände annehmen:



### Spezielle Aufgaben von Betriebssystemen

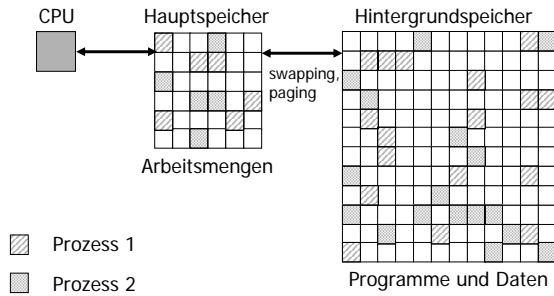
#### □ Speicherverwaltung (*memory management*)

Durch immer größer werdende Programme sowie mehrerer quasi gleichzeitig laufender Programme

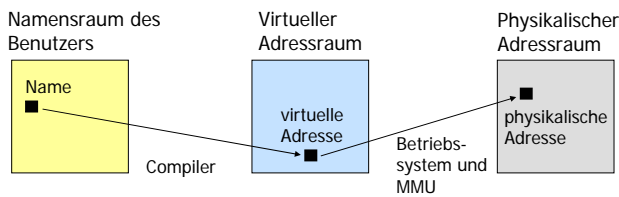
➔ der zur Verfügung stehende Arbeitsspeicher wird schnell zu klein

**Abhilfe:** Nur die gerade benötigten Teile der aktiven Programme werden wirklich im Arbeitsspeicher gehalten, der Rest befindet sich im Hintergrundspeicher und wird bei Bedarf geladen (*swapping, paging*).

# Grundstruktur virtueller Speicherverwaltung



## Virtuelle Speicherverwaltung



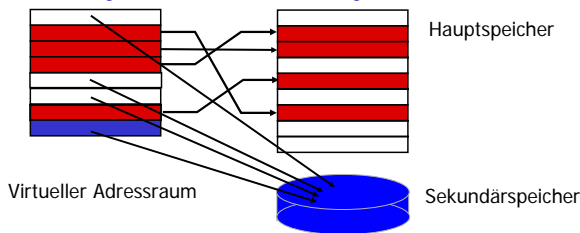
**Benutzer:** kennzeichnet seine Objekte (Programme, Unterprogramme, Variablen, ...) durch Namen

**Compiler:** übersetzt diese Namen in virtuelle (logische) Adressen.

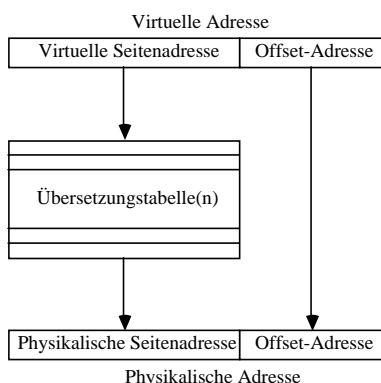
**Virtuelle Speicherverwaltung:** wandelt diese Adressen zur Laufzeit je nach gerade gegebener Speicherbelegung in die physikalische Adresse (wirklicher Ort des Objekts im Hauptspeicher) um

## Speicherverwaltung

- Virtuelle Speicherkapazität > effektive Hauptspeicherkapazität
- Betriebssystem lagert bei Bedarf Speicherbereiche ein und aus
- Speicherverwaltungseinheiten (*memory management units, MMU*) unterstützt hardwaremäßig eindeutige Adressberechnung
- Abbildungsinformation in Übersetzungstabellen



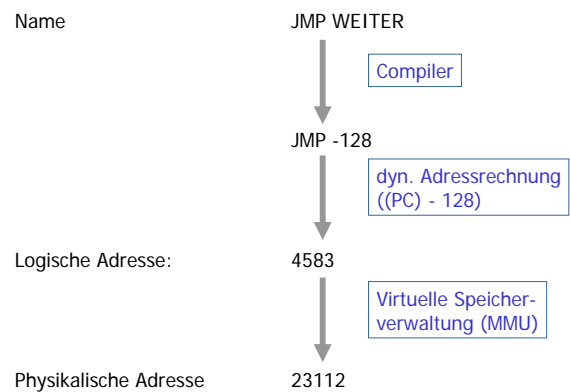
## Abbildung virtueller auf physikalische Adressen



