Rechner-architektur

Hochschule München, Wintersemester 2010/11

Hans-Georg Eßer
hans-georg.esser@hm.edu

Herzlich willkommen!

Rechnerarchitektur

Hans-Georg Eßer

IF3A, IF3B

Web-Seite: http://hm.hgesser.de/

(Folien, MP3s, Termine, sonstige Informationen)

Termine

IF3A	IF3B
Do 10:00-11:30 Vorlesung, R0.058	Do 10:00-11:30 Vorlesung, R0.058
Do 13:30-15:00 Übung: R1.009, Vorl.: R2.007	
	Do 15:15-16:45 Übung: R1.009, Vorl.: R2.007
Do 17:00-18:30 *Übung: R1.009*	*Fr 13:30-15:00* *Übung: R1.009 (kein PC-Raum)*

Details: heute in der zweiten Stunde **in R2.007** (IF3A: 13:30 bzw. IF3B: 15:15)

Ziele (1/2)

Modulbeschreibung:

- Architekturen verstehen und hinsichtlich der Tauglichkeit für den praktischen Einsatz in verschiedenen Gebieten bewerten.
- Technische Fachartikel zur Rechnerarchitektur verstehen, um in diesem sehr dynamischen Gebiet auf dem Laufenden zu bleiben.
- Auf der Basis des Verständnisses von Rechner-architekturen aktuelle Komponenten auswählen, daraus einen für das Spielen aktueller Games geeigneten Rechner aufbauen und diesen hinsichtlich Preis-/ Leistungsverhältnis untersuchen und bewerten können.

Ziele (2/2)

Modulbeschreibung:

- Grundlegende Methoden und Wirkungsweisen von Computer Games kennen lernen und anwenden.
- Methoden-, Medien- und Sozialkompetenzen im Team trainieren

Inhalte (1/2)

Modulbeschreibung:

- Prinzipien und Methoden für
  - Analyse,
  - Implementierung,
  - Bewertung und Klassifikation

  von Rechnerarchitekturen
- Architekturprinzipien und Merkmale moderner RISC- und CISC- (Mikro-) Prozessoren wie
  - Befehlssätze
  - Superskalarität
  - Pipelining
  - Cache-Organisation

Inhalte (2/2)

Modulbeschreibung (Forts.):

- Organisationsprinzipien von Multiprozessor-Systemen und wichtige Architekturmodelle
- Gaming-PCs
  - Komponenten, Konfiguration und Zusammenbau
  - Benchmarking

Organisatorisches

- Kaffee, Wasser, Snacks etc.: Mir egal (der Hausordnung evtl. nicht...), aber bitte weitgehend geräuschlos
- Vorlesungsbeginn um 10:00 Uhr – bitte **pünktlich** kommen, ich fange pünktlich an
- Handy-Telefonate, Privatgespräche: bitte nicht (wie im Kino: wenn der Film beginnt, ist das Handy aus)
- mich unterbrechen, um eine klärende Frage zu stellen: jederzeit!
- Generell: Fairness
- **Pausen:** ja, ca. fünf Minuten in der Mitte

Kontakt / Fragen

Sprechstunde: nach Vereinbarung per E-Mail
oder unmittelbar nach der Vorlesung
(Do 11:30-13:30)

Fragen:
- direkt in der Vorlesung (Handzeichen)
- oder danach
- oder per E-Mail

Über den Dozenten

Hans-Georg Eßer

- Dipl.-Math. (RWTH Aachen, 1997)
  Dipl.-Inform. (RWTH Aachen, 2005)
  Fachjournalist (DFJS Berlin, 2006)
- Chefredakteur Computerzeitschrift (seit 2000)
- Autor diverser Computerbücher
- Doktorand (Univ. Erlangen-Nürnberg)
- Lehrbeauftragter (nicht Professor) (seit 2006)
- Lehraufträge an der Hochschule München:
  - **Rechnerarchitektur**, Betriebssysteme (Informatiker)
  - Grundlagen der Informatik (Wirtsch.-Ing.)

