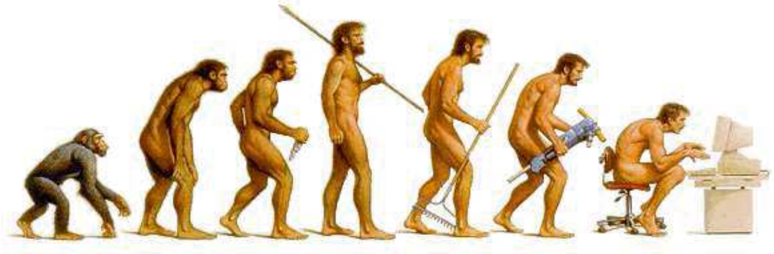
Wenn wir nicht absichtlich unsere Augen verschließen, so können wir nach unseren jetzigen Kenntnissen annähernd unsere Abstammung erkennen, und dürfen uns derselben nicht schämen. *Charles Darwin*



Systemgenerationen

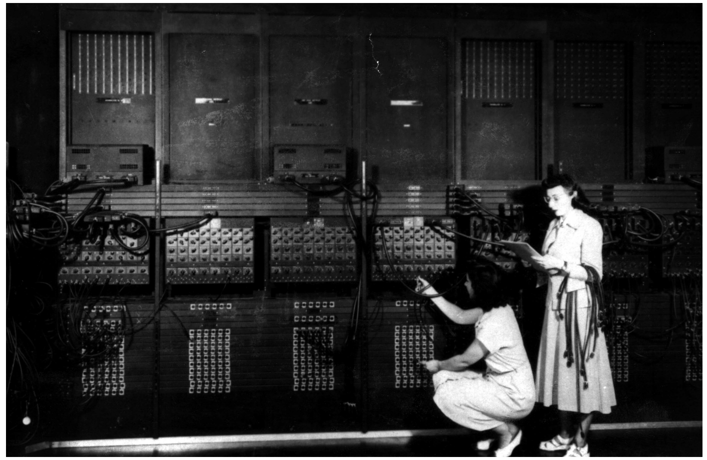
- Verfolgung der Entwicklungslinie entlang der Rechnergenerationen:[4]

Generation	Epoche	Merkmal	Betriebsart
1	1945	Röhrentechnik	Stapelbetrieb
2	1955	Transistortechnik	Dialogbetrieb
3	1965	integrierte Schaltkreise	Teilnehmerbetrieb
4	1975	(sehr) hochintegrierte Schaltkreise	Netzwerkbetrieb
5	1985	(massiv) parallele Systeme	Integrationsbetrieb

- Betriebssysteme wandelten sich in Funktion/Mächtigkeit mit den Epochen
  - ihre Komplexität in zeitlichen und räumlichen Belangen variiert(e) enorm

[4] Die Fachliteratur ist unpräzise bzgl. der exakten Daten der einzelnen Epochen und der Generationsanzahl.

ENIAC, 1945



Stapelbetrieb (1)

batch processing Betriebsart eines Rechensystems, bei der eine Aufgabe aus einer Menge von Aufgaben vollständig gestellt sein muss, bevor mit ihrer Abwicklung begonnen werden kann.

- Programme werden auf **Lochkarten** (*punch cards*) „geschrieben“ und dem Operateur (*operator*) „gestapelt“ übergeben
  - der Operateur bestückt den Rechner mit dem Stoß (*batch*) von Lochkarten
- zur „Optimierung“ des Ablaufs werden die Aufträge nach Art (z. B. Fortran[5] oder COBOL[6]) gebündelt und in entsprechender Reihenfolge ausgeführt

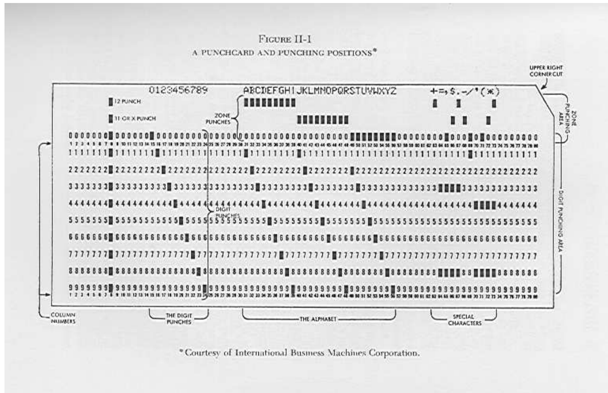
[5] *formula translator*, älteste höhere Programmiersprache, um 1954 entwickelt von John Backus (IBM).
[6] *common business oriented language*, zweitälteste höhere Programmiersprache, um 1960 entwickelt (DoD).

Stapelbetrieb (2)

batch mode eignet sich grundsätzlich zur Ausführung von Routineaufträgen

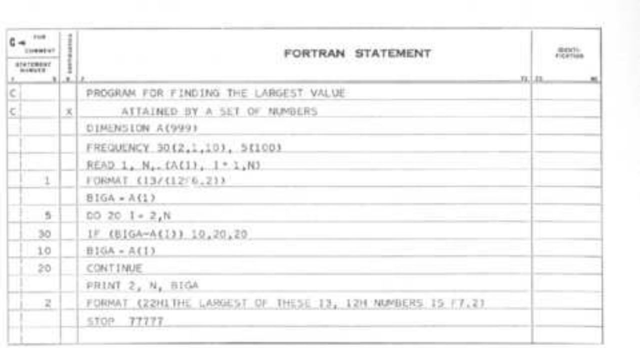
- Stapelaufträge laufen meist periodisch, in regelmäßigen Zeitabständen ab
  - ☞ täglich zur kurzfristigen Wettervorhersage
  - ☞ wöchentlich zur Fortschreibung von Marktdaten
  - ☞ monatlich zur Lohnabrechnung
- typisch ist die *interaktionslose Ausführung* einer Folge von Aufträgen (*job*)
  - sobald die Ausführung eines Jobs beginnt, wird er ohne Interaktion mit dem Benutzer (bis zum Ende bzw. Fehler) abgearbeitet
  - der Benutzer nimmt das Ausführungsergebnis vom Operateur entgegen
- bewährt zur Verarbeitung rechenintensiver Programme (*number crunching*)

