



Angewandte Computer- und Biowissenschaften



**HOCHSCHULE
MITTWEIDA**
University of Applied Sciences

Betriebssysteme

Betriebssystemarchitektur

Autor: Ronny Bodach, Felix Fischer

Stand: 27.04.2022



Fraunhofer
SIT



Bundeskriminalamt

[hs-mittweida.de](https://www.hs-mittweida.de)

Agenda

1. Grundlegender Aufbau von Rechnern
2. Von Neumann Rechnerarchitektur
3. Harvard-Architektur
4. Computerarchitektur nach Tanenbaum
 1. Transistorebene
 2. Logische Ebene
 3. Microarchitektur
 4. Instruction Set Architecture
 5. **Betriebssystem**
5. **Betriebssystem-klassifikation**
6. **Interrupts**
7. Prozess, Task und Thread
8. Scheduling
9. Parallelität und Nebenläufigkeit
10. Speicherverwaltung

Ressourcenverwaltung

- **Physikalische Ressourcen (Hardwareressourcen)**
 - CPU
 - Speicher
 - Ein/Ausgabegeräte
 - Schnittstellen extern
 - Festspeicher
- **Virtuelle Ressourcen (geschaffene, logische Ressourcen)**
 - Speichersegmente
 - Dateien
 - Gerätedateien

Aufgaben für das Betriebssystem

- Belegung der Ressourcen je nach Anforderung
- Multiplexen von Ressourcen für unterschiedliche Benutzer bzw. Prozesse
- Anwendung von Schutzmechanismen
 - Wechselseitiger Ausschluss von Ressourcenzugriffen
 - Umsetzung von Zugriffsberechtigungen
- Fehlerbehandlung und Fehlertoleranz

Ressourcennutzungs koordinierung

Die Nutzung von Ressourcen lässt sich in folgende drei Punkte einteilen:

- Aktive, zeitlich aufteilbare Ressource
 - Prozessor
 - Grafikkarte
- Passive, ausschließlich exklusiv nutzbare Ressource
 - Drucker
 - Andere E/A-Geräte
- Passive, räumlich teilbare Ressource
 - Festspeicher
 - Andere Speicher

Ressourcenverwaltung

Dafür genutzte Einzelkomponenten des Betriebssystems:

- Anwendungs- und Prozessverwaltung
- Dateisysteme
- Speicherverwaltung
- Verwaltung der Ein-/Ausgabegeräte

Steuerung der Ressourcenverwaltung durch:

- Kommandozeileninterpreter (Shell)
- GUI (Graphical User Interface)
- Programmablaufsteuerungen
- Systemkonfigurationen und Systemdiensten

Realisierungsabhängigkeiten

Hardware und Konfiguration

(Verteilungen und Abstraktionen)

- Basisarchitektur
- Betriebsarten

Programmiersprachen und Programmstrukturierungen

- Ablaufsteuerungen und deren Komponenten:
 - Tasks
 - Prozesse
 - Threads
- Interaktionsmöglichkeiten:
 - Programmaufrufe
 - Benachrichtigungen
 - Datenaustausch
 - Datenkommunikation

Betriebssystem- klassifikation

Betriebssystemklassifikation

Hinsichtlich der **Betriebssysteme** finden sich in der Literatur **unterschiedliche Klassifizierungsansätze**.

- Tannenbaum beschreibt 2009 eine Artenliste, die sich insbesondere an der Hardware orientiert, für welche die jeweilige Betriebssystemart konzipiert wurde.
- Mandl 2013 klassifiziert in den Kapiteln eher nach dem Einsatzszenario, unter dem das jeweilige Betriebssystem nach seiner Installation betrieben werden soll.

Klassifizierung nach Betriebsarten

Die **Betriebsart** (Nutzungsform, operation mode) legt die **Art** und **Weise** der **Kommunikation mit dem Benutzer** fest.

- Stapelverarbeitung (batch processing)
- Dialogbetrieb (interactive processing)
- Zeitgesteuerter Betrieb (time sharing processing)
- Echtzeitverarbeitung (real time processing)
- Verteilte Verarbeitung (distributed processing)

Stapelverarbeitung

- Abarbeitung einer Folge von Stapelaufträgen (Batch Job)
- Benutzer definiert für Auftrag mit Job Control Language (JCL)
 - erforderlichen Programme
 - Daten
 - Anweisungen zur Ablaufsteuerung
- Stapelauftrag wird ohne Interaktion des Benutzers vollständig abgearbeitet
- Beispiele:
 - IBM OS/370
 - OS/390
 - MVS
 - BS 2000

Dialogbetrieb

- Ständiger Wechsel zwischen
 - Eingabeaktionen des Benutzers (z. B. Kommandoeingaben)
 - Systemaktionen (z.B. Kommandoausführung)
 - Nutzer kann Arbeitsablauf durch Dialog jederzeit beeinflussen
- Beispiele:
 - MS-DOS
 - MS Windows 95/98/ME/NT/2000/XP/7/8/10
 - UNIX/Linux

Zeitgesteuerter Betrieb

- Aufteilung von Rechenzeit auf
 - mehrere Benutzer
 - mehrere Programme
- feste Timeslots für Ressourcenzuordnung
- Beispiele:
 - Spezialhardware mit fester Arbeitstaktung
 - Backbone Router