Grobe Gliederung

1. Grundlagen
Begriffsbestimmung, Abgrenzung
Historische Entwicklung
Chipentwurf/Technologie

2. Instruction Set Architecture (ISA)
Registerstruktur, Adressierungsarten
Maschinenbefehlssätze, Spezialbefehle
Stack-Maschinen

3. Leistungsbewertung und Leistungsmessung

Grobe Gliederung

4. Pipelining
Klassische Fünf-Stufen-Pipeline
Pipeline-Hemmnisse
Superskalarität, out of order execution
Spekulative Befehlsausführung, Sprungvorhersage

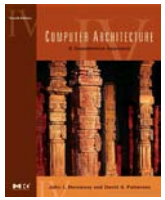
5. Speichersysteme
Von der Speicherzelle zum Modul
Caches
Adressübersetzung virtual / physical → *Betriebssysteme*

6. Sprungvorhersage

7. Mehrprozessorsysteme

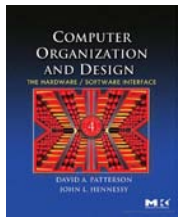
Literatur (1/2)

Rechneraufbau und Rechnerarchitektur
Axel Böttcher (Hochschule München)
ISBN: 3540209794



Computer Architecture: A Quantitative Approach
4th ed.
John L. Hennessy, David A. Patterson
ISBN: 978-0123704900

Literatur (2/2)



Computer Organization and Design, The Hardware/Software Interface
4th ed.
David A. Patterson, John L. Hennessy
ISBN: 978-0123744937

Computerarchitektur. Strukturen - Konzepte – Grundlagen
Andrew S. Tanenbaum
ISBN: 3827371511

Hinweis

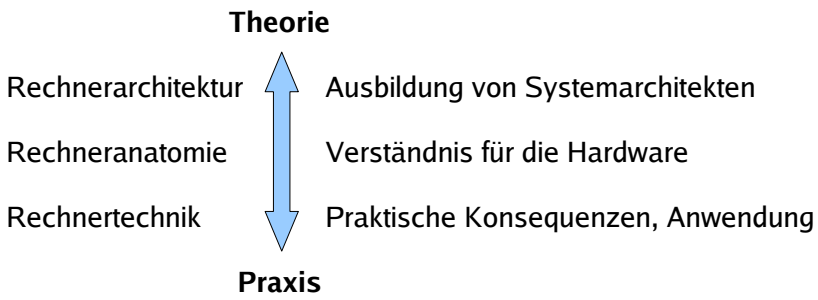
- Einen Großteil der Folien habe ich von

  Prof. Dr. Axel Böttcher

  übernommen, der diese Vorlesung zuletzt gehalten hat und mir sein Material freundlicherweise zur Verfügung gestellt hat.

1. Grundlagen

Begriffsbestimmung (1/2)



Rechnerarchitektur umfasst
- die Analyse
- den Entwurf
- die Bewertung
- die Synthese

von Rechnern und Rechnerkomponenten.

Begriffsbestimmung (2/2)

Dazu müssen
- strukturelle
- organisatorische
- implementierungstechnische

Aspekte berücksichtigt und auf der
- globalen Systemebene
- Maschinenbefehlssatzebene
- Mikroarchitekturebene

untersucht werden.

Zwischen den beteiligten Ebenen und den verschiedenen Teilaspekten der Rechnerarchitektur sind Rückkopplungen möglich

Auf allen Ebenen umfangreiche Wechselwirkungen mit anderen Disziplinen der
- Informatik,
- Ingenieur- und Naturwissenschaften
- Mathematik

Betrachtungsebenen (1/3)

Verschiedene Betrachtungsebenen

Globale Systemebene (Prozessoren, Busse, Speicher)
- *Für:* System-Architekt für Chip/Motherboard, Vertrieb
- *Sicht auf:* ganzes System
- *Elemente:* Welche Hauptelemente besitzt das System, und wie sind diese miteinander verbunden.

Maschinenbefehlssatzebene
- *Für:* Compilerbauer, Assembler-Programmierer, Vertrieb
- *Sicht auf:* Prozessor-Funktionen
- *Elemente:* Satz von Befehlen (Maschineninstruktionen), den der Prozessor beherrscht.