Lochkarte



Lochkartenstapel

„*Don't worry, be happy*" . . . [7]



C FOR COMMENT / STATEMENT NUMBER	CONTINUATION	FORTRAN STATEMENT	IDENTIFICATION
C		PROGRAM FOR FINDING THE LARGEST VALUE	
C	X	ATTAINED BY A SET OF NUMBERS	
		DIMENSION A(999)	
		FREQUENCY 30(2,1,10), 5(100)	
		READ 1, N, (A(I), I = 1,N)	
1		FORMAT (I3/(12F6.2))	
		BIGA = A(1)	
5		DO 20 I = 2,N	
30		IF (BIGA-A(I)) 10,20,20	
10		BIGA = A(I)	
20		CONTINUE	
		PRINT 2, N, BIGA	
2		FORMAT (22H1THE LARGEST OF THESE I3, 12H NUMBERS IS F7.2)	
		STOP 77777	

[7] Jede einzelne Zeile (FORTRAN) des Formulars wurde mit Hilfe eines Stanzgeräts auf eine Lochkarte übertragen. Mehrzeilige Anweisungen mussten entsprechend gekennzeichnet werden ('X'). Als Sprungziele dienten frei wählbare Zeilennummern. Mit einer Lochkartenlehre war die visuelle Kontrolle einer korrekten und sauberen Lochung möglich.

Manuelle Bestückung des Rechners

- Operateure/Programmierer haben die volle Kontrolle über den Rechner:
  1. Programm/Daten mit Kartenlocher auf Lochkarten stanzen
  2. Programmkarten in den Kartenleser einlegen
  3. Lochkartenleseprogramm (über Konsole eingeben, dann) starten
  4. Kompilierer (selbst über Lochkarten eingespeist) starten
  5. Eingabekarten (Daten) in den Kartenleser einlegen
  6. leere Lochkarten (für die Ausgabe) in den Kartenlocher einlegen
  7. übersetztes Programm starten, stanzen der Ausgabekarten ermöglichen
  8. Ausgabekarten dem Kartenleser des Druckers übergeben
  9. Ergebnisse vom Drucker abholen
- Schwachstelle: Bedienung, Mensch

Systeme der 1. Generation (um 1950)

IBM 701, 1952
„*Defence Calculator*“

IBM 650, 1953
„*Workhorse*“

Automatische Bestückung des Rechners

- Durchsatzerhöhung durch Einschränkung/Reduzierung der manuellen Eingriffe
  1. Programmkarten in den Rechner einspeisen
  2. Lochkartenleseprogramm starten
  3. Ergebnisse vom Kartenlocher/Drucker abholen
- ein speicherresidentes Kontrollprogramm agiert als **Kommandointerpretierer**
  - Kontrollkarten regeln den Ablauf (*job control language*, JCL)
  - das „embryonale Betriebssystem“ besteht aus Lochkartenleseprogramm, Kommandointerpretierer und den E/A-Prozeduren
- Schwachstelle: langsame Peripherie, sequentielle E/A

Jobkontrollsprache — FMS [8]

```
*JOB, 4711, MATZI SCHLOEDEL
*XEQ
*FORTRAN
```

Programmkarten

```
*DATA
```

Datenkarten

```
*END
```

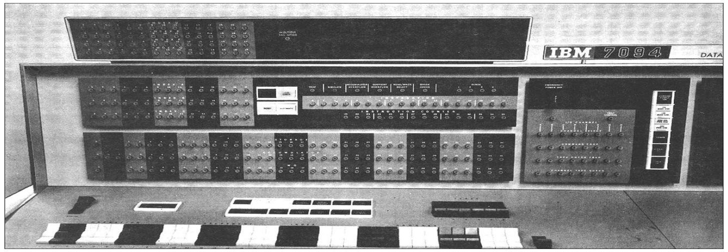
[8] Das *FORTRAN Monitoring System* war ein weitverbreitetes Betriebssystem für die IBM 709, um 1960.

Automatisiertes Hochfahren

- das Kontrollprogramm wird im Arbeitsspeicher gehalten (*resident monitor*)
  - Arbeitsspeicher ist flüchtiger (d. h. eben nicht permanenter) Speicher
  - Abschalten des Rechners führt zum Verlust des Speicherinhalts
- dauerhaft/permanent ist das Kontrollprogramm auf Lochkarten gespeichert
  - es ist bei Rechnerinbetriebnahme in den Arbeitsspeicher einzulesen
  - ein spezielles Lochkartenleseprogramm dient als **Urlader** (*bootstrap loader*)
  - das vom Urlader eingelesene Programm fährt den Rechner schließlich hoch
- der Rechner muss manuell mit dem Urlader programmiert werden ☞ ¬ROM

