

Übungsblatt 6

Abgabe bis spätestens 14.12.2023 in Stud.IP

Allgemeiner Hinweis

Das in StudIP zur Verfügung gestellte Archiv ueb6-vorgabe.zip enthält TikZ-Quellen, die Ihr bei Bedarf als Gerüst für die in Aufgabe 3 geforderten Tabellen nutzen könnt.

Aufgabe 1 (2 Punkte)

Ein Programm setzt (vereinfacht) die folgenden acht Systemaufrufe ab. Geht davon aus, dass keiner von ihnen fehlschlägt. Der Inhalt der Datei ti2-dateivw.pdf sei 2835977 Bytes groß.

```
const char fn[] = "/home/irma/ti2/sld/ti2-dateivw.pdf";
ssize_t len;
char buf[500];
/* ... */
... lstat(fn, &statbuf) ...
int fd = open(fn, O_RDWR);
/* ... */
len = read(fd, buf, sizeof(buf));
/* ... etwas mit buf machen ... */
len = write(fd, buf, sizeof(buf));
lseek(fd, 2835548, SEEK_SET);
len = write(fd, "test", 4);
lseek(fd, 3500, SEEK_CUR);
memset(buf, 0, sizeof(buf));
len = write(fd, buf, sizeof(buf));
```

Es wird mit einem Unix-V7-ähnlichen Dateisystem gearbeitet und es gilt zudem:

- Die logische Blockgröße betrage 4096 Bytes.
- Die Inodes seien 128 Bytes groß.
- Ein Indirektblock enthalte 1024 Verweise
- Nur der Inode für / befinde sich bereits in der Inode-Tabelle.
- Die verwendeten Dateien haben die folgenden Inode-Nummern: home (5), ti2 (124), irma (99), sld (847) und ti2-dateivw.pdf (3537).
- Die Verzeichnis-Dateien seien jeweils nur einen Datenblock lang.
- Der Buffer-Cache sei initial leer und so groß, dass im Rahmen dieser Aufgabe nicht verdrängt werden muss.

Verfolgt die Abarbeitung der Systemaufrufe. Ziel ist es zu erarbeiten, wie Ihr Euch „von oben nach unten“ über die Inodes und Datenblöcke der Verzeichnisse durch dieses Dateisystem bewegen könnt. Welche Inodes und Blöcke müssen eingelesen werden, welche werden geschrieben? Bedenkt die Rolle des Buffer-Caches.

Aufgabe 2 (4 Punkte)

Die Datei `rechnernetze.mkv` ist 558787327 B groß. Sie ist auf einer Festplatte mit den folgenden technischen Parametern gespeichert:

- 5600 Umdrehungen in der Minute,
- vier Oberflächen zur Datenspeicherung,
- es kann immer nur von einer Oberfläche zur Zeit gelesen werden,
- 48000 Spuren pro Oberfläche,
- 8000 Sektoren pro Spur (sei vereinfachend für alle Spuren gleich),
- 512 Bytes pro Sektor,
- 4096 Bytes pro Block im Dateisystem,
- 0.6 ms Kopfschaltung beim Wechsel der Oberfläche (in einem Zylinder) und
- ein beliebiger Spurwechsel dauert (stark vereinfacht) insgesamt 5 ms (dies beinhaltet eine möglicherweise erforderliche Kopfschaltung). Die Lese-/Schreibköpfe sind starr untereinander verbunden und bewegen sich daher nur gemeinsam (→ alle Köpfe befinden sich immer in demselben Zylinder).

Wir gehen für die Berechnungen davon aus, dass keine weitere Abbildung stattfindet, es sich also um die tatsächliche Geometrie der Festplatte handelt.

