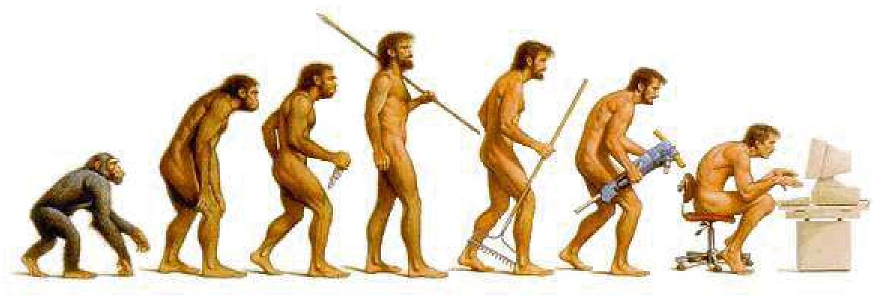
Überblick

> *Wenn wir nicht absichtlich unsere Augen verschließen, so können wir nach unseren jetzigen Kenntnissen annähernd unsere Abstammung erkennen, und dürfen uns derselben nicht schämen. (Charles Darwin)*



Zusammenhänge erkennen — Gestern & Heute

Sensibilität für (die Notwendigkeit von) Betriebssystemfunktionen schärfen

Betriebssystem als Dienstleister verstehen, nicht als Selbstzweck

- eine Maschine, mit dessen Hilfe Rechnerhardware für (Anwendungs- und/oder System-) Programme nutzbar gemacht werden soll
- die Fähigkeiten dieser Maschine bestimmen sich in erster Linie durch den Anwendungszweck und die Hardware des Rechners
- in zweiter Linie stehen Fertigkeiten des Entwicklungspersonals, obwohl diese wesentlich zur Zufriedenheit über das Endprodukt beitragen

Evolutionsgeschichte von Betriebssystemen im Zeitraffer

- mehr als nur ein „Abfallprodukt" der nachfolgenden Betrachtungen
- zeigt das „Aufwachsen" von Systemsoftware in funktionaler Hinsicht

Generationen von Hardware und Betriebssoftware

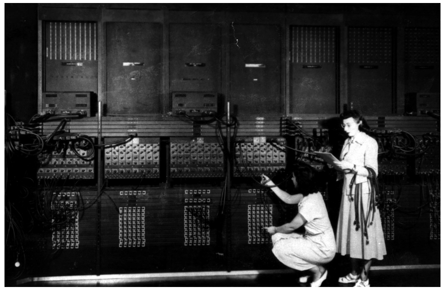
Zeitliche Einordnung von Technologiemerkmalen: „unscharf", Übergänge fließend

Epoche	Hardware	Betriebssoftware	Rechnerbetriebsart
1945	Röhre	Lader	Stapelbetrieb
1955	Halbleiter, Gatter	Stapelsystem	Echtzeitbetrieb
1965	MSI/LSI, WAN	Dateisystem, VM	Mehrprogrammbetrieb
1975	VLSI, LAN	Kommunikation	Mehrzugangsbetrieb
1985	ULSI, RISC	Klient/Anbieter	Netzbetrieb
1995	WLAN, RFID, SoC	*Middleware*	Integrationsbetrieb

Legende:
- MSI, LSI, VLSI, ULSI — *medium, large, very large, ultra large scale integration*
- LAN — *local area network* (Ethernet)
- WAN — *wide area network* (Arpanet)
- WLAN — *wireless* LAN
- RISC — *reduced instruction set computer*
- RFID — *radio frequency identification* (Funkerkennung)
- SoC — *system on chip*
- VM — *virtual memory*

Am Anfang war das Feuer. . .

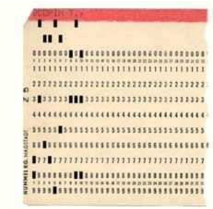
ENIAC (*electronic numerical integrator and computer*), 1945



A small amount of time is required in preparing the ENIAC for a problem by such steps as setting program switches, putting numbers into the function table memory by setting its switches, and establishing connections between units of the ENIAC for the communication of programming and numerical information.
(Pressemitteilung, DoD, 1946)

Ein elektronischer Allzweckrechner von 30 Tonnen Gewicht, mit einer 15 $m \times 3\,m \times 5\,m$ großen Zentraleinheit und 30 m langen Frontplatte. Für seinen Betrieb waren 18 000 Röhren zuständig, die elektrische Anschlussleistung belief sich auf etwa 174 000 Watt.

Lochkartenverarbeitung



Hermann Holerith (1860–1929), Begründer der maschinellen Datenverarbeitung. IBM ließ sich 1928 das Format patentieren: 80 Spalten, 12 Zeilen und rechteckige Löcher an den Schnittpunkten.

Ziffernlochkarte (engl. *numeric punch card*)

- Datenaufzeichnung durch Lochung (Loch $\mapsto$ 1, kein Loch $\mapsto$ 0)
  - Dezimalzahlen werden mit einer Lochung dargestellt
  - Buchstaben und Sonderzeichen mit zwei oder drei
  - negative Vorzeichen ggf. durch eine Lochung in Zeile 11
- manuell ca. 15 000 Zeichen/h, maschinell 4 000–10 000 Lochkarten/h

Manuelle Bestückung des Rechners

Vollständige Kontrolle beim Programmierer/Operateur

1. Programme mit Lochkartenstanzer (engl. *card puncher*) ablochen
2. Übersetzerkarten in den Lochkartenleser (engl. *card reader*) einspeisen
3. Lochkartenleser durch Knopfdruck starten
4. Programmkartenstapel (zur Übersetzung) in den Kartenleser einlegen
5. Eingabelochkartenstapel in den Lochkartenleser einlegen
6. Leere Lochkarten für die Ausgabe in den Lochkartenstanzer einlegen
7. Ausgabelochkartenstapel in den Lochkartenleser des Druckers einlegen
8. Ergebnisse der Programmausführung vom Drucker abholen

⚠ Urlader (z.B. Lochkartenleseprogramm) in den Rechner einspeisen

Manuelle Bestückung des Rechners (Forts.)

Eingabe eines Urladers, z.B. des Lochkartenleseprogramms

Rechnersystem durch Ureingabe (engl. *bootstrap*) laden:

1. Bitmuster einer Speicherwortadresse über Schalter einstellen
   - die Adresse der nächsten zu beschreibenden Stelle im Arbeitsspeicher
2. eingestellten Adresswert in den PC der CPU laden
3. Bitmuster des Speicherwortinhalts über Schalter einstellen
   - ein Befehl, ein Direktwert oder eine Operandenadresse des Programms
4. eingestellten Datenwert in den Arbeitsspeicher laden

☞ heute sind Urlader nicht-flüchtig im ROM/EPROM gespeichert!!!

⚠ Bedienung, Mensch, Permanenz

Automatisierte Bestückung des Rechners

Verzicht auf Maßnahmen zur manuellen Einspeisung von Dienstprogrammen

Dienstprogramme sind zum Abruf bereit im Rechnersystem gespeichert:

- Systembibliothek, „Datenbank“ (Dateiverwaltung)

Anwendungsprogramme fordern Dienstprogramme explizit an:

- spezielle Steuerkarten sind dem Lochkartenstapel hinzugefügt
  - Systemkommandos, die auf eigenen Lochkarten kodiert sind
  - Anforderungen betreffen auch Betriebsmittel (z.B. Speicher, Drucker)
- der erweiterte Lochkartenstapel bildet einen Auftrag (engl. *job*)
  - an einen Kommandointerpretierer (engl. *command interpreter*)
  - formuliert als Auftragssteuersprache (engl. *job control language*, JCL)
- die Programmausführung erfolgt ohne weitere Interaktion

☞ eignet sich (nach wie vor) zur Bewältigung von „Routineaufgaben“

⚠ vollständige Auftragsbeschreibung (inkl. Betriebsmittelbedarf)

Automatisierte Bestückung des Rechners (Forts.)

Auftragssteuersprache (FORTRAN Monitoring System, FMS, 1957)

```
*JOB, 42, ARTHUR DENT
*XEQ
*FORTRAN
```

Programmkarten

```
*DATA
```

Datenkarten

```
*END
```


Automatisierte Bestückung des Rechners (Forts.)

