Übungen zu Systemnahe Programmierung in C (SPiC)

Moritz Strübe, Rainer Müller
(Lehrstuhl Informatik 4)

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Wintersemester 2012/2013

Inhalt

Linux-Benutzerumgebung

POSIX-Verzeichnis-Systemschnittstelle
- opendir / closedir
- readdir
- stat

Aufgabe 6

Arbeitsumgebung

- Abgabe:
  - Gleiches Arbeitsverzeichnis wie in GSPIC jedoch höhere Aufgabennummern

    ```
    cd /proj/i4gspic/aufgabeX
    ```

  - Anders Abgabeskript: `/proj/i4spic/bin/submit aufgabe0`
- Editor:
  - Windows: AVRStudio oder Notepad++
  - Linux: kate, mcedit
- Übersetzen:
  - Manuell

    ```
    gcc -std=c99 -pedantic -Wall -Werror -D_BSD_SOURCE
                                    -o spider spider.c
    ```

  - Mit Make:

    ```
    export CFLAGS="-std=c99 -pedantic -Wall -Werror
                                    -D_BSD_SOURCE"
    export CC=gcc
    make spider
    ```

POSIX-Verzeichnis-Systemschnittstelle

- Verzeichnisse öffnen: `opendir(3)`
- Verzeichnisse lesen: `readdir(3)`
- Verzeichnisse schließen: `closedir(3)`

opendir/closedir

- Funktions-Prototypen:

```
1 #include <sys/types.h>
2 #include <dirent.h>
3 DIR *opendir(const char *dirname);
4 int closedir(DIR *dirp);
```

- Argument von opendir
  - `dirname`: Verzeichnisname
- Rückgabewert: Zeiger auf Datenstruktur vom Typ `DIR` oder `NULL`
- initialisiert einen internen Zeiger des directory-Funktionsmoduls auf den ersten Directory-Eintrag (für den ersten `readdir`-Aufruf)
- `closedir` schliesst ein geöffnetes Verzeichnis nach Bearbeitungsende

readdir

- liefert einen Directory-Eintrag (interner Zeiger) und setzt den Zeiger auf den folgenden Eintrag
- Funktions-Prototyp:

```
1 #include <sys/types.h>
2 #include <dirent.h>
3 struct dirent *readdir(DIR *dirp);
```

- Argumente
  - `dirp`: Zeiger auf `DIR`-Datenstruktur (von `opendir(3)`)
- Rückgabewert: Zeiger auf Datenstruktur vom Typ `struct dirent` oder `NULL`, wenn `EOF` erreicht wurde oder im Fehlerfall

⇒ bei `EOF` bleibt errno unverändert (behält vorherigen Wert), im Fehlerfall wird `errno` entsprechend gesetzt

⇒ `errno` vorher auf `0` setzen, sonst kann `EOF` nicht sicher erkannt werden!

readdir

- Problem: Der Speicher für die zurückgelieferte struct dirent wird von den dir-Bibliotheksfunktionen selbst angelegt und bei jedem Aufruf wieder verwendet!
  - werden Daten aus der dirent-Struktur länger benötigt, müssen sie vor dem nächsten readdir-Aufruf in Sicherheit gebracht (kopiert) werden
  - konzeptionell schlecht
    ⇒ aufrufende Funktion arbeitet mit Zeiger auf internen Speicher der readdir-Funktion
  - in nebenläufigen Programmen (mehrere Threads) nicht einsetzbar!
    ⇒ man weiß evtl. nicht, wann der nächste readdir-Aufruf stattfindet
- readdir ist ein klassisches Beispiel für schlecht konzipierte Schnittstellen in der C-Funktionsbibliothek

struct dirent

- Definition unter Linux (`/usr/include/bits/dirent.h`)

```
1 struct dirent {
2     __ino_t d_ino;
3     __off_t d_off;
4     unsigned short int d_reclen;  /*  tatsaechl. Laenge  der Struktur *
5     unsigned char d_type;
6     char d_name[256];
7 };
```

