TI2 11.7.2012

7. Moderene Architekturentwicklungen
(eine Übersicht)

Warum? Erhöhung der Rechnerleistung.

Leistung: Arbeit pro Zeit

Rechnerleistung:
➔ Näherung F7_10

Durchschnittsbefehle, warum?
Befehle unterschiedlich lang in der Ausführung und treten mit unterschioedlicher Häufigkeit in den Anwendungsprogrammen auf.

Bsp. Zu Rechnerleistung

Prozesssor mit 2GHz Frequenz vom Takt
Tcycle=0,5 ns

1-Taktbefehle p1=0,33
2-Taktbefehle p2=0,33
3-Taktbefehle p3=0,33

CPI= $\sum_{i=1}^{3} t_i \cdot p_i = (0{,}5 \cdot 0{,}33 + 1 \cdot 0{,}33 + 1{,}5 \cdot 0{,}33)$ ns

$0{,}5$ ns

$= 1 \cdot 0{,}33 + 2 \cdot 0{,}33 + 3 \cdot 0{,}33$

≈ 2

$IPC = 0{,}5$

$L = \frac{0{,}5}{0{,}5\ ns} = \frac{1}{ns}$

≈ 1000 MIPS

Relative Häufigkeit eines Befehls:

- konkrete Applikation: Messung des Auftretens bein n-maligem Durchlauf, n möglichst groß
- konkretes Anwendungsgebiet – Sammlung von Programmen über dieses Anwendungsgebiet, weiter w.o.
- ganz allgemein: Sammlung aus allen Fachgebieten, weiter wie oben

Architektubezogene Maßnahmen zur Leistungserhöhung:

RISC: (reduced instruction set computer, Rechner mit reduziertem Befehlssatz)

➔ wenige, wenig komplexe Befehle (einfacher Befehlssatz) -> F7_50

IF: instruction fetch (Befehl lesen, BL)
ID: instruction decode (Befehl dekodieren, BD)
OF: Operand fetch (Operand lesen, nur aus Register)

EX: execute (Befehl ausführen)
WB: write back (Operand schreiben, OS)

➔ komplexere Funktionen: -> mehrere Befehle -> Compiler

Vorteile im weiteren: Befehle sind im Prozessor parallelisierbar (d.h. im Prozessor sind mehrere Befehle gleichzeitig in Bearbeitung, allerdings evtl. in unterschiedlichen Phasen)

Bsp.
Rechner 1000 MIPS nicht parallel (ein Befehl aktuell immer in Arbeit)

Core 1000 MIPS Prozessor, 2 Cores parallel (Dual Core)

➔ 2000 MIPS Rechner

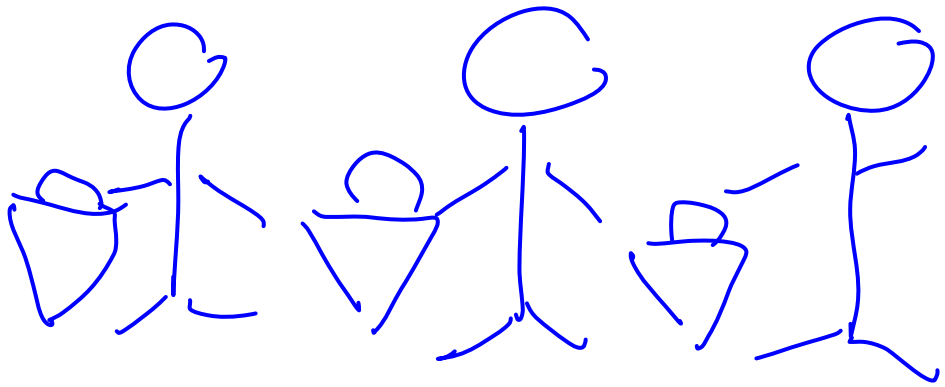
1 Single-Core (ein Prozessor parallel, aber Prozessor hat intern Parellelitäten)

(-> Mikroparallelität)

Mehrere Cores -> Makroparallelität

Pipeline (Mikroparallelität)

