Betriebssysteme

Danilo Beuche
pure-systems GmbH

Eingebettete Systeme und Betriebssysteme (EOS)
Otto-von-Guericke-Universität Magdeburg

Wintersemester 2008/2009

Struktur

Webseite:

http://www-ivs.cs.uni-magdeburg.de/eos/lehre/WS0809/vl_bs1/

- Vorlesung Mo 9-11 Uhr c.t. (*2 SWS*)
- Übung 5 Termine im SUN-Pool G29-333, G29-336 (*2 SWS*)
- Selbststudium **4-... SWS**

Anmeldung:

http://eos.cs.uni-magdeburg.de/register/

Schein- und Prüfungsleisungen

Klausur für **ALLE** Teilnehmer

- Klausurzulassung
  - Theoretische Aufgaben zu 60% votieren
  - Praktische Aufgaben 5/6 (+1)
- Details in den Übungen erfragen

Kontakt

Dozent: Danilo Beuche

danilo.beuche@pure-systems.com

Übungsleiter: Michael Schulze, Thomas Kiebel, Stefan Sokoll

{mschulze,kiebel,sokoll}@ivs.cs.uni-magdeburg.de

Ziele

- Verständniss für
  - Struktur von Rechensystemen
  - Zusammenspiel von
    - Hard- und Software
    - Betriebssystemen und Anwendungen
    - Geräten und Treibern
- Grundlegende Fähigkeiten
  - Betriebssystemprogrammierung
  - Treiberprogrammierung

Zielgruppe:

Nicht Anwender heutiger, sondern Designer zukünftiger Betriebssysteme

Inhalte

Vorlesung

- Aufbau von Rechensystemen aus Betriebssystemsicht
- Grundlagen von Betriebssystemen
- Grundlagen maschinennaher Programmierung

Übung

Vertiefung und Verständniss des Vorlesungsstoffs durch

- Theorie (*Besprechung, Aufgaben*)
- Praxis (*Programmierung eines Minibetriebssystems*)

Literatur I

E. Ehses, L. Köhler, P. Riemer, H. Stenzel, and F. Victor.
Ein Lehrbuch mit Übungen zur Systemprogrammierung in UNIX/Linux.
Pearson Studium, 2005.

J. Nehmer and P. Sturm.
Systemsoftware: Grundlage moderner Betriebssysteme.
dpunkt Verlag GmbH, 2nd edition, 2001.

A. Silberschatz, P.B. Galvin, and G. Gagne.
Operating System Concepts.
John Wiley and Sons Inc., 6ed edition, 2001.

W. Stallings.
Betriebssysteme, Prinzipien und Umsetzung.
Pearson Studium, 4th edition, 2003.

A.S. Tanenbaum.
Moderne Betriebssysteme.
Prentice Hall, 2nd edition, 2003.

FRAGEN ?

Einführung

Quizfrage: Was ist ein Betriebssystem?

Betriebssystem

- (Fast) jedes Rechensystem besitzt ein BS
  - Definition Begriff „Betriebssystem“ nicht einheitlich
  - Betriebssysteme sind kein Selbstzweck
- Verständnis für Betriebssysteme hilft „Anwendungsphänome“ zu verstehen
  - Betriebssysteme interagieren mit Anwendungen
  - BS-Eigenschaften können Anwendungen ermöglichen/verhindern
- Betriebssysteme können klein oder umfangreich sein

Definitionen (1)

Ewert et al Ein Computer ist, wenn er genau betrachtet wird, nur eine Ansammlung von Plastik und Metall, das zur Leitung von Strom benötigt wird. Dieser ”Industriemüll“ kann somit nicht ausschließlich das sein, was wir unter einem modernen Computer verstehen, etwas, das dem Computer ”Leben“ einhaucht und ihn zu dem Werkzeug unseres Jahrhunderts macht. *Es ist das Betriebssystem, das die Kontrolle* über das Plastik und Metall (Hardware) *übernimmt und* anderen Softwareprogrammen (Excel, Word, ...) *eine standardisierte Arbeitsplattform* (Windows, Unix, OS/2) *schafft*.