Residentes Steuerprogramm (engl. *resident monitor*)

Auftragssteuerung durch (Arbeitsspeicher-) residente Systemsoftware:

- Kommandointerpretierer, Lochkartenleseprogramm, E/A-Prozeduren
- Systemprogramme, die ein „embryonales Betriebssystem“ bilden

Solitär: einzelstehende und getrennt übersetzte Einheit, verschiedenartig vom Anwendungsprogramm entkoppelt

- Einsprungtabelle (engl. *jump table*)
- partielle Interpretation von Monitoraufrufen
- getrennte physikalische Adressräume (Schutzgatter)
- Arbeitsmodi (Benutzer-/Systemmodus, privilegiert/nicht-privilegiert)

⚠ Arbeitsgeschwindigkeit der Peripherie, sequentielle Ein-/Ausgabe

Adressraumschutz durch Abteilung

Partitionierung des physikalischen Adressraums

Schutzgatter trennt den vom residenten Steuerprogramm und vom transienten Programm belegten Arbeitsspeicherbereich

- Anwendungsprogramme werden komplett, d.h. statisch gebunden
  - Schutzgatter ≡ unveränderliche, physikalische Speicheradresse
  - Relokationskonstante beim Binden, Ladeadresse transienter Programme
- ggf. ist gewisse Flexibilität durch ein Schutzgatterregister gegeben
  - veränderliche, physikalische Speicheradresse; programmierbarer Wert
  - Relokationsvariable beim Binden, relative Programmadressen (Basis 0)

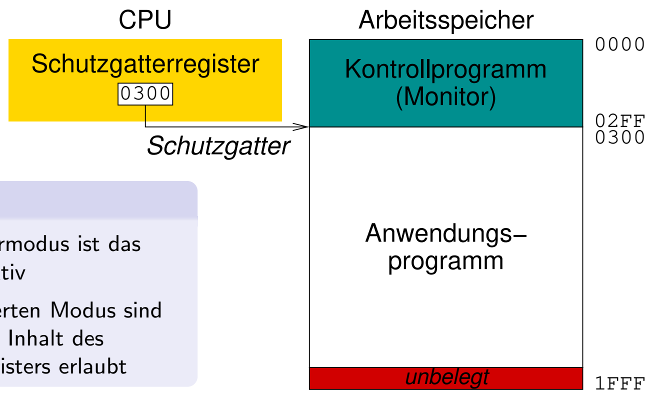
Speicherverwaltung ist statisch, geschieht vor Programmausführung

- ein verschiebender Lader legt die Lage im Arbeitsspeicher fest
  - obligatorisch bei Verwendung des Schutzgatterregisters, optional sonst
- zur Programmlaufzeit ist zusätzlicher Speicher nur bedingt zuteilbar
  - spätestens zum Ladezeitpunkt muss weiterer Bedarf bekannt sein

⚠ Überbelegung des Arbeitsspeichers (zu großes Programm)

Adressraumschutz durch Abteilung (Forts.)

Schutzgatterregister (engl. *fence register*)

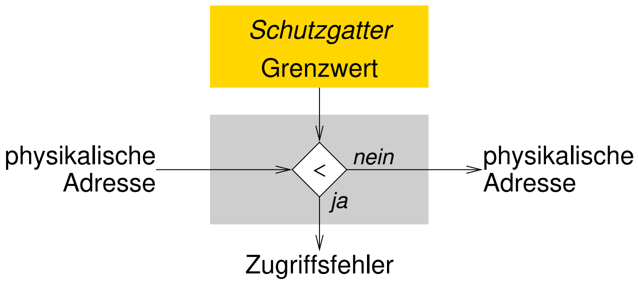


Arbeitsmodi

- nur im Benutzermodus ist das Schutzgatter aktiv
- nur im privilegierten Modus sind Änderungen am Inhalt des Schutzgatterregisters erlaubt

Adressraumschutz durch Abteilung (Forts.)

Zugriffsfehler führt zum Abbruch (synchrone Programmunterbrechung, Trap)



⚠ „interne Fragmentierung“ (Zugriff auf unbelegten Bereich)

Abgesetzter Betrieb

Berechnung erfolgt getrennt von Ein-/Ausgabe (engl. *off-line*)

Satellitenrechner zur Ein-/Ausgabe mit „langsamer Peripherie“

- Kartenleser, Kartenstanzer, Drucker
- Ein-/Ausgabedaten werden über Magnetbänder transferiert

Hauptrechner zur Berechnung mit „schneller Peripherie“

- Be-/Entsorgung des Hauptspeichers auf Basis von Bandmaschinen
- dadurch erheblich verkürzte Wartezeiten bei der Ein-/Ausgabe

☞ heute finden zusätzlich Wechselplatten Verwendung

⚠ sequentieller Bandzugriff, feste Auftragsreihenfolge

Abgesetzter Betrieb (Forts.)

Dedizierte Rechner/Geräte für verschiedene Arbeitsphasen

Phase	Medium			Rechner
Eingabe	Text/Daten Lochkarten	⇒ ⇒	Lochkarten Magnetband	Satellitenrechner
↓				↓
Verarbeitung	Magnetband Arbeitsspeicher	⇒ ⇒	Arbeitsspeicher Magnetband	Hauptrechner
↓				↓
Ausgabe	Magnetband Daten	⇒ ⇒	Lochkarten Drucker	Satellitenrechner

Überlappte Ein-/Ausgabe
Nebenläufige Ausführung der Gerätetreiber

Speicherdirektzugriff (engl. *direct memory access*, DMA, 1954):

- unabhängig von der CPU arbeitende E/A-Kanäle
  - zwischen einem E/A-Gerät und dem Arbeitsspeicher (RAM)
- Datentransfer durch ein Kanalprogramm (*cycle stealing*, 1958)

E/A-Bereitschaft signalisiert durch asynchrone Programmunterbrechungen

- die E/A-Geräte fordern der CPU weitere E/A-Aufträge ab
- Ein-/Ausgabe und Berechnung desselben Programms überlappen sich
- nebenläufige Aktivitäten werden koordiniert: Synchronisation

Leerlauf beim Auftragswechsel

Überlappte Auftragsverarbeitung
(engl. *single-stream batch monitor*)

Verarbeitungsstrom sequentiell auszuführender Programme bzw. Aufträge

- während der Ausführung eines laufenden Auftrags den nachfolgenden Auftrag bereits in den Arbeitsspeicher einlesen (d.h. zwischenpuffern)
- Programme im Vorgriffsverfahren (engl. *prefetching*) abarbeiten

Auftragseinplanung (engl. *job scheduling*) liefert Abarbeitungsreihenfolge

- statische/dynamische Einplanung nach unterschiedlichsten Kriterien
  - Ankunftszeit, erwartete Laufzeit, erwarteter Betriebsmittelbedarf
- die Aufträge werden dazu im Hintergrundspeicher zusammengestellt
  - *wahlfreier Zugriff* (z.B. Plattenspeicher) vermeidet Engpässe

Arbeitsspeicher, Monopolisierung der CPU, Leerlauf bei Ein-/Ausgabe

Abgesetzte Ein-/Ausgabe
Spooling (engl. *simultaneous peripheral operations online*)

Pufferbereiche zur Entkopplung von Berechnung und Ein-/Ausgabe

- Programmausführung ist eine periodische Abfolge von zwei Phasen:
  CPU-Stoß (engl. *CPU burst*) vergleichsweise schnell
    - Aktivitäten eines Programms, die Berechnungen betreffen

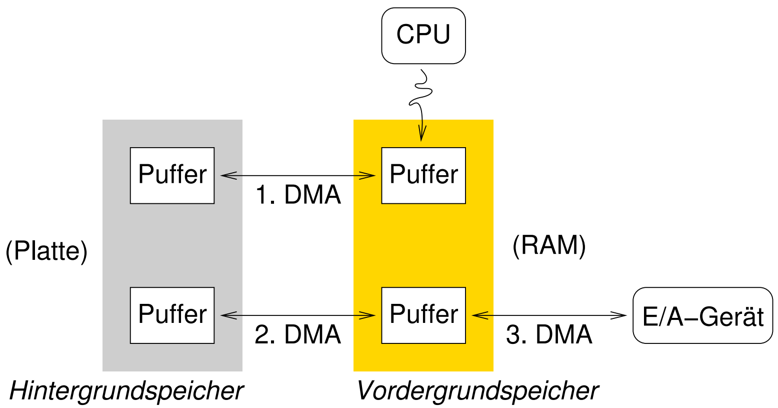
  E/A-Stoß (engl. *I/O burst*) vergleichsweise langsam
    - Aktivitäten eines Programms, die Ein-/Ausgabe betreffen
- Pufferung im Hintergrundspeicher (engl. *secondary/backing store*)
  - Berechnungen im Vordergrundspeicher (engl. *primary/main store*)
- per DMA werden die Daten („nebenbei“) hin- und hertransferiert

Systemprogramme starten/überwachen die E/A-Vorgänge

- heute z.B. `lpr(1)` und `lpd(8)` unter Unix

Leerlauf im Wartezustand (Einprogramm(-/-prozess)betrieb)

Abgesetzte Ein-/Ausgabe (Forts.)



