Rechnerstrukturen

Vorlesung im Sommersemester 2008

Prof. Dr. Wolfgang Karl

Universität Karlsruhe (TH)

Fakultät für Informatik

Institut für Technische Informatik

- HÖRSAALVERLEGUNG:
  - Wann:
    - 29. Mai 2008, 14:00 – 15:30 Uhr
    - 03. Juli 2008, 14:00 – 15:30 Uhr
  - Wo:
    - jeweils im Hörsaal HMO

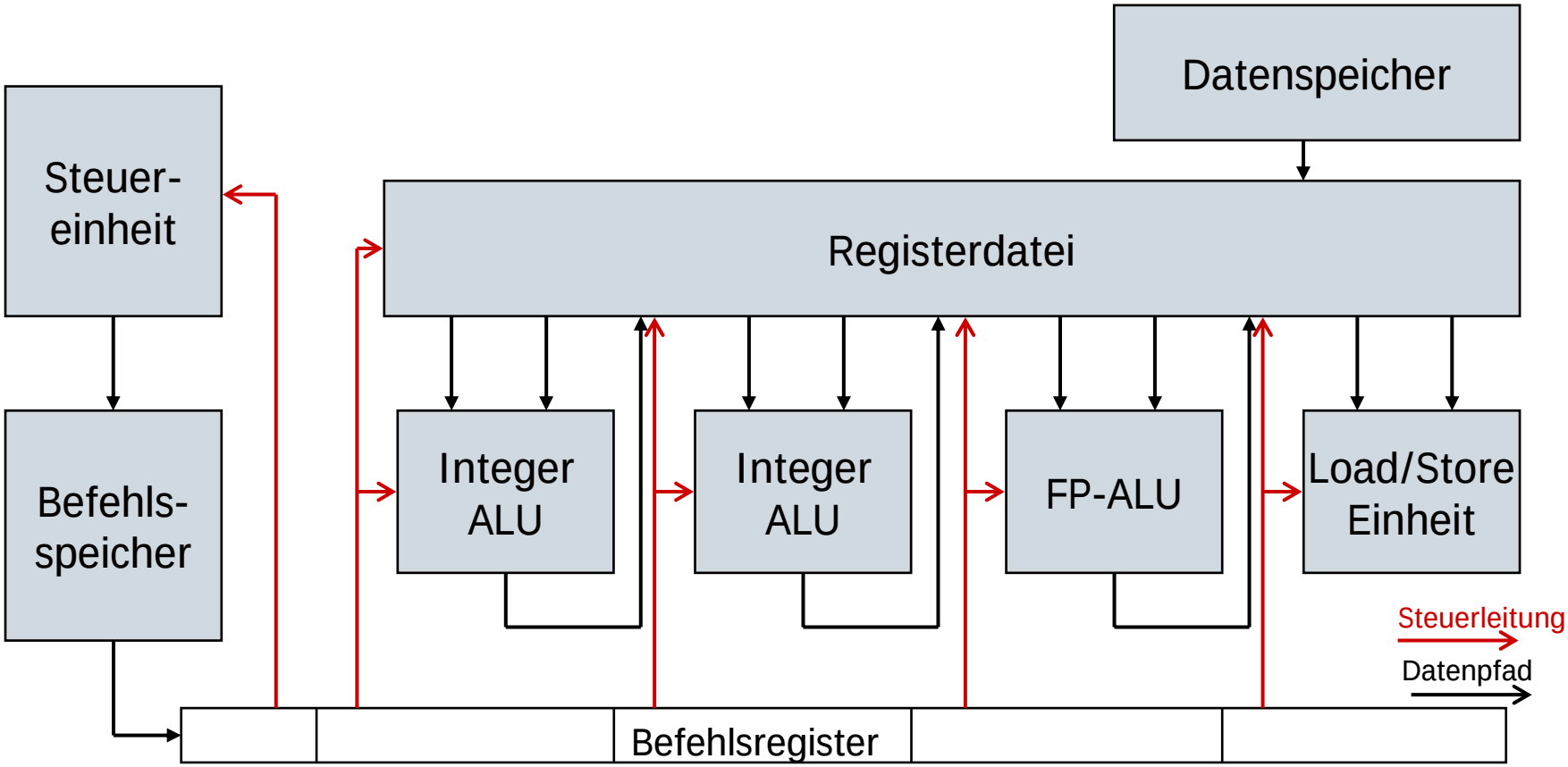
- **Kapitel 2: Parallelismus auf Befehlsebene**

