Betriebssysteme

MMU, ÜTabellen und Kleine Seitenfehler

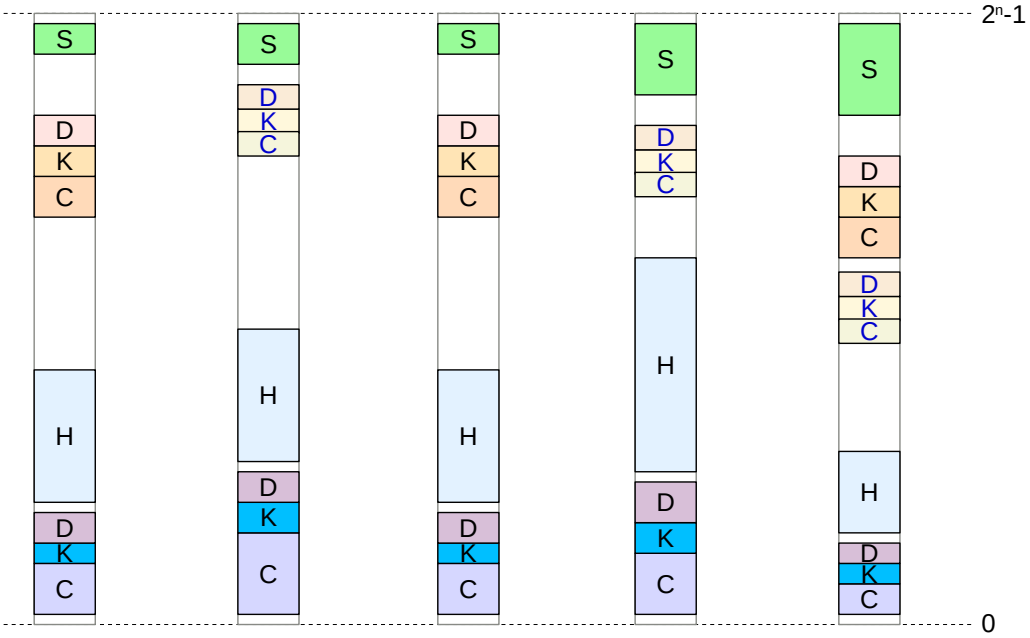
DHBW Stuttgart

Roland Weber

Nur teilweise im Buch…

Kapitel “Übersetzungstabellen”
2025: Kapitel 14 ab Seite 141 bis 146
außer 14.3.x auf Seite 145

Logische Adressräume

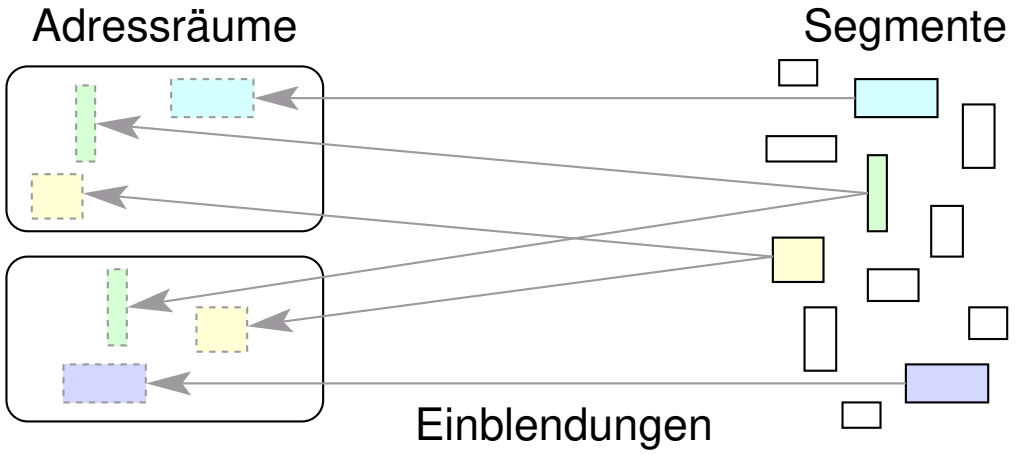


Jede Instanz hat ihre eigenen Adressen, die nur in ihrem Adressraum gelten. Die gleiche Adresse hat in verschiedenen Adressräumen unterschiedliche Bedeutungen. Wie Telefonnummern ohne Vorwahl.

In einem Adressraum dürfen sich die Segmente nicht überlappen. Jede gültige Adresse bezieht sich auf ein Byte des Speichers, dessen Inhalt von nur einem Segment bestimmt wird.

