

# Hardwarenahe Programmierung

## Übungsaufgaben – 16. November 2023

Diese Übung enthält Punkteangaben wie in einer Klausur. Um zu „bestehen“, müssen Sie innerhalb von 90 Minuten unter Verwendung ausschließlich zugelassener Hilfsmittel 15 Punkte (von insgesamt 31) erreichen.

### Aufgabe 1: Trickprogrammierung

Wir betrachten das folgende Programm (Datei: [aufgabe-1.c](#)):

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

int main (void)
{
    uint64_t x = 4262939000843297096;
    char *s = &x;
    printf ("%s\n", s);
    return 0;
}
```

Das Programm wird kompiliert und auf einem 64-Bit-Little-Endian-Computer ausgeführt:

```
$ gcc -Wall -O aufgabe-1.c -o aufgabe-1
aufgabe-1.c: In function 'main':
aufgabe-1.c:7:13: warning: initialization from incompatible pointer type [...]
$ ./aufgabe-1
Hallo
```

- (a) Erklären Sie die Warnung beim Compilieren. (2 Punkte)
- (b) Erklären Sie die Ausgabe des Programms. (5 Punkte)
- (c) Wie würde die Ausgabe des Programms auf einem 64-Bit-Big-Endian-Computer lauten? (3 Punkte)

Hinweis: Modifizieren Sie das Programm und lassen Sie sich Speicherinhalte ausgeben.

### Aufgabe 2: Thermometer-Baustein an I<sup>2</sup>C-Bus

Eine Firma stellt einen elektronischen Thermometer-Baustein her, den man über die serielle Schnittstelle (RS-232) an einen PC anschließen kann, um die Temperatur auszulesen. Nun wird eine Variante des Thermometer-Bausteins entwickelt, die die Temperatur zusätzlich über einen I<sup>2</sup>C-Bus bereitstellt.

Um das neue Thermometer zu testen, wird es in ein Gefäß mit heißem Wasser gelegt, das langsam auf Zimmertemperatur abkühlt. Alle 10 Minuten liest ein Programm, das auf dem PC läuft, die gemessene Temperatur über beide Schnittstellen aus und erzeugt daraus die folgende Tabelle:

Zeit / min.	Temperatur per RS-232 / °C	Temperatur per I <sup>2</sup> C / °C
0	94	122
10	47	244
20	30	120
30	24	24
40	21	168

- (a) Aus dem Vergleich der Meßdaten läßt sich auf einen Fehler bei der I<sup>2</sup>C-Übertragung schließen. Um welchen Fehler handelt es sich, und wie ergibt sich dies aus den Meßdaten? (5 Punkte)
- (b) Schreiben Sie eine C-Funktion `uint8_t repair(uint8_t data)`, die eine über den I<sup>2</sup>C-Bus empfangene fehlerhafte Temperatur `data` korrigiert. (5 Punkte)

### Aufgabe 3: Speicherformate von Zahlen

Wir betrachten das folgende Programm ([aufgabe-3.c](#)):

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

typedef struct
{
    uint32_t a;
    uint64_t b;
    uint8_t c;
} three_numbers;

int main (void)
{
    three_numbers xyz = { 1819042120, 2410670883059281007, 0 };
    printf ("%s\n", &xyz);
    return 0;
}
```

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

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

```
$ gcc -Wall -m32 aufgabe-2.c -o aufgabe-2
aufgabe-2.c: In function "main":
aufgabe-2.c:14:13: warning: format "%s" expects argument of type "char *", but
argument 2 has type "three_numbers * {aka struct <anonymous> *}" [-Wformat=]
    printf ("%s\n", &xyz);
                ^
$ ./aufgabe-2
Hallo, Welt!
```

- (a) Erklären Sie die beim Compilieren auftretende Warnung. (2 Punkte)
- (b) Erklären Sie die Ausgabe des Programms. (4 Punkte)
- (c) Welche Endianness hat der verwendete Rechner? Wie sähe die Ausgabe auf einem Rechner mit entgegengesetzter Endianness aus? (2 Punkte)
- (d) Dasselbe Programm wird nun für einen 64-Bit-Rechner compiliert und ausgeführt.  
(Die `gcc`-Option `-m64` sorgt dafür, daß `gcc` Code für einen 64-Bit-Prozessor erzeugt.)

```
$ gcc -Wall -m64 aufgabe-2.c -o aufgabe-2
aufgabe-2.c: In function "main":
aufgabe-2.c:14:13: warning: format "%s" expects argument of type "char *",
but argument 2 has type "three_numbers * {aka struct <anonymous> *}"
[-Wformat=]
    printf ("%s\n", &xyz);
                ^
$ ./aufgabe-2
Hall15V
```

(Es ist möglich, daß die konkrete Ausgabe auf Ihrem Rechner anders aussieht.)

Erklären Sie die geänderte Ausgabe des Programms. (3 Punkte)

*Viel Erfolg!*