# Virtuelle Speicherverwaltung

- Dieser Vorgang bleibt dem Anwender völlig verborgen, d. h. der Arbeitsspeicher erscheint dem Anwender wesentlich größer, als er in Wahrheit ist.
- Ein nach diesem Konzept verwalteter Speicher heißt deshalb virtueller Speicher.
- Von modernen Prozessoren wird die Verwaltung dieses virtuellen Speichers hardwaremäßig durch eine MMU (memory management unit) unterstützt.
- Hauptaufgabe dieser virtuellen Speicherverwaltung: **Umsetzung virtueller (logischer) Adressen in physikalische Adressen**

## Beispiel



## Speicherverwaltung

- Das **Betriebssystem** führt Buch über die freien Speicherbereiche und lagert bei nicht ausreichendem Freiplatz im Hauptspeicher diejenigen Speicherinhalte auf den Hintergrundspeicher aus, die gegenüber ihrem Originalen auf dem Hintergrundspeicher verändert worden sind.
- Die eindeutige Abbildung des großen virtuellen Speichers auf die effektive Hauptspeicherkapazität wird von der Hardware durch **Speicherverwaltungseinheiten** unterstützt (*memory management units, MMU*)
- Die erforderliche Abbildungsinformation stellt das Betriebssystem in Form einer oder mehrerer **Übersetzungstabellen** zur Verfügung.

## Speicherverwaltung

- Um den Umfang der Übersetzungstabellen gering zu halten, bezieht man die Abbildungsinformation nicht auf einzelne Adressen, sondern auf zusammenhängende Adressbereiche
- Es gibt zwei Möglichkeiten:
  - **Segmentierung**
  - **Seitenwechsel-Verfahren**

Es existieren zwei grundlegende Verfahren zur virtuellen Speicherverwaltung (Segmentierung und Seitenwechsel)

**Segmentierung**

- > Hierbei wird der virtuelle Adressraum in Segmente verschiedener Länge zerlegt.
- > Jedem Prozess sind ein oder mehrere Segmente z. B. für den Programmcode und die Daten, zugeordnet.
- > Die einzelnen Segmente enthalten logisch zusammenhängende Informationen (Programm- und Datenmodule) und können relativ groß sein.

**Segmentierung**

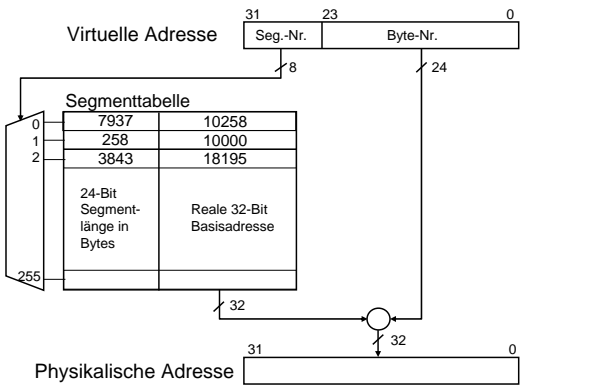
**Vorteile:**

- > Segmentierung spiegelt logische Programmstruktur wieder.
- > durch große Segmente relativ seltener Datentransfer.

