

# Hardwarenahe Programmierung / Angewandte Informatik

## Übungsaufgaben – 19. Dezember 2016

Diese Übung enthält Punkteangaben wie in einer Klausur. Um zu „bestehen“, müssen Sie innerhalb von 120 Minuten unter Verwendung ausschließlich zugelassener Hilfsmittel 20 Punkte (von insgesamt 42) erreichen.

### Aufgabe 1: Bürgerentscheid

Bei einem [Bürgerentscheid](#) stimmen 66066 Wahlberechtigte mit „Ja“ und 65104 Wahlberechtigte mit „Nein“. Ein Rechenwerk vergleicht die Zahlen miteinander. Zu welchem Wahlergebnis – jeweils mit Begründung – kommt man

- (a) auf einem vorzeichenbehafteten 32-Bit-Rechenwerk? (2 Punkte)
- (b) auf einem vorzeichenlosen 16-Bit-Rechenwerk? (3 Punkte)
- (c) auf einem vorzeichenbehafteten 16-Bit-Rechenwerk? (3 Punkte)

### Aufgabe 2: Lokale Variable im Speicher

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

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

int main (void)
{
    int8_t a = -1;
    uint32_t b = 770;
    uint16_t c = 513;
    printf ("%8x\n", &a);
    printf ("%8x\n", &b);
    printf ("%8x\n", &c);
    return 0;
}
```

Das Programm wird auf einem 32-Bit-Little-Endian-Rechner kompiliert (mit drei Warnungen) und gestartet:

```
$ gcc -Wall -O aufgabe-2.c -o aufgabe-2
aufgabe-2.c: In function 'main':
aufgabe-2.c:9:3: warning: format '%x' expects argument of type 'unsigned int',
                but argument 2 has type 'int8_t *' [-Wformat]
aufgabe-2.c:10:3: warning: format '%x' expects argument of type 'unsigned int',
                but argument 2 has type 'uint32_t *' [-Wformat]
aufgabe-2.c:11:3: warning: format '%x' expects argument of type 'unsigned int',
                but argument 2 has type 'uint16_t *' [-Wformat]
$ ./aufgabe-2
ffd148ef
ffd148e8
ffd148e6
```

- (a) Begründen Sie die Warnungen und, soweit möglich, die ausgegebenen Zahlen. (4 Punkte)
- (b) Skizzieren Sie die Anordnung der Variablen – einschließlich Zahlenwerte – in den Speicherzellen (Bytes). (4 Punkte)
- (c) Wie würde die Anordnung der Variablen – einschließlich Zahlenwerte – in den Speicherzellen (Bytes) auf einem 8-Bit-Big-Endian-Rechner lauten? (4 Punkte)

### Aufgabe 3: Blinkende LEDs

Wir betrachten das folgende Programm für einen ATmega32-Mikro-Controller (Datei: [aufgabe-3.c](#)).

```
#include <stdint.h>
#include <avr/io.h>
#include <avr/interrupt.h>

uint8_t counter = 1;
uint8_t leds = 0;

ISR (TIMER0_COMP_vect)
{
    if (counter == 0)
    {
        leds = (leds + 1) % 8;
        PORTC = leds << 4;
    }
    counter++;
}

void init (void)
{
    cli ();
    TCCR0 = (1 << CS01) | (1 << CS00);
    TIMSK = 1 << OCIE0;
    sei ();
    PORTB = 0;
    DDRC = 0x70;
}