- **Be’trieb·sys·tem** *Programmbündel, das die Bedienung eines Computers ermöglicht.*
- **Hansen** *Der Zweck eines Betriebssystems [besteht] in der Verteilung von Betriebsmitteln auf sich bewerbende Benutzer.*

Definitionen (2)

- **Lexikon der Informatik** *Summe derjenigen Programme, die* als residenter Teil einer EDV-Anlage *für den Betrieb der Anlage und für die Ausführung der Anwenderprogramme erforderlich ist.*
- **DIN 44300** Die *Programme* eines digitalen Rechensystems, *die* zusammen mit den Eigenschaften der Rechenanlage *die Grundlage der möglichen Betriebsarten des digitalen Rechensystems bilden und insbesondere die Abwicklung von Programmen steuern und überwachen.*
- **Silberschatz/Galvin** Ein Programm, das als *Vermittler zwischen Rechnernutzer und Rechnerhardware* fungiert. Der Sinn des Betriebssystems ist eine Umgebung bereitzustellen, in der Benutzer bequem und effizient Programme ausführen können.

Definitionen (3)

- **Tanenbaum** *Eine Softwareschicht, die alle Teile des Systems verwaltet und dem Benutzer* eine Schnittstelle oder *eine virtuelle Maschine anbietet*, die einfacher zu verstehen und zu programmieren ist [als die nackte Hardware].
- **Hofstadter** *The operating* system is itself a programm which *has the functions of shielding the bare machine from access by users (thus protecting the system), and also of insulating the programmer from the many extremely intricate and messy problems* of reading the program, calling a translator, running the translated program, directing the output to the proper channels at the proper time, and passing control to the next user.

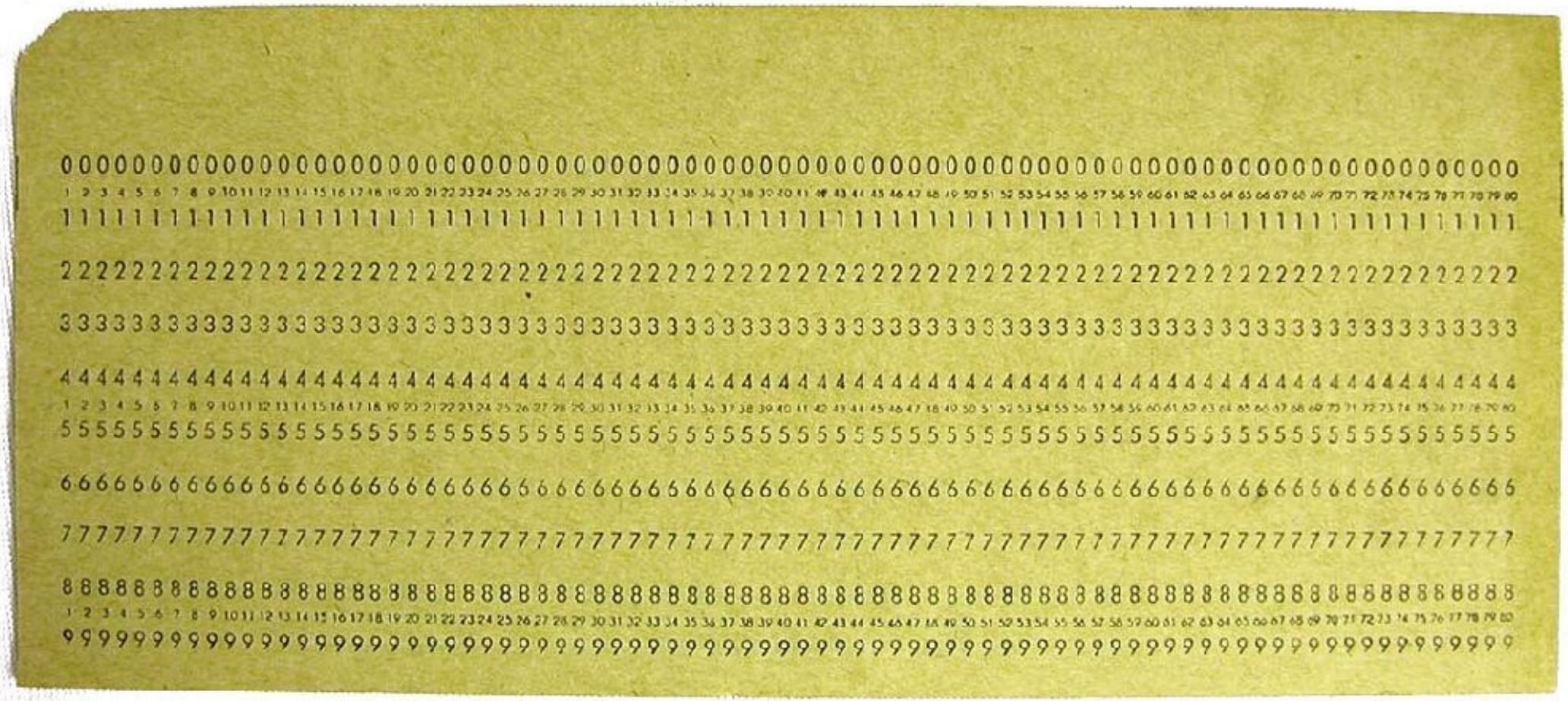
Definitionen (4)

