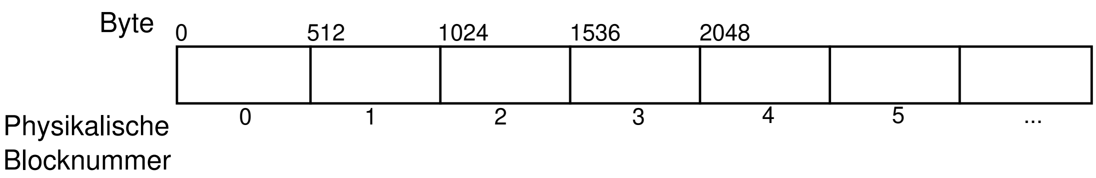
1 Sysprog - Filesystem (*spfs*)

- Vorbild: SystemV Filesystem *s5fs* (original Unix Filesystem von ca.1970)
- Aufbau einer Partition der Festplatte:



- Device-Treiber rechnet Blocknummer in Zylinder/Spur/Sektor um
- Formatierung:

2 Inodes (s5fs)

- enthalten Informationen über eine Datei, 64 Byte groß (= 8 Inodes pro Block)

Feld	Größe/Byte	Erklärung
`di_mode`	2	Dateityp (0xf000), Zugriffsrechte (0x0fff)
`di_nlinks`	2	Anzahl der Hardlinks auf die Datei
`di_uid`	2	UID des Besitzers
`di_gid`	2	GID des Besitzers
`di_size`	4	Größe der Datei
`di_addr`	3*13	Adressen der Datenblöcke
`di_gen`	1	Generationsnummer (wird bei Wiederverwendung inkr.)
`di_atime`	4	Zeit des letzten Zugriffs
`di_mtime`	4	Zeit der letzten Veränderung
`di_ctime`	4	Zeit der letzten Änderung der Inode-Daten

3 Inodes (spfs)

- enthalten Informationen über eine Datei, 128 Byte groß (= 4 Inodes pro Block)

Feld	Größe/Byte	Erklärung
di_mode	2	Dateityp (0xf000), Zugriffsrechte (0x0fff)
di_nlinks	2	Anzahl der Hardlinks auf die Datei
di_size	4	Größe der Datei
di_addr	4*27	Adressen der Datenblöcke
di_atime	4	Zeit des letzten Zugriffs
di_mtime	4	Zeit der letzten Veränderung
di_ctime	4	Zeit der letzten Änderung der Inode-Daten

4 Die Blockliste einer Datei (di_addr)

- s5fs
  - Liste aus 13 physikalischen Blocknummern zu je 24 Bit (= 3 Byte)
  - damit sind 2^24 Blöcke = 8192 MByte = 8 GByte adressierbar
  - die 10. Blocknummer wird verwendet für einfach indirekte Adressierung, die 11. für 2-fach indirekt, die 12. für 3-fach indirekt (beginnend bei 0)
- spfs
  - Liste aus 27 physikalischen Blocknummern zu je 32 Bit (= 4 Byte)
  - damit sind 2^32 Blöcke = 2097152 MByte = 2048 GByte = 2 TByte adressierbar
  - in *spfs* nur direkte Adressierung, d.h. Maximallänge einer Datei ist 27* 512 Byte = 13,5 kByte

5 Der Superblock

- enthält
  - Größe der Partition (in Blocks)
  - Größe der Inode-Liste (in Blocks)
  - Anzahl der freien Blocks/Inodes
  - Liste der freien Inodes
  - Liste der freien Datenblocks (im spfs nur Blocknummer des ersten Blocks der Liste der freien Datenblöcke)

- Liste der freien Inodes
  - wird unvollständig gehalten
  - kann jederzeit aus der Inode-Liste erzeugt werden, indem nach Inodes mit di_mode=0 gesucht wird

5.1 Liste der freien Datenblöcke

- freier Datenblock kann nicht anhand seines Inhalts identifiziert werden

- Größe der Liste ist durch Verkettung der Blöcke dynamisch an die Anzahl der freien Datenblöcke anpaßbar

5.2 Liste der freien Datenblöcke (2)

- ungültige Einträge = 0
- Sonderfall: alle Free-listen Blöcke sind voll und es ist genau ein freier Datenblock übrig
  - ◆ dieser Block wird als Free-listen Block verwendet
- allocate_datablock
  - suche nach Eintrag!=0 in erstem Block der Freeliste
  - wenn vorhanden, setze auf 0 und gib Eintrag zurück
  - wenn alle Einträge==0, klinke Block aus Liste aus und gib den Free-listen Block als Datenblock zurück
- reclaim_datablock(blocknr)
  - suche nach Eintrag==0 in erstem Block der Freeliste
  - wenn vorhanden, setze Eintrag auf blocknr
  - sonst verwende blocknr als neuen (ersten) Freelisten-Block

6 In-core Inodes

- die Inode-Information offener Dateien wird im Hauptspeicher gehalten

7 Verzeichnisse in spfs

- Verzeichnis ist besondere Datei (**`di_mode & 0xf000 = SP_IFDIR`**)
- enthält Records zu je 16 Byte
  - 1Byte Inode-Nr. der Datei
  - 15 Byte Name der Datei (wenn Name<15 Byte, dann '\0'-terminiert)
- maximal 255 Dateien pro Partition (da Inodeadresse 8 Bit und Inode 0 reserviert)
- die ersten beiden Einträge sind "." und ".."
- im Wurzelverzeichnis (*Root-Directory*) haben "." und ".." die Inode-Nr. 2