



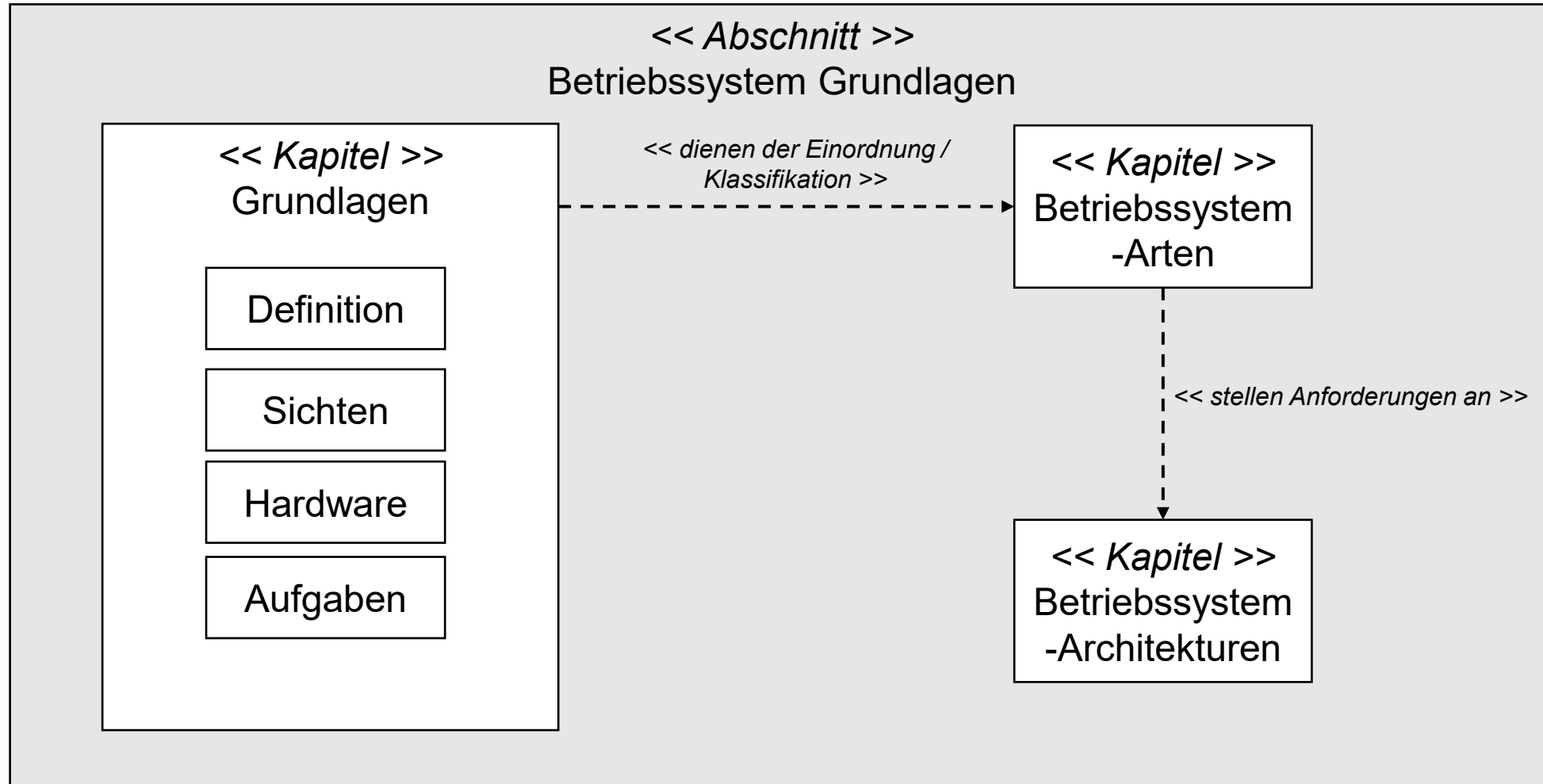
Vorlesung Betriebssysteme

Abschnitt 2 – Grundlagen der Betriebssysteme

Inhalt: Grundlagen der Betriebssysteme / Betriebssystemarten / Betriebssystem-Architekturen

M.Sc. Patrick Eberle

Symbol	Bedeutung
	Übung
	Beispiel
	Kommentar
	Definition



- Sie können den Begriff des Betriebssystems definieren
- Sie sind in der Lage, die beiden Sichten auf ein Betriebssystem zu erläutern und das Betriebssystem zwischen Hard- und Software einzuordnen
- Sie können die Operationsmodi eines Betriebssystems erläutern und kennen die Unterschiede
- Sie kennen die wesentlichen Hardware-Komponenten eines Rechners und können Zusammenhänge zwischen Hardware und Betriebssystem erklären
- Sie können die wesentlichen Aufgaben eines Betriebssystems nennen und anhand weniger Beispiele erläutern



Kapitel III

Grundlagen der Betriebssysteme

- Grundsätzlich: Viele unterschiedliche Definitionen auffindbar. Unterschiede liegen in der Sicht auf das System, sowie dem jeweiligen Anwendungskontext.

1. Betriebssystem nach DIN 44300:

„Die Programme eines digitalen Rechensystems, die zusammen mit den Eigenschaften dieser Rechenanlage die Basis der möglichen Betriebsarten des digitalen Rechensystems bilden und die insbesondere die Abwicklung von Programmen steuern und überwachen.“

2. Betriebssystem nach [MB17]:

„... Software [...], die im Kernmodus läuft [...] [und] Anwendungsprogrammierern (und natürlich Anwendungsprogrammen) saubere Abstraktionen der Betriebsmittel anstelle der unschönen Hardware [...] [zur Verfügung stellt] und andererseits diese Hardwareressourcen [...] [verwaltet].“

3. Betriebssystem nach Duden:

„System von Programmen für die Steuerung und Überwachung einer Datenverarbeitungsanlage“



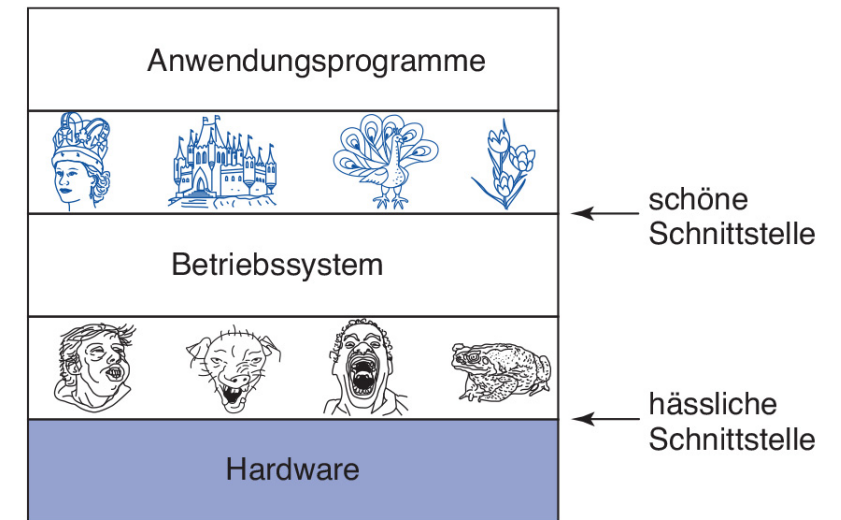
Definition 2.1: **Betriebssystem**

Software zur Überwachung und Steuerung der Hardwareressourcen eines Rechners,
sowie zur Abstraktion von Komplexität.