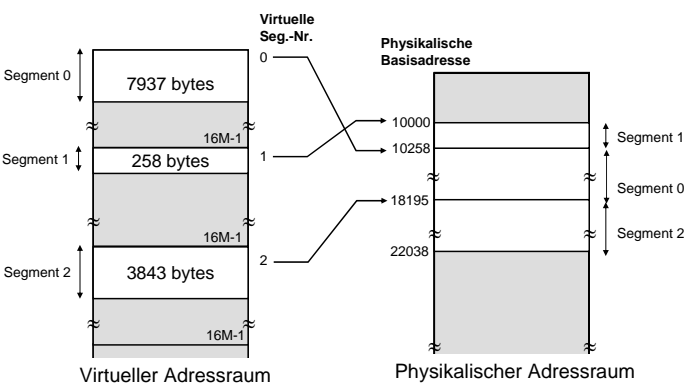
**Nachteile:**

- > wenn Datentransfer, dann jedoch umfangreich.
- > besteht ein Programm nur aus einem Code- und Daten-Segment (wird vom Compiler oder Benutzer festgelegt), so muss es vollständig eingelagert werden.

**Segmentbasierte Speicherverwaltung**



**Virtueller und physikalischer Adressraum**



**Aufteilung in Seiten**

- > Hierbei wird der logische und der physikalische Adressraum in "Segmente fester Länge", die sogenannten Seiten (pages) unterteilt.
- > Die Seiten sind relativ klein (256 Byte - 4 kByte)
- > Ein Prozess wird auf viele dieser Seiten verteilt (keine logischen Zusammenhänge wie bei der Segmentierung)

**Seitenaufteilung**

**Vorteile:**

- > durch kleine Seiten wird nur der wirklich benötigte Teil eines Programms eingelagert.
- > geringerer Verwaltungsaufwand als Segmentierung

**Nachteil:**

- > häufiger Datentransfer

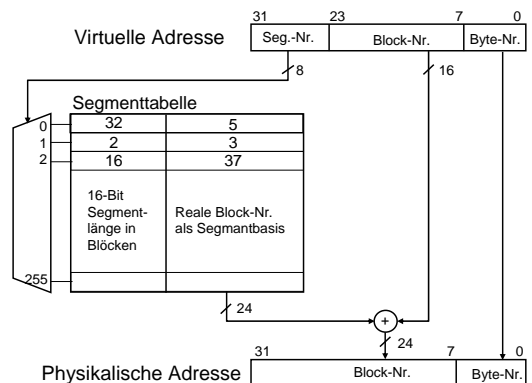
Ältere Prozessoren unterstützen jeweils nur ein Verfahren, z. B. Segmentierung bei 80286, 68010 und Seitenwechsel bei Z8003/4, National 32000

Heutige Mikroprozessoren, z.B. 80386, 80486, Pentium, ... unterstützen beide Verfahren

**Segmentbasierte Speicherverwaltung**

- Virtuelle Adresse wird in eine Segmentnummer (n höherwertige Bits der virtuellen Adresse) als Kennung eines Segments und in eine Bytenummer (verbleibenden m Bits der Adresse) als Abstand zum Segmentanfang unterteilt.
- Maximale virtuelle Segmentanzahl =  $2^n$ , Maximale Segmentgröße =  $2^m$
- Die Adressabbildung erfolgt über eine Segmenttabelle (im Registerspeicher der MMU).
- Reale Adresse ergibt sich aus der Segmentbasisadresse, zu der die virtuelle Byte-Nummer addiert wird.
- Segmentlängenangaben in der Segmenttabelle, um segmentüberschreitende Zugriffe feststellen und ggf. verhindern zu können.
- Bei Segmenten mit einer geringeren Größe als  $2^m$ , gilt der ungenutzte Raum als Verschnitt.
- Gute Ausnutzung des Hauptspeichers, wenn man die Segmentgrenzen an jeder Byteadresse zulässt.

