Einführung in die Informatik

`08 Programmierung`

Prof. Dr. Niels Streekmann

Stand: 14. November 2024

Inhaltsverzeichnis

1 Kompetenzen und Lernergebnisse

Durch das Bearbeiten dieses Materialpaketes erwerben Sie diese Kompetenzen (Wissen, Fähigkeiten, Fertigkeiten zur Problemlösung):

TODO

Die oben genannten Kompetenzen erwerben Sie, indem Sie Lernziele erreichen, welche sich prüfen lassen.

Lernergebnisse: Sie können nachweislich[1]:

- TODO

2 Konzepte

- Nutzung des Simple 8-Bit CPU Simulator[2]

[1] Sie können das Erzielen der einzelnen Lernergebnisse beispielsweise bei einem Testat im Praktikum oder einer Aufgabe in der Modulprüfung nachweisen

[2] https://schweigi.github.io/assembler-simulator/

2.1 Prozessorbefehlssätze

2.1.1 CISC

2.1.2 RISC

2.2 Maschinensprache / Assembler

Unter Maschinensprache verstehen wir binär codierte Befehle, die eine CPU interpretieren und ausführen kann. Jedes von einem Computer auszuführende Programm muss in Maschinensprache im Arbeitsspeicher vorliegen, um von der CPU verstanden zu werden. Die Erstellung eines Programms in Maschinensprache ist für Menschen nur schwierig umsetzbar, da sich Befehle in ihrer Binärdarstellung nicht von Daten wie z.B. Zahlen oder Buchstaben unterscheiden.

Um die Programmierung von Computern einfacher zu machen, wurden daher bereits früh für Menschen verständlichere Programmiersprachen eingeführt. Die ersten dieser Programmiersprachen waren Assembler-Sprachen. Das Abstraktionsniveau dieser Sprachen orientiert sich im Gegensatz zu heute geläufigen Programmiersprachen wie z.B. Java sehr nah an der Funktionsweise einer CPU und den Befehlen der Maschinensprachen. Das führt auch dazu, dass Programme in Assembler-Sprachen nicht auf andere CPUs bzw. CPUs mit anderen Befehlssätzen übertragbar sind.

Auch wenn die Befehlssätze von CPUs unterschiedlich sind, verfügen sie doch über große Ähnlichkeiten, die sich z.B. aus der von-Neumann-Architektur ergeben. Daher sind viele Befehle in ähnlicher Form in allen Assembler-Sprachen zu finden. Dazu gehören u.a. die folgenden Befehle:

2.2.1 Move

Mit dem Move-Befehl lassen sich Daten innerhalb des Speichers verschieben bzw. vom Speicher in Register kopieren.

2.2.2 Mathematische Operationen

Die mathematischen Operationen umfassen z.B. die arithmetischen Befehle, die von der ALU ausgeführt werden. Typischerweise umfassen diese Befehle z.B. Addition, Subtraktion, Multiplikation, Division, sowie den Vergleich von Daten.

3 Material zum aktiven Lernen

3.1 Testat zu `Maschinensprache / Assembler`

Erstellen Sie ein Assembler-Programm, das eine einfache Caesar-Chiffre berechnet. Das Programm soll in einem Eingabe-String die Buchstaben, um eine gegebene Anzahl an Stellen verschieben. Das Programm soll im 8BitCPUSimulator lauffähig sein und das Ergebnis im Output-Fenster ausgeben.

Der Start des Programms könnte z.B. wie folgt aussehen (Sie können die Variablen aber auch nach dem Programm deklarieren):

```
; Caesar chiffre

    JMP start
offset:     DB 4 ;
plaintext:  DB "Hello World!" ;
            DB 0 ; String terminator
```

A Anhang: Literatur und weiterführendes Material