„Am Anfang war das Feuer“

der **Urlader**
- transferiert das (Kontroll-) Programm in den Arbeitsspeicher
- übergibt die Kontrolle an das eingelesene Programm
- wird zur Rechnerinbetriebnahme über die Konsole eingegeben
- muss von einfacher, „leicht handhabbarer“ Gestalt sein



IBM model 7151 (*control console*) 1959

Off-Line Betrieb (1)

- Einführung struktureller/organisatorischer Maßnahmen zur Arbeitsteilung

  Satellitenrechner bedienen die „langsame Peripherie“ und werden gesteuert durch ein **Spezialzweckbetriebssystem** (*special purpose operating system*)
  - *Typewriter*, Kartenleser, Kartenlocher, Drucker
  - die Ein-/Ausgabe wird über Magnetbänder transferiert

  Hauptrechner nutzen die „schnelle Peripherie“ und werden gesteuert durch ein **Allgemeinzweckbetriebssystem** (*general purpose operating system*)
  - der Hauptspeicher wird auf Basis von *Bandmaschinen* be-/entsorgt
  - Wartezeiten bei der Ein-/Ausgabe werden erheblich verkürzt

- Schwachstelle: sequentieller Bandzugriff, feste Jobfolge

Off-Line Betrieb (2)

Eingabe	Typewriter	$\Longrightarrow$	Lochkarten	Satellitenrechner
	Lochkarten	$\Longrightarrow$	Magnetband	
↓				↓
Verarbeitung	Magnetband	$\Longrightarrow$	Hauptspeicher	**Hauptrechner**
	Hauptspeicher	$\Longrightarrow$	Magnetband	
↓				↓
Ausgabe	Magnetband	$\Longrightarrow$	Lochkarten	Satellitenrechner
	Lochkarten	$\Longrightarrow$	Drucker	

Prozessautomatisierung

- Einführung von Digitalrechnern zur robusten Kontrolle „externer Prozesse“

  Prozessrechner (um 1960) ersetzen die Spezialsysteme zur automatisierten Führung, Überwachung, Sicherung und Optimierung von Prozessabläufen
  - ein direkt mit einem technischen Prozess gekoppeltes Rechensystem
  - wird auch als „Kurzwortmaschine“ bezeichnet (8- oder 12-Bit Worte)
  - verfügt über spezielle Ein-/Ausgabekanäle zur Analog-/Digitalwandlung

- *Echtzeitprogrammierung* anwendungsspezifischer Ablaufsteuerungen läuft an
  - wiederkehrende Basisfunktionen bilden erste „Echtzeitbetriebssysteme“
  - eine eigenständige Entwicklungslinie von Betriebssystemen zweigt sich ab

- *Echtzeitbetrieb* garantiert ein deterministisches Laufzeitverhalten des Systems

Systeme der 2. Generation (um 1960)

IBM 1401, 1959 IBM 7090, 1959

Abgesetzte Ein-/Ausgabe

Spool (*simultaneous peripheral operations on line*) Die Entkopplung langsamer E/A-Stöße von schnellen CPU-Stößen durch Pufferbereiche im Speicher [9]

- drei Phasen der Job-/Programmverarbeitung werden dabei unterschieden:
  - **Eingabe** (z. B. vom Kartenleser/Magnetband) erfolgt hinein in den Puffer, wenn das „langsame" Eingabegerät Bereitschaft signalisiert
  - **Verarbeitung** durch die CPU geschieht über den Puffer relativ frei von Verzögerung und vergleichsweise schnell
  - **Ausgabe** (z. B. zum Kartenlocher/Magnetband bzw. Drucker) erfolgt aus dem Puffer heraus, wenn das „langsame" Ausgabegerät frei ist
- spezielle Systemprogramme starten bzw. überwachen die Ein-/Ausgabephase

[9] Vordergrund- d. h. Hauptspeicher bzw. Hintergrundspeicher (Trommel/Platte).

Überlappte Ein-/Ausgabe

- Ein-/Ausgabegeräte verfügen über „*direct memory access*" (DMA)
  - d. h. unabhängig von der CPU arbeitende Ein-/Ausgabekanäle
- eine *asynchrone Unterbrechung* (*interrupt*) meldet Ein-/Ausgabebereitschaft
  - die Ein-/Ausgabegeräte zwingen die CPU zum Kontextwechsel:
    1. Sicherung des PC [10] und Verzweigung an eine feste Speicheradresse
    2. Unterbrechungsbehandlung
    3. Rückkehr zum unterbrochenen Programm und gesicherten PC laden
  - im (embryonalen) Betriebssystem entsteht Synchronisationsbedarf
- Schwachstelle: Lehrlauf beim Jobwechsel

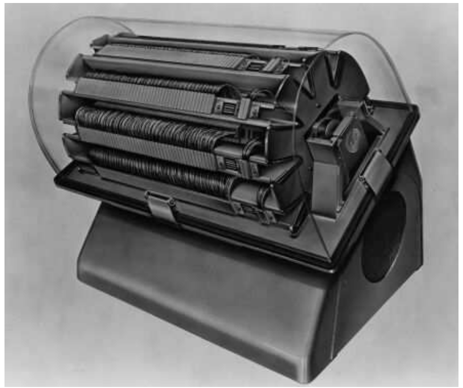
[10] PC = *program counter*.

