



Übungsblatt 12

Paralleles Programmieren, vom Programm zur Maschine

Abgabe bis: 12.07.2002, 12:00

Bonussystem: (*) Theorieaufgabe zum Korrigieren, (**) Programmieraufgabe zum Korrigieren

*Aufgabe 1: Systolische Automaten (3 Punkte)

Gegeben sei die Matrix $A = \begin{pmatrix} 1 & 5 & 8 & 2 \\ 6 & 7 & 8 & 5 \\ 9 & 4 & 1 & 3 \\ 1 & 4 & 2 & 9 \end{pmatrix}$ und der Vektor $b = \begin{pmatrix} 3 \\ 7 \\ 1 \\ 4 \end{pmatrix}$.

1. Berechnen Sie mit Hilfe eines geeigneten systolischen Automaten das Matrix-Vektor-Produkt $c = Ab$. Stellen Sie in einer Tabelle die jeweiligen Berechnungsschritte der parallel arbeitenden Prozessoren dar.
2. Nach wie vielen Schritten liegt der Ergebnisvektor vor?
3. Welchen Durchmesser hat das Prozessornetzwerk?

*Aufgabe 2: Netzwerk zur Berechnung von zweitgrößter Zahl (2 Punkte)

1. Entwerfen Sie ein Netzwerk aus $n/2$ Vergleichsmoduln, das für beliebige Zahlenfolgen der Länge n die zweitgrößte Zahl in einem Zeitaufwand von $O(\log n)$ bestimmt. Sie können davon ausgehen, dass die Zahlen über n Eingabeleitungen simultan an dem Netzwerk anliegen.
Hinweis: Überlegen Sie sich, was ein solches Vergleichsmodul leisten muss. Wie viele Eingänge braucht es und wie viele Ausgänge?
2. Wie viele Schritte (Taktzyklen) benötigen Sie, um für k Zahlenfolgen der Länge n jeweils die zweitgrößte Zahl zu berechnen?

Aufgabe 3: Grundkenntnisse maschinennaher Programmierung

1. Nennen Sie die 5 Hardware-Komponenten eines von-Neumann-Rechners.
2. Zählen Sie die wichtigsten semantisch unterschiedlichen "Befehlsarten" typischer Mikroprozessoren auf.

*Aufgabe 4: Speicherzugriffsarten (4 Punkte)

Gegeben sei der folgende Ausschnitt eines Speichers:

0x00:	ADD
0x01:	0x05
0x02:	0x0B
0x03:	0x01
0x04:	0x0D
0x05:	0x03
0x06:	0x04
0x07:	0x0C
0x08:	0x08
0x09:	0x0F
0x0A:	0x01
0x0B:	0x07
0x0C:	0x04
0x0D:	0x0C
0x0E:	0x07
0x0F:	0x0A

Die Speicherzellen seien nummeriert von 0x00 bis 0x0F wie angegeben. Der Befehl ADD führe eine Addition aus mit zwei Argumenten, die in den beiden nachfolgenden Speicherzellen stehen.

1. Was ist das Ergebnis der Addition (in hexadezimaler Schreibweise) bei direkter Adressierung beider Argumente?
2. Was ist das Ergebnis der Addition (in hexadezimaler Schreibweise) bei absoluter Adressierung beider Argumente?
3. Was ist das Ergebnis der Addition (in hexadezimaler Schreibweise) bei indirekter Adressierung beider Argumente?
4. Es sollen nun Werte aus zwei Arrays addiert werden. Dazu sind die beiden Argumente indiziert adressiert, es sollen die beiden Elemente mit dem Index 2 addiert werden. Was ist das Ergebnis der Addition?

Hinweis: Das erste Element eines Arrays habe den Index 0 und die Größe eines Arrayelements sei eine Speicherzelle.

***Aufgabe 5:** *Repräsentation verketteter Listen im Speicher (2 Punkte)*

Der folgende Speicherabschnitt (Zellengröße: 8 Bit) beinhaltet eine verkettete Liste. Ein Listenelement bestehe dabei aus einer 16-Bit Zahl und einer Referenz auf den Beginn des nächsten Elements. Der null-Zeiger sei dabei durch den Wert 0x00 repräsentiert. Das erste Element beginnt in der Speicherzelle 0x10. Stellen Sie die Liste graphisch dar und geben sie die Inhalte der Elemente in hexadezimaler Schreibweise an.

0x10:	0x35
0x11:	0x16
0x12:	0x19
0x13:	0x4B
0x14:	0x05
0x15:	0x1D
0x16:	0x1D
0x17:	0x01
0x18:	0xB3
0x19:	0x1C
0x1A:	0x15
0x1B:	0x14
0x1C:	0x16
0x1D:	0x00
0x1E:	0x14
0x1F:	0x00

****Aufgabe 6:** *Assembler-Programmierung (auf Papier)(4 Punkte)*

Gegeben seien folgende Assembler-Befehle.

- push n** legt den Wert **n** oben auf einen Stapel
- pop** entfernt den obersten Wert vom Stapel
- load a** legt den Wert, der in Speicherstelle Nr. **a** steht, oben auf den Stapel
- store a** entfernt den obersten Wert vom Stapel und schreibt ihn in Speicherstelle Nr. **a**
- add** entfernt die beiden obersten auf dem Stapel liegenden Werte, addiert sie und legt das Ergebnis oben auf den Stapel
- sub** entfernt die beiden obersten auf dem Stapel liegenden Werte, subtrahiert den obersten vom zweitobersten und legt das Ergebnis oben auf den Stapel
- mul** entfernt die beiden obersten auf dem Stapel liegenden Werte, multipliziert sie und legt das Ergebnis oben auf den Stapel
- div** entfernt die beiden obersten auf dem Stapel liegenden Werte, dividiert den zweitobersten durch den obersten und legt das Ergebnis oben auf den Stapel

Geben Sie ein in dieser Sprache formuliertes Programm an, das die Anweisung $m[9] = (3 + m[0]) - 5 * ((m[3] - 7) / (m[2] / m[1] + 7))$ berechnet. Dabei ist $m[i]$ der Wert, der in Speicherstelle i steht.