Verarbeitung von Programmen in Echtzeit

Zustandsänderung von Programmen wird zur Funktion der realen Zeit [16]

- korrektes Verhalten des Systems hängt nicht nur von den logischen Ergebnissen von Berechnungen ab
- zusätzlicher Aspekt ist der physikalische Zeitpunkt der Erzeugung und Verwendung der Berechnungsergebnisse

DIN 44300

Echtzeitbetrieb ist ein Betrieb eines Rechensystems, bei dem Programme zur Verarbeitung anfallender Daten ständig betriebsbereit sind derart, dass die Verarbeitungsergebnisse innerhalb einer vorgegebenen Zeitspanne verfügbar sind. Die Daten können je nach Anwendungsfall nach einer zeitlich zufälligen Verteilung oder zu vorbestimmten Zeitpunkten anfallen.

☞ Whirlwind (MIT, 1951), AN/FSQ-7 (Whirlwind II, IBM, 1957)
☞ SAGE (*semi-automatic ground environment*, 1958–1983)

Echtzeitfähigkeit bedeutet Rechtzeitigkeit

Zuverlässige Reaktion des Rechensystems auf Umgebungsereignisse

Geschwindigkeit liefert keine Garantie, um rechtzeitig Ergebnisse von Berechnungen abliefern und Reaktionen darauf auslösen zu können

- die im Rechensystem verwendete Zeitskala muss mit der durch die Umgebung vorgegebenen identisch sein
- der Begriff „Zeit“ ist keine intrinsiche Eigenschaft des Rechensystems

Determiniertheit und Determinismus

- bei ein und derselben Eingabe sind verschiedene Abläufe zulässig, alle Abläufe liefern jedoch stets das gleiche Resultat
- zu jedem Zeitpunkt ist bestimmt, wie weitergefahren wird

Terminvorgaben gelten als weich, fest oder hart (siehe S. 5-49)

Maßnahmen zur Leistungssteigerung

Trennung von Belangen (engl. *separation of concerns*)

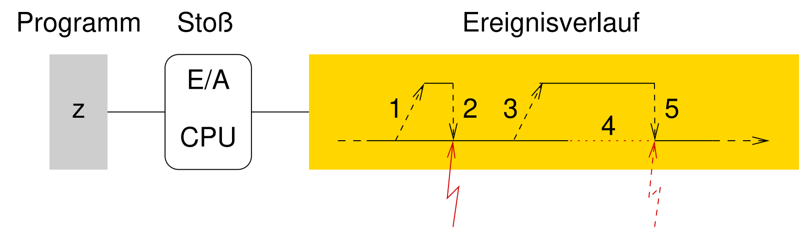
Multiprogrammbetrieb (engl. *multiprogramming*) und Simultanbetrieb (engl. *multiprocessing*), auch Maßnahmen zur Strukturierung:

- Multiplexen der CPU zwischen mehreren Programmen
  - Nebenläufigkeit (engl. *concurrency*)
- Abschottung der sich in Ausführung befindlichen Programme
  - Adressraumschutz (engl. *address space protection*)
- Überlagerung unabhängiger Programmteile

If we believe in data structures, we must believe in independent (hence simultaneous) processing. For why else would we collect items within a structure? Why do we tolerate languages that give us the one without the other? (Alan Perlis [17])

Überlappte Ein-/Ausgabe im Einpro{gramm,zess}betrieb

Leerlauf der CPU im Wartezustand



(1) Programm$_z$ löst einen E/A-Stoß nebenläufig zum CPU-Stoß aus
(2) ein Interrupt zeigt die Beendigung des E/A-Stoßes an
(3) die Unterbrechungsbehandlung startet einen weiteren E/A-Stoß
(4) Programm$_z$ muss auf E/A warten, sein CPU-Stoß endet
(5) die CPU läuft leer, bis der E/A-Stoß zum Abschluss kommt

⚠ Durchsatz, Auslastung

Multiplexen des Prozessors

Die CPU „gleichzeitig“ von mehreren Programmen benutzen

Wartezeit von Programm$_x$ wird als Laufzeit von Programm$_y$ genutzt

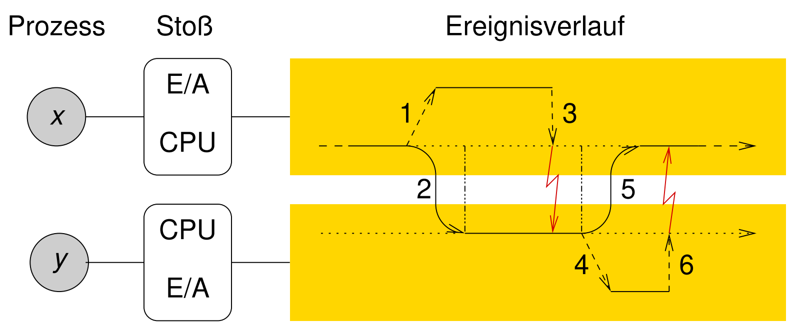
- aktives Warten (engl. *busy waiting*) auf Ereigniseintritte vermeiden
- stattdessen passives Warten betreiben und die CPU einem anderen lauffähigen Programm zuteilen

Abstraktion von wartenden/laufenden Programmen, liefert ein generisches Konzept zur Programmverarbeitung -und verwaltung

- der Begriff „Prozess“: ein beliebiges „Programm in Abarbeitung“
- für jedes zu ladende Programm wird ein Prozess erzeugt

Einplanung (engl. *scheduling*) und Einlastung (engl. *dispatching*) von Prozessen regeln die Verarbeitung lauffähiger Programme

Programmverarbeitung im Mehrprozessbetrieb



Betriebssystem $\equiv$ mehrfädiges Programm (engl. *multi-threaded program*)

- ein Programm, in dem mehrere Prozesse „gleichzeitig“ aktiv sind

Programmverarbeitung im Mehrprozessbetrieb (Forts.)

Kooperation von Prozess$_x$ und Prozess$_y$, nach erfolgter Einplanung:

(1) Prozess$_x$ löst einen E/A-Stoß und beendet seinen CPU-Stoß

(2) Prozess$_y$ wird eingelastet und beginnt seinen CPU-Stoß

(3) Beendigung des E/A-Stoßes von Prozess$_x$ unterbricht Prozess$_y$
- die Unterbrechungsbehandlung überlappt sich mit Prozess$_y$

(4) Prozess$_y$ löst einen E/A-Stoß und beendet seinen CPU-Stoß

(5) Prozess$_x$ wird eingelastet und beginnt seinen CPU-Stoß

(6) Beendigung des E/A-Stoßes von Prozess$_x$ unterbricht Prozess$_x$
- die Unterbrechungsbehandlung überlappt sich mit Prozess$_x$

⚠ Adressraumschutz, Koordination

Adressraumschutz

Abschottung von Programmen durch Eingrenzung/Segmentierung

Bindungszeitpunkt von Programm- an Arbeitsspeicheradressen beeinflusst das Modell zur Abschottung der Programme — und umgekehrt:

vor Laufzeit Schutz durch Eingrenzung

- Programme laufen (weiterhin) im physikalischen Adressraum
- ein verschiebender Lader besorgt die Bindung zum *Ladezeitpunkt*
- die CPU überprüft die generierten Adressen zur Ausführungszeit

zur Laufzeit Schutz durch Segmentierung

- jedes Programm läuft in seinem eigenen logischen Adressraum
- der Lader bestimmt die Bindungsparameter (phys. Basisadresse)
- die CPU (bzw. MMU) besorgt die Bindung zur Ausführungszeit

In beiden Fällen richtet der Binder Programme relativ zu einer logischen Basis aus, meistens zur Basis 0.