Überlappte Jobverarbeitung

- während die CPU Jobs abarbeitet, werden weitere Jobs bereits eingelesen
  - der Vorgriff (*prefetch*) geschieht nebenläufig zur Jobausführung
  - die Festplatte (*IBM 350 Disk File*, RAMAC) [11] ersetzt Trommel/Magnetband
- das Betriebssystem besitzt *wahlfreien Zugriff* auf die zu verarbeitenden Jobs
  - die einzulesenden Jobs werden gesichtet, sortiert und bereitgestellt
  - diese **Einplanung** (*scheduling*) der Jobs unterliegt einer festen Strategie
  - die Vergabe/Zuteilung der CPU erfolgt nach einem Ablaufplan
- Schwachstelle: Hauptspeicher, Monopolisierung der CPU, Leerlauf bei E/A

[11] *Random Access Method of Accounting and Control*

Trommelspeicher ⟹ „Festplatte“



Magnettrommel der IBM 650, 1953

IBM 350 Disk File

IBM 305/650 RAMAC, 1956

Mehrprogrammbetrieb

- Jobs/Programme werden *Betriebsmittel-orientiert* zur Ausführung eingeplant[12]
  - z. B. das Programm mit $\left\{\begin{array}{l}\text{dem geringsten Speicherplatzbedarf}\\ \text{den wenigsten Ein-/Ausgabekanälen}\\ \text{der kürzesten erwarteten Laufzeit}\end{array}\right\}$ zuerst
- *Duchsatzoptimierung* (Anzahl der Jobs pro Zeiteinheit) wird praktiziert
  - Wartephasen von Jobs werden für Ausführungsphasen anderer Jobs genutzt
  - bei Ein-/Ausgabe finden (sofern möglich) Jobwechsel statt
  - die Phase von Untätigkeit der CPU (*idle phase*) wird minimiert
- Schwachstelle: Speicher, Interaktionslosigkeit (*single-stream batch monitor*)

[12] Die zur Ausführung vorgesehenen Programme sind alle im Arbeitsspeicher vorrätig.

Dynamisches Laden

overlay **1** überziehen, überlagern, belegen, bedecken. **2** die Bedeckung; Auflage, der Überzug; das Tischdeckchen; die Auflegemaske; Planpause.[13]

- Programmfragmente werden nur bei Bedarf („*on demand*“) nachgeladen
  - im Hauptspeicher bereits befindliche Teile werden überlagert
  - das Nachladen ist programmiert, d. h. in den Programmen festgelegt
- die Entscheidung zum Nachladen erfolgt zur Laufzeit: *dynamisches Laden*
  - welche Teile sich überlagern sollen, ist jedoch vorher zu definieren
  - „vorher“ bedeutet „statisch“: Programmier- oder Übersetzungszeit
- Schwachstelle: finden der (zur Laufzeit) „optimalen“ Überlagerungsstruktur

[13] 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

Dialogbetrieb

conversational mode kennzeichnet den anhaltenden Wechsel zwischen Benutzereingaben und deren Verarbeitung durch Anwendungsprogramme

- die Einplanung (*scheduling*) behandelt *interaktive Anwendungen* bevorzugt
  - E/A-intensive Anwendungsprogramme interagieren mit den Benutzern
  - die Beendigung von Ein-/Ausgabe führt zur „prompten“ Neueinplanung
    - ☞ im Falle von Ein-/Ausgabeoperationen, die sich blockierend auswirkten
  - interaktive Programme werden vergleichsweise zügig abgearbeitet
  - Benutzer erfahren allg. eine schnelle Reaktion insbesondere auf Eingaben
- Schwachstelle: Fairness (beim Mix mit interaktionslosen Jobs/Programmen)

Hintergrundbetrieb

- Programme werden interaktiv gestartet, aber „im Hintergrund“ ausgeführt

$$\text{interaktions-}\left\{\begin{array}{c}\text{behaftete}\\\text{lose}\end{array}\right\}\text{Programme laufen im}\left\{\begin{array}{c}\text{Dialog}\\\text{Stapel}\end{array}\right\}\text{-betrieb}$$

- interaktive Programme werden (weiterhin) „im Vordergrund“ abgearbeitet
  - weitere Jobs/Programme können in den Hintergrund geschickt werden
  - im Ergebnis ist es möglich, mehrere Aufgaben „gleichzeitg“ zu bearbeiten
  - zur selben Zeit sind mehrere Programme nebenläufig/parallel aktiv
- Variante: Echtzeitbetrieb im Vordergrund und Dialogbetrieb im Hintergrund

Symmetrischer Multiprozessorbetrieb (1)

SMP (*symmetric multiprocessing*) [14] Zwei oder mehr gleiche/identische und über ein Verbindungssystem gekoppelte Prozessoren:

- jeder Prozessor besitzt gleichberechtigten Zugriff auf den Hauptspeicher (*shared memory*) und auf die Ein-/Ausgabegeräte
- der Zugriff auf alle Speicherzellen ist für alle Prozessoren gleich effizient (*uniform memory access*, UMA)
- die Prozessoren bilden ein *homogenes System* und werden von demselben Betriebssystem verwaltet

☞ **Parallelverarbeitung** (*parallel processing*) <u>eines</u> oder mehrerer Programme

[14] SMP wird verschiedentlich auch als Abkürzung für „*shared-memory processor*“ verwendet. Diese Prozessorart unterstützt aber ebenso den asymmetrischen Multiprozessorbetrieb (*asymmetric multiprocessing*).