2.3: Nebenläufigkeit, VLIW, EPIC

- Grundprinzip: VLIW
  - VLIW: Very Long Instruction Word
  - Eine VLIW-Architektur (Very Long Instruction Word) ist gekennzeichnet durch ein breites Befehlsformat, das in mehrere Felder aufgeteilt ist, aus denen die Funktionseinheiten gesteuert werden;
  - Eine VLIW-Architektur mit *n* voneinander unabhängigen Funktionseinheiten kann bis zu *n* Operationen gleichzeitig ausführen;
  - Operationen: RISC-Architektur
  - Steuerung der parallelen Abarbeitung zur Übersetzungszeit (automatisch parallelisierender Compiler)

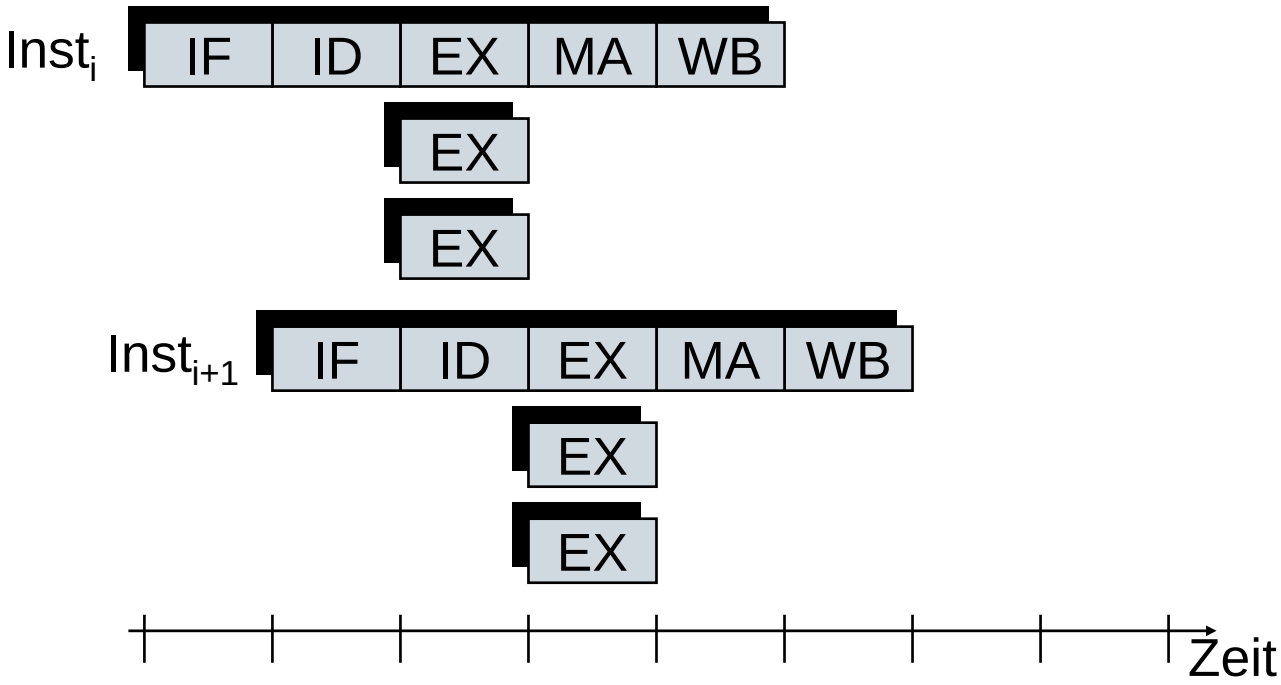
VLIW-Prinzip

- Grundprinzip VLIW
  - Prinzipieller Aufbau



VLIW-Prinzip

- Grundprinzip VLIW
  - Prinzipielle Pipeline-Struktur



VLIW-Prinzip

- Statische Steuerung der parallelen Abarbeitung
  - Aufgaben des Compilers
    - Frontend:
      - Lexikalische, syntaktische und semantische Analyse
    - Code-Generierung / Parallelisierung
      - Kontrollflussanalyse
      - Datenflussanalyse
      - Datenabhängigkeitsanalyse
      - Schleifenparallelisierung
        - » Loop Unrolling
