4. Foliensatz Betriebssysteme

Prof. Dr. Christian Baun

Frankfurt University of Applied Sciences
(1971–2014: Fachhochschule Frankfurt am Main)
Fachbereich Informatik und Ingenieurwissenschaften
christianbaun@fb2.fra-uas.de

Lernziele dieses Foliensatzes

- Am Ende dieses Foliensatzes kennen/verstehen Sie...
  - den Aufbau, die Arbeitsweise und die Eckdaten von **Festplatten**
  - den Aufbau, die Arbeitsweise und die Eckdaten von **Solid State Drives**
  - die Arbeitsweise und die am häufigsten verwendeten Varianten von Redundant Array of Independent Disks (**RAID**)

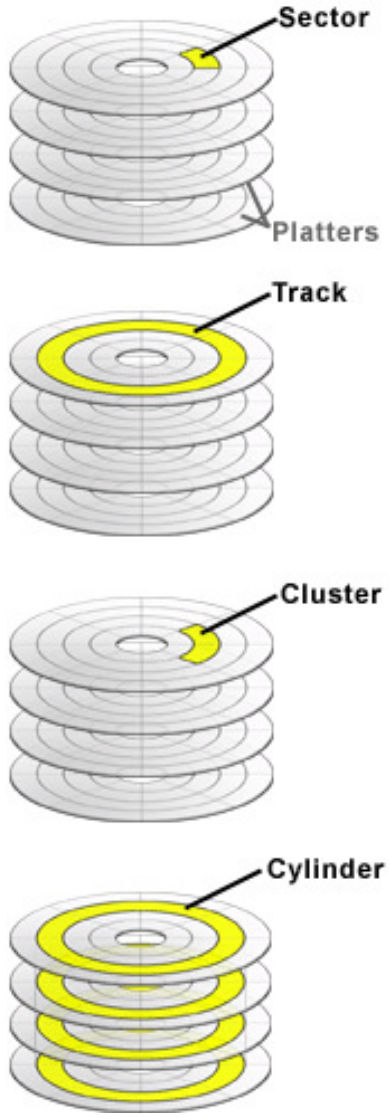
Übungsblatt 4 wiederholt die für die Lernziele relevanten Inhalte dieses Foliensatzes

Festplatten

- Festplatten sind ca. Faktor 100 preisgünstiger pro Bit als Hauptspeicher und bieten ca. Faktor 100 mehr Kapazität
  - Nachteil: Zugriffe auf Festplatten sind um ca. Faktor 1000 langsamer
- Grund für die geringere **Zugriffsgeschwindigkeit**:
  - Festplatten sind mechanische Geräte
    - Sie enthalten eine oder mehrere Scheiben, die mit 4200, 5400, 7200, 10800 oder 15000 Umdrehungen pro Minute rotieren
- Für jede Seite jeder Platte existiert ein Schwungarm mit einem **Schreib-/Lesekopf**
  - Der Schreib-/Lesekopf magnetisiert Bereiche der Scheibenoberfläche und schreibt bzw. liest so die Daten
  - Zwischen Platte und Kopf ist ein Luftpolster von ca. 20 Nanometern
- Auch Festplatten haben einen Cache (üblicherweise $\leq 32\,\text{MB}$)
  - Dieser puffert Schreib- und Lesezugriffe

Logischer Aufbau von Festplatten (1/2)

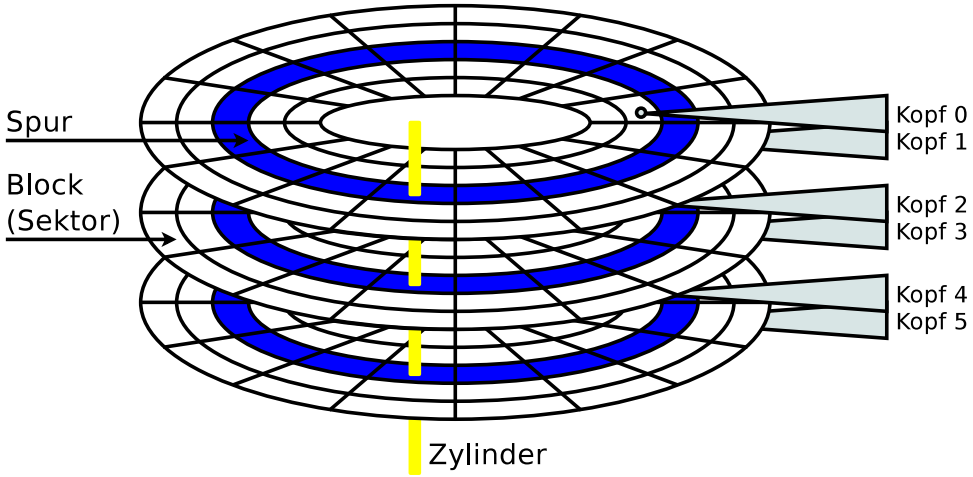
- Die Oberflächen der Scheiben werden in kreisförmigen **Spuren** (*Tracks*) von den Köpfen magnetisiert
- Alle Spuren auf allen Platten bei einer Position des Schwungarms bilden einen **Zylinder** (*Cylinder*)
- Die Spuren sind in logische Einheiten (Kreissegmente) unterteilt, die **Blöcke** oder **Sektoren** heißen
  - Typischerweise enthält ein Block 512 Bytes Nutzdaten
  - Sektoren sind die kleinsten adressierbaren Einheiten auf Festplatten
  - Müssen Daten geändert werden, muss der ganze Sektor gelesen und neu geschrieben werden
- Heute werden auf Softwareseite **Cluster** angesprochen
  - Cluster sind Verbünde von Sektoren mit fester Größe, z.B. 4 oder 8 kB
  - Bei modernen Betriebssystemen sind Cluster die kleinste Zuordnungseinheit

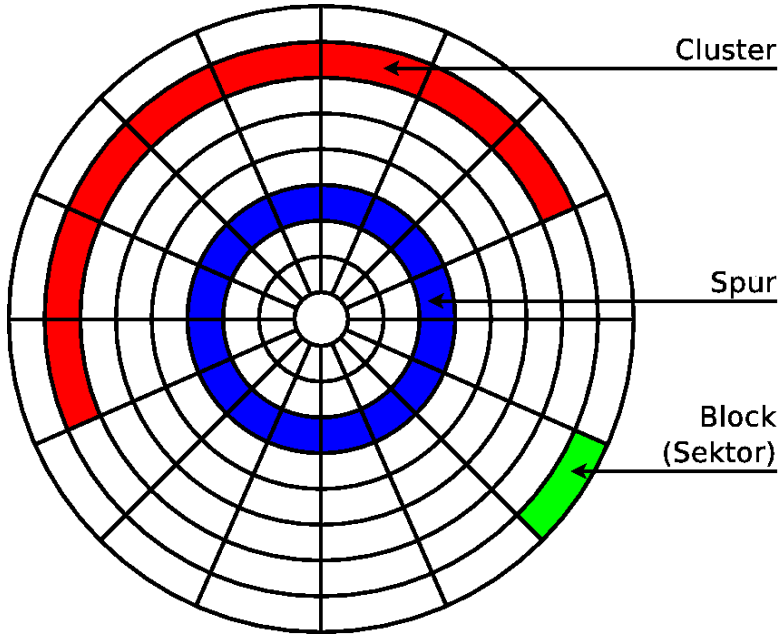


Bildquelle: SweetScape

Logischer Aufbau von Festplatten (2/2)

Bildquelle: `http://www.hitechreview.com`





Adressierung der Daten auf Festplatten (1/4)

- Festplatten $\leq$ 8 GB verwenden ***C**ylinder-**H**ead-**S**ector-Adressierung*
- CHS unterliegt mehreren Einschränkungen:
  - Die Schnittstelle Parallel ATA verwendet 28 Bits für CHS-Adressierung und davon...
    - 16 Bits für die Zylinder (maximal 65.536)
    - 4 Bits für die Köpfe (maximal 16)
    - 8 Bits für die Sektoren/Spur (maximal 255. Sektornummer 0 wird nicht verwendet)
  - Das BIOS verwendet 24 Bits für CHS-Adressierung und davon...
    - 10 Bits für die Zylinder (maximal 1.024)
    - 8 Bits für die Köpfe (maximal 255. Kopfnummer 0 wird nicht verwendet)
    - 6 Bits für die Sektoren/Spur (maximal 63. Sektornummer 0 wird nicht verwendet)
- Bei den Grenzen ist der jeweils niedrigere Wert entscheidend
  - Darum können alte BIOS-Versionen maximal 504 MB adressieren

