

Hardwarenahe Softwareentwicklung – Klausur – 18. März 2013

Prof. Dr. Peter Gerwinski, Sommersemester 2012

Name:	
Matrikel-Nr.:	

Zeit: 120 Minuten

Zulässige Hilfsmittel:

- Schreibgerät
- Beliebige Unterlagen in Papierform
- Beliebige Unterlagen in elektronischer Form
(Skripte, Dokumentation, Beispiel-Programme, Software)
- Von der Hochschule zur Verfügung gestellte Rechner

Nur die o. a. zulässigen Hilfsmittel dürfen sich während der Klausur im Arbeitsbereich befinden. Mobiltelefone, Geräte mit mobilem Internet-Zugang u. ä. sind auszuschalten und in der Tasche zu verstauen.

Die reguläre Maximalpunktzahl beträgt 42 Punkte.
Bei besonderen Leistungen sind Zusatzpunkte möglich.
Mit 20 erreichten Punkten gilt die Klausur als bestanden.

Aufgabe 1: Länge von Strings

Strings werden in der Programmiersprache C durch Zeiger auf **char**-Variable realisiert.

Beispiel: **char** *hello_world = "Hello,_world!\n"

Das Ende des Strings wird durch ein Null-Zeichen gekennzeichnet (Zeichen mit numerischem Wert 0 – nicht zu verwechseln mit der Ziffer '0').

Die Systembibliothek stellt eine Funktion `strlen()` zur Ermittlung der Länge von Strings zur Verfügung (`#include <string.h>`).

- (a) Wie lang ist die Beispiel-String-Konstante "Hello, world!\n", und wieviele **char**-Speicherzellen belegt sie? (2 Punkte)

- (b) Schreiben Sie eine eigene Funktion `int strlen (char *s)`, die die Länge eines Strings zurückgibt. (4 Punkte)

- (c) Was passiert, wenn man als Parameter an `strlen()` einen Zeiger auf ein Array von `char`-Variablen übergibt, das *kein* Null-Zeichen am Ende hat? (3 Punkte)

Wir betrachten nun die folgenden Funktionen:

```
int fun_1 (char *s)
{
    int x = 0;
    for (int i = 0; i < strlen (s); i++)
        x += s[i];
    return x;
}
```

```
int fun_2 (char *s)
{
    int i = 0, x = 0;
    int len = strlen (s);
    while (i++ < len)
        x += *s++;
    return x;
}
```

(d) Was bewirken die beiden Funktionen? (2 Punkte)

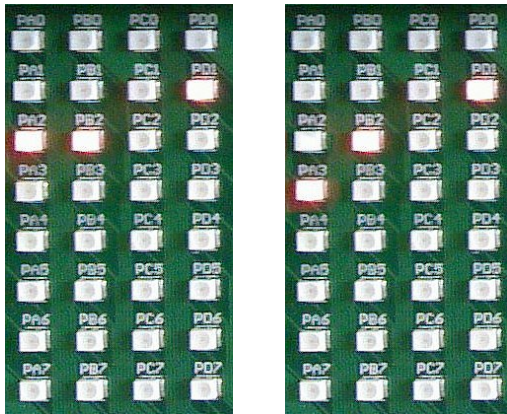
(e) Von welcher Ordnung (Landau-Symbol) sind die beiden Funktionen hinsichtlich der Anzahl ihrer Zugriffe auf die Zeichen im String – und warum? (4 Punkte)

(f) Schreiben Sie eine eigene Funktion, die dieselbe Aufgabe erledigt wie `fun_2()`, nur effizienter. (4 Punkte)

Aufgabe 2: Mikro-Controller

An die vier Ports eines ATmega16-Mikro-Controllers sind Leuchtdioden angeschlossen:

- von links nach rechts
an die Ports A, B, C und D,
- von oben nach unten
an die Bits Nr. 0 bis 7.



Wir betrachten das folgende C-Programm für diesen Mikro-Controller:

```
#include <avr/io.h>
#include <util/delay.h>
```

```
int main (void)
{
    DDRA = 0xff;
    PORTA = 4;
    while (1)
    {
        _delay_ms (250);
        PORTA ^= 12;
    }
    return 0;
}
```

Das Programm bewirkt ein Blinken zweier LEDs: Man sieht abwechselnd für jeweils eine Viertelsekunde die beiden links abgebildeten Leuchtmuster.

(a) Erklären Sie, wie das Programm das LED-Blinkmuster hervorruft. (5 Punkte)

(b) Wie würde sich das Programm verhalten, wenn es anstelle der Anweisung `PORTA = 4` die Anweisung `PORTA = 0` enthielte? (2 Punkte)

Aufgabe 4: Prüfwert für Nachrichten

Bei Datenübertragung wird häufig zusätzlich zur eigentlichen Nachricht ein Prüfwert mitgesendet, um Übertragungsfehler erkennen (und evtl. sogar korrigieren) zu können. Ein häufig verwendeter Bestandteil dieses Prüfwerts ist die Anzahl der 1-Bits in der Nachricht.

- (a) Wieviele 1-Bits enthält die Nachricht "Hello, world!\n"? (1 Punkt)

[illegible]

- (b) Schreiben Sie eine Funktion `int count_bits (char *s)`, die für eine gegebene Nachricht `s` die Anzahl der 1-Bits zurückliefert. (8 Punkte)