Code und Konstanten von dynamischen Bibliotheken und mehrfach laufenden Programmen erscheinen in mehreren Adressräumen. Sie müssen ggfs. an unterschiedlichen Adressen anfangen, um Überlappungen zu vermeiden. Die Größe des Segments ist aber in jedem Adressraum gleich.

Bausteine für Adressräume



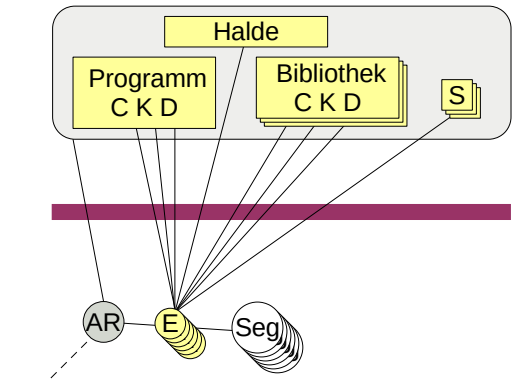
Wdh. aus dem Kapitel “Software”…

Jede Instanz hat ihren eigenen Adressraum.

Segmente sind Speicherbereiche.

Einblendungen bestimmen, welche Segmente in welchen Adressräumen zugreifbar sind.

Verwaltungsdaten



Für Instanzen sichtbar:

Größe des Segments
=> Anzahl Seiten / Einträge
Basisadresse der Einblendung
=> Logische Seitennummern
Rechte für Segment und Einblendung
=> Flags

Interne Daten des Kerns:

Lage jeder Seite des Segments
=> Physische Seitennummern

Übersetzungstabelle
pro Adressraum

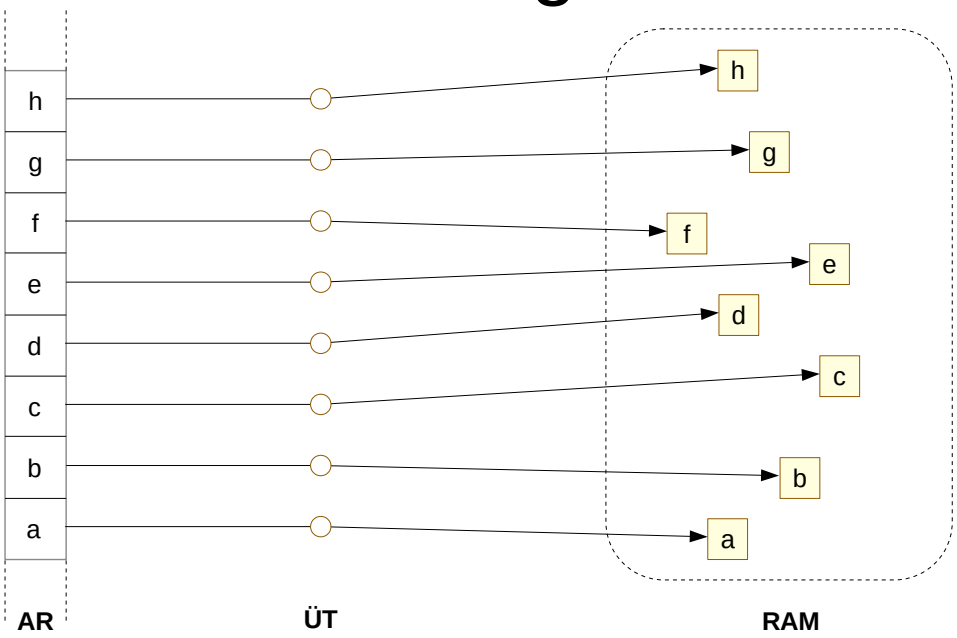
Logische Seitennummer	Physische Seitennummer	Flags

Der Kern muss für jeden Adressraum eine Übersetzungstabelle pflegen. Das genaue Format hängt von der Prozessorarchitektur ab. Instanzen arbeiten nur mit Kernobjekten: AR, E, Seg

- Jedes Segment hat eine Größe, die ein Vielfaches der Seitengröße ist. Daher weiß der Kern, wieviele Einträge das Segment in einer ÜT umfasst.
- Jede Einblendung hat eine Basisadresse, ab der das Segment im jeweiligen AR liegt. Daraus leitet der Kern logische Seitennummern ab.
- Zugriffsrechte können bei Segment und Einblendung stehen, sie bestimmen die Flags.

In einer internen Tabelle merkt sich der Kern, wo jede Seite eines Segments liegt. Daher stammen die physischen Seitennummern in den ÜT-Einträgen.

Übersetzungstabelle



ÜT für MMU zeigt auf physische Seiten im RAM.