- 1.024 Zylinder * 16 Köpfe * 63 Sektoren/Spur * 512 Bytes/Sektor = 528.482.304 Bytes
- 528.482.304 Bytes / 1024 / 1024 = 504 MB

Adressierung der Daten auf Festplatten (2/4)

Bildquelle: `http://www.eak-computers.com`

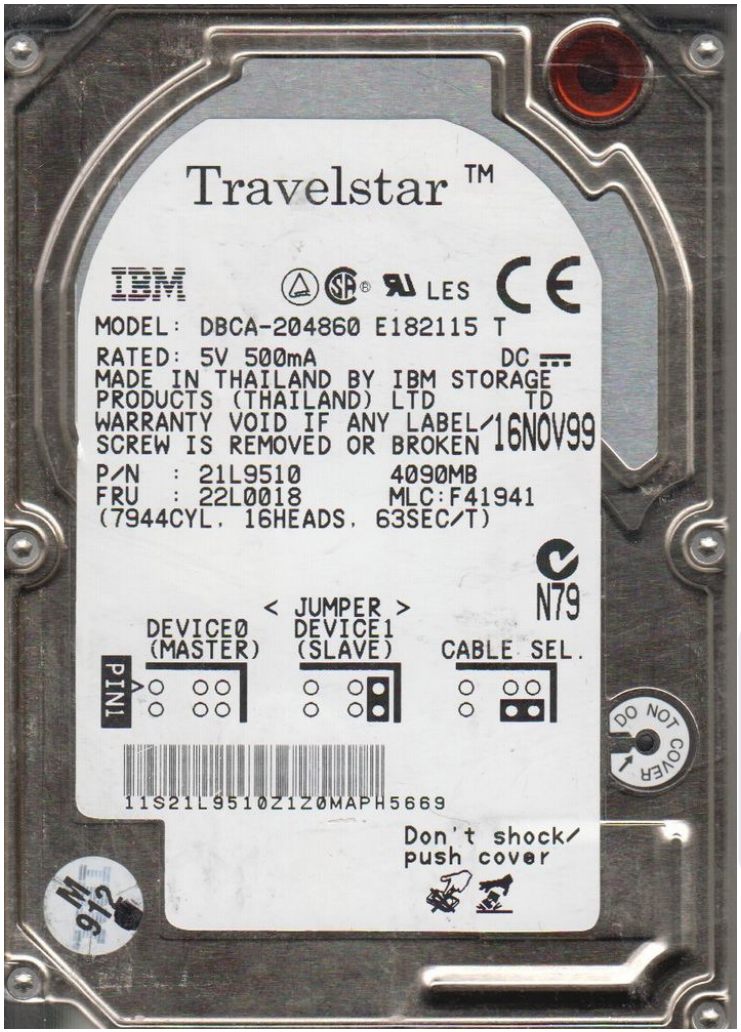
- 1.024 Zylinder * 16 Köpfe * 63 Sektoren/Spur * 512 Bytes/Sektor = 528.482.304 Bytes
- 528.482.304 Bytes / 1024 / 1024 = 504 MB

- Problem: Keine 2,5" oder 3,5" Festplatte hat $\geq$ 16 Köpfe
- Lösung: Logische Köpfe
  - Festplatten verwenden üblicherweise 16 logische Köpfe

$\Longrightarrow$ **Erweitertes CHS** (*Extended CHS*)

Adressierung der Daten auf Festplatten (3/4)

Bildquelle: eBay



- Spätere BIOS-Versionen verwendeten **Erweitertes CHS** (*Extended CHS*)
  - Erhöht via Multiplikation die Anzahl der Köpfe auf bis zu 255 und verringert die Anzahl der Zylinder um den gleichen Faktor
  - Dadurch sind Kapazitäten bis 7,844 GB möglich

- 1.024 Zylinder * 255 Köpfe * 63 Sektoren/Spur * 512 Bytes/Sektor = 8.422.686.720 Bytes
- 8.422.686.720 Bytes / 1.024 / 1.024 / 1.024 = 7,844 GB

Bessere Erklärung $\Longrightarrow$ siehe nächste Folie

Erweitertes CHS – Bessere Erklärung (1/2)

Standard	Max. Cylinders	Max. Heads	Max. Sectors	Max. Capacity
IDE/ATA	65,536	**16**	256	128 GB
BIOS	**1,024**	256	**63**	7.88 GB
Combination (Smaller of Each)	**1,024**	**16**	**63**	504 MB

- The IDE/ATA standard allows more cylinders than the BIOS does, and the BIOS allows more heads than IDE/ATA does
- Remember: These are logical disk parameters, not physical ones
- The BIOS takes the logical geometry that the hard disk specifies according to the IDE/ATA standard, and translates it into an equivalent geometry that will „fit“ into the maximums allowed by the BIOS
- This is done by **dividing the number of logical cylinders by an integer**, and then **multiplying the number of logical heads by the same number**

Quelle: `http://www.pcguide.com/ref/hdd/bios/modesECHS-c.html`

Erweitertes CHS – Bessere Erklärung (2/2)

Standard	Max. Cylinders	Max. Heads	Max. Sectors	Max. Capacity
IDE/ATA	65,536	**16**	256	128 GB
BIOS	**1,024**	256	**63**	7.88 GB
Combination (Smaller of Each)	**1,024**	**16**	**63**	504 MB

- Let's take the case of a 3.1 GB Western Digital Caviar hard drive, AC33100
- This drive actually has a capacity of 2.95 binary GB, and logical geometry of 6,136 cylinders, 16 heads and 63 sectors. This is well within the bounds of the IDE/ATA limitations, but exceeds the BIOS limit of 1,024 cylinders
- The BIOS picks a translation factor such that dividing the logical number of cylinders by this number will produce a number of cylinders below 1,024
- Usually one of 2, 4, 8, or 16 are selected; in this case the optimal number is 8
- The BIOS divides the number of cylinders by 8 and multiplies the number of heads by 8
- This results in a translated geometry of 767 cylinders, 128 heads and 63 sectors. The capacity is of course unchanged, and the new geometry fits quite nicely into the BIOS limits

Standard	Max. Cylinders	Max. Heads	Max. Sectors	Max. Capacity
IDE/ATA	65,536	16	256	128 GB
Hard Disk Logical Geometry	**6,136**	**16**	**63**	**2.95 GB**
BIOS Translation Factor	**divide by 8**	**multiply by 8**	—	—
BIOS Translated Geometry	**767**	**128**	**63**	**2.95 GB**
BIOS	1,024	256	63	7.88 GB

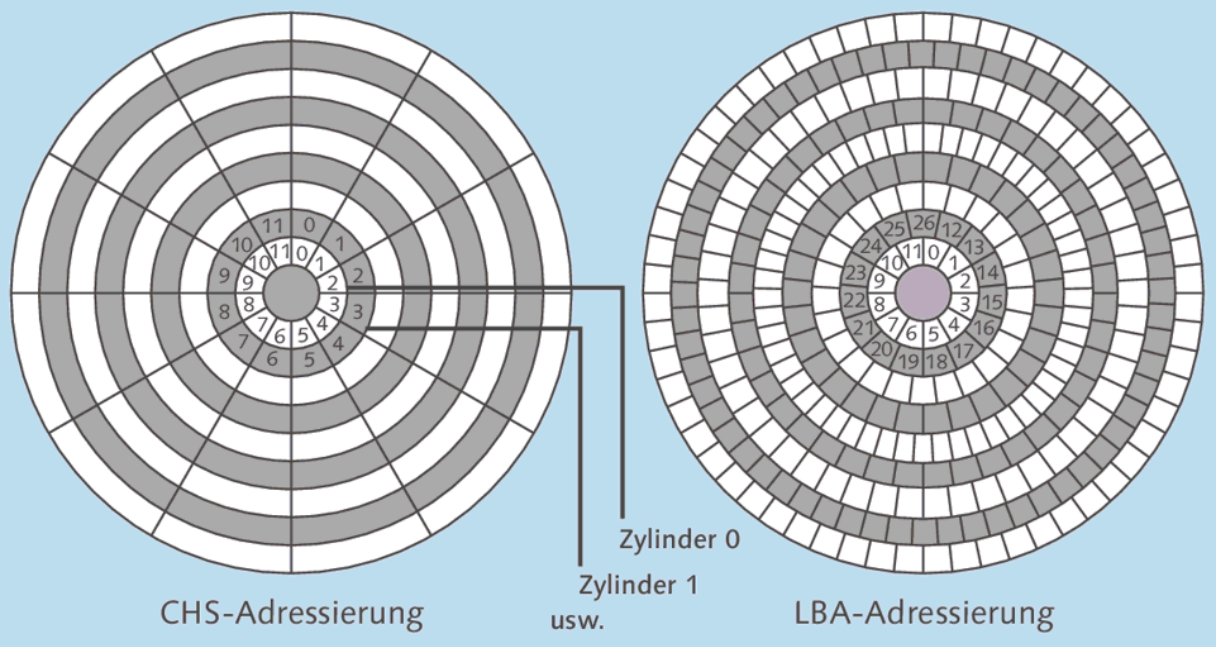
Quelle: `http://www.pcguide.com/ref/hdd/bios/modesECHS-c.html`

