# Systemsoftware

## Dateisysteme unter Linux

Prof. Dr. Michael Mächtel

Informatik, HTWG Konstanz

Version vom 26.03.17

# Übersicht

# Übersicht

# Übersicht

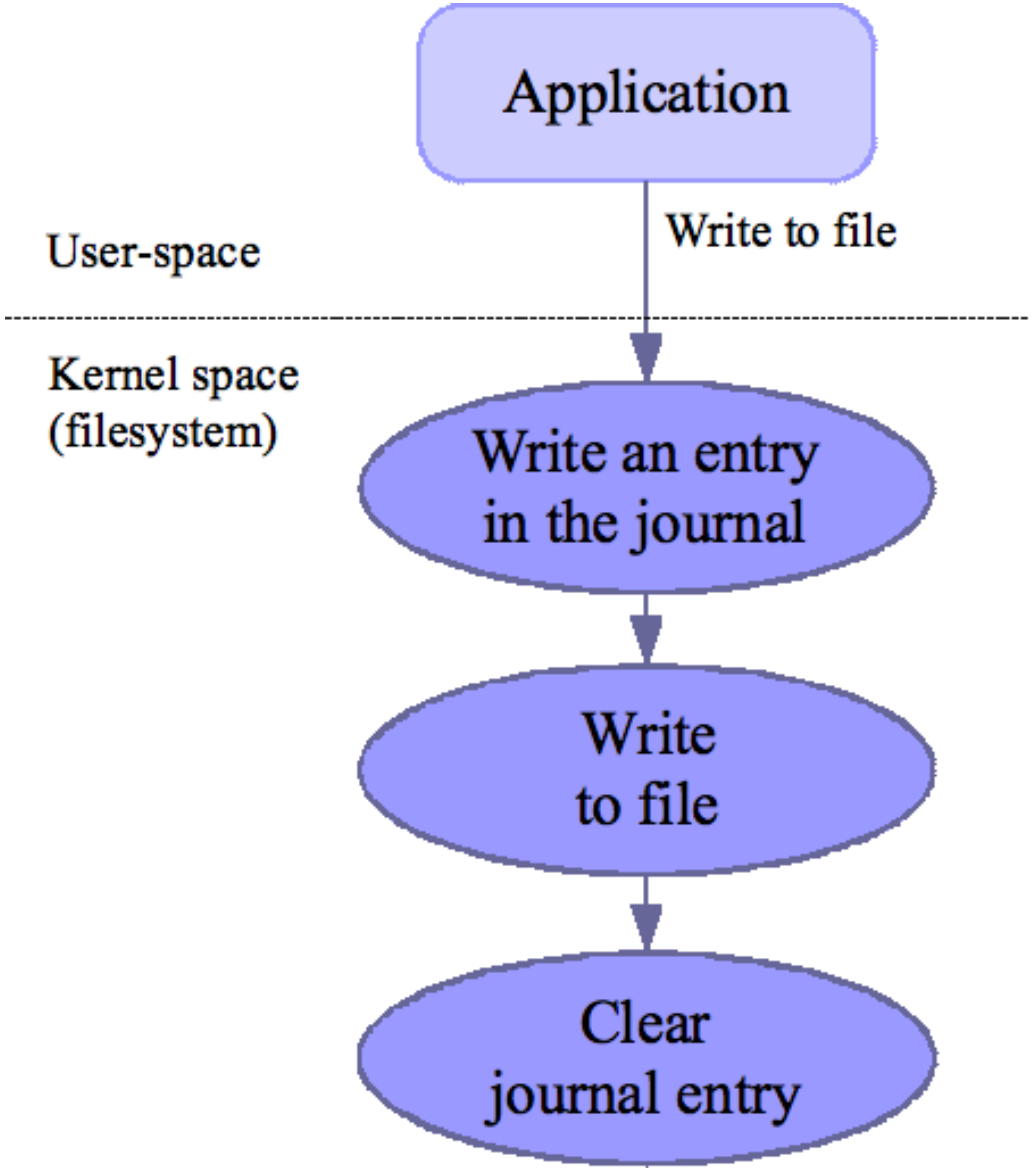
- Block Geräte
  - Floppy oder Festplatte (SCSI, IDE)
  - Compact Flash (als IDE Gerät ansprechbar)
  - RAM Disk
  - Loopback Geräte
- Memory Technology Devices (MTD)
  - Flash, ROM or RAM chips
  - MTD emulation on block devices