        - » Software-Pipelining
    - Scheduling
      - Packen der voneinander unabhängigen Befehle in breite Befehlswörter

VLIW-Prinzip

- Statische Steuerung der parallelen Abarbeitung
  - Scheduling
    - Beispiel (Quelle: K. Asanovic, MIT)

```
for (i=0;i<N;i++)
B[i]=A[i]+C
```

Übersetzung ↓

```
Loop:  ld f1,0(r1)
       add r1,8
       fadd f2,f0,f1
       sd f2, 0(r2)
       add r2,8
       bne r1,r3,loop
```

VLIW-Prinzip

- Statische Steuerung der parallelen Abarbeitung
  - Scheduling (Fortsetzung Beispiel)

```
for (i=0;i<N;i++)
B[i]=A[i]+C
```

Übersetzung

```
Loop:  ld f1,0(r1)
       add r1,8
       fadd f2,f0,f1
       sd f2, 0(r2)
       add r2,8
       bne r1,r3,loop
```

Scheduling

FE Int1	FE Int2	FE Mem1	FE Mem2	FE FP+	FE FPx
add r1		ld			
				fadd	
add r2	bne	sd			

VLIW-Prinzip

- Statische Steuerung der parallelen Abarbeitung
  - Scheduling
    - Loop Unrolling

```
for (i=0;i<N;i++)
B[i]=A[i]+C
```

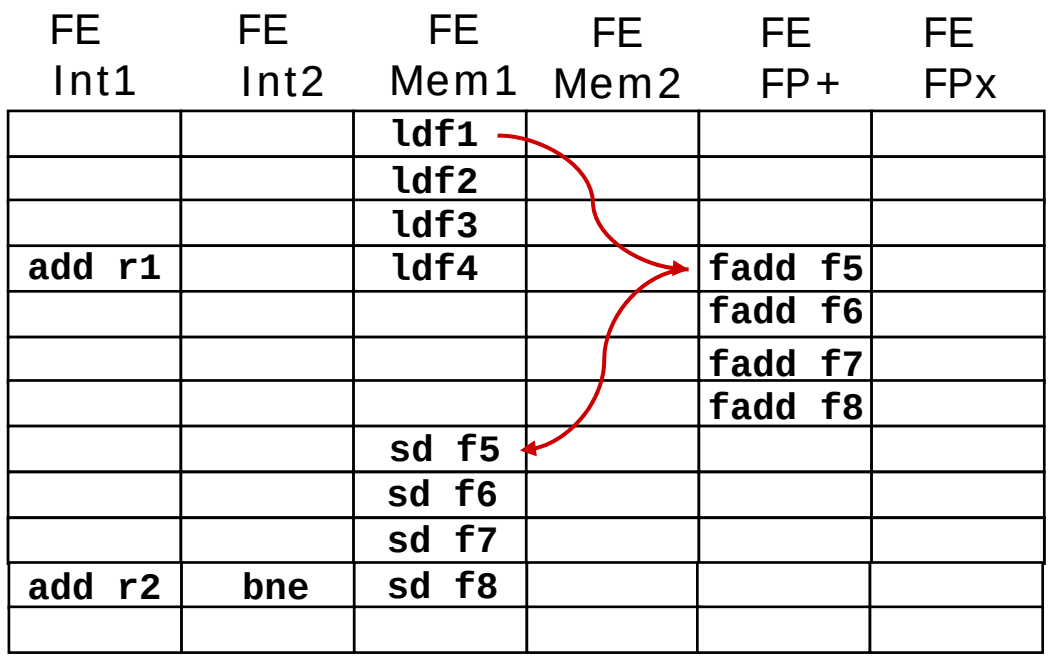
Abrollen der inneren Schleife

```
for (i=0;i<N-3;i+4)
{
B[i]=A[i]+C
B[i+1]=A[i+1]+C
B[i+2]=A[i+2]+C
B[i+3]=A[i+3]+C
}
```