- Unterscheidung zweier verschiedener Sichten auf Betriebssysteme:
 - **Betriebssystem als erweiterte Maschine**
 - Fokus auf Abstraktion
 - **Betriebssystem als Ressourcenverwalter**
 - Fokus auf Betriebsmittelverwaltung

▪ Sicht 1: Betriebssystem als erweiterte Maschine

- Top-down-Sicht
- Abstraktion von Hardware und systemnahen Schnittstellen, die Detailwissen erfordern
- Anzahl der Abstraktionsschichten dabei variabel und problemangemessen: 1 ... *
- Bereitstellung einfacher Schnittstellen für Anwendungsprogramme

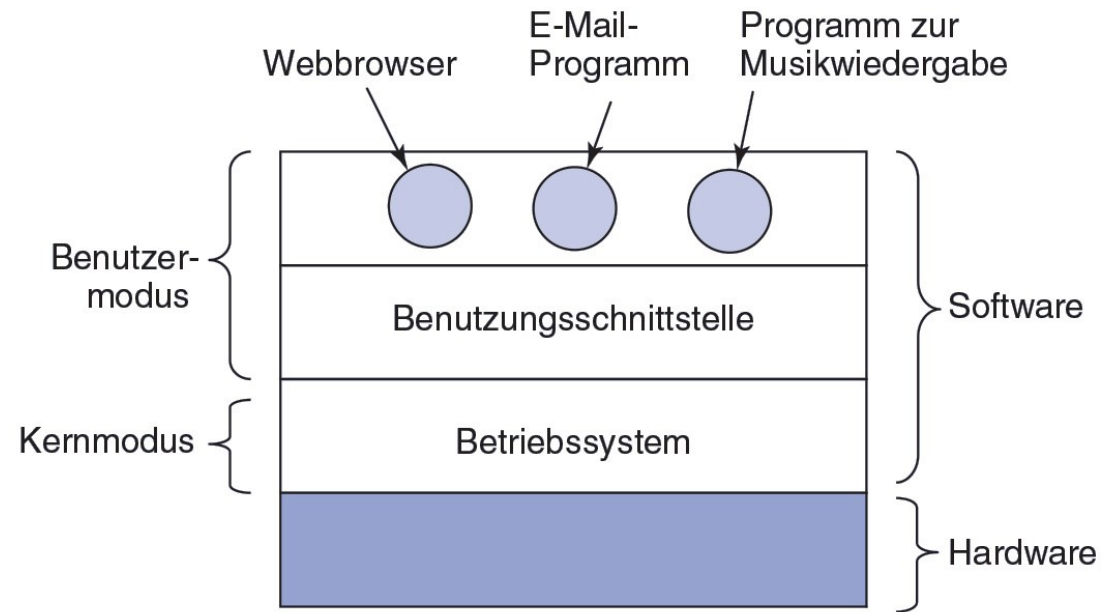


▪ Sicht 2: Betriebssystem als Ressourcenverwalter

- Bottom-up-Sicht
- Verwaltung aller Systembestandteile eines Rechners
- Geordnete und kontrollierte Zuteilung der Ressourcen an anfordernde Anwendungsprogramme, insbesondere bei konkurrierenden Zugriffen
- Dazu: Einsatz von *Multiplexing* in der zeitlichen oder räumlichen Dimension, bspw. mittels Prozess-Scheduling oder Speicherunterteilung (vgl. spätere Vorlesungs-Abschnitte)

▪ Einordnung des Betriebssystems

- Betriebssystem ordnet sich zwischen Hardware und Benutzerprogrammen ein
- Dabei: Kontrolle und Zugriffe von Hardware-Ressourcen ist komplexe Aufgabe
- Betriebssystem abstrahiert diese Komplexität vor den Benutzerprogrammen und stellt einfaches, klares Modell des Rechners zur Verfügung





Definition 2.2: **Operationsmodus**

Der Operationsmodus definiert den Ausführungskontext, unter dem ein Programm ausgeführt wird.

Der Ausführungskontext erstreckt sich dabei über Zugriffsrechte auf Hardware, sowie verfügbare Maschinenbefehle.

- I. d. R. Unterscheidung zwischen den folgenden Operationsmodi:
 - **Kernel-Modus (Kernel Mode):**
 - Exklusiv für Betriebssystem
 - Vollständiger Zugriff auf gesamte Hardware
 - Alle auf der Maschine verfügbaren Befehle ausführbar (*Befehlssatz, Instruction Set*)
 - Auch Supervisormodus genannt
 - **Benutzermodus (User Mode):**
 - Ausführmodus für Benutzerprogramme
 - Eingeschränkter Zugriff auf Hardware (keine Kontrolle über Maschine, keine I/O-Operationen)
 - Lediglich Teilmenge des Befehlssatzes ausführbar

Zusammenfassung

- Betriebssystem besitzt zwei zentrale Aufgaben, welche aus den folgenden Sichten hervorgehen:
 - Sicht 1 - Erweiterte Maschine: Aufgabe der Abstraktion komplexer Hardware und systemnaher Schnittstellen
 - Sicht 2 - Ressourcenverwaltung: Verwaltung, Zuordnung und Kontrolle von Ressourcen