- POSIX: `d_name` ist ein Feld unbestimmter Länge, max. `NAME_MAX` Zeichen

Datei-Attribute ermitteln: stat

- liefern Datei-Attribute aus dem Inode
- Funktions-Prototyp:

```
1 #include <sys/types.h>
2 #include <sys/stat.h>
3 int stat(const char *path, struct stat *buf);
```

- Argumente:
  - `path`: Dateiname
  - `buf`: Zeiger auf Puffer, in den Inode-Informationen eingetragen werden
- Rückgabewert: `0` wenn OK, `-1` wenn Fehler
- Beispiel:

```
1 struct stat buf;
2 stat("/etc/passwd", &buf); /* Fehlerabfrage ... */
3 printf("Inode-Nummer: %d\n", buf.st_ino);
```

stat: Ergebnisrückgabe im Vergleich zur readdir

- problematische Rückgabe auf funktions-internen Speicher wie bei readdir gibt es bei stat nicht
- Grund: stat ist ein Systemaufruf - Vorgehensweise wie bei readdir wäre gar nicht möglich
- der logische Adressraum des Anwendungsprogramms ist nur eine Teilmenge (oder sogar komplett disjunkt) von dem logischen Adressraum des Betriebssystems

⇒ Betriebssystemspeicher ist für Anwendung nicht sichtbar/zugreifbar

⇒ Funktionen des Kernels (wie stat) können keine Zeiger auf ihre internen Datenstrukturen an Anwendungen zurückgeben

stat / lstat: stat-Struktur

- foo
  - `dev_t st_dev`; Gerätenummer (des Dateisystems) = Partitions-Id
  - `ino_t st_ino`; Inodenummer (Tupel st_dev,st_ino eindeutig im System)
  - `mode_t st_mode`; Dateimode, u.a. Zugriffs-Bits und Dateityp
  - `nlink_t st_nlink`; Anzahl der (Hard-) Links auf den Inode
  - `uid_t st_uid`; UID des Besitzers
  - `gid_t st_gid`; GID der Dateigruppe
  - `dev_t st_rdev`; DeviceID, nur für Character oder Blockdevices
  - `off_t st_size`; Dateigröße in Bytes
  - `time_t st_atime`; Zeit des letzten Zugriffs (in Sekunden seit 1.1.1970)
  - `time_t st_mtime`; Zeit der letzten Veränderung (in Sekunden ...)
  - `time_t st_ctime`; Zeit der letzten Änderung der Inode-Information
  - `blksize_t st_blksize`; Blockgröße des Dateisystems
  - `blksize_t st_blocks`; Anzahl der von der Datei belegten Blöcke

stat / lstat: st_mode

- st_mode enthält Informationen über den Typ des Eintrags:
  - `S_IFMT  0170000` bitmask for the file type bitfields
  - `S_IFSOCK 0140000` socket
  - `S_IFLNK  0120000` symbolic link
  - `S_IFREG  0100000` regular file
  - `S_IFBLK  0060000` block device
  - `S_IFDIR  0040000` directory
  - `S_IFCHR  0020000` character device
  - `S_IFIFO  0010000` FIFO
- Zur einfacheren Auswertung werden Makros zur Verfügung gestellt:
  - `S_ISREG(m)` - is it a regular file?
  - `S_ISDIR(m)` - directory?
  - `S_ISCHR(m)` - character device?
  - `S_ISLNK(m)` - symbolic link? (Not in POSIX.1-1996.)

Aufgabe 6

- einfaches find-Programm (SPIc Directory Evaluator Recursive)

```
spider <path> [<minSize>]
```

- durchlauft rekursiv den Verzeichnisbaum mit Wurzel path
  - Eintrage, deren Namen mit einem `"."` beginnen, werden ignoriert
  - symbolische Links werden nicht verfolgt (Gefahr von Zyklen!)
  - optional: nur Eintrage mit der Mindestgrosse (in Bytes) minSize
- gibt die Namen aller gefundenen Verzeichniseintrage aus
  - den Pfad des Eintrags relativ zu path angehangt an path
- welche stat-Funktion ist zu verwenden?
- Wichtig: Fehlerbehandlung nicht vergessen!