- **Kittler** *Ein Betriebssystem kennt auf jeden Fall keinen Prozessor mehr, sondern ist neutral gegen ihn, und das war es vorher* [d. h., bevor die Schnittstelle zum real existierenden Prozessor softwaremäßig formuliert wurde] *noch nie.* Und auf diese Weise kann man eben jeden beliebigen Prozessor auf jedem beliebigen anderen emulieren, wie das schöne Wort lautet.

Zusammenfassung

- ein Betriebssystem (operating system)
  - ist eine Menge von (mehr oder weniger wohl organisierten) Programmen, die
    - Programmen, Anwendungen oder BenutzerInnen assistieren sollen
    - die Ausführung von Programmen überwachen und steuern
    - den Rechner für eine Anwendungsklasse betreiben
    - eine abstrakte Maschine implementieren
  - hat die Aufgabe, die Betriebsmittel des Rechners zu verwalten
  - d. h., Ressourcen
    - kontrolliert nutzen zu können
    - ggf. gerecht zu verteilen
  - definiert sich nicht über die Architektur, sondern über Funktionen (Dienste)

Eine kurze Geschichte der Betriebssysteme



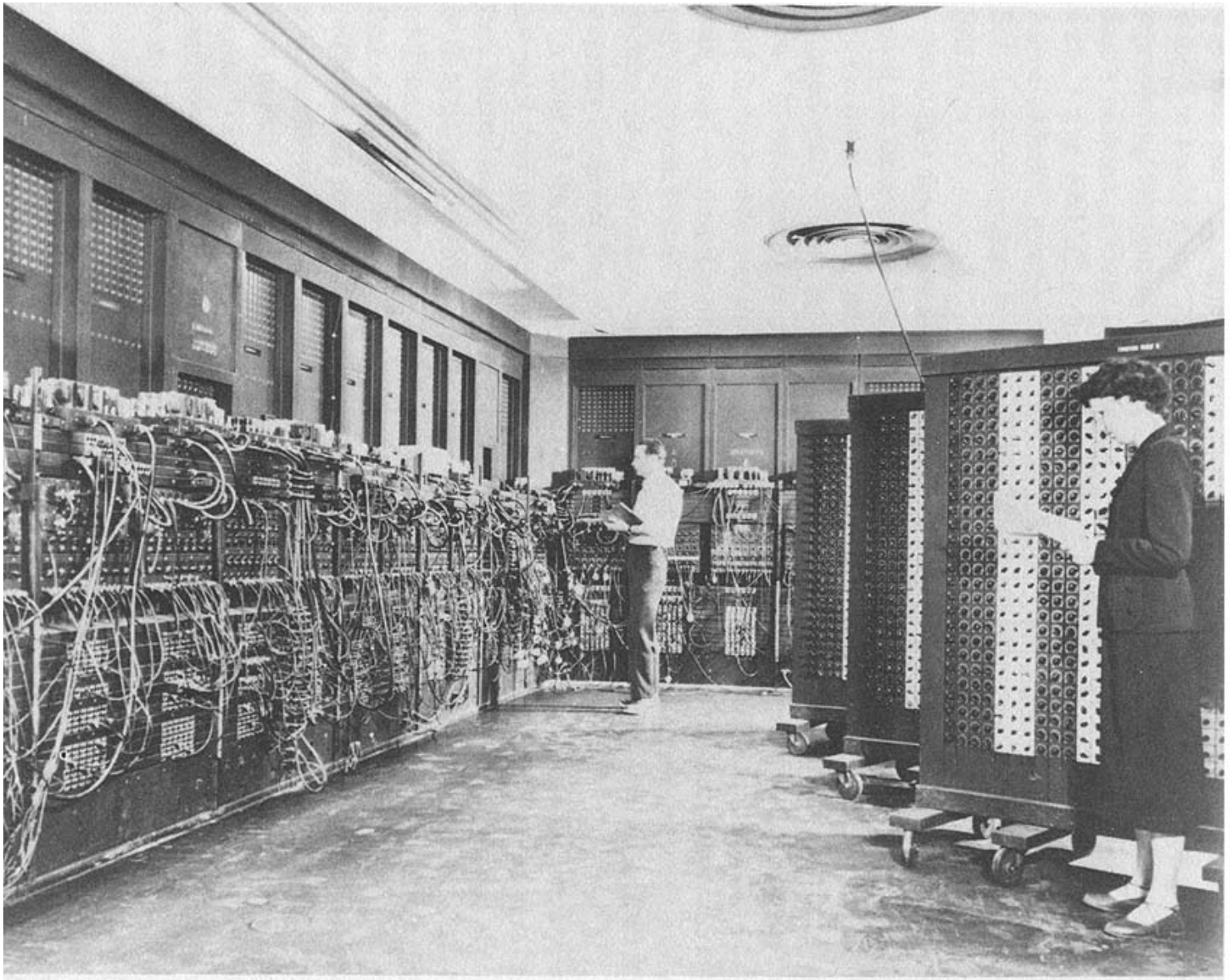
Generations

- Betriebssysteme und Rechentechnik entwickelten sich gemeinsam:

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