Echtzeitverarbeitung

- Steuerung und Überwachung von zeitkritischen, technischen Prozessen
- Echtzeit-Betriebssystem garantiert Rechtzeitigkeit (Einhaltung von Zeitbedingungen)
- Beispiele:
 - VxWorks
 - LynxOS
 - QNX
 - VRTX
 - Enea OSE
 - OSEK/VDX (speziell im Automobilbau, aber auch SCADA Systeme)
 - pSOS
 - MS Windows CE

Verteilte Verarbeitung

- Besteht aus miteinander gekoppelten Computern
- Betriebssystem dient vorrangig der Verteilung von
 - Daten
 - Ressourcen
 - Arbeitslast
- Beispiele:
 - Amoeba
 - CHORUS
 - MACH
 - Netzwerk-Betriebssysteme
 - Novell Netware
 - IBM AS-400
 - Parallelrechner-Betriebssysteme
 - UNICOS
 - SPP-UX
 - KSR-OS

Klassifizierung nach Benutzeranzahl

- Aufteilung in Benutzeranzahl mit Gleichzeitiger Verwaltung
 - Einzelnutzer-System (single user system)
 - Mehrnutzer-System (multi user system)
- Vorteile:

Einzelnutzer-System	Mehrnutzer-System
einfacher Aufbau	mehrere Nutzer gleichzeitig
vereinfachte Rechteverwaltung	unterschiedliche Rechte für Programme
ressourcensparender	
schnellere Ausführung	

Einzelnutzer-System

- Authentifizierung und Rechte mehrerer Nutzer auf dem System möglich
- jedoch stets nur ein Nutzer zeitgleich
- keine parallele Erkennung bzw. Verwaltung von mehreren Nutzern
- Beispiele
 - MS-DOS
 - MS Windows 3.X/95/98/ME
 - Linux im single user mode

Mehrnutzer-System

- gleichzeitige Systemnutzung durch mehrere Benutzer
- Programme können für verschiedenen Nutzer laufen
- gleiche Programme mit verschiedenen Rechten möglich
- über mehrere Terminals oder einzelne Anmeldungen
- Beispiele:
 - UNIX / Linux
 - IBM OS/390
 - OS/400
 - BS2000
 - OpenVMS
 - Windows XP/Vista/2000/Server/7/8/10

Klassifikation nach Prozessanzahl

Einzelprozess-System (single tasking system)

- nur ein Auftrag (Task)
- nächster Auftrag startet nach Ende des Bisherigen
- Beispiele:
 - CP/M
 - MS-DOS.

Mehrprozess-System (multi tasking system)

- „gleichzeitige“ Ausführung von Prozessen
- parallel oder zumindest quasi-parallel (zeitlich geschachtelt) verarbeitet
- Beispiele:
 - MS Windows 95/98/ME/NT/2000/CE/XP/SERVER/VISTA/7/8/10
 - UNIX/Linux
 - IBM OS/390
 - OS/400
 - OS/2
 - BS2000
 - OpenVMS
 - VxWorks
 - VRTX
 - LynxOS
 - Enea OSE

Betriebssystem Modi

- Verwaltung von Zugriffsrechten bei parallelen Prozessen notwendig
- Ressourcen-Wechsel muss gewährleistet werden
 - Zugriff auf externe Hardware
 - Speichern und Laden vom Daten
 - Hauptspeicherreservierung oder -freigabe
- Abarbeitung auf CPU in zwei Modi
 - Kernel Mode
 - User Mode
- Systemaufrufe ermöglichen Wechsel zwischen Modi

Kernel Mode (privilegierter Modus)

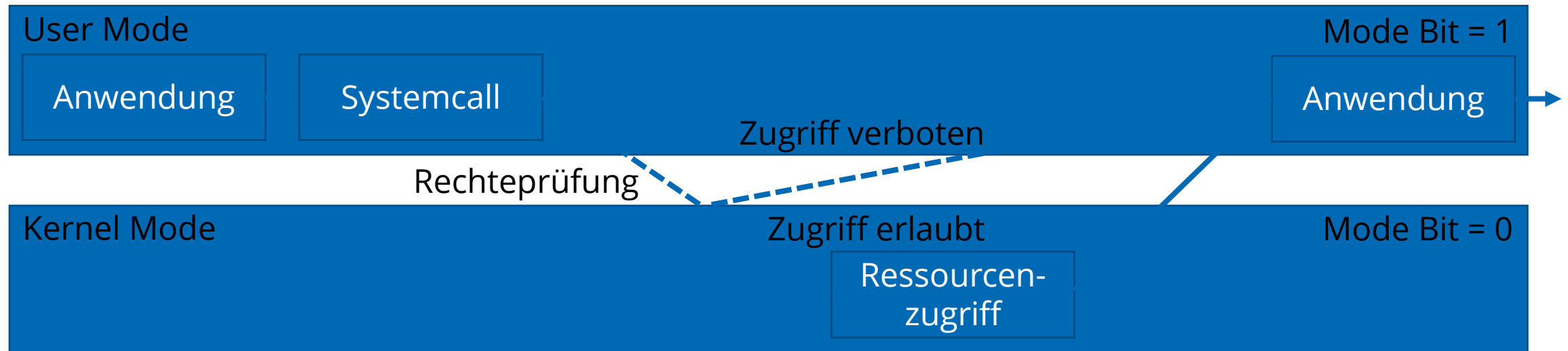
- jeder beliebige Befehl zur Ausführung zugelassen
- Steuer- oder Kontrollregister markieren Kernel Mode auf CPU
- höchsten Privilegien
- Zugriff auf
 - sämtliche Speicherbereiche für Daten- und Programmtext
 - alle Betriebsmittelressourcen
- Betriebssystem arbeitet im Kernel-Mode
- besitzt alle Möglichkeiten für Aufgabenerfüllung

