

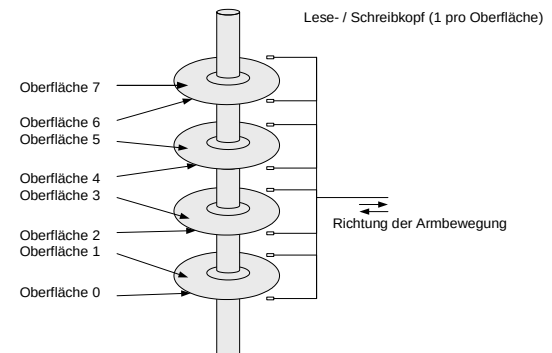
MAS: Betriebssysteme

Geräte- und Dateiverwaltung

T. Pospíšek

Gesamtüberblick

1. Einführung in Computersysteme
2. Entwicklung von Betriebssystemen
3. Architekturansätze
4. Interruptverarbeitung in Betriebssystemen
5. Prozesse und Threads
6. CPU-Scheduling
7. Synchronisation und Kommunikation
8. Speicherverwaltung
- 9. Geräte- und Dateiverwaltung**
10. Betriebssystemvirtualisierung



Zielsetzung

- Die Grundlagen der Geräteverwaltung am Beispiel der Festplatte und der Dateiverwaltung anhand praktisch relevanter Filesysteme verstehen
- Konzepte von Storage Systemen kennenlernen

Überblick

1. Geräteverwaltung

- Überblick
- Treiber, Gerätetypen, Gerätemodelle,...
- Fallbeispiel: Unix

2. Dateisysteme

- Grundlagen
- Fallbeispiel: Unix
- Fallbeispiel: Windows

3. Storage Systeme

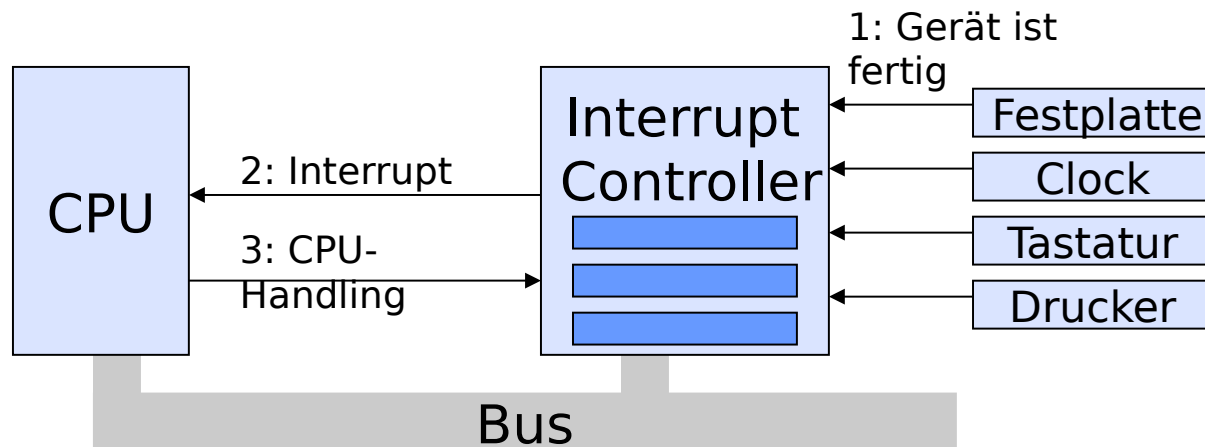
- RAID
- SAN, NAS

I/O-Manager

- Die Geräteverwaltung (**I/O-Manager**) dient der optimierten Verwaltung von externen Geräten zur Ein- und Ausgabe
 - Festplatten, Wechselmedien
 - Tastatur und Bildschirm
 - Netzwerkanbindung
- Schnittstelle zwischen den Geräten und dem Rest des Betriebssystems
- Aufgaben im Einzelnen:
 - Ausgabe von Kommandos an externe Geräte
 - Interruptbearbeitung
 - Fehlerbehandlung

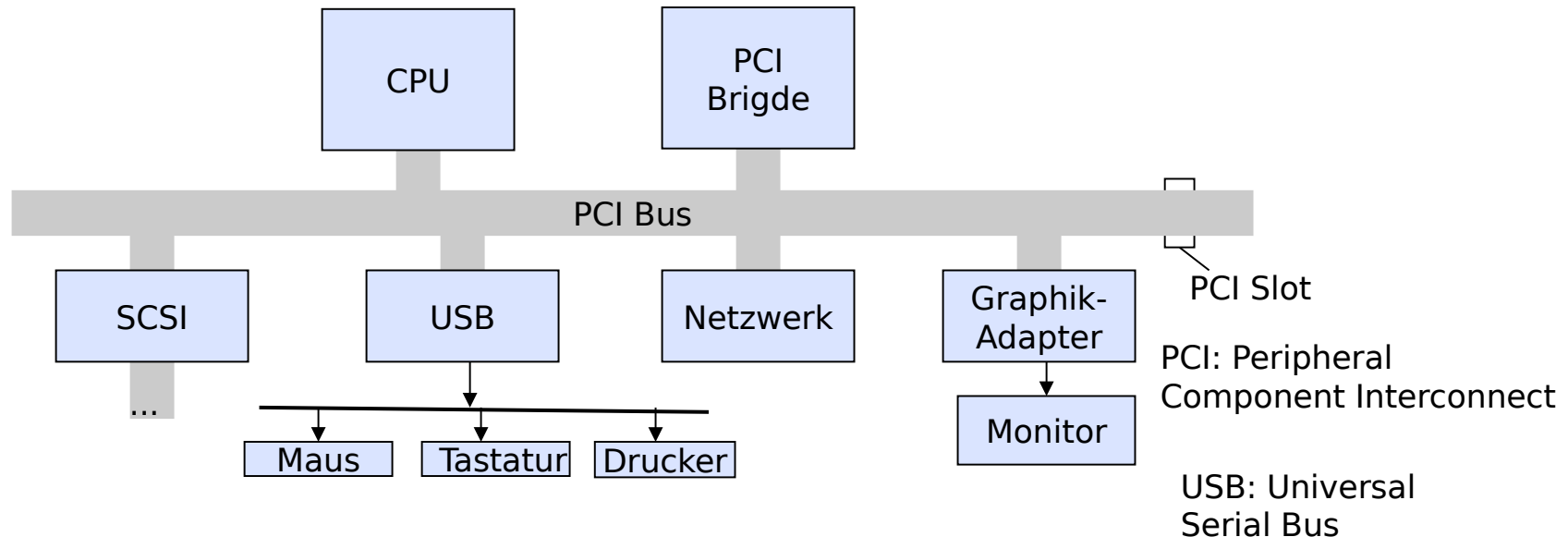
Interrupt-Bearbeitung: Interrupt-Controller

■ Auffrischung



Bussysteme am Beispiel des Pentium-Bussystems

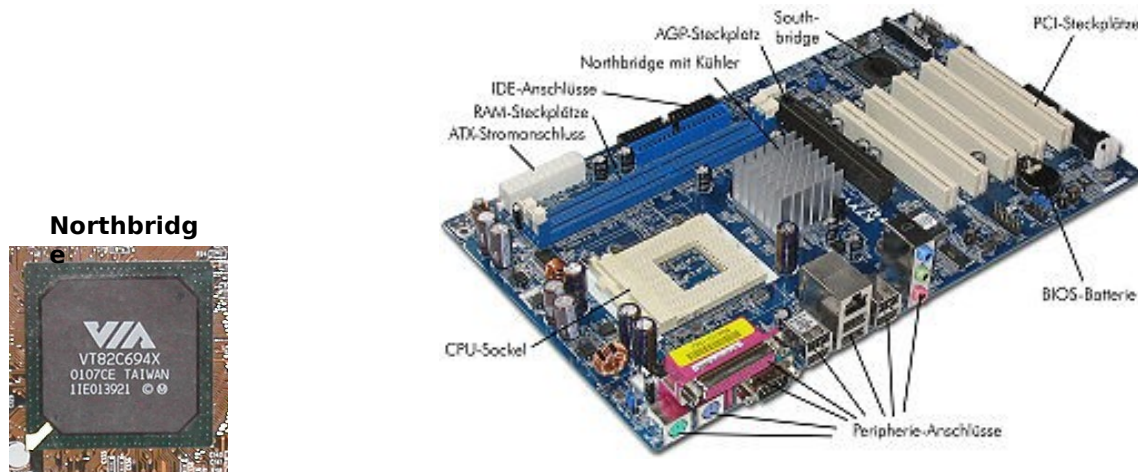
- Geräte sind über Bussysteme (**Datenverbindung**) an die CPU angebunden
- In heutigen Rechnern gibt es eine **Hierarchie** von Bussystemen



Vgl. Tanenbaum

Bussysteme am Beispiel des Pentium-Bussystems

- Northbridge und Southbridge (zwei Chips auf der Hauptplatine = Motherboard eines PCs) → Chipsatz:
 - Northbridge verbindet CPU mit Hauptspeicher (breitbandig)
 - Southbridge schließt Bussysteme an
 - Southbridge und Northbridge kommunizieren zum Beispiel über PCI



Hauptplatine, Quelle: Wikipedia



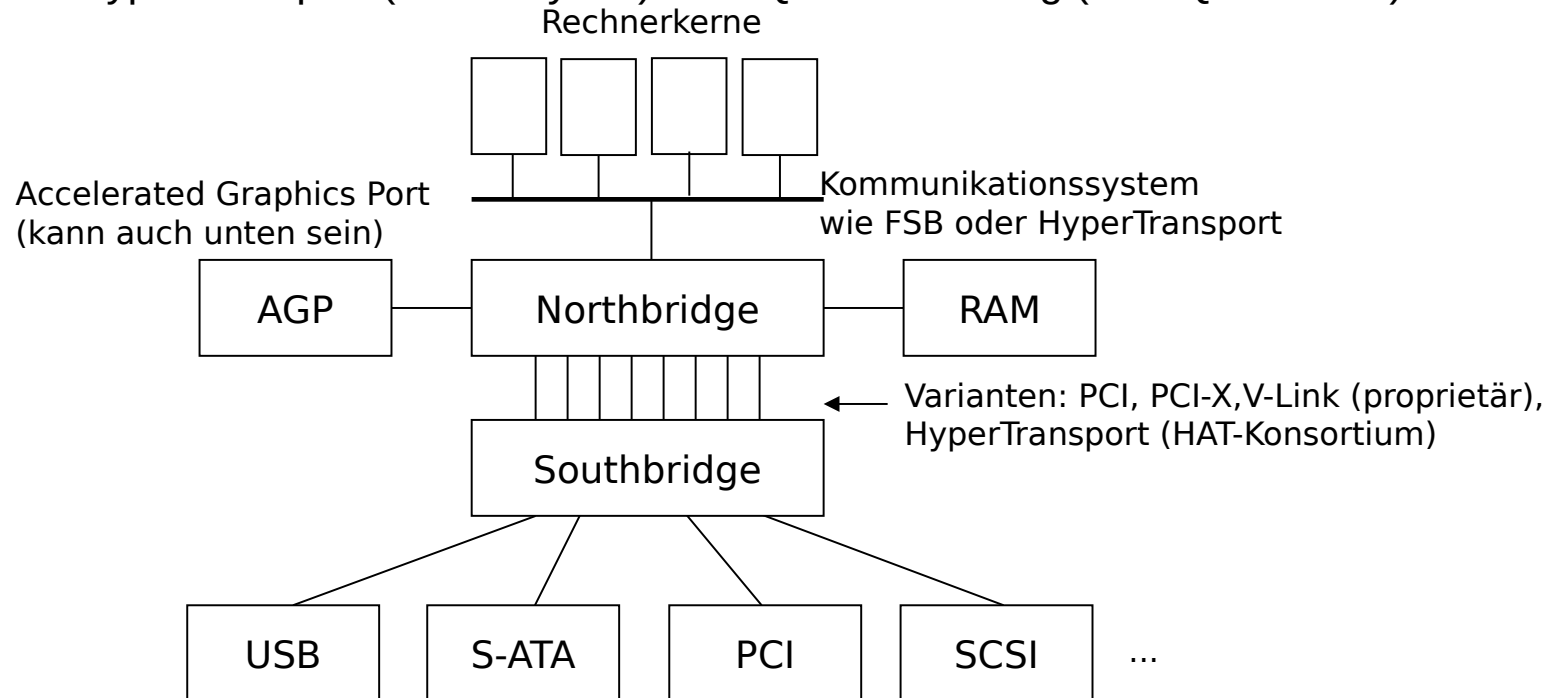
Northbridge



Southbridge

Verbindungen im Chipsatz

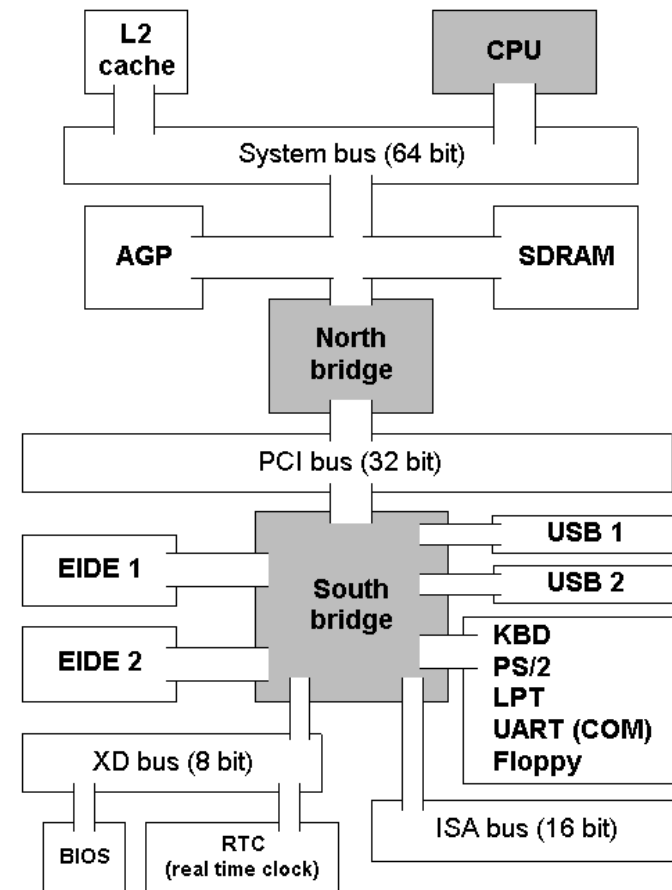
- Verbindung CPU – Northbridge über FSB
 - Systembus alter Prägung gibt es nicht mehr seit CPUs zu schnell sind
 - FSB (Front Side Bus) oder schnellere Verbindungen wie eine HyperTransport (> 50 Gbyte/s) oder QPI-Verbindung (Intel Quick Path)



Ergänzung: Northbridge, Southbridge

Beispiel eines modernen Chipsatzes

- North- und Southbridge sind "Router"
- Routing von Traffic von Bus zu Bus
- Die Northbridge bedient den umfangreichen Traffic zwischen CPU und RAM
- Die Southbridge bedient die anderen Routen (Bussysteme) zu den externen Geräten
- Verbindung über PCI-Bus

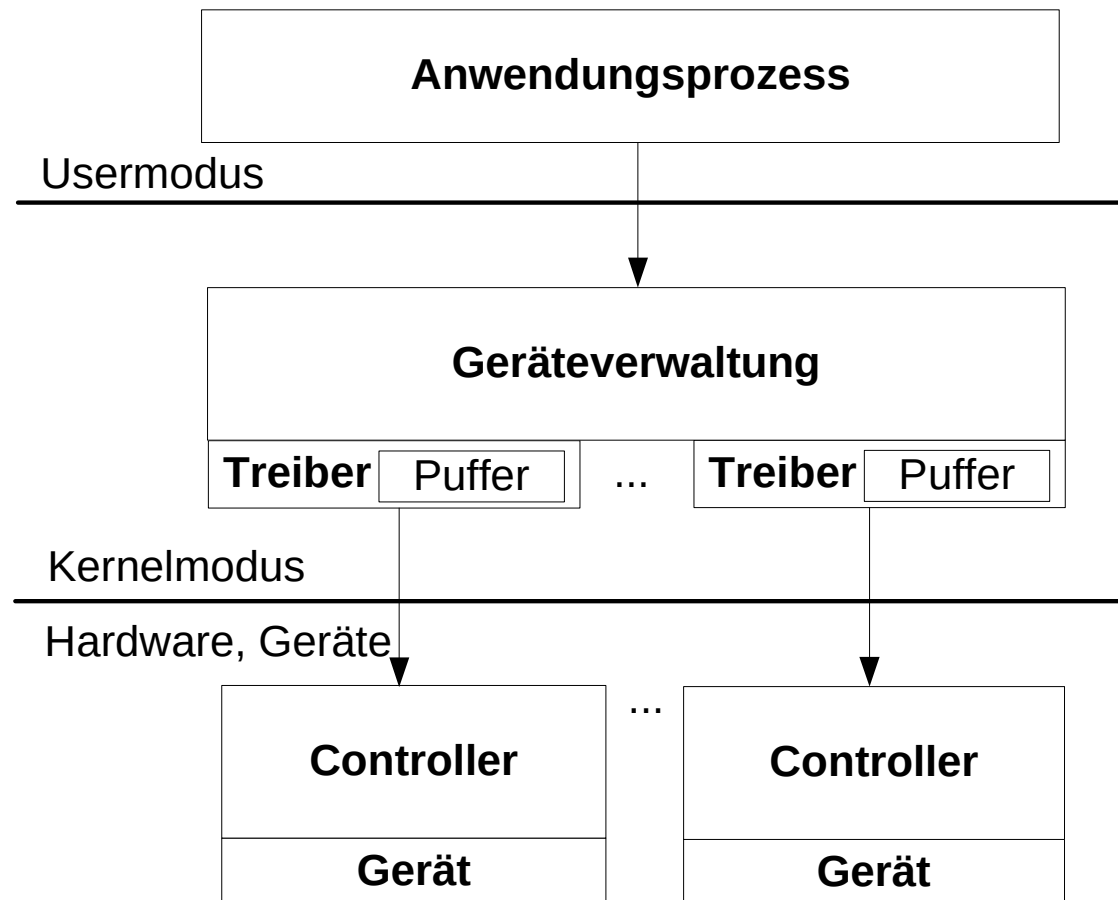


Typische Datenraten

- Sehr unterschiedliche Transferraten [Bit/s] bzw. Datenraten [Byte/s] für die einzelnen Geräte

Gerät	Transferrate bzw. Datenrate
Keyboard	ca. 10 Byte/s
Maus	ca. 100 Byte/s
Laserdrucker	ca. 100 KByte/s
USB3	5 GBit/s
10Gbit Ethernet	1,25 GByte/s
Serial Attached SCSI	750 MByte/s
Tape	ca. 320 MByte/s
PCI Express	32 GBit/s
IDE-Festplatten mit ATA	ca. 130 MByte/s
Festplatten mit S-ATA	Bis zu knapp 2 GByte/s
Bus-Kommunikationssystem wie AMD HyperTransport oder Intel QPI	50 GBit/s bis über 100 GBit/s

Schichtung des I/O-Systems



Treiber

- Treiber sind gerätespezifische Programmteile
- Treiber werden in eigenen Modulen meist durch den Gerätehersteller realisiert
- Die wichtigsten Aufgaben eines Treibers:
 - Initialisierung und Bekanntgabe des Geräts
 - Datenübertragung von und zu einem Gerät
 - Logisches Programmiermodell auf gerätespezifische Anforderungen übersetzen
 - Pufferung von Daten
 - Interruptbearbeitung
 - Koordination der nebenläufigen Zugriffe

Gerätearten (1)

- **Zeichenorientierte** Geräte
 - Information wird in einzelnen Zeichen gespeichert und gelesen
 - Einzelne Zeichen sind nicht adressierbar, es wird in Streams gelesen bzw. gespeichert
 - Beispiele: Drucker, Maus, Netzwerkkarten
- **Blockorientierte** Geräte
 - Information wird in Blöcken von mind. 512 Bytes gespeichert und gelesen
 - Blöcke sind adressierbar, Zugriff erfolgt auf Blöcke
 - Beispiele: Festplatten, Tapes
- **Sonstige** Geräte wie Clocks (Zeitgeber)

Gerätearten (2)

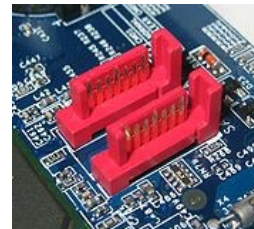
■ Geräte mit wahlfreiem Zugriff

- Adressierungsinformation notwendig
- Beispiel: Plattenspeicher
 - Adressierung über
 - Spurnummer
 - Sektornummer
 - Zylindernummer

Festplatte, Quelle: Wikipedia



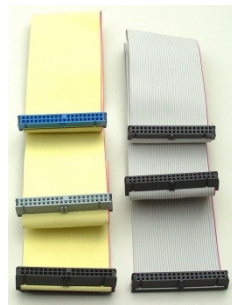
S-ATA-Anschluss, Quelle: Wikipedia



Alte IBM-Festplatte, 1979, Quelle: Wikipedia



IDE-Schnittstelle, Quelle: Wikipedia



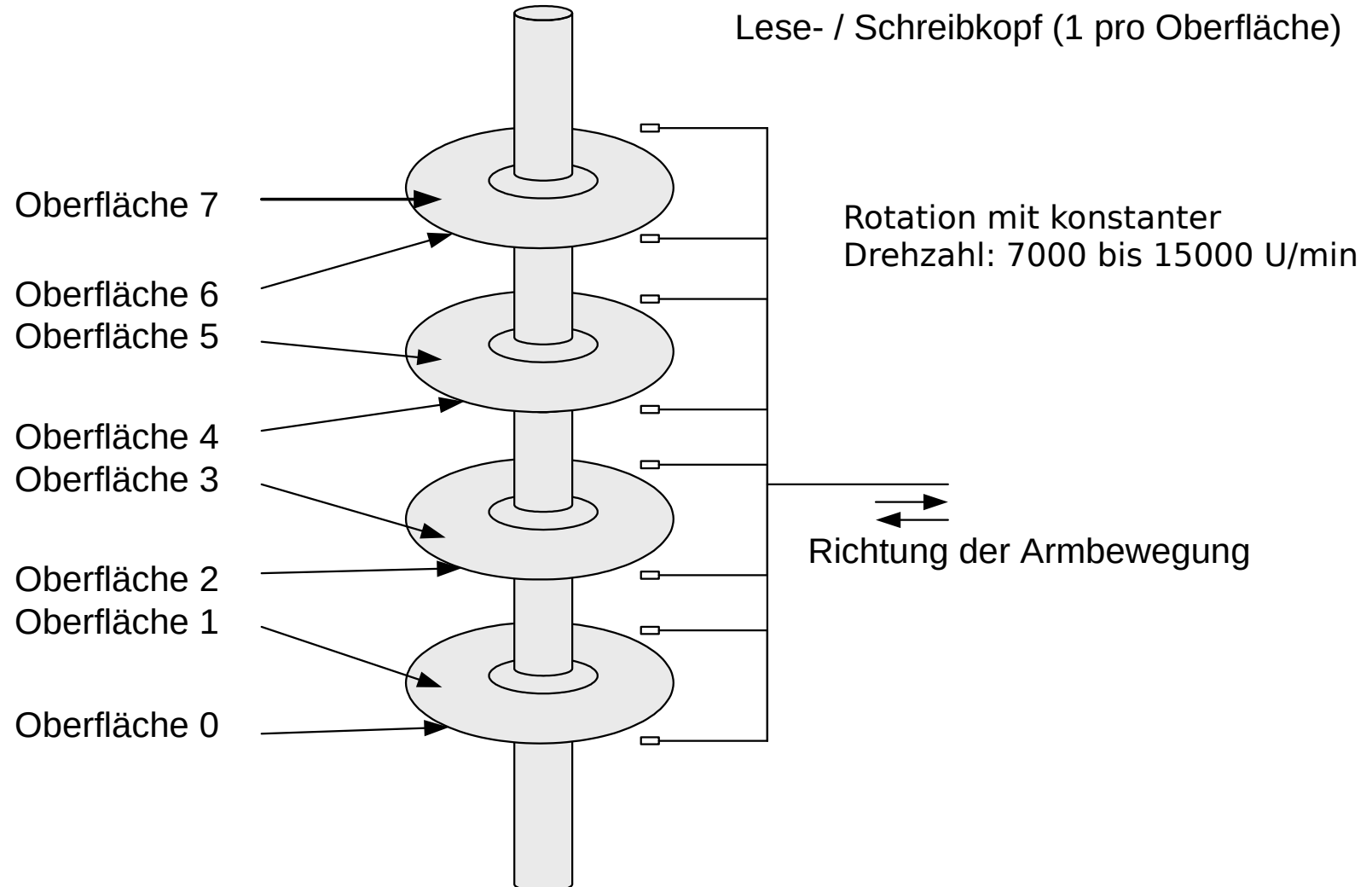
ISA-Steckplatz, Quelle: Wikipedia



Gerätearten: Beispiel Festplatte (1)

- Festplatten bestehen aus **Kunststoff-** oder **Aluminiumscheiben**, die mit einer hauchdünnen Magnetschicht überzogen sind
- **Schreib-/Leseköpfe** sind mit kleinem Elektromagneten ausgestattet
- Daten sind als Blöcke fester Größe in **Sektoren** gespeichert, die in **Spuren** zu finden sind
- Anordnung in sog. Zylindern
 - Bei übereinander liegenden Platten alle Spuren mit gleicher Nummer, auf welche die Schreib-/Leseköpfe gleichzeitig positionieren können

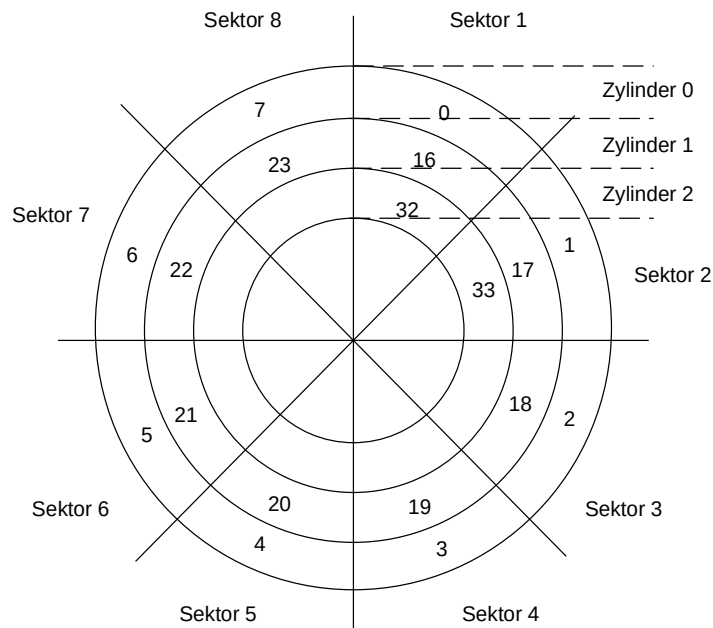
Gerätearten: Beispiel Festplatte (2)



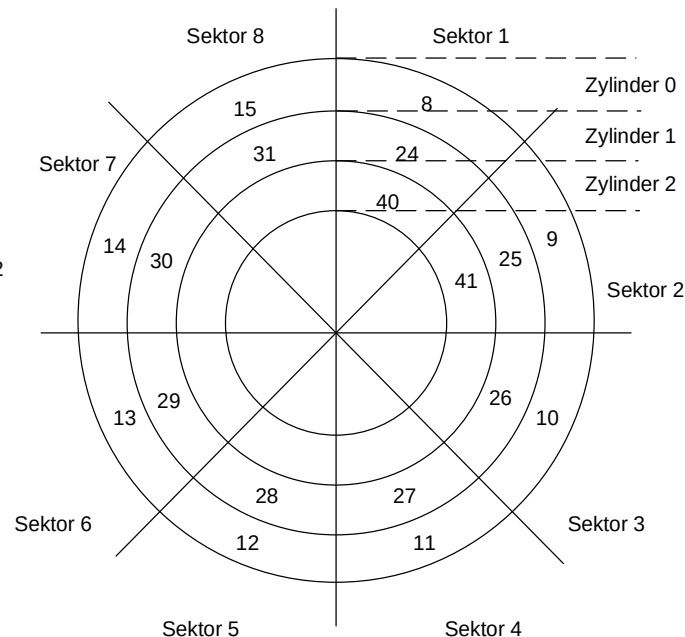
Gerätearten: Beispiel Festplatte (3)

- Plattenaufteilung: Spuren, Zylinder, **physikalische** Sektoren (Kuchenstücke, Kreissektoren)
- Alle Spuren untereinander nennt man Zylinder (eine Position des Kopfes)
- Durchnummerierung **logischer** Sektoren entlang der Spuren → keine mechanische Aktion erforderlich

Seite 0

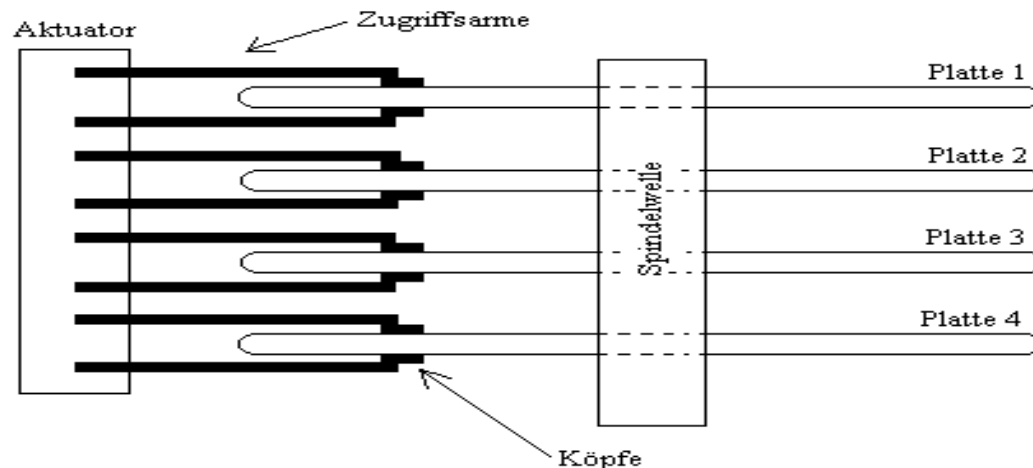


Seite 1



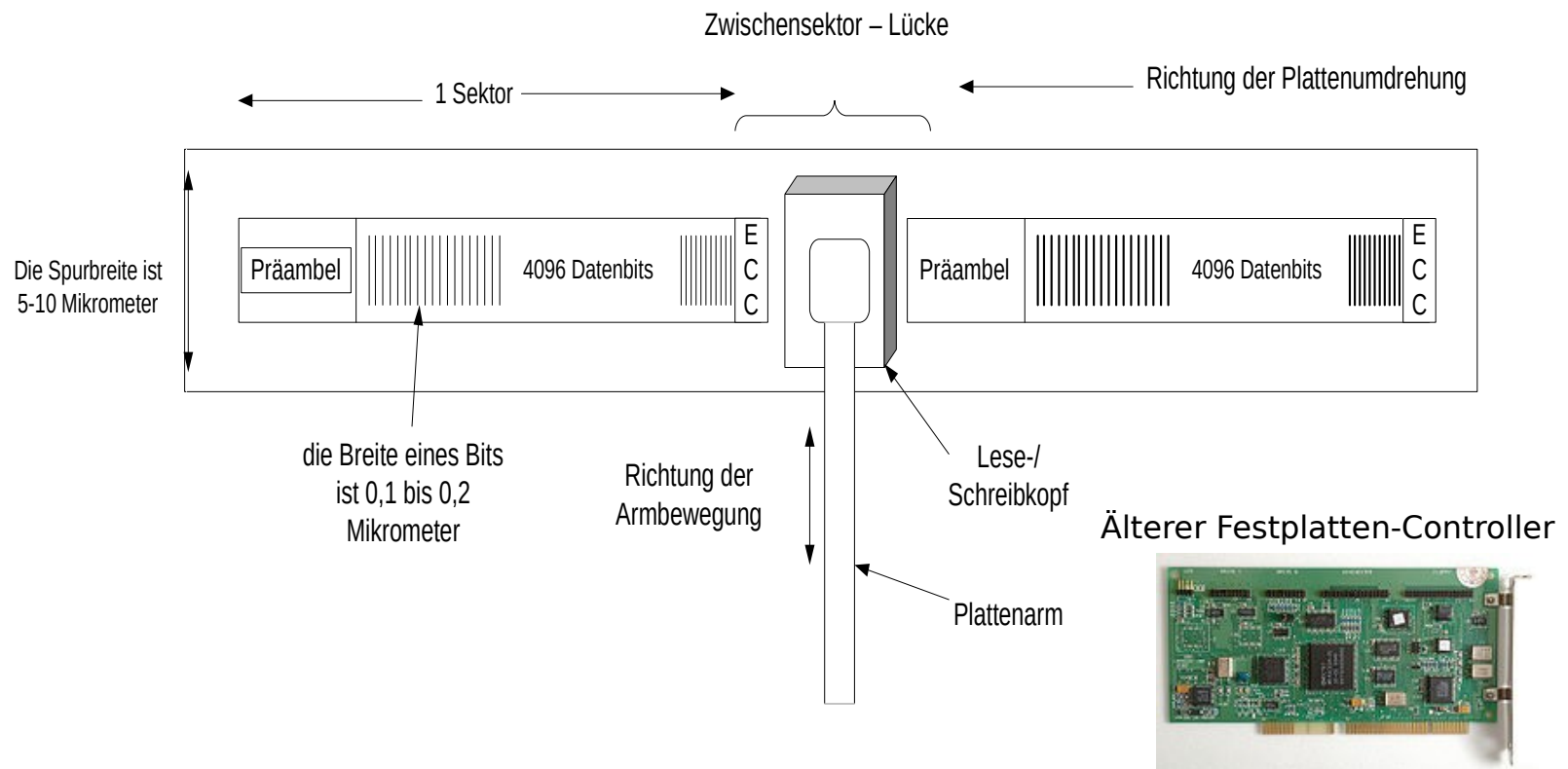
Gerätearten: Beispiel Festplatte (4)

- Üblicher Plattendurchmesser 3,25 oder 2,5 Zoll oder kleiner (1,8 Zoll)
- Zwischen 2 und 8 Platten, Beispiel: 135.000 Spuren/Zoll
- Kapazität: Bei 39 Sektoren, 761 Zylinder und 8 Köpfen und 512 Byte großen Sektoren:
 - $39 \times 761 \times 8 \times 512 = 121.565.184$ Byte
- Kapazität: mehrere TiB
 - Hat sich in den letzten 30 Jahren mehr um als das 200-fache erhöht
 - Zugriffszeiten aber nicht!



Gerätearten: Beispiel Festplatte (5)

- Betrachtung zweier Sektoren aus einer Festplattenspur
- Sektoreninhalt: Präambel, Nutzdaten (hier 4096 Bit) und ECC



Gerätearten: Beispiel Festplatte (6)

- Das Betriebssystem kennt nur logische Sektoren in durchnummerierter Reihenfolge (sog. **LSN** = Logical Sector Number)
- Eine Übersetzung der logischen Adressen in phys. Adressen ist erforderlich

Einfaches lineares Adressierungsmodell
mit den Speicheradressen 0..N



Adressierungslogik der Festplatte
(Laufwerk-, Zylinder-, ...,
Sektornummer)

- Folge 1 bis N von Blöcken, die eindeutig durch ein **Quadrupel der Form (g, z, h, s)** adressierbar sind, mit
 - g: Laufwerk
 - z: Zylindernummer
 - h: Nummer des Schreib-Lesekopfes
 - s: Sektornummer

Gerätearten: Beispiel Festplatte (7)

- Scheduling-Strategie
 - Ein Plattentreiber muss die Lese- und Schreibaufträge möglichst optimal und fair bearbeiten
- Bekannte Scheduling-Strategien
 - **FCFS** (First Come First Serve)
 - Einfach und fair: Alle Aufträge in der Reihenfolge ausführen, in der sie ankommen
 - **SSTF** (Shortest Seek Time First)
 - Aufträge in näherer Umgebung der aktuellen Schreib-/Lesekopf-Position ausführen
 - Vermeidung von Kopfbewegungen
 - Wie bei SJF-Prozessverwaltung besteht das Problem des Verhungerns von Aufträgen

Gerätearten: Geräte mit serielltem Zugriff

■ Geräte mit serielltem Zugriff

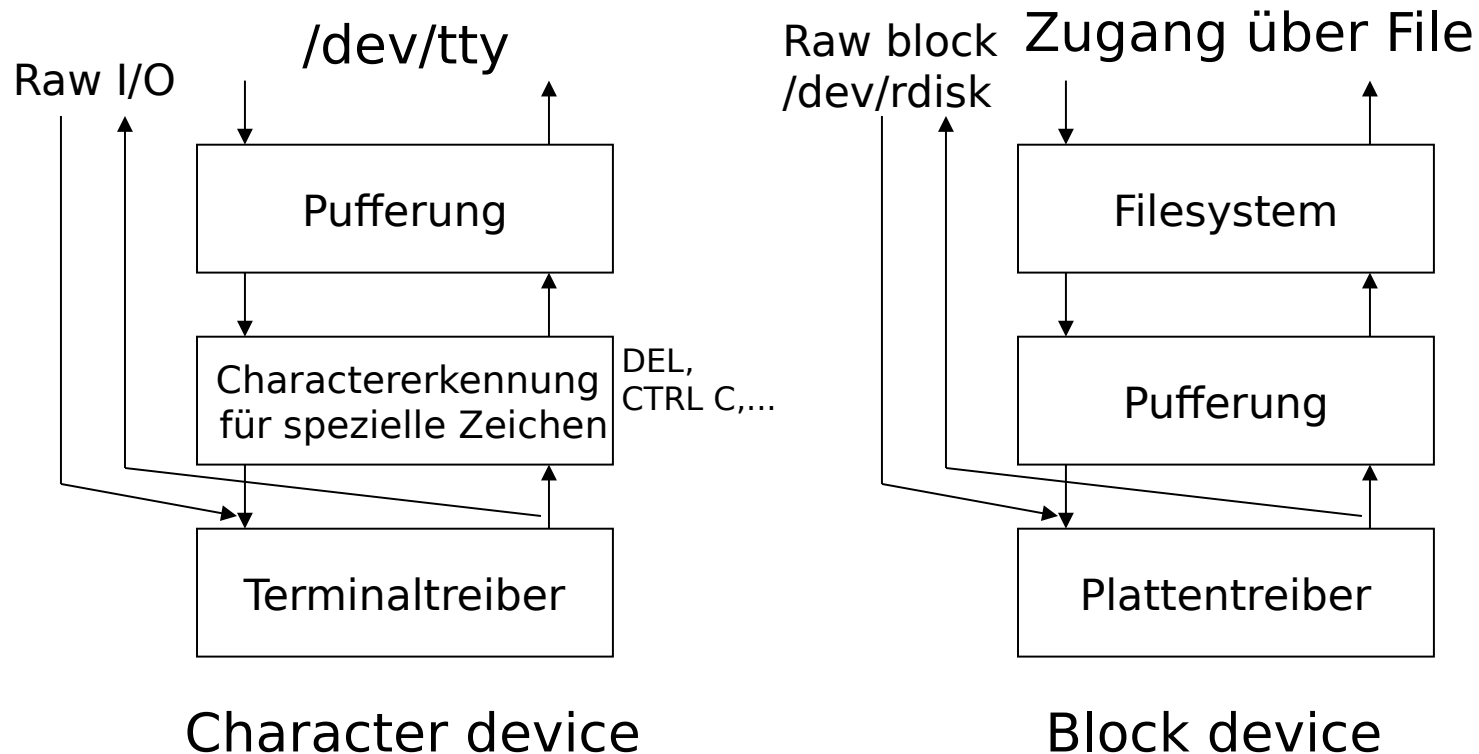
- „Serielle Geräte“
- Keine Adressierungsinformation notwendig
- Lesen bzw. Schreiben geht Zeichen für Zeichen
- Beispiele
 - Maus
 - langsame Zeilendrucker
 - Tastatur
- Es wird eine Flusssteuerung benötigt (XON/XOFF)

Fallbeispiel Unix: Geräte als spezielle Dateien

- „Jedes“ Gerät hat unter Unix ein „special file“ zugeordnet, über das es wie eine Datei behandelt werden kann
 - Verzeichnis z.B. **/dev**
 - Genaue Namen sind abhängig vom Unix-Derivat
 - Operationen (System Calls): open, read, write,...
- Bei den sog. special files unterscheidet man:
 - **Block special file**, z.B. /dev/hd1 (Festplatte)
 - **Character special file**, z.B. /dev/tty (Bildschirm, Tastatur)
- Beispiel:
 - Kommando *cp file /dev/lp* kopiert eine Datei namens *file* direkt auf den Drucker

Fallbeispiel Unix: Streams

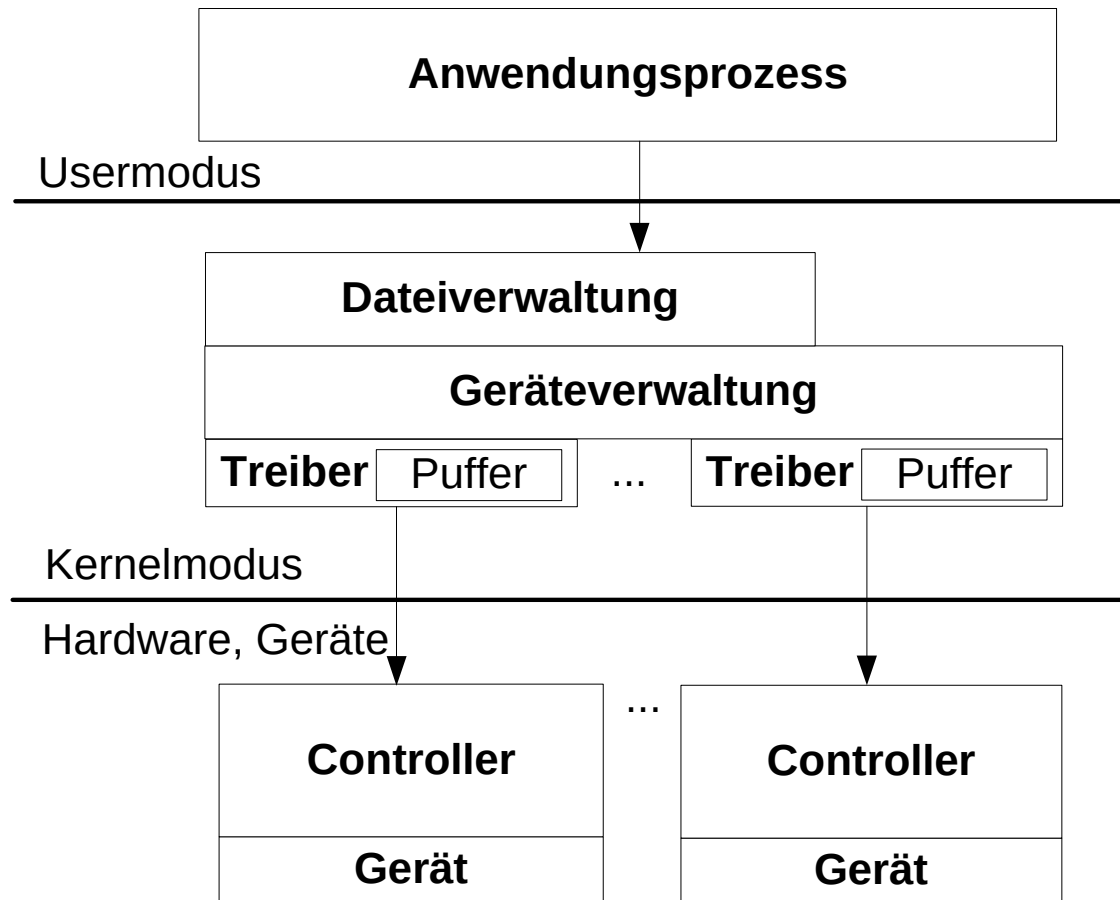
- Streams-orientiertes Modell bei System V:
Treiberschichten (Filter) können in den Stream eingebaut werden



Überblick

1. Geräteverwaltung
 - Überblick
 - Treiber, Gerätetypen, Gerätemodelle,...
 - Fallbeispiel: Unix
2. **Dateisysteme**
 - Grundlagen
 - Fallbeispiel: Unix
 - Fallbeispiel: Windows
3. Storage Systeme
 - RAID
 - SAN, NAS

Einordnung in die Schichten



Dateien

- Informationen müssen von Betriebssystemen **persistent** gespeichert werden können
- **Dateien** sind ein abstrakter Mechanismus zur Speicherung und zum Wiederauffinden von Informationen
- Der Teil des Betriebssystems, der sich mit Dateien und deren Organisation befasst, wird als Dateiverwaltung bezeichnet
- Zu jeder Datei sind bestimmte Informationen zu verwalten

Informationen zu Dateien

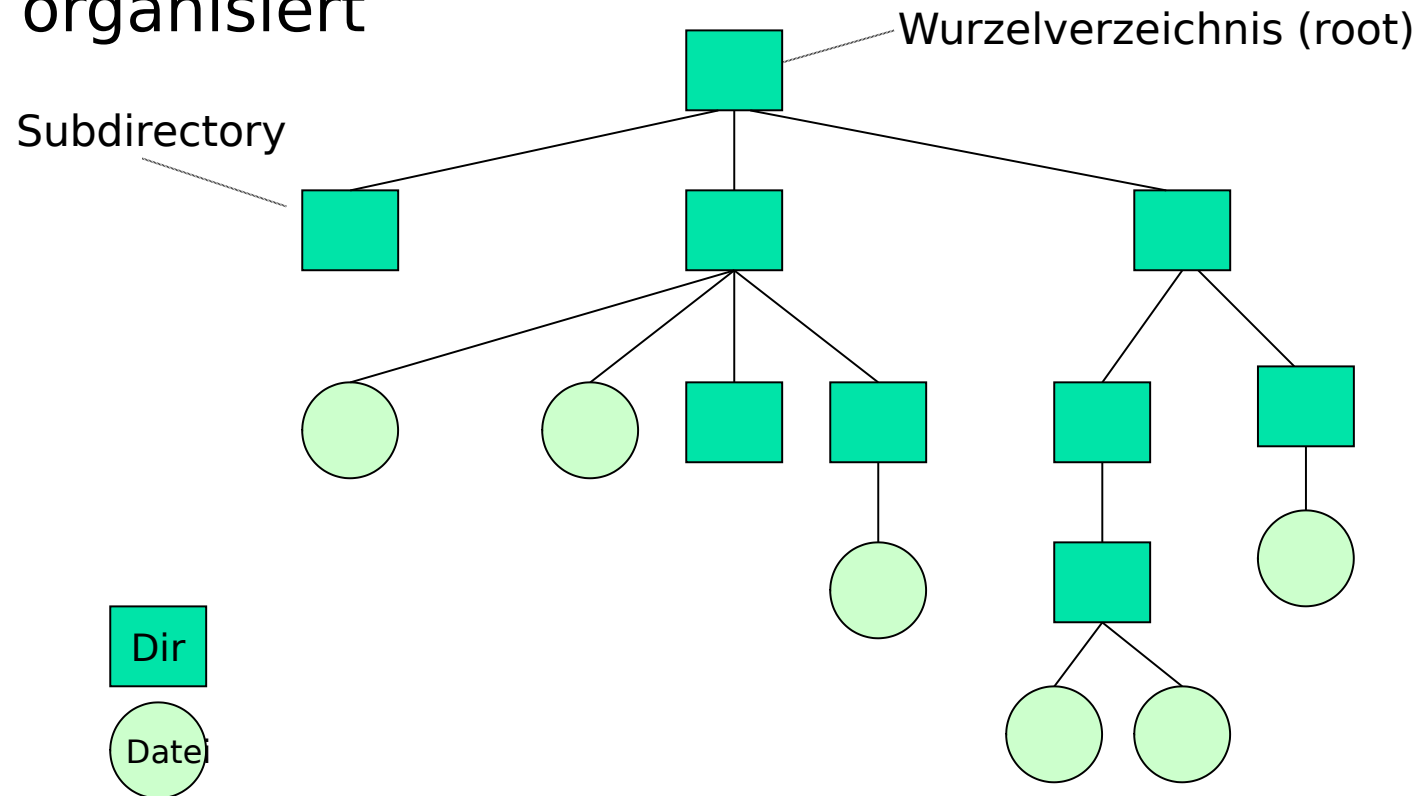
- Dateien haben einen Dateinamen und einen Inhalt
- Jede Datei besitzt einen Eintrag mit Informationen über:
 - Namen der Datei
 - Typ der Datei: Normal- oder Katalogdatei
 - Länge der Datei in Bytes
 - Blöcke, aus denen die Datei besteht
 - Zugriffsrechte zu dieser Datei
 - Passwörter zum Schutz der Datei
 - Statistikdaten: z.B.: Datum letzte Änderung
 - ...

Dateisysteme

- Dateien werden in Dateisystemen (file system) verwaltet
- Dateisysteme oft hierarchisch organisiert
 - Dateigruppen (Dateiverzeichnis) mit Verwaltungsinformation
 - Dateien (mit den eigentlichen Daten)
- Strukturierung mit Unterverzeichnissen (Subdirectory)
- Jedes (Sub)directory enthält Einträge zu weiteren Katalogen oder *normalen* Dateien
- Beispiele:
 - Unix verfügt über ein systemweites Dateisystem
 - MS-DOS verwaltet für jedes Gerät ein unabhängiges Directory

Hierarchische Verzeichnissysteme

- Hierarchische Verzeichnissysteme beliebiger Tiefe sind heute „State of the Art“
- Directories sind in einer Baum-Struktur organisiert



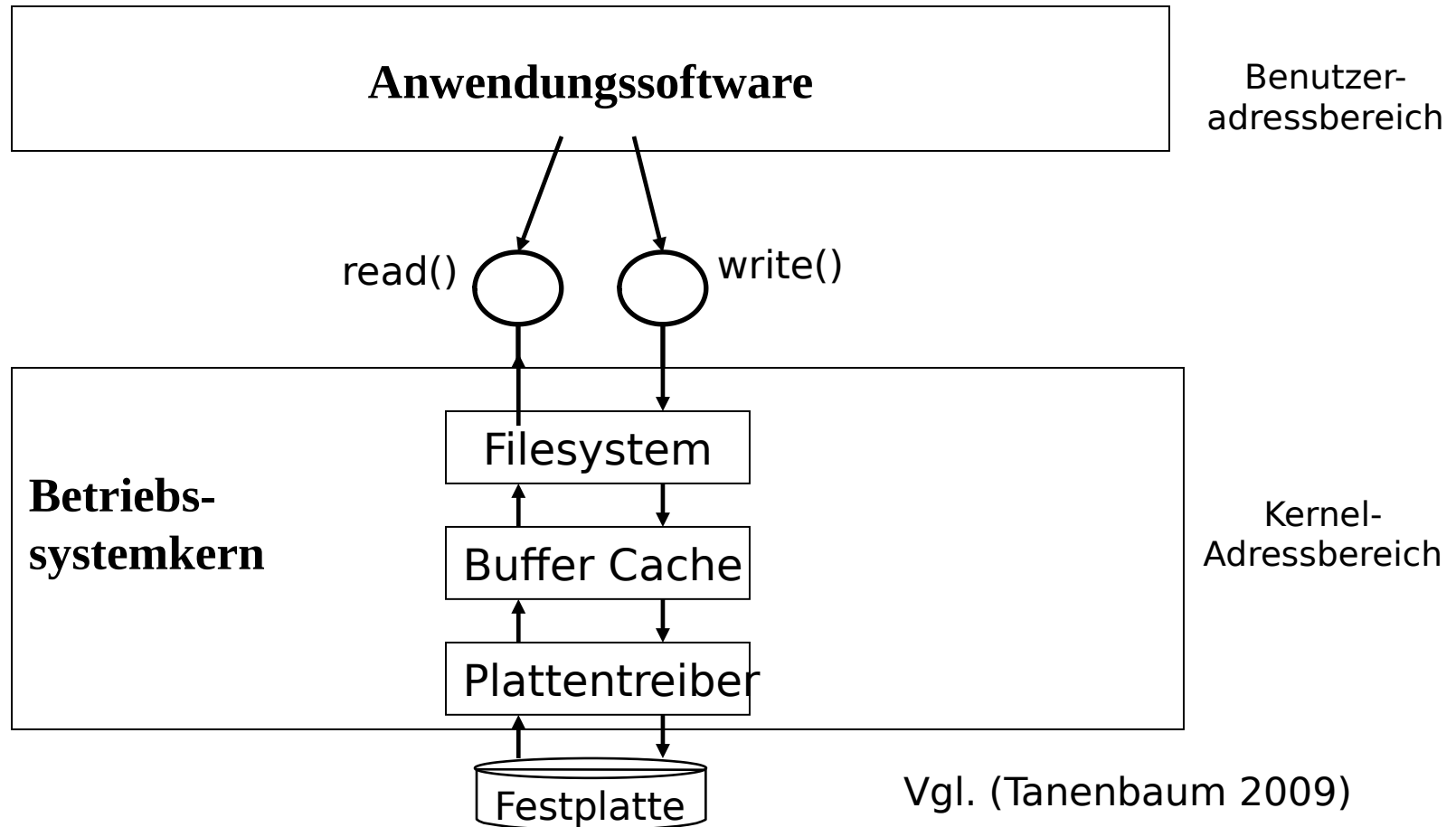
Operationen auf Dateien

- Systemdienste zur Verwaltung von Dateien:
 - **Erzeugen** eines Katalogeintrages (Dateiname)
 - **Löschen** des Katalogeintrags u. Dateiinhalts
 - **Kopieren** von Dateien
 - **Ändern** des Dateinamens oder anderer Katalogeinträge
 - **Übertragen** eines Katalogeintrags in einen anderen Katalog (Directory)

Blockverwaltung

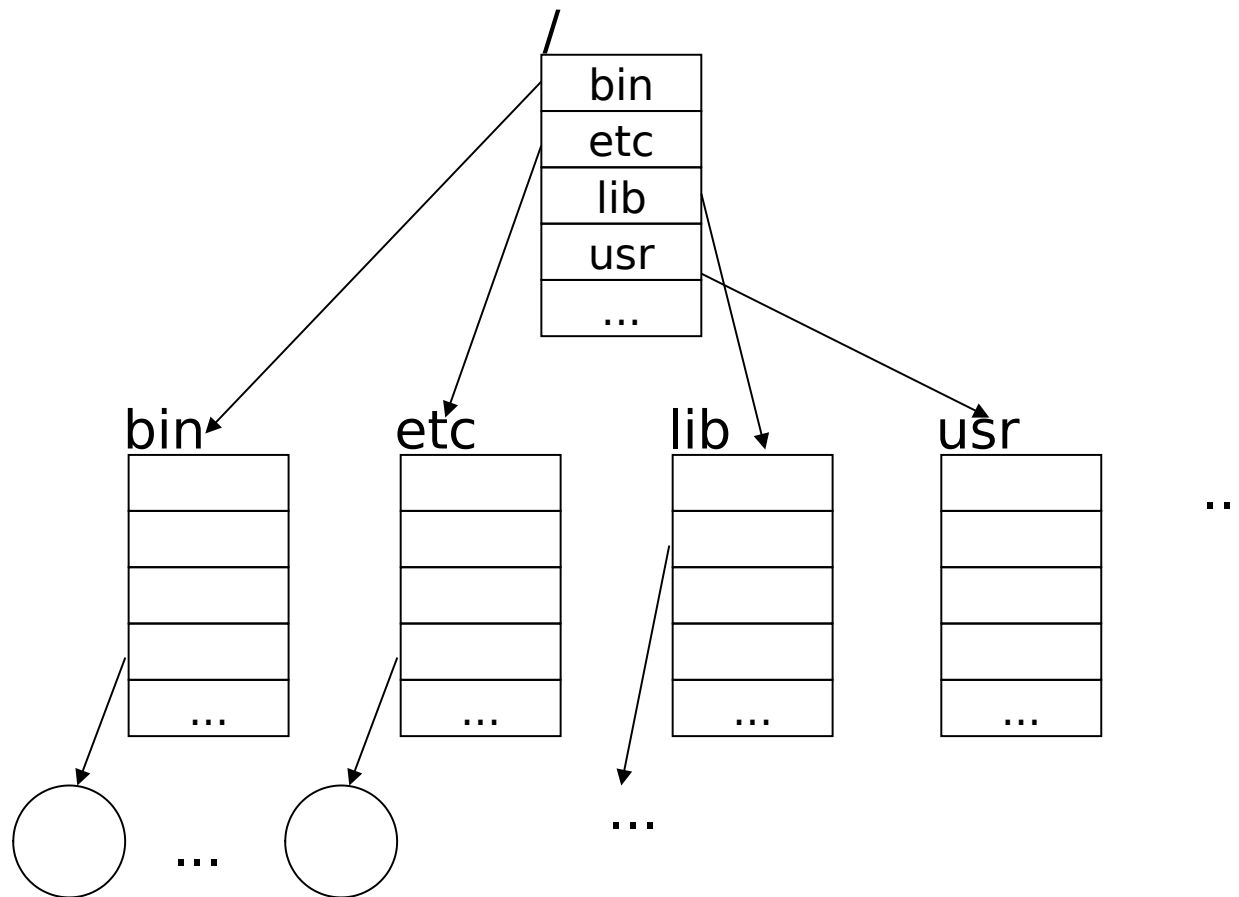
- Die Bearbeitung von Dateien ist blockweise oder zeichenweise möglich
- Jede gefüllte Datei hat eine Liste mit von ihr belegten Blöcken
- Der Katalog enthält Hinweise auf die Blöcke, aus denen die Dateien bestehen
- Das Dateisystem verwaltet zudem:
 - Eine **Pseudodatei** bestehend aus allen **freien** Blöcken eines Datenträgers
 - Eine **Pseudodatei** mit einer Liste aller **unzuverlässigen** Blöcke

Fallbeispiel: Unix, Einordnung von Filesystemen



Fallbeispiel: Unix, Verzeichnisstruktur

- Unix verwendet ein beliebig tiefes, hierarchisches Dateisystem



Fallbeispiel: Unix Verzeichnisstruktur

- Die Wurzel des Baums heißt **root**; das **root-directory** wird mit „/“ abgekürzt. (bei MS-DOS „\\“)
- Darunter befinden sich die **home-directories** der einzelnen User (/home)
- Die inneren Knoten des Baumes sind **subdirectories**, die Blätter des Baumes sind die eigentlichen Files
- Geräte (Drucker, Terminals, Maus etc.) werden logisch wie Dateien behandelt (**special files**)
- Für die Organisation des Unix-Baumes haben sich gewisse Konventionen eingebürgert

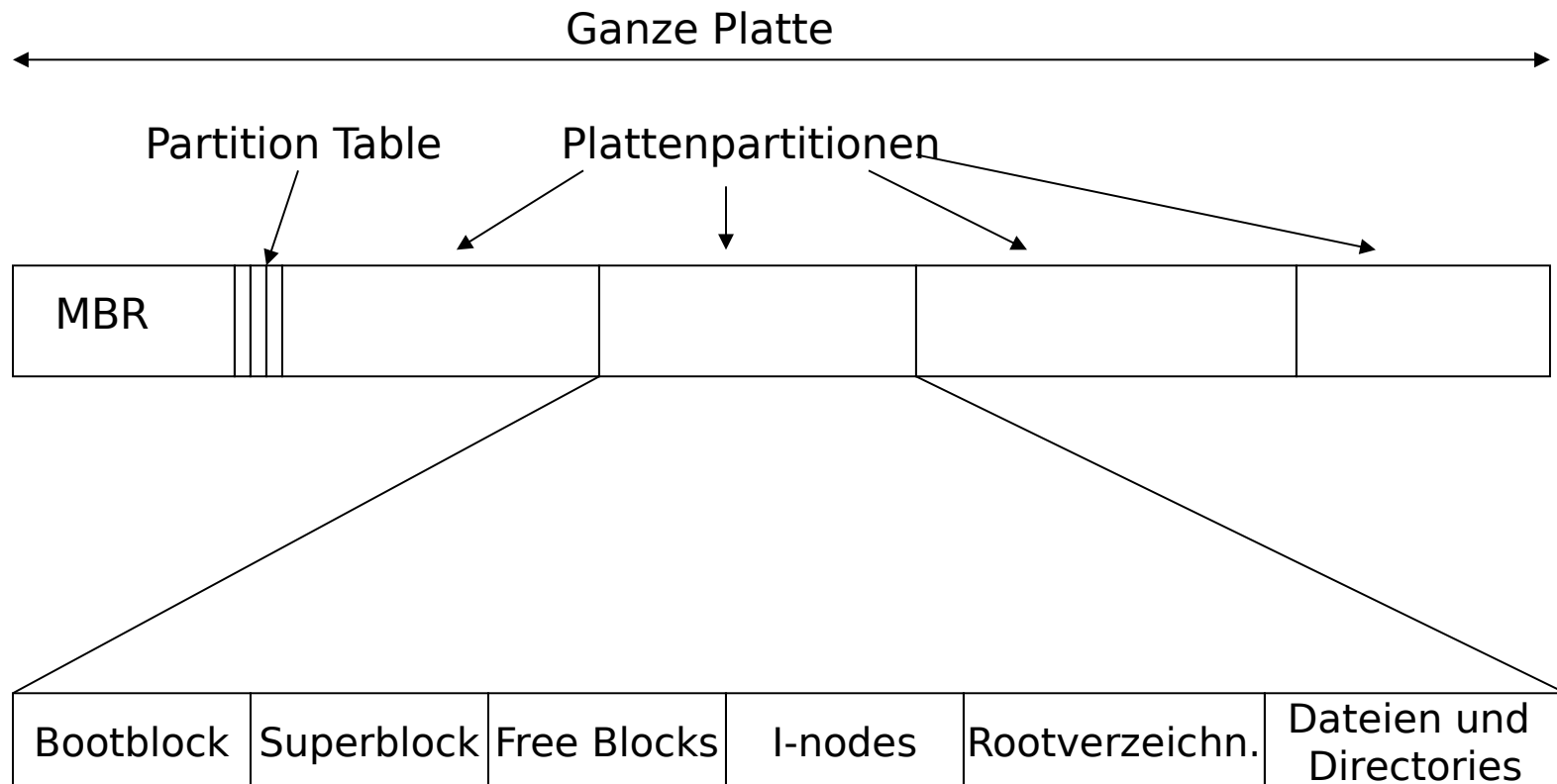
Fallbeispiel: Unix, Verzeichnisstruktur

- Unter den Kindern der Wurzel finden sich meist subdirectories mit folgender Bedeutung:
 - bin: Ausführbare Systemdateien (binär)
 - dev: Gerätedateien (devices)
 - lib: Bibliotheken (libraries)
 - usr: Systemdateien (ursprüngl.: user)
- Externe Dateisysteme, also physikalische Datenträger wie Harddisk, CDROM, Floppy,... können unabhängige Filesysteme enthalten
 - Diese kann man mit dem Befehl **mount** an beliebiger Stelle in den Unix-Dateibaum einklinken
 - Danach gibt es keine Unterscheidung mehr zwischen Files auf der Diskette und anderen block special files
 - Mit **umount** wird das Subdirectory entfernt

Fallbeispiel: Unix, Implementierung des Dateisystems

■ Beispiel eines Dateisystem-Layouts einer Festplatte

Vgl. Tanenbaum



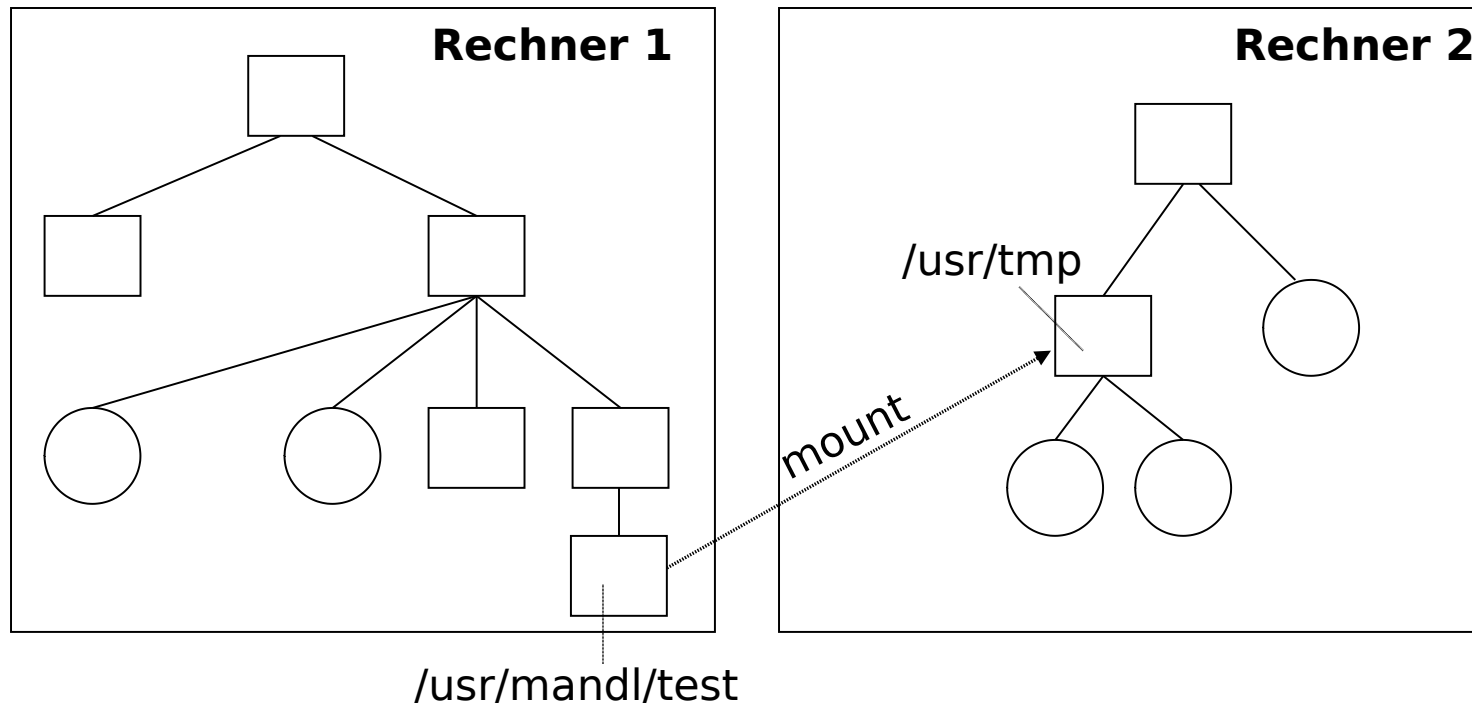
Fallbeispiel: Unix, Implementierung des Dateisystems

- **MBR** = Master Boot Record auf Sektor 0 zum Booten des Rechnersystems
- Am Ende des MBR liegt die **Partition Table**: Eine Platte kann i.d.R. in mehrere Partitionen eingeteilt werden
- **Bootblock** wird bei Hochfahren gelesen und ausgeführt
- **Superblock** enthält Verwaltungsinformationen zum Dateisystem (Anzahl der Blöcke,...)
- **Free Blocks** (z.B. Bitmap) gibt die freien Blöcke des Dateisystems an
- **Rootverzeichnis** enthält den Inhalt des Dateisystems (nach der Wurzel)
- **I-nodes**: I-nodes sind Einträge im Inhaltsverzeichnis des Dateisystems

The diagram illustrates the mapping from an i-node to logical block numbers. On the left, the 'Dateikopf (i-node)' is shown as a vertical stack of slots. Slots 10 and 13 are specifically labeled. Slot 10 points directly to logical block 9 ('direkt'). Slot 13 points to a 'doppelt indirekt' structure, which is a stack of three intermediate blocks. The first intermediate block points to logical block 10 ('indirekt'), and the second points to logical block 12 ('indirekt'). Below this, a 'dreifach indirekt' structure is shown, where the first intermediate block points to logical block 16.521 ('indirekt'), and the second points to logical block 16.522 ('indirekt'). The final logical block shown is 2.113.673. A legend at the top right explains that the 13 pointers in the i-node refer to data blocks: 10 point directly to the first 10 blocks, and 11, 12, and 13 can be further subdivided.

Einschub: NFS, Verteiltes Dateisystem

- Das **Network File System** (NFS) verbindet Dateisysteme mehrerer Rechner zu einem logisch zusammengehörigen Dateisystem
 - NFS-Protokoll

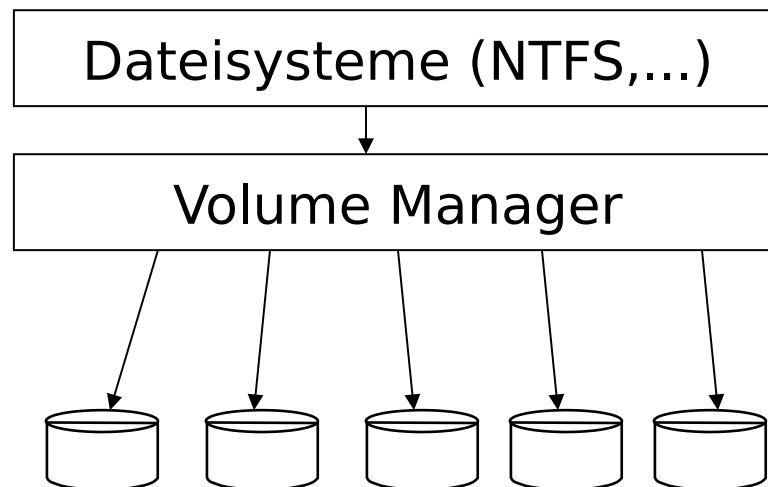


Fallbeispiel: Unix, Verzeichnisstruktur

- Jedes Unterverzeichnis und jede Datei lassen sich eindeutig durch den Weg von der Wurzel zur Datei beschreiben
- Die Folge der Knoten auf dem Weg wird, mit „/“ getrennt, als **Pfad** bezeichnet.
- Das Dateisystem jedes Users ist ein Subdirectory des gesamten Unix-Baums
- Die Wurzel des Unterbaums des Users nennt man **Home-Directory**
- Der User arbeitet meist in seinem Home-Directory
- Bei der Arbeit bezieht man sich oft auf Dateien des selben Verzeichnisses (**working directory**)

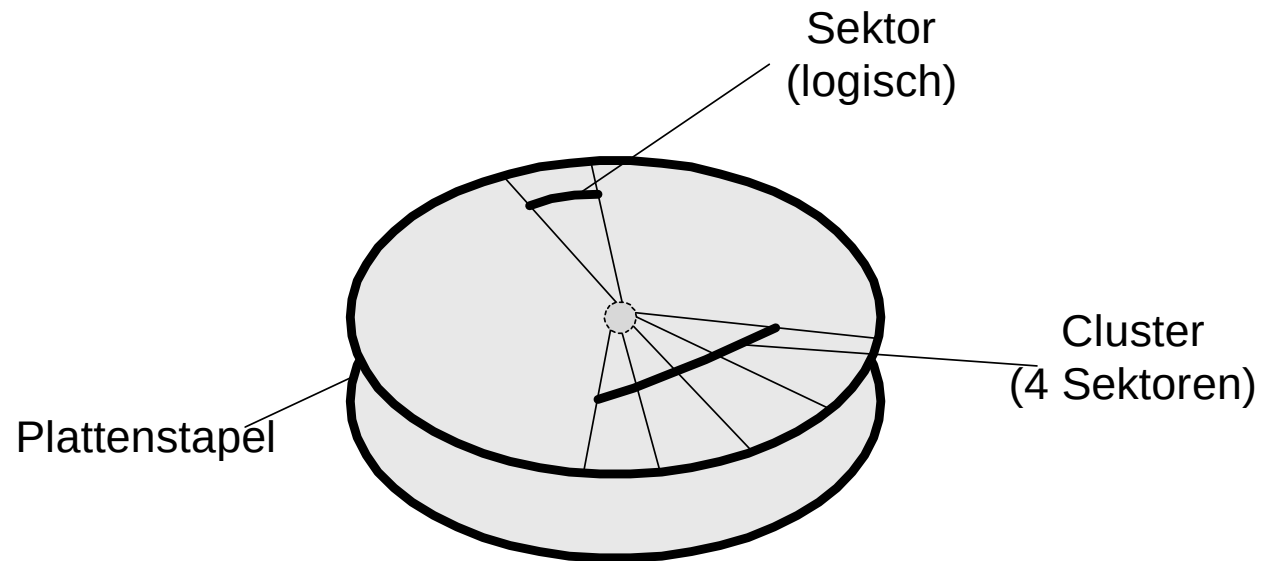
Fallbeispiel: Windows, Volume Manager

- Unter Windows gibt es einen **Volume Manager**
 - (Unter Linux LVM)
 - Verwaltung der Platten auf logischer Ebene
 - Stellt logische Laufwerke für die Anwendungen bereit
 - Veränderung, Neupartitionierung ist dynamisch, ohne Systemneustart möglich



Fallbeispiel: Windows, Cluster

- Cluster = in Windows adressierbarer Filesystemblock
- Cluster besteht aus einem oder mehreren Sektoren
- **FAT-x** → x gibt die Anzahl der Bits an, die für die Clusteradressierung verwendet werden
- Bit-Anzahl bestimmt die Größe des Filesystems



Fallbeispiel: Windows, Überblick zu Dateisystemen

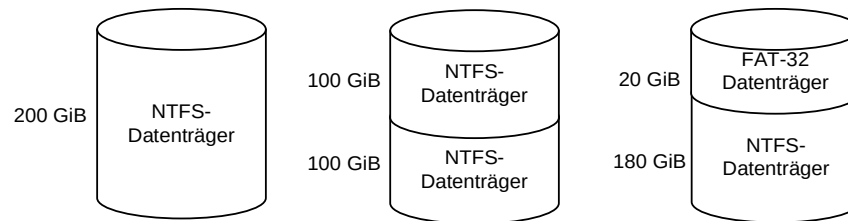
- **FAT-16** ist das alte MS-DOS Filesystem
 - nutzt 16-Bit Plattenadressen und max. 2 GiB (GibiByte) große Partitionen
- **FAT-32** ist das alte MS-DOS Filesystem
 - nutzt 32-Bit Plattenadressen und max. 2 TiB große (TebiByte) Partitionen
- **NTFS** (NT File System) ist das moderne, hierarchische Filesystem von Windows 2000
 - NTFS nutzt 64-Bit Plattenadressen und unterstützt Partitionen bis zu einer Größe von 2^{64} Byte
- Windows unterstützt auch **Read-Only Filesysteme** für DVDs (UDF) und CD-ROMs (CDFS)

Fallbeispiel: Windows, Filesysteme-Überblick

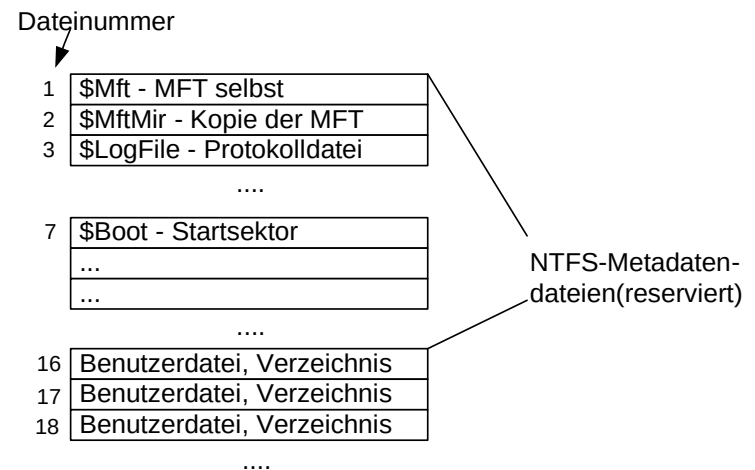
File-system	Bits für Clusterindex	Anzahl Cluster	Unterstützte Clustergrößen	Maximale Filesystemgröße
FAT-12	12 Bit	$2^{12} = 4.096$	512 Byte – 8 KiB	32 MiB
FAT-16	16 Bit	$2^{16} = 65.536$	512 Byte – 64 KiB	4 GiB (GibiByte)
FAT-32	32 Bit, aber nur 28 genutzt	2^{28}	512 Byte – 32 KiB	8 TiB (TebiByte) begrenzt auf 32 GiB
NTFS	64 Bit	2^{64}	512 Byte – 64 KiB	16 EiB (ExbiByte) begrenzt auf 256 TiB

Fallbeispiel: Windows, Partitionierung

- Plattenpartitionierung in NTFS
- Eine Platte kann mehrere logische Datenträger (Volumes) enthalten
- Festlegung bei der Formatierung, Beispiel:



- Eine Master File Table (MFT) je Volume
- Für jede Datei und jedes Verzeichnis ein oder mehrere Einträge in der MFT



Überblick

1. Geräteverwaltung

- Überblick
- Treiber, Gerätetypen, Gerätemodelle,...
- Fallbeispiel: Unix

2. Dateisysteme

- Grundlagen
- Fallbeispiel: Unix
- Fallbeispiel: Windows

3. **Storage Systeme**

- RAID
- SAN, NAS

Storage Systeme

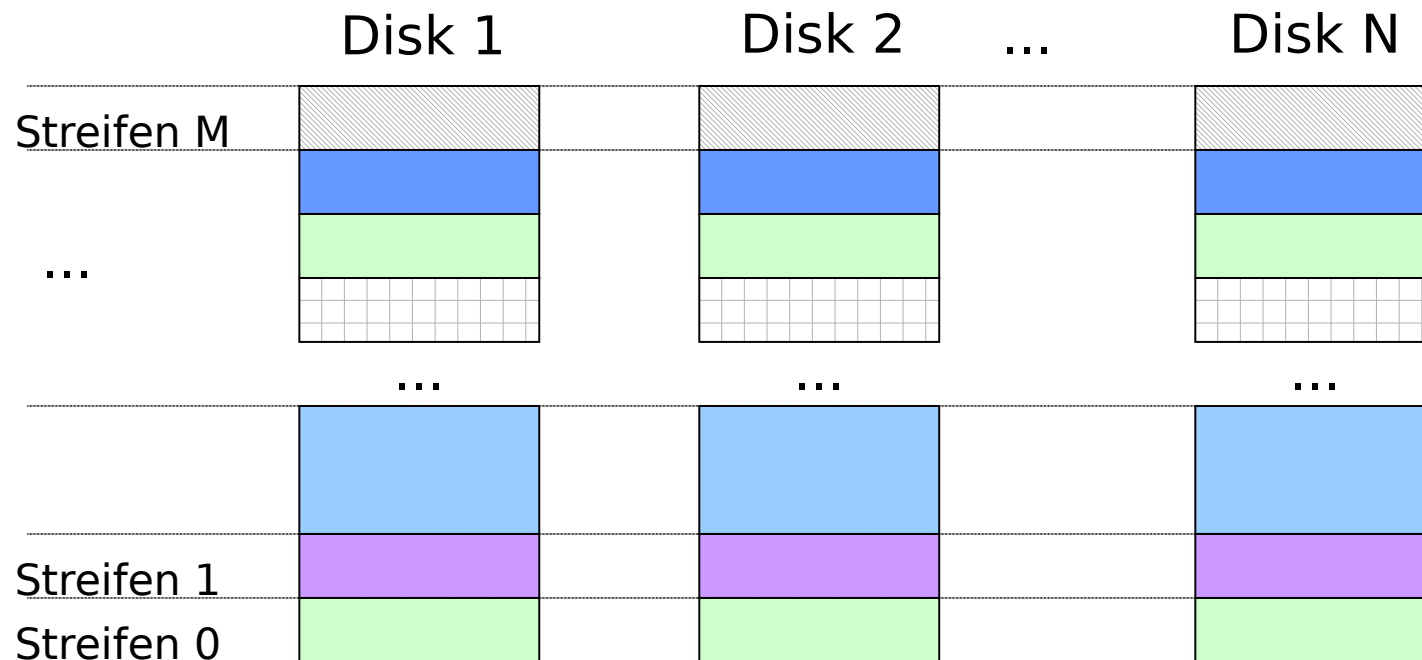
- Bisher betrachtet:
 - **DAS:** Direct Attached Storage = herkömmliche Speichersysteme
- Weitere wichtige Begriffe:
 - **RAID-Systeme**
 - **SAN:** Storage Area Network, Hochgeschwindigkeitsnetzwerk zwischen Servern und Subsystemen
 - **NAS:** Network Attached Storage, NAS-Systeme sind Dateiserver, keine Festplattenserver

RAID-Plattensysteme

- Multiple Plattenspeicher, sog. RAIDs
 - RAID steht für **R**edundant **A**rray of **I**nexpensive (heute **I**ndependent) **D**isks
 - sind heute sehr verbreitet
- Mehrere kleine Platten werden hier als große **virtuelle Platte** verwaltet
- Man verwendet RAID-Systeme auch zur Verbesserung der **Leistung** und zur Erhöhung der **Ausfallsicherheit**
- In HW implementiert für Betriebssystem transparent, als SW vom BS selbst betrieben
- Es werden verschiedene Varianten unterschieden

RAID-Plattensysteme

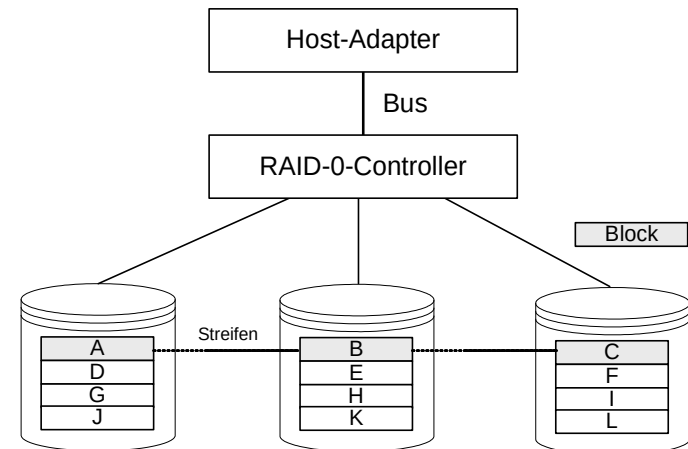
- Der gesamte logische Plattenbereich wird in Streifen (stripes) eingeteilt, die sich über mehrere Disks erstrecken



RAID-Plattensysteme

■ RAID-0-System:

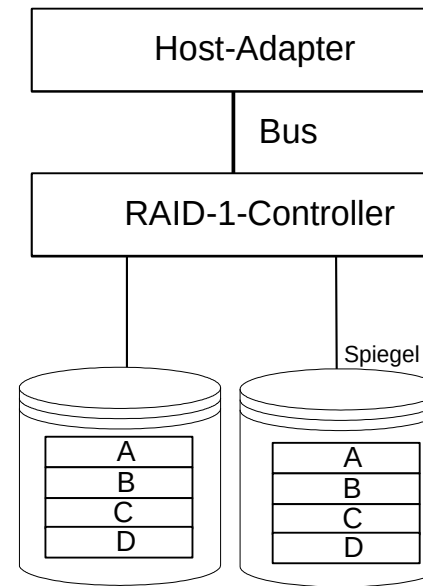
- Stripes werden über mehrere Platten eines Arrays verteilt
- Größe der Stripes beeinflusst die Leistung erheblich
 - Stripes-Größe: 32 KB, 64 KB, 128 KB (Striping-Granularität)
- Verteilung übernimmt entweder das Betriebssystem oder ein eigener RAID-Controller
- **Hoher** I/O-Durchsatz, schnelle Variante, aber **nicht** ausfallsicher



RAID-Plattensysteme

■ RAID-1-System:

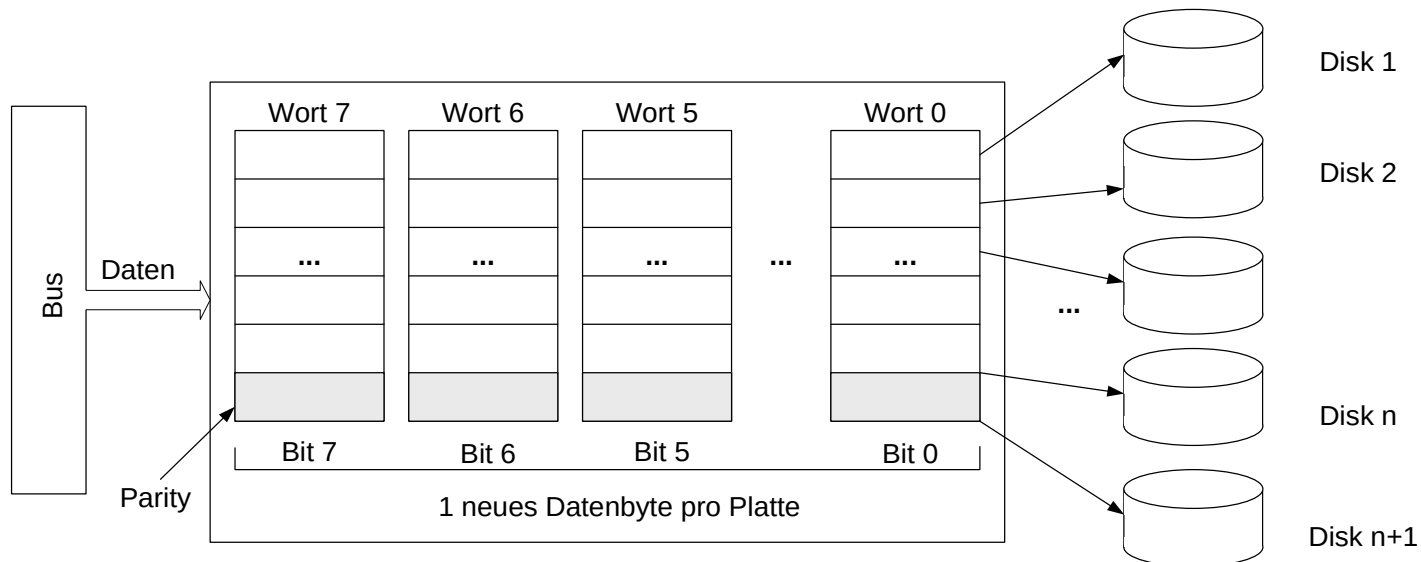
- Spiegelung der Daten, volle Redundanz der Daten
- Spiegelung durch Betriebssystem oder eigenen RAID-Controller möglich
- Ausfallsicher aber langsam



RAID-Plattensysteme

■ **RAID-2-System** (heute nicht mehr eingesetzt):

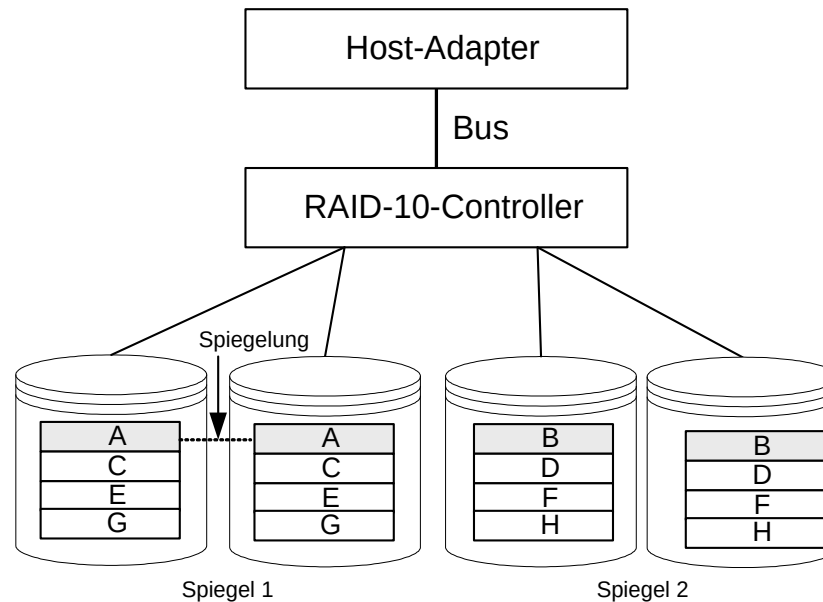
- Bitweises Striping und Ergänzung von Prüfdaten (Paritätsbits) zur Fehlerkorrektur
- Jedes Byte wird um ein Paritätsbit ergänzt
- Bei Ausfall einer Platte kann diese rekonstruiert werden
 - → Ausfallsicherheit für eine Platte des Arrays



RAID-Plattensysteme

■ **RAID-10-System: häufig verwendet!**

- Striping und Spiegelung, Kombination RAID-0 und RAID-1
- Schnell und ausfallsicher
- Zuerst Striping und danach Spiegelung



RAID-Plattensysteme

■ **RAID-3-System:**

- Wie RAID-2, aber byteweises Striping
- Prüfdaten zusammenfassen und auf dedizierter Platte speichern
- Dient der Reduzierung der Kosten für die 1:1-Spiegelung (heute nicht mehr so relevant)
- Ausfall einer Platte führt nicht zu einer Ausfallzeit, Ausfall von zwei Platten schon

RAID-Plattensysteme

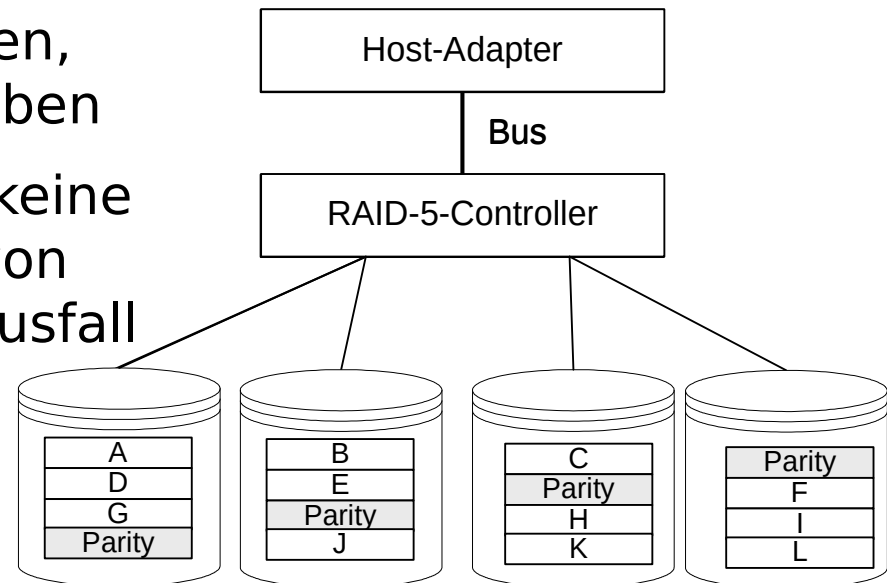
■ **RAID-4-System:**

- Hier werden auch die Datenbits wie bei RAID-0 zu Streifen zusammengefasst
- Pro Streifen wird eine Prüfsumme gebildet und auf einer eigenen Platte gespeichert
- Ausfallsicherheit wie bei RAID-3, Speicheraufwand weniger als bei RAID-1, aber langsameres Schreiben

RAID-Plattensysteme

■ RAID-5-System:

- In Gegensatz zu RAID-4 werden die Paritätsabschnitte auf die beteiligten Platten verteilt → gleichmäßige Plattenauslastung
- Gute Leistung beim Lesen, schlechtere beim Schreiben
- Verlust einer Platte hat keine Auswirkungen, Verlust von zwei Platten bedeutet Ausfall
- Nicht so geeignet bei hoher Transaktionslast



RAID-Plattensysteme

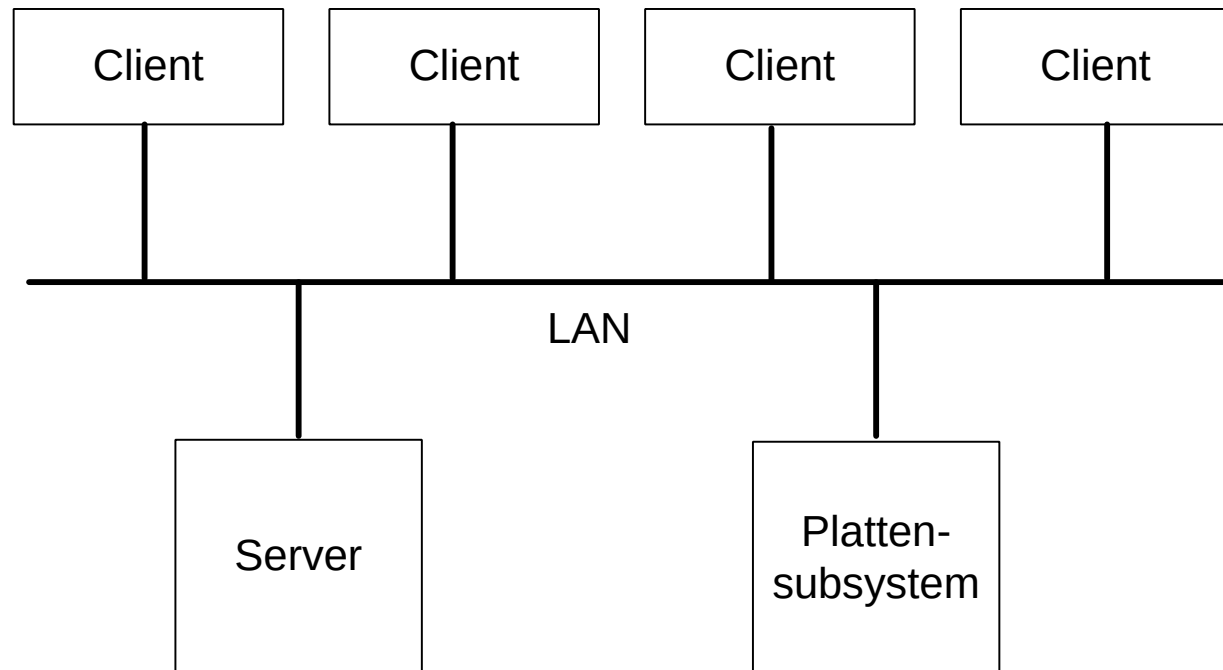
■ **RAID-6-System:**

- Wie RAID-5 mit redundanten Prüfdaten auf weiterer Platte, so dass sogar der Ausfall von zwei Platten ohne Auswirkung bleibt
- Gute Leistung beim Lesen, Schreibleistung schlechter als RAID-5
- Geeignet für sehr hohe Ausfallsicherheitsanforderungen

Network Attached Storage

■ NAS

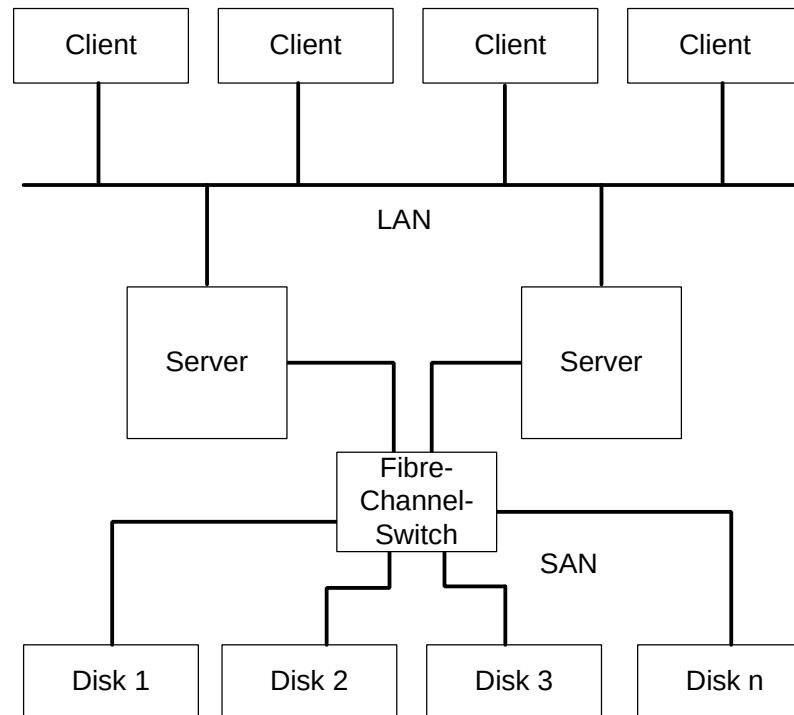
- Massenspeichereinheiten, die an ein lokales Netzwerk (LAN) angeschlossen sind
- Nachteil: belasten das Netzwerk



Storage Area Networks

■ SAN

- Eigenes Netzwerk zwischen Servern und den Speicherressourcen
- Speicher kann virtuell wie eine einzige Festplatte behandelt werden



Zusammenfassung

- Geräte- und Dateiverwaltung sind eigene Softwareschichten im Kernel
- Festplatten wichtigstes externes Speichermedium
- Dateien und Dateisysteme (FAT, NTFS, Unix-Dateisysteme) als Abstraktion
- RAID-Systeme dienen der gesicherten Datenspeicherung
 - RAID 1, RAID-10 und RAID 5 oft im Einsatz
- NAS- bzw. SAN sind Storage-Systeme, die sich weiter Verbreitung erfreuen

Gesamtüberblick

- ✓ Einführung in Computersysteme
 - ✓ Entwicklung von Betriebssystemen
 - ✓ Architekturansätze
 - ✓ Interruptverarbeitung in Betriebssystemen
 - ✓ Prozesse und Threads
 - ✓ CPU-Scheduling
 - ✓ Synchronisation und Kommunikation
 - ✓ Speicherverwaltung
 - ✓ Geräte- und Dateiverwaltung
10. Betriebssystemvirtualisierung