Adressierung der Daten auf Festplatten (4/4)

- Festplatten $>$ 7,844 GB verwenden logische Blockadressierung **Logical Block Addressing** (LBA)
  - Alle Sektoren werden von 0 beginnend durchnummeriert
- Aus Kompatibilitätsgründen können bei allen Festplatten $>$ 7,844 GB die ersten 7,844 GB via CHS adressiert werden

Logical Block Addressing (LBA)

Bildquelle: Sascha Kersken (Rheinwerk Verlag)



- Bei CHS-Adressierung sind alle Spuren (Tracks) in **gleich viele Sektoren** unterteilt
  - Jeder Sektor speichert 512 Bytes Nutzdaten
- Nachteil: Es wird **Speicherkapazität verschwendet**, weil die Datendichte nach außen hin immer weiter abnimmt
- Bei LBA existiert dieser Nachteil nicht

Zugriffszeit bei Festplatten

- Die Zugriffszeit ist ein wichtiges Kriterium für die Geschwindigkeit
- 2 Faktoren sind für die Zugriffszeit einer Festplatte verantwortlich
  1. **Suchzeit** (*Average Seek Time*)
     - Die Zeit, die der Schwungarm braucht, um eine Spur zu erreichen
     - Liegt bei modernen Festplatten zwischen 5 und 15 ms
  2. **Zugriffsverzögerung durch Umdrehung** (*Average Rotational Latency Time*)
     - Verzögerung durch die Drehgeschwindigkeit bis der Schreib-/Lesekopf den gewünschten Block erreicht
     - Hängt ausschließlich von der Drehgeschwindigkeit der Scheiben ab
     - Liegt bei modernen Festplatten zwischen 2 und 7,1 ms

$$\text{Zugriffsverzögerung durch Umdrehung [ms]} = \frac{30.000}{\text{Drehgeschwindigkeit [U/min]}}$$

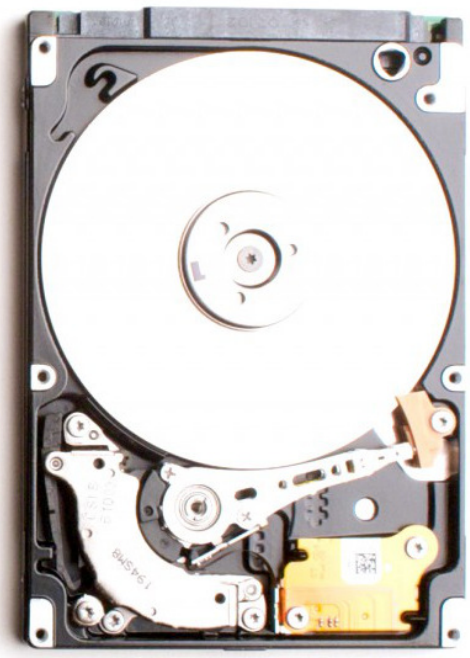
Solid State Drives (SSD)

Bildquelle: http://hardwrk.com

- Werden manchmal fälschlicherweise Solid State Disks genannt

- Enthalten keine beweglichen Teile
- Vorteile:
  - Kurze Zugriffszeit
  - Geringer Energieverbrauch
  - Keine Geräuschentwicklung
  - Mechanische Robustheit
  - Geringes Gewicht
  - Die Position der Daten ist irrelevant $\Longrightarrow$ Defragmentieren ist sinnlos

Linke Abbildung: SSD

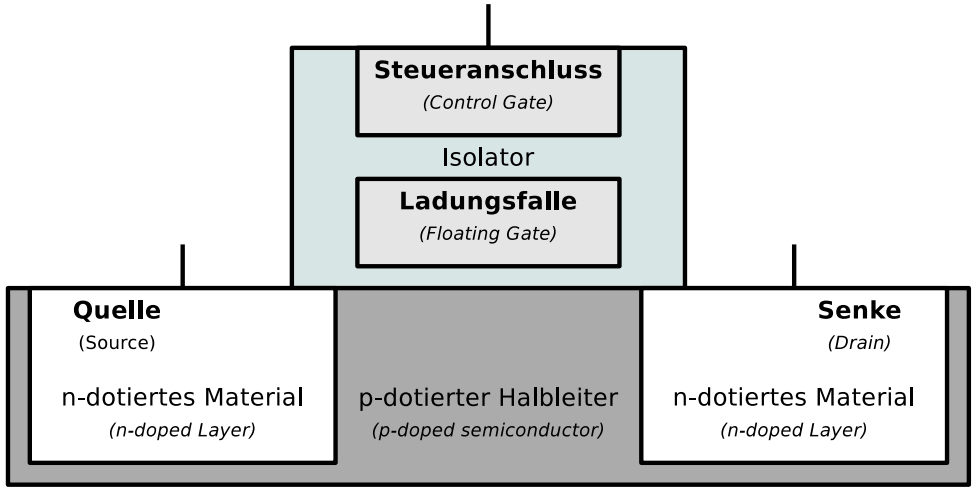


Rechte Abbildung: HDD

- Nachteile:
  - Höherer Preis im Vergleich zu Festplatten gleicher Kapazität
  - Sicheres Löschen bzw. Überschreiben ist schwierig
  - Eingeschränkte Anzahl an Schreib-/Löschzyklen

Arbeitsweise von Flash-Speicher

- Daten werden als elektrische Ladungen gespeichert
- Im Gegensatz zum Hauptspeicher ist kein Strom nötig, um die Daten im Speicher zu halten



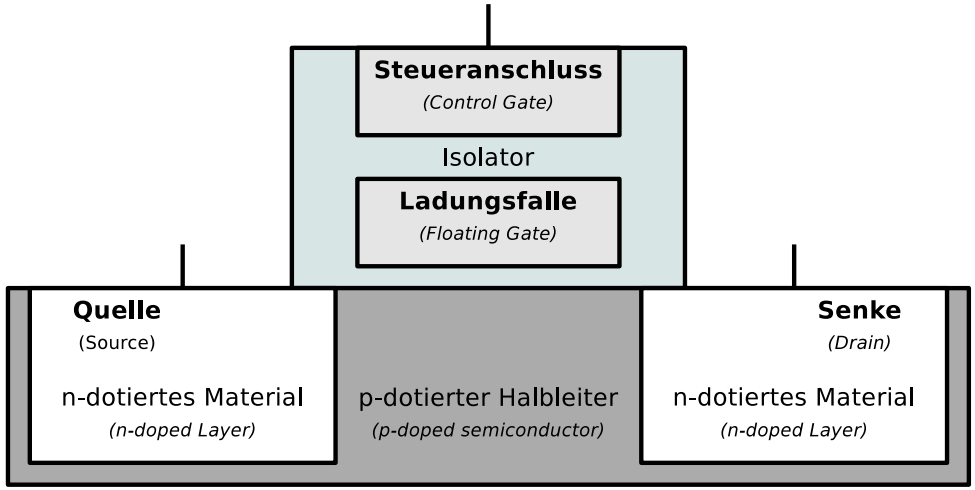
- Jede Flash-Speicherzelle ist ein Transistor und hat 3 Anschlüsse
  - **Gate** (deutsch: *Tor*) = Steuerelektrode
  - **Drain** (deutsch: *Senke*) = Elektrode
  - **Source** (deutsch: *Quelle*) = Elektrode
- Das Floating-Gate speichert Elektronen (Daten)
  - Ist komplett von einem Isolator umgeben
  - Die Ladung bleibt über Jahre stabil