Adressraumschutz durch Eingrenzung

Begrenzungsregister (engl. *bounds register*) legen die Unter-/Obergrenze eines Programms im physikalischen Adressraum fest

- für jedes Programm wird ein solches Registerpaar verwaltet
  - die sog. *Softwareprototypen* der Adressüberprüfungshardware
- bei Prozesseinlastung erfolgt die Abbildung auf die Hardwareregister

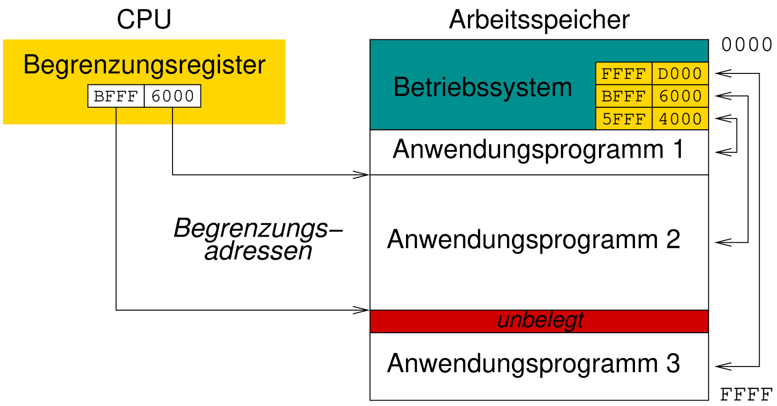
Speicherverwaltung ist statisch, geschieht vor Programmausführung

- der verschiebende Lader legt die Lage im Arbeitsspeicher fest
- zur Programmlaufzeit ist zusätzlicher Speicher nur bedingt zuteilbar
  - spätestens zum Ladezeitpunkt muss weiterer Bedarf bekannt sein

⚠ Fragmentierung, Verdichtung

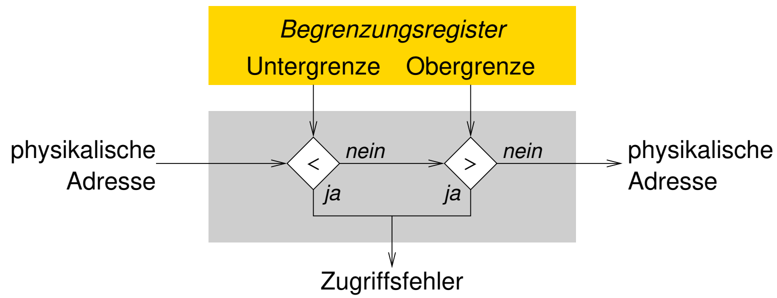
Adressraumschutz durch Eingrenzung (Forts.)

Begrenzungsregister (engl. *bounds register*)



Adressraumschutz durch Eingrenzung (Forts.)

Zugriffsfehler führt zum Abbruch (synchrone Programmunterbrechung, Trap)



Adressraumschutz durch Segmentierung

Basis-/Längenregister (engl. *base/limit register*) geben Programme vom physikalischen Adressraum abstrahierende logische Adressräume

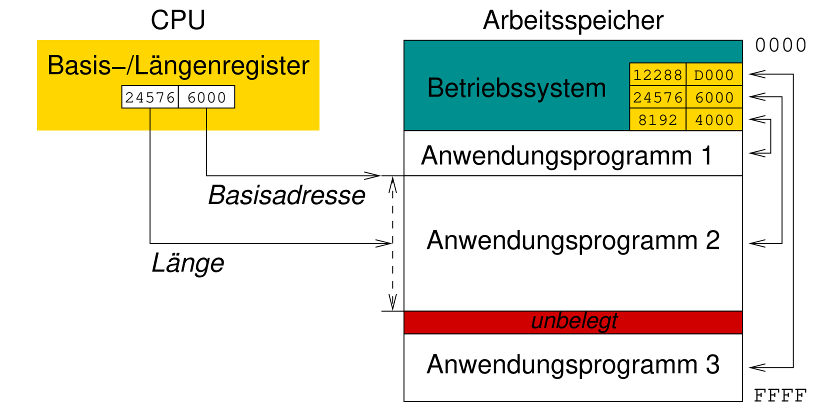
- für jedes Programm wird ein solches Registerpaar verwaltet
  - die sog. *Softwareprototypen* der Adressumsetzungshardware
- bei Prozesseinlastung erfolgt die Abbildung auf die Hardwareregister

Speicherverwaltung ist dynamisch, geschieht zur Programmausführung

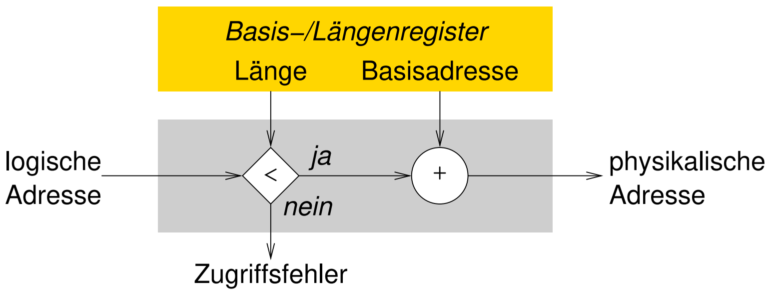
- der Lader legt die initiale Lage im Arbeitsspeicher fest
- zur Programmlaufzeit ist zusätzlicher Speicher „beliebig" zuteilbar
  - limitierend ist der jeweils physikalisch noch freie Arbeitsspeicher

⚠ Überbelegung des Arbeitsspeichers (zu große/viele Programme)

Adressraumschutz durch Segmentierung (Forts.)
Basis-/Längenregister (engl. *base/limit register*)



Adressraumschutz durch Segmentierung (Forts.)
Zugriffsfehler führt zum Abbruch (synchrone Programmunterbrechung, Trap)



☞ Grundlage für eine MMU des heutigen Stands der Technik

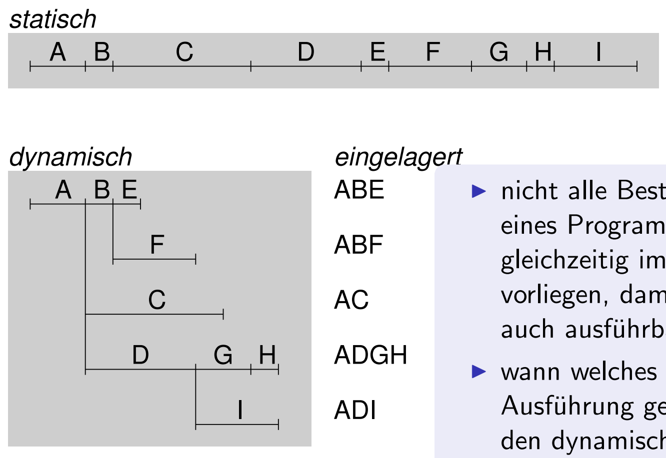
Dynamisches Laden
Überlagerung von Programmteilen im Arbeitsspeicher

Technik der Überlagerung (engl. *overlay*)
Ein Programm, das einschließlich seiner Daten die Kapazität des Hauptspeichers übersteigt, wird in hinreichend kleine Teile zergliedert, die nicht ständig im Hauptspeicher vorhanden sein müssen sondern stattdessen im Hintergrundspeicher (Trommel, Platte) vorgehalten werden.

Überlagerungen werden nur bei Bedarf (engl. *on demand*) nachgeladen
- das Nachladen ist programmiert, d.h. in den Programmen festgelegt
- die Entscheidung zum Nachladen fällt zur Programmlaufzeit

⚠ finden der (zur Laufzeit) „optimalen“ Überlagerungsstruktur

Dynamisches Laden (Forts.)
Einsparung von Arbeitsspeicher zur Programmlaufzeit



- nicht alle Bestandteile (A – I) eines Programms müssen gleichzeitig im Arbeitsspeicher vorliegen, damit das Programm auch ausführbar ist
- wann welches Programmteil zur Ausführung gelangt, ist durch den dynamischen Ablauf des Programms selbst festgelegt

Dynamisches Laden (Forts.)
Überlagerungsstruktur eines Programms

Programme, die Überlagerungen enthalten, sind (grob) dreigeteilt:

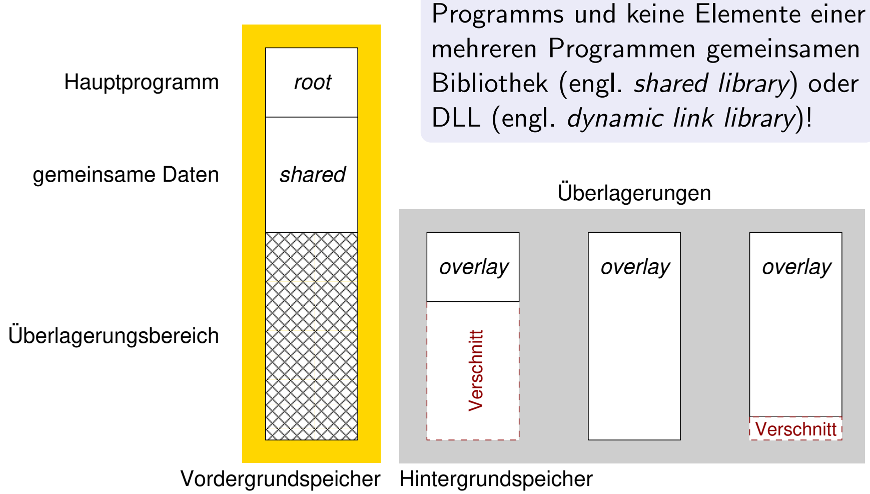
1. ein arbeitsspeicherresidenter Programmteil (engl. *root program*)
2. mehrere in Überlagerungen zusammengefasste Unterprogramme
3. ein gemeinsamer Datenbereich (engl. *common section*)

Überlagerungen sind das Ergebnis einer Programmzerlegung zur Programmier-, Übersetzungs- und/oder Bindezeit

- manuelle bzw. automatisierte Abhängigkeitsanalyse des Programms
- Anzahl und Größe von Überlagerungen werden statisch festgelegt
- lediglich aktivieren (d.h. laden) von Überlagerungen ist dynamisch

Dynamisches Laden (Forts.)
Funktion der Speicherverwaltung

Überlagerungen sind Teile eines Programms und keine Elemente einer mehreren Programmen gemeinsamen Bibliothek (engl. *shared library*) oder DLL (engl. *dynamic link library*)!