VLIW-Prinzip

- Statische Steuerung der parallelen Abarbeitung
  - Loop Unrolling

```
Loop:   ld f1,0(r1)
        ld f2,8(r1)
        ld f3,16(r1)
        ld f4,24(r1)
        add r1,32
        fadd f5,f0,f1
        fadd f6,f0,f2
        fadd f7,f0,f3
        fadd f8,f0,f4
        sd f5,0(r2)
        sd f6,8(r2)
        sd f7,16(r2)
        sd f8,24(r2)
        add r2,32
        bne r1,r3,loop
```



VLIW-Prinzip

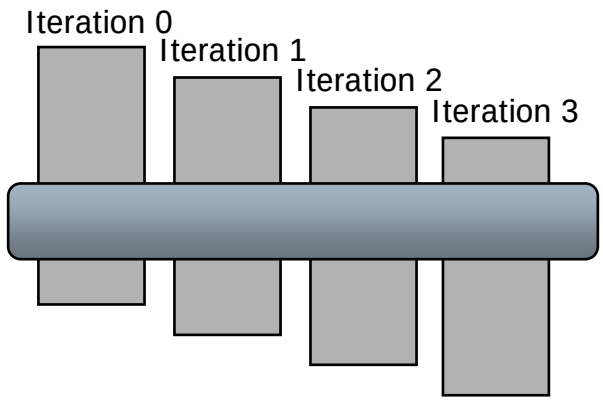
- Statische Steuerung der parallelen Abarbeitung
  - Software-Pipelining

```
Loop:   ld f1,0(r1)
        ld f2,8(r1)
        ld f3,16(r1)
        ld f4,24(r1)
        add r1,32
        fadd f5,f0,f1
        fadd f6,f0,f2
        fadd f7,f0,f3
        fadd f8,f0,f4
        sd f5,0(r2)
        sd f6,8(r2)
        sd f7,16(r2)
        add r2,32
        sd f8,-8(r2)
        bne r1,r3,loop
```

	FE Int1	FE Int2	FE Mem1	FE Mem2	FE FP+	FE FPx
Prolog			ldf1			
			ldf2			
			ldf3			
	add r1		ldf4			
			ldf1		fadd f5	
			ldf2		fadd f6	
			ldf3		fadd f7	
	add r1		ldf4		fadd f8	
Schleife			ldf1	sd f5	fadd f5	
			ldf2	sd f6	fadd f6	
		add r2	ldf3	sd f7	fadd f7	
	add r1	bne	ldf4	sd f8	fadd f8	
Epilog				sd f5	fadd f5	
				sd f6	fadd f6	
				sd f7	fadd f7	
				sd f8	fadd f8	
				sd f5		

VLIW-Prinzip

- Statische Steuerung der parallelen Abarbeitung
  - Software-Pipelining
    - Technik zur Reorganisation von Schleifen
      - Jede Iteration in dem mit Software-Pipelining generierten Code enthält Befehle aus verschiedenen Iterationen der ursprünglichen Schleife

