Echtzeitsysteme: Einführung

Sommersemester 2020
Dr.-Ing. René Tschirley

BEUTH HOCHSCHULE
FÜR TECHNIK
BERLIN
University of Applied Sciences

Leitfaden zu den Übungsaufgaben

letzte Änderung: 29. April 2020

Es folgen ein paar Regel für die Lösung der Aufgaben.

1. Befolgen Sie die Aufgabenstellung genau. Alles, was nicht als „Tipp“ gekennzeichnet ist, ist eine An- forderung. Ist diese nicht erfüllt, werden Punkte abgezogen.
2. Wenn die Aufgabenstellung fordert, dass Sie eine Variable var vom Typen uint16_t verwenden oder berechnen, werden Punkte abgezogen, falls die Variable VAR, variable oder sonstwie heißt. Es werden Punkte abgezogen, falls der Datentyp irgendetwas anderes ist als gefordert. Auch unsigned short wäre hier nicht korrekt. Für Funktionen und Datentypen gilt das Gleiche.
3. Legen Sie für jedes Aufgabenenblatt einen Ordner an in dem sich alle Implementationen finden. (Ausnahme Aufgabenblatt 4, Aufgabe 1)
4. Alternativ:
   (a) Schreiben Sie für jedes Aufgabenblatt ein Makefile.
   (b) Erstellen Sie für jedes Aufgabenblatt ein neues Projekt für Ihre IDE. Sorgen Sie dafür, dass Ihr Programm die geforderte Struktur enthält und sich auch per Makefile bauen lässt.

   Code der sich nicht bauen lässt, wird nicht akzeptiert.
5. Jede Datei benötigt einen Kommentarkopf:

```
/**
 * Name      : Ihr Familienname
 * Matrikelnr: Ihre Matrikelnummer
 */
```

6. Wenn Ihr Code nicht oder nur mit Warnungen übersetzt werden kann, werden Punkte abgezogen.
7. Externe Quellen und Autoren sind anzugeben.
8. Formatieren Sie ihren Quellcode ordentlich, damit Ihn auch andere Leute lesen können. Falls Ihre IDE das nicht kann, ist das Tool `astyle` ihr Freund.
9. Sie müssen bei der Abgabe in der Lage sein, Ihren Code zu erläutern.
10. Verwenden Sie immer die folgenden Compilerflags: `-W -Wall -Wextra -Werror -std=c11`
11. Geben Sie Fehlermeldungen auf `stderr` und nicht auf `stdout` aus.
12. Fehlermeldungen sollen dem Benutzer helfen.
13. Programme, welche Kommandozeilenparameter benötigen, müssen bei einem fehlerhaften Aufruf eine sinnvolle Usage-Meldung (auf `stderr`) ausgeben.

14. Ihre `main()`-Funktion sollte signifkant kürzer als 50 Zeilen sein, ansonsten ist Ihr Softwaredesign schlecht. Unterteilen Sie Ihren Quellcode in sinnvolle Funktionen.

15. Ihr Code soll keine Code-Duplikate enthalten, dafür gibt es Funktionen.