Symmetrischer Multiprozessorbetrieb (2)

Sperry Rand/UNIVAC 1108A (1965) Multiprozessorsystem, 36-Bit

- drei CPUs und zwei IOCs (*input/output controller*) [15] als Prozessorbasis
  - die IOCs werden i. A. mit Kanalprogrammen zur Ein-/Ausgabe geladen
- Synchronisation zwischen den Prozessoren auf Basis einer im CLC-Modul (*central logic and control*) implementierten *test-and-set* Anweisung
- maximal fünf Aktivitäten können parallel (d. h., gleichzeitig) stattfinden

andere:

- Burroughs B 5000 (1961, [6]), max. zwei identische (48-Bit) Prozessoren
- GE 635 (1962), erster Dualprozessor, 36-Bit

[15] An United Airlines wurde eine Sonderanfertigung mit vier CPUs und drei IOCs ausgeliefert.

Virtueller Speicher (1)

paging Atlas (1961, [7]) Adressraumteile fester Größe (Seiten) auf entsprechend große Teile (Rahmen, Kacheln) des Haupt-/Hintergrundspeichers abbilden

segmentation B 5000 (1961) Adressraumteile variabler Größe (Segmente) auf entsprechend große Teile des Haupt-/Hintergrundspeichers abbilden

paged segmentation GE 635 (1962), IBM 360/67 (1968) Kombination beider Verfahren: Segmente sind in Seiten untergliedert

Auf Basis von Adressumsetzungshardware (*memory management unit*, MMU) lädt das Betriebssystem Ausnahme-bedingt (*Trap*) die nicht im Hauptspeicher vorhandenen Teile vom Hintergrundspeicher (Trommel, Platte) nach.

Virtueller Speicher (2)

ring-protected paged segmentation GE 645 (1965, Multics [8]) [16]

- bahnbrechende Weiterentwicklung der Adressraum-/Speicherverwaltung:
  1. *jede* im System gespeicherte „*on-line*“ Information ist direkt von einem Prozessor adressierbar und auch von jeder Berechnung referenzierbar
  2. jede Referenzierung unterliegt einer Schutzring-basierten Zugriffskontrolle
- Ausnahme-bedingtes dynamisches Binden/Laden: „*trap on use*“
  - Textsegmente (Prozeduren) werden bei Bedarf automatisch nachgeladen
  - nicht eingelagerte Programmteile werden zur Laufzeit eingebunden

[16] `http://www.multicians.org`

Systeme der 3. Generation (um 1965)

Burroughs B 5000, 1961

IBM 360, 1964

Teilnehmerbetrieb

time-sharing pseudo-parallele Verarbeitung von Programmen

- Programmabläufe erhalten *Zeitscheiben* zur Ausführung zugeteilt
- ein Zeitscheibenablauf bedeutet (ggf.) einen Programmwechsel
- zyklische Zeitscheibenvergabe (*scheduling*) führt zum *CPU-Multiplexing*

multi-access mehrere Benutzer arbeiten „gleichzeitig“ am Rechner

- jede Dialogstation (Terminal) kann einen eigenen *Dialogprozess* absetzen[17]
- das Rechnersystem wird im **Mehrbenutzerbetrieb** gefahren

Schwachstelle: ein Betriebssystem für unterschiedlichste Anwendungen

[17] Im Gegensatz zum **Teilhaberbetrieb**, bei dem alle Benutzer dieselbe, residente Dialogprozessinstanz wieder verwenden. Beispiele sind Buchungs- oder (Patienten-) Aufnahmesysteme.

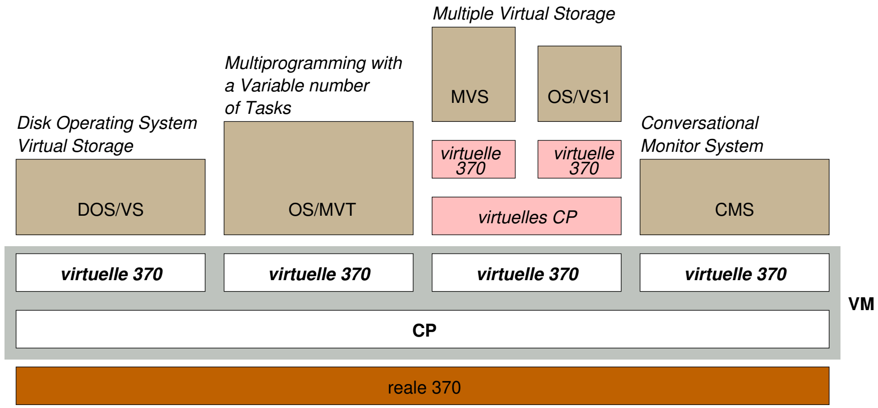
Selbstvirtualisierung

„Zauberei“: die Illusion von einer eigenen realen Maschine

- jedem einzelnen Benutzer wird der Eindruck vermittelt, er verfüge über die alleinige Kontrolle über das gesamte System mit allen seinen Betriebsmitteln
- ein Kontrollprogramm (*control program*, CP), das von der realen Maschine ausgeführt wird, virtualisiert die eigene Hardware
- der eigene reale Prozessor wird *emuliert*, indem „privilegierte Befehle“ abgefangen und vom Kontrollprogramm verarbeitet werden

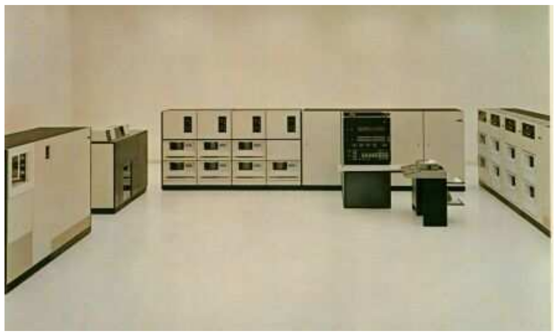
☞ **IBM 360/67** (1968) CP/CMS, später VM 370

