

Grundbegriffe der Informatik — Aufgabenblatt 7

Matr.nr.:

Nachname:

Vorname:

Tutorium Nr.: Tutor*in:

Ausgabe: Freitag, 9.12.2022, 14:30 Uhr

Abgabe: Freitag, 16.12.2022, 12:30 Uhr
Online, oder in dem Holzkasten neben Raum -119
im UG des Info-Gebäudes (50.34)

Lösungen werden nur korrigiert, wenn sie

- handschriftlich erstellt sind (Tablet-Ausdruck erlaubt) und
- mit dieser Seite als Deckblatt
- in der oberen **linken** Ecke zusammengeheftet **rechtzeitig** abgegeben werden.

Abgaberegeln für Teilnehmer der Tutorien mit Online-Abgabe:

- handschriftlich erstellt (Scans und lesbare Fotos akzeptiert)
- **rechtzeitig**, mit diesem Deckblatt in **genau einer** PDF-Datei
- in ILIAS unter "Tutorien" im Ordner des richtigen Tutoriums abgeben.

*Von Tutor*in auszufüllen:*

erreichte Punkte

Blatt 7: / 20 (+4)

Blätter 7 – 7: / 20 (+4)

Blätter 1 – 7: / 144 (+4)

Aufgabe 7.1 (3 + 3 = 6 Punkte)

In dieser Aufgabe sollen Sie mit kleinen Programmen mit der Programmierung der MIMA etwas warm werden. Es bezeichnen in dieser Aufgabe $x, y, z \in \mathbb{Z}_2^{20}$ drei verschiedene Speicherstellen, die außerhalb des Speicherbereichs, in dem das Programm steht, liegen. Sie können die drei Speicherstellen zum Zwischenspeichern von Ergebnissen in Ihren Programmen verwenden.

- a) Sie haben die MIMA-Befehl **RAR** kennengelernt, die die Bits im Akkumulator um eine Stelle zyklisch nach rechts rotiert. Formal beschrieben setzt sie damit folgende Funktionalität

$$Akku \leftarrow \text{rar}(Akku) \text{ mit } \text{rar}(wx) = xw \quad (\text{mit } x \in \mathbb{Z}_2, w \in \mathbb{Z}_2^{23})$$

um. Schreiben Sie ein MIMA-Programm, das die Umkehrfunktion $\text{ral} : \mathbb{Z}_2^{24} \rightarrow \mathbb{Z}_2^{24}$ von rar realisiert, also den Wert des Akkumulator um eine Stelle zyklisch nach links rotiert und ihn zurück in den Akkumulator schreibt.

- b) Der Programmablauf in der MIMA kann durch bedingte Sprünge gesteuert werden. In der MIMA funktionieren bedingte Sprünge, indem überprüft wird, ob der Akkumulator $Akku$ eine negative Zahl enthält (also das höchstwertige Bit gesetzt ist).

Es gibt einen MIMA-Befehl **EQL** zur Prüfung der Gleichheit zweier Werte. Ziel dieser Teilaufgabe ist es, ein Programm zu schreiben, dass einen bedingten Sprung ausführt, wenn der Wert an der Speicherstelle a (echt) kleiner ist als der Wert an der Speicherstelle b (jeweils im Zweierkomplement interpretiert).

- i) Schreiben Sie ein Programm, das zwei positive Zahlen, die an den Adressen a und b hinterlegt sind, wie folgt vergleicht: Das Programm fährt genau dann mit der Ausführung an der Speicherstelle a_less_b fort, wenn die an a hinterlegte Zahl echt kleiner ist als die an b hinterlegte Zahl. Andernfalls fährt das Programm an der Speicherstelle a_geq_b fort.
- ii) Funktioniert ihr Programm auch, wenn beliebige Zahlen aus $\mathbb{K}_{24} = \{x \in \mathbb{Z} \mid -2^{23} \leq x \leq 2^{23} - 1\}$ in Zweierkomplementdarstellung an den Adressen a und b hinterlegt sind? Wenn ja, begründen Sie. Wenn nein, geben Sie ein Gegenbeispiel an.

Aufgabe 7.2 (2 + 3 + 1 = 6 Punkte)

In dieser Aufgabe geht es darum, die Funktionsweise eines gegebenen MIMA-Programms zu verstehen. Betrachten Sie dazu das Programm in Abbildung 1. Die Konstanten **NULL**, *item-ptr*, *next-ptr*, und x , bezeichnen darin MIMA-Adressen, die außerhalb des Speicherbereichs des Programms liegen, *START*, *add-next* und *was-last-element* sind Sprungziel-Adressen im Speicherbereich des Programms

