

# Betriebssysteme

## Tutorium 6

Philipp Kirchhofer  
philipp.kirchhofer@student.kit.edu  
<http://www.stud.uni-karlsruhe.de/~uxbtt/>

Lehrstuhl Systemarchitektur  
Universität Karlsruhe (TH)

10. Februar 2010

# Was machen wir heute?

## 1 Tutorien Übungsblatt

- Device Classes
- I/O Techniques
- Polling I/O
- DMA

## Tutorien Übungsblatt - Device Classes

### Frage 13.1.a

Weshalb wird eine Festplatte als Blockgerät und nicht als zeichenbasiertes Gerät kategorisiert? Was wären die Konsequenzen eines direkten byteweisen Zugriffs auf eine Festplatte?

### Antwort

Jedes Gerät hat bestimmte Zugriffscharakteristiken.

Eine Festplatte bietet hohen Durchsatz beim Lesen von großen Blöcken an Daten. Andererseits hat eine Festplatte hohe Latenzzeiten.

Für eine Festplatte bietet sich für die maximale Gesamtperformance an den Zugriff blockweise zu gestalten. Ein byteweiser Zugriff würde den Durchsatz erheblich reduzieren.

Anderer Geräte dagegen, wie z.B. ein Modem arbeiten auf Zeichenebene, bieten keinen wahlfreien Zugriff und können auch keine bereits ausgelesenen Daten erneut bereitstellen. Für diese Geräte bietet sich eine Kategorisierung als zeichenbasiertes Gerät an.

## Tutorien Übungsblatt - Device Classes

### Frage 13.1.b

Weshalb wird eine serielle Schnittstelle als zeichenbasiertes Gerät charakterisiert?

### Antwort

Daten werden von einer seriellen Schnittstelle sequentiell gelesen oder gesendet. Es gibt keine Datenblöcke, die am Stück verarbeitet werden müssen und es gibt keine Möglichkeit des wahlfreien Zugriffs auf den Datenstrom.

### Frage 13.1.b

Wäre es möglich eine serielle Schnittstelle als blockbasiertes Gerät zu verwalten?

### Antwort

Man könnte z.B. die empfangenen Daten sammeln und als Block fester Größe an ein Programm weitergeben. Die Empfangslatenz der Schnittstelle wird dabei aber erhöht und es muss zusätzlicher Speicher für den Puffer bereitgestellt werden. Weiterhin kann es beim Senden von Datenblöcken passieren, dass das Gerät keine ausreichend großen internen Sendepuffer besitzt und Daten verloren gehen.

Welche Rolle spielt der Prozessor beim Transfer von Daten von einem Gerät in den Speicher?

## Frage 13.2.a

Programmed I/O (polling)

## Antwort

Der Prozessor koordiniert den gesamten Ablauf: Er schickt Kommandos an das Gerät, wartet auf eine Antwort (polling) und schreibt die Antwort des Geräts in den Speicher.

## Frage 13.2.c

DMA (Direct Memory Access)

## Antwort

Der Prozessor startet die I/O Transaktion und übergibt einen Speicherbereich, in dem die Antwort des Geräts gespeichert werden soll. Anschließend lässt der Prozessor einen anderen Prozess laufen bis das Gerät einen Interrupt auslöst. Danach kann der Prozessor direkt mit der bereits im Speicher vorliegenden Antwort des Geräts weiterarbeiten.

## Frage 13.2.b

Interrupt-driven I/O

## Antwort

Der Prozessor startet die I/O Transaktion und lässt anschließend einen anderen Prozess laufen. Das Gerät löst nach dem Bearbeiten der Anfrage einen Interrupt aus. Beim Behandeln des Interrupts schreibt der Prozessor dann die Antwort des Geräts in den Speicher.

Ein Prozessor läuft mit 400 MHz Taktfrequenz. Eine Poll-Operation dauert 400 Zyklen. Wie sieht der Overhead bei den folgenden Geräten aus?

