Rheinische Friedrich-Wilhelms Universität Bonn
Institut für Informatik VI, Technische Informatik

Prof. Dr. Joachim K. Anlauf
Oguzhan Sezenlik

5. Übung für die Vorlesung

Rechnerorganisation

Sommersemester 2019

Abgabe: Donnerstag, 9.5.2019; Schicken Sie bitte den Quellcode (als lauffähige `.asm` Quelldateien) für die Programmieraufgabe zusätzlich per E-Mail an Ihren Tutor:
Alexandra Chebotareva: `s6alcheb@uni-bonn.de`
Michel Fischer: `michel-fischer@hotmail.de`

Aufgabe 1. *MIPS-Assembler Konventionen* 10 P.

Auf der Homepage finden Sie ein Assembler-Programm *mipsprog.asm*, das ein Unterprogramm `algorithm` aufruft und leider nur fehlerhaft funktioniert. Leider gibt es keine Dokumentation. Es ist nur bekannt, dass `algorithm` mit einer Zahl zwischen 1 bis 10 eine Berechnung durchführen und ansonsten eine Fehlermeldung ausgeben soll.

1. Führen Sie *mipsprog.asm* mit SPIM aus und probieren verschiedene gültige Eingaben. Sie werden merken, dass der Simulator eine Fehlermeldung ausgibt. Beschreiben Sie den Grund dafür und ergänzen Sie den Code, sodass der Fehler behoben ist.
2. Welche Berechnung führt das Unterprogramm `algorithm` aus?
3. Beschreiben Sie was bei einer ungültigen Eingabe, das heißt einer Zahl kleiner 1 oder größer 10 passiert. Erläutern Sie den Grund für das Verhalten in SPIM und ändern Sie `algorithm`, sodass das Programm korrekt beendet wird.
4. Sie haben bestimmt gemerkt, dass der verantwortliche Programmierer offenbar nicht sorgfältig genug gearbeitet hat. Untersuchen Sie das Programm auf Lesbarkeit und den Programmier-Konventionen für MIPS-Assembler, die in der Vorlesung vorgestellt wurden. Was fällt Ihnen auf? Welche Probleme sind zu erwarten, falls das Unterprogramm `algorithm` in einem größeren Projekt integriert werden soll?
5. Überarbeiten und verbessern Sie *mipsprog.asm*, sodass alle identifizierten Probleme und Fehler vermieden werden.

Aufgabe 2. *Bubblesort* 10 P.

Im Folgenden soll der Bubblesort-Algorithmus in MIPS-Assembler programmiert werden. Auf der Homepage finden sie hierzu die Datei *bubblesort.asm*. Ihr Hauptprogramm soll die dort vorgegebene Struktur aufweisen. Die drei Routinen `printString`, `printInt` und `readInt` können Sie für Ein- und Ausgabe-Operationen nutzen.

1. Implementieren Sie die Prozedur `readValues`, welche 10 Werte von der Tastatur einliest. Achten Sie darauf, Speicherplatz für die Eingaben zu reservieren!
2. Implementieren Sie die Prozedur `sortValues`, die diese 10 Werte sortiert.
3. Implementieren Sie die Prozedur `printValues` , welche das sortierte Array ausgibt.