# Probleme

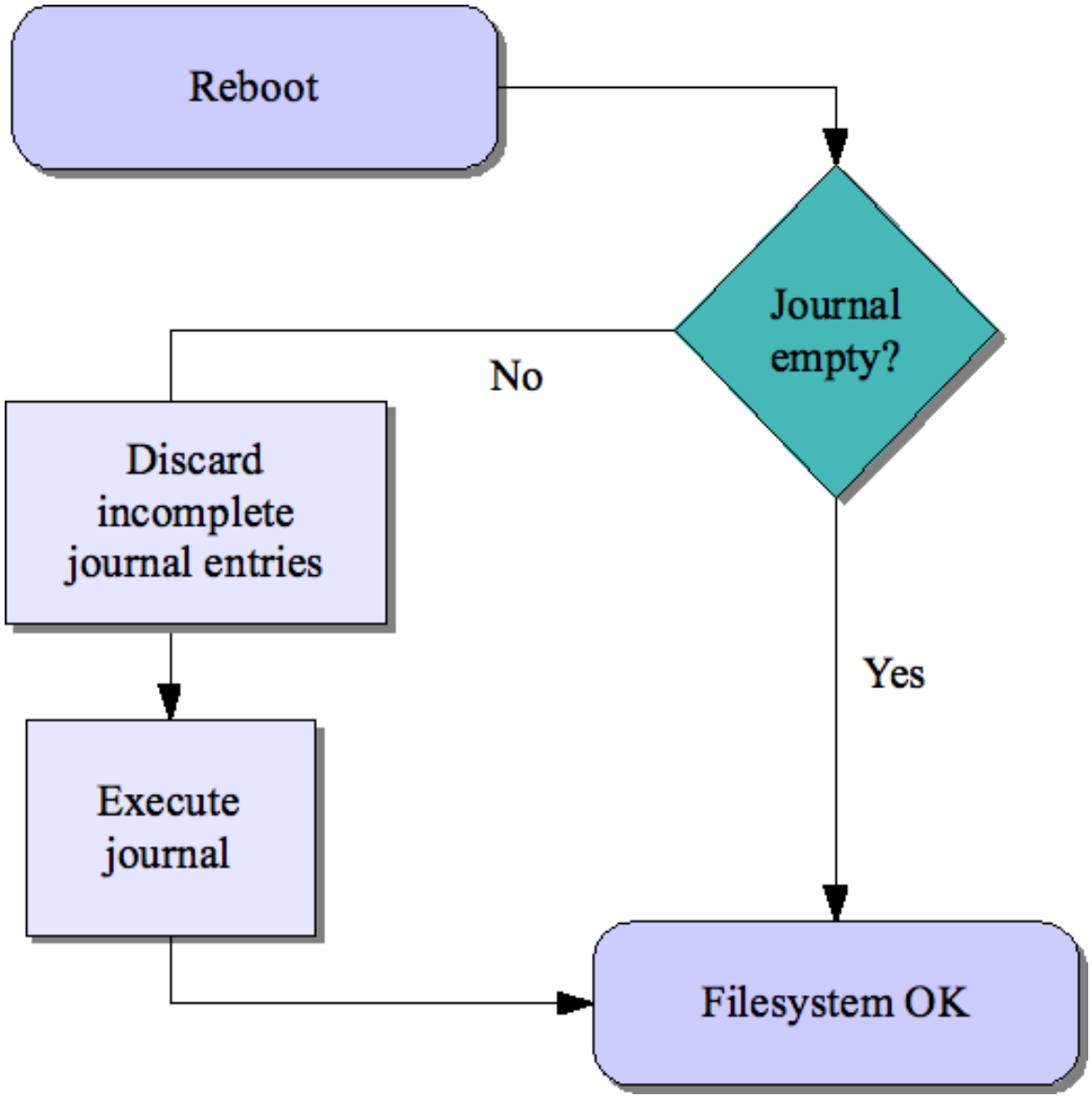
- Wiederherstellung nach Crash schwierig: Filesystem verbleibt im Zustand 'half finisched' nach dem Reboot
- ext2: Traditionelles Linux Dateisystem (Wiederherstellung: **fsck.ext2**)
- vfat: Traditionelles Windows Dateisystem (Wiederherstelltung: **fsck.vfat** (unter Linux) oder **scanndisk** (unter Windows)

# Übersicht

# Journaling Prinzip



# Journaling Boot



# Vorteil Journaling

- Dateisystem bleibt in einem korrekten Zustand, auch nach einem Systemcrash
- Alle Schreibvorgänge werden zuerst im Journal gesichert, bevor die Daten in die Dateien 'commited' werden.
- Durch das Journal bleibt das Dateisystem immer in einem korrekten Zustand.
- Daten können jedoch verloren gehen, wenn sie noch nicht 'commited' sind.
- Beispiele:
  - ext4,ext3: ext2 mit Journal Erweiterung
  - raiserFS/raiser4, JFS (IBM), XFS (SGI) ,...