Simultanverarbeitung mehrerer Auftragsströme
(engl. *multiprocessing, multi-stream batch monitor*)

Verarbeitungsströme sequentiell auszuführender Programme/Aufträge

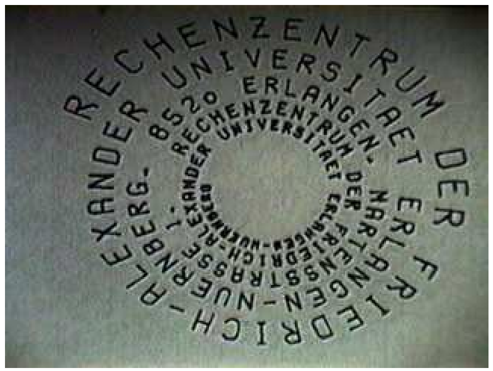
- pseudo-parallele Abarbeitung mehrerer Stapel im Multiplexverfahren
  - sequentielle Ausführung der Programme desselben Auftragstapels
  - überlappende Ausführung der Programme verschiedener Auftragstapel
- „wer [innerhalb des Stapels] zuerst kommt, mahlt zuerst"...

Stapelwechsel verlaufen kooperativ (engl. *cooperative*) oder kommen verdrängend (engl. *preemptive*) zustande

- kooperativ bei (programmierter) Ein-/Ausgabe und Programmende
  - d.h., bei Beendigung des CPU-Stoßes eines Prozesses
- verdrängend bei Ablauf einer Frist (engl. *time limit*)

⚠ interaktionsloser Betrieb, Mensch/Maschine-Schnittstelle

Rechnerbetrieb am RRZE um 1973
http://www.rrze.uni-erlangen.de/wir-ueber-uns/publikationen/das-rrze-der-film.shtml



Gleichzeitiger, interaktiver Zugang für mehrere Benutzer
Dialogbetrieb (engl. *conversational mode*)

Benutzereingaben und deren Verarbeitung wechseln sich anhaltend ab

- E/A-intensive Anwendungsprogramme interagieren mit den Benutzern
- Zugang über Dialogstationen (engl. *interactive terminals*)
  - Datensichtgerät und Tastatur (seit 1950er Jahren, Whirlwind/SAGE)
  - später die *Maus* (Engelbart/English, SRI, 1963/64; vorgestellt 1968)

Einplanung bevorzugt interaktive (E/A-intensive) Prozesse

- Beendigung von Ein-/Ausgabe führt zur „prompten“ Neueinplanung
  - im Falle von E/A-Operationen, die sich blockierend auswirken
- Benutzer erfahren eine schnelle Reaktion insb. auf Eingaben
  - sofern auch die Einlastung von Prozessen „prompt“ geschieht

Zusatz zum Stapelbetrieb, Monopolisierung der CPU, Sicherheit

Hintergrundbetrieb
„Graue Eminenz“ gleich im Hintergrund die Fäden (engl. *threads*) ziehen

Programme „im Vordergrund“ starten und „im Hintergrund“ verarbeiten

- interaktiv Aufträge annehmen, ausführen und dabei überwachen
  - d.h. starten, stoppen, fortführen und ggf. abbrechen
- zur selben Zeit sind mehrere Programme nebenläufig/parallel aktiv
- im Ergebnis werden mehrere Aufgaben „gleichzeitig“ bearbeitet

Mischbetrieb

- E/A-intensive Programme im Vordergrund ausführen
- CPU-intensive Programme im Hintergrund ausführen

Vordergrund	Hintergrund
Echtzeitbetrieb	Dialogbetrieb
Dialogbetrieb	Stapelbetrieb

Arbeitsspeicher

Teilnehmerbetrieb
Dialogstationen können eigene Dialogprozesse absetzen

Zeitscheibe (engl. *time slice*) als „Betriebsmittel“

- jeder Prozess bekommt die CPU nur für eine Zeitspanne zugeteilt

  time sharing (engl.) Prozesse erhalten das Recht, die CPU für einige Zeit zur Abarbeitung des ihnen zugeordneten Programme zu nutzen
- endet die Zeitspanne, werden die Prozesse neu eingeplant
  - ggf. erfolgt die Verdrängung (engl. *preemption*) des laufenden Prozesses
  - d.h. ihm wird die CPU zu Gunsten eines anderen Prozesses entzogen

Zeitgeber (engl. *timer*) sorgen für zyklische Programmunterbrechungen

- das Intervall ist durch die Zeitscheibenlänge vorgegeben
- Monopolisierung der CPU ist nicht mehr möglich: CPU-Schutz

Arbeitsspeicher, Einplanung, Einlastung, Ein-/Ausgabe, Sicherheit

Teilnehmerbetrieb (Forts.)
Bahnbrecher und Wegbereiter

CTSS (*Compatible Time-Sharing System* [18], MIT, 1961)

Time-sharing introduced the engineering constraint that the interactive needs of users [were] just as important as the efficiency of the equipment. (Fernando J. Corbató)

Time-sharing was a misnomer. While it did allow the sharing of a central computer, its success derives from the ability to share other resources: data, programs, concepts. (Louis Pouzin)

ITS (*Incompatible Time-sharing System* [19], MIT, 1969)

- Pionierarbeit zur Ein-/Ausgabe und Prozessverwaltung:
  - geräteunabhängige Ausgabe auf Grafikbildschirme, virtuelle Geräte
  - netzwerktransparenter Dateizugriff (über ARPANET [20])
  - Prozesshierarchien, Kontrolle untergeordneter Prozesse
- „Seitenhieb“ auf CTSS und Multics, wegen der eingeschlagenen Richtung

Teilhaberbetrieb
Dialogstationen teilen sich einen gemeinsamen Dialogprozess

Bedienung mehrerer Benutzer durch ein Programm (bzw. einen Prozess)
- gleichartige, bekannte Vorgänge an vielen Daten-/Dialogstationen
- Wiederverwendung derselben residenten Dialogprozessinstanz

Endbenutzerdienst mit festem, definiertem Funktionsangebot
- Kassen, Bankschalter, Auskunft-/Buchungssysteme, ...
- Transaktionssysteme

Klient/Anbieter System (engl. *client/server system*)
- sehr viele Dienstnehmer (engl. *service user*)
- einen/wenige Dienstgeber (engl. *service provider*)

Antwortverhalten (weiche/feste Echtzeit, S. 5-49), Durchsatz

Multiprozessorbetrieb
Im Sinne eines homogenen Systems

SMP (engl. *symmetric multiprocessing*)

Zwei oder mehr gleiche (bzw. identische) Prozessoren, die über ein gemeinsames Verbindungssystem (meist ein Bus, aber z.B. auch die Kreuzschiene (engl. *crossbar*)) gekoppelt sind:
- jeder Prozessor hat gleichberechtigten Zugriff auf Hauptspeicher (engl. *shared-memory acccess*) und E/A-Geräte
- der Zugriff auf alle Speicherzellen ist für alle Prozessoren gleichförmig (engl. *uniform memory access*, UMA)
- die Prozessoren bilden ein homogenes System und werden von demselben Betriebssystem verwaltet

Koordination, Skalierbarkeit

Multiprozessorbetrieb (Forts.)
Im Sinne eines homogenen bzw. heterogenen Systems

SMP (engl. *shared-memory processor*)

Ein Parellelrechnersystem, bei dem sich alle Prozessoren denselben Arbeitsspeicher teilen, nicht jedoch gleichberechtigten Zugriff darauf haben müssen — sie können zusammen ein heterogenes System bilden.

asymmetrischer SMP impliziert hardwarebedingt den asymmetrischen Multiprozessorbetrieb (engl. *asymmetric multiprocessing*) durch das Betriebssystem. Programme (inkl. BS) sind ggf. prozessorgebunden.

symmetrischer SMP lässt dem Betriebssystem in Abhängigkeit von den Anwendungsprogrammen die Wahl der Multiprozessorbetriebsart (symmetrisch/asymmetrisch).

Koordination, Skalierbarkeit, Anpassbarkeit

Multiprozessorbetrieb (Forts.)
Parallelverarbeitung (engl. *parallel processing*) eines oder mehrerer Programme

N Prozessoren können...
- N verschiedene Programme
- N Prozesse verschiedener Programme
- N Prozesse desselben Programms

In einem Programm können mehrere Fäden (engl. *threads*) gleichzeitig aktiv sein.