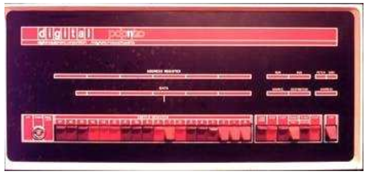
Virtuelle Maschinen



Systeme der 4. Generation (um 1970)

DEC PDP 11/20, 1970





IBM 370M145, 1970

1. Edition, 1971

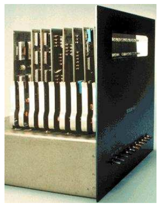
UNIX — Mehr als nur ein Wortspiel

<u>Multi</u>plexed $\Longleftrightarrow$ <u>Uni</u>plexed

<u>I</u>nformation and
<u>C</u>omputing
<u>S</u>ystem

Multics $\Longrightarrow$ UNICS
$\downarrow$
UNIX

Personal Computer



Scelbi-8H, 1974
Intel 8008
resident monitor

MITS Altair 8800, 1975
Intel 8080
CP/M (Digital Research)

Apple 1, 1976
MOS Technology 6502
resident monitor

IBM PC model 5150, 1981
Intel 8088
PC-DOS (Microsoft)
CP/M (Digital Research)

PC ein Rechner der 4. Generation mit Systemsoftware der $1\frac{1}{2}$. Generation

Massiv paralleles Rechnen

- koordinierte dynamische Einplanung von Prozessen desselben Jobs
  - alle Prozesse des parallelen Programms bilden eine „Bande“ (*gang*)
  - jede Bande führt ein Programm aus, das mehrere Datenströme verarbeitet
    ☞ *single program, multiple data* (SPMD)
  - die Bande ist einer bestimmten Anzahl von Prozessoren (fest) zugeteilt
  - Prozesse einer Bande werden als Einheit verwaltet: *gang scheduling*
- Jobs werden nebenläufig zueinander in eigenen Partitionen ausgeführt
  - massiv parallele Rechner bestehen aus hunderten/tausenden von Prozessoren
  - Banden teilen sich den Parallelrechner Zeitscheiben-basiert und verdrängend
- Schwachstelle: Skalierbarkeit, Programm-Mix

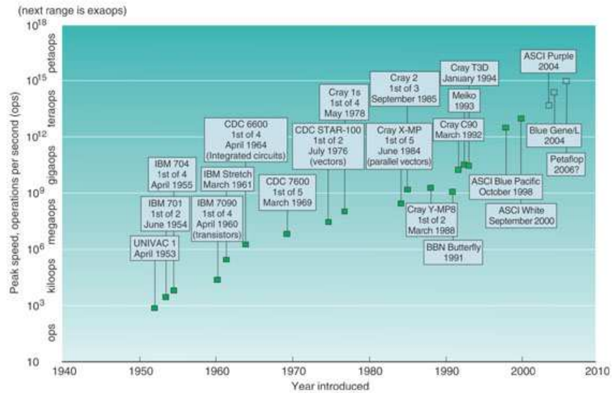
Systeme der 5. Generation (um 1990)

Cray X-MP, 1990

4640 Rechenknoten (2 Intel Pentium II Xeon mit 256 MB RAM)

ASCI Red, 1997

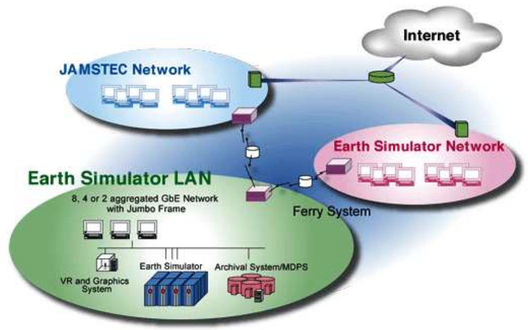
Hochleistungsrechnen[18] — *High-Performance Computing*



[18] Zeitstrahl für die Superrechner und ihre Spitzenleistung am *Lawrence Livermore National Laboratory*, CA, USA. „Purple“ ist projektiert mit 20000 Prozessoren, „BlueGene/L“ („*light*“) mit über 65000.

„Weltsimulator“ — *Earth Simulator*

- 640 Knoten à 8 Vektorprozessoren
  - insgesamt 5120 CPUs (NEC)
- UNIX-basiertes Betriebssystem
  - „SUPER-UX“ (NEC)
- Fortran90, HPF, C, C++, MPI2



☞ derzeit schnellster Rechner (1. April 2004)

☞ `http://www.top500.org`

Netzwerkbetrieb

- Benutzer/Programme haben Zugriff auf Betriebsmittel eines Rechnerverbunds
  - das Netzwerk ist in verschiedener Weise „transparent“[19] (d. h., nicht sichtbar)
  - ferne Betriebsmittel werden über lokale Repräsentanten virtualisiert
- die verteilte Verarbeitung von Programmen wird (ein Stück weit) unterstützt
  - verteilte Kompilierung, verteilt ablaufendes `make(1)`, `ftp(1)`, `rsh(1)`
  - der *Prozedur-Fernaufruf* [9][20] liefert die Illusion eines lokalen Zugriffs
  - Betriebssystemkerne enthalten Kommunikationsprotokolle (TCP/IP)
- *Middleware* zw. Anwendungsprogramme und Betriebssystem schlägt die Brücke

[19] Zugriffs-, Orts-, Nebenläufigkeits-, Replikations-, Fehler-, Migrations-, Leistungs-, Skalierungstransparenz.
[20] *remote procedure call*, RPC.