- Die Komplexität und Mächtigkeit nimmt mit den Epochen stark zu.

ENIAC, 1945



The ENIAC.
Smithsonian Institution Photo No. 53102

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 auch Lochkarten (*punch cards)* „geschrieben“ und dem Operator „gestapelt“ übergeben.
- der Operator bestückt den Rechner mit einem Stoß (*batch*) von Lochkarten
- Operator sortiert verschiedene Stapelabarbeitungsaufträge (**jobs**) entsprechend Vorgaben (Priorität, Ablaufoptimierung)

Stapelbetrieb (2)

batch mode ist zur Ausführung von Routineaufgaben geeignet:

- Stapelaufträge laufen meist in gleichen Zeitabständen ab
  - täglich zur Wettervorhersage
  - wöchentlich zur Auswertung von Marktdaten
  - monatlich zur Lohnabrechnung
- typisch ist die *interaktionslose Ausführung* von Aufträgen
  - sobald Ausführung des Jobs beginnt wird er ohne Eingaben vom Benutzer bis zum Ende/Abbruch ausgeführt
  - der Benutzer erhält das Ausführungsergebnis vom Operator
- bewährt für rechenintensive Prozesse (*number crunching)*

Programmieren mit Lochkarten (1)

- Operator/Programmierer hatte volle Kontrolle über 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 (f¨ur 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 1. Generation (ca. 1950)

