



## 6. Dateisysteme

2

## Aufgaben

- Persistente Daten
- Information
- Organisation



Aktuell Computer & Technik **Cebit 2008**

Cebit

### Ballmers Visionen

Von Holger Schmidt, Hannover



Dreidimensionale Hologramme? In den "Star Wars"-Filmen längst Realität

04. März 2008 Steve Ballmer, der Vorstandsvorsitzende des auf der Welt größten Softwarekonzerns Microsoft, hat eine Vision: Auf der Computermesse Cebit hat er nun die fünfte Computerrevolution angekündigt. „Ich habe während meiner 28 Jahre in der Computerindustrie schon vier Computer-Revolutionen erlebt“, sagte Ballmer. Mit der ersten Revolution hätten sich erstmals die Massen einen eigenen PC leisten können.

Die weiteren Meilensteine seien die Entwicklung von grafischen Benutzeroberflächen, der Aufstieg des Internets und zuletzt das interaktive „Web 2.0“ gewesen, das aus statischen Internetseiten Plattformen für den Austausch von Informationen und persönlichen Nachrichten gemacht habe. „Wenn es bei dem Sieben-Jahres-Rhythmus bleibt, dann stehen wir heute am Ende der vierten und am Beginn der fünften Revolution.“ Sie werde geprägt von enormen Rechenkapazitäten und **einem quasi unendlichen Speicherplatz**. „Hochgeschwindigkeitsverbindungen sind allgegenwärtig und die Systeme können mit Sprache und Gesten bedient werden“, sagte Ballmer.



- 5 MByte Festplatte wird mit dem Gabelstapler verladen (1956)

## Beispiel Seagate

- Liefert 1979 erste Festplatte aus
  - ST506, 5 MB Daten, \$1500
- April 2008
  - 1 Milliarde Laufwerke ausgeliefert
  - Prognostiziert 2 Milliarden in 5 Jahren
- Kummulative ausgelieferte Speicherkapazität seit 1979
  - 79.000.000 Terabytes
  - $7.5 \cdot 10^{18}$  Bytes (75 Exabyte)
- 1.5 TByte Platten (3.5 Zoll)



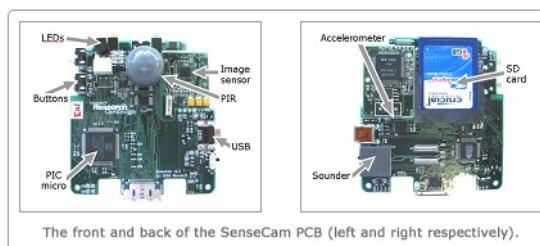
## Rechenbeispiele

- Schrift
  - Brockhaus paßt auf eine DVD
- Lebenslange Audioaufzeichnung
  - 32 Kbit pro Sekunde
  - 120 GByte pro Jahr
  - 9.6 TByte bei 80 Jahren Lebenserwartung
- Ein Leben in Einzelbildern
  - Microsoft SenseCam, 1 Tag = 7 -10 MByte
- Lebenslanges Filmarchiv
  - 4 GByte pro Stunde (DVD-Qualität)
  - 4 Stunden pro Tag (im Alter 0 .. 80)
  - 456 TByte

## Microsoft SenseCam

- Digitalkamera, die selbständig Bilder macht
  - Wahl geeigneter Zeitpunkte

- Sensoren
  - Lichtintensität
  - Farbsensoren
  - Infrarot (Körperwärme)
  - Temperatur
  - Beschleunigung (über mehrere Achsen)

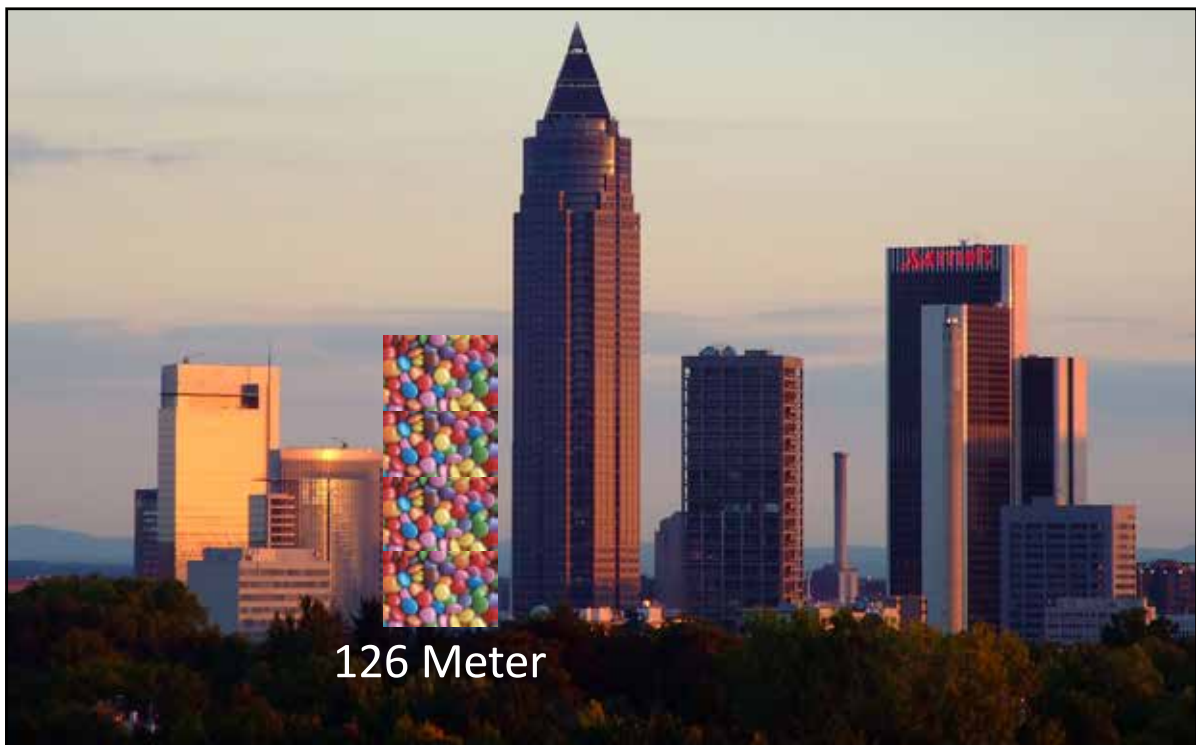


## Terabyte

- Aktuell gängige Größe

$$1024^4 = 1024 \cdot 1024 \cdot 1024 \cdot 1024 = 1.099.511.627.776$$

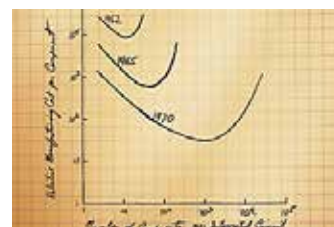
- Wieviel ist das?
  - Fußballfeld voller Smarties?



# Probleme

## Moore'sche Gesetz

- Selbsterfüllende Prophezeiung
  - Vorgabe an den Fortschritt im Entwicklungsprozeß
- Laut Intel mindestens bis 2029 gültig!



Intel

## Folgen

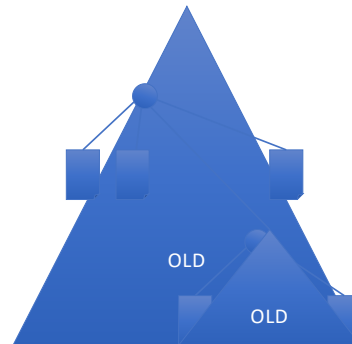
- $2029 - 2016 \approx 16 = 8 \cdot 2 \rightarrow$  Steigerung um Faktor 256
- Gilt bestimmt für Hauptspeicher
  - 1-4 TByte pro PC realistisch
- Festplatten ebenfalls
  - 500 TByte bis 1 PByte pro Platte (3.5 Zoll) erreichbar
- Smartphone mit 500 Tbyte? Möglich!
- Sollen wir so weiter machen wie bisher?

## Hibernate

- Retten des Hauptspeichereinhalts auf Festplatte
- System "schneller" wieder fortführbar
  - Einfach den Hauptspeichereinhalt wieder herstellen
- Sinnvoll bei 8 GB? 32 GB? 1 TByte?
- Ja bei nicht-flüchtigem RAM
- Nein sonst

## Speichermenge beherrschen?

- Hierarchische Ordnungsprinzipien für Menschen nur bedingt geeignet
  - ... und wer noch daran glaubt, weiß es nur noch nicht ☺
- Subjektive Erkenntnis
  - Kurz vor der Organisationskatastrophe
  - Entartete Verzeichnisstruktur
- Ja ...
  - mag sein, daß nur ich unfähig bin!



## Wiederfinden?

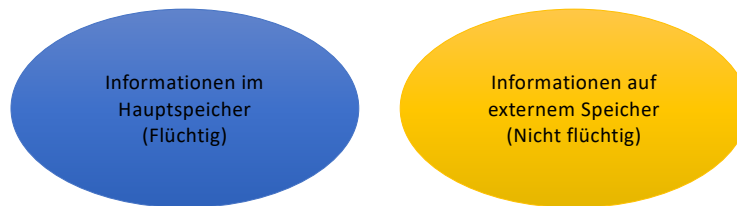
- Finden Sie alles wieder?
  - Ich nicht ☹
  - 14 Jahre gewachsenes Homeverzeichnis: > 1 TB
- Ich habe ausgedruckte Beweise, daß bestimmte Dateien mal existiert haben
  - dümpeln vielleicht in irgendwelchen Archiv-CDs rum
- Droht digitaler Alzheimer?
- Ja ...
  - im kommerziellen Umfeld wird sauber gearbeitet: hier nicht!



## Geräte-Allerlei

- Eine Vielzahl an angeschlossenen physischen Speichermedien prägen das Bild!
  - PC: Sticks, Festplatten, portable Platten, ...
  - Backend: Speicherplatz für Zig-TByte oder gar -PByte Daten
- Einzelmedium häufig isoliert
- Viele Entscheidungen beeinflussen Systemleistung
  - Welches Dateisystem für welches Gerät?
  - Kleine Änderung – große Wirkung
- Ja ...
  - Im professionellen Bereich gibt es Lösungen (z.B. Volume Manager)

## Anwendungsentwicklung



- Zwei Welten
- Externer Speicher
  - Einfacher Dateizugriff
  - SQL-Queries gegen Datenbanken
- Interner Speicher
  - Container-Klassen mit entsprechenden Zugriffsmöglichkeiten

## Beispiel Container-Klassen

```
class Datensatz
{
    public Datensatz(int a, string w)
    {
        Alter = a;
        Wohnort = w;
    }

    public int Alter;
    public string Wohnort;
}

static void Main(string[] args)
{
    Dictionary<string, Datensatz> mitarbeiter = new Dictionary<string, Datensatz>();
    mitarbeiter.Add("Sturm", new Datensatz(42, "Kons"));
    mitarbeiter.Add("Jean Luc Piccard", new Datensatz(-50, "Enterprise"));

    if (mitarbeiter.ContainsKey("Sturm"))
        Console.WriteLine("Es gibt einen Sturm");

    foreach (Datensatz d in mitarbeiter.Values)
    {
        if (d.Wohnort == "Enterprise")
            Console.WriteLine("Es wohnt jemand auf der Enterprise");
    }
}
```

## Zuverlässigkeit?



Zuverlässigkeit??



Dateisysteme

## Typische Dateinutzung

- Dateien sind klein oder sehr groß
- Lesender Zugriff überwiegt
- Sequentieller Zugriff dominiert
- (Dateien werden selten oder nie gelöscht)
- Einige Dateien werden von vielen Nutzern verwendet
  - Dabei überwiegt lesender Zugriff
- “Make the common case fast”
  - Sequentieller lesender Zugriff auf Dateien
  - Caching und Read Ahead sinnvoll

## Meta Data

- File information (inode)
  - Size, owner, access rights, timestamps
  - Where are the blocks on disk
- Directory information
  - Content, owner, access rights, timestamps
- Device information
  - root directory
  - Free blocks
  - Bad blocks

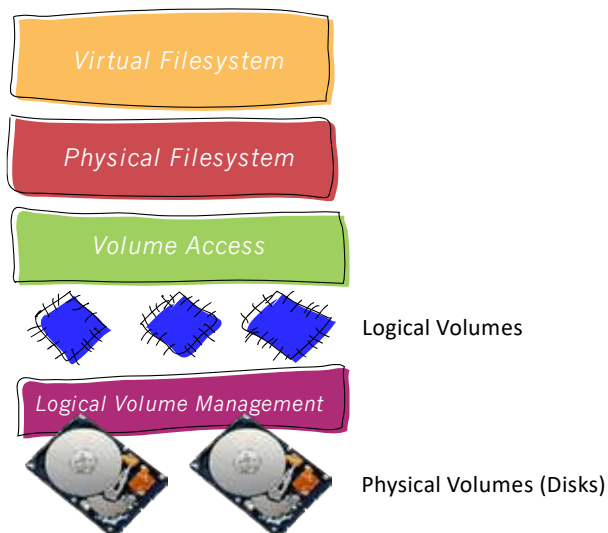
## Requirements

- *Minimize time-consuming head movement*
  - *Store file data & meta-data in close neighborhood*
  - *Collect mutiple disk requests and optimize movement*
  - *Cluster blocks consecutive blocks of a file phyiscally*
- Improve read/write performance
  - Utilize caching aggressively (disk & OS)
  - Exploit striping and mirroring
- Keep pace with increasing disk capacity
  - “Disk usage is like an ideal gas, which takes up every available space”
  - Deal with more and more files
  - Recovery in case of system failure must be fast

## Functions

- Formatting and initializing media
  - Create initial content to access media
- Keeping track of bad blocks
  - Recognize bad blocks and exclude them from future use
- Managing of meta-data
  - Small set of data but high update frequency
- Storing and retrieval of files
  - Sequential reading and writing
  - Positioning of file pointer
  - Mapping files into virtual address space
- Security
  - Who’s allowed to access a file?