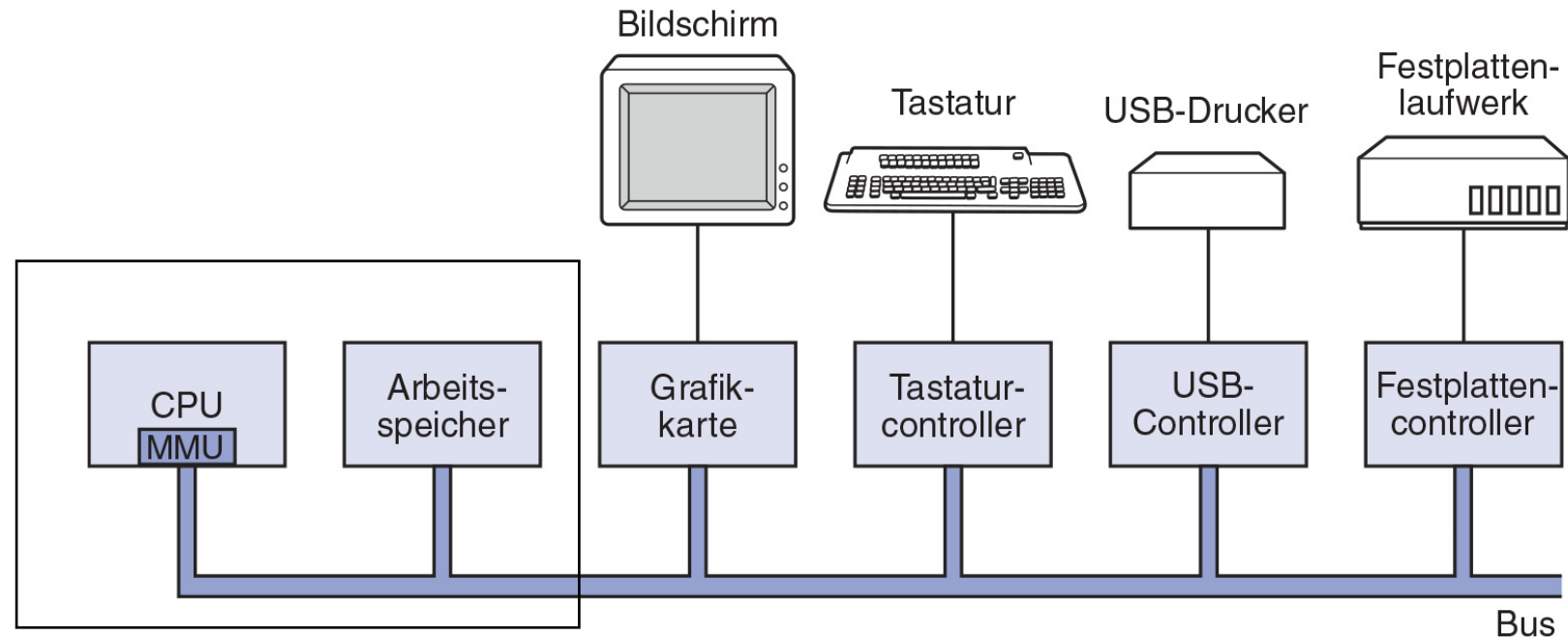
- I. A. Unterscheidung zwischen zwei Operationsmodi:
 - Kernel Mode: Exklusiv für Betriebssystem, vollständiger Befehlssatz verfügbar, vollständiger HW-Zugriff
 - User Mode: Für Benutzerprogramme, Teilmenge des Befehlssatzes verfügbar, eingeschränkter HW-Zugriff

Motivation

- Betriebssystem eng mit Hardware verknüpft
- Gemäß [Sicht 2](#) besteht zentrale Aufgabe des Betriebssystems in der Verwaltung der Hardware-Ressourcen
 - Grundlagenwissen über Hardwarekomponenten vonnöten
- Erst durch Erfüllung von Sicht 2 (Ressourcenverwaltung) wird Sicht 1 (Abstraktion für Anwenderprogramme) ermöglicht:

Abstraktion \Rightarrow Ressourcenverwaltung

- Überblick über Hardwarekomponenten



Nähere Betrachtung im Rahmen der Vorlesung, da für das Verständnis von Betriebssystemen von zentraler Bedeutung

- Prozessor (Central Processing Unit)
 - In den letzten Jahren: Übergang von Einkern-Prozessoren zur Mehrkern-Prozessoren
 - Ursache für diese Entwicklung: Sogenannte „Frequency Wall“¹ bzw. „Power Wall“¹:
 - **Frequency Wall:** Erhöhung der Taktfrequenz geht mit Erhöhung der Betriebsspannung einher.
Dabei: Überproportionaler Anstieg des Stromverbrauchs, sehr hohe Verlustleistung.
 - **Power Wall:** Die durch Frequency Wall entstehende Wärme lässt sich nicht mit vertretbarem Aufwand abführen.
 - Im Folgenden: Betrachtung von Einkern- und Mehrkern-Prozessoren

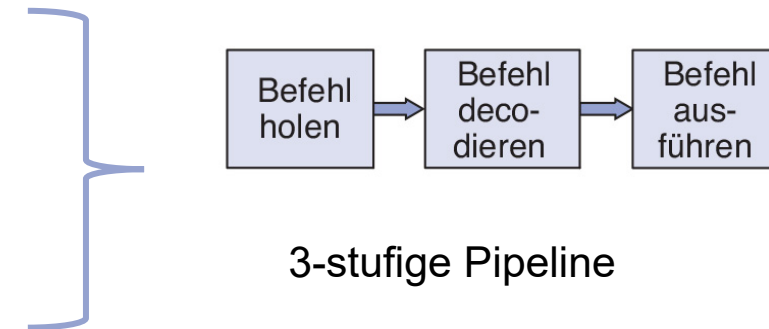
1: Vgl. hierzu [MS12, Seite 10]

Bestandteile eines Prozessors

- Befehlssatz
- Register (Allgemein)
- Register (Speziell):
 - Befehlszähler (Program Counter)
 - Stackpointer (Kellerregister / Stapelregister)
 - Statusregister / Programmstatuswort (Program Status Word)
- Cache

▪ Arbeitsweise eines Prozessors

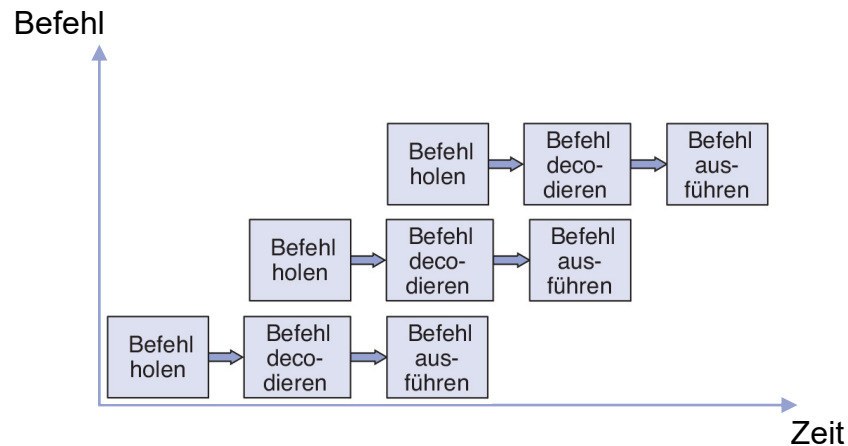
1. Befehl aus dem Speicher laden (Befehl holen)
2. Befehl decodieren und notwendige Operanden holen
3. Befehl ausführen



