

Hardwarenahe Programmierung

Übungsaufgaben – 21. November 2022

Diese Übung enthält Punkteangaben wie in einer Klausur. Um zu „bestehen“, müssen Sie innerhalb von 85 Minuten unter Verwendung ausschließlich zugelassener Hilfsmittel 15 Punkte (von insgesamt 29) erreichen.

Aufgabe 1: Kondensator

Ein Kondensator der Kapazität $C = 100 \mu\text{F}$ ist auf die Spannung $U_0 = 5 \text{ V}$ aufgeladen und wird über einen Widerstand $R = 33 \text{ k}\Omega$ entladen.

- (a) Schreiben Sie ein C-Programm, das den zeitlichen Spannungsverlauf in einer Tabelle darstellt. (5 Punkte)
- (b) Schreiben Sie ein C-Programm, das ermittelt, wie lange es dauert, bis die Spannung unter 0.1 V gefallen ist. (4 Punkte)
- (c) Vergleichen Sie die berechneten Werte mit der exakten theoretischen Entladekurve: $U(t) = U_0 \cdot e^{-\frac{t}{RC}}$ (3 Punkte)

Hinweise:

- Für die Simulation zerlegen wir den Entladevorgang in kurze Zeitintervalle dt . Innerhalb jedes Zeitintervalls betrachten wir den Strom I als konstant und berechnen, wieviel Ladung Q innerhalb des Zeitintervalls aus dem Kondensator herausfließt. Aus der neuen Ladung berechnen wir die Spannung am Ende des Zeitintervalls.
- Für den Vergleich mit der exakten theoretischen Entladekurve benötigen Sie die Exponentialfunktion `exp()`. Diese finden Sie in der Mathematik-Bibliothek: `#include <math.h>` im Quelltext, beim `gcc`-Aufruf `-lm` mit angeben.
- $Q = C \cdot U$, $U = R \cdot I$, $I = \frac{dQ}{dt}$

Aufgabe 2: Personen-Datenbank

Wir betrachten das folgende Programm ([aufgabe-2.c](#)):

```
#include <stdio.h>
#include <string.h>

typedef struct
{
    char first_name[10];
    char family_name[20];
    char day, month;
    int year;
} person;

int main (void)
{
    person sls;
    sls.day = 26;
    sls.month = 7;
    sls.year = 1951;
    strcpy (sls.first_name, "Sabine");
    strcpy (sls.family_name, "Leutheusser-Schnarrenberger");
    printf ("%s_%s_wurde_am_%d.%d.%d_geboren.\n",
            sls.first_name, sls.family_name, sls.day, sls.month, sls.year);
    return 0;
}
```

Die Standard-Funktion `strcpy()` bewirkt ein Kopieren eines Strings von rechts nach links, hier also z. B. die Zuweisung der String-Konstanten "Sabine" an die String-Variable `sls.first_name[]`.

Das Programm wird für einen 32-Bit-Rechner kompiliert und ausgeführt.

(Die `gcc`-Option `-m32` sorgt dafür, daß `gcc` Code für einen 32-Bit-Prozessor erzeugt.)

```
$ gcc -Wall -O -m32 aufgabe-2.c -o aufgabe-2
$ ./aufgabe-2
Sabine Leutheusser-Schnarrenberger wurde am 110.98.1701278309 geboren.
Speicherzugriffsfehler
```

- (a) Erklären Sie die Ausgabe des Programms einschließlich der Zahlenwerte. (4 Punkte)
- (b) Welche Endianness hat der verwendete Rechner? Begründen Sie Ihre Antwort. (1 Punkt)
- (c) Wie sähe die Ausgabe auf einem Rechner mit entgegengesetzter Endianness aus? (2 Punkte)
- (d) Erklären Sie den Speicherzugriffsfehler. (Es kann sein, daß sich der Fehler auf Ihrem Rechner nicht bemerkbar macht. Er ist aber trotzdem vorhanden.) (2 Punkte)

Aufgabe 3: Fakultät

Die Fakultät $n!$ einer ganzen Zahl $n \geq 0$ ist definiert als:

$$\begin{aligned} &1 \quad \text{für } n = 0, \\ &n \cdot (n - 1)! \quad \text{für } n > 0. \end{aligned}$$

Mit anderen Worten: $n! = 1 \cdot 2 \cdot 3 \cdot \dots \cdot n$.

Die folgende Funktion `fak()` berechnet die Fakultät *rekursiv* (Datei: `aufgabe-3.c`):

```
int fak (int n)
{
    if (n <= 0)
        return 1;
    else
        return n * fak (n - 1);
}
```

- (a) Schreiben Sie eine Funktion, die die Fakultät *iterativ* berechnet, d. h. mit Hilfe einer Schleife anstelle von Rekursion. (3 Punkte)
- (b) Wie viele Multiplikationen (Landau-Symbol) erfordern beide Versionen der Fakultätsfunktion in Abhängigkeit von `n`? Begründen Sie Ihre Antwort. (2 Punkte)
- (c) Wieviel Speicherplatz (Landau-Symbol) erfordern beide Versionen der Fakultätsfunktion in Abhängigkeit von `n`? Begründen Sie Ihre Antwort. (3 Punkte)

Viel Erfolg!