## Hierarchical Structure

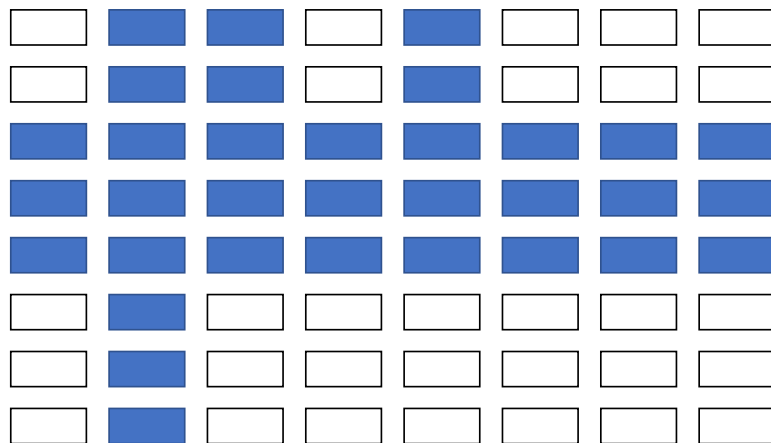


## In Place vs. Log

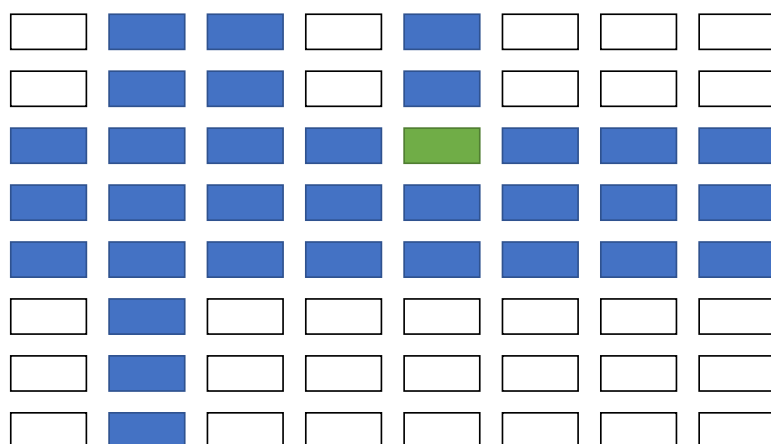
- Sometimes high update frequency
  - e.g. every access changes some timestamps
- Small updates
  - 4 bytes timestamp
- Cache changes might lead to inconsistencies
- How to update
  - In place
  - Log-based