# Übersicht

# cramfs

- einfaches und kleines read-only Dateisystem mit integrierter Datenkompression.
- Maximale Dateisystemgröße: 256 MB
- Minimale Dateisystemgröße: 16 MB
- kein Entpacken nötig, direkter Zugriff!
- Dateien im CramFS sind mit der zlib komprimiert.
- Metainformationen der Dateien sind unkomprimiert.

# squashfs

- read-only Dateisystem mit integrierter Datenkompression.
- gegenüber cramfs: bessere Kompression und bessere Performance
- Benchmark: 3* kleiner als ext3, 2–4 mal schneller als ext3
- Maximale Dateisystemgröße: $2^{64}$ bytes!
- SquashFS kann mit Blockgrößen bis zu 64K benutzt werden (Standardwert 4K). Je größer die Blockgröße gewählt wird, desto höher sind die Kompressionsraten.
- erkennt Datei-Duplikate: Dateien, die mehrfach vorhanden sind werden nur einmal gespeichert.

# Übersicht

# Übersicht

- Zwei Technologien
  - NAND
  - NOR
- Softwarebereiche
  - Bootloader
  - Kernel
  - Rootfs
  - Config (z.B. für den Bootloader)
- Ziel beim Einsatz:
  - Zugriffe auf Flash-Speicher reduzieren…

# Merkmale

- Adressierung byte- oder wortweise
- Setzen von Bits (’1’) ist nur in Blöcken möglich (Löschen)
- Rücksetzen von Bits (’0’) ist manchmal byte- oder wortweise möglich (Schreiben)
- Konsequenz: Erst löschen, danach schreiben
- Schreiben ist oft nur über eine Kommandofolge möglich (NAND)