Vernetzung von Rechnersystemen (1)

LAN (*local area network*)

- typisch für die Vernetzung von Arbeitsplatzrechnern (*workstations*)
- *homogenes System* (zumeist): dieselben Rechner, dieselben Betriebssysteme

MAN (*metropolitain area network*)

- typisch für die Vernetzung von Großrechnern (*mainframes*) und LANs
- *heterogenes System*: verschiedene Rechner, verschiedene Betriebssysteme

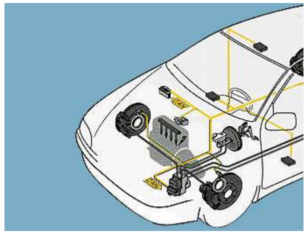
WAN (*wide area network*)

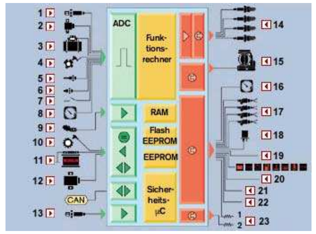
- typisch für die Vernetzung von LANs und WANs: *Internet*
- *heterogenes System*: verschiedene Rechner, verschiedene Betriebssysteme

Vernetzung von Rechnersystemen (2)

CAN (*control area network*)

- typisch für die Vernetzung von Steuergeräten (*electronic control units*, ECU)
- je nach Anwendungsfall ein *heterogenes System*, z. B. KFZ[21]:





[21] Quelle: Bosch — Antriebsstrangnetz, Motorsteuerungsgerät und -anschlüsse.

„Integrationsbetrieb“

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

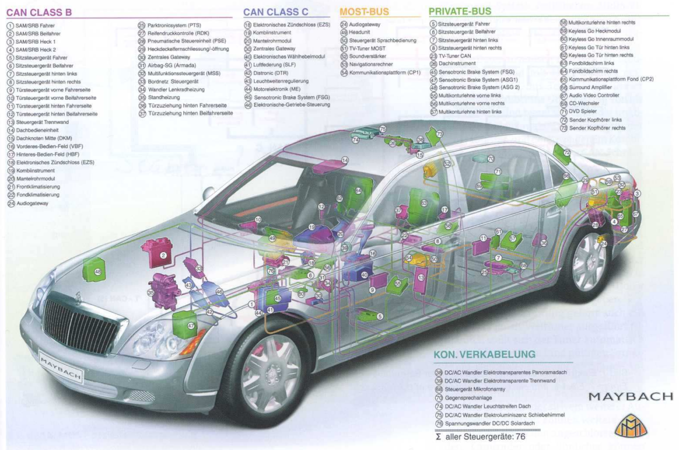
  „miteinander“ im Sinne der Funktionalität
  - das Betriebssystem ist „maßgeschneidert“ und „anwendungsgewahr“

  „ineinander“ im Sinne der Repräsentation
  - das Betriebssystem liegt in Form einer (Quelltext-) Bibliothek vor
  - die Bibliothek wird mit dem Anwendungsprogramm zusammengebunden
- wenn Kompromisslösungen impraktikabel sind ☞ eingebettetes System

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

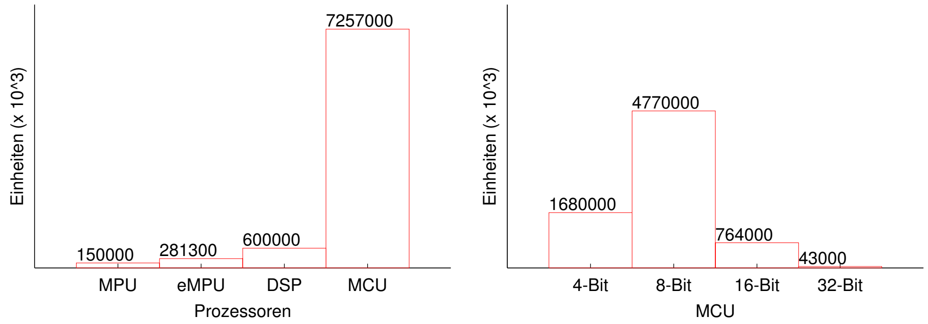
Eingebettete Systeme

Automobile — Verteilte Systeme auf Rädern



Quelle: DaimlerChrysler AG [10]

Y2K Prozessorproduktion [11]

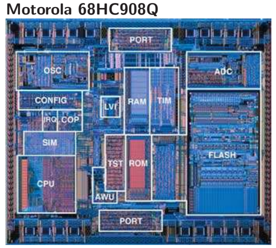


8 Mrd. CPUs ☞

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

Mikrocontroller

Motorola 68HC908Q



8-Bit Technologie

Motorola 68HC912B32

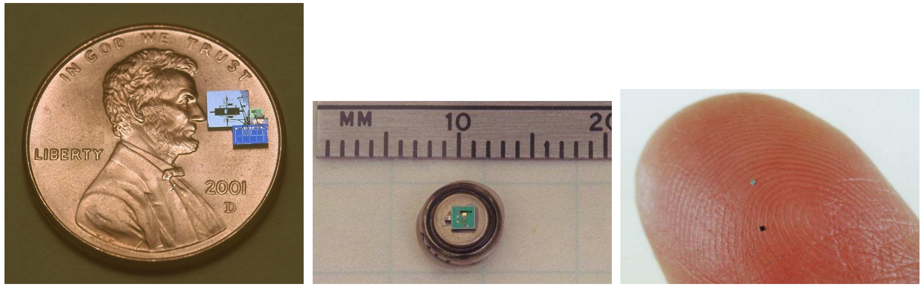
16-Bit Technologie

Motorola's MPC565 "Spanish Oak"
PowerPC™ Microprocessor

32-Bit Technologie

☞ `http://wikipedia.org/wiki/Microcontroller`

Drahtlose Sensornetze — „*Smart Dust*“



- über Radiofrequenztechnik kommunizierende μ-Controller von Sandkorngröße
  - jeder einzelne Kleinstrechner bildet einen kubischen Sensor (*mote*)
- u. A. gedacht zur Überwachung menschenfeindlicher Umgebungen (Weltraum)