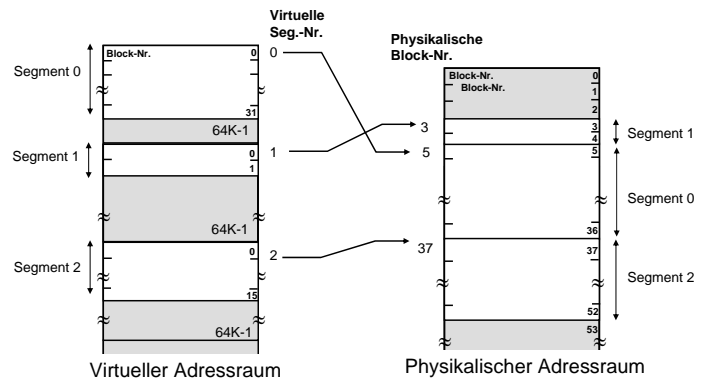
**Segmentbasierte Speicherverwaltung**



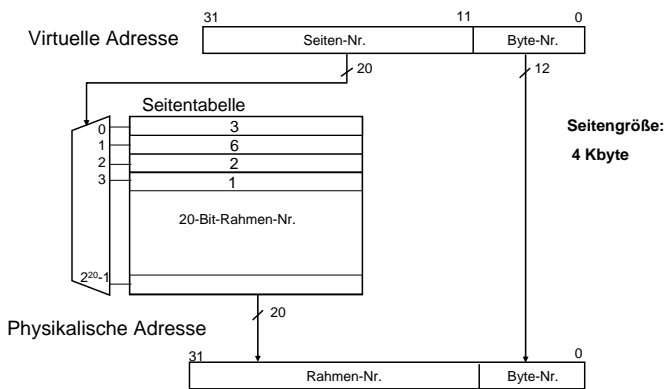
# Segmentbasierte Speicherverwaltung

- Segmentgrenzen nicht an jeder Byteadresse, sondern an Vielfachen von Blöcken (hier von 256 Bytes).
- Segmente werden im virtuellen physikalischen Adressraum in Blöcke von 256 Bytes unterteilt.
- D.h. die Bytenummer aus  $m$  Bits wird aufgeteilt in eine kürzere Bytenummer für die Byteadressierung im Block und eine Blocknummer.
- Bei Adressumsetzung wird die virtuelle Segmentnummer auf eine reale 24-Bit-Blocknummer als Segmentbasis abgebildet.
- Virtuelle Bytenummer für die Adressierung innerhalb des Blocks wird unverändert übernommen.

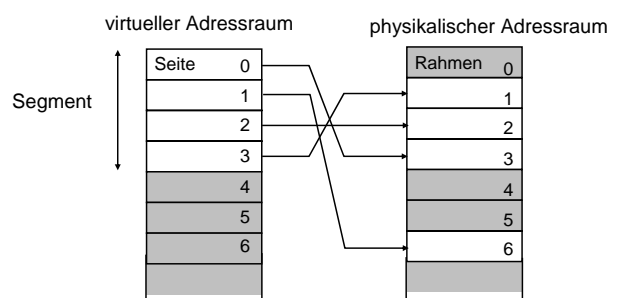
# Virtueller und physikalischer Adressraum



# Seitenwechsel (Paging)



# Virtueller und physikalischer Adressraum



# Probleme der virtuellen Speicherverwaltung

Beim Austausch von Daten zwischen Haupt- und Hintergrundspeicher ergeben sich drei Problemkreise:

- Einlagerungszeitpunkt:** Wann werden Segmente oder Seiten in den Hauptspeicher eingelagert?
- Zuweisungsproblem:** An welche Stelle des Hauptspeichers werden die Seiten oder Segmente eingelagert?
- Ersetzungsproblem:** Welche Segmente oder Seiten müssen ausgelagert werden, um Platz für neu benötigte Daten zu schaffen?

# Probleme der virtuellen Speicherverwaltung

Beim Austausch von Daten zwischen Haupt- und Hintergrundspeicher ergeben sich drei Problemkreise:

## 1. Der Einlagerungszeitpunkt

Wann werden Segmente oder Seiten in den Arbeitsspeicher eingelagert ?