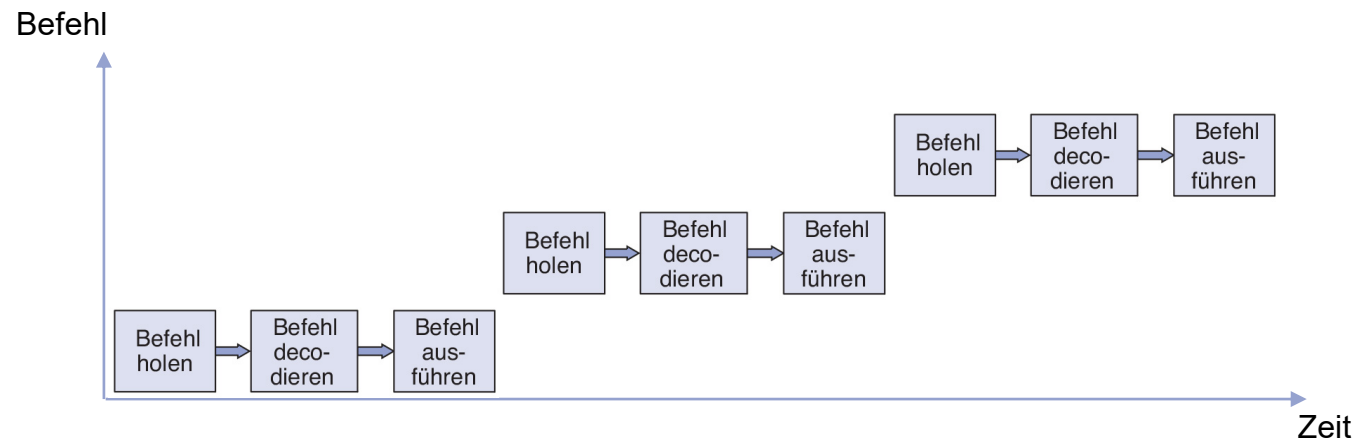
- Für jeden Befehl, so lange bis das Programm beendet ist.
- Dabei: Laden von Befehlen / Daten aus dem Speicher erheblich zeitaufwändiger, als Ausführung eines Befehls → Verwendung von Registern

■ Prozessor-Pipelining

- Zur Effizienzsteigerung: Parallele Abarbeitung der Befehle mittels Pipelining
- CPU verfügt dabei über mehrere getrennte Hol-, Dekodier- und Ausführungseinheiten



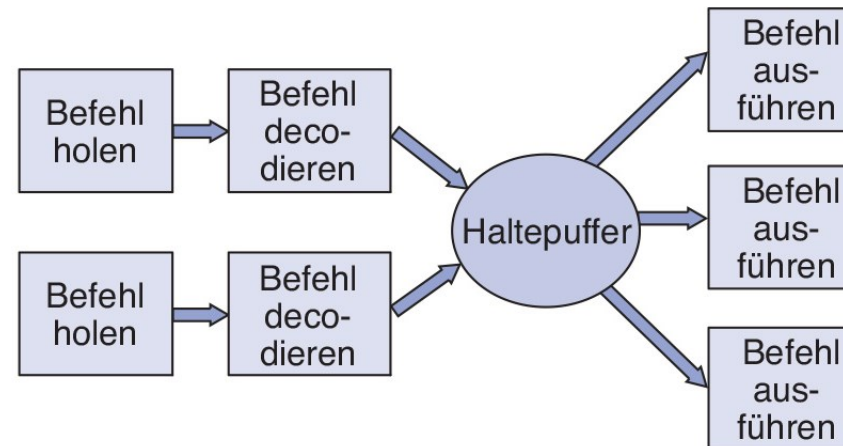
Befehlsverarbeitung mit Pipelining



Befehlsverarbeitung ohne Pipelining

- Superskalare CPU (I)
 - Fortschrittlichere Variante zu Pipeline-Architektur
 - Mehrere Ausführungseinheiten: 1 Festkommaarithmetik, 1 Gleitkommaarithmetik, 1 Boolesche Operationen
 - 1 ... * Befehle werden pro Takt geholt, dekodiert und in Puffer abgelegt
 - Jede Ausführungseinheit prüft nach Abschluss ihrer letzten Operation, ob weitere Operationen in Puffer abgelegt sind, holt diese und verarbeitet sie gegebenenfalls

- Superskalare CPU (II)



▪ Multithreading / Multithreadingchips

- Synonym: **Hyperthreading** (Bezeichnung von Intel)
- Thread: Leichtgewichtiger Prozess, welcher innerhalb eines Prozesses ausgeführt wird
- Konzept:
 - Prozessor kann in einem Zustand zwei unterschiedliche Threads verwalten und zwischen diesen umschalten
 - Umschaltzeit zwischen Threads liegt im Nanosekunden-Bereich
 - Thread-Umschaltung innerhalb eines Prozesses
- Gewinn:
 - Wird Thread 1 durch I/O-Zugriff blockiert, kann währenddessen in Thread 2 umgeschaltet und weitergearbeitet werden
 - Beim Hyperthreading: zwar keine echte Parallelität, aber Möglichkeit des asynchronen Wartens auf I/O, beim echten Multithreading: reale Parallelität in der Programmverarbeitung

Auswirkungen des Hyperthreading auf das Betriebssystem

- Jeder Thread erscheint dem Betriebssystem wie eigenständiger Prozessor
- Daraus folgt: $Prozessoranzahl_{Logisch} = Prozessoranzahl_{Physikalisch} * Threadanzahl$
- I.d.R.: Threadanzahl = 2

