TI II: Rechnerarchitektur SS 2007 Übungsblatt Nr. 3

Prof. Dr.-Ing. Jochen Schiller, AG Technische Informatik, Freie Universität Berlin

Ausgabe am 25.05.2007 — Abgabe spätestens 08.06.2007, 12:00 Uhr

Bitte bei der Abgabe beide Namen/Matr.Nr. der Mitglieder einer Gruppe, NUMMER DER ÜBUNG/TEILAUFGABE und DATUM auf den Lösungsblättern **nicht vergessen**! Darauf achten, dass die Lösungen beim richtigen Tutor/der richtigen Tutorin abgegeben werden.
Zu spät abgegebene Lösungen werden nicht mehr berücksichtigt!

---

1. Aufgabe: Befehlssatz (4 Punkte)

Welchen Grund könnte es geben, dass die Anweisung `beq $t0, $t1, LABEL` ohne Modifikation nicht ausgeführt werden kann, bzw. es ein Problem mit dem Label geben kann? Schauen Sie sich dazu das entsprechende Befehlsformat an. (2 Punkte)
Geben Sie eine Lösung für dieses Problem an. (2 Punkte)

2. Fehlerhaftes Programm (5 Punkte)

Das folgende Programm kopiert Wörter von der Adresse in Register `$a0` in die Adresse in Register `$a1` und zählt dabei die übertragenen Wörter in Register `$v0`. Das Programm beendet den Kopiervorgang, wenn es ein Wort mit dem Wert `0` erkennt. Der Inhalt der Register `$v1`, `$a0` und `$a1` muss nicht erhalten bleiben. Das abschließende Wort muss kopiert, darf aber nicht gezählt werden.

```
      addi $v0, $zero, 0    # initialisiere Zähler
loop: lw   $v1, $a0         # lese nächstes Wort ein
      sw   0($a1), $v1      # schreibe Wort an Zieladresse
      addi $a0, $a0, 4      # setze Zeiger auf nächstes Wort
      addi $a1, $a0, 4      # erhöhe Zieladresse um ein Wort
      beq  $v1, $zero, loop # verzweige, wenn nächstes Wort = 0
```

In diesem Programm befinden sich mehrere Fehler. Beheben Sie die Fehler und geben Sie eine fehlerfreie Version ab.

3. Aufgabe: Einfache Pipeline (8 Punkte)

Geben Sie für jede der folgenden Codesequenzen an, ob die Pipeline angehalten werden muss, ob Verzögerungen nur mit Forwarding vermieden werden können oder ob die Codesequenzen ohne Verzögerung oder Forwarding ausgeführt werden können. Gehen Sie von einer einfachen DLX-Pipeline aus.

Sequenz 1	**Sequenz 2**	**Sequenz 3**	**Sequenz 4**

`lw  $t0,0 ($t0)`
`add $t1,$t0,$t0`
	`add  $t1,$t0,$t0`
	`addi  $t2,$t0, 5`
	`addi  $t4,$t1, 5`
		`addi  $t1,$t0,1`
		`addi  $t2,$t0,2`
		`addi  $t3,$t0,2`
		`addi  $t3,$t0,4`
		`addi  $t5,$t0,5`
			`subi $t1, $t1, 1`
			`bne  $t1, $zero, label`
			`add  $t2, $t1, $t2`
			`sll  $t2, $t2, 2`
			`label:`

4. Aufgabe: Datenabhängigkeiten (5 Punkte)

Geben Sie sämtliche Datenabhängigkeiten im folgenden Code an. Bei welchen Abhängigkeiten handelt es sich um Datenkonflikte, die mittels Forwarding behoben werden können? Bei welchen Abhängigkeiten handelt es sich um Datenkonflikte, die zu einem Pipeline-Leerlauf führen? Gehen Sie von einer einfachen DLX-Pipeline aus.

```
add  $t3, $t4, $t2
sub  $t5, $t3, $t1
lw   $t6, 200($t3)
add  $t7, $t3, $t6
```

5. Aufgabe: Pipeline-Takte – Teil 1 (6 Punkte)

Gehen Sie im Folgenden von einer einfachen 5-stufigen Pipeline aus:
Befehl holen (IF), Befehl dekodieren (ID), Operanden holen (OF), Ausführung (EX), Rückspeichern (WB).
Weiterhin liegt eine reine Load/Store-Architektur ohne architekturelle Beschleunigungsmaßnahmen (z.B. forwarding, reordering etc.) oder Hardware zur Erkennung von Hemmnissen vor. Operanden können erst dann aus Registern geholt werden, nachdem sie zurück gespeichert wurden. `ADD X, Y, Z` steht für `Z:= X+Y`.

1. Ist die Die Befehlsfolge
   ```
   ADD R2, R3, R1
   ADD R1, R5, R4
   ```
   ausführbar?
2. Ist die Befehlsfolge
   ```
   ADD R2, R3, R1
   NOP
   ADD R1, R5, R4
   ```
   ausführbar?
3. Wie viele Takte dauert die vollständige Abarbeitung der folgenden Befehlsfolge? Fügen Sie unter Umständen eine geeignete Anzahl NOPs ein, damit die Folge bearbeitet werden kann.
   ```
   Z:=X+Y
   A:=Z+B
   ```

6. Aufgabe: Pipeline-Takte – Teil 2 (6 Punkte)

Gehen Sie im Folgenden von einer einfachen 6-stufigen Pipeline aus:
Befehl holen (IF), Befehl dekodieren (ID), Operanden holen (OF), Ausführung (EX), Befehlsbestätigung (CO), Rückspeichern (WB).
Weiterhin liegt eine reine Load/Store-Architektur ohne architekturelle Beschleunigungsmaßnahmen (z.B. forwarding, reordering etc.) oder Hardware zur Erkennung von Hemmnissen vor. Operanden können erst dann aus Registern geholt werden, nachdem sie zurück gespeichert wurden.

1. Wie lange (in Takten) dauert die vollständige Abarbeitung der folgenden Befehlsfolge? (4 Punkte)
   ```
   Z:=X*Y
   A:=Z*B
   D:=Z+B
   ```

2. Wie groß ist im Idealfall die Beschleunigung der Programmabarbeitung mit dieser Pipeline? (2 Punkte)