



5. Aufgabenblatt

22.5.2023

Konzepte der maschinennahen Programmierung, Unterprogramme, Rekursion

Aufgabe 1: Theoriefragen

- (a) Erläutern Sie die Funktionsweise des Registers `sp`.
- (b) Wie werden (lokale) Variablen üblicherweise realisiert?

Aufgabe 2: Variablentausch

Speziell bei Prozessoren und Mikrokontrollern mit einer geringen Anzahl von Registern stellt sich häufig das Problem, den Inhalt zweier Register miteinander vertauschen zu müssen, ohne dass ein weiteres, unbenutztes Register zur Verfügung steht. Eine erste Lösung des Problems liegt in der Verwendung eines Zwischenspeicherbereiches im Hauptspeicher. Eine andere Möglichkeit, die ausschließlich unter Verwendung z.B. der Register `r1` und `r2` funktioniert, besteht in der Verwendung einer logischen Verknüpfung mittels XOR.


1. Welchen Nachteil hat die Lösung mit Verwendung des Hauptspeichers oder des Stacks?
2. Schreiben Sie ein ARM-Assemblerprogramm, welches unter Verwendung des XOR den Variablentausch durchführt.¹ Erklärung zu XOR: `a` und `b` sind die Eingangsvariablen, `c` ist die Ausgangsvariable.

a	b	c
0	0	0
0	1	1
1	0	1
1	1	0

¹ Der Maschinenbefehl beim ARM-Assembler lautet `eor`.

Aufgabe 3: Bitmanipulation als Unterprogramm

Implementieren Sie die Bitmanipulation (vgl. 4. Übung) als Unterprogramm. Bisher haben Sie bei Übergabe von Parametern (Zahlen-)Werte übergeben. Dies wird auch als Call-by-Value Übergabe bezeichnet. Bei der sogenannten Call-by-Reference Übergabe werden die (Speicher-)Adressen der (Zahlen-)Werte übergeben. Übergeben Sie dem Un-



terprogramm zur Bitmanipulation die Adressen der Seriennummer und der Bitmasken über den Stack. Es ist hilfreich, sich den Stack aufzuzeichnen und die Offsets anzugeben.


Aufgabe 4: Berechnung eines Binomialkoeffizienten durch Rekursion

Einem Binomialkoeffizienten begegnet man nicht nur in der Mathematik. Dieser ist wie folgt definiert für $n, k \in \mathbb{N}$ mit $n \geq k \geq 0$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Durch die Fakultät wird ggf. mit sehr großen Zahlen gerechnet, was schnell zu einem Overflow oder aufwändigen Rechnungen führt. Es gibt eine rekursive Definition, die mit weniger großen Zahlen auskommt. Dabei gilt

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}, \binom{n}{0} = \binom{n}{n} = 1$$



Implementieren Sie den rekursiven Algorithmus in C und anschließend in ARM-Assembler.