IBM 701, 1952
„Defense Calculator“

IBM 650, 1953
„Workhorse“

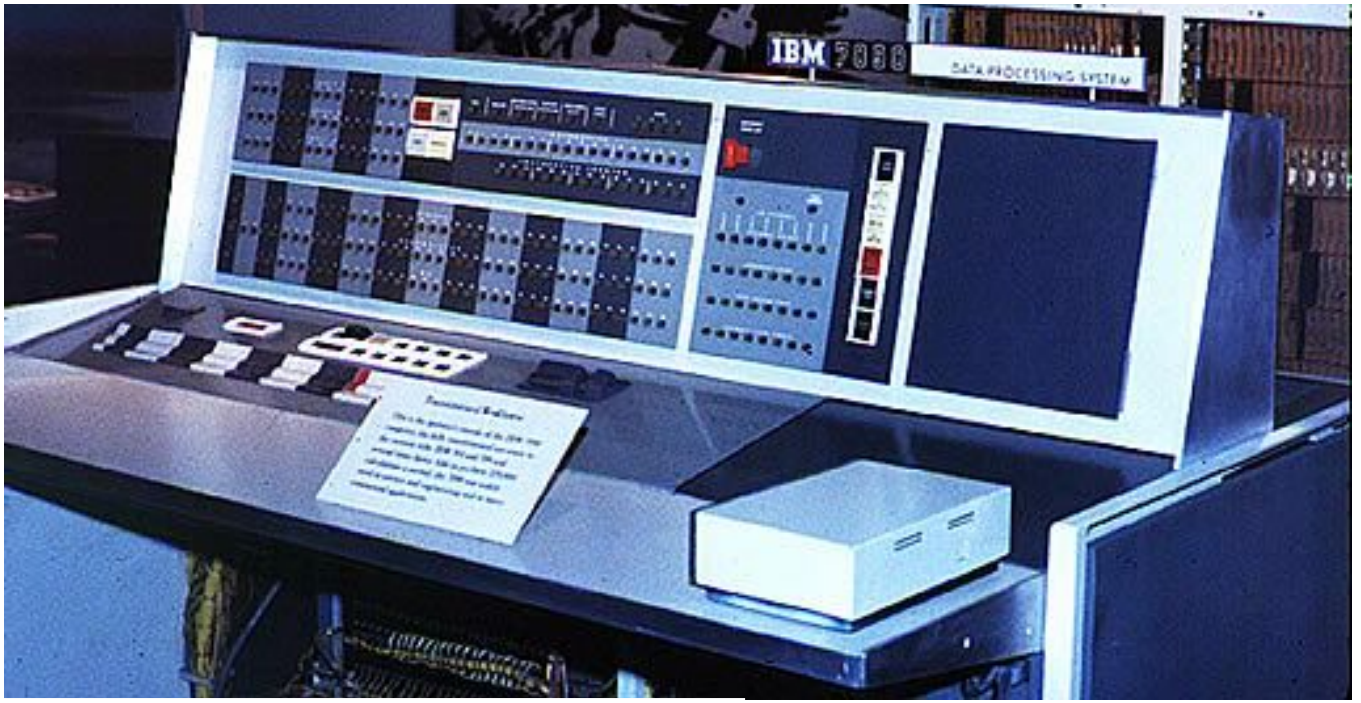
Programmieren mit Lochkarten (2)

- 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

Wie alles beginnt

Urlader (*bootstrap loader*)

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



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 2. Generation (ca. 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