...parallel ausführen und jeder der N Prozessoren kann...
- M verschiedene Programme
- M Prozesse verschiedener Programme
- M Fäden desselben Programms

...pseudo-parallel (nebenläufig) im Multiplexbetrieb ausführen

Sicherheit (engl. *security*)

Schutz (engl. *protection*) vor nicht authorisierten Zugriffen

Adressraumisolation: Schutz durch Eingrenzung/Segmentierung (S. 6-28)

- Zugriffsfehler führen zum Abbruch der Programmausführung
  - die CPU (bzw. MMU) fängt, das Betriebssystem bricht ab
- i.A. keine selektive Zugriffskontrolle möglich und sehr grobkörnig

Prozessen eine Befähigung (engl. *capability* [21]) zum Zugriff erteilen:

> Verschiedenen „Subjekten" unterschiedliche Zugriffsrechte (ausführen, lesen, schreiben, ändern) auf dasselbe „Objekt" (Datum, Datei, Gerät, Prozedur, Prozess) einräumen. [22, 23]

Alternative: Zugriffskontrollliste (engl. *access control list*, ACL [24])

⚠ Widerruf (engl. *revocation*), verdeckter Kanal (engl. *covered channel*)

Arbeitsspeicher und der Grad an Mehrprogrammbetrieb

Überlagerung durch programmiertes dynamisches Laden hat seine Grenze

Anzahl × Größe arbeitsspeicherresidenter Text-/Datenbereiche begrenzt die Anzahl der gleichzeitig zur Ausführung vorgehaltenen Programme

- variabler Wert, abhängig von Struktur/Organisation der Programme
- eine Variable auch in Bezug auf die Fähigkeiten der Programmierer

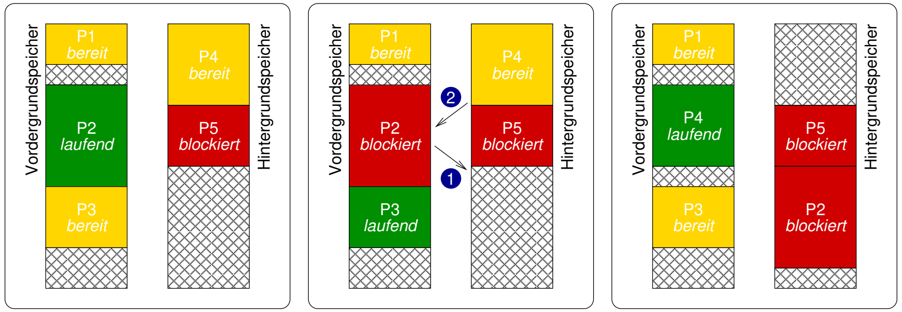
Umlagerung von solchen Bereichen gegenwärtig nicht ausführbereiter Programme (engl. *swapping*) verschiebt die Grenze nach hinten

- schafft Platz für ein oder mehrere andere (zusätzliche) Programme
- lässt mehr Programme zu, als insgesamt in den Arbeitsspeicher passt

Auslagerung von Bereichen sich in Ausführung befindlicher Programme (engl. *paging*, *segmentation*) liefert weiteren Spielraum [25]

Umlagerung von Programmen

Funktion der mittelfristigen Einplanung (engl. *medium-term scheduling*) von Prozessen



Ausgangssituation:

- P[1-3] im RAM
- P2 belegt die CPU

Umlagerung P2 & P4:

1. *swap out*
2. *swap in*

Resultat:

- P[134] im RAM
- P4 belegt die CPU

⚠ Fragmentierung, Verdichtung, Körnigkeit

Virtueller Speicher

Überbelegung des Arbeitsspeichers zulassen

Programme sind trotz (evtl.) Arbeitsspeicherknappheit ausführbar. . .

- nicht benötigte Programmteile liegen im Hintergrundspeicher
  - sie werden erst bei Bedarf (engl. *on demand*) nachgeladen
  - ggf. sind als Folge andere Programmteile vorher zu verdrängen
- Zugriff auf ausgelagerte Teile bedeutet partielle Interpretation
  - logisch bleibt das unterbrochene Programm weiter in Ausführung
  - physisch wird es jedoch im Zuge der Einlagerung (E/A) blockieren
- Aus- und Einlagerung wechseln sich mehr oder wenig intensiv ab

> Der Arbeitsspeicherbedarf eines/mehrerer scheinbar (virtuell) komplett im Arbeitsspeicher liegender und laufender/laufbereiter Programme kann die Größe des wirklichen (physikalischen) Arbeitsspeichers weit überschreiten.

⚠ Lade- und Ersetzungsstrategien

Virtueller Speicher (Forts.)

Körnigkeit (engl. *granularity*) ist fest oder variabel

Programmteile, die ein-, aus- und/oder überlagert werden können, sind Seiten (engl. *pages*) oder Segmente (engl. *segments*):

Seitenüberlagerung (engl. *paging*) Atlas [26]
: Adressraumteile fester Größe (Seiten) werden auf entsprechend große Teile (Seitenrahmen) des Haupt-/Hintergrundspeichers abgebildet

Segmentierung (engl. *segmentation*) B 5000 [27]
: Adressraumteile variabler Größe (Segmente) werden auf entsprechend große Teile des Haupt-/Hintergrundspeichers abgebildet

seitennummerierte Segmentierung (engl. *paged segmentation*) GE 635 [28]
: Kombination beider Verfahren: Segmente in Seiten untergliedern

⚠ interne Fragmentierung (Seiten), externe Fragmentierung (Segmente)

Virtueller Speicher (Forts.)

Ringschutz (engl. *ring-protected paged segmentation*)

Multics [14, 29], Maßstab in Bezug auf Adressraum-/Speicherverwaltung:

1. *jede* im System gespeicherte abrufbare Information ist direkt von einem Prozessor adressierbar und von jeder Berechnung referenzierbar
2. *jede* Referenzierung unterliegt einer durch Hardwareschutzringe implementierten mehrstufigen Zugriffskontrolle [30, 31]

„*trap on use*“, ausnahmebedingtes dynamisches Binden/Laden:

- Textsegmente werden bei Bedarf automatisch nachgeladen
- mögliche synchrone Programmunterbrechung beim Prozeduraufruf

⚠ hochkomplexes System (Hardware/Software)

Virtueller Speicher (Forts.)

Automatische Überlagerung durch partielle Interpretation

Ähnlichkeit zur Überlagerungstechnik (S. 6-35), jedoch. . .

- Seiten-/Segmentanforderungen sind nicht in den Anwendungen programmiert, stattdessen im Betriebssystem
  - die Anforderungen stellt stellvertretend ein Systemprogramm
  - Ladeanweisungen sind so vor dem Anwendungsprogramm verborgen
- Zugriffe auf ausgelagerte Seiten/Segmente werden von der Befehlssatzebene abgefangen und ans Betriebssystem weitergeleitet
  - das Anwendungsprogramm wird von CPU bzw. MMU unterbrochen
  - der Zugriff wird vom Betriebssystem partiell interpretiert
- Wiederholungen des unterbrochenen Maschinenbefehls sind in Betracht zu ziehen und vom Betriebssystem zu veranlassen

⚠ Komplexität, Determiniertheit

Allgemeinzweckbetriebssystem

(engl. *general purpose operating system*)

Betriebssystemdilemma — Software zwischen Baum und Borke

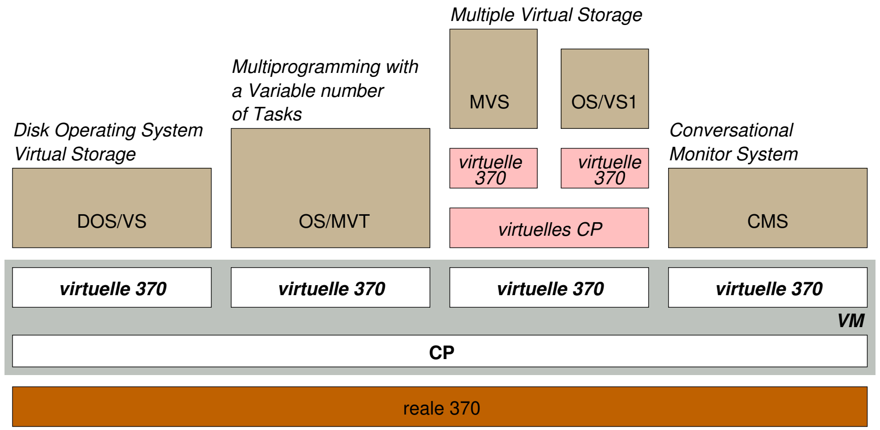
- *Clearly, the operating system design must be strongly influenced by the type of use for which the machine is intended.*
- *Unfortunately it is often the case with ‘general purpose machines’ that the type of use cannot easily be identified;*
- *a common criticism of many systems is that, in attempting to be all things to all individuals, they end up being totally satisfactory to no-one.*

(Lister, „Fundamentals of Operating Systems“ [32])

Ansätze zur Verbesserung von Benutzerzufriedenheit: Selbstvirtualisierung (virtuelle Maschinen, S. 5-57) und Konzentration auf das Wesentliche

Allgemeinzweckbetrieb durch virtuelle Maschinen

VM 370 (IBM, 1968)

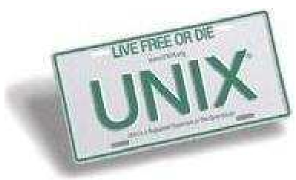


Konzentration auf das Wesentliche

UNIX [34, 35, 36] — Ein Kern von 10^4 Zeilen C und nicht 10^6 Zeilen PL/I

Multics		UNICS
Multiplexed	$\Longleftrightarrow$	Uniplexed
Information and Computing System		

☞ ITS nicht zu vergessen (S. 6-44)



„Lotta hat einen Unixtag", Astrid Lindgren [33]

Die drei Jahre alte Lotta ist die kleine Schwester der Erzählerin. Läuft am Tag vieles schief bei ihr, sagt sie „Unixtag", meint aber „Unglückstag".