Einbettbare Betriebssysteme — „Kleinvieh macht Mist“

BlueCat Linux, Embedix, HardHat Linux, Windows CE, Windows NT Embedded

☞ „Mist“ vielleicht, aber sicher kein „Kleinvieh“ ;-)

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

☞ über 50 % des Marktes sind proprietäre Lösungen für eingebettete Systeme

„Welt am Draht“

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

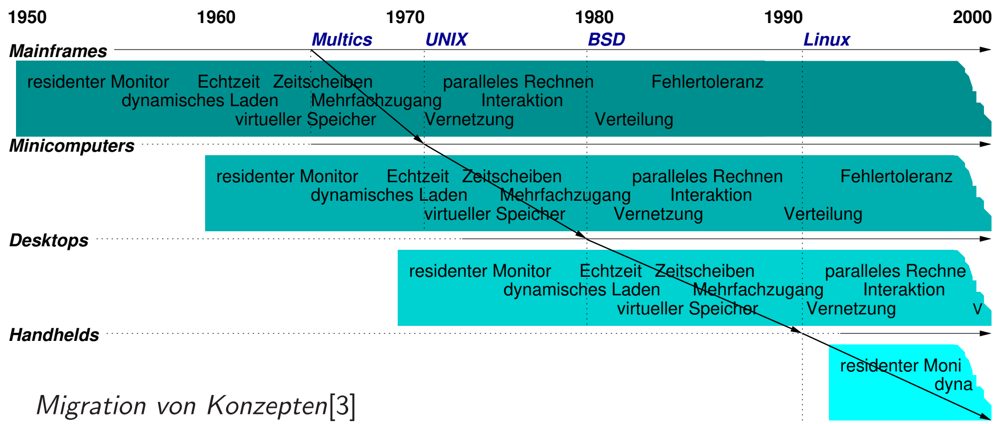
☞ Nahezu jedes „Gerät“[22] ist mit Kleinstrechnern (Sensoren, Aktoren) bestückt, die die unbegrenzte globale Vernetzung ermöglichen.

☞ Die Gerätenetze sind in einer Art und Weise in die Umgebung eingebettet, dass ihre Konnektivität jederzeit verfügbar und höchst unaufdringlich ist.

Fiktion? Ja, noch . . .

[22] Im weitesten Sinn des Wortes, so auch Kleidung und der menschliche Körper.

„Des Kaisers neue Kleider“



Migration von Konzepten[3]

Stand der Kunst

Linux „*yet another UNIX-like operating system*“, was soll’s . . .

- Entwicklungsprozess und -modell sind bedeutsam, *der* eigentliche „Kick“
- 70er-Jahre Technologie — ohne Multics erreicht zu haben

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 Panther, ein vergleichsweise echter Fortschritt

- *die* Symbiose: solides FreeBSD auf solider Mikrokernbasis (Mach)
- Apple bringt PC-Technologie abermals voran — bei $<$ 3 % Marktanteil

Betriebssysteme — *Quo Vadis*?

☞ Rob Pike, 2000: „*Systems software research is irrelevant.*“ [12]

☞ David Tennenhouse, 2000: „*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.*“ [11]

pro·ac·tive Acting in advance to deal with an expected difficulty; anticipatory

Zusammenfassung (1)

- größtenteils Systemsoftware entscheidet über die Betriebsart eines Rechners:

$$\left\{\begin{array}{l}\text{Stapel, \textit{Off-Line}, Mehrprogramm}\\ \text{Dialog, Hintergrund}\\ \text{Teilhaber, Teilnehmer, Mehrbenutzer}\\ \text{\textit{Time-Sharing/Multi-Access}}\\ \text{Multiprozessor, Netzwerk}\\ \text{Echtzeit, Integrations}\end{array}\right\}\text{-betrieb, Selbstvirtualisierung}$$

- Betriebssysteme sind für die Hardware/den Rechner das „Salz in der Suppe“

Zusammenfassung (2)

- Betriebssysteme haben sich in ihrer Geschichte großen Wandlungen unterzogen
  - von elementaren Funktionssammlungen hin zu komplexen Softwaregebilden
  - *Mainframe*-Betriebssysteme „von damals“ laufen heute auf Handhelds[23]
- ein Betriebssystem ist schlechthin *das* „Chamäleon“ aller Softwaremaschinen
  - die Anwendung bestimmt maßgeblich Erscheinungsbild und Funktionalität
  - eine für alle möglichen Einsatzbereiche gleich gute Lösung gibt es nicht
  - manche (nicht wenige) Anwendungsfelder dulden keine Kompromisse
- Betriebssysteme sind (nach wie vor — und weiterhin) Schlüsseltechnologie

[23] Die damit verbundene Konzeptmigration ist jedoch in erster Linie eine Errungenschaft der Hardwareentwicklung!