Sehr gute Erklärung zur Arbeitsweise von Flash-Speicher

Benjamin Benz. *Die Technik der Flash-Speicherkarten*. c't 23/2006

Daten aus Flash-Speicherzellen lesen

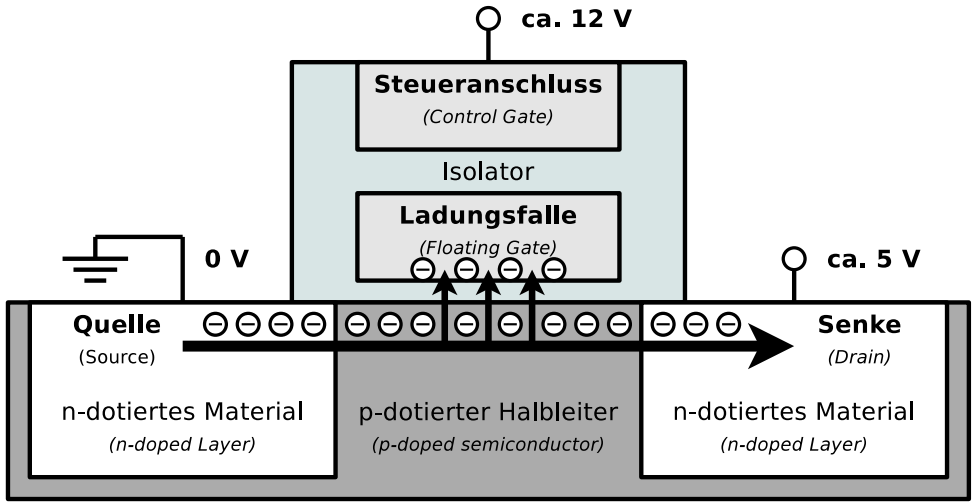
- Ein positiv-dotierter (p) Halbleiter trennt die beiden negativ-dotierten (n) Elektroden Drain und Source
  - Wie beim npn-Transistor ohne Basisstrom leitet der npn-Übergang nicht



- Ab einer bestimmten positiven Spannung (5V) am Gate (**Threshold**) entsteht im p-Bereich ein n-leitender Kanal
  - Durch diesen kann Strom zwischen Source und Drain fließen
- Sind Elektronen im Floating-Gate, verändert das den Threshold
  - Es ist eine höhere positive Spannung am Gate nötig, damit Strom zwischen Source und Drain fließen kann
    - **So wird der gespeicherte Wert der Flash-Speicherzelle ausgelesen**

Daten in Flash-Speicherzellen schreiben

- Flash-Speicherzellen werden durch den **Fowler-Nordheim-Tunneleffekt** beschrieben



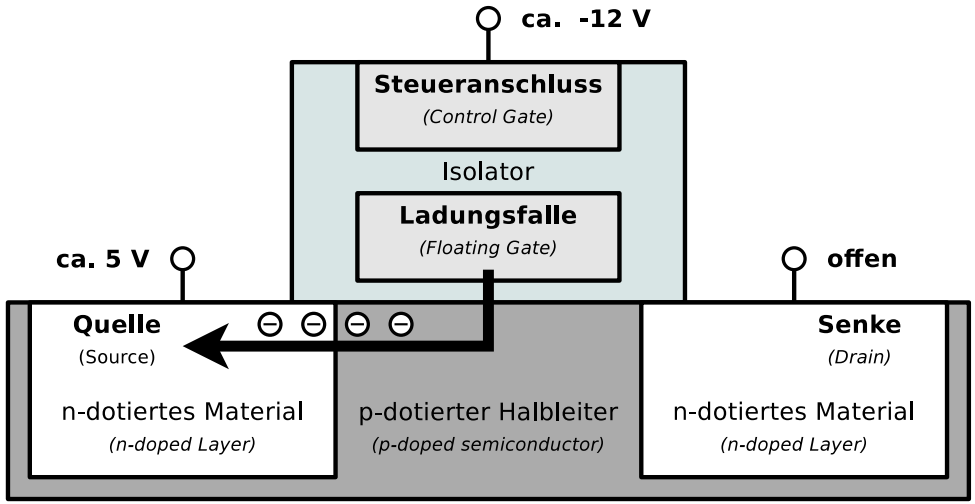
- Eine positive Spannung (5V) wird am Control-Gate angelegt
  - Darum können Elektronen zwischen Source und Drain fließen
- Ist die positive Spannung am Control-Gate groß genug (6 bis 20V), werden einige Elektronen durch den Isolator in das Floating-Gate getunnelt ($\Longrightarrow$ Fowler-Nordheim-Tunnel)
- Das Verfahren heißt auch **Channel Hot Electron Injection**

Empfehlenswerte Quelle

Flash memory. *Alex Paikin*. 2004. `http://www.hitequest.com/Kiss/Flash_terms.htm`

Daten in Flash-Speicherzellen löschen

- Um eine Flash-Speicherzelle zu löschen, wird eine negative Spannung (-6 bis -20V) am Control-Gate angelegt
  - Die Elektronen werden dadurch in umgekehrter Richtung aus dem Floating-Gate herausgetunnelt



- Die isolierende Schicht, die das Floating-Gate umgibt, leidet bei jedem Löschvorgang
  - Irgendwann ist die isolierende Schicht nicht mehr ausreichend, um die Ladung im Floating-Gate zu halten
  - Darum überlebt Flash-Speicher nur eine eingeschränkte Anzahl Schreib-/Löschzyklen

Arbeitsweise von Flash-Speicher

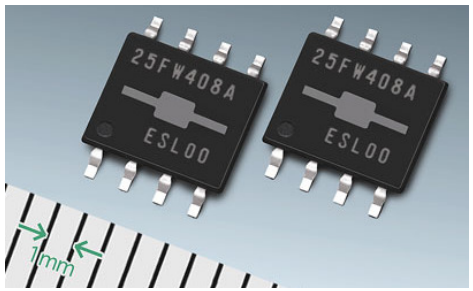
- Die Speicherzellen sind in Gruppen zu **Blöcken** und (abhängig vom Aufbau auch in **Seiten**) angeordnet
  - Ein Block enthält immer eine feste Anzahl an Seiten
- Schreib- und Löschoperationen können nur für komplette Seiten oder Blöcke durchgeführt werden
  - Darum sind Schreib- und Löschoperationen aufwendiger als Leseoperationen
- Sollen Daten in einer Seite verändert werden, muss der komplette Block gelöscht werden
  - Dafür wird der Block in einen Pufferspeicher kopiert
  - Im Pufferspeicher werden die Daten verändert
  - Danach wird der Block im Flash-Speicher gelöscht
  - Abschließend wird der veränderte Block in den Flash-Speicher geschrieben

Unterschiede beim Flash-Speicher

- Es existieren 2 Arten von Flash-Speicher:
  - **NOR-Speicher**
  - **NAND-Speicher**
- Das Schaltzeichen bezeichnet die interne Verbindung der Speicherzellen
  - Das beeinflusst Kapazität und Zugriffsgeschwindigkeit

NOR-Speicher

Bildquelle: Sanyo

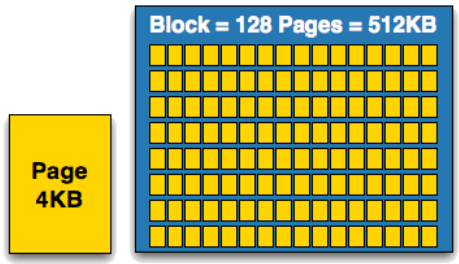


- Jede Speicherzelle hat eine eigene Datenleitung
  - Vorteil:
    - Wahlfreier Lese- und Schreibzugriff
      $\Longrightarrow$ Bessere Zugriffszeit als NAND-Speicher
  - Nachteil:
    - Komplexer ($\Longrightarrow$ kostspieliger) Aufbau
    - Höherer Stromverbrauch als NAND-Speicher
    - Üblicherweise geringe Kapazitäten ($\leq 32\,\text{MB}$)
- Enthält keine Seiten
  - Die Speicherzellen sind zu Blöcken zusammengefasst
    - Typische Blockgrößen: 64, 128 oder 256 kB
- Bei Löschoperationen ist kein wahlfreier Zugriff möglich
  - Es muss immer ein kompletter Block gelöscht werden
- Einsatzbereiche:
  - Industrielles Umfeld
  - Speicherung der Firmware eines Computersystems