- a) Geben Sie die Werte an den Speicherstellen *item-ptr*, *next-ptr* und x nach der einmaligen Ausführung des Programms vom Start bis inklusive Zeile 16 an. Geben Sie hierfür Ausdrücke an, die die Werte an diesen Adressen in Abhängigkeit vom Speicherzustand $m \in \text{Val}^{\text{Adr}}$ vor der Ausführung des Programmabschnittes beschreiben. Gehen Sie davon aus, dass der bedingte Sprung in Zeile 11 nicht durchgeführt wird.

```

START:  LDC  1
        ADD  item-ptr
        STV  next-ptr
        LDC  0
        STV  x
        STV  NULL
add-next: LDIV item-ptr
        ADD  x
        STV  x
        LDIV next-ptr
        EQL  NULL
        JMN  was-last-element           (11)
        LDIV next-ptr
        STV  item-ptr
        LDC  1
        ADD  item-ptr
        STV  next-ptr                   (16)
        JMP  add-next
was-last-element: LDV  x
                  HALT

```

Abbildung 1: Das zu analysierende Programm für Aufgabe 7.2

b) Sei nun der folgende Speicherausschnitt gegeben:

Bezeichner	Adresse	Wert
NULL	4	12
x	5	154
next-ptr	6	29
	7	33
	8	5
	9	14
	10	0
	11	29
	12	3
	13	0
	14	17
	15	12
item-ptr	16	8

Das Programm wird nun auf einer MIMA-Maschine ausgeführt, wobei zu Beginn die Adresse *START* des Programmbeginns im IAR steht. Die Ausführung endet, wenn sie den **HALT**-Befehl erreicht.

Das Programm beschreibt die Speicheradresse x mehrfach. Geben Sie die Folge von Werten an, die durch den Ablauf des Programmes an die Adresse x geschrieben werden.

- c) Beschreiben Sie informell, was das Programm berechnet.

Hinweis: Können Sie eine Datenstruktur, die Sie bereits in der Vorlesung „Programmieren“ kennengelernt haben, wiedererkennen?

Nun sollen Sie eigene kleine MIMA Programme entwerfen und implementieren. Verwenden Sie zum Prüfen Ihrer Programme den MIMA-Simulator *Mima Flux*, der in der Vorlesung vorgestellt wurde.¹

Bitte geben Sie die Lösungen für die beiden folgenden Aufgaben als Eingabedateien für das Programm *Mima Flux*, die von den Tutor:innen in diesem Programm geladen und getestet werden können.

Assemblerprogramme sind meist eher kryptisch und schwerer zu verstehen. Machen Sie sich und Ihrem:r Tutor:in eine Gefallen und verteilen Sie großzügig erklärende Kommentare über das Programm. Unzureichende Dokumentation (in Kommentaren) führt zu Punktabzug.

Aufgabe 7.3 (8 Punkte)

Sie haben auf dem vierten Aufgabenblatt die *Spiegelungsabbildung* $\bullet^R : A^* \rightarrow A^*$ für ein Alphabet A kennengelernt. Zur Erinnerung:

$$\begin{aligned} |w^R| &= |w| \\ w^R(i) &= w(|w| - i - 1) \quad \text{für alle } 0 \leq i < |w|. \end{aligned}$$

Schreiben Sie ein Programm, das ein Wort $w \in Z_2^{24}$ aus dem Speicher liest und dessen Spiegelung $w^R \in Z_2^{24}$ an eine andere Speicherstelle schreibt.

Geben Sie hierzu eine Datei im Eingabeformat von *Mima Flux* ab. Das Programm soll seine Eingabe von einer Speicherstelle mit dem symbolischen Namen *in* lesen und seine Ausgabe an die Speicherstelle mit dem symbolischen Namen *out* schreiben und dann anhalten.

Aufgabe 7.4 (4 Bonus²-Punkte)

Sie haben auf dem letzten Übungsblatt die Funktion *to_string* kennengelernt, die einen 0-terminierten Speicherbereich als Zeichenkette interpretiert. In dieser Bonus-Aufgabe sollen Sie ein Programm schreiben, das die Reihenfolge der Zeichen in einem 0-terminierten Speicherbereich umkehrt. Ihr Programm soll folgende Spezifikation erfüllen:

Die Eingabe des Programmes steht in *array*. Die Speicherstelle mit dem symbolischen Namen *array* enthalte zu Beginn als Wert die Adresse a des Beginns eines Speicherbereichs. Die Adresse b des Endes des Speicherbereichs ist die erste Adresse mit $b \geq a$, an der Wert 0 steht.

Nach Ausführung des Programmes sollen die Werte zwischen den Speicherstellen a (einschließlich) und b ausschließlich in umgekehrter Reihenfolge im Speicher stehen.

¹Mehr Information auf der Ilias-Seite. Verfügbar unter: <https://github.com/mattulbrich/mimaflux>

²Die Punkte dieser Aufgabe zählen nicht zur Gesamtpunktzahl, die über das Bestehen des Übungsscheines entscheidet.

Sie dürfen annehmen, dass es eine Speicherstelle jenseits von a gibt, die den Wert 0 enthält.