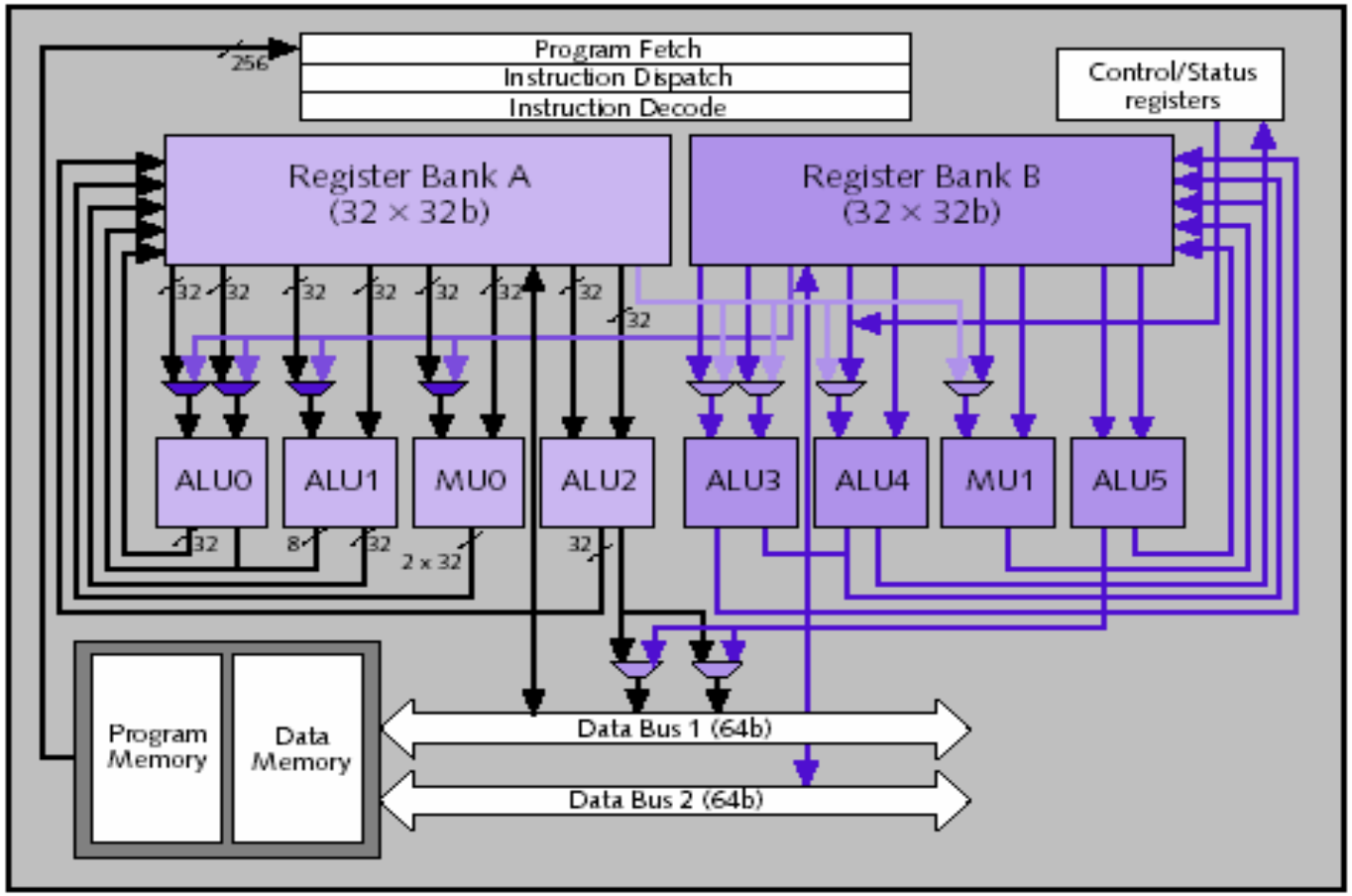


VLIW-Prinzip

- Frühe VLIW-Rechner
  - Multiflow Trace (J. Fisher)
    - Rechnerkonfigurationen mit 7, 14 und 28 Operationen/VLIW-Instruktion
    - 28 Operationen in einem 1024 Bit Wort
    - Numerische Anwendungen
    - Trace-Scheduling
  - Cydrome Cydra-5 (B. Rau)
    - 7 Operationen/256Bit Wort
    - Rotating Register File
- Einsatz VLIW-Technik heute:
  - Allzweck-Mikroprozessoren
    - Transmeta Crusoe
  - Bereich eingebetteter Prozessoren (DSP)
    - Beispiel: Trimedia TM32 Architecture
    - Agere/Motorola Star Core