NAND-Speicher

Bildquelle: engadget.com und Samsung

- Die Speicherzellen sind zu Seiten zusammengefasst
  - Typische Seitengröße: 512 bis 8.192 Bytes
    - Jede Seite hat eine eigene Datenleitung
  - Mehrere Seiten umfassen einen Block
    - Typische Blockgröße: 32, 64, 128 oder 256 Seiten

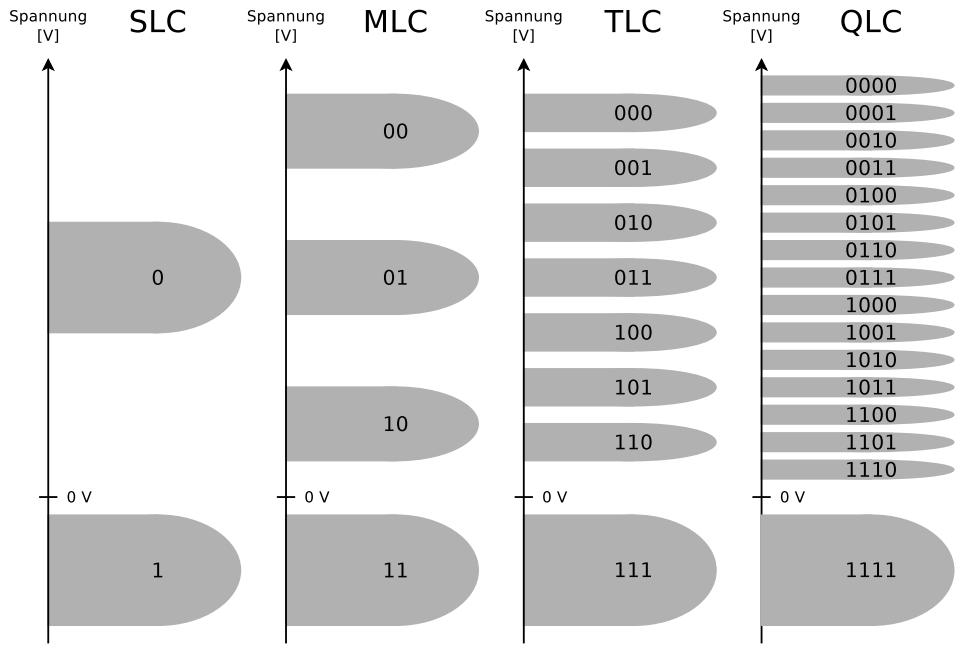


- Vorteil:
  - Weniger Datenleitungen $\Longrightarrow$ Benötigt $< 50\%$ Fläche von NOR-Speicher
  - Herstellung ist preisgünstiger im Vergleich zu NOR-Flash-Speicher
- Nachteil:
  - Kein wahlfreier Zugriff
    $\Longrightarrow$ Schlechtere Zugriffszeit als NOR-Speicher
  - Lese- und Schreibzugriffe sind nur für ganze Seiten möglich
  - Löschoperationen sind nur für ganze Blöcke möglich

- Einsatzbereiche: USB-Sticks, SSDs, Speicherkarten

Single/Multi/Triple/Quad-Level Cell

- 4 Arten von NAND-Flash-Speicher existieren
  - QLC-Zellen speichern 3 Bits
  - TLC-Zellen speichern 3 Bits
  - MLC-Zellen speichern 2 Bits
  - SLC-Zellen speichern 1 Bit
- SLC-Speicher...
  - ist am teuersten
  - hat die höchste Schreibgeschwindigkeit
  - hat die höchste Lebensdauer (überlebt die meisten Schreib-/Löschzyklen)

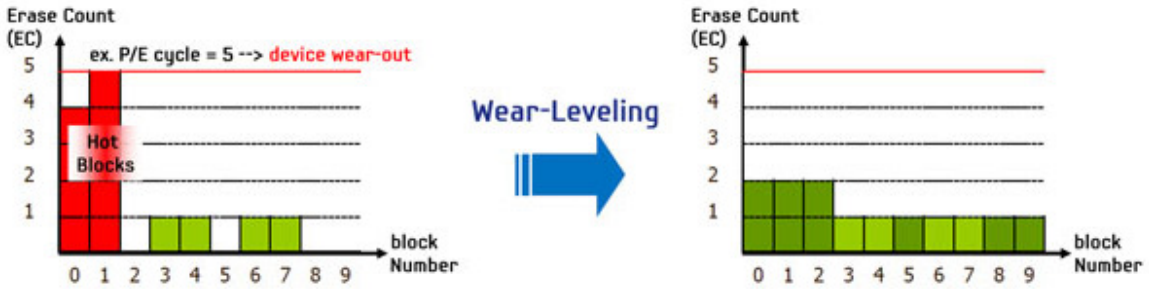


- SLC-Speicher überlebt ca. 100.000 bis 300.000 Schreib-/Löschzyklen
- MLC-Speicher überlebt ca. 10.000 Schreib-/Löschzyklen
- TLC-Speicher und QLC-Speicher überleben ca. 1.000 Schreib-/Löschzyklen
- Es existieren auch Speicherzellen, die mehrere Millionen Schreib-/Löschzyklen verkraften

Wear Leveling

Bildquelle: http://notebookitalia.it

- Wear Leveling-Algorithmen verteilen Schreibzugriffe gleichmäßig



- Dateisysteme, die speziell für Flash-Speicher ausgelegt sind, und darum Schreibzugriffe minimieren, sind u.a. JFFS, JFFS2, YAFFS und LogFS
  - JFFS enthält einen eigenen Wear Leveling-Algorithmus
    - Das ist bei eingebetteten Systemen häufig nötig, wo Flash-Speicher direkt angeschlossen wird

Zugriffszeiten bei Festplatten

- Die Geschwindigkeit von Prozessoren, Cache und Hauptspeicher wächst schneller als die Zugriffsgeschwindigkeit der Festplatten:
  - **Festplatten**
    1973: IBM 3340, 30 MB Kapazität, 30 ms Zugriffszeit (*Latenz*)
    1989: Maxtor LXTI00S, 96 MB Kapazität, 29 ms Zugriffszeit
    1998: IBM DHEA-36481, 6 GB Kapazität, 16 ms Zugriffszeit
    2006: Maxtor STM320820A, 320 GB Kapazität, 14 ms Zugriffszeit
    2011: Western Digital WD30EZRSDTL, 3 TB Kapazität, 8 ms Zugriffszeit
    2018: Seagate BarraCuda Pro ST14000DM001, 14 TB Kapazität, 4-5 ms Zugriffszeit
  - **Prozessoren**
    1971: Intel 4004, 740 kHz Taktfrequenz
    1989: Intel 486DX, 25 Mhz Taktfrequenz
    1997: AMD K6-2, 550 Mhz Taktfrequenz
    2007: AMD Opteron Santa Rosa F3, 2,8 GHz Taktfrequenz
    2010: Core i7 980X Extreme (6 Cores), 3,33 Ghz Taktfrequenz
    2018: Ryzen Threadripper 2990WX (32 Cores), 3 Ghz Taktfrequenz
- Die Zugriffszeit von SSDs ist $\leq 1\,\mu s \Longrightarrow \approx$ 100x besser als bei HDDs
  - Dennoch vergrößert sich der Abstand in Zukunft weiter wegen der Leistungsgrenzen der Schnittstellen und Mehrkernprozessoren
- Weitere Herausforderung
  - Laufwerke können ausfallen $\Longrightarrow$ Gefahr des Datenverlustes
- **Zugriffszeit** und **Datensicherheit** bei HDDs/SSDs erhöhen $\Longrightarrow$ **RAID**

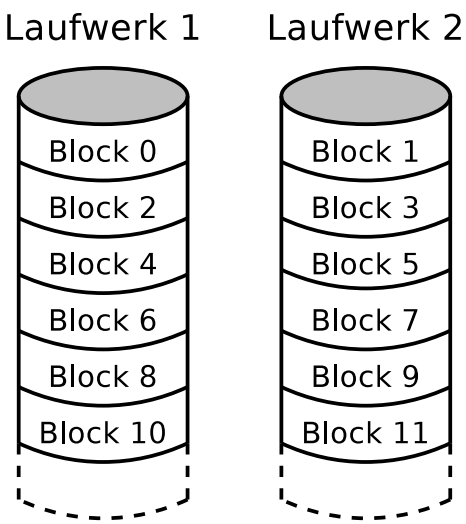
Redundant Array of independent Disks (RAID)