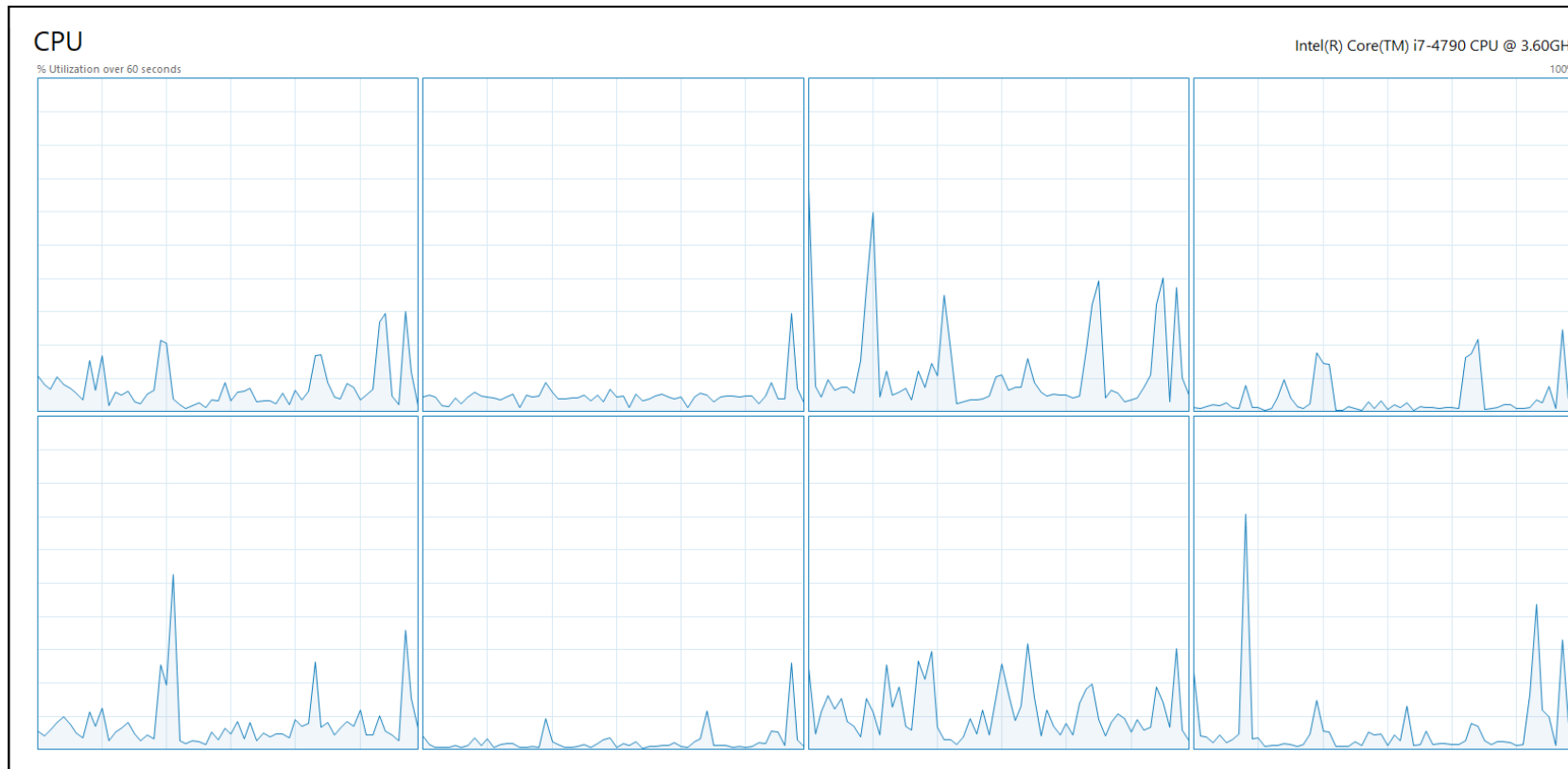


Beispiel: Intel Core i7-4790 – Logische vs. Physikalische Prozessoren

Anzahl der physikalischen Prozessorkerne: 4

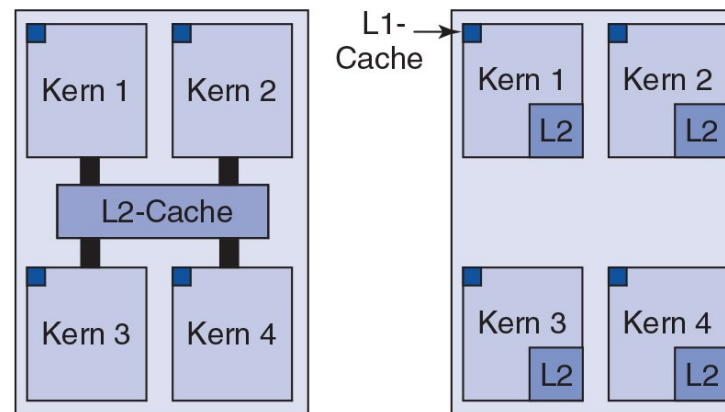
Anzahl der logischen Prozessoren gegenüber dem Betriebssystem: 8

(Fortsetzung Beispiel)



▪ Mehrkernchips

- Im Gegensatz zu Multithreadingchips: Tatsächliche Existenz mehrerer Prozessorkerne
Im Allgemeinen: 4, 8 oder 16 Kerne, auf Serversystemen tendenziell mehr
- Prozessorkerne isoliert voneinander, parallele Verarbeitung
- Unterschiedliche Ausgestaltung hinsichtlich CPU-Cache:



Speicher

- Grundsätzlich: Unterscheidung zwischen vier Arten von Speicher:
 - CPU-Register
 - CPU-Cache (L1-Cache, L2-Cache, (L3-Cache))
 - Arbeitsspeicher (RAM, Random Access Memory)
 - Magnetplatte / Festplatte bzw. heutzutage Solid State Disk (SSD)
- Anforderungen an Speicher (idealistisch):
 - Performanter Zugriff (bestenfalls schneller als CPU-Befehlsausführung → sonst Verzögerung)
 - Großes Speichervolumen
 - Günstige Preise

Dabei gilt: Tradeoff zwischen den drei Größen

(Schneller Zugriff → Hoher Preis, Großer Speicher → verhältnismäßig langsamer Zugriff, ...)

▪ Speicherhierarchie

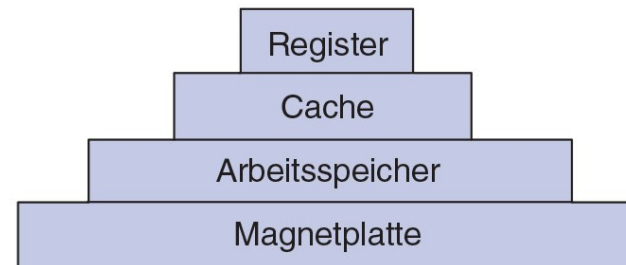
Typische Zugriffszeit

1 ns

2 ns

10 ns

10 ms



Register

Cache

Arbeitsspeicher

Magnetplatte

Typische Kapazität

<1 KB

4 MB

1-8 GB

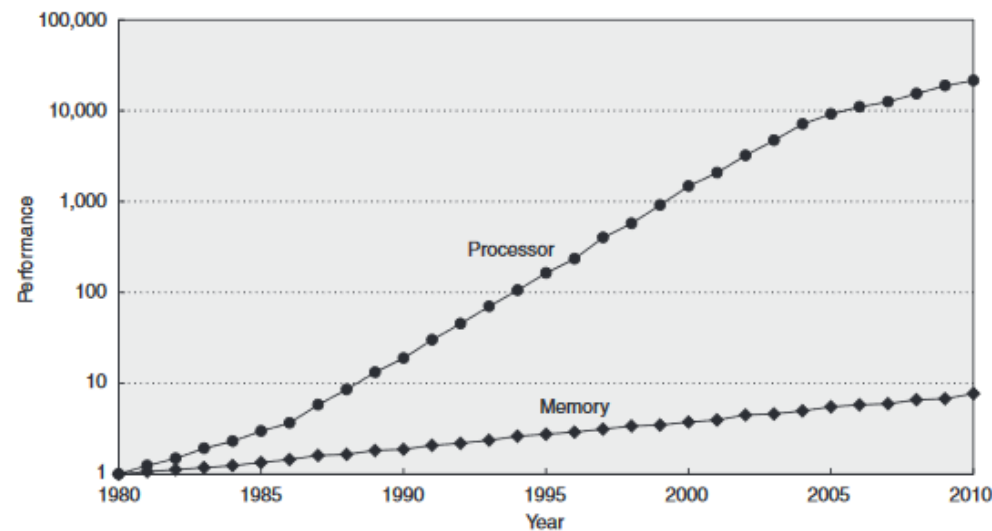
1-4 TB

■ Memory Wall

- Begriff, der das wachsende Ungleichgewicht zwischen CPU- und Speicher-Geschwindigkeit beschreibt
- Grund für das Ungleichgewicht:

Limitierte Kommunikations-Bandbreite über Chipgrenzen hinweg (sog. Bandwidth Wall)

- Wird verschärft bei Von-Neumann-Architektur: Daten und Befehle teilen sich Kommunikationsbus



Quelle:

John L. Hennessy, David A. Patterson:
Computer Architecture – A Quantitative Approach,
Fourth Edition, Morgan Kaufmann, 1990

Auswirkungen des Speichers auf das Betriebssystem

- Speicherhierarchie muss beim Datenzugriff berücksichtigt werden
- Optimierung der Speicherzugriffe ist Aufgabe des Betriebssystems, Bsp.: Prefetching
- Speicher muss verwaltet werden: Aktualisierung und Invalidierung sind Aufgaben des Betriebssystems
- Speicherverwaltung erfolgt durch Algorithmen, welche durch Betriebssystem ausimplementiert werden

▪ **Wesentliche Aufgaben des Betriebssystems:**

- Abstraktion
- Verwaltung von Hardwareressourcen
 - Prozessor(en)
 - Hauptspeicher
 - Festspeicher
 - Peripheriegeräte
 - Rechenzeit
- Prozessverwaltung
- Speicherverwaltung
- Dateisystemverwaltung
- Sicherheit und Rechteverwaltung
- Bereitstellung einer Benutzer-Schnittstelle (Shell)

- Abstraktion
 - Verbergen der komplexen Hardware-Ansteuerung vor dem Anwender
 - Bereitstellung einfacher Programmierschnittstellen (APIs)
 - Abstraktion des Maschinenbegriffs nach Coy (siehe nächste Folie)

- Abstraktion

Der Maschinenbegriff nach Coy

Reale Maschine	=	Zentraleinheit (CPU) + Geräte (Hardware)
Abstrakte Maschine	=	Reale Maschine + Betriebssystem
Benutzermaschine	=	Abstrakte Maschine + Anwendungsprogramm

$\text{Reale Maschine} \subset \text{Abstrakte Maschine} \subset \text{Benutzermaschine}$

- Verwaltung von Hardwareressourcen
 - Ansteuerung der Geräte
 - Interrupt-Steuerung
 - Fehlererkennung und -Behebung

■ Prozessverwaltung

- Zunächst:
 - Prozesse stellen eine der wichtigsten Abstraktionen im Betriebssystem dar
 - Prozesse sind Grundlage für parallele Verarbeitung von Aufgaben
- Ermöglichung des sogenannten Prozess-Scheduling, mit dem sich mehrere Prozesse eine physikalische CPU teilen
- Überwachung der Prozesse während ihrer Ausführung
- Erzeugung und Beendigung von Prozessen
- Kommunikation und Synchronisation von Prozessen
- Verwaltung von Prozess-Ressourcen und Berechtigungen

- Speicherverwaltung
 - Abstraktion des Speicherzugriffs
 - Abstraktion der Speicherhierarchie
 - Effiziente Speicherverwaltung
 - Bereitstellung und Freigabe von Adressräumen für Prozesse
 - Speicherzugriffs-Sicherheit

▪ Dateisystemverwaltung

- Abstraktion und Verwaltung des Zugriffs auf persistente Speicher
- Erstellen, Lesen, Modifizieren und Löschen von Dateien und Ordnern
- Caching-Mechanismen für effizienten Zugriff
- Zugriffssicherheit, insbesondere bei Zugriffen durch mehrere Prozesse
- Verwaltung der Berechtigungen
- Verwaltung physikalischer / logischer Laufwerke (Massenspeicher-Verbund: RAID)
- Weitere Werkzeuge wie beispielsweise Defragmentierung, Partitionierung usw.

- Sicherheit und Rechteverwaltung
 - Authentifizierung und Autorisierung
 - Verwaltung des Zugriffs auf Ressourcen
 - Unterstützung kryptografischer Funktionen (Public-Key-Kryptografie, Digitale Signaturen, ...)
 - Bereitstellung von Abwehrmechanismen gegen Malware
 - Gewährleistung der Ausfallsicherheit
 - Backup und Recovery

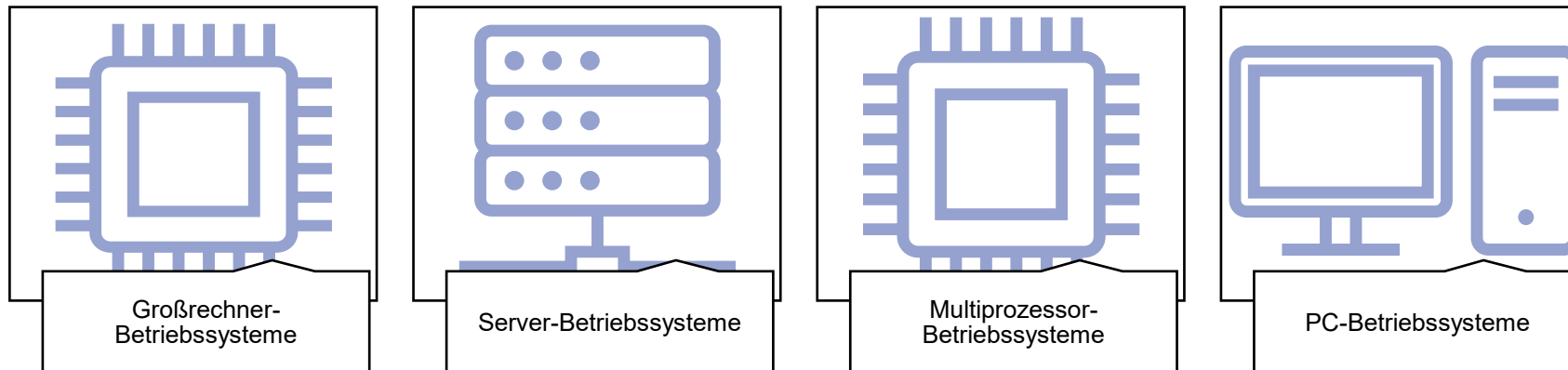
- Bereitstellung einer Benutzer-Schnittstelle
 - In Form einer Kommandozeile und/oder einer grafischen Benutzeroberfläche
 - Zur Verwaltung des Betriebssystems
 - Zur Steuerung und Verwaltung von Peripheriegeräten
 - Für die Ein- und Ausgabe
 - Zur Konfiguration und Verwaltung der Berechtigungen
 - Zum Ausführen und Beenden von Benutzerprogrammen