Betrachtungsebenen (2/3)

Verschiedene Betrachtungsebenen

Register-Transfer-Ebene (Mikroprogramm-Ebene)
- *Für:* CPU-Entwickler, Programmierer, Mikroprogrammierer (Informatik-Grundwissen)
- *Sicht auf:* Infrastruktur auf dem Chip
- *Elemente:* Register, wie diese mit ihrer Umgebung verbunden sind und welche Verknüpfungen vorgesehen sind, um den Befehlssatz zu realisieren.

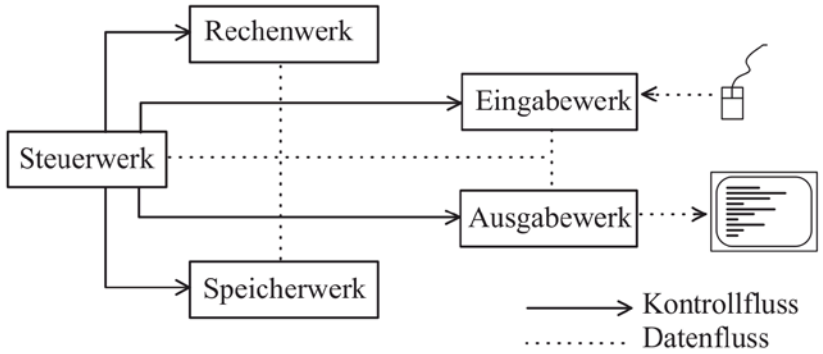
Betrachtungsebenen (3/3)

Teile der Mikroarchitekturebene

- Logikebene
  - *Für:* Chip-Designer
  - *Sicht auf:* Schaltung
  - *Elemente:* Gatter und Flipflops und wie diese verbunden sind.
- Hardware-Realisierungs-Ebene
  - *Für:* Designer
  - *Sicht auf:* Chiplayout
  - *Elemente:* Bauteile (z.B. Transistoren, Kondensatoren etc.) und wie diese verbunden sind, um Gatter oder Flipflops zu realisieren.

Universalrechner (1/4)

- Bestandteile eines Universalrechners (Computer) nach John v. Neumann



Universalrechner (2/4)

- 5 Funktionseinheiten: Steuerwerk, Rechenwerk, Speicherwerk, Eingabewerk und Ausgabewerk
- Aufbau unabhängig vom zu bearbeitenden Problem
- Programme und Daten, Zwischenergebnisse und Endergebnisse im gleichen Speicher
- Eingabewerk liest Daten und Programme ein und legt sie im Speicherwerk ab.
- Speicher in gleich große Zellen aufgeteilt (fortlaufend nummeriert)
- Aufeinander folgende Befehle befinden sich in aufeinander folgenden Speicherzellen.
- Im Steuerwerk gibt es einen **Befehlszähler**, der immer auf den nächsten auszuführenden Befehl zeigt.

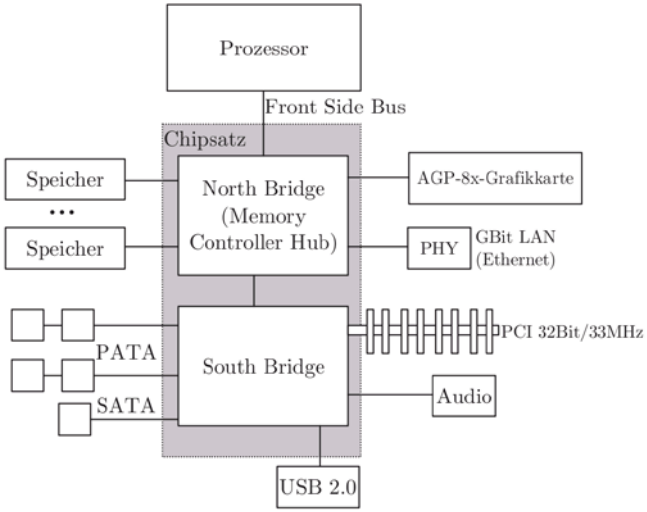
Universalrechner (3/4)

- Der aktuelle Befehl wird aus dem Speicher ausgelesen und im **Befehlsregister** des Steuerwerks zwischengespeichert.
- Der Befehl wird dann dekodiert, und die Ausführung des Befehls durch Steuersignale veranlasst.