int main (void)
{
    init ();
    while (1)
        ; /* do nothing */
    return 0;
}
```

An die Bits Nr. 4, 5 und 6 des Output-Ports C des Mikro-Controllers sind LEDs angeschlossen. Sobald das Programm läuft, blinken diese in charakteristischer Weise:

Phase	LED oben (rot)	LED Mitte (gelb)	LED unten (grün)
1	aus	aus	an
2	aus	an	aus
3	aus	an	an
4	an	aus	aus
5	an	aus	an
6	an	an	aus
7	an	an	an
8	aus	aus	aus

Jede Phase dauert etwas länger als eine halbe Sekunde. Nach 8 Phasen wiederholt sich das Schema.

Erklären Sie das Verhalten des Programms anhand des Quelltextes:

- Wieso macht das Programm überhaupt etwas, wenn doch das Hauptprogramm nach dem Initialisieren lediglich eine Endlosschleife ausführt, in der *nichts* passiert? (1 Punkt)
- Wieso wird die Zeile `PORTC = leds << 4;` überhaupt aufgerufen, wenn dies doch nur unter der Bedingung `counter == 0` passiert, wobei die Variable `counter` auf 1 initialisiert, fortwährend erhöht und nirgendwo zurückgesetzt wird? (2 Punkte)
- Wie kommt das oben beschriebene Blinkmuster zustande? (2 Punkte)
- Wieso dauert eine Phase ungefähr eine halbe Sekunde? (2 Punkte)
- Was bedeutet „`ISR (TIMER0_COMP_vect)`“? (1 Punkt)

Hinweis:

- Die Funktion `init()` sorgt dafür, daß der Timer-Interrupt Nr. 0 des Mikro-Controllers etwa 488mal pro Sekunde aufgerufen wird. Außerdem schaltet sie eventuell an Port B angeschlossene weitere LEDs aus und initialisiert Port C als Output-Port. Sie selbst brauchen die Funktion `init()` nicht weiter zu erklären.

## Aufgabe 4: Objektorientierte Tier-Datenbank

```
#include <stdio.h>

#define ANIMAL 0
#define WITH_WINGS 1
#define WITH_LEGS 2

typedef struct animal
{
    int type;
    char *name;
} animal;

typedef struct with_wings
{
    int wings;
} with_wings;

typedef struct with_legs
{
    int legs;
} with_legs;

int main (void)
{
    animal *a[2];

    animal duck;
    a[0] = &duck;
    a[0]—>type = WITH_WINGS;
    a[0]—>name = "duck";
    a[0]—>wings = 2;  ← ((with_wings *) a[0])—>wings = 2;

    animal cow;
    a[1] = &cow;
    a[1]—>type = WITH_LEGS;
    a[1]—>name = "cow";
    a[1]—>legs = 4;  ← ((with_legs *) a[1])—>legs = 4;

    for (int i = 0; i < 2; i++)
        if (a[i]—>type == WITH_LEGS)
            printf ("A_%s_has_%d_legs.\n", a[i]—>name,
                    ((with_legs *) a[i])—>legs);
        else if (a[i]—>type == WITH_WINGS)
            printf ("A_%s_has_%d_wings.\n", a[i]—>name,
                    ((with_wings *) a[i])—>wings);
        else
            printf ("Error_in_animal:_%s\n", a[i]—>name);

    return 0;
}
```

Das oben in Blau dargestellte Programm (Datei: [aufgabe-2a.c](#)) soll Daten von Tieren verwalten.

Beim Compilieren erscheinen die folgende Fehlermeldungen:

```
$ gcc -std=c99 -Wall -O aufgabe-2a.c -o aufgabe-2a
aufgabe-2a.c: In function 'main':
aufgabe-2a.c:31: error: 'animal' has no member named 'wings'
aufgabe-2a.c:37: error: 'animal' has no member named 'legs'
```

Der Programmierer nimmt die oben in Rot dargestellten Ersetzungen vor (Datei: [aufgabe-2b.c](#)).

Daraufhin gelingt das Compilieren, und die Ausgabe des Programms lautet:

```
$ gcc -std=c99 -Wall -O aufgabe-2b.c -o aufgabe-2b
$ ./aufgabe-2b
A duck has 2 legs.
Error in animal: cow
```

- Erklären Sie die o. a. Compiler-Fehlermeldungen. (2 Punkte)
- Wieso verschwinden die Fehlermeldungen nach den o. a. Ersetzungen? (3 Punkte)
- Erklären Sie die Ausgabe des Programms. (5 Punkte)
- Beschreiben Sie – in Worten und/oder als C-Quelltext – einen Weg, das Programm so zu berichtigen, daß es die Eingabedaten ("A duck has 2 wings. A cow has 4 legs.") korrekt speichert und ausgibt. (4 Punkte)

*Viel Erfolg!*