User Mode

- eingeschränkter Befehlssatz zur Ausführung zugelassen
- eingeschränkte Zugriffsmöglichkeiten auf
 - Speicherbereiche
 - Betriebsmittelressourcen
- Anwendungsprogramme arbeiten im User-Mode
- Übergang Kernel-Mode in User-Mode
 - unproblematisch
 - keine Rechteeinschränkung
 - keine Sicherheitsbedenken
- Übergang vom User-Mode in Kernel-Mode
 - problematisch
 - Programm erhält zusätzliche Rechte

Systemaufruf

- Betriebssystem stellt Systemschnittstellen zur Verfügung
- User-Mode-Prozess führt Systemaufruf (Systemcall, Syscall) durch
- Betriebssystem führt Sicherheitsüberprüfungen durch
- Betriebssystem führt Zugriff im Kernel-Mode aus
- Betriebssystem übergibt Kontrolle an Programm im User Mode



Betriebssystemarchitekturen

Architekturmodelle verdeutlichen die **Anordnung** der **Komponenten** des **Betriebssystems** und ihr **funktionales Zusammenwirken**. Sie geben **zudem** auch **Aufschluss** über die **Portabilität des Systems**.

In der Literatur zu Betriebssystemen beschreibt Mandl 2013 unter anderem die **Architektur** des **monolithischen Kernels** und dessen Weiterentwicklung, den **schichtenorientierten Kernel** und stellt beiden Varianten die **Mikrokern-Architektur** gegenüber.

Architekturmodelle

Monolithische Architektur:

Alle wesentlichen Komponenten des Systems sind zu einem homogenen Gebilde zusammengefügt, das zwar u. U. effizient, aber nicht flexibel anpassbar ist. Kernel Mode (privilegiert) für das gesamte Betriebssystem und der User Mode (nicht privilegiert) für Anwendungsprogramme.

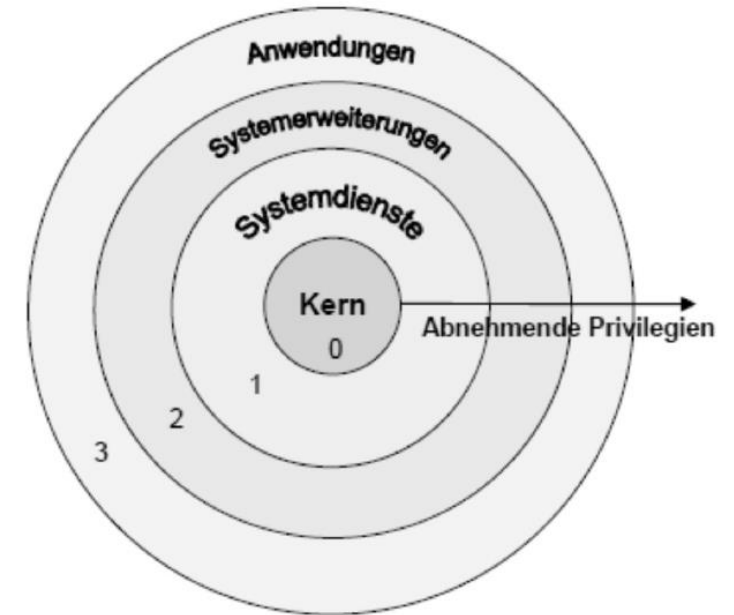
Kern-Schale-Architektur:

Das System besteht aus dem privilegierten Kern (kernel), der die wichtigsten Komponenten vereint (z.B. die Prozessverwaltung), und einer Schale (shell) für ergänzende Bestandteile (z.B. Kommando-Interpreter). Typischer Vertreter dieses Modells ist UNIX.

Hierarchische Schichten

Das System wird modularisiert und in einzelne Schichten geteilt. Zwischen diesen sind Schnittstellen definiert, so dass sie austauschbar sind. Sie können mit abgestuften Privilegien ausgestattet sein.

Diese Architektur ist weit verbreitet, z.B. bei MS-DOS, OS/2, und OpenVMS.