- a) Wieviel Speicherplatz (netto, ohne Dateisystem usw.) bietet diese Festplatte rechnerisch? Angabe bitte in Bytes und auf ganzzahlige GiB abgerundet.
- b) Wie lange würde es bei bestmöglichen Bedingungen dauern, die genannte Datei von der beschriebenen Festplatte zu lesen und welche durchschnittliche Übertragungsrate würde dabei erreicht, wenn immer nur von einer Oberfläche zur Zeit gelesen werden kann? **Dokumentiert die zugrunde gelegten Annahmen.**
- c) Wie lange würde es bei bestmöglichen Bedingungen dauern, die genannte Datei von der beschriebenen Festplatte zu lesen und welche durchschnittliche Übertragungsrate würde dabei erreicht, wenn der Plattencontroller **von allen vier Oberflächen parallel** lesen kann und die gelesenen Sektoren immer in der richtigen Reihenfolge an das Betriebssystem liefert? **Dokumentiert die zugrunde gelegten Annahmen.**
- d) Wie lange würde es bei der schlechtestmöglichen Verteilung der Datenblöcke auf der Festplatte dauern, bis die genannte Datei von der beschriebenen Festplatte gelesen wurde und welche durchschnittliche Übertragungsrate würde dabei erreicht, wenn immer nur von einer Oberfläche zur Zeit gelesen werden kann? **Dokumentiert die zugrunde gelegten Annahmen.**

Hinweis: Ihr könnt in eurer Rechnung von den Verwaltungsinformationen der Datei bzw. des Dateisystems abstrahieren. Externe Faktoren wie Stromausfall usw. sollen in der Betrachtung natürlich nicht berücksichtigt werden, ebenso sollen mögliche Optimierungen durch den Festplattencontroller oder das Betriebssystem, wie z. B. vorausschauendes Lesen, nicht in die Betrachtung einbezogen werden, soweit nicht anders angegeben (vgl. Aufgabenteil c).

Achtet bei euren Berechnungen darauf, erst bei der Bestimmung des Endergebnisses zu runden. Das Rechnen mit gerundeten Zwischenergebnissen führt zu Punktabzug. Rundet die Datenrate auf **KiB/s bzw. MiB/s**.

Aufgabe 3 (4 Punkte)

Gegeben sei der folgende Schnappschuss aus den kerninternen Dateiverwaltungsstrukturen eines Unix-Systems. Was würden die folgenden Systemaufrufe bewirken und ggf. an den gezeigten Datenstrukturen verändern? (Markiert Eure Änderungen entsprechend der betreffenden Teilaufgabe. Alle Änderungen sollen in derselben Grafik in der angegebenen Reihenfolge durchgeführt werden.)

- a) Prozess A: `lseek(3, 430, SEEK_SET)`
- b) Prozess B: `read(4, buf, 20)`
- c) Prozess B: `dup(5)`
- d) Prozess A: `close(2)`

Deskriptortabelle von Prozess A		Deskriptortabelle von Prozess B		Filetabelle			Inodetabelle (Incore-Inodes)					
				Zähler	Position	Inodetab.	Zähler	Gerät	Inodenr.	Kopie Inode		
0	5	0	4	0	1	435	4	0	1	4711	2	... (Inode von /)
1	3	1	8	1	1	32	4	1	1	4711	6	... (Inode von /usr)
2	6	2		2				2	2	4711	23	... (Inode von /dev1)
3	0	3		3	1	0	2	3	1	4711	17	... (Inode von /dev2)
4		4	1	4	1	0	3	4	2	4711	15	... (Inode von /bla)
5		5	7	5	1	25	5	5	2	4711	13	... (Inode von /puffer)
				6	1	0	2	6	1	4711	3	... (von /usr/myprog)
				7	1	85	8	7				
				8	1	57	5	8	1	4711	11	... (Inode von /err)

Abgabe

Bis 23:59 Uhr am 14.12.2023 digital in StudIP. **Es gelten die vereinbarten Scheinbedingungen (siehe StudIP).** Bitte beachtet unsere ergänzenden Hinweise ebenda.

Ladet die abgaberelevanten Dateien in DoIT hoch.

Eure Ansätze und der gewählte Lösungsweg müssen nachvollziehbar sein. Achtet insofern auf eine saubere Dokumentation im Quelltext und im abgegebenen PDF-Dokument. Benennt alle von Euch verwendeten Quellen, auch Zusammenarbeit mit anderen Gruppen und verwendete Unterlagen aus früheren Jahrgängen.

Programmieraufgaben sind im Zweifel in C++20 zu entwickeln. Die Korrektheit der Lösung bzw. deren Grenzen sind grundsätzlich in der Abgabe nachzuweisen. Dies geschieht neben der Dokumentation des Programmcodes in den Quelldateien und zusätzlich in dem abgegebenen PDF-Dokument mit der Lösungsbeschreibung durch geeignete Tests, deren Auswahl und Eignung begründet werden müssen. Als **Referenzplattform** gelten die Linux-Rechner im Rechnerpool in MZH E0.