- 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

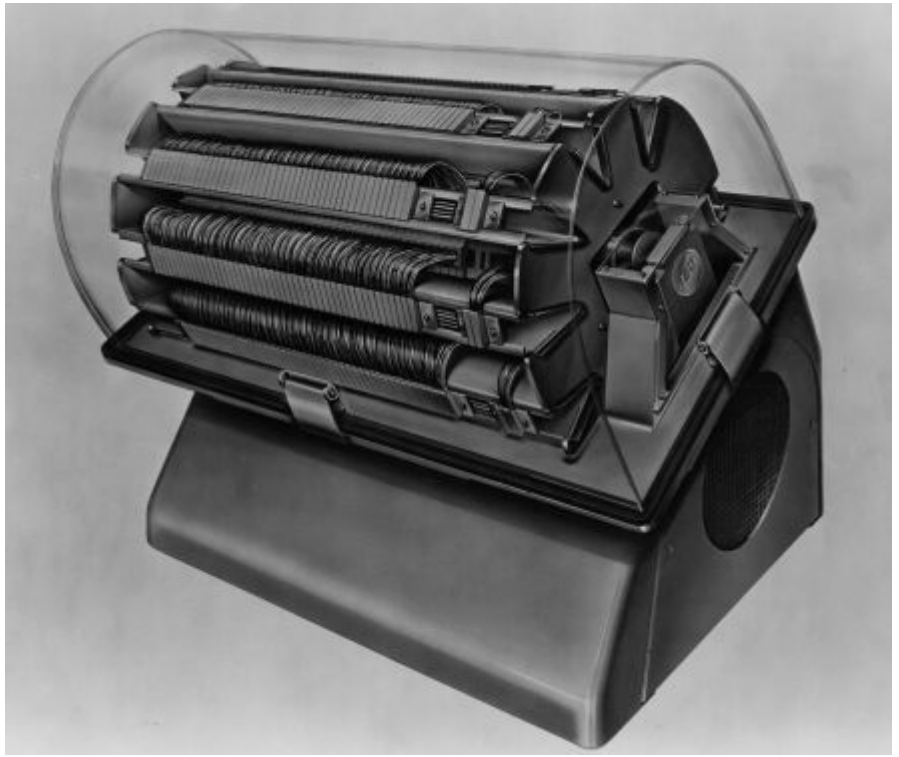
Überlappte Ein-/Ausgabe

- 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:
    - Sicherung des Instruktionszeigers (PC) und Verzweigung an eine feste Speicheradresse
    - Unterbrechungsbehandlung
    - Rückkehr zum unterbrochenen Programm und gesicherten PC laden
  - im (embryonalen) Betriebssystem entsteht Synchronisationsbedarf
- Schwachstelle: Lehrlauf beim Jobwechsel

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

Trommelspeicher vs. Festplatte



Magnettrommel, IBM 650, 1953

20.000 Ziffern / 2000 Addressen

IBM 305/650 RAMAC, 1956

Kapazität 5/10 Millionen Ziffern

Mehrprogrammbetrieb

- Jobs/Programme werden Betriebsmittel-orientiert zur Ausführung eingeplant (alle Programme sind gleichzeitig im Arbeitsspeicher)
  - z.B. zuerst das Programm
    - mit dem geringsten Speicherplatzbedarf
    - den wenigsten Ein-/Ausgabekanälen
    - der kürzesten erwarteten Laufzeit
- 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*)