Offene Systeme — Klassenkampf gegen Monopole

„Rechner aller Länder, vereinigt euch!"

Programme haben Zugriff auf Betriebsmittel eines Rechnerverbunds

- das Netzwerk ist in verschiedener Weise „transparent"
  - Netzwerktransparenz (Zugriffs- und Ortstransparenz)
  - Replikations-, Migrations-, Fehler-, Ausfall-, Leistungs-, Skalierungs-, Nebenläufigkeits-, . . . , X-transparenz
- ferne Betriebsmittel werden über lokale Repräsentanten virtualisiert

Programmverarbeitung geschieht (ein Stück weit) verteilt

- verteilte Kompilierung, verteilt ablaufendes `make(1)`, `ftp(1)`, `rsh(1)`
- Betriebssystemkerne enthalten Kommunikationsprotokolle (TCP/IP)

⚠ Heterogenität, netzwerkzentrische Betriebsmittelverwaltung

Prozessgrenzen überwindende Unterprogrammaufrufe

Diensteschicht (engl. *middleware*)

Prozedurfernaufruf (engl. *remote procedure call* [37], RPC)

- liefert die Illusion des lokalen Zugriffs auf entfernte Prozeduren

Stümpfe (engl. *stubs*) virtualisieren Aufgerufenen/Aufrufer

- Stumpf beim Aufrufer (Klient) repräsentiert Aufgerufenen (Anbieter)
  - Klientenstumpf (engl. *client stub*)
- Stumpf beim Aufgerufenen (Anbieter) repräsentiert Aufrufer (Klient)
  - Anbieterstumpf (engl. *server stub*)

(Die Art eines Stumpfes bezeichnet seine Lage, nicht wovon er abstrahiert!)

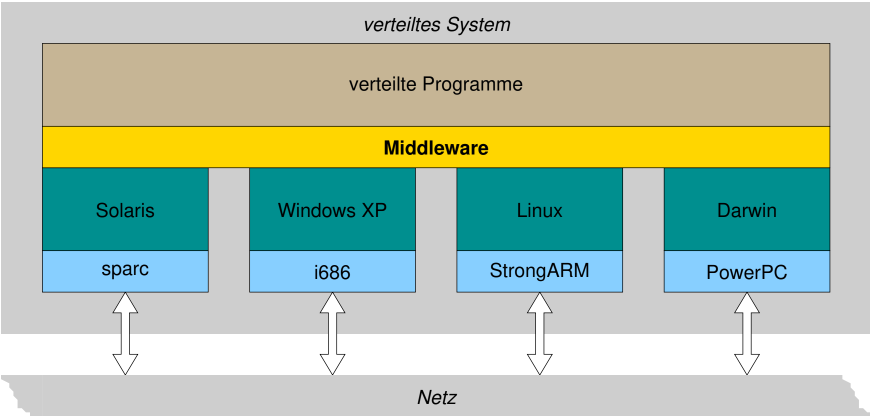
Klient/Anbieter $\mapsto$ verschiedene Fäden, Adressräume und/oder Rechner

- Interprozesskommunikation (engl. *inter-process communication*, IPC)

⚠ Parameterübergabe, Serialisierung von Datenstrukturen, Fehlermodell

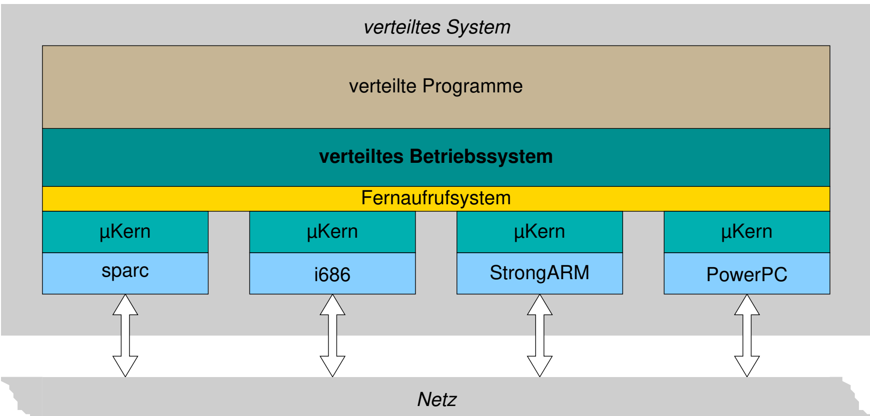
Verteiltes System

Als einzelnes System präsentierter Zusammenschluss vernetzter Rechner



Verteiltes System (Forts.)

Virtualisierung aller im Rechnerverbund verfügbaren Betriebsmittel



Hauptarten von Rechnernetzen

Klassifikation nach Abdeckungsbereiche

LAN (engl. *local area network*)
- typisch für die Vernetzung von Arbeitsplatzrechnern
- zumeist ein homogenes System: dieselben Rechner/Betriebssysteme

MAN (engl. *metropolitain area network*)
- typisch für die Vernetzung von Großrechnern und LANs
- heterogenes System: verschiedene Rechner/Betriebssysteme

WAN (engl. *wide area network*)
- typisch für die Vernetzung von LANs und WANs: Internet
- heterogenes System: verschiedene Rechner/Betriebssysteme

⚠ Skalierbarbeit, Echtzeitfähigkeit

Hauptarten von Rechnernetzen (Forts.)

Feldbus (engl. *field bus*)

CAN (engl. *control area network*)
- typisch für die Vernetzung elektronisch gesteuerter Maschinen
  - Netzknoten sind Steuergeräte (engl. *electronic control units*, ECU)
- je nach Anwendungsfall ein heterogenes System, z.B. KFZ

(Quelle: Bosch — Antriebsstrangnetz, Motorsteuerungsgerät und -anschlüsse)

Spezialzweckbetriebssystem

(engl. *special purpose operating system*)

Betriebssystem und Anwendungsprogramm(e) sind mit- bzw. ineinander verwoben, sie bilden eine (in sich geschlossene) Einheit:

im Sinne der Funktionalität

- das Betriebssystem ist „maßgeschneidert“ und „anwendungsgewahr“

im Sinne der Repräsentation

- das Betriebssystem liegt in Form einer (Quelltext-) Bibliothek vor

Eingebettetes System

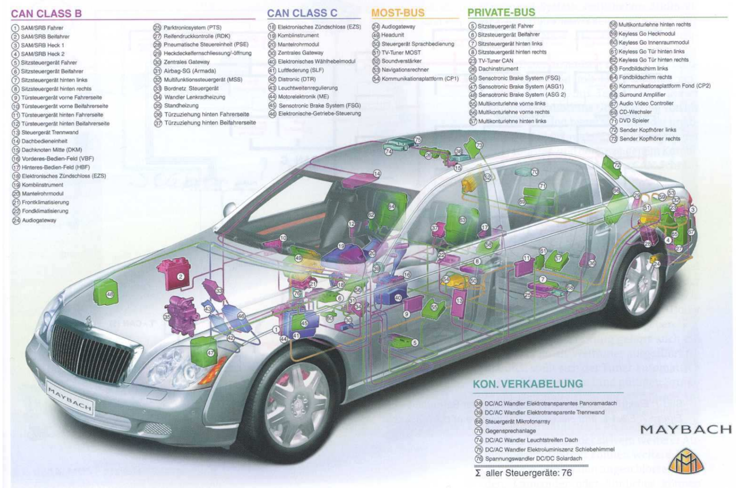
Jedes in einem Produkt versteckte Rechensystem, wobei das Produkt selbst jedoch kein Rechner ist: Kühlschrank, Mikrowelle, Kochplatte, Backofen, Esse, Wasserkocher, Waschmaschine, ...