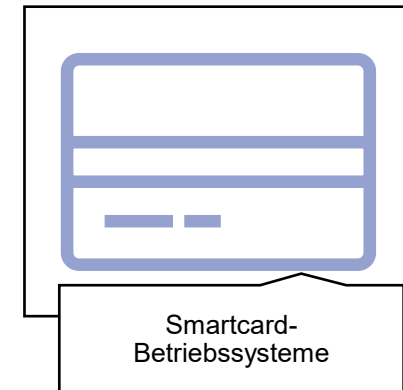
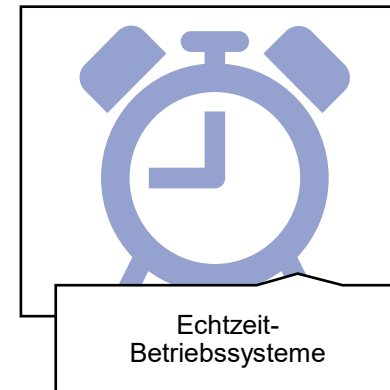
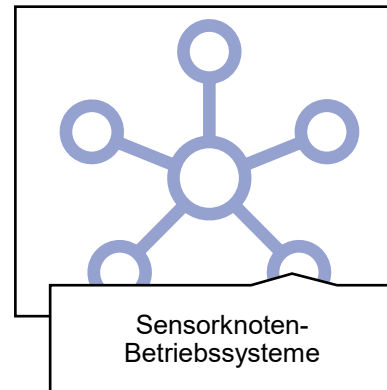
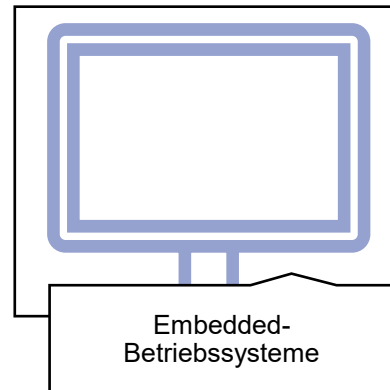
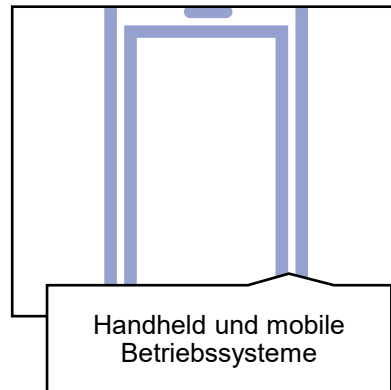
Kapitel IV

Betriebssystemarten

Basierend auf den vorgestellten Hardware-Komponenten und Aufgaben unterscheidet man Betriebssysteme anhand ihrer Charakteristika. In Anlehnung an [MB17] existiert dabei folgende Artenliste:

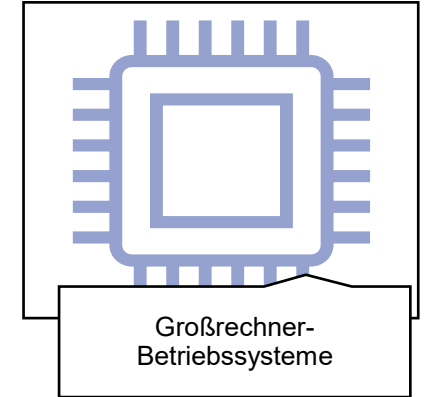


Forts. Artenliste:



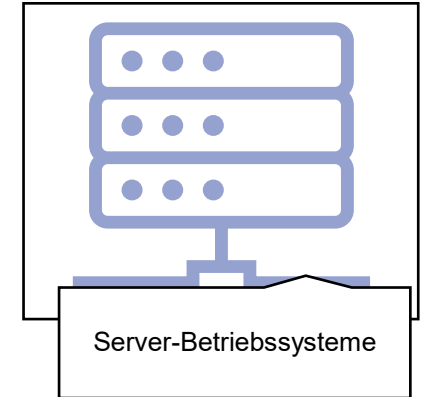
Großrechner-Betriebssysteme:

- Sehr hohe Ein- und Ausgabeleistung
- Hohe Kapazität an Ressourcen:
> 1.000 Festplatten und >1.000.000 Gigabyte an Daten sind nicht ungewöhnlich
- Sind stark darauf ausgelegt, viele Prozesse gleichzeitig auszuführen, die hohen Bedarf an schneller Ein- und Ausgabe aufweisen
- Typischerweise: 3 Arten der Prozessverwaltung:
 - Stapelverarbeitung / Batch Processing
(Automatisierte Routineaufgaben großen Umfangs ohne interaktive Benutzersitzung, z. B. Fortschreiben von Girokonten bei Banken, größere Druckaufträge, ETL-Prozess im Data Warehouse)
 - Dialogverarbeitung / Interaktive Verarbeitung
(Auftragsbearbeitung im Wechsel zwischen Benutzer und System. Dem Betriebssystem werden kleine Teilaufträge erteilt, welche nacheinander abgearbeitet werden, z. B. Sachbearbeitung im Reisebüro, Lagerfachkräfte in Logistikzentren)
 - Timesharing
(Praktisch gleichzeitige Abarbeitung vieler unterschiedlicher Aufgaben durch das Betriebssystem, z. B. zeitgleiches Ausführen mehrerer Datenbank-Anfragen)
- Beispiel-Betriebssysteme: OS/360 und OS/390 → nach und nach Verdrängung durch Unix-Varianten wie Linux, IBM z/OS



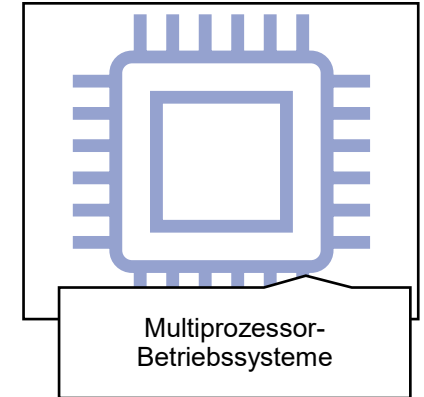
Server-Betriebssysteme:

- Entworfen für Server, die leistungsfähige PCs, Workstations oder Großrechner sein können
- Zentrale Verantwortung besteht in gleichzeitiger Abarbeitung vieler Benutzeranfragen über ein Netzwerk zur Verteilung von Hard- und Softwareressourcen
- Anwendungsfälle sind z. B.: Druckserver, Dateiserver, Webserver
- Beispiel-Betriebssysteme: Solaris, FreeBSD, Linux, Windows Server 20xx