VLIW-Prinzip: Beispiel

- TI TMS320C6400



VIIW-Prinzip: Beispiel

- TI TMS320C6000 Architekturmerkmale
  - 2 x 4 Funktionseinheiten (A und B Seite)
    - Jede Seite enthält 16 Register
      - Programmierer sieht 32 Register A0 – A15, B0 – B15
      - Ausgewählte Register für Boole'sche Ergebnisse von bedingten Befehlen
      - 9 32 Bit Lese- und 6 32 Bit Schreibports
      - Jeweils ein Crossover-Pfad: Beschränkter wechselseitiger Zugriff
    - Jede Seite enthält
      - Eine 40 Bit Integer-ALU (L Unit)
        - » Arithmetische und logische Operationen, Vergleiche, Normalisierung, Bit-Count,
      - 16 Bit Multiplizierer
        - » 16 x 16 → 32 Bit Multiplikation

VIIW-Prinzip: Beispiel

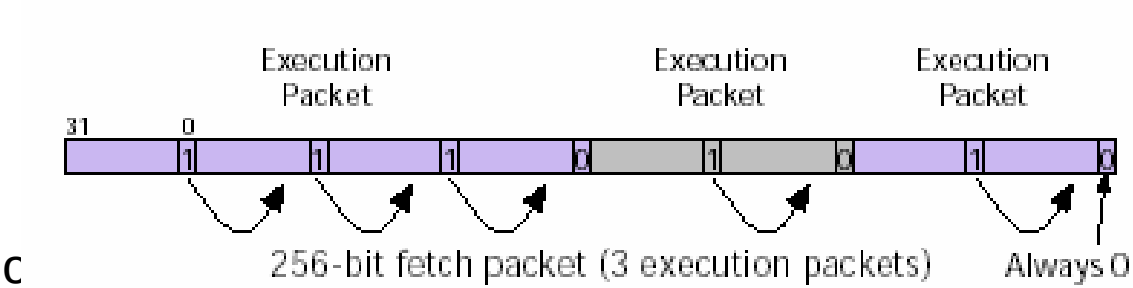
- TI TMS320C6000 Architekturmerkmale
  - 2 x 4 Funktionseinheiten (A und B Seite)
    - 40 Bit Schiebeeinheit
      - Arithmetische Operationen, Verzweigung und Verzweigungsadressgenerierung
    - 32 Bit Addierer
      - Adressgenerierung

VlIW-Prinzip: Beispiel

- TI TMS320C6000 Architekturmerkmale
  - VLIW-Prinzip
    - Holen von 8 32 Bit Befehlen über 256 Bit Befehlsbus (Fetch Packet)
      - Geholte Befehle müssen nicht unbedingt gleichzeitig ausgeführt werden
      - Ein Befehl in einem Fetch Packet ist nicht auf eine Ausführungseinheit beschränkt, jeder Befehl enthält Kodierung, mit der spezifiziert wird, auf welche Einheit der Befehl ausgeführt werden soll
      - Befehle sind nicht positionsabhängig
      - Programmierer / Compiler bestimmt Bindung

VlIW-Prinzip: Beispiel

- TI TMS320C6000 Architekturmerkmale
  - VLIW-Prinzip
    - Bis zu 8 Befehle können gleichzeitig ausgeführt werden: Gruppierung von gleichzeitig ausführbaren Befehlen (Execution Packets)
    - Werden im niedrigstwertigen Bit gekennzeichnet: alle nachfolgenden Befehle werden gleichzeitig ausgeführt
    - Prc
      ver



VLIW-Prinzip

- Literatur:
  - Hennessy/Patterson: Computer Architecture A Quantative Approach. Kap. 4.1-4.4, 4.8