### Gängiges Verfahren:

Einlagerung auf Anforderung (Demand Paging bei Seitenverfahren)

# Probleme der virtuellen Speicherverwaltung

Einlagerung auf Anforderung (*Demand Paging*):

Hierbei werden Daten eingelagert, sobald auf sie zugegriffen wird, sie sich aber nicht im Hauptspeicher befinden.

Der Zugriff auf ein nicht im Hauptspeicher vorhandenes Segment oder Seite heißt **Segment-** oder **Seiten-Fehler** (*segment fault, page fault*).

# Probleme der virtuellen Speicherverwaltung

## 2. Das Zuweisungsproblem

An welche Stelle des Hauptspeichers werden die Seiten oder Segmente eingelagert ?

### Bei Segmentierungsverfahren:

Hier muss eine ausreichend große Lücke im Hauptspeicher gefunden werden.

### Drei Strategien (Betriebsystem):

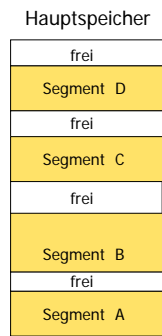
- first-fit*: erste passende Lücke wird genommen
- best-fit*: kleinste passende Lücke wird genommen
- worst-fit*: größte passende Lücke wird genommen

## Probleme der virtuellen Speicherverwaltung

### Problem bei allen drei Verfahren:

Der Speicher zerfällt nach einiger Zeit in belegte und unbelegte Speicherbereiche  
→ externe Fragmentierung.

Die unbelegten Speicherbereiche sind oft zu klein, um weitere Segmente aufnehmen zu können



## Probleme der virtuellen Speicherverwaltung

### Zuweisungsproblem bei Seitenwechselverfahren

Hier taucht dieses Problem nicht auf, da alle Seiten gleich groß sind und somit immer "passende Lücken" entstehen  
→ keine externe Fragmentierung.

Jedoch: Problem der **internen Fragmentierung**  
Diese entsteht bei der Aufteilung eines Programms auf die Seiten.

Einheitliche Seitengröße → auf der letzten Seite jedes Programm-Moduls entsteht mit hoher Wahrscheinlichkeit ein ungenutzter Leerraum.



## Probleme der virtuellen Speicherverwaltung

### 3. Das Ersetzungsproblem

Welche Segmente oder Seiten müssen ausgelagert werden, um Platz für neu benötigte Daten zu schaffen ?

#### Bei Segmentierungsverfahren:

Meist wird die Anzahl der gleichzeitig von einem Prozeß benutzbaren Segmente limitiert:

→ bei Einlagerung eines neuen Segments wird ein zuvor für diesen Prozeß benutztes Segment ausgelagert

Es ist jedoch auch eine der im folgenden für Seitenwechsel-Verfahren beschriebenen Methoden möglich:



## Probleme der virtuellen Speicherverwaltung

### Ersetzungsproblem bei Seitenwechselverfahren

5 gängigste Strategien zum Ersetzen einer Seite:

- **FIFO (first-in-first-out)**: die sich am längsten im Hauptspeicher befindende Seite wird ersetzt
- **LIFO (last-in-first-out)**: die zuletzt eingelagerte Seite wird ersetzt
- **LRU (least recently used)**: die Seite, auf die am längsten nicht zugegriffen wurde, wird ersetzt



## Probleme der virtuellen Speicherverwaltung

- **LFU (least frequently used)**: die seit ihrer Einlagerung am seltensten benutzte Seite wird ersetzt
- **LRD (least reference density)**: Mischung aus LRU und LFU. Die Seite mit der geringsten Zugriffsdichte (Anzahl Zugriffe / Einlagerungszeitraum) wird ersetzt

Daneben werden bevorzugt solche Seiten ersetzt, die nicht verändert wurden → kein Rückschreiben der geänderten Seite erforderlich.