- Die Geschwindigkeit der Festplatten lässt sich nicht beliebig verbessern
  - Festplatten bestehen aus beweglichen Teilen
    - Physikalische und materielle Grenzen müssen akzeptiert werden
- Eine Möglichkeit, die gegebenen Beschränkungen im Hinblick auf Geschwindigkeit, Kapazität und Datensicherheit zu umgehen, ist das gleichzeitige Verwenden mehrerer Komponenten
- Ein RAID besteht aus mehreren Laufwerken (Festplatten oder SSDs)
  - Diese werden vom Benutzer und den Prozessen als ein einziges großes Laufwerk wahrgenommen
- Die Daten werden über die Laufwerke eines RAID-Systems verteilt
  - Das RAID-Level spezifiziert, wie die Daten verteilt werden
    - Die gebräuchlichsten RAID-Level sind RAID 0, RAID 1 und RAID 5

Patterson, David A., Garth Gibson, and Randy H. Katz, **A Case for Redundant Arrays of Inexpensive Disks (RAID)**, Vol. 17. No. 3, ACM (1988)

RAID 0 – Striping – Beschleunigung ohne Redundanz

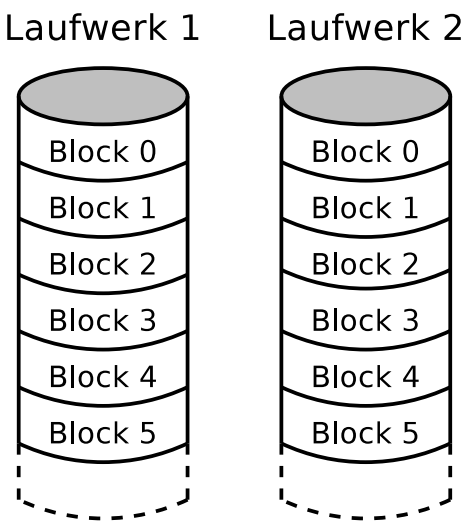
- Keine Redundanz
  - Steigert nur die Datentransferrate
- Aufteilung der Laufwerke in Blöcke gleicher Größe
- Sind die Ein-/Ausgabeaufträge groß genug ($>$ 4 oder 8 kB), können die Zugriffe parallel auf mehreren oder allen Laufwerken durchgeführt werden



- Fällt ein Laufwerk aus, können die Daten nicht mehr vollständig rekonstruiert werden
  - Nur kleinere Dateien, die vollständig auf den verbliebenen Laufwerken gespeichert sind, können gerettet werden
- RAID 0 eignet sich nur, wenn die Sicherheit der Daten bedeutungslos ist oder eine geeignete Form der Datensicherung vorhanden ist

RAID 1 – Mirroring – Spiegelung

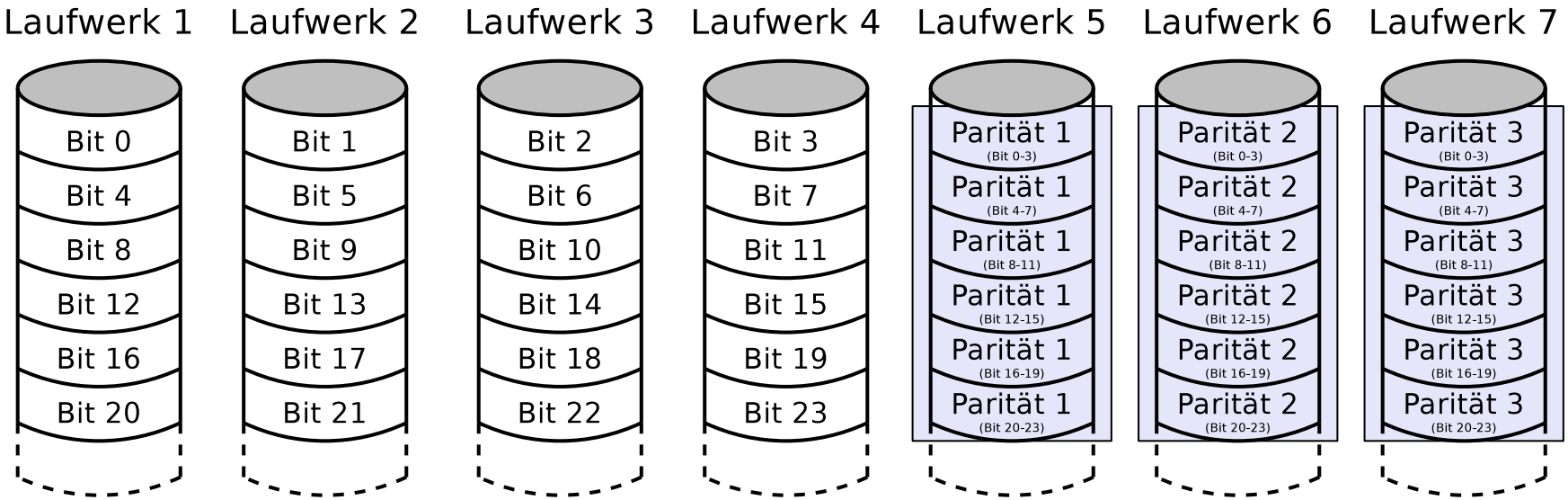
- Mindestens 2 Laufwerke gleicher Kapazität enthalten identische Daten
  - Sind die Laufwerke unterschiedlich groß, bietet ein Verbund mit RAID 1 höchstens die Kapazität des kleinsten Laufwerks
- Ausfall eines Laufwerks führt nicht zu Datenverlust
  - Grund: Die übrigen Laufwerke halten die identischen Daten vor
- Zum Totalverlust kommt es nur beim Ausfall aller Laufwerke



- Jede Datenänderung wird auf allen Laufwerken geschrieben
- Kein Ersatz für Datensicherung
  - Fehlerhafte Dateioperationen oder Virenbefall finden auf allen Laufwerken statt
- Die Lesegeschwindigkeit kann durch intelligente Verteilung der Zugriffe auf die angeschlossenen Laufwerke gesteigert werden

RAID 2 – Bit-Level Striping mit Hamming-Code-Fehlerkorrektur

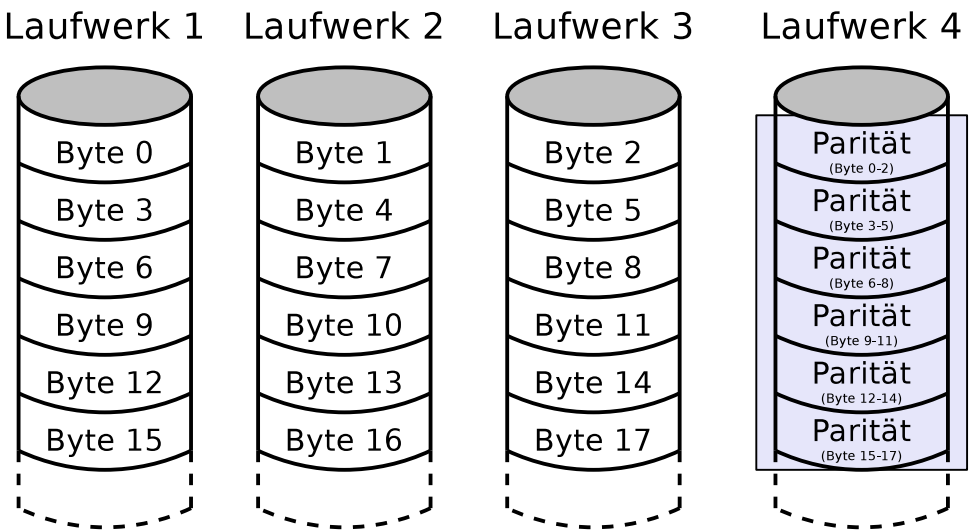
- Daten werden bitweisen auf die Laufwerke verteilt
  - Bits, die Potenzen von 2 sind (1, 2, 4, 8, 16, usw.) sind Prüfbits



- Prüfbits werden über mehrere Laufwerke verteilt $\Longrightarrow$ Datendurchsatz wird gesteigert
- Wurde nur bei Großrechnern verwendet
  - Spielt heute keine Rolle mehr