Dialogbetrieb

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

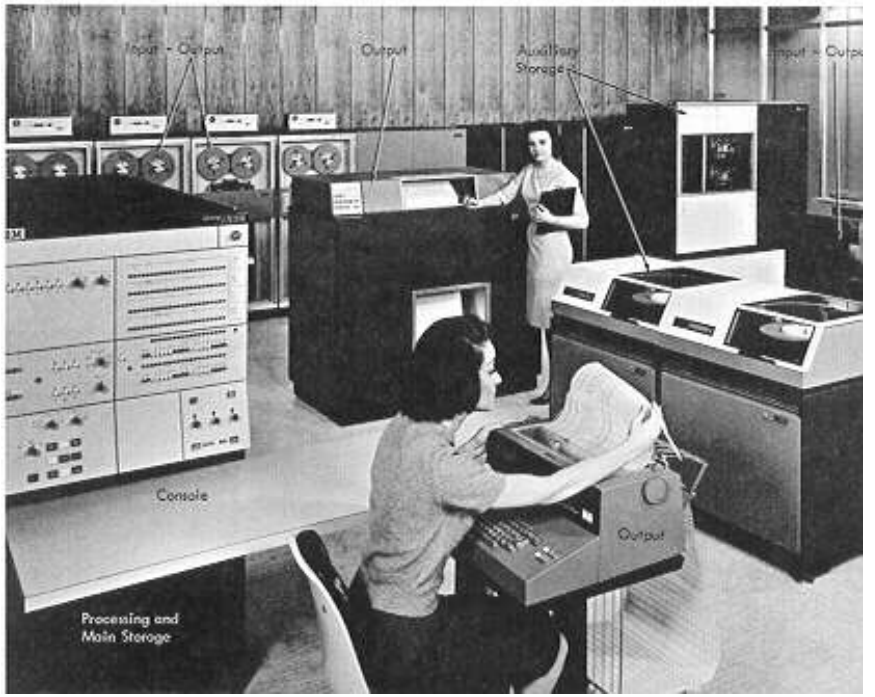
- die Einplanung (scheduling) behandelt interaktive Anwendungen bevorzugt
  - E/A-intensive Anwendungsprogramme interagieren mit den Benutzern
  - die Beendigung von Ein-/Ausgabe führt zur „prompten“ Neueinplanung im Falle von Ein-/Ausgabeoperationen, die sich blockierend auswirkten
  - interaktive Programme 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
  - interaktionsbehaftete Programme laufen im Dialogbetrieb
  - interaktionslose Programme laufen im Stapelbetrieb
- interaktive Programme werden (weiterhin) „im Vordergrund“ abgearbeitet
  - weitere Jobs/Programme können in den Hintergrund geschickt werden
  - im Ergebnis ist es möglich, mehrere Aufgaben „gleichzeitig“ zu bearbeiten
  - zur selben Zeit sind mehrere Programme nebenläufig/parallel aktiv
- Variante: Echtzeitbetrieb im Vordergrund und Dialogbetrieb im Hintergrund

Systeme der 3.Generation (um 1965)

BurroghsB5000, 1961



IBM 360, 1964

Teilnehmerbetrieb

time-sharing pseudo-parallele Verarbeitung von Programmen

- Programmabläufe erhalten Zeitscheiben zur Ausführung zugeteilt
- ein Zeitscheibenablauf bedeutet (ggf.) einen Programmwechsel

multi-access mehrere Benutzer arbeiten „gleichzeitig“ am Rechner

- zyklische Zeitscheibenvergabe (scheduling) führt zum CPU-Multiplexing
- jede Dialogstation (Terminal) kann einen eigenen Dialogprozess absetzen[1]
- das Rechnersystem wird im **Mehrbenutzerbetrieb** gefahren

- Schwachstelle: ein Betriebssystem für unterschiedlichste Anwendungen

- [1] Im Gegensatz zum „Teilhaberbetrieb“ bei dem eine Dialoginstanz für alle Benutzer verwendet wird (z.B.bei Buchungssystemen)

Systeme der 4.Generation (um 1970)

IBM 370/138, 1970

Digital PDP11/20, 1970

Unix 1.Edition, 1971

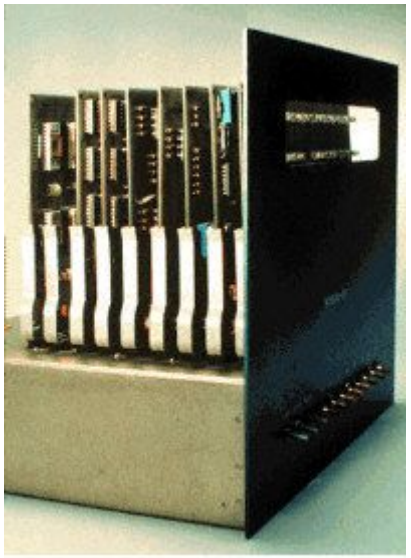
Unix – mehr als ein Wortspiel

Multiplexed Uniplexed

Information and
Computing
System

Multics Unics
Unix

Personal Computer



Scelbi 8H, 1974
Intel 8008

Apple 1, 1976
Motorola 6502

IBM PC 5150, 1981
Intel 8088

Technik der 4. Generation mit Systemsoftware der 1½. Generation