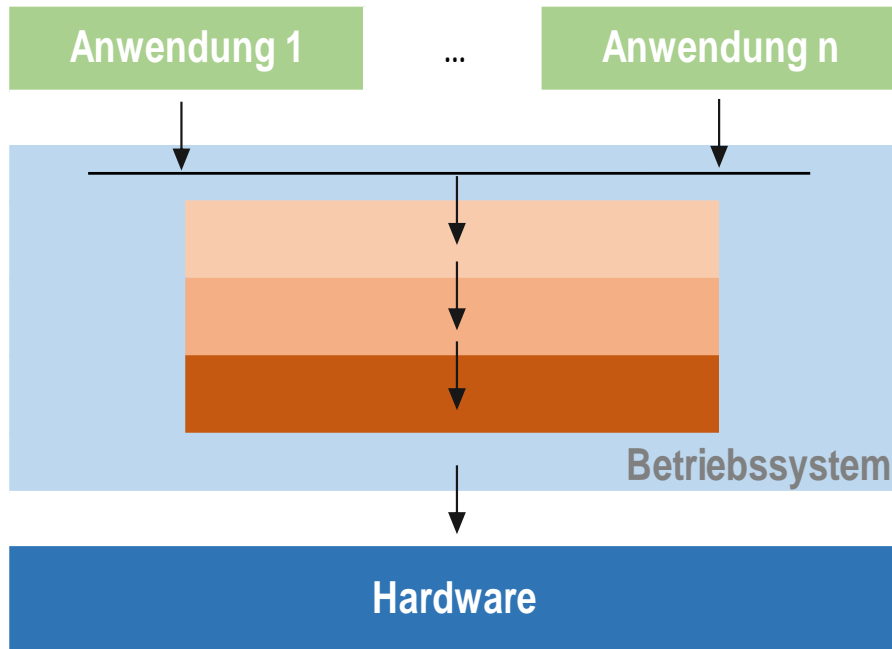


Hierarchische Schichten

In der Hierarchie bietet jede Schicht ihre Dienstleistungen der (den) darüber liegenden Schicht(en) an und greift gleichzeitig zur Erfüllung ihrer Aufgaben auf die Dienste der tiefer liegenden Schicht(en) zurück. Die unterste Schicht ist im Allgemeinen die Hardware, die oberste oft ein Kommandointerpreter.

- **Konsistente Schichten** sind leicht austauschbar, das strenge Durchlaufen aller Schichten führt aber oft zu Effizienzverlusten.
- Bei **quasi-konsistenten Schichten (Treppenstufenmodelle)** kann dagegen bei Bedarf auf verschiedene tiefer liegende Schichten zugegriffen werden. Dies führt jedoch u. U. zu unkontrollierbaren Abläufen.

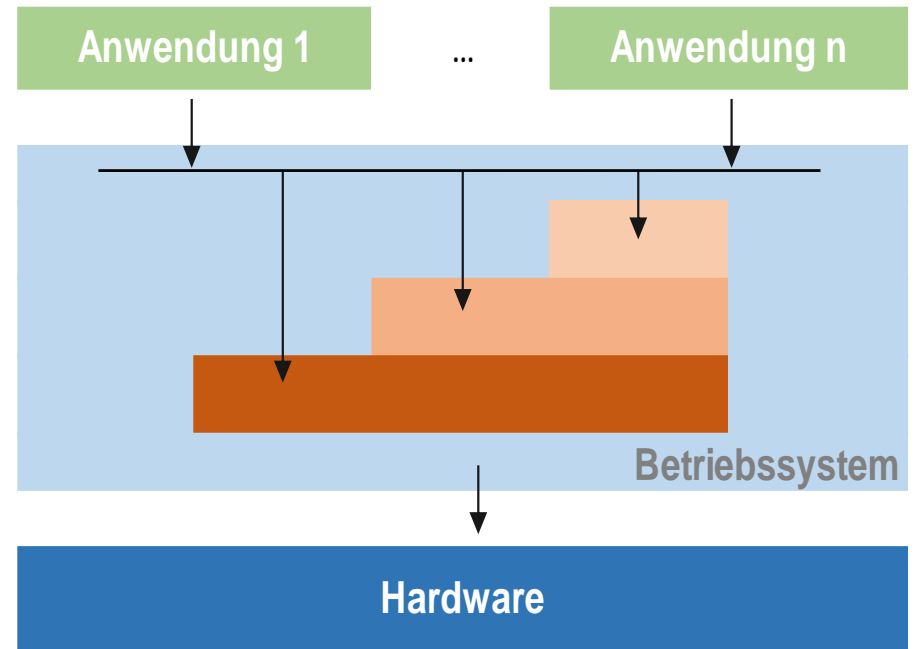
Hierarchische Schichten



Konsistente Schichten

User Mode

Kernel Mode



Quasi-konsistente Schichten

Mikrokern (micro kernel)

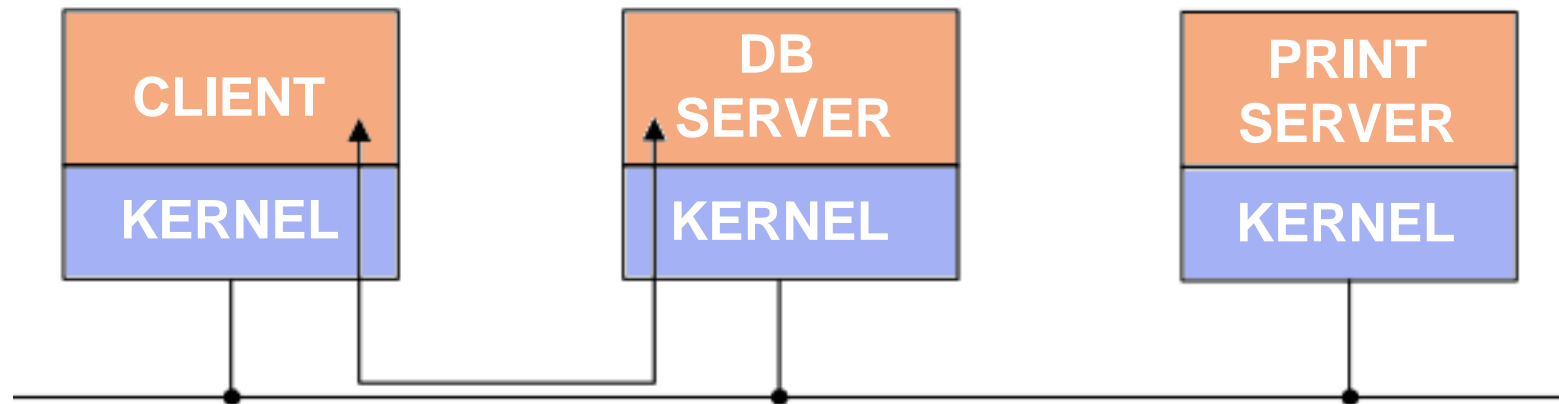
Er bildet nur noch eine Art Infrastruktur mit minimalem Funktionsumfang. Alle anderen Betriebssystemfunktionen werden durch Systemprozesse außerhalb des Kerns erbracht, die flexibel modifiziert oder erweitert werden können.