➔ F7_60
Pipeline (formale Übersetzungsbedeutung „Eimerkette“ (weitergeben von Eimern, schrittweise: n Personen geben n Eimer weiter)



Ausgabe 1 Eimer pro Zeit-Einheit

Zeiteinh. Weitergabe von 1 Pers. z nächsten

Ohne Pipeline oben
Takt 1GHz: 5 Takte/Befehl
200 MIPS

Unten mit Pipeline

Die einzelnen Phasen benutzen relativ unterschiedliche Funktionsteile im Prozessor, dh. Wenn Phase i von Befehl j abgearbeitet und Bef. i Phase j+1 ausführt kann Bef. j+1 Phase i ausführen usw.

➔ Abarbeitungszeit pro Befehl bleiben 5 Takte aber je Takt wird ein Befehl fertig

d.h.
1000 MIPS

➔ Leistung steigt um Faktor 5

Superscalar (Mikroparallelität)

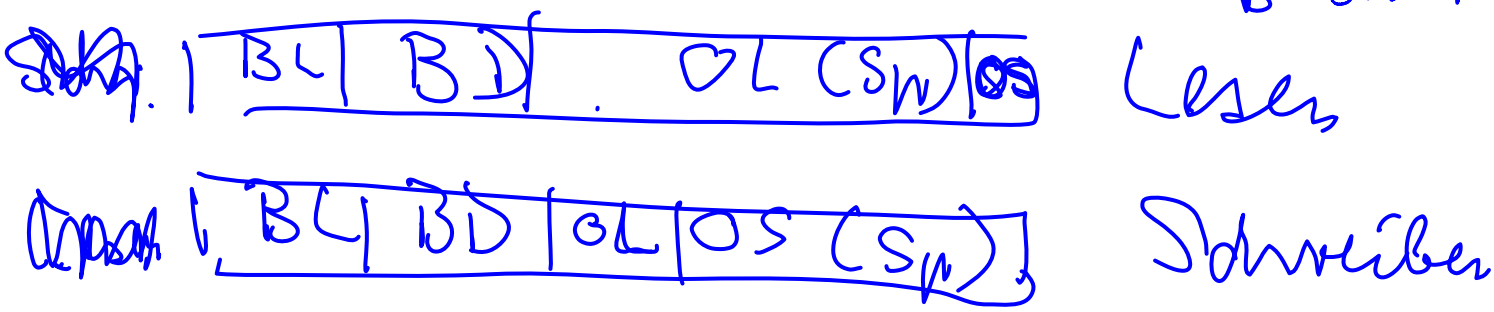
Wenn es möglich ist, in der gleichen Zeit mehr Befehle zu lesen als auszuführen, bestimmt die Ausführungszeit die Taktlänge (1 Phase ist ein Takt)

➔ Ausweg: mehrere Befehle gleichzeitig in der Ausführung -> parallele Ausführungseinheiten

Hardwarereduktion (als nicht Verdopplung be zwei Ausführungseinheite durch zwei spezialisiertemit nich überdeckender Funktionalität)

Begrenzung der Leistungssteigerung des Prozessors erfolgt auch der Speicherzugriffszeit
Phase Befehl Leesen (BL)
Bei Speicherbefehlen (OL,OS) zum Speicher

Speicherbefehle



Speicherhierarchie und Zugriffszeit bezogen auf Größe -> F7_160

Bisher Prozessorregister (BR, ORi) am schnellsten im Zugriff, wenige)
Hauptspeicher viel langsamer, sehr viele Daten- oder Befehlsworte

- > sinnvoll zwischen Registern und Hauptspeicher einen oder mehrere kleinere (und damit schnellere Speicher) anordnen, in denen die aktuell am häufigsten benötigten Befehle und Operanden als Kopien vom Hauptspeicher sind.
- Cache -> Grundidee Folie 7_170
  Enthät sowohl von der Kopie die Hauptspeicheradresse (im Tagbereich HS Ai,j,k) + zugeordneter Dateneintrag D (HS Ai,j,k)

➔ wenn Cache schneller ist als Hauptspeicher ist, ist der Zugriff auf Befehle und Daten, die sich in ihm befinden (HS-Adresse vorhanden), schneller