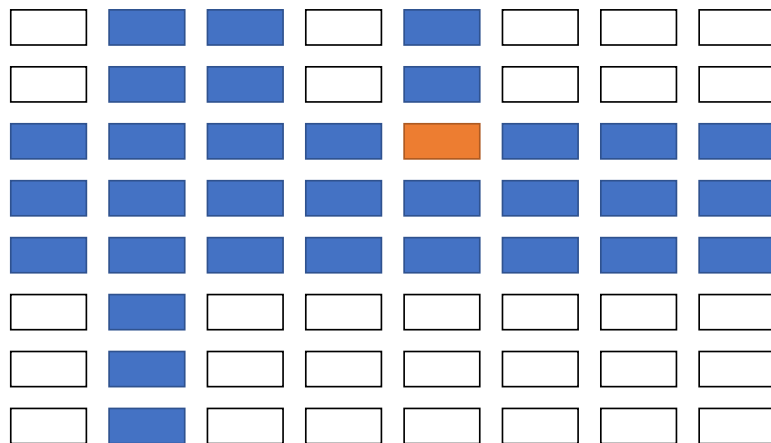
Update in Place ☹️



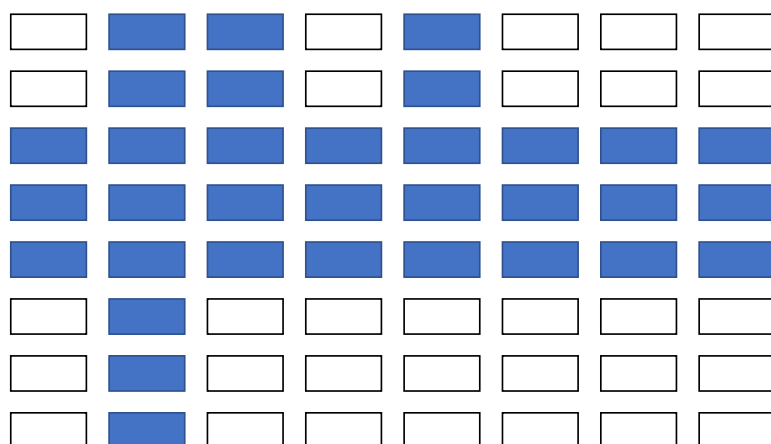
Update in Place ☹️



Update in Place ☹️

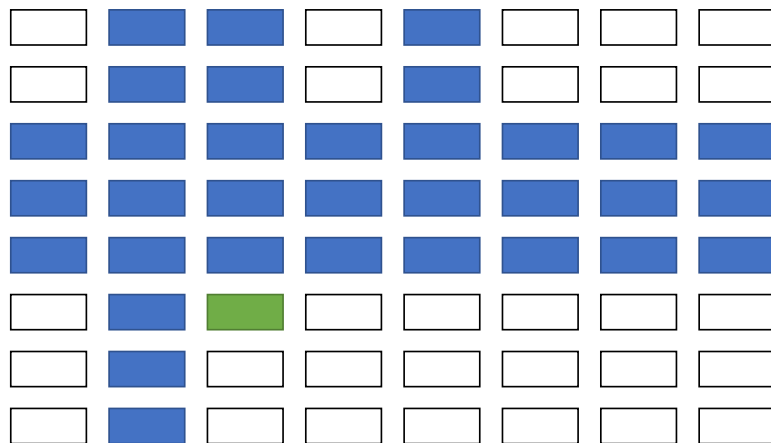


Update in Place ☹️

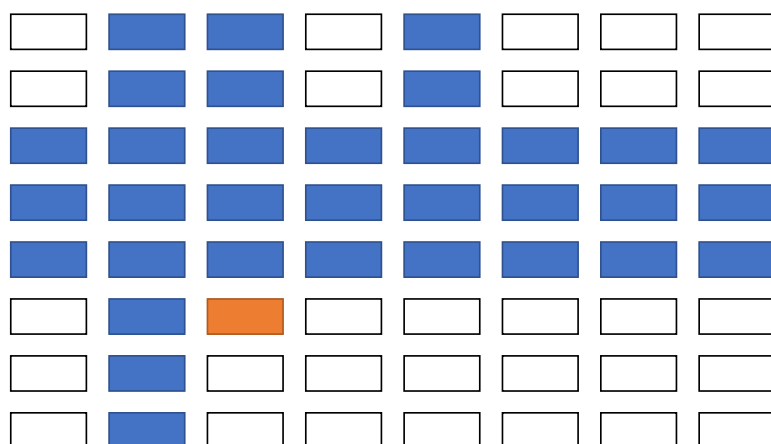




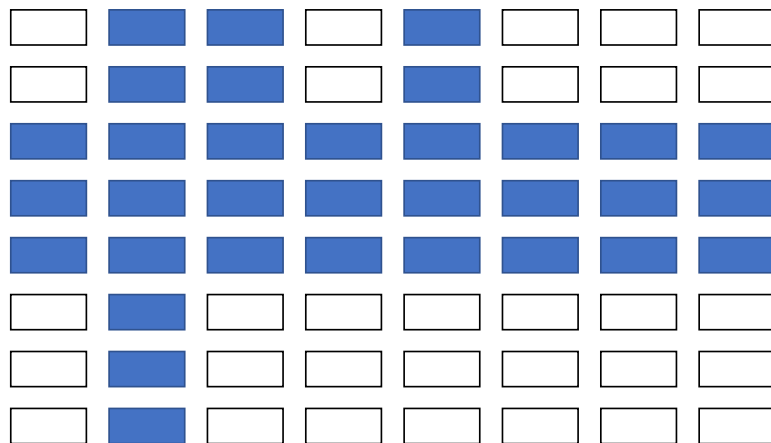
Update in Place ☹️



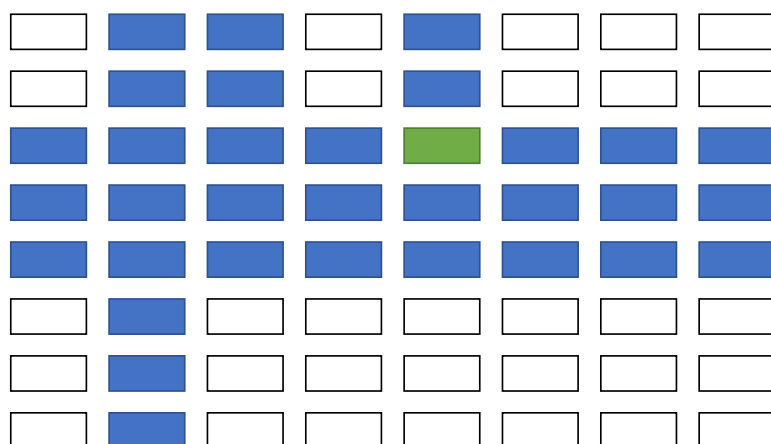
Update in Place ☹️



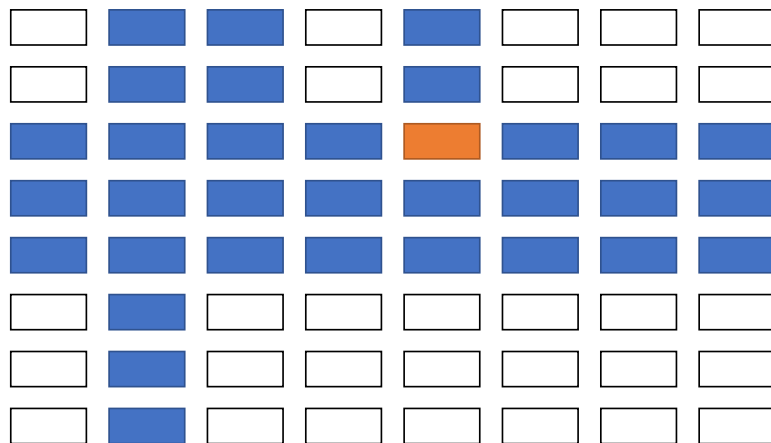
Update in Place ☹️



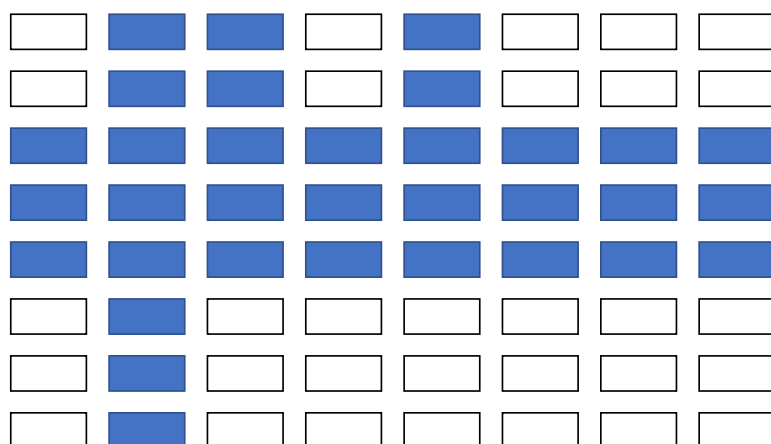
Update in Place ☹️



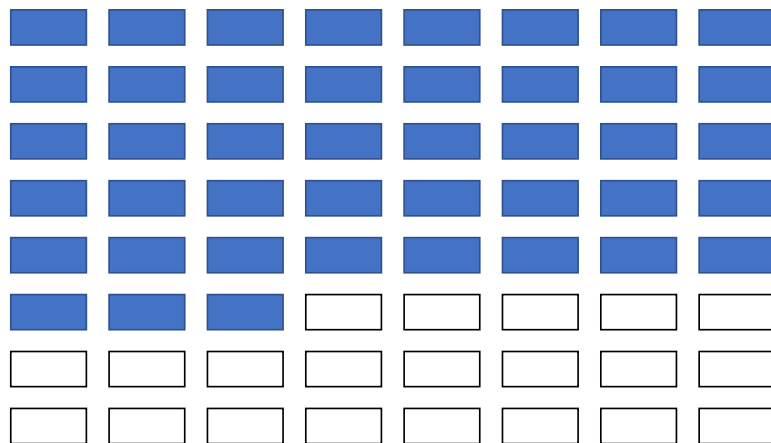
Update in Place ☹️



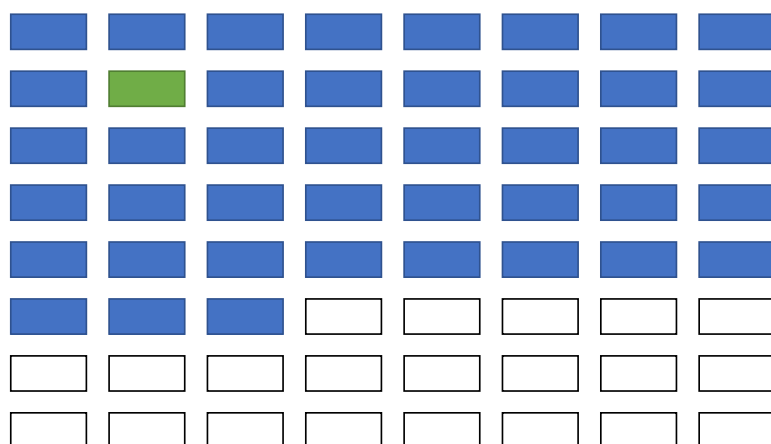
Update in Place ☹️



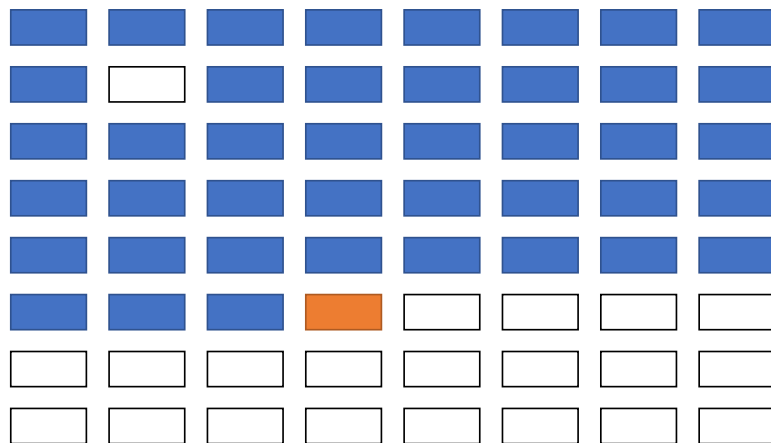
Log-Based ☺



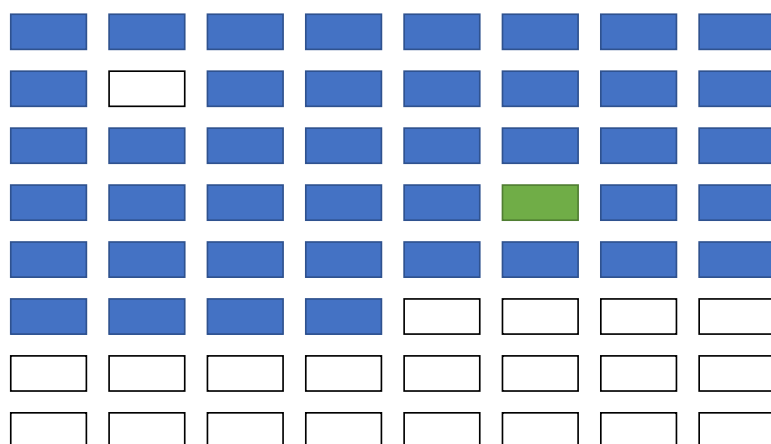
Log-Based ☺



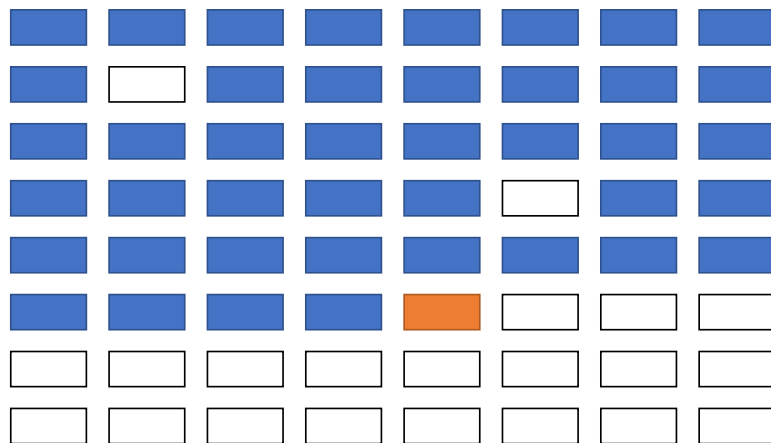
Log-Based ☺



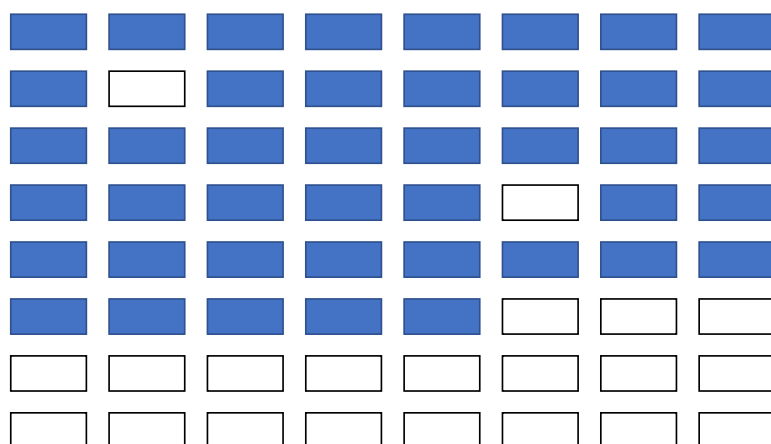
Log-Based ☺



Log-Based ☺



Log-Based ☺



## Classification

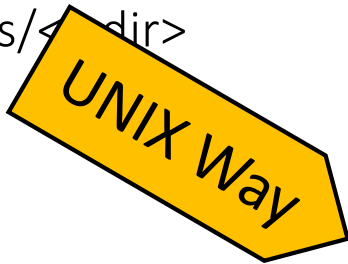
		Data	
		Update in Place	Log
Meta Data	Update in Place	Traditional filesystems ext2, FAT, ...	unknown
	Log	Journaling filesystems ext4, NTFS, HFS, XFS, ...	Log-based filesystems ZFS, btrfs, ReFS, ...

# Linux

## Linux 4.3/fs

9p adfs affs afs autofs4 befs bfs btrfs cachefiles ceph cifs  
coda configfs cramfs debugfs devpts dlm ecryptfs  
efivarfs efs exofs exportfs ext2 ext4 f2fs fat freevxfs  
fscache fuse gfs2 hfs hfsplus hostfs hpfs hugetlbfs isofs  
jbd2 jffs2 jfs kernfs lockd logfs minix ncpfs nfs  
nfs\_common nfsd nilfs2 nls notify ntfs ocfs2 omfs  
openpromfs overlayfs proc pstore qnx4 qnx6 quota  
ramfs reiserfs romfs squashfs sysfs sysv tracefs ubifs udf  
ufs xfs

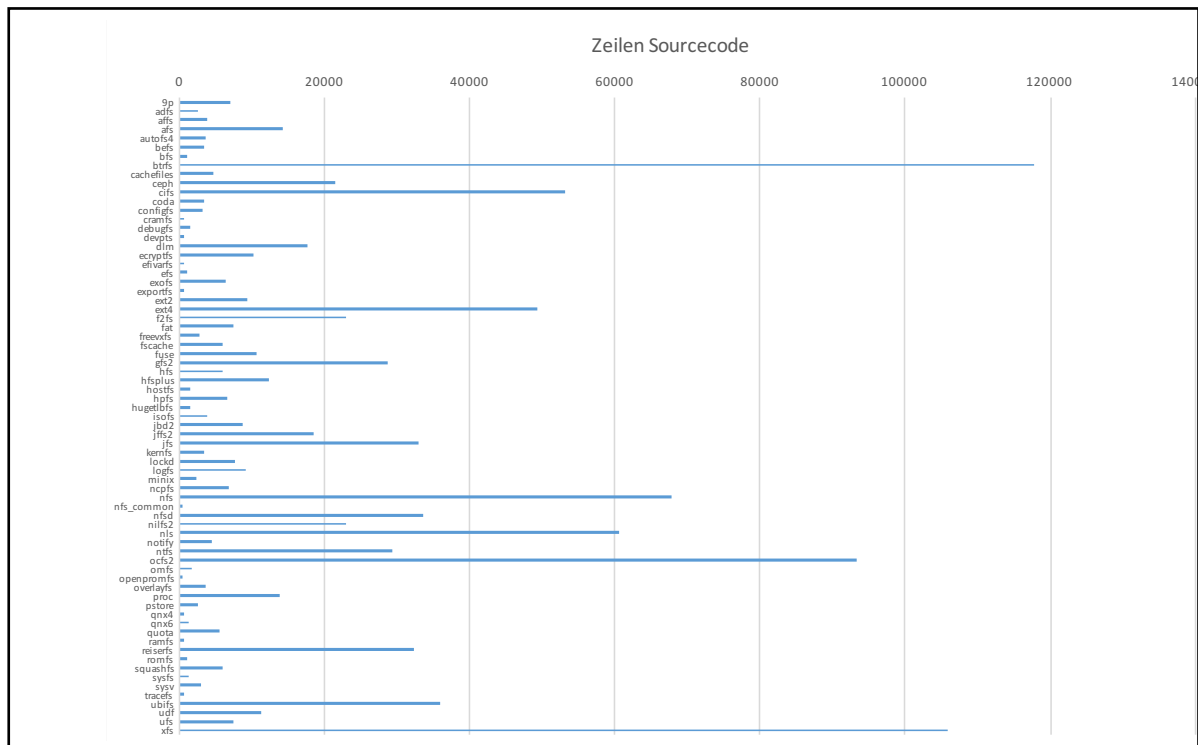
./fs/<dir>



```
find linux-4.3/fs/* -maxdepth 0 -type d |  
xargs -n 1 -J dir find dir -name "*.ch" -exec cat {} \; |  
wc
```

- Linux 4.3
  - 1.084.731 Zeilen von 4.614.490 Zeilen Kernel

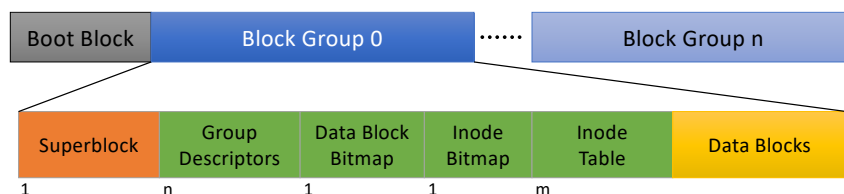




ext2

- The original Linux filesystem
  - Origin was Minix filesystem
  - Extended filesystem (inefficient extension)
  - 2nd Extended filesystem (since 1994)
- Properties
  - Selectable block size between 1024 and 4096 bytes
  - Selectable number of inodes per partition
  - Clustering of blocks
  - Preallocation of file blocks (enlargement)

## Structure



- Reason for block groups?
- Maximum size of block group:
  - Data Block Bitmap = 1 Block
  - 8 MB (1 KB blocks) up to 128 MB (4 KB blocks)
- Superblock and group descriptors are replicated in each group
  - Kernel will access block group 0 only

- Total number of inodes
- Block size
- Size of overall filesystem in blocks
- Number of reserved blocks
  - Only accessible by root processes
  - Accessing an otherwise full filesystem
- Number of free blocks and free inodes
- Characteristics of block groups
- Statistics
  - Timestamps (Last mount operation, Last write, Last filesystem check)
  - Mount statistics (Number of mount operations, Maximum mounts before check)
- Status flag
  - 0 mounted or missed unmount, 1 cleanly unmounted, 2 Filesystem has errors