Mikrokerne findet man z.B. bei MACH (OS X), CHORUS, QNX/Neutrino, z.T. bei MS Windows NT/XP/VISTA/2000/VISTA/7/8/10/SERVER

Mikrokerne enthalten nur noch elementare Funktionen, z.B. zur Speicherverwaltung, IPC, Prozessverwaltung, Scheduling-Mechanismus, sowie einige hardwarenahe E/A-Funktionen. Systemprozesse realisieren dagegen den Rest der Verwaltung wie z.B. die Dateiverwaltung

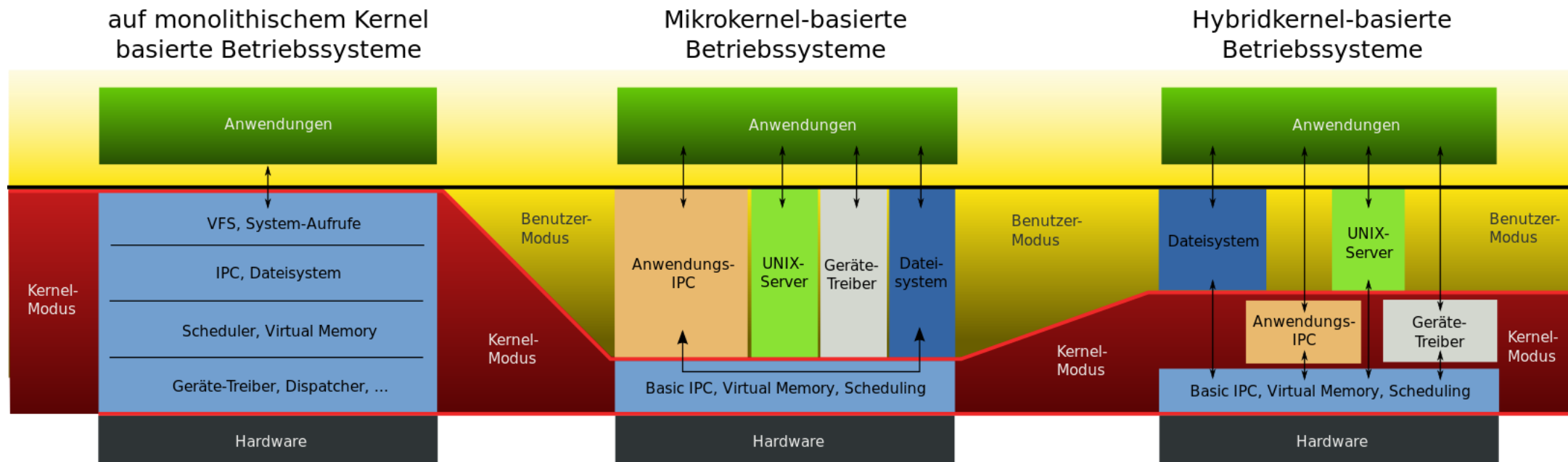
Client-Server-Modelle

- Basieren häufig auf Micro Kernel Architekturen
- ein Server bietet Dienste an, die von den Clients genutzt werden
- Kommunikation erfolgt durch einen Nachrichtenaustausch



Hybridkernel (Makrokernel)

ist ein Kompromiss zwischen einem Mikrokernel und einem monolithischen Kernel, bei dem aus Performancegründen Teile des monolithischen Kernel in den Kern integriert werden und dieser dadurch kein reiner Mikrokernel mehr ist.



Quelle: Goltheman übersetzt von Hagen

Hybridkernel (Makrokernel)

- Vorteile des Hybridkernel:
 - weniger fehleranfällig wie ein monolithischer Kernel, da nicht alle Treiber im privilegierten Modus laufen und bei einem Absturz das ganze System zum Absturz bringen
 - höhere Geschwindigkeit des Kernels, da nicht so viele Kontextwechsel und Kommunikation nötig wie bei einem Mikrokernel
- Beispiele:
 - OSX
 - NT/XP/VISTA/2000/VISTA/7/8/10/SERVER

Virtuelle Maschinen

Einsatz eines Basis-Betriebssystems dem Hypervisor auch als Virtual Machine Monitor bezeichnet, auf dem virtuelle Maschinen in Form unterschiedlicher verschiedener Betriebssysteme ablaufen. Diese sind von der realen Hardware und untereinander völlig entkoppelt.

Beispiel: IBM VM/370, ESXi, VMWare, VirtualBox, unter anderem auch im Zusammenhang mit Java etwa bei der Dalvik Engine von Android von Bedeutung

Klassifizierung Betriebssystemarten

- Betriebssysteme für Großrechner
- Betriebssysteme für Server
- Betriebssysteme für Laptops und Personal Computer
- Echtzeitbetriebssysteme
- Betriebssysteme für Embedded Systems
- Betriebssysteme für Chipkarten

Großrechner (Mainframes)

verarbeiten im Regelfall eine sehr hohe Zahl an Aufgaben in sehr kurzer Zeit. Meist sind sie optimiert auf eine sehr hohe Ein/Ausgabe-Rate und verfügen optional über eine sehr große Speicherkapazität auf angeschlossenen Speichersystemen.