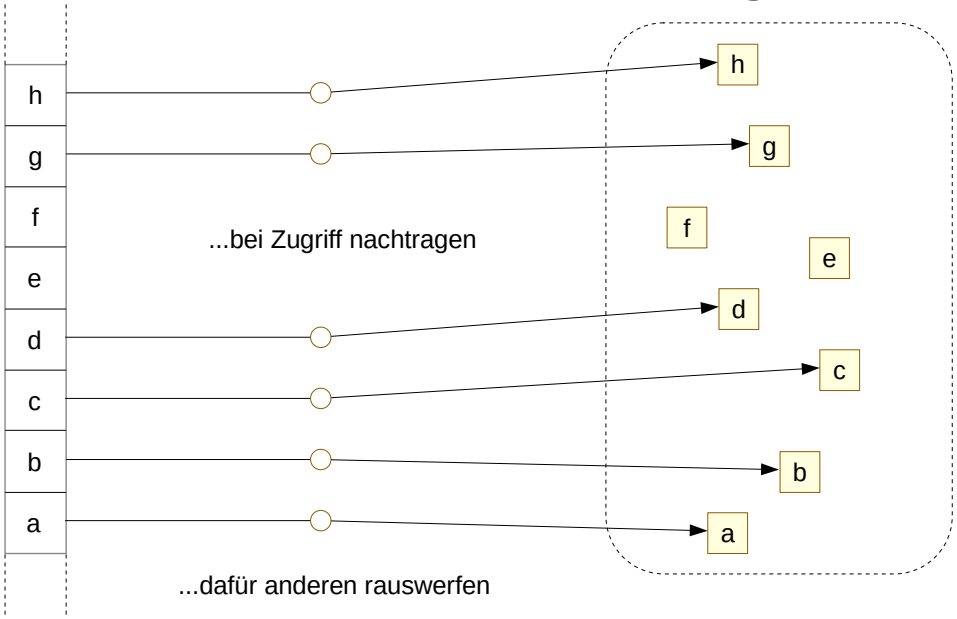
Einfachster Fall: vollständige Übersetzungstabelle

Unrealistisch.

Rechenbeispiel:
Instanz belegt 4 GB Speicher, Seitengröße 4K
=> mehr als 1 Million Übersetzungseinträge (2^20)

Bei 8 Byte pro ÜE sind das 8 MB Speicherbedarf nur für die Übersetzungstabelle.

Fehlende Einträge



Zugriffsindikator hilft bei der Auswahl

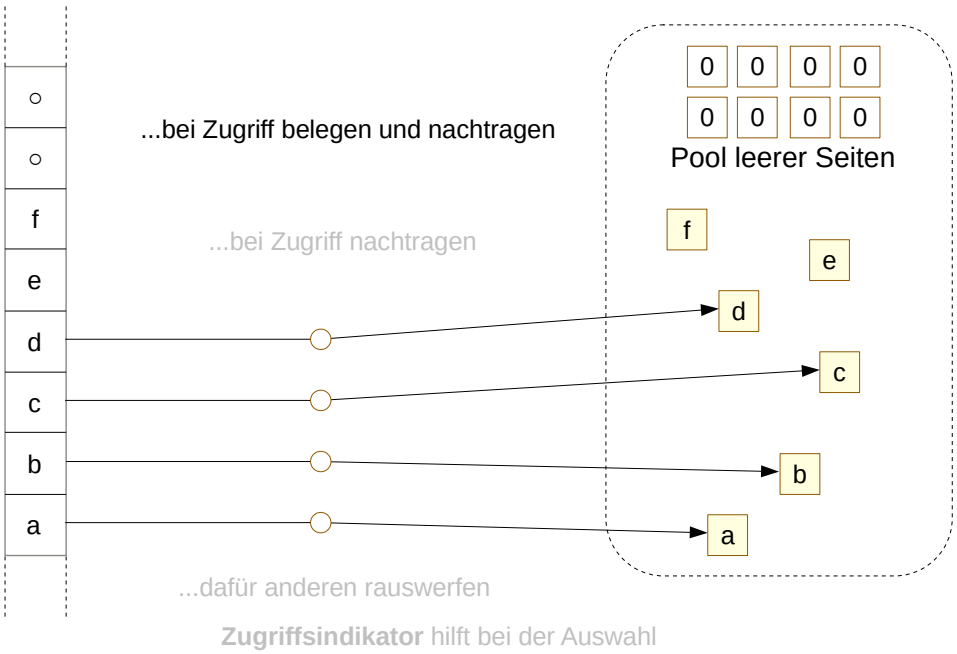
Übersetzungstabelle enthält nur einen Teil der theoretisch möglichen Übersetzungseinträge.