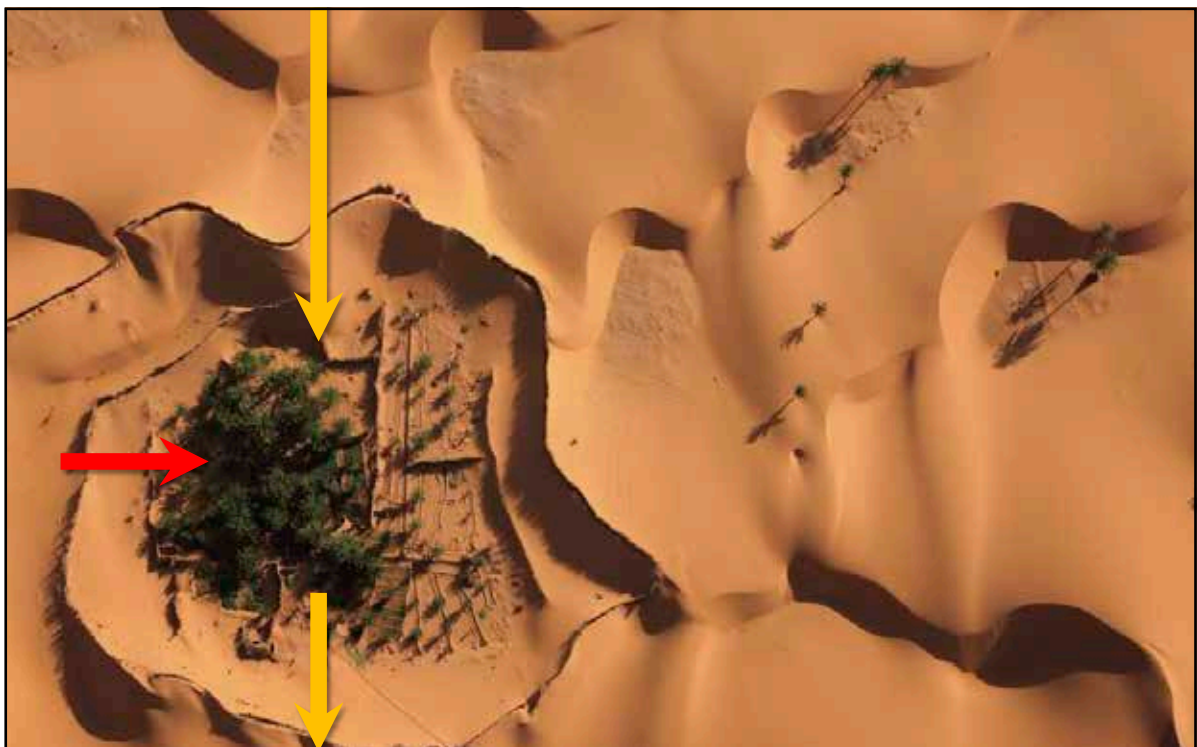
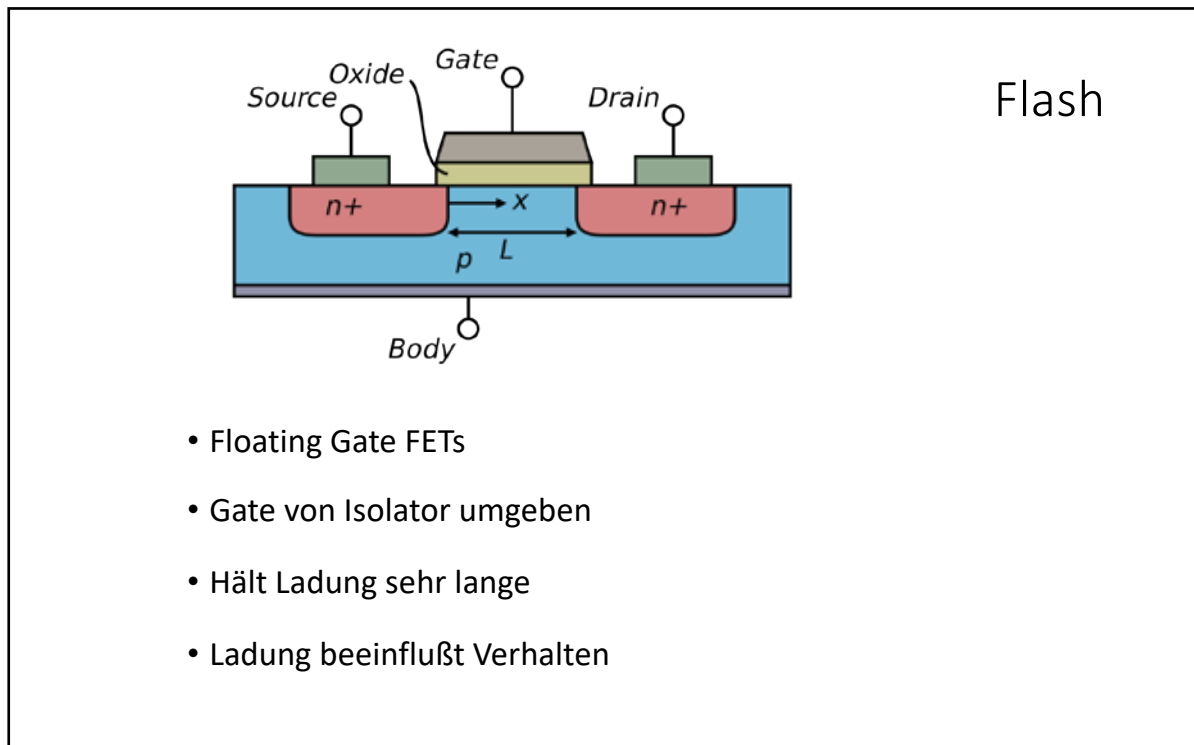
## Superblock



## ext3 & ext4

- Journaling der Metadaten
- Ext4
  - Größere Platten (1 EB)
  - Größere Dateien (16 TB statt 16 GB)
  - Größere Directories
  - Checksums, Delayed allocation, ...





## Besonderheiten

- Hohe Spannungen beim Ändern
  - 1 kann zu 0 werden
- Block-basiertes Löschen
  - 0 wieder zu 1
- Memory Wear
  - $10^5$  bis  $10^6$  Erase-Zyklen
  - Flash Translation Software Layer (FTL)
- Read Disturb (NAND)

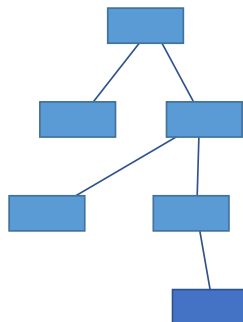
Recht verbreitet ...



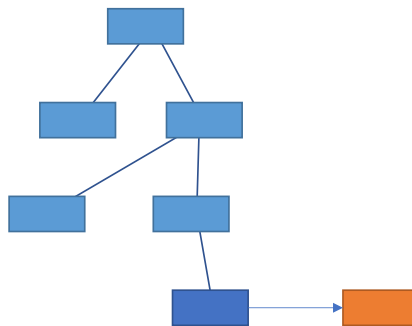
## Flash File Systems

- **CASL** is a filesystem designed by **Nimble Storage** that uses solid-state devices to cache traditional hard disk drives.
- **ETFS** - Embedded Transactional File System. Designed primarily for NAND devices by **GNX Software Systems**.
- **exFAT** - Microsoft proprietary system intended for flash cards
- **ExtremeIO** - Internal file system for SSDs.
- **F2FS** - Flash Friendly File System. An open source Linux file system introduced by **Samsung** in 2012.<sup>[6]</sup>
- **FFS2** (presumably preceded by **FFS1**), one of the earliest flash file systems. Developed and patented by **Microsoft** in the early 1990s.<sup>[7]</sup>
- **JFFS** - Original log structured Linux file system for NOR flash media
- **JFFS2** - Successor of JFFS, for NAND and NOR flash
- **LogFS** - Intended to replace JFFS2, better scalability. In early development.
- **Non-Volatile File System**—the “non-volatile file system” for flash memory introduced by **Palm, Inc.**.
- **OneFS** - OneFS is a file system utilized by **Isilon**. It supports selective placement of meta-data directly onto flash SSD.
- **RFS** - Robust File System (developed and used by **Samsung**)
- **Sogget Microcontroller Systems emFile** - File system for deeply embedded applications which supports both NAND and NOR flashes. Wear leveling, fast read and write, and very low RAM usage.
- **SafeFLASH** - HCC-Embedded - Fail-safe file system that supports NAND and NOR flash types with integrated wear-leveling and bad-block handling.
- **TFAT** - A transactional version of the FAT filesystem.
- **TrueFFS** - Internal file system for SSDs, implementing error correction, bad block re-mapping and wear leveling.
- **UBIFS** - Successor of JFFS2 optimized to utilize non-volatile DRAM
- **UFFS** - Ultra low cost flash file system for embedded system
- **Uvision RTOS** - Flash-Nand/Nor small footprint low cost flash file system for embedded systems
- **Write Anywhere File Layout** - **WAFL** is an internal file system utilized by **NetApp** within their **DataONTAP** OS, originally optimized to use non-volatile DRAM
- **XCP/ies** - an exFAT implementation from **Datalight** for **Wind River VxWorks** and other embedded operating systems
- **YAFFS** - A Log structured file system designed for NAND flash, but also used with NOR flash.
- **ZFS** - Allows placing write-ahead log (ZIL) on flash, and using flash as a second-level read cache (L2ARC)
- **OTFS** - Used in SSD

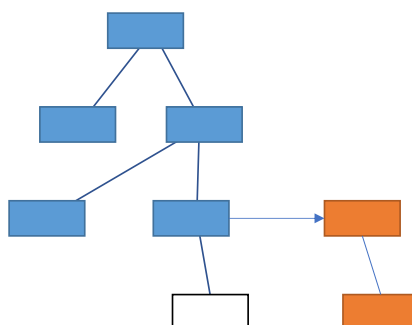
## Indexstrukturen



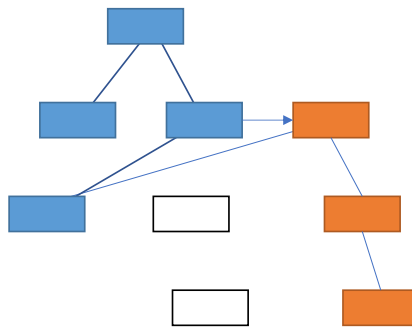
## Indexstrukturen



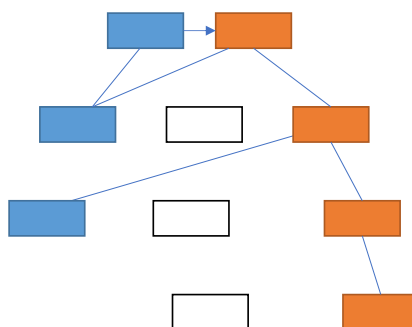
## Indexstrukturen



## Indexstrukturen

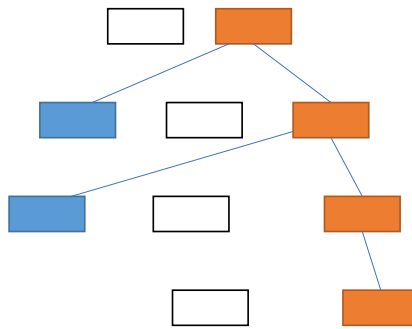


## Indexstrukturen

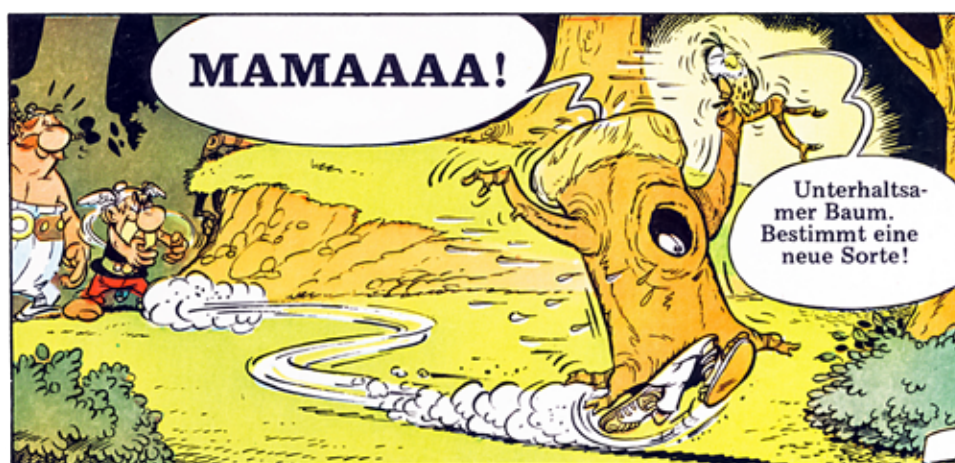




## Indexstrukturen



## Walking Trees



Der Kampf der Häuptlinge



## Flash Friendly File System



### **F2FS: A New File System for Flash Storage**

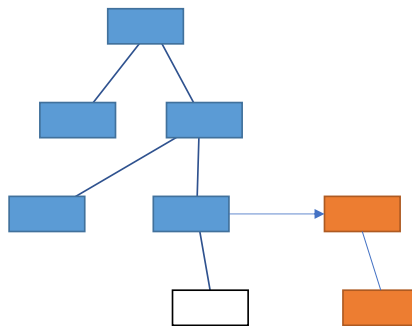
Changman Lee, Dongho Sim, Joo-Young Hwang, and Sangyeun Cho,  
*Samsung Electronics Co., Ltd.*