Mainframe-Betriebssysteme arbeiten üblicherweise im Stapelbetrieb (Batch-Jobs) oder sind transaktionsorientiert, was bedeutet, dass eine sehr große Zahl an Transaktionen in sehr kurzer Zeit ausgeführt werden kann. Verschiedene Subsysteme ermöglichen den gleichzeitigen Betrieb im Dialog und im Batchbetrieb.

Ein Beispiel für ein Mainframe-Betriebssystem ist z/OS von IBM. Es ist u.a. auf Großrechnern bei Banken und Versicherungen im Einsatz.

Betriebssysteme für Server

Server werden überwiegend von vielen verschiedenen Clients über ein Netzwerk angesprochen. Charakterisiert werden Server-Betriebssysteme dadurch, dass sie eine möglichst schnelle Reaktionszeit mit nur minimaler Verzögerung für die Datenbereitstellung benötigen. Zudem werden rechenintensive Aufgaben auf die zentrale Server Architektur ausgelagert (Print/Datenbank/Terminal).

Vertreter für Server-Betriebssysteme sind Windows Server, Unix bzw. Linux aber auch Mac OS X. Eine grafische Oberfläche ist nicht immer vorhanden.

Die Einsatzzwecke für Server-Betriebssysteme liegen beispielsweise bei:

- File-Server (Dateiablagen)
- Print-Server
- E-Mail-Server
- Terminal-Server
- Datenbank Server

Betriebssysteme für Laptops und Personal Computer

Mit dieser Art Rechnersystem dürften die meisten täglich in Berührung kommen. Durch verschiedene Benutzer können verschiedene Anwendungen quasi-parallel genutzt werden.

Es handelt sich also um eine überwiegend dialogorientierte Nutzungsart: Der Anwender tätigt eine Eingabe und das Betriebssystem reagiert darauf.

In der Praxis werden heute überwiegend Betriebssysteme mit grafischer Oberfläche eingesetzt, z.B. Windows, MacOS X oder Linux mit KDE, Gnome bzw. Unity.

Echtzeitbetriebssysteme

Spezielles Betriebssystem bei denen die Verarbeitung von Informationen zeitkritisch ist und diese in nahezu Echtzeit erfolgen muss. Das Erfordernis eines Echtzeitbetriebssystem ergibt sich immer dann, wenn Rechner mit der physikalischen Welt messend und/oder steuernd in Verbindung stehen.

Das bedeutet das gesicherte Verarbeiten von Anfragen eines Anwendungsprogramms oder dem Eintreffen von Signalen über Hardware-Schnittstellen innerhalb einer im Vorhinein bestimmbar Frist (Timeout, t_{max}). So sind zum Beispiel bei einer Heizungssteuerung längere t_{max} -Werte anwendbar als bei der Auslösung eines Airbags.

Beispiele: Steuerungsanlagen, Ampelsteuerungen, Robotertechnik

Betriebssysteme für Embedded Systems

Hierbei handelt es sich um spezielle Betriebssysteme für Mikrocomputersysteme. Typischerweise im Einsatz bei tragbaren Geräten aber auch bei integrierten Bauteilen mit CPU.

Typische Anwendungsbeispiele:

- Automobilindustrie
- Industriesteuerungsanlagen
- tragbare Navigationsgeräte
- Waschmaschinen, Haushaltsgeräte
- IoT Geräte

Betriebssysteme:

- Windows CE, Windows Mobile, Palm OS, Android, Linux

Betriebssysteme für Chipkarten

Spezialbetriebssysteme für Chipkarten mit meist sehr reduziertem Funktionsumfang. Lauffähig auf Chipkarten wie etwa Smartcards von Bezahldiensten oder SIM Karten von Telefonanbietern.

Vorrangig werden dabei Rechenoperationen abgebildet, die durch das Chipkarten Betriebssystem durchgeführt werden, um Verschlüsselungen und Speicherzugriffe zu realisieren.

Betriebssysteme und deren Hardwareunterstützung

Moderne Betriebssysteme werden durch verschiedene Hardwarebestandteile unterstützt:

- CPU / Prozessor
- Memory Management Unit (MMU)
- DMA Controller (Direct Memory Access)
- Ein-/Ausgabe Controller (USB, PCIe, SATA etc...)