Eingebettete Systeme

Wenn Kompromisslösungen impraktikabel sind

Eingebettete Systeme (Forts.)

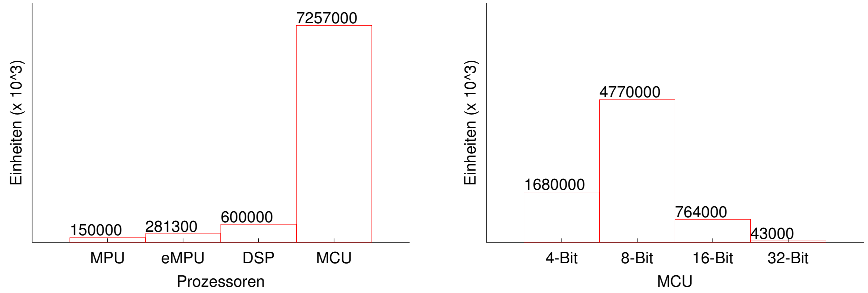
Verteiltes System auf Rädern



(Quelle: DaimlerChrysler [38])

Y2K Prozessorproduktion

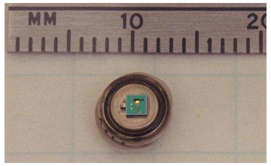
„*Where have all the processors gone?*“

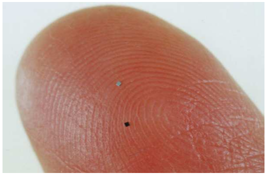


über 8 Mrd. Prozessoren [39]

1.8 %	(MPU)	Server, Desk-/Laptops, ...
98.2 %	(eMPU, DSP, MCU)	eingebettete Systeme

Drahtlose Sensornetze
„Intelligenter Staub“ (engl. *smart dust*)

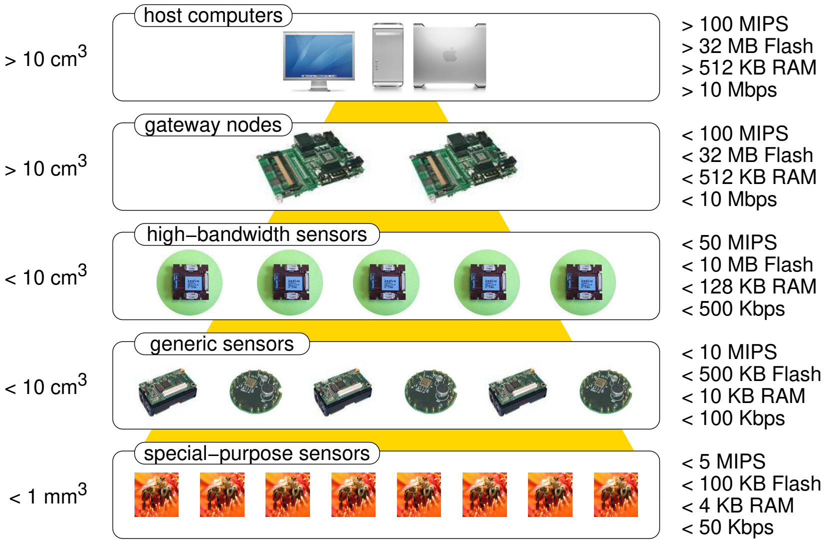




μController von Sand-/Staubkorngröße, die über Radiofrequenztechnik miteinander kommunizieren [40]:

- jeder einzelne Kleinstrechner bildet einen kubischen Sensor (*mote*)
- u. A. gedacht zur Überwachung menschenfeindlicher Umgebungen

Drahtlose Sensornetze (Forts.)
Referenzarchitektur und Technologie



Drahtlose Sensornetze (Forts.)
„Welt am Draht“

$$\left\{\begin{array}{rl} \text{[verteiltes]} & \textit{grid} \\ \text{[durchdringendes]} & \textit{pervasive} \\ \text{[allgegenwärtiges]} & \textit{ubiquitous} \end{array}\right\} \textit{computing} \Longrightarrow \textit{ambient intelligence}$$

- nahezu jedes „Gerät“ ist mit Kleinstrechnern (Sensoren, Aktoren) bestückt, die die unbegrenzte globale Vernetzung ermöglichen
- die Gerätenetze sind in einer Art und Weise in die Umgebung eingebettet, dass ihre Konnektivität jederzeit verfügbar und höchst unaufdringlich ist

Fiktion? Ja, noch ...

Einbettbare Betriebssysteme
„Kleinvieh macht Mist“

{BlueCat, HardHat} Linux, Embedix, Windows {CE, NT Embedded}, ...

Skalierbarkeit, Betriebsmittelbedarf (insb. RAM und Energie)

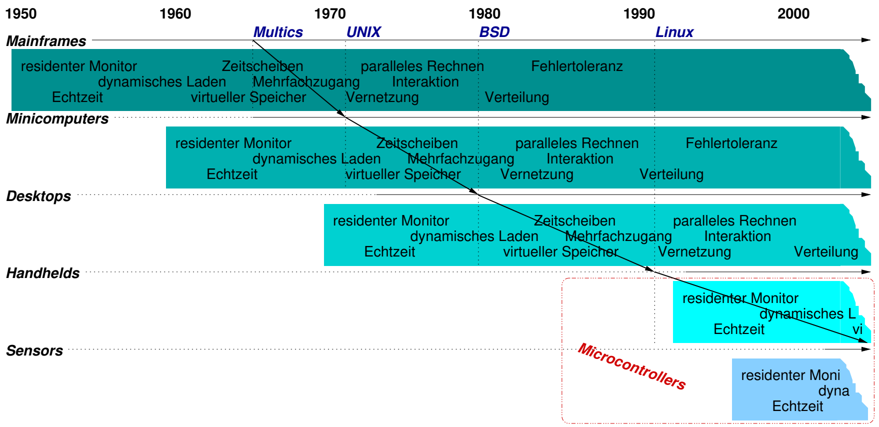
..., BOSS, C{51, 166, 251}, CMX RTOS, C-Smart/Raven, eCos, eRTOS, Embos, Ercos, Euros Plus, Hi Ross, Hynet-OS, ITRON, LynxOS, MicroX/OS-II, Nucleus, OS-9, OSE, OSEK {Flex, Plus, Turbo, time}, Precise/{MQX, RTCS}, proOSEK, pSOS, PURE, PXROS, QNX, Realos, RTMOSxx, Real Time Architect, RTA, RTOS-UH, RTX{51, 166, 251}, RTXC, Softune, SSXS RTOS, ThreadX, TinyOS, VRTX, VxWorks, ...

- der Anteil proprietärer Lösungen liegt bei über 50 % [41]
  - in vielen Fällen wird das Rad neu erfunden...

Softwaretechnik (insb. Wiederverwendbarkeit und Variantenverwaltung)

Migration von Betriebssystemkonzepten

Fortschritt in der Hardwareentwicklung kommt Betriebssystemen zu Gute



(Vorlage: Silberschatz/Galvin/Gagne [3])

Stand der Kunst — „Quietschbuntes Brimborium"

Viel Getöse für altbackene Technik?

Linux „*yet another UNIX-like operating system*", aber was soll's...

- Entwicklungsprozess und -modell sind der eigentliche „Kick"
- 70er-Jahre Technologie — ohne Multics (funktional) zu erreichen

Windows „*new technology*", wirklich?

- vor WNT entwickelte Cuttler VMS (DEC), m.a.W.: $WNT = VMS + 1$
- mit 94 % Marktführer im PC-Sektor — für 2 % des Prozessormarktes

MacOS X ein vergleichsweise echter Fortschritt

- solides UNIX (FreeBSD) auf solider Mikrokernbasis (Mach)
- Apple bringt PC-Technologie erneut voran — bei $\leq 4\,\%$ Marktanteil

Quo vadis Betriebssystem ?

Y2K — Jahr gemischter Gefühle in vielerlei Hinsicht...

Rob Pike [42]

Systems software research is irrelevant.

David Tennenhouse [39]

Over the past 40 years, computer science have addressed only about 2 % of the world's computing requirements. It's time to get physical, get real, and get out to build proactive systems."

([pro·ac·tive] Acting in advance to deal with an expected difficulty; anticipatory)

Schlüsseltechnologie „Betriebssystem"

Von elementaren Funktionssammlungen hin zu komplexen Softwaregebilden

Rechnerbetriebsarten...

- Stapelbetrieb
- Echtzeitbetrieb
- Mehrprogrammbetrieb
- Mehrzugangsbetrieb
- Netzbetrieb
- Integrationsbetrieb

Betriebssysteme sind für ein Rechensystem das „Salz in der Suppe"