8. Aufgabenblatt

Ausgabe	Abgabe
21.06.11	01.07.11

Hinweis zu Programmieraufgaben: Bitte machen Sie zu jeder Programmieraufgabe Testläufe, die die Funktionialität ihrer Programme ausreichend dokumentieren. Die Testläufe und den Quellcode drucken Sie bitte aus, und geben beides mit der jeweiligen Übung auf Papier ab. Der Quellcode muss zusätzlich in einer kompilierbaren Datei per Mail an den jeweiligen Tutor geschickt werden. Als Betreff der Email geben Sie bitte Ihre Matrikelnummer und die Nummer des Aufgabenblattes an.
Die bewertete Aufgabe ist als solche gekennzeichnet!

8. Übungszettel

Problem 1: Sprungvorhersage

Betrachten wir 3 Vorhersagemethoden:

(1) Sprünge werden nie ausgeführt

(2) Sprünge werden immer ausgeführt

(3) Dynamische Vorhersage mit einer Vorhersagegenauigkeit im Durchschnitt von 90 %

Nehmen wir an, bei diesen Methoden entstehen keine Kosten, wenn die Vorhersage stimmt, und es entstehen Kosten in Höhe von 2 Zyklen, wenn die Vorhersage nicht stimmt.
Welches Verfahren eignet sich für die folgenden Sprünge am besten?

- Ein Sprung, der mit einer Häufigkeit von 5% ausgeführt wird.
- Ein Sprung, der mit einer Häufigkeit von 95% ausgeführt wird.
- Ein Sprung, der mit einer Häufigkeit von 70% ausgeführt wird.

Problem 2: dynamische Sprungvorhersage (bewertet)

a) Warum verwendet man eine dynamische Sprungvorhersage?

b) Wie funktioniert ein 1-Bit-Predictor bei der dynamischen Sprungvorhersage?

c) Was ist beim 2-Bit-Predictor anders? Für welche Fälle ist er besser geeignet?

d) Wie häufig liegen bei den folgenden Fällen der 1-Bit-Predictor und der 2-Bit-Predictor(Hysteresis Scheme) statistisch betrachtet richtig bei ihrer Vorhersage? Gehen Sie von einem großen Wert von n aus, sowie für eine Wahrscheinlichkeit von 50%, dass p wahr ist. Beide Prädiktoren starten dabei im Zustand „Predict Strongly Taken“.

1)
```
for i=1:n {
  ...
  }
```

2)
```
for i=1:n {
    for j=1:n {
    ...
    }
```

```
}
```

3)
```
for i=1:n {
  if p {
  ...
  }
}
```

Problem 3 – Pipelining - bewertet

Gegeben ist folgende Befehlsfolge (die Großbuchstaben dienen nur der Identifikation der Zeilen):

```
A       : a = a+1;
B       : a = a+1;
C       : a = a+1;
IF      : if( x == 0 ) {
E       :       a = a / 2;
F       :       a = a - 1;
          }
G       : a = a+1;
H       : a = a+1;
```

Die Befehlsfolge werde von einem Rechner abgearbeitet, der über die gleiche 5-Stufige Pipeline wie in der letzten Übung verfügt. Gehen Sie davon aus, dass in der if-Abfrage die Bedingung (x == 0) false liefert, die Sprungvorhersage aber falsch liegt, also (x == 0) = true vorhersagt.

a) Bestimmen Sie die Anzahl der Takte, die der Rechner für die vollständige Abarbeitung der Folge benötigt. Gehen Sie dabei davon aus, dass ein Löschen (“flushen”) der Pipeline zusätzliche 10 Takte kostet.

b) Schreiben Sie den Code so um, dass er für die bedingte Anweisung Predicated Instructions (s. Folien ab 3.146) verwendet. Bestimmen Sie wieder die Anzahl der Takte, die der Rechner für die vollständige Abarbeitung benötigt.

Hinweis: Für beide Teilaufgaben ist eine Visualisierung der Pipelinestufen notwendig, die den Pipelinezustand bei jedem Befehl erläutert!