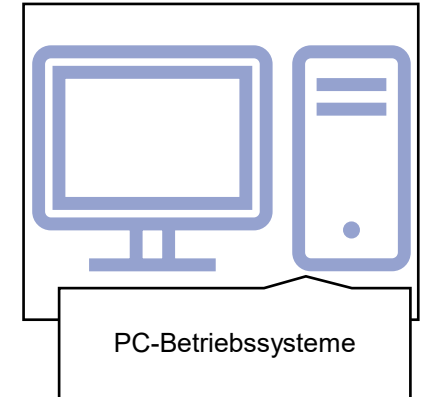
Multiprozessor-Betriebssysteme:

- Sind darauf ausgelegt, mehrere hardwareseitig existierende Prozessoren zu einem System zusammenzuschalten
- Hierbei gelten besondere Anforderungen in Bezug auf (Auswahl):
 - Kommunikation zwischen Prozessen und/oder Prozessoren
 - Konsistenz
 - Speicherverwaltung
- Beispiel-Betriebssysteme: U. a. Windows, Linux



PC-Betriebssysteme:

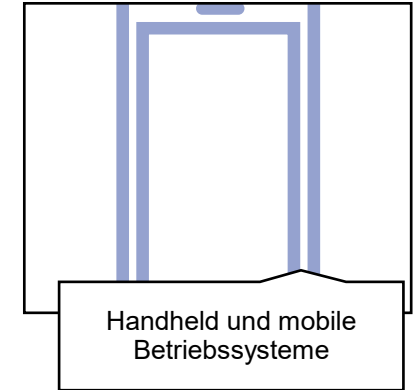
- Zielsetzung besteht in der Unterstützung einer großen, universellen Gruppe von privaten und geschäftlichen Anwendern
- Dabei:
Betriebssystem sollte möglichst umfangreiches Tooling¹ mit sich bringen und gleichzeitig robust und fehlertolerant in der Bedienung sein
- Heutzutage:
Alle modernen PC-Betriebssysteme sind Multiprogrammiersysteme, d. h. sie können eine Vielzahl von Applikationen gleichzeitig ausführen
- Beispiel-Betriebssysteme: Windows, Linux, MacOS



¹Der Begriff „Tooling“ meint in diesem Zusammenhang die zur Verfügung gestellte Werkzeugunterstützung, z. B. durch integrierte Browser, Textverarbeitung, Media Player usw.

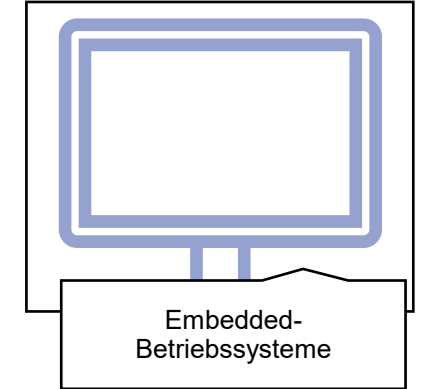
Handheld und mobile Betriebssysteme:

- Fokus auf kleinere, mobile Hardwaresysteme, die im Verhältnis sehr leistungsfähig sind
- Zeichnen sich durch eine Vielzahl kleiner Anwendungen (Apps) von Drittanbietern aus und gewinnen bzw. verlieren dadurch Marktanteile
- Herausforderungen:
 - Energieeffizienz
 - User Experience / Developer Experience
 - Weitere vgl. VL-Abschnitt 1
- Bei Handheld-Computern: Betriebssystem zumeist nur auf eine spezielle Aufgabe ausgelegt
- Beispiel-Betriebssysteme: Android, iOS



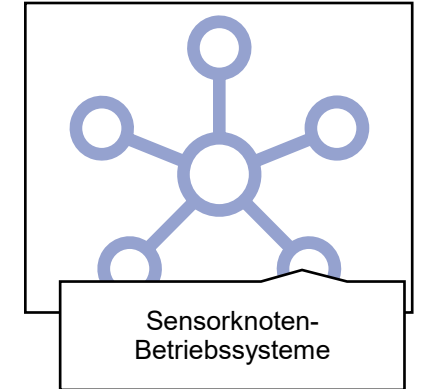
Embedded Betriebssysteme:

- Fokussierung auf Rechensysteme, die Geräte steuern, welche man als solche in der Regel nicht als Computer wahrnimmt
- Benutzer können auf diese Systeme in aller Regel keine eigene Software installieren, sondern haben vorinstallierte Applikationen und Funktionsumfang im ROM
- Beispiele für Geräte: Mikrowellen, TVs (ausgenommen Smart-TVs), Autos, MP3-Player, ...
- Beispiel-Betriebssysteme: Embedded Linux, QNX, VxWorks, Windows Embedded bzw. Windows IoT



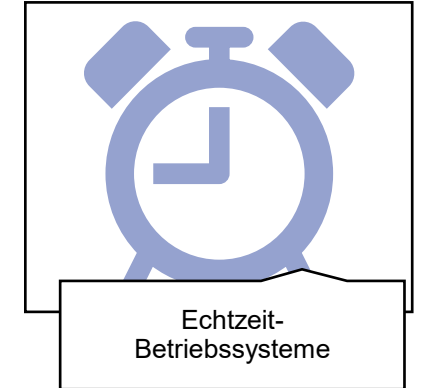
Sensorknoten-Betriebssysteme:

- Fokussierung auf Netzwerke winziger Sensorknoten, die für unzählige Zwecke eingesetzt werden:
 - Gebäudeschutz
 - Überwachung von Landesgrenzen
 - Temperatur- und Niederschlagsmessung
 - ...
- Hierbei: Verwendung sehr kleiner Betriebssysteme, die ereignisorientiert arbeiten oder periodische Messungen durchführen und Daten via Netzwerk übermitteln
- Wie bei eingebetteten Systeme: alle Anwendungen sind bereits vorinstalliert
- Beispiel-Betriebssystem: TinyOS



Echtzeit-Betriebssysteme (RTOS):

- Dienen der Erfüllung zeitkritischer Aufgaben
- Augenmerk liegt auf der richtigen, priorisierten Ressourcenvergabe entsprechend den Echtzeit-Anforderungen → Reaktionszeit ist wichtiger als Durchsatz!
- Dabei: Einteilung in 2 Klassen:
 - Weiche Echtzeitsysteme: Verpasste Deadlines nicht erwünscht, aber tolerierbar
Beispiel: Digitale Audio- oder Multimediasysteme
 - Harte Echtzeitsysteme: Verpasste Deadlines nicht tolerierbar
Beispiel: Medizingeräte, Airbagsteuerung
- Beispiel-Betriebssystem: eCos, FreeRTOS, VxWorks (Luft- und Raumfahrt / Verteidigung)



Smartcard-Betriebssysteme:

- Laufen auf Smartcards, die die Größe einer Kreditkarte aufweisen und eigenen Prozessor besitzen
- Werden z. B. verwendet für:
 - Bezahlvorgänge
 - Authentifizierung und Autorisierung
- Ausgelegt auf minimale Rechenleistung und Speicherkapazität
- Beispiel-Betriebssysteme: Zumeist proprietär, BasicCard, CardOS, Java Card