<https://www.usenix.org/conference/fast15/technical-sessions/presentation/lee>

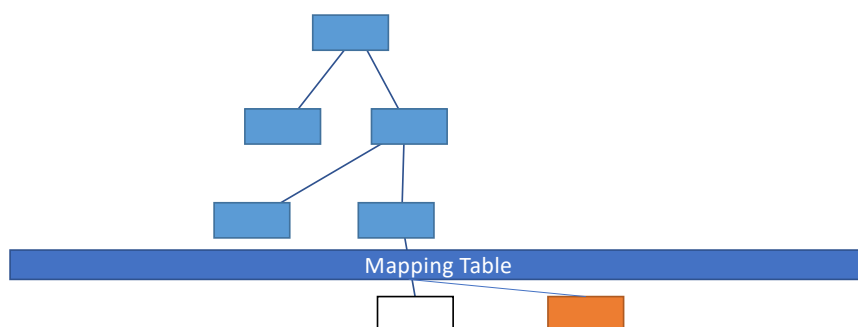
This paper is included in the Proceedings of the  
13th USENIX Conference on  
File and Storage Technologies (FAST '15).  
February 16–19, 2015 • Santa Clara, CA, USA

ISBN 978-1-931971-201

## Fight Walking Trees



## Fight Walking Trees



... noch mehr

- Optimisiertes Layout
- Multi-Head Logging
  - Hot/Cold data
- Adaptive Logging
- Role-Forward Recovery

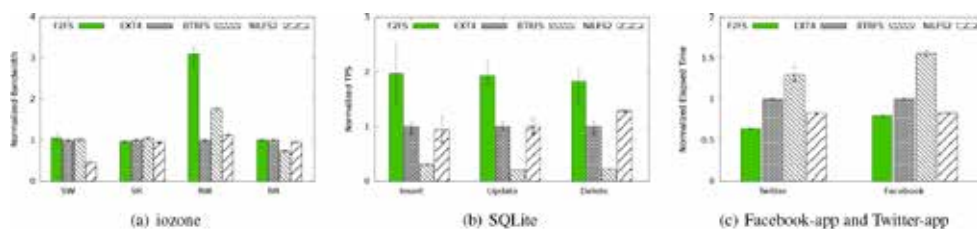


Figure 3: Performance results on the mobile system.

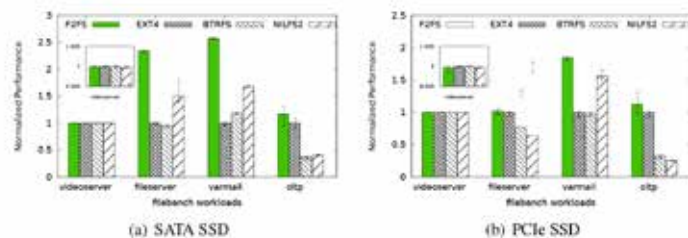



Figure 4: Performance results on the server system.

Diagramme aus dem angegeben Artikel

## Andere Dateisysteme

- BTRFS
  - Wächst und gedeiht
  - Bootbar, Software RAID
- NILFS2
  - New Implementation of a Log-based File System
- XFS
  - In vielen Benchmarks das Schnellste
  - Parallel I/O
- OCFS2
  - Cluster-Dateisystem (für RDBMS-Cluster)



General				
The native Linux kernel port of the ZFS filesystem. To get started with ZFS on Linux simply download the latest release and install using the directions for your distribution.				
Releases				
Version	Checksums	ChangeLog	Release Date	
<a href="#">spl-0.6.5.3 / zfs-0.6.5.3</a>	<a href="#">sha256</a>	<a href="#">v0.6.4 v0.6.5.1 v0.6.5.2 v0.6.5.3</a>	Oct 13 2015	
<a href="#">spl-0.6.4.2 / zfs-0.6.4.2</a>	<a href="#">sha256</a>	<a href="#">v0.6.4 v0.6.4.1 v0.6.4.2</a>	June 26 2015	
<a href="#">spl-0.6.3 / zfs-0.6.3</a>	<a href="#">sha256</a>	<a href="#">v0.6.3 v0.6.3-1.1 v0.6.3-1.2 v0.6.3-1.3</a>	June 12 2014	
<a href="#">spl-0.6.2 / zfs-0.6.2</a>	<a href="#">sha256</a>	<a href="#">v0.6.2</a>	Aug 23 2013	
<a href="#">spl-0.6.1 / zfs-0.6.1</a>	<a href="#">sha256</a>	<a href="#">v0.6.1</a>	Mar 27 2013	
<a href="#">archived versions</a>				
Packages				
Distribution	Distribution	Distribution	Distribution	
<a href="#">Arch Linux</a>	<a href="#">Debian</a>	<a href="#">Fedora</a>	<a href="#">Pentoo</a>	
<a href="#">Generic DEBs</a>	<a href="#">Generic RPMs</a>	<a href="#">Gentoo</a>	<a href="#">RHEL / CentOS / SL</a>	
<a href="#">Sabayon</a>	<a href="#">SprockOS</a>	<a href="#">Ubuntu</a>		
Community Resources				
General	SPL	ZFS	Lustre	
<a href="#">ZFS Mailing Lists</a>	<a href="#">SPL Downloads</a>	<a href="#">ZFS Downloads</a>	<a href="#">Lustre Downloads</a>	
<a href="#">ZFS Documentation</a>	<a href="#">SPL Issue Tracker</a>	<a href="#">ZFS Issue Tracker</a>	<a href="#">Lustre Issue Tracker</a>	
<a href="#">ZFS on Linux FAQ</a>	<a href="#">SPL Source Repository</a>	<a href="#">ZFS Source Repository</a>	<a href="#">Lustre Source Repository</a>	
<a href="#">ZFS on Linux Roadmap</a>				
<a href="#">ZFS on Linux Public Key</a>				
<a href="#">OpenZFS</a>				



## ZFS

- SUN Microsystems
  - Vorreiter im Bereich logbasierter Dateisysteme
- Entwickelt für Solaris 10 / OpenSolaris
- ZFS setzt den "State of the Art" geschickt um
- Vollständige Neuentwicklung
- 128 Bit Dateisystem



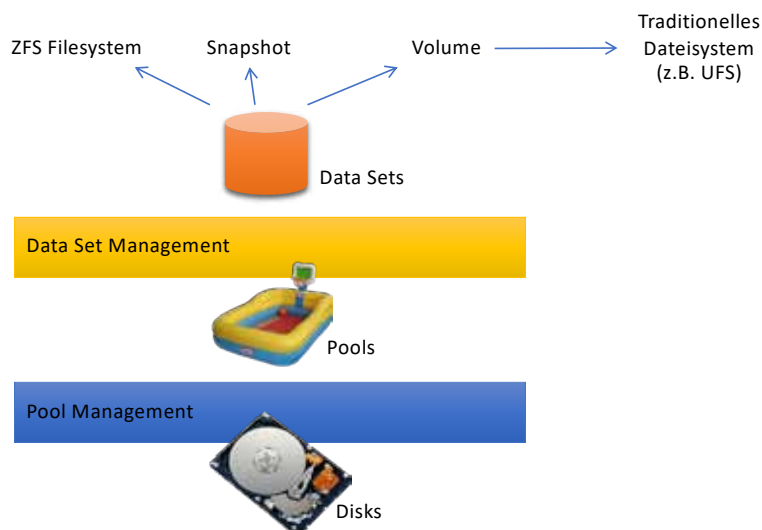


## Eckdaten

- Kernziele
  - Transaktional
  - Integriertes Volume-Management
  - Einfache Bedienung und Administration
- Pooled Storage
  - Alle Speichersysteme werden in einem Pool verwaltet
  - Selbst Hardware-RAID ist prinzipiell nicht mehr nötig
- Ende-zu-Ende Datenintegrität
  - Fehlerisolation zwischen Daten und Prüfsumme
  - Erkennt exotische Fehlersituationen (Phantom Writes, ...)
  - Datenredundanz und Selbstheilung
  - RAID-Z



## Grundstruktur



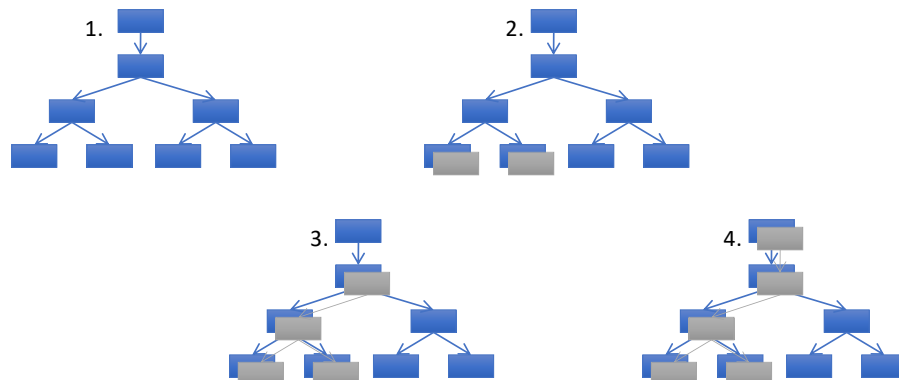
## Charakteristische Größen

- Wortlänge: 128 Bit
- Höchstanzahl Dateien:  $2^{48}$
- Größe Dateisystem (max.): 16 EB
- Größe Datei (max.): 16 EB
- Größe Einzelpool (max.):  $3 \times 10^{23}$  PB
- Höchstanzahl Attribute / Datei:  $2^{56}$
- Höchstanzahl Geräte in zPool:  $2^{64}$
- Dateisysteme in zPool:  $2^{64}$
- zPools in System:  $2^{64}$



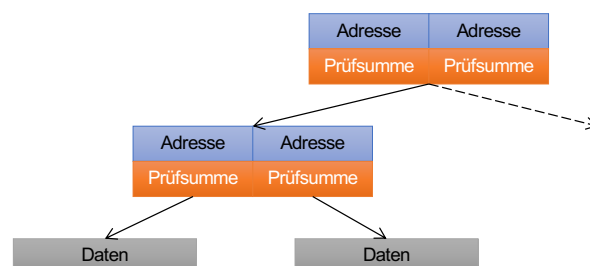
## Datenintegrität

- Alle Operationen copy-on-write / transaktionsbasiert
  - Daten auf Platte immer konsistent
  - Kein Journaling notwendig



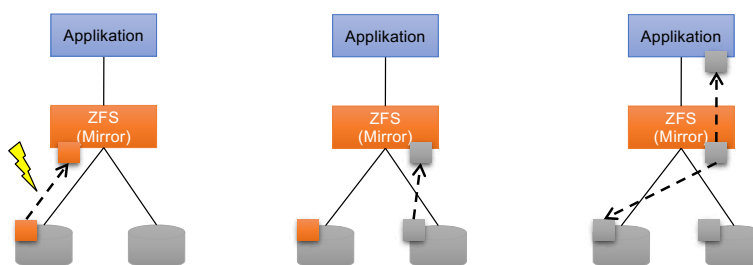
## Datenintegrität II

- Ende-zu-Ende-Prüfsummen
  - Daten und Metadaten
  - 256 Bit (fletcher2 u.a.)
  - Prüfsumme in übergeordnetem Blockpointer



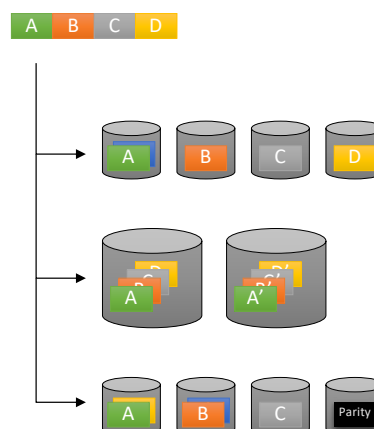
## Datenintegrität III

- Automatische Erkennung und Korrektur korrupter Daten
- 'Selbst-Heilung' der Daten
- Disk Scrubbing



## Redundant Arrays of Independent Disks (RAID)

- 6 Levels
  - RAID 0: Striped
  - RAID 1: Mirrored
  - RAID 2: Striped+ ECC
  - RAID 3: Striped+ Parity
  - RAID 4: Huge stripes + Parity
  - RAID 5: Huge stripes + Rotating Parity
  - RAID 6: RAID 5 + additional parity
- Most common are RAID 0,1,5 and combinations (RAID 10 = 1 + 0)
- Favor huge writes
  - Overhead  $1/(n-1)$  for ndisks
- Small writes are expensive
  - Read data
  - Write data
  - Read parity
  - Write parity



## Raid 5 und Raid 6



- Raid 5
  - Feste Stripe-Größe für schnelle Positionsberechnung
  - Große Writes anstreben
  - Anfällig in der Recoveryphase
- Raid 6
  - Zusätzlicher Parity-Block macht Recoveryphase robust
- Write Hole
  - Stromausfall zwischen Schreiben von Daten- und Parity-Block
  - HW-Controller verwenden NVRAM (z.B. Batteriepufferung)

## Raidz / Raidz2

- In ZFS integrierte Raid-Schemata
- Raidz (= Software Raid5)
  - Atomares Copy-On-Write
  - Stripes unterschiedlicher Länge
    - d.h. nur Full-Stripe-Writes
    - Parity lesen entfällt
- Raidz2 ähnelt in seiner Funktion Raid6

# Schnappschüsse und Klone

## Snapshots

- Read-only Kopie eines Dateisystems oder Volumes
- Platzeffizient (Copy-on-Write)
- Snapshot-Name:

*filesystem@snapname*  
*volume@snapname*

## Clones

- Beschreibbares Volume oder Dateisystem
- Entsteht aus Snapshots (Copy-On-Write)
- Snapshots möglich
- Anwendungsbeispiel
  - Boot Environment von OpenSolaris

## Benutzung

## Administration

- Einfache Administration
- Zwei zentrale Kommandos
  - zpool
    - Erzeugen, Verwalten und Löschen von Pools
    - Zustandsinformation
    - Verlaufslog
  - zfs
    - Erzeugen, Verwalten und Löschen von Datasets
    - Schnappschüsse und Klone

## ZFS Datasets sind “kostengünstig”

- Fein-granulare Nutzung empfohlen
  - Jedes Benutzer-Account = eigenes Dateisystem
  - Jeder besonders zu behandelnde Teilbaum
- Vorteile
  - Weiterhin einzeln verwalt- und konfigurierbar
  - Zuteilung vorhandener Pool-Ressourcen



## Beispiel

```
# zpool create home mirror disk1 disk2

# zfs create home/volker /export/home/volker
# zfs create home/michael /export/home/michael
# zfs create home/annett /export/home/annett

# zpool add home mirror disk3 disk4

# zfs compression=on home/michael
# zfs create home/michael@donnerstag

# zfs quota=42g home/volker

# zfs reservation=10g home/annett

# zfs clone home/michael@donnerstag home/tmp

# zfs send home/michael@donnerstag | ssh sturm@...
```

## Metadaten Repositories

Daten

Daten

Daten

Daten

...

Daten

Metadaten


Metadaten

Metadaten

Metadaten

Finden

- Datenbank speichert
- ... alle vorhandenen Metadaten
- ... zusätzlich inhaltsbezogene Keywords
  - Ermitteln spezifische Plugins abhängig vom Dateityp
  - Ereignisgesteuerter, asynchroner Aufruf bei Dateiänderung
- Komplexe Suchanfragen formulierbar



Repository

## Beispiel Spotlight

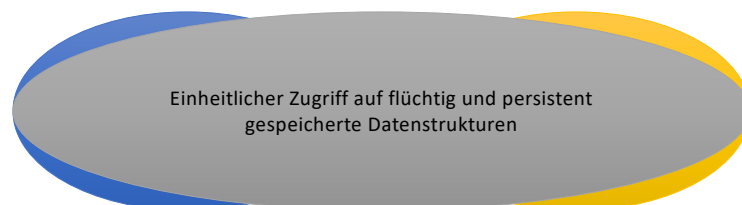


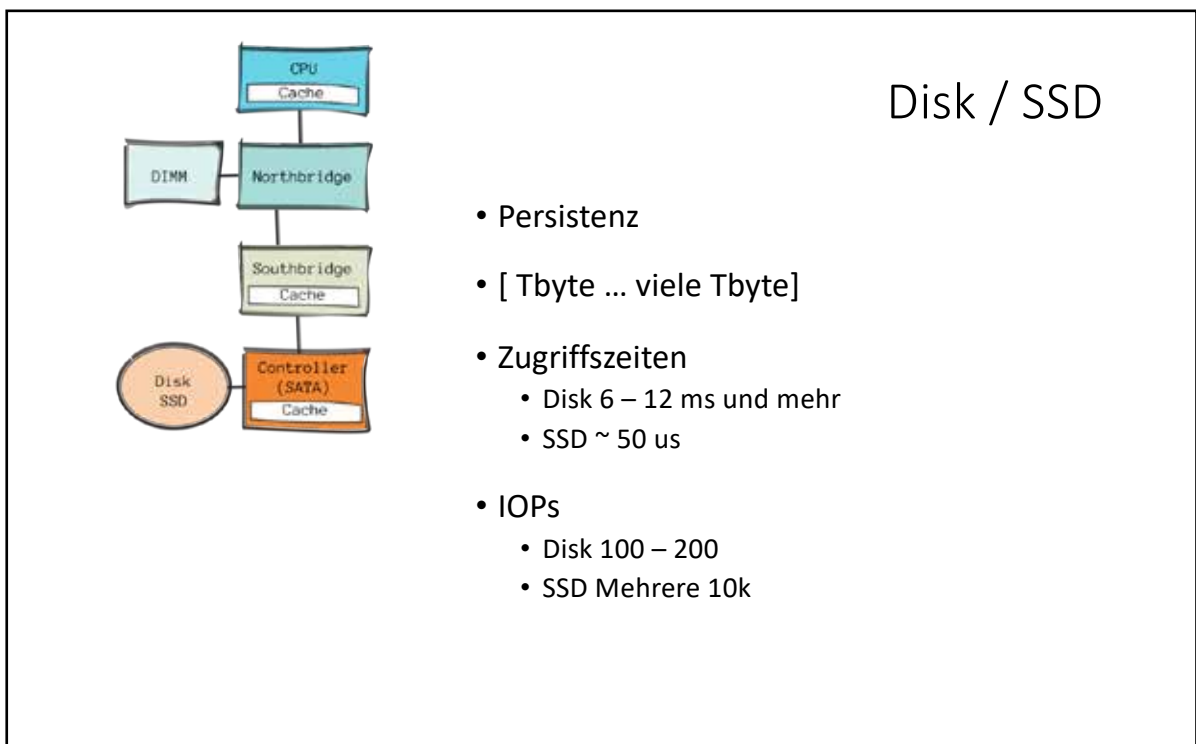
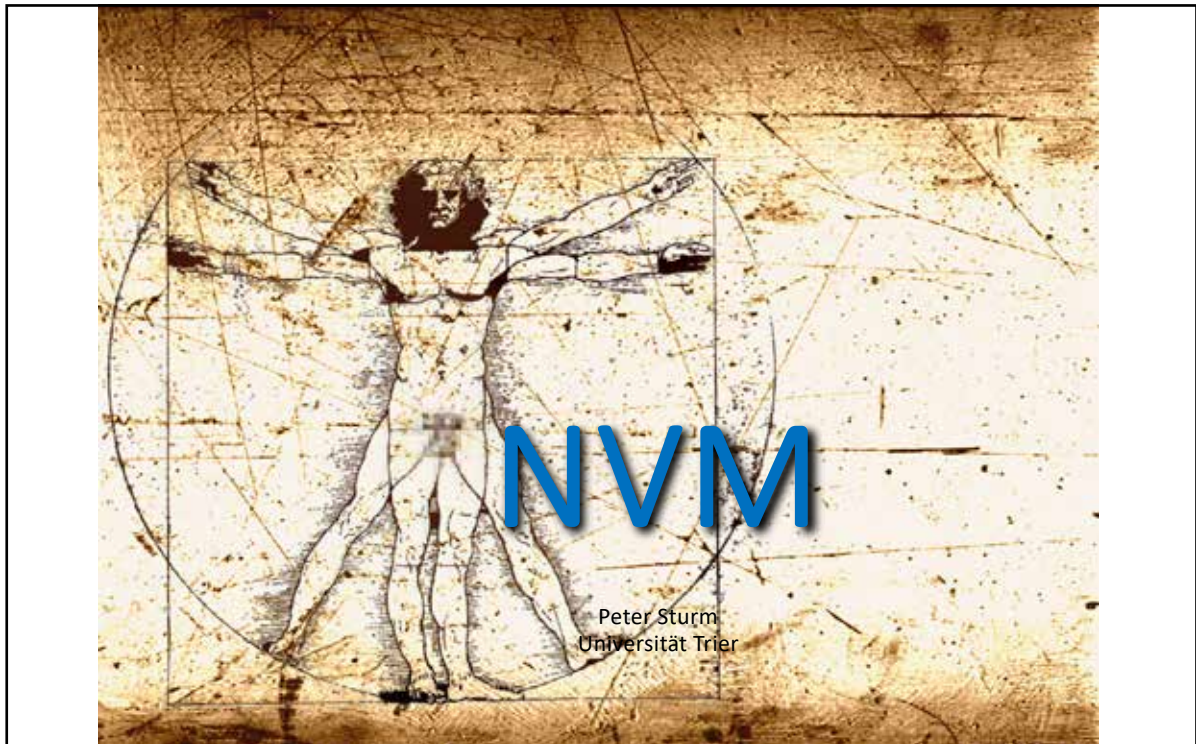


# (P)LINQ

## LINQ

- Language integrated Query
  - Bestandteil von .NET 3.5
- SQL-ähnliche Queries auf Container-Klassen und alle mengenorientieren Datenstrukturen
- Zusätzliche Indirektionsstufe





**V**olume

**V**elocity

**V**erzögerung

**V**olatility

**V**ortune (Vermögen)

## Der “Große Graben”

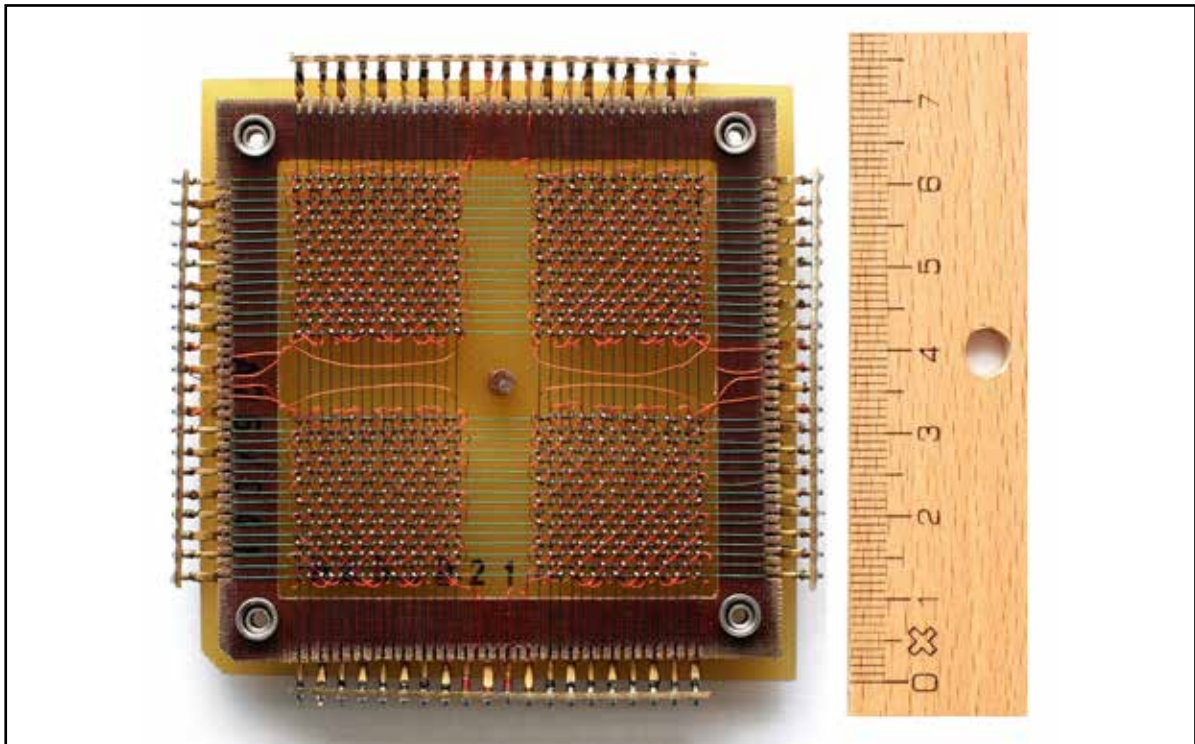


Wir fordern!!!

Volume	<input type="text" value="Wenig"/>	<input type="text" value="Viel"/>
Velocity	<input type="text" value="Lahm"/>	<input type="text" value="Schnell"/>
Verzögerung	<input type="text" value="Viel"/>	<input type="text" value="Wenig"/>
Volatility	<input type="text" value="Ja"/>	<input type="text" value="Nein"/>
Vortune	<input type="text" value="Teuer"/>	<input type="text" value="Günstig"/>







128 Gbyte?

- $1.099.511.627.776 \text{ Bits} / 256 / 4 \text{ cm}^2$
- $1.073.741.824 \text{ cm}^2 = 107.374 \text{ m}^2$
- $328\text{m} \times 328\text{m}$

## RAM / Cache

Volume	Wenig	Viel
Velocity	Lahm	Schnell
Verzögerung	Viel	Wenig
Volatility	Ja	Nein
Vortune	Teuer	Günstig

## Disk (Magnetisch)

Volume	Wenig	Viel
Velocity	Lahm	Schnell
Verzögerung	Viel	Wenig
Volatility	Ja	Nein
Vortune	Teuer	Günstig

## SSD

Volume	Wenig	Viel	
Velocity	Lahm		Schnell
Verzögerung	Viel	Wenig	
Volatility	Ja		Nein
Vortune	Teuer		Günstig

## Storage Class Memory (SCM)

Volume	Wenig	Viel	
Velocity	Lahm		Schnell
Verzögerung	Viel	Wenig	
Volatility	Ja		Nein
Vortune		Teuer	Günstig

## Fast angekommen!?

Volume	Wenig		Viel	Viel
Velocity	Lahm		Schnell	
Verzögerung	Wenig		Wenig	Viel
Volatility	Ja		Nein	
Vortune	Teuer	Teuer	Günstig	

## Indizien



Hard



Auf Lager							
	HGST HUH7210DALLR600 10 TB, FastStart, Ultrastar Archive He10, SATA 600, 24/7	10 TB	8.5 ms/256 MB/7.200 U/min	€ 0,27	REVIEW	5	€ 1.999,-
nicht mehr lieferbar							
	HGST HUH7210DALLR600 10 TB, FastStart, Ultrastar He10, SATA 600, 24/7	10 TB	8.5 ms/256 MB/7.200 U/min	€ 0,00	REVIEW	5	€ 789,-

## Samsung 16TB SSD is the World's Largest Hard Drive

APR 13, 2019 MICHAEL ZHANG

4718 Shares

2.5K SHARE TWEET 781 SHARE

42 COMMENTS



At the 2015 Flash Memory Summit in California this past week, Samsung unveiled the largest capacity hard drive in the world. It's a 2.5-inch solid state drive (SSD) that can hold 16 terabytes of data (15.36TB, to be exact).



## TSV, HBM und HMC

- Through Silicon Via (TSV)
- High Bandwidth Memory (HBM)
- Hybrid Memory Cube (HMC)
- 4, 8 ..., 64? Layers
- 20 – 40 Gbyte/s Durchsatz

## HBM Specs

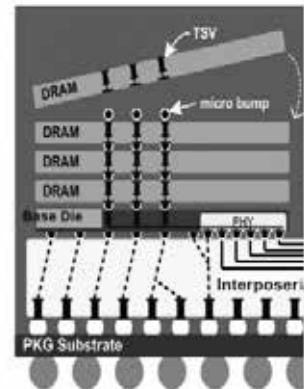
### > HBM1

- 2Gb Density per DRAM die
- 1Gbps speed /pin
- 128GB/s Bandwidth
- 4 HI Stack (1GB)

- x1024 IO
- 1.2V VDD
- KGSD w/  $\mu$ Bump

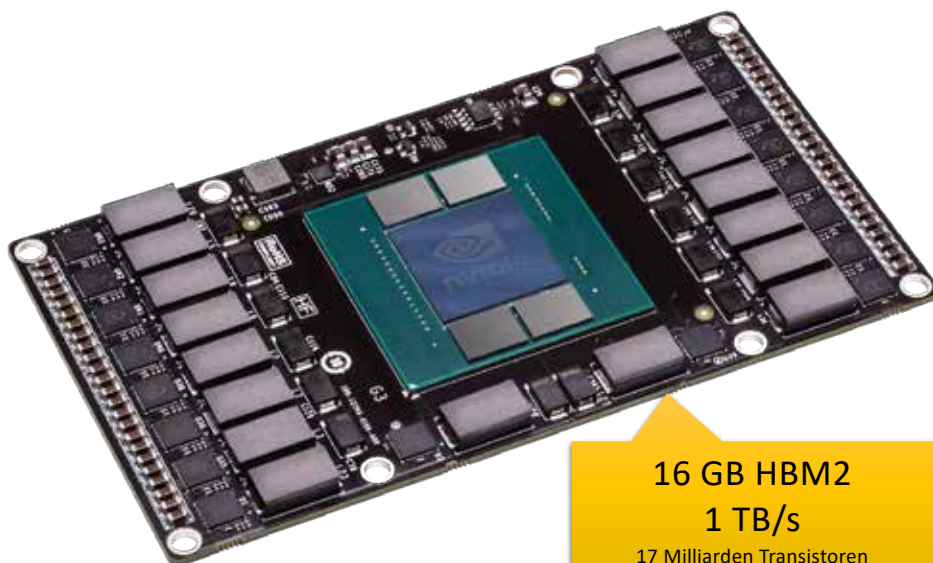
### > HBM2

- 8Gb per DRAM die
- 2Gbps speed/pin
- 256GBps Bandwidth/Stack
- 4/8 HI Stack (4GB/8GB)



SK hynix

## Nvidia Pascal GPU



16 GB HBM2  
1 TB/s  
17 Milliarden Transistoren

## Details

**HBM2 Specification Comparison**

WCCFTech	DDR3	GDDR5	4-Hi HBM1	4-Hi HBM2
I/O	16	32	1024	1024
Prefetch (per I/O)	8	8	2	2
Max. Bandwidth	4.3GB/s (2133 per pin)	32GB/s (8Gbps per pin)	128GB/s (1Gbps per pin)	256GB/s (2Gbps per pin)
tRC	4x - 5xns	40ns(= 1.5V) 48ns(= 1.35V)	48ns	45ns
tCCD	4ns (=4tCK)	2ns (=4tCK)	2ns (=1tCK)	2ns (=1tCK)
VPP	Internal VPP	Internal VPP	External VPP	External VPP
VDD	1.5V, 1.35V	1.5V, 1.35V	1.2V	1.2V
Command Input	Single Command	Single Command	Dual Command	Dual Command

## 128 GB DDR4 DIMM



- Samsung, Ende 2015
  - TSV Interconnects, 144 DRAM Chips
  - 2.4 GT/s \* 8 Byte = 19.2 Gbyte/s/DIMM

30.1.2018

**Samsung DIMM 128GB, DDR4-2666, CL22-19-19, reg ECC (M393AAK40B42-CWD) ab € 2522,12**

Übersicht & Preise | Preisverteilung | Bewertungen | Info beim Hersteller

Typ: DOR4 DIMM 288-Pin, reg ECC • Rammer/Bänke: octa rank, x4 • Module: 1x 128GB • JEDEC: PC4-21300R • Spannung: 1,20V • Besonderheiten: N/A • Herstellerparameter: bitte vorführenden Link beachten

Letztes Preisupdate: 30.01.2018, 09:45  
Gelistet seit: 05.09.2017, 15:09

Es liegen noch keine Bewertungen für dieses Produkt vor (Produkt bewerten).

Produkt empfehlen  
Privatguten: aktivieren  
Fehler melden

**Bezugsart**  
☒ alle Angebote  
 nur Abholung in der Nähe von:   
☐ inklusive Versand  
 per günstigste Variante: nach:

**Angebote aus**  
☒ Österreich  
☒ Deutschland  
☐ Polen  
☐ UK  
☐ allen Ländern

**Verfügbarkeit**  
☒ belüftige Verfügbarkeit  
☐ lagernd beim Händler  
☐ kurzfristig lieferbar (bis 4 Werktage)



(Die abgebildete Illustration ist eine schematische Darstellung)

weiterführende Links:  
[Datenblatt \(Stand: 05.09.2017\)](#) **PDF** [3,14 MiB]  
[samsung.com: Garantiebestimmungen](#)  
[samsung.com: Warranty-Information](#)

Preis*	Anbieter	Händler-Bewertung	Verfügbarkeit Versand**	Artikelbezeichnung des Händlers
<b>C 2522,12</b> zum Angebot		Note: 2,63 243 Bewertungen	Die Lieferung erfolgt innerhalb von 4 Tagen nach Zahlungseingang. Verkäufe, PayPal kostenlos. Abholung nach Vereinbarung	Samsung M393AAK40B42-CWD Samsung • DDR4 • 128 GB • DIMM 288-PIN • 2666 MHz PC4-21300 • CL22 • 1.2 V • registriert • ECC <b>Preis vom: 30.03.2018, 09:45:12</b> (Preis kann jetzt höher sein!)
	Info: 602			

Viel Speicher 😊

IBM Power System  
E880

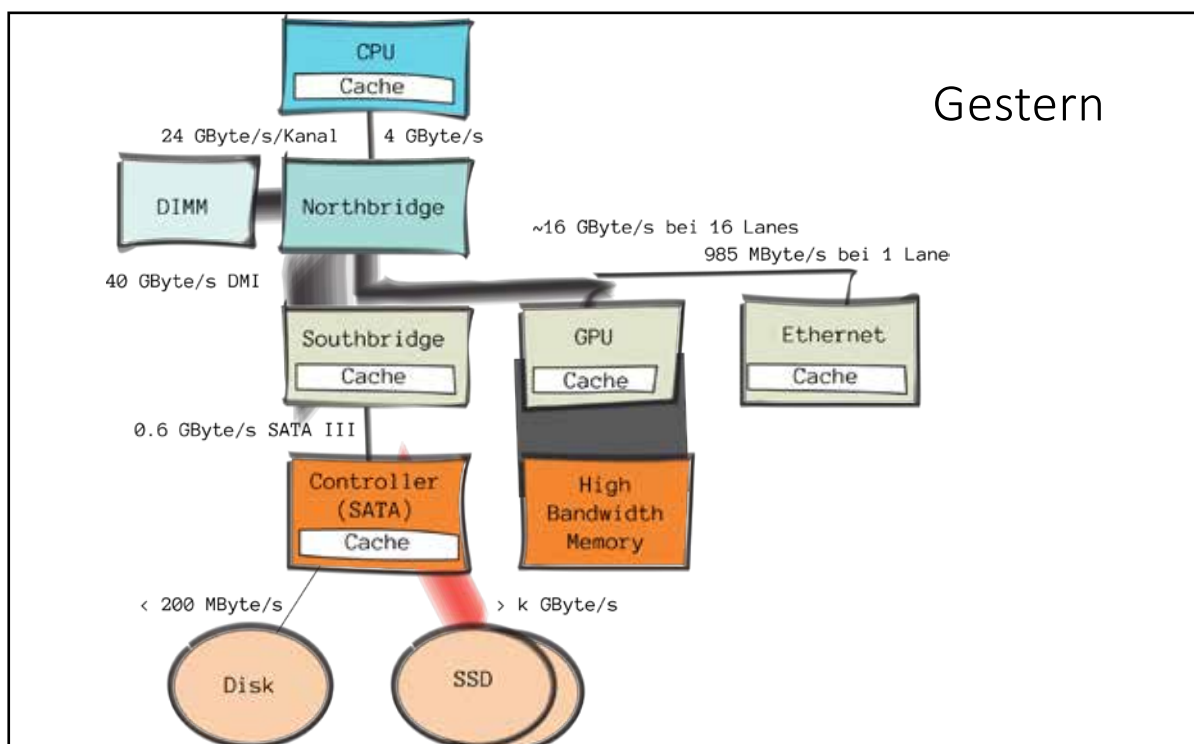
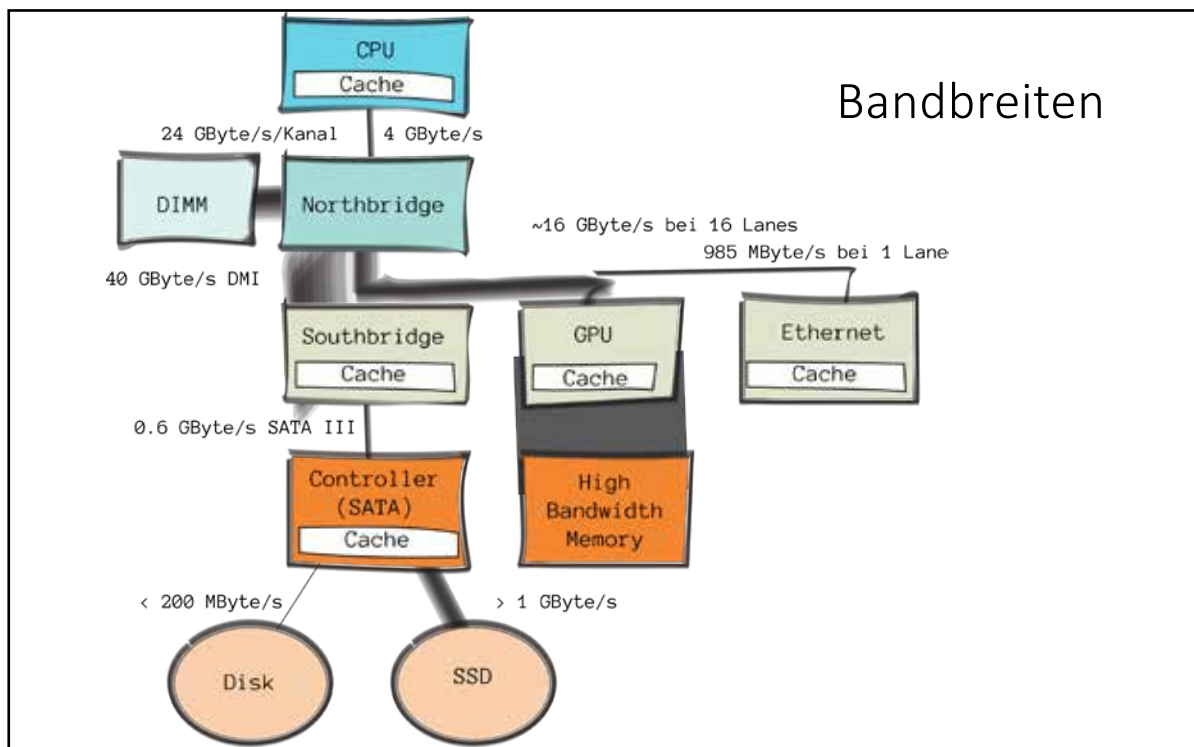
- Product details
- Browse and buy
- Support

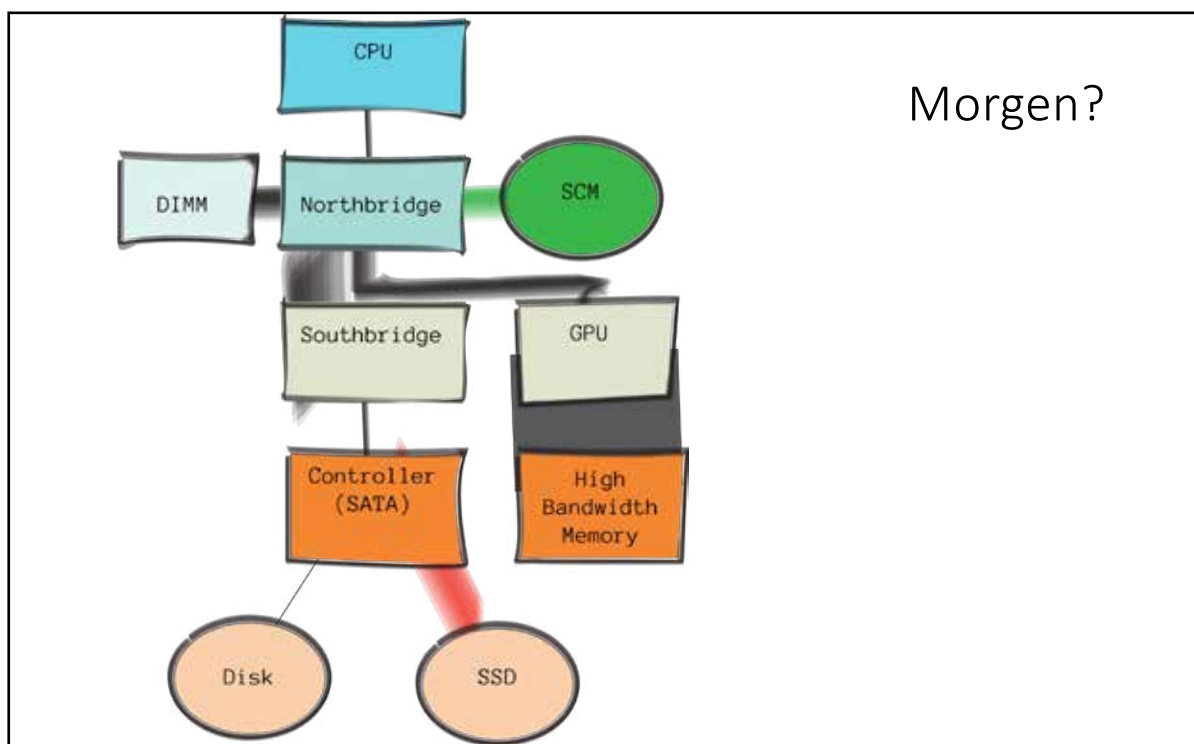
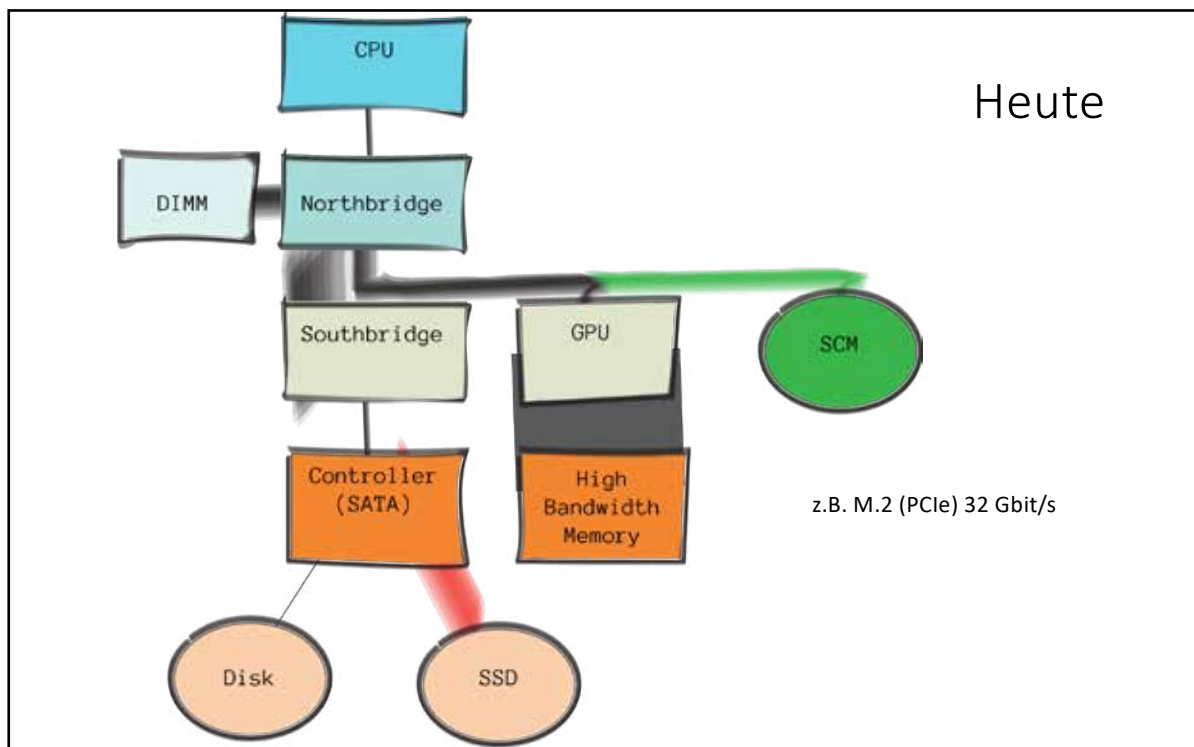
Call for price\*

Processors	Four, eight, twelve or sixteen POWER8 processors
Cores	8 to 128 (8-core processors) 8 to 160 (10-core processors) 8 to 192 (12-core processors)
Clock rates (Min/Max)	Either 4.02 GHz, 4.19 GHz or 4.35 GHz
System memory (Min/Max)	Up to 32 TB
System control unit	One
Integrated PCIe adapter slots	8, 16, 24 or 32 PCIe Gen3 x16
Max PCIe Gen3 I/O Drawers (12 PCIe Gen3 slots each)	Up to 16
Performance (rPerf range)***	716 /3,905.8

128 DIMMs

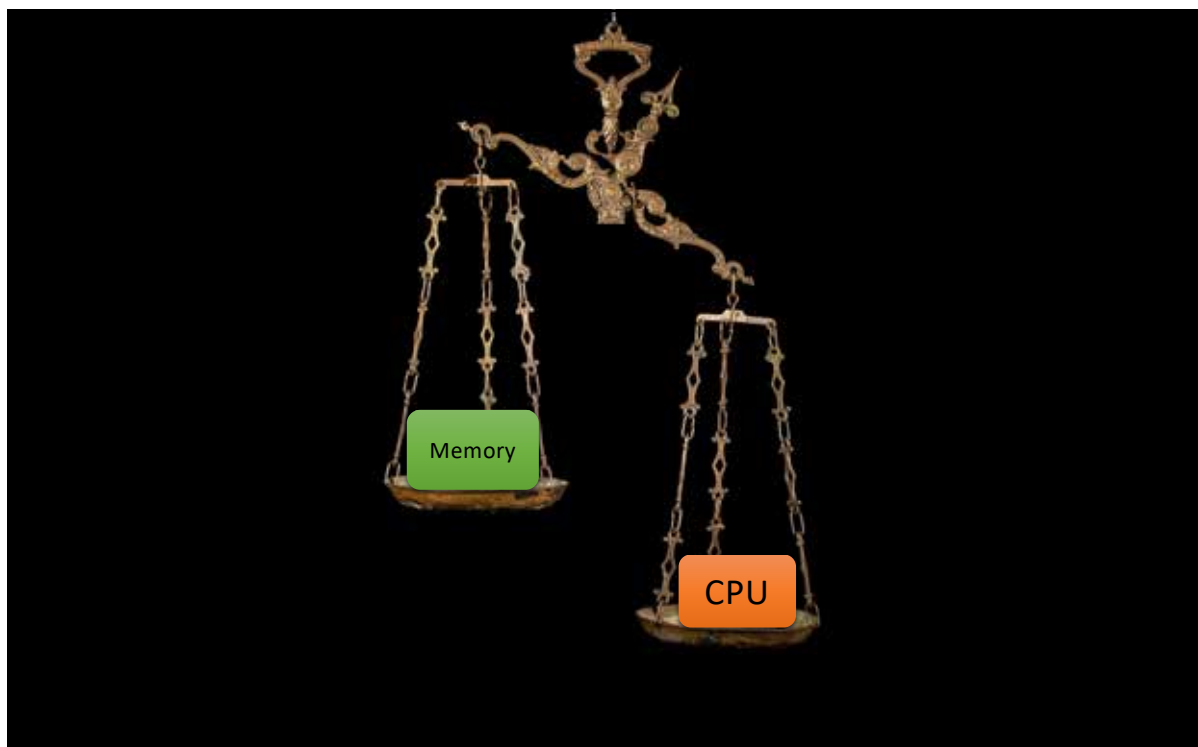




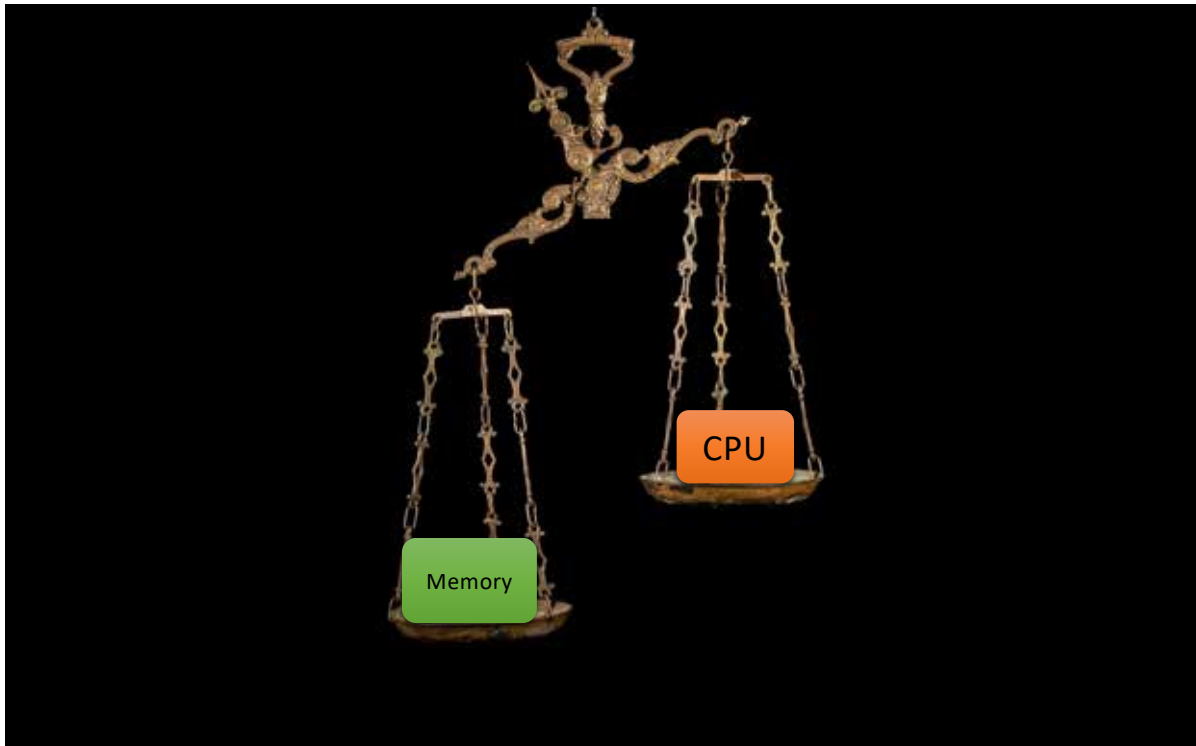


## Andere Stimmen

- Geoffrey W. Burr, IBM
  - “es fehlt nur noch eine Zehnerpotenz”
- Nanavati:







## Strukturfolgen

- Komplexität reduzieren
- Weniger Software-Caches
- Geänderte Prioritäten
  - CPUs müssen nicht mehr warten
  - Weniger Asynchronität
  - Mehr Polling ~ ManyCores

## Neue Paradigmen

- Neue Datenbankprinzipien
  - Jede DB ist In-Memory 😊
- "Kriegen wir später"
  - Deklarativ
  - Knowledge-based
  - Machine Learning