
Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Institut für Technische Informatik
Prof. Dr. rer. nat. Wolfgang Karl

Zentralübung Rechnerstrukturen

Aufgabenblatt 3: Parallelismus auf Befehlsebene

1 Moderne Mikroprozessoren

1.1 Leistungsbewertung

a) Wie definieren sich Latenz und Durchsatz? Geben Sie Latenz und maximalen Durchsatz bei einer 7-stufigen Pipeline und 5 maximal gleichzeitig in der Pipeline befindlichen Befehlen an.

b) Berechnen Sie den Speedup S zwischen der sequentiellen Ausführung von 95 Befehlen ohne Pipelining und der Ausführung in einer 6-stufigen Pipeline.

1.2 Pipelining

a) Betrachten Sie das Codestück in Listing 1. Führen Sie dieses zunächst sequentiell durch. Welchen Wert erhalten Sie am Ende in Register 2, welchen in 4? Welche Funktionalität erbringt dieses Programmstück?

```
1  INIT:   XOR   R0,R0,R0   ; R0=0
2          ADDI  R2,R0,#24  ; R2=24
3          ADDI  R3,R0,#8   ; R3=8
4
5  START:  SUB   R4,R2,R3   ; R4=R2-R3?
6          BLTZ  R4,L1      ; if (R2<R3) goto L1
7          BGTZ  R4,L2      ; if (R2>R3) goto L2
8          J     END        ; jump to END
9
10 L1:     SUB   R3,R3,R2   ; R3 = R3 - R2
11         J     START      ; goto START
12
13 L2:     SUB   R2,R2,R3   ; R2=R2-R3
14         J     START      ; goto START
15
16 END:
```

Listing 1: Programmstück in MIPS-Assembler zu Aufgabe 1.2

b) Bilden Sie die ersten 6 Befehle des Codes auf eine einfache 5-stufige Pipeline ab, in der Konflikte mittels Einfügen von Leerzyklen aufgelöst werden.

c) Welche Abhängigkeiten führen in dem Codefragment unter Verwendung der bekannten 5-stufige Pipeline zu Konflikten und wie lassen sich diese Konflikte vermeiden?

d) Wie lassen sich sogenannte *Stall Cycles* zwischen den Befehlen vermeiden?

e) Führen Sie den Code nun unter Berücksichtigung der Hardware-Techniken zur Vermeidung von Konflikten aus bei stets korrekter Sprungvorhersage. Die Berechnung der Sprungrichtung des Sprungziels während der Dekodierphase erfolge in einer separaten Adressberechnungseinheit nach Auswertung der Bedingung, die Sprungvorhersage erfolge in der Befehlsholstufe.

f) Vergleichen Sie die Ausführungszeiten zwischen sequentieller, einfacher 5-stufiger, und 5-stufiger Pipeline mit Result Forwarding und Pipeline Interlocking. Wie groß ist die Geschwindigkeitszunahme? Wie groß ist jeweils der Durchsatz?

1.3 Sprungvorhersage

a) Führen Sie den Codeabschnitt 2 (R0 sei das „Zero"-Register) auf einem Zwei-Bit-Prädiktor mit Hysteresezähler aus, wobei für jeden Sprung ein eigener Prädiktor zur Verfügung stehe. Diese Prädiktoren seien mit *Predict Weakly Taken* (WT) initialisiert.

```
1  INIT:   ADD  R1,R0,#0  ; R1=0
2          ADD  R2,R0,#2  ; R2=2
3
4  START:  BNE  R1,R0,L1  ; if (R1!=0) goto L1
5          ADD  R1,R0,#1  ; R1=1
6
7  L1:     SUB  R3,R1,R2  ; R3=R1-R2
8          BNE  R3,R0,L2  ; if (R1!=R2) goto L2
9          ADD  R1,R0,#0  ; R1=0
10         J    START     ; goto START
11
12 L2:     ADD  R1,R0,#2  ; R1=2
13         J    START     ; goto START
```

Listing 2: Programmstück in MIPS-Assembler zu Aufgabe 1.3

b) Vergleichen Sie anschließend mit der Ausführung auf einem (1,1)-Korrelationsprädiktor. Das globale Schieberegister sei dabei mit (NT) gefüllt, die zwei Prädiktoren seien mit T vorbelegt.

1.4 Befehlsausführung mit Prädikaten

a) Betrachten Sie nochmals Codeabschnitt 1. Wie könnte der Assembler-Code für die IA64-Architektur aussehen?

2 Superskalartechnik

2.1 Algorithmus von Tomasulo

a) Gegeben sei ein superskalarer Prozessor mit folgenden Eigenschaften:

- Interne Parallelisierung nach Tomasulo
- Pipeline bestehend aus Fetch, Decode, Issue+Renaming, 1x-7x Execute, 0x oder 1x oder 3x Memory Access, ggf. Writeback. Die Dispatch-und Retirement-Phasen werden weggelassen.
  IF ID IS EX MEM WB
- Zwei Lade-Speichereinheiten (*Load/Store Unit*), eine Integer-Additionseinheit, eine Integer-Multiplikationseinheit, eine FP-Additionseinheit
- *Delayed branches*, kein Anhalten (*Stalling*) und keine dynamische Vorhersage, sondern fortwährendes Füllen der Pipeline; dazu sei ein Sprungzieladresscache vorhanden und die statische Vorhersage laute auf *Taken*
- Volles Bypassing
- FP-Register und normale Register können gleichzeitig in der WB-Stufe beschrieben werden
- Die Befehlszuordnungs- und Rückordnungsbandbreite betrage 4 Befehle; zwei Befehle werden pro Takt maximal geholt
- Die Auswertung der Sprungzieladresse erfolge in der Stufe *Execute*; das Schreiben des Befehlszählers in der WB-Stufe
- Auf dem Ergebnisbus können Ressourcenkonflikte entstehen

Folgender Code werde darauf ausgeführt, wobei R0=0, R1 eine Speicheradresse, R2=R1+24 und F2 beliebig sei:

```
1 LOOP: LD.D   F0,0(R1)    ; loads Mem[i]
2       ADD.D  F4,F0,F2    ; adds to Mem[i]
3       S.D    0(R1),F4    ; stores into Mem[i]
4       ADD    R1,R1,#8    ;
5       SUB    R3,R1,R2    ; R3 = R1-R2
6       BLTZ   R3,LOOP     ; delayed branch if R1 < R2
```

Für die Ausführungseinheiten gelten folgende Zeiten:

Einheit	**L/S**	**Integer-Add**	**Integer-Mul**	**FP-Add**
Anzahl	2	1	1	1
Bearbeitungsdauer (in Takten)	3	1	3	2

Alle Einheiten bis auf die Divisionseinheit seien intern mit Pipelines implementiert.

Zeichnen Sie den Verlauf der Pipeline ab Beginn der Schleife unter der Annahme, dass alle vorhergehenden Befehle bereits vollständig durchgeführt und gültig gemacht worden seien, und führen Sie Buch über die Befehlswarteschlange (*Instruction Queue*), die Reservation Stations, die Registerstatustabelle und den Rückordnungspuffer (*Reorder Buffer*).

3 Hinweis

Benutzen Sie zur Beantwortung der Fragen auch die weiterführende Literatur.