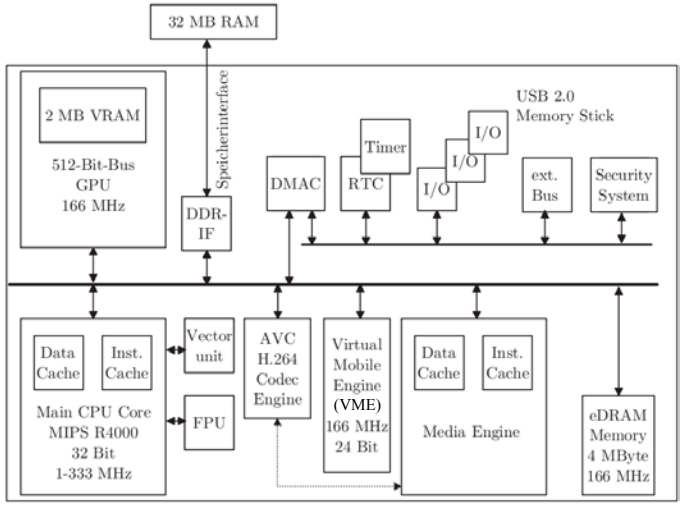
Universalrechner (4/4)

- Es gibt folgende Befehlsarten:
  - Arithmetische und logische Befehle zur Verknüpfung von Daten
  - Transportbefehle zum Verschieben von Daten zwischen diesen Komponenten
  - Bedingte und unbedingte Sprungbefehle, Unterprogrammaufrufe
  - Ein-/Ausgabebefehle zur Kommunikation mit der Peripherie
  - Sonstige Befehle wie Unterbrechen, Warten, Stop etc.

Beispiel: Aufbau eines PCs



Beispiel: Aufbau der PSP



Hier höhere Integrationsdichte (→ embedded systems)

Komplexität von Schaltungen

Chip		Anzahl Transistoren
Flipflop	Speichert ein Bit	6
Gatter (und/oder)	Verknüpft zwei binäre Werte	4
Addierer	Addiert 64-Bit-breite Worte	> 400
Datenpfad	Komplettes Rechenwerk mit Puffern	> 200.000
Pentium II	Ganzer Prozessor	4,5 Millionen
Pentium 4	Ganzer Prozessor	42 Millionen
Xeon-Dunnington	6-Kern-Prozessor	1,9 Milliarden

Landschaft der Prozessoren I (Wichtige Begriffe)

- **Hardwarebeschreibungssprachen** sind Programmiersprachen zur Beschreibung von Hardware. Können verwendet werden, um Hardware zu generieren. Wichtigste Vertreter: VHDL (Very High Speed Integrated Circuit Hardware Description Language) und Verilog. (siehe HP-Forschungsprogramm PICO: „Program In – Chip Out“)
- **Synthetisierbarer Kern** ist ein Prozessorkern, der in einer Hardwarebeschreibungssprache vorliegt (**Softcore**) und von Lizenznehmern in eigene Entwürfe eingebunden werden kann; z. B.: MIPS32 24K, ARM1136J, Intel XScale 80200T.

Landschaft der Prozessoren I (Wichtige Begriffe)

- **System on a Chip (SoC)** enthält auf einem einzigen Chip neben dem Prozessorkern auch Speicher, Schnittstellen, Timer und ggf. auch Grafik-, DMA- und Interrupt-Controller sowie PCMCIA, Touch-Panel-Interface etc. (z.B. PowerPC 405LP). Daher kein Chipsatz erforderlich
- **Embedded Systeme** (ältere Bezeichnung: Prozessrechnersysteme) sind kleine Computer, die in bestimmten Produkten eingesetzt werden (also **eingebettet** sind). Sie übernehmen dort Steuerungs-, Kontroll- oder Bedienaufgaben.
  Einsatzgebiete: Haushaltsgeräte, Unterhaltungselektronik, Fahrzeuge, ...