Betriebssysteme und deren Hardwareunterstützung

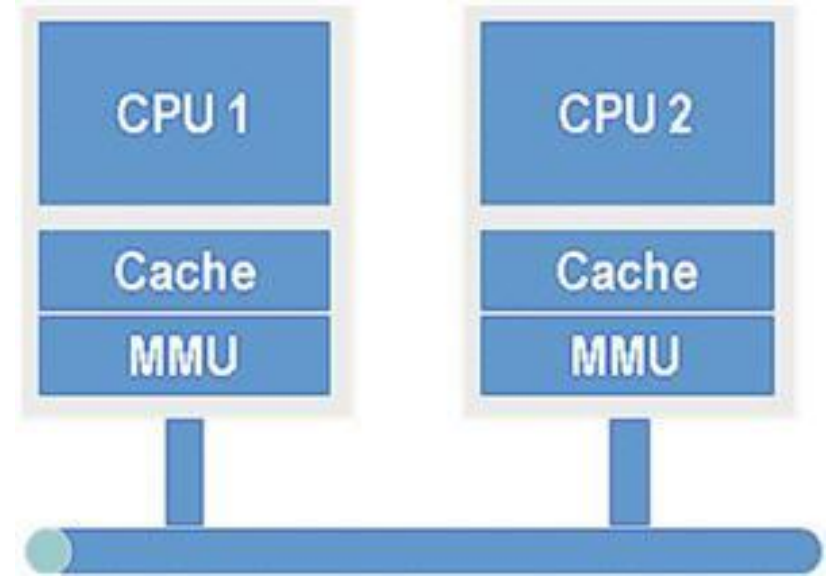
Der **Prozessor** ist **Kernbestandteil** der modernen Hardware für **Betriebssysteme**.

Er **enthält** die digitalen **Schaltungen** für die **Rechenarithmetiken**, die **Register** für die **Daten** und **Befehle** und **führt** die einzelnen **Instruktionen** auf Grund des **Programmzählers aus**.

Dabei arbeitet der Prozessor wie bereits festgestellt in **zwei Modi**. Dem **Benutzermodus (User Mode)** mit eingeschränktem Befehlssatz und dem **privilegierter Modus (Kernel Mode)** der die Ausführung privilegierter Befehle erlaubt.

Multiprozessoren

Multiprozessorsysteme enthalten mehrere Prozessoren, die sich auf verschiedenen Chips befinden. Diese Systeme verbreiteten sich in den 1990ern, v. a. im Bereich von IT-Servern. Zu dieser Zeit waren das meist Prozessorplatinen, die in einen Rack-Mount-Server installiert wurden. Heute befinden sich Multiprozessoren oft auf derselben Platine und werden über eine Hochgeschwindigkeits-Kommunikation angebunden.

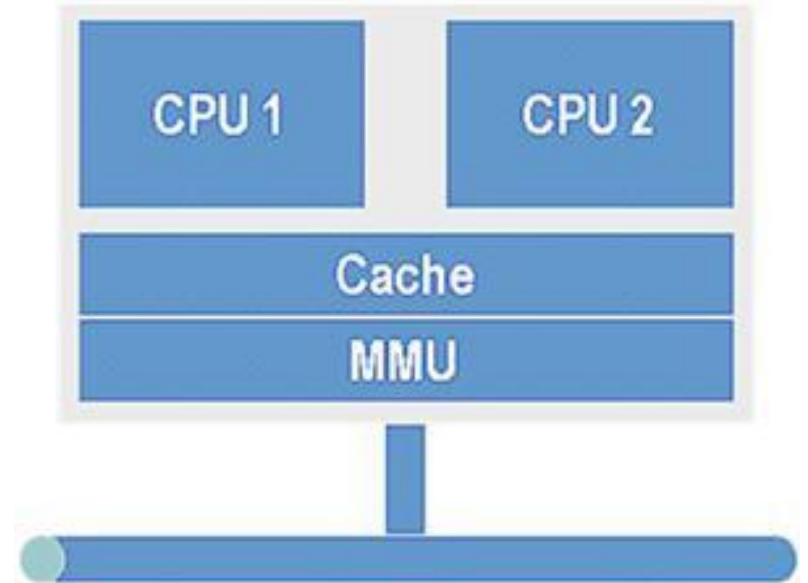


Quelle: National Instruments

Multicore-Prozessoren

Multicore-Prozessoren werden von einer „Familie“ von Prozessoren gebildet, die mehrere Prozessoren, z. B. 2, 4 oder 8, auf einem Chip integrieren.

Allerdings stellen Multicore-Prozessoren Softwareentwickler vor eine Herausforderung. Höhere Geschwindigkeiten und damit bessere Leistung sind direkt davon abhängig, wie parallel der Quellcode einer Anwendung geschrieben wurde.



Quelle: National Instruments

Betriebssysteme und deren Hardwareunterstützung

Bei der Prozessausführung in Prozessoren kann es notwendig sein auf vordefinierte Ereignisse zu reagieren:

- Fehlerbedingung
- Ankommende Netzwerknachrichten
- Rückmeldung durch externe Geräte



Diese Unterbrechungen werden bezeichnet als Interrupts

Interrupts

Interrupts

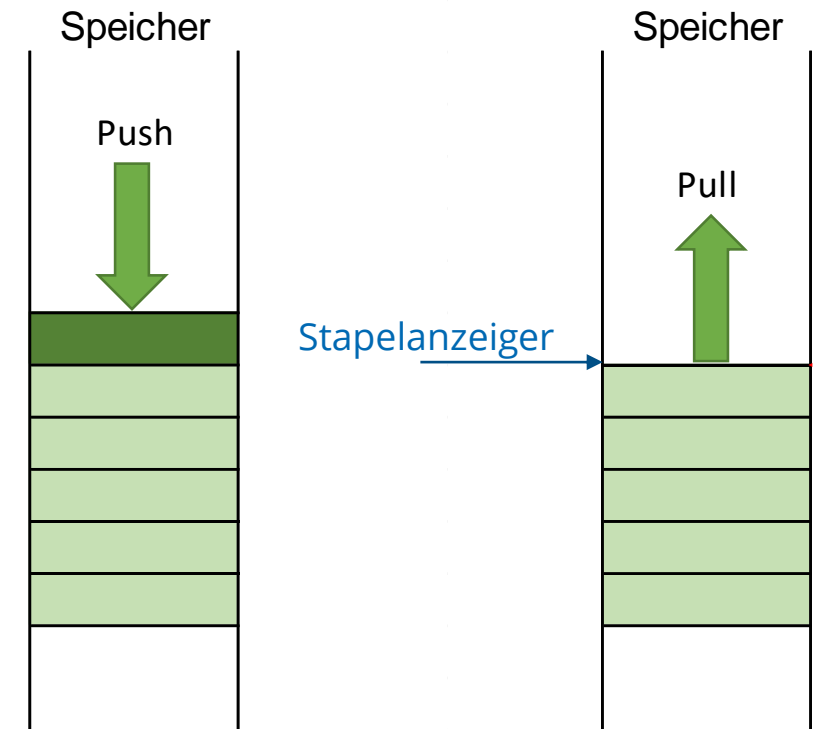
- Prozessor unterbricht laufende Bearbeitung und führt eine definierte Befehlsfolge aus (diese sind vom privilegierten Modus aus konfigurierbar)
- Vorher werden alle Register einschließlich Programmzähler gesichert (z.B. auf einem Stapelspeicher/Stack)
- Nach einer Unterbrechung kann der ursprüngliche Zustand wiederhergestellt werden
- Unterbrechungen werden im privilegierten Modus bearbeitet
- Signalisieren der Unterbrechung durch Interrupts (Interrupt Request, IRQ)

Interrupts

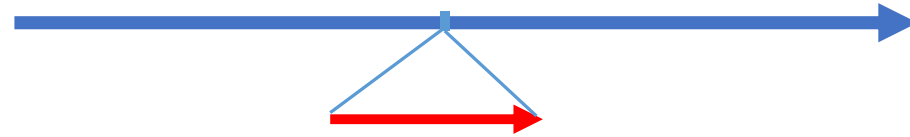
- Die Ansprung Adresse der Unterbrechungsbehandlung steht fest
 - diese ist meist konfigurierbar
- Rückkehr von der Unterbrechungsbehandlung erfolgt durch
 - Nutzung eines speziellen Befehl, z.B. RTI, Return from Interrupt
 - Befehlsausführung bei einem RTI
 - Wiederherstellung gesicherter Register
 - Rückschaltung in den alten Modus
 - Abarbeitung ab der alten Befehlsfolge

Interrupts

- Einsatz eines Stapelspeichers (Stack)
 - **Push**
Speichert ein Datum „oben“ auf dem Stapel
 - **Pull**
Holt zuletzt gespeichertes Datum vom Stapel



Charakteristika für Externe Unterbrechungen durch Interrupts



- Unterbrochene Befehlsfolge bleibt in der Regel unberührt
 - solche Unterbrechung sind transparent und berühren den Programmablauf nicht
- Verschachtelte Unterbrechungen sind möglich
 - d.h. die erneute Unterbrechung der Unterbrechungsbehandlung
 - wie im Vorfeld erfolgt eine koordinierte Handhabung der gesicherten Register
- Unterbrechung beliebiger Betriebsmodi möglich
 - Unterbrechung auch des privilegierten Modus

Interne Unterbrechungen durch Interrupts

Auch **interne Unterbrechungen** im **Programmablauf** sind möglich. Damit gemeint sind interne **Unterbrechungen** der **Befehlsausführung** durch *Exceptions*.

Bei **bestimmten Fehlersituationen** (z.B. Division durch Null) **unterbricht** Prozessor **laufende Befehlsbearbeitung** und führt eine **vorher definierte Befehlsfolge** aus (ähnlich wie bei externen Unterbrechungen).

Zusammenfassung

ZUSAMMENFASSUNG

Wir haben uns im heutigen Tutorium mit den Betriebssystemen an sich beschäftigt.

Zur Besprechung der Ressourcenverwaltung von Betriebssystemen haben wir zudem gesehen welche Klassifikationen es gibt, um Betriebssysteme einzuordnen und deren Funktion zu beschreiben. Dabei haben wir die unterschiedlichen Architekturen kennengelernt die in verschiedenen Betriebssystemen zur Anwendung kommen können und deren Grundkomponenten aufgeklärt.

Neben der Klassifizierung haben wir zudem die einzelnen Privileg-Level moderner Betriebssysteme besprochen und Möglichkeiten aufgezeigt, zwischen einzelnen Level zu agieren. Dabei wurden auch Möglichkeiten der Prozessunterbrechung durch eine der wichtigen Bestandteile moderner Hardware, den Interrupts kennengelernt.

Vielen Dank



**HOCHSCHULE
MITTWEIDA**
University of Applied Sciences

Prof. Ronny Bodach

Hochschule Mittweida | University of Applied Sciences
Technikumplatz 17 | 09648 Mittweida
Fakultät Angewandte Computer- und Biowissenschaften

T +49 (0) 3727 58-1011
F +49 (0) 3727 58-21011
bodach@hs-mittweida.de
www.cb.hs-mittweida.de

Haus 8 | Richard-Stücklen Bau | Raum 8-205
Am Schwanenteich 6b | 09648 Mittweida

Felix Fischer

Hochschule Mittweida | University of Applied Sciences
Technikumplatz 17 | 09648 Mittweida
Fakultät Angewandte Computer- und Biowissenschaften

fische11@hs-mittweida.de
www.cb.hs-mittweida.de

[hs-mittweida.de](https://www.hs-mittweida.de)