Zugriff auf Seite ohne ÜT-Eintrag:
- Seitenfehler
- Kern erkennt, dass ÜT-Eintrag fehlt
- Kern ergänzt den fehlenden Eintrag
  ggfs. wird ein anderer überschrieben
- Prozess läuft weiter, wiederholt den Zugriff

“kleiner Seitenfehler”, d.h. schnell behoben

Problem: Beim Ersetzen eines bestehenden Eintrags sollte der Kern besser einen wählen, der nicht gleich wieder gebraucht wird. Als Hilfe setzt die MMU einen **Zugriffsindikator** in Einträgen, die sie verwendet.

Leere Seiten



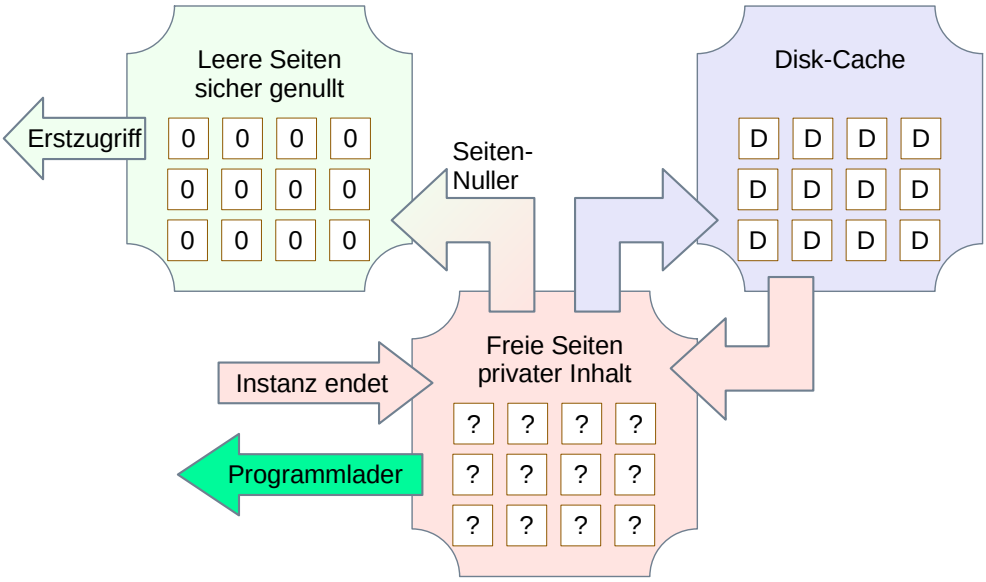
Frisch angeforderter Speicher ist leer, enthält keine sinnvollen Daten. Am besten mit Nullen gefüllt, damit keine Speicherinhalte anderer Instanzen drin stehen.

Rechenbeispiel: JVM mit 2 GB Halde startet
=> 2 GB müssen mit Nullen überschrieben werden
...aber Instanz nutzt nur 100 KB der Halde. Doof.

Leere Seiten brauchen (noch) keinen Speicher.
Zugriff auf leere Seite:
- Seitenfehler
- Kern erkennt, dass leere Seite benötigt wird
- Kern nimmt leere Seite aus einem Pool
- Kern ergänzt ÜT-Eintrag
- Prozess läuft weiter, wiederholt den Zugriff

Eine Systemfunktion verwaltet Pool mit leeren Seiten.
Falls Pool erschöpft: Prozess blockiert bis leere Seite verfügbar. Das ist dann ein größerer Seitenfehler.

Pools Freier Seiten



Links ein Pool leerer, d.h. genullter Seiten, die man für kleine Seitenfehler braucht. Eine Systemfunktion “Seitennuller” versucht, immer ein paar leere Seiten auf Vorrat in diesem Pool zu halten.

In der Mitte freie Seiten, mit schutzwürdigen Inhalten beendeter Instanzen, die überschrieben werden müssen. Dem Programmlader könnte man zugestehen, dass er beim Laden von Segmenten die Seiten überschreibt. Aber nur, wenn auch alle beteiligten Treiber vertrauenswürdig sind.

Rechts ein Disk-Cache, er beschleunigt Zugriffe auf Dateisysteme. Die dafür benötigten Seiten im RAM sind prinzipiell frei, da die Inhalte bei Bedarf wieder von Disk gelesen werden können. Wieder müssen alle beteiligten Treiber vertrauenswürdig sein, um auf das Nullen zu verzichten.