Landschaft der Prozessoren II

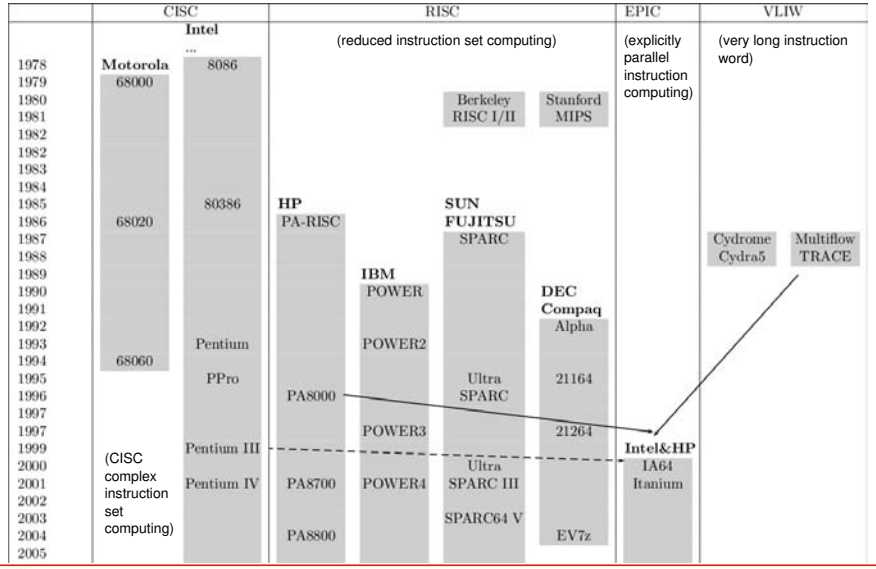
Prozessortyp	Einsatzgebiet	Bemerkung
Intel Pentium	Desktop	Mainstream
Intel Itanium	Server/Desktop	64-Bit
AMD Athlon	Desktop	x86-Befehlssatz
AMD Opteron	Server/Desktop	64-Bit
IBM PowerPC	MAC (bis 2006)	
HP/DEC	Alpha	Desktop, Supercomputer
SUN SPARC	Workstations	
MIPS	Embedded (Chipkarten)	
ARM (Advanced RISC Machine)	PDAs, Gameboy	Erster RISC (von Acorn) in einem Desktop (1987)

Landschaft der Prozessoren II

Prozessortyp	Einsatzgebiet	Bemerkung
Transmeta Crusoe	Mobile (Notebooks, PDAs, Thin Clients)	Code Morphing
Atmel	div. Controller	
Siemens	TriCore	Embedded Controller; synthetisierbar
TI TMS320C64xx	Signalprozessor	Video(de-)kodierung
aj-100	Ajile Systems	Java Prozessor

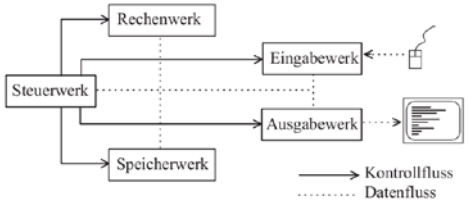
Daneben: Landschaft der Supercomputer, die meist aus vielen der genannten Prozessoren zusammen gesetzt sind. International Supercomputer Conference: halbjährlich Liste der Top 500

Landschaft der Prozessoren III

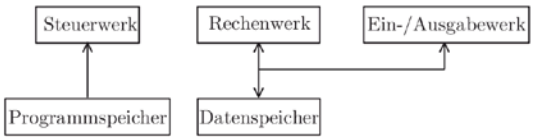


von Neumann vs. Harvard

- Von-Neumann-Architektur



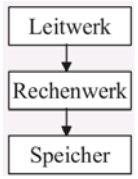
- Harvard-Architektur



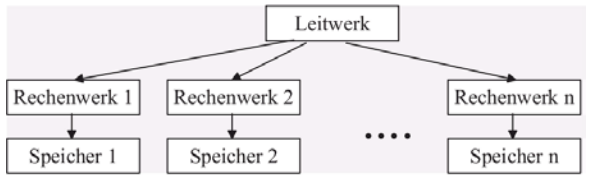
Getrennte Speicher fur Programm und Daten. Anzutreffen bei manchen Signalprozessoren. Als modifizierte Harvard-Architektur in praktisch allen modernen Prozessoren mit getrennten Level-1-Caches fur Code und Daten verwendet

Klassifikation nach Flynn: SISD, SIMD

- Flynn (1966). Heute nicht mehr ganz tragfähig, aber die Begriffe werden noch verwendet.
- Single Instruction, Single Data (SISD):