## Frage 13.3.a

Maus mit einer Pollrate von 3000 Hz

## Antwort

$$3.000 \frac{\text{PollOps}}{\text{s}} * 400 \frac{\text{Zyklen}}{\text{PollOp}} = 1.200.000 \frac{\text{Zyklen}}{\text{s}}$$

$$\frac{1.200.000 \frac{\text{Zyklen}}{\text{s}}}{400 * 10^6 \frac{\text{Zyklen}}{\text{s}}} = 3.000 * 10^{-6} = 3 * 10^{-3} = 0,3\%$$

## Frage 13.3.b

Diskette, Datentransfer 16 Bit pro Polloperation mit 50 kB/s

## Antwort

$$\frac{50 \frac{\text{kB}}{\text{s}} / 2 \frac{\text{Byte}}{\text{PollOp}}}{(25.000 \frac{\text{PollOps}}{\text{s}} * 400 \frac{\text{Zyklen}}{\text{PollOp}}) / 400 * 10^6 \frac{\text{Zyklen}}{\text{s}}} = 25.000 \frac{\text{PollOp}}{\text{s}}$$

$$(25.000 \frac{\text{PollOps}}{\text{s}} * 400 \frac{\text{Zyklen}}{\text{PollOp}}) / 400 * 10^6 \frac{\text{Zyklen}}{\text{s}} = 25 * 10^{-3} = 2,5\%$$

## Frage 13.3.c

Festplatte, Datentransfer 32 Bit pro Polloperation mit 8 MB/s

## Antwort

$$\frac{8 \frac{\text{MB}}{\text{s}} / 4 \frac{\text{Byte}}{\text{PollOp}}}{(2.000.000 \frac{\text{PollOps}}{\text{s}} * 400 \frac{\text{Zyklen}}{\text{PollOp}}) / 400 * 10^6 \frac{\text{Zyklen}}{\text{s}}} = 2.000.000 \frac{\text{PollOp}}{\text{s}}$$

$$(2.000.000 \frac{\text{PollOps}}{\text{s}} * 400 \frac{\text{Zyklen}}{\text{PollOp}}) / 400 * 10^6 \frac{\text{Zyklen}}{\text{s}} = 2 = 200\%$$

## Tutorien Übungsblatt - DMA

## Frage 13.4.b

Was für Vorteile bringt ein eigener E/A Bus für angeschlossene Geräte?

## Antwort

Bei jeder DMA Operation wird ein Lese- und ein Schreibzugriff ausgelöst, dadurch wird zweimal der Hauptbus, an dem auch der Prozessor und die angeschlossenen Geräte hängen, blockiert. In dieser Zeit kann der Prozessor keine weiteren Daten oder Instruktionen aus dem Speicher holen.

Durch die Einführung eines eigenen E/A Bus für die angeschlossenen Geräte wird der Hauptbus nur noch einmal (Schreiben in Hauptspeicher) pro DMA Operation blockiert. Es wird also der Datenverkehr auf dem Hauptbus deutlich verringert.

## Frage 13.4.a

Wie kann in Pseudocode eine DMA Übertragung dargestellt werden?

## Antwort

```
baseAddr = getMemoryDestination();
for count = 0 to blockSize - 1 do
    data = deviceRead(count);
    memoryWrite(baseAddr + count, data);
od
```

## Tutorien Übungsblatt - DMA

## Frage 13.4.c

Was ist ein fly-by Transfer?

## Antwort

Daten werden direkt vom Gerät in den Speicher kopiert und nicht erst in einem internen Register des DMA Controllers zwischengespeichert.

Fragen & Kommentare?

Wiederholungsstunde

AN x64 PROCESSOR IS SCREAMING ALONG AT BILLIONS OF CYCLES PER SECOND TO RUN THE XNU KERNEL, WHICH IS FRANTICALLY WORKING THROUGH ALL THE POSIX-SPECIFIED ABSTRACTION TO CREATE THE DARWIN SYSTEM UNDERLYING OS X, WHICH IN TURN IS STRAINING ITSELF TO RUN FIREFOX AND ITS GECKO RENDERER, WHICH CREATES A FLASH OBJECT WHICH RENDERS DOZENS OF VIDEO FRAMES EVERY SECOND

BECAUSE I WANTED TO SEE A CAT JUMP INTO A BOX AND FALL OVER.



I AM A GOD.

xkcd: Abstraction