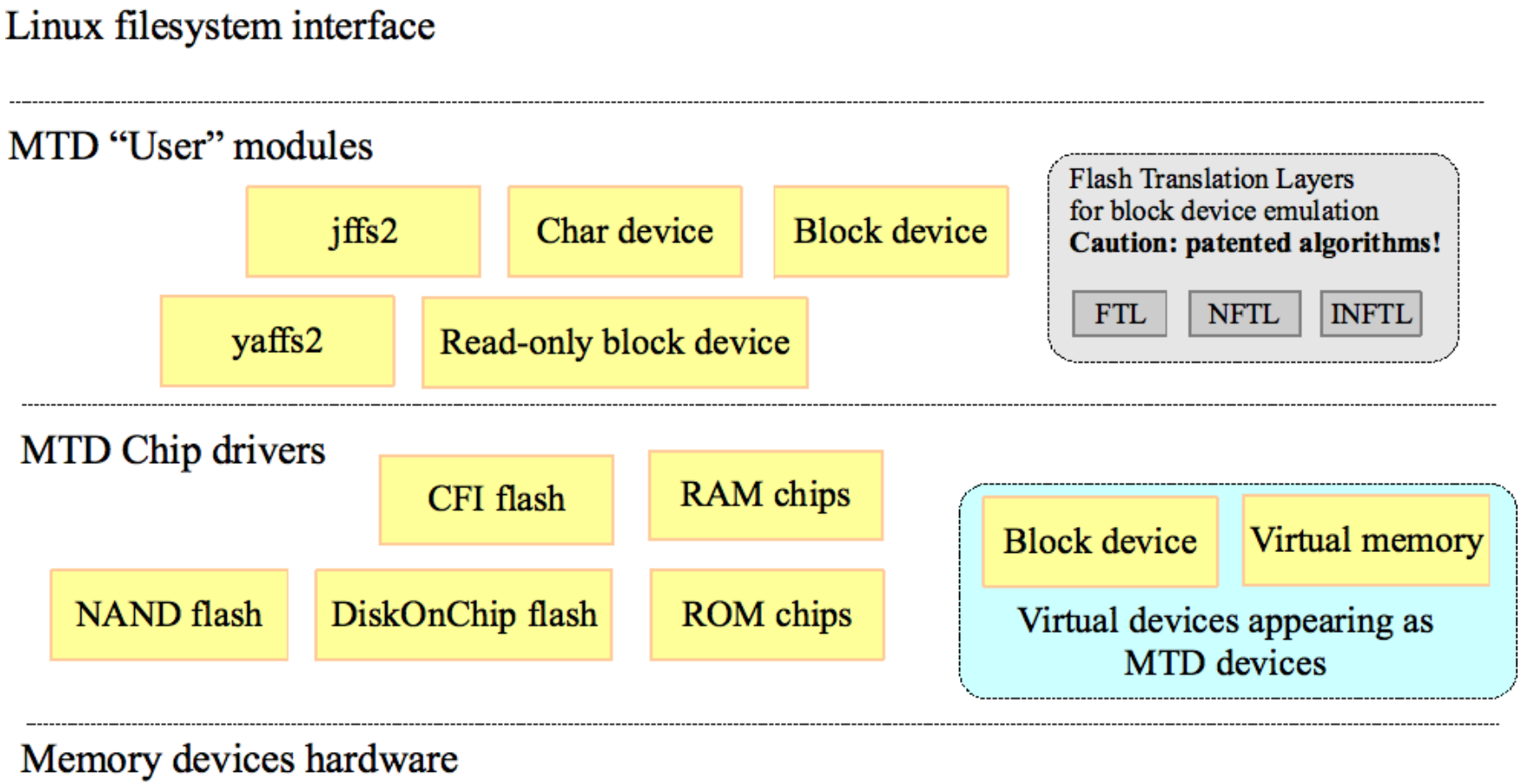
# NAND

- Einsatz: Ersatz von Festplatten
  - USB-Stick, SD-Card
  - MP3-Player
  - …
- bis zu 1 Millionen Löschzyklen möglich
- kompakter: weniger Chipfläche (40% zu NOR)
- langsamer Zugriff
- Lesen und Schreiben ist nur blockweise möglich. 1 Block entspricht 16 kByte (32 Pages a 512 Byte)
- Bad Block Mangement notwendig

# NOR

- bis zu 100000 Schreibzyklen
- schneller Lesezugriff
- kleine Datenmengen lassen sich schnell schreiben
- wahlfreier Zugriff
- relativ hohe Leistungsaufnahme
- kleine Speicherkapazität (mehrere Megabyte)
- Adressierung byte- oder wortweise

# Memory Technology Devices (MTD)



# jffs2

- jffs2: Journaling Flash File System v2
- Flash Sektoren werden nach 'wear leveling' beschrieben. Schreibzugriffe werden über das gesamte Flash Device verteilt!
- Dateisystem mit Kompression
- Mit Journal, dadurch reboot ohne Interaktion in geordnetes Dateisystem möglich
- Block-Geräte Schnittstelle erlaubt die Erstellung eines jffs2 Dateisystems auf dem Hostsystem
- Aber: Langsam und hoher RAM Bedarf ( 4MB RAM für 128 MB Flashspeicher)

# yaffs2

- yaffs2: Yet Another Flash File System v2
- Nur für NAND Flash.
- Keine Kompression!
- Sehr schnell (gegenüber z.B. jffs2) insbesondere während des Bootvorgangs (Initialisierung)
- benötigt sehr wenig RAM
- Flash Sektoren werden nach 'wear leveling' beschrieben. Schreibzugriffe werden über das gesamte Flash Device verteilt!
- Portiert auf: GNU/Linux, WinCE, eCOS, pSOS, VxWorks.

# Weitere Entwicklungen

- Dateisystem, speziell für FLASH Devices werden ständig weiterentwickelt:
  - LogFS is a Linux log-structured and scalable flash file system
    - intended for use on large devices of flash memory
    - LogFS was motivated by the difficulties of JFFS2 with larger flash-memory drives
  - The Unsorted Block Image File System (UBIFS) is a
    - successor to JFFS2, and
    - competitor to LogFS, as a file system for use with raw flash memory media.
  - ...

# Übersicht

# *tmpfs*

- Bisher bekannte RAM-FS:
  - initrd
  - initramfs
- Neu: *tmpfs*
  - Dynamische Größe: Filesystem wird vergrössert und verkleinert, entsprechend der Konfigurationsdaten.
  - Dateien nur 1* im RAM (da ohne File-Cache zugegriffen wird)
  - Integriert in der Speicherverwaltung, Pages können ausgelagert werden, falls benötigt.

# *unionfs*

- *unionfs*
  - Overlay Dateisystem
  - Spinoff davon: *aufs*

# Übersicht

# Qual der Wahl