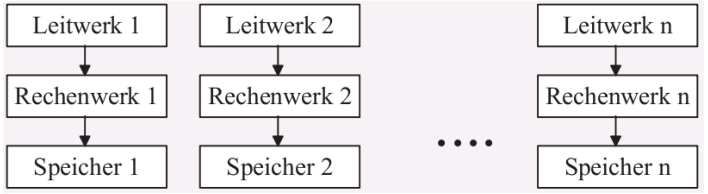


- Single Instruction, Multiple Data (SIMD):

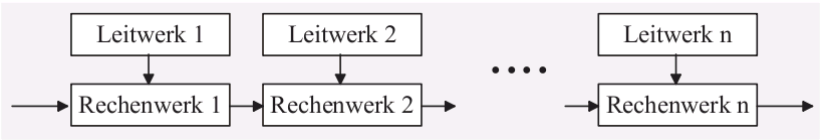


Klassifikation nach Flynn: MIMD, MISD

- Multiple Instruction, Multiple Data (MIMD):



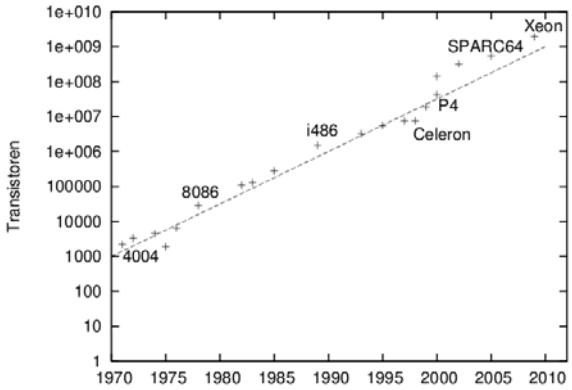
- Multiple Instruction, Single Data (MISD):



Technologie

Moore's Law

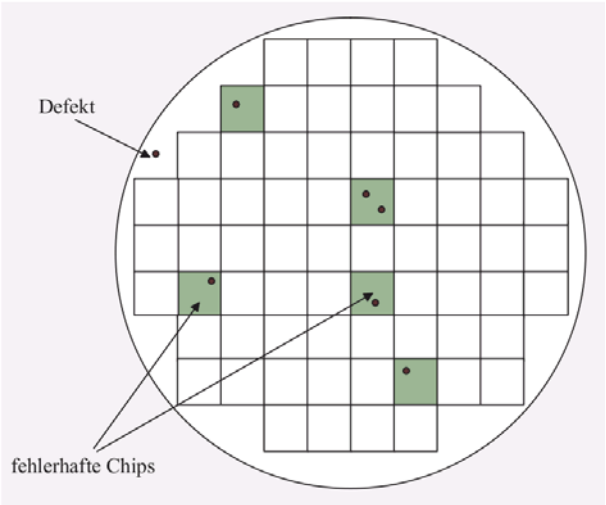
- Gordon Moore: „Die Anzahl der Transistoren pro Chip verdoppelt sich alle 18-24 Monate.“



Phasen der Chip-Entwicklung

Phase	Produkt	Prüfmethode
Spezifikation	Text	
Implementierung (VHDL/Verilog)	SW-Modell	Model-Checking, Simulation, Emulation, Equivalence-Checking
Synthese	Gatternetzliste	statische Timing-Analyse
Platzierung und Routing	Layout	Design rule check, Layout vs. Schematic
Fertigung	Chip	Test

Wafer (Die)



2. ISA: Instruction Set Architecture (Befehlssatzarchitektur)

Befehlssatzarchitektur (Instruction Set Architecture – ISA)

Beschreibung umfasst:

- Maschinenbefehlssatz
- Registerstruktur
- Adressierungsarten
- Interruptbehandlung

Klassisch: Unterscheidung in
Ein-, Zwei- und Drei-Adress*maschinen*

Heute üblicher: unterscheiden nach
Ein-, Zwei- und Drei-Adress*befehlen*

Register und Registersätze (1/2)

- Register: die schnellsten speichernden Elemente eines Prozessors
- meist allgemein verwendbare Register (General Purpose Registers, GPR) und Spezialregister.
- Gesamtheit aus Befehlssatz und verfügbaren Registern heißt **Programmiermