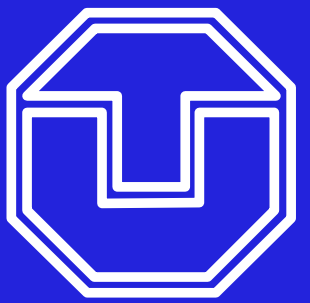
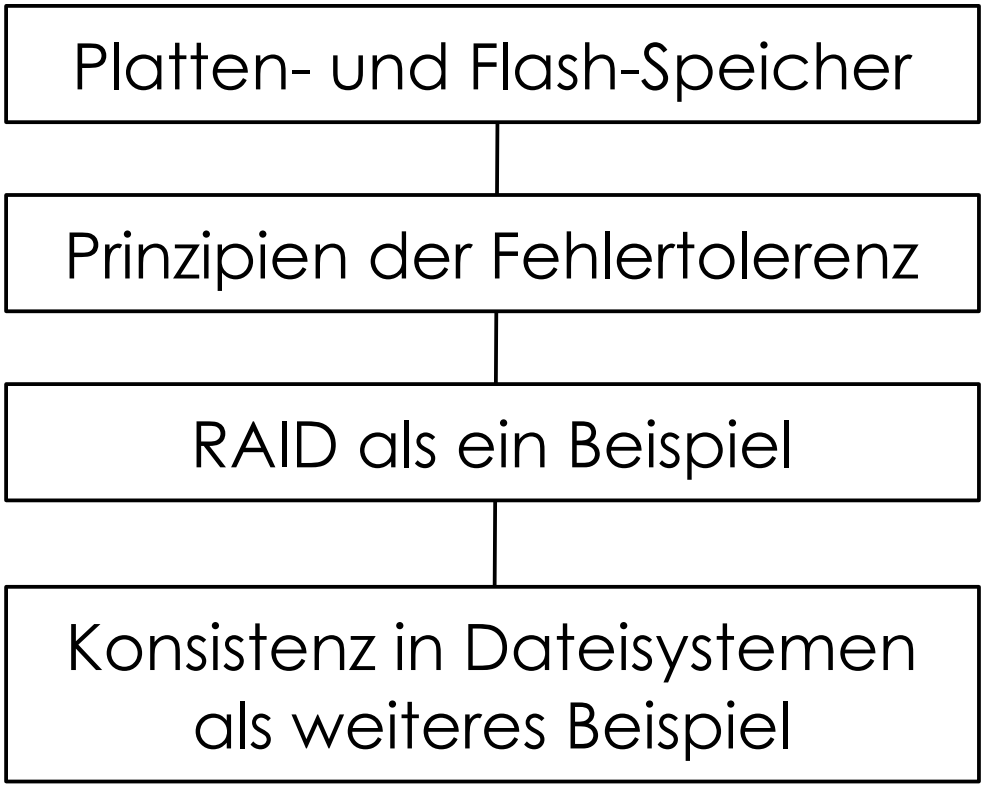
Hardware, Fehlertoleranz, Am Beispiel Dateisysteme

Betriebssysteme

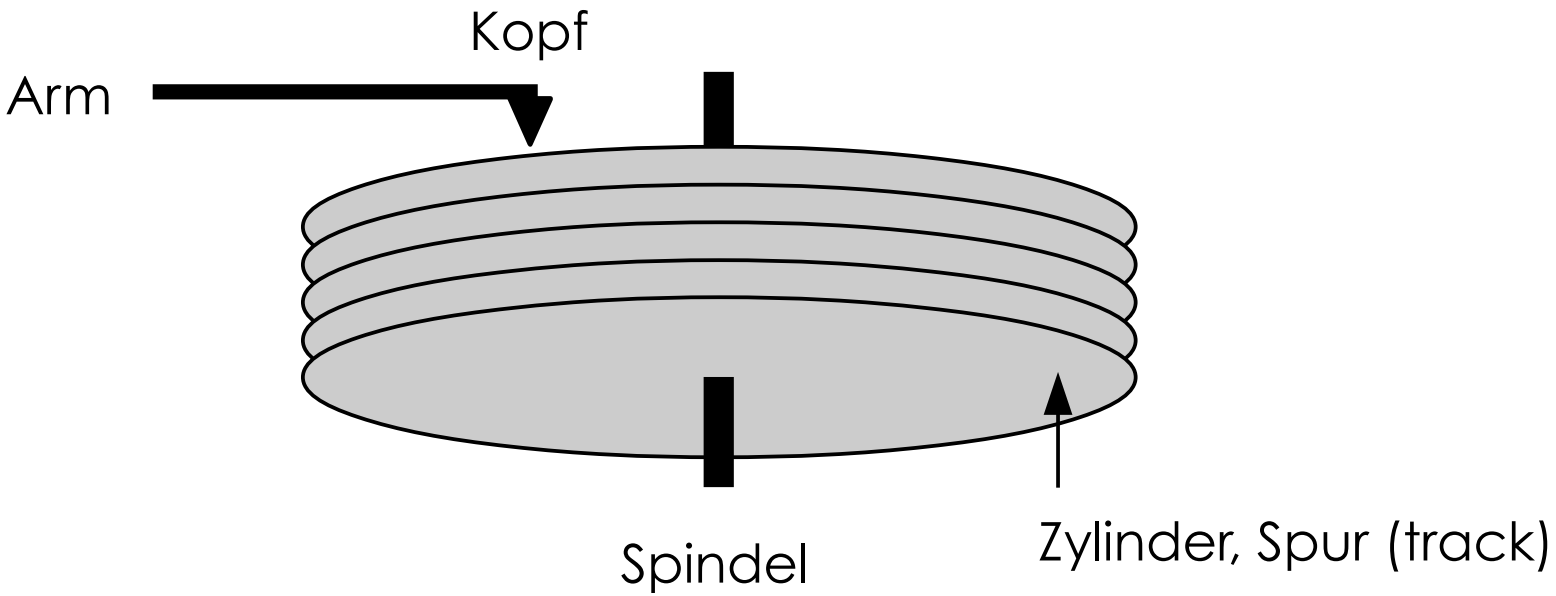
Hermann Härtig
TU Dresden



Wegweiser


Platten- und Flash-Speicher
Prinzipien der Fehlertolerenz
RAID als ein Beispiel
Konsistenz in Dateisystemen
als weiteres Beispiel

Funktionsweise von Platten



Funktionsweise von Platten

Eigenschaften

- persistente Speicherung
- Lesen/Schreiben in Einheiten von Blöcken (Sektoren) an beliebiger Stelle der Platte
- Beliebig häufiges Lesen und Schreiben
- Mechanik:
  Kopf verschieben / Scheiben rotieren

Block:

Header Trailer: error correction codes

Eigenschaften von Platten

Entwicklung der Plattentechnik

- Latenz: verbessert sich nur sehr allmählich
- Kapazität: wächst sehr schnell
- (neu: flash-memory in Plattenlaufwerken)

im Vergleich zu anderen Komponenten

- CPU-Leistung wächst sehr schnell
- Netzleistung wächst sehr schnell

➔ Dateisystem: außerordentlich kritisch für die Leistungsfähigkeit von Rechnern

Kenngrößen von Platten

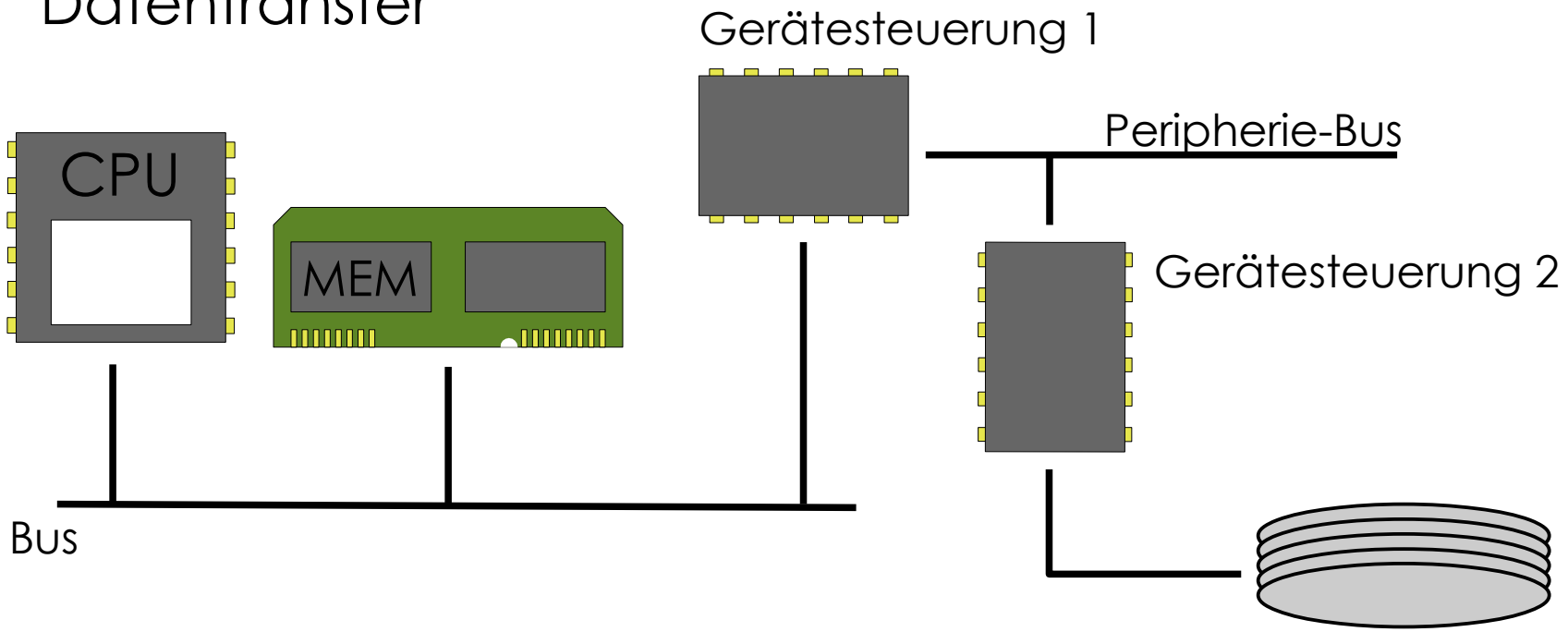
Größe, Kapazität und Zugriffsgeschwindigkeit sind bestimmt durch:

- Drehgeschwindigkeit
- Spurdichte (Zoning)
- Schreibdichte
- Beschleunigungs- und Bewegungszeiten
- Settle-Time (Kopfberuhigung, Kalibrierung)
- Caching
- Bus

Ablauf eines Plattenzugriffs (SCSI)

Rechner - Gerätesteuerung (1): Kommando, Adresse
Gerätesteuerung (2): Kommando, Adresse

- Umrechnung der Adresse in plattengeometrische Größen
- „Seek“: Start-Zeit, Bewegung, Stop-Zeit
- Abwarten Rotation
- Datentransfer



Zugriffszeit

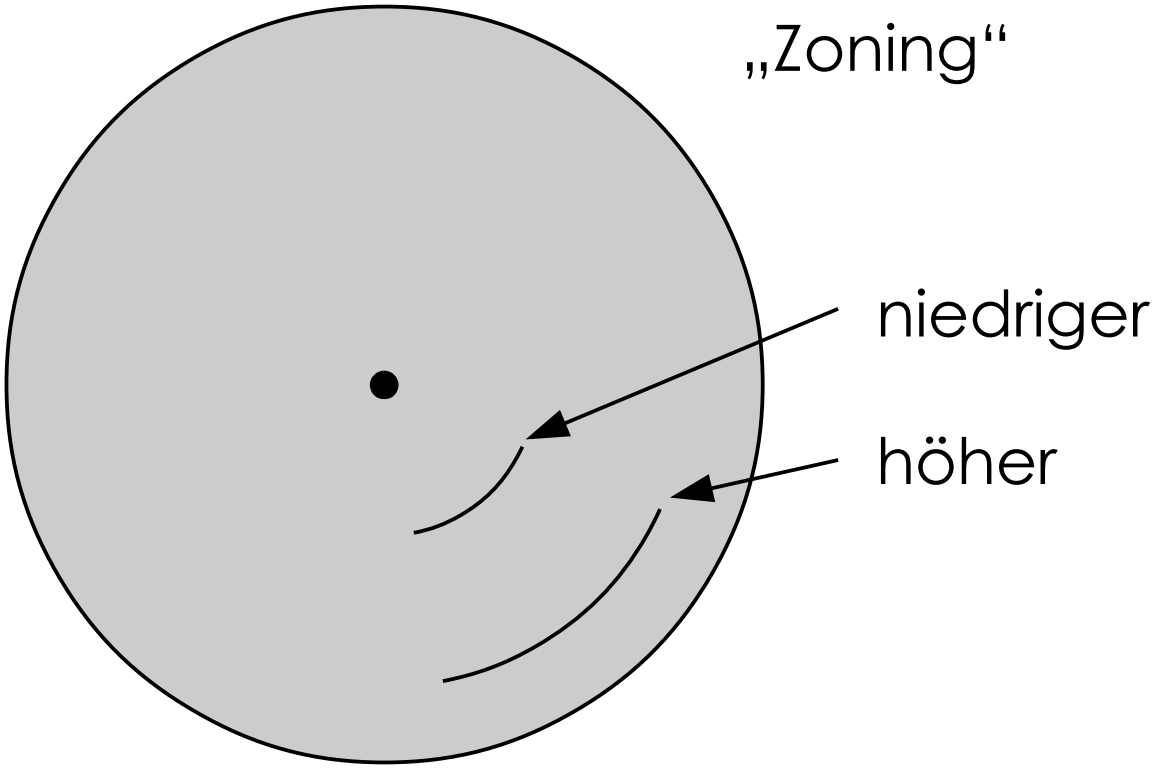
... setzt sich zusammen aus:

- Kommandogenerierung und -transfer
- seek(Positionieren):
  - Beschleunigung
  - Bewegung
  - Abbremsen
  - Kopfberuhigung
  - Spur-Identifikation
- Latency (Rotationslatez)
- Daten-Transfer

Bandbreite

... ist die übertragene Informationsmenge pro Zeiteinheit (MByte/s) (Drehgeschwindigkeit konstant)

Laufwerk:



Beispiel

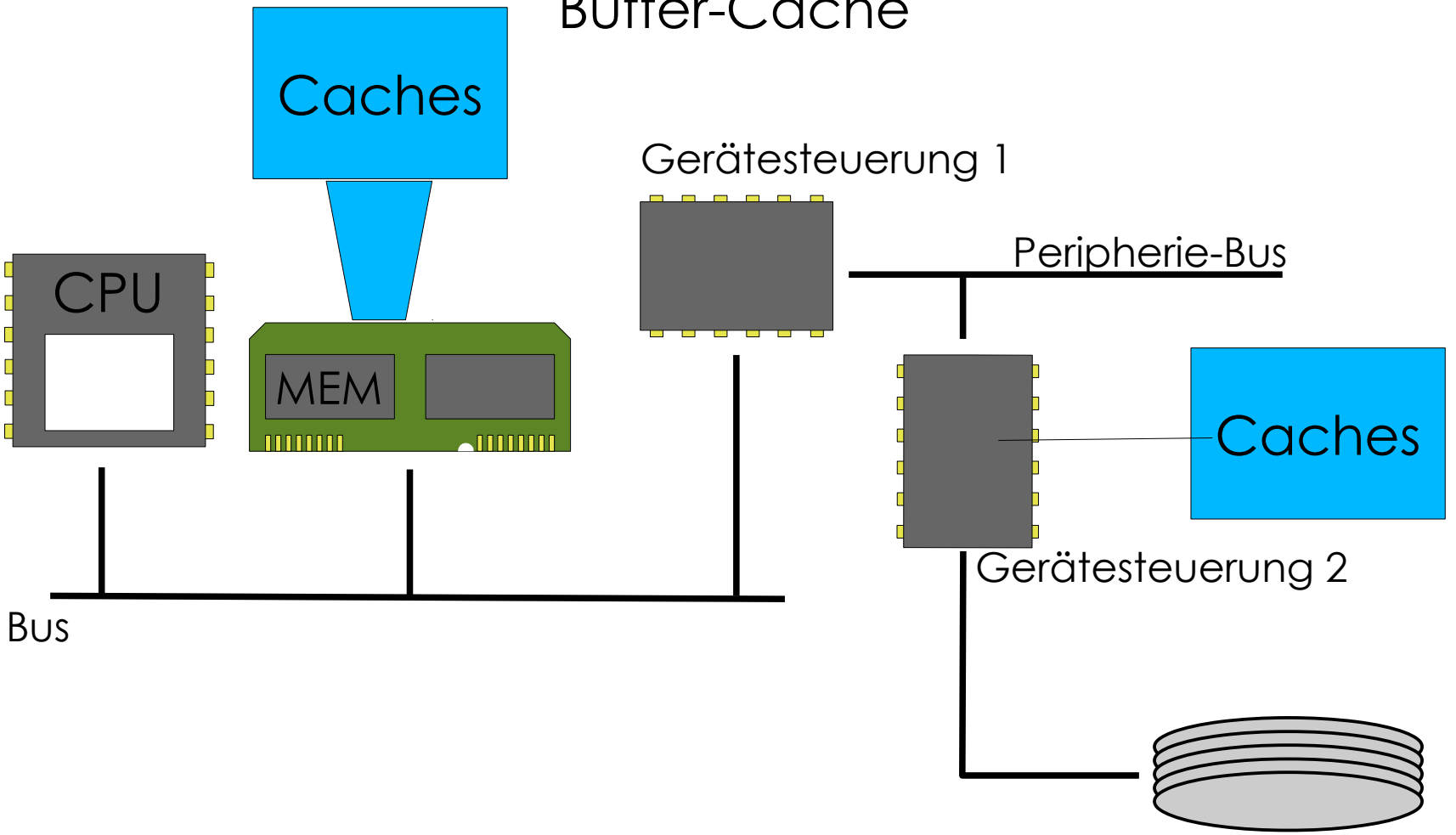
- Abstand von Spindel: 1 cm / 2 cm
- Spurumfang: 6 cm / 12 cm

Konsequenzen

- Allokation: Block → Dateien
  - Blöcke einer Datei in der „Nähe“ zueinander
  - Dateien mit großer Anforderung an Bandbreite nach „außen“
  - Ziel: möglichst große Lese-/Schreiboperationen
- Vorauslesen („read ahead“)
- Plattenaufträge sortieren
  - Nicht mehr FIFO,
  - minimiere mechanische Bewegungen (seek, rotate)
-

Caches

Virtueller Speicher
Buffer-Cache



Beispiel Serverplatte (leicht veraltet)

SCSI(3) Ultra320 (BUS):

- max. Bandbreite 320 MB/s

IBM ...

- Kapazität 73,4 GB
- Zonen 17
  - innen 29,8 MB/s
  - außen 58,0 MB/s
- Tracks per Inch 27.312
- Umdrehungen/Min. 10.000
- Speicherdichte 13,2 GB/square inch
- Average seek time 4,9 ms

Flash-Speicher

Siehe Folien Nr. 8-21

aus dem Foliensatz von Prof Tei Wei Kuo, NTU

Beispiel SSD (Flash)

Serial ATA 6G:

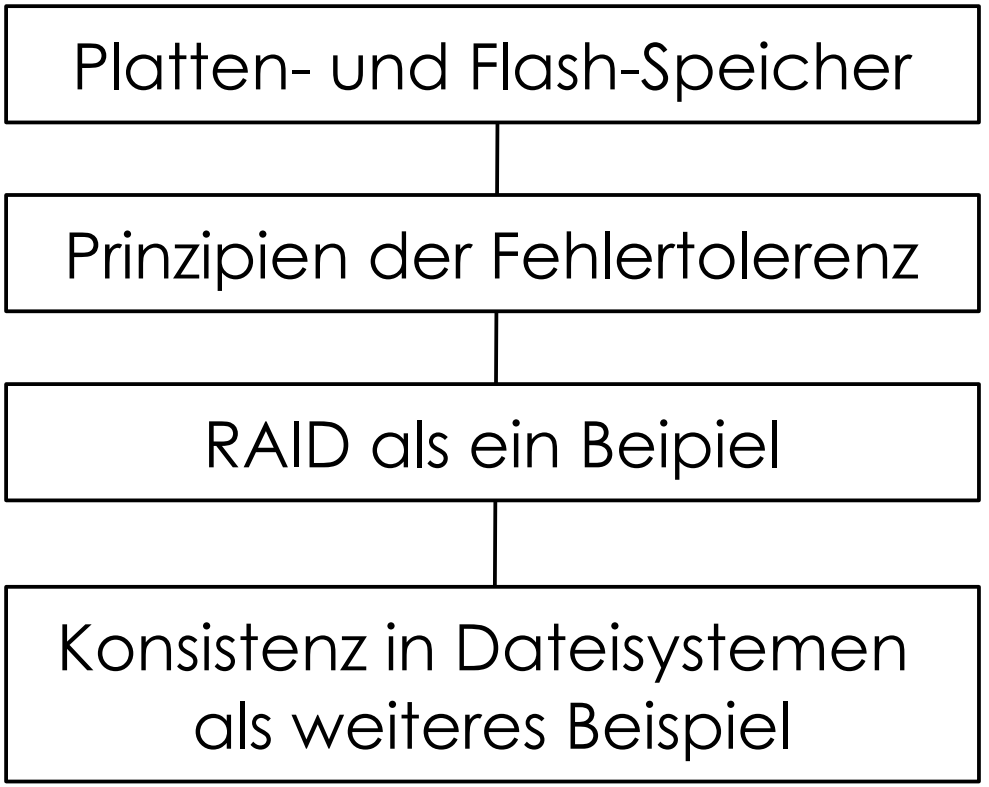
- maximale Bandbreite — 6 Gbit/s
- Netto-Datenrate — ca. 600 MB/s

Aktuelle SSD:

- Kapazität — 240 GB
- Leserate:
  - 128KB-Blöcke, sequentiell — 529 MB/s
  - 4KB-Blöcke, random — 150 MB/s
- Schreibrate:
  - 128KB-Blöcke, sequentiell — 234 MB/s
  - 4KB-Blöcke, random — 231 MB/s
- Zugriffszeiten — < 0,1 ms

Quelle: Benchmarks in c't 9/2011, S. 133

Wegweiser


Platten- und Flash-Speicher
Prinzipien der Fehlertolerenz
RAID als ein Beipiel
Konsistenz in Dateisystemen
als weiteres Beispiel

Fehlertoleranz: die generelle Fragestellung

Wie baut man aus unzuverlässigen (fehlerhaften) Komponenten zuverlässigere Systeme ?

Fehler:

- Entwurfsfehler
- Ausfälle

Fehlertoleranz: Aufgaben

- Fehlertypen festlegen („Fehlermodell“)
- Fehler erkennen und Ausbreitung beschränken
- Fehler beheben („Recovery“)
- Redundanz bereitstellen
- Reparieren

Formen der Redundanz

- Zeit
- Struktur
- Information
- Funktion

Fehlertypen (am Beispiel Dateisystem)

Bitfehler in Blöcken

- transient
- permanent:
  - nicht korrigierbarer Blockfehler (defekter Block)
  - Ausfall eines gesamten Laufwerkes
  - Systemabsturz (Inkonsistenz von Datenstrukturen auf Platte)

Transiente Lesefehler

- Jeder Block auf Platte hat header/trailer mit redundanter Codierung
- Fehlererkennung und -korrektur durch Auswertung dieses Codes
- Buchführung zu Wartungszwecken

Permanente Lesefehler: Verwaltung defekter Blöcke

Erkennung durch redundante Codes

- ➔ häufig schon bei Herstellung

Behebung durch eine Vielfalt von Techniken:

- Plattenspuren haben BadSektor-Markierungen
- Ausblendung häufig schon durch Controller
  - Datei, die alle defekten Blöcke allokiert hat
  - ➔ Vorsicht: muss bei Backup berücksichtigt werden
  - defekte Blöcke werden von vornherein als allokiert markiert; logische Ersetzung dieser Blöcke durch andere intakte Blöcke

Ausfall von Laufwerken

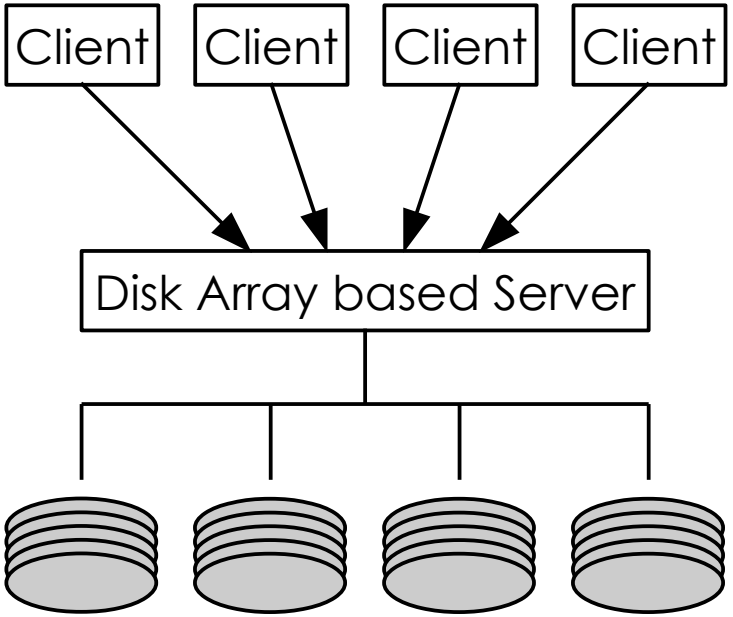
- Entdeckung:
  - Timeout
  - sich häufende Lesefehler
- Recovery + Redundanz: RAID
- Reparatur: Plattentausch

RAID - Redundant Array of Independent Disks

Zusammenbau mehrere unabhängig ansteuerbarer Platten zu einem Feld

Ziel

- bessere Leistung durch parallele Zugriffe
- Ausgleich der Lücke zwischen immer schnelleren Prozessoren/Speichern und nach wie vor langsamen Platten

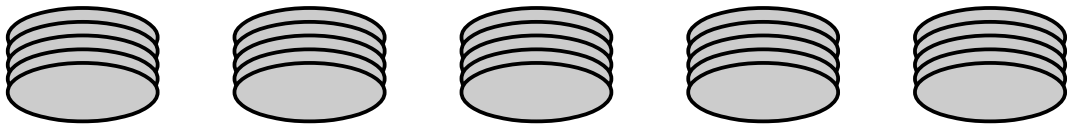


Nachteil

- mittlere Ausfallzeit des Feldes ist kleiner
- ➔ Maßnahmen zur Fehlertoleranz !

Ohne Redundanz (RAID 0)

- Disk Array aus n Platten
- keine Kosten für Redundanz (trivial)
- keine Fehlertoleranz



Disk 1 . . . Disk n

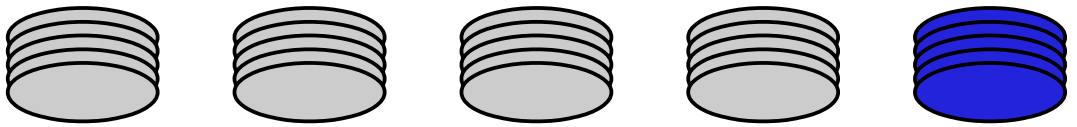
Spiegelplatten (RAID 1)

- Disk Array aus 2n Platten
- jede Platte gespiegelt (identische Kopien)
- schreiben: alle Daten zweimal
- lesen: einmal, von schnellerer Platte
- fehlertolerant (ein beliebiger Ausfall)
- hohe Speicherkosten

Disk 1 . . . Disk n Disk 1' . . . Disk n'

Einzelne Paritätsplatte (RAID 4)

- Nutzerdaten werden blockweise auf n Platten verteilt (striping)
  Dateiblock 1 auf Disk 1, 2 auf 2, ...
- Blockweise Paritätsbildung
  (Disk 1, Block 1) XOR (Disk 2, Block 1) ... → (Disk P, Block 1)
- Fehlererkennung durch jede Platte separat
  Rekonstruktion durch Paritätsbildung



Disk 1 . . . Disk n Parity

P1	P2	P3	P4	P5	PP		
1	0	F	1	1	1	→	P3 =

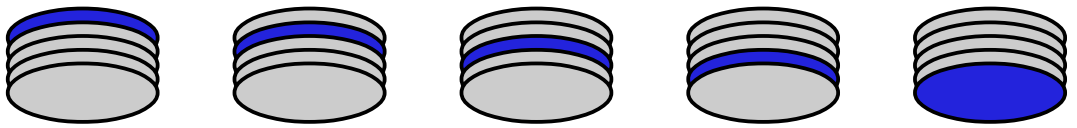
RAID Level 5

Probleme mit RAID 4:

- bei hohem Leseanteil: Paritäts-Platte nicht richtig ausgelastet
- bei vielen kleinen Dateien und Schreiboperationen: Engpass an Paritätsplatte

Block Interleaved Distributed Parity

- Paritätsblöcke über gesamtes Array verteilt



- beste Leistung bei kleinen und großen Leseanforderungen sowie bei großem Anteil Schreiboperationen
- kleine Schreibanforderungen immer noch ineffizienter, da Paritätsblock jedesmal lokalisiert und geschrieben werden muss

Zusammenfassung

Aufgabe: Verwaltung und persistente Speicherung großer Datenmengen

Problem: Lücke in der Zugriffsgeschwindigkeit

Weiterführendes

Konsistenzmechanismen in Dateisystemen
(Journaling, Logging, etc. → Freitag)

Verteilte Dateisysteme (später in dieser Vorlesung