RAID 3 – Byte-Level Striping mit Paritätsinformationen

- Paritätsinformationen sind auf einem Paritätslaufwerk gespeichert

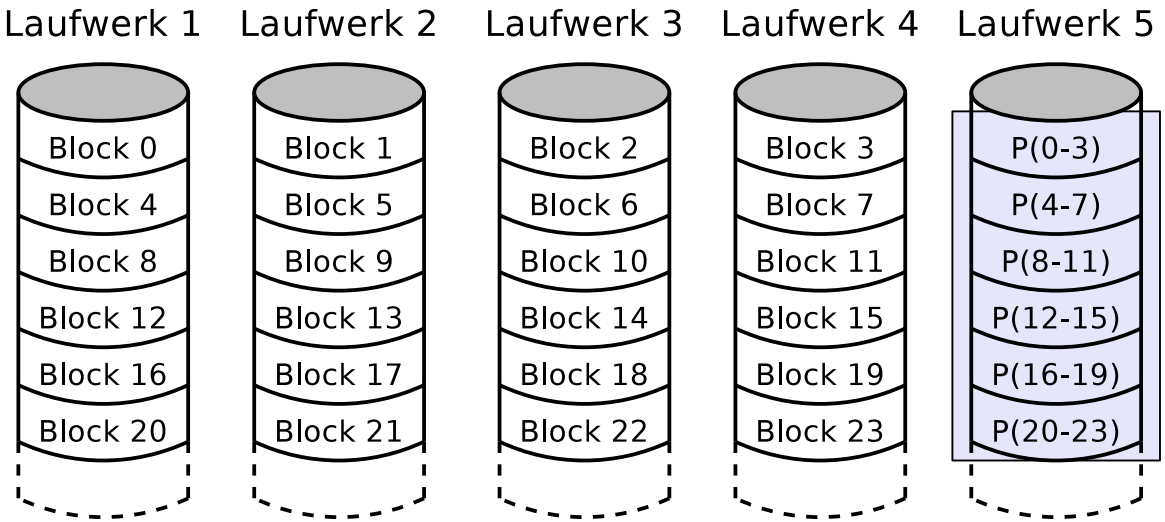


- Jede Schreiboperation auf das RAID führt zu Schreiboperationen auf das Paritätslaufwerk
  $\Longrightarrow$ Flaschenhals
- Wurde durch RAID 5 ersetzt

Datenlaufwerke		Summe		gerade/ungerade		Paritätslaufwerk
Bits sind $0+0+0$	$\Longrightarrow$	0	$\Longrightarrow$	Summe ist gerade	$\Longrightarrow$	Summen-Bit 0
Bits sind $1+0+0$	$\Longrightarrow$	1	$\Longrightarrow$	Summe ist ungerade	$\Longrightarrow$	Summen-Bit 1
Bits sind $1+1+0$	$\Longrightarrow$	2	$\Longrightarrow$	Summe ist gerade	$\Longrightarrow$	Summen-Bit 0
Bits sind $1+1+1$	$\Longrightarrow$	3	$\Longrightarrow$	Summe ist ungerade	$\Longrightarrow$	Summen-Bit 1
Bits sind $1+0+1$	$\Longrightarrow$	2	$\Longrightarrow$	Summe ist gerade	$\Longrightarrow$	Summen-Bit 0
Bits sind $0+1+1$	$\Longrightarrow$	2	$\Longrightarrow$	Summe ist gerade	$\Longrightarrow$	Summen-Bit 0
Bits sind $0+1+0$	$\Longrightarrow$	1	$\Longrightarrow$	Summe ist ungerade	$\Longrightarrow$	Summen-Bit 1
Bits sind $0+0+1$	$\Longrightarrow$	1	$\Longrightarrow$	Summe ist ungerade	$\Longrightarrow$	Summen-Bit 1

RAID 4 – Block-Level Striping mit Paritätsinformationen

- Paritätsinformationen sind auf einem Paritätslaufwerk gespeichert
- Unterschied zu RAID 3:
  - Nicht einzelne Bits oder Bytes, sondern Blöcke (**Chunks**) werden geschrieben



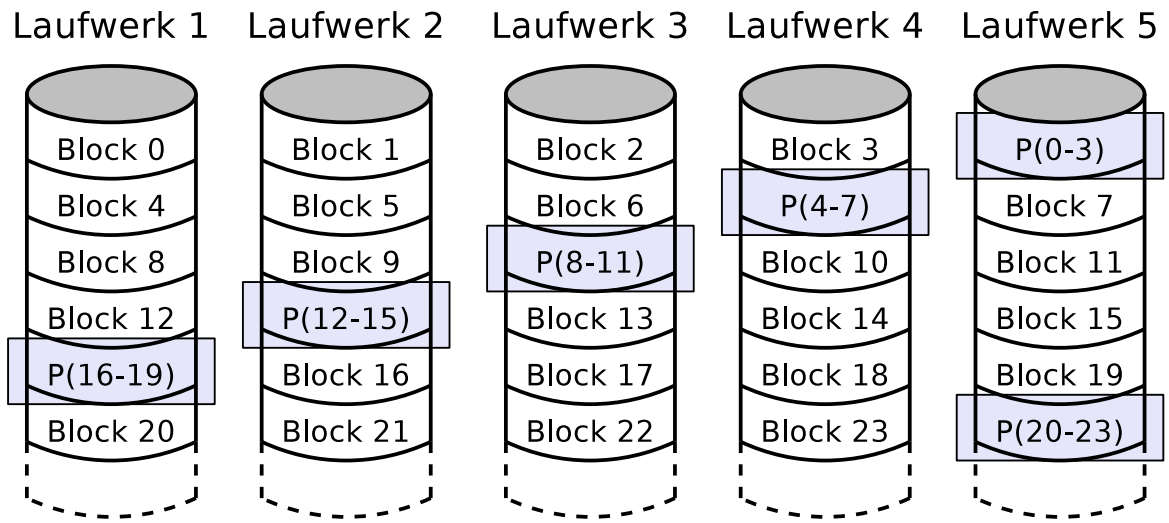
- Jede Schreiboperation auf das RAID führt zu Schreiboperationen auf das Paritätslaufwerk
  - Nachteile:
    - Flaschenhals
    - Paritätslaufwerk fällt häufiger aus

P(16-19) = Block 16 XOR Block 17 XOR Block 18 XOR Block 19

- Wird selten eingesetzt, weil RAID 5 nicht diese Nachteile hat
- Die Firma NetApp verwendet in ihren NAS-Servern RAID 4
  - z.B. NetApp FAS2020, FAS2050, FAS3040, FAS3140, FAS6080

RAID 5 – Block-Level Striping mit verteilten Paritätsinformationen

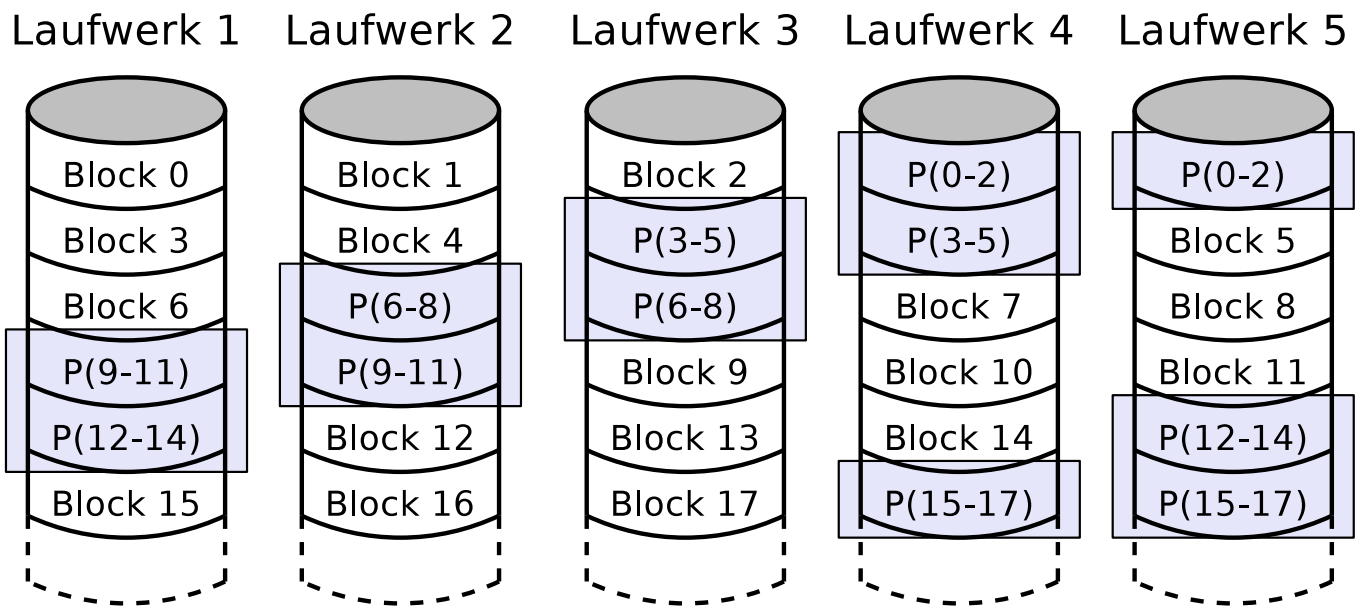
- Nutzdaten und Paritätsinformationen werden auf alle Laufwerke verteilt
- Vorteile:
  - Hoher Datendurchsatz
  - Hohe Datensicherheit
  - Kein Flaschenhals



P(16-19) = block 16 XOR block 17 XOR block 18 XOR block 19

RAID 6 – Block-Level Striping mit doppelt verteilten Paritätsinformationen

- Funktioniert ähnlich wie RAID 5
  - Verkraftet aber den gleichzeitigen Ausfall von bis zu 2 Laufwerken
- Im Gegensatz zu RAID 5...
  - ist die Verfügbarkeit höher, aber die Schreibgeschwindigkeit ist niedriger
  - ist der Schreibaufwand für die Paritätsinformationen höher



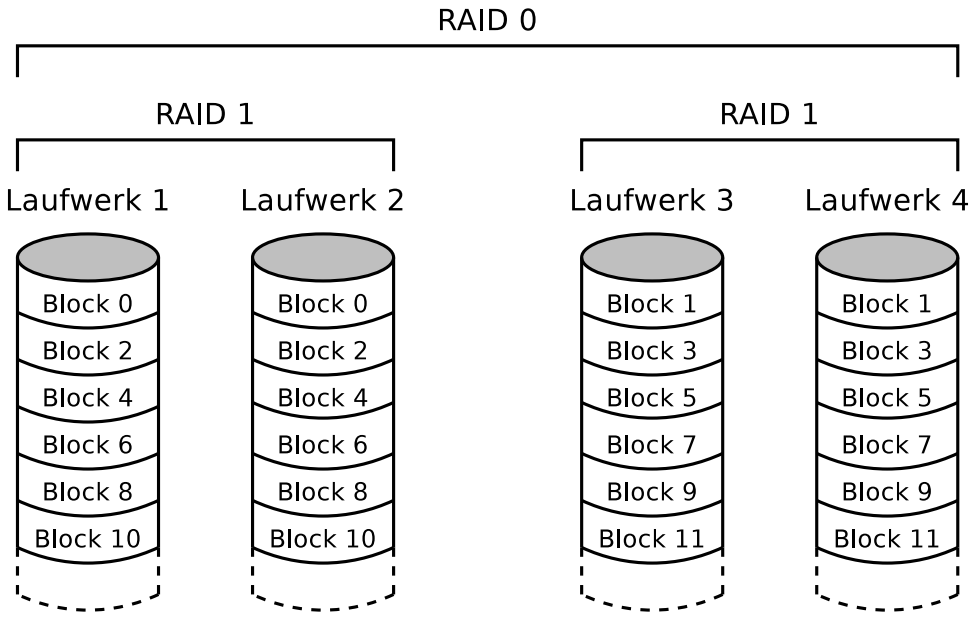
Übersicht über die RAID-Level

RAID	n (Anzahl Laufwerke)	k (Nettokapazität)	Ausfall-sicherheit	Leistung (Lesen)	Leistung (Schreiben)
0	≥ 2	n	0 (keine)	$n * X$	$n * X$
1	≥ 2	1	$n-1$ Laufwerke	$n * X$	X
2	≥ 3	$n - [\log_2 n]$	1 Laufwerk	variabel	variabel
3	≥ 3	$n-1$	1 Laufwerk	$(n-1) * X$	$(n-1) * X$
4	≥ 3	$n-1$	1 Laufwerk	$(n-1) * X$	$(n-1) * X$
5	≥ 3	$n-1$	1 Laufwerk	$(n-1) * X$	$(n-1) * X$
6	≥ 4	$n-2$	2 Laufwerke	$(n-2) * X$	$(n-2) * X$

- X ist die Leistung eines einzelnen Laufwerks beim Lesen bzw. Schreiben
- Die maximale theoretisch mögliche Leistung wird häufig vom Controller bzw. der Rechenleistung des Hauptprozessors eingeschränkt

Sind die Laufwerke in einem RAID 1 unterschiedlich groß, entspricht die Nettokapazität des RAID 1 der Kapazität seines kleinsten Laufwerks

RAID-Kombinationen



- Meist wird RAID 0, 1 oder 5 verwendet
- Zusätzlich zu den bekannten RAID-Standards (*Leveln*) existieren verschiedene RAID-Kombinationen
  - Mindestens 2 RAIDs werden zu einem größeren RAID zusammengefasst

Beispiele

- RAID 00: Mehrere RAID 0 werden zu einem RAID 0 verbunden
- RAID 01: Mehrere RAID 0 werden zu einem RAID 1 verbunden
- RAID 05: Mehrere RAID 0 werden zu einem RAID 5 verbunden
- RAID 10: Mehrere RAID 1 werden zu einem RAID 0 verbunden **(siehe Abbildung)**
- RAID 15: Mehrere RAID 1 werden zu einem RAID 5 verbunden
- RAID 50: Mehrere RAID 5 werden zu einem RAID 0 verbunden
- RAID 51: Mehrere RAID 5 werden zu einem RAID 1 verbunden

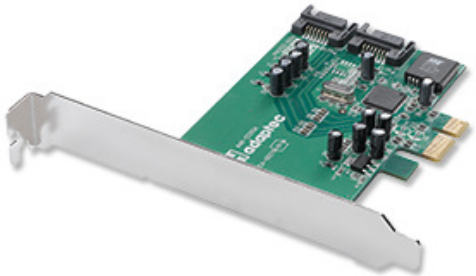
Hardware-/Host-/Software-RAID (1/2)

Bildquelle: Adaptec

Adaptec SATA RAID 2410SA

- **Hardware-RAID**
  - Ein RAID-Controller mit Prozessor berechnet die Paritätsinformationen und überwacht den Zustand des RAID

Vorteile:	Betriebssystemunabhängigkeit
	Keine zusätzliche CPU-Belastung
Nachteil:	Hoher Preis (ca. € 200)



Adaptec SATA II RAID 1220SA

- **Host-RAID**
  - Entweder ein preiswerter RAID-Controller oder der Chipsatz erbringen die RAID-Funktionalität
  - Unterstützt meist nur RAID 0 und RAID 1

Vorteile:	Betriebssystemunabhängigkeit
	Geringer Preis (ca. € 50)
Nachteile:	Zusätzliche CPU-Belastung
	Eventuelle Abhängigkeit von seltener Hardware

Hardware-/Host-/Software-RAID (2/2)

- **Software-RAID**
  - Linux, Windows und MacOS ermöglichen das Zusammenschließen von Laufwerken zu einem RAID auch ohne RAID-Controller

Vorteil:	Keine Kosten für zusätzliche Hardware
Nachteile:	Betriebssystemabhängigkeit
	Zusätzliche CPU-Belastung

- Beispiel: RAID 1 (`md0`) mit den Partitionen `sda1` und `sdb1` erstellen:

```
mdadm --create /dev/md0 --auto md --level=1
      --raid-devices=2 /dev/sda1 /dev/sdb1
```

- Informationen über alle Software-RAIDs im System erhalten:

```
cat /proc/mdstat
```

- Informationen über ein bestimmtes Software-RAID (`md0`) erhalten:

```
mdadm --detail /dev/md0
```

- Partition `sdb1` entfernen und Partition `sdc1` zum RAID hinzufügen:

```
mdadm /dev/md0 --remove /dev/sdb1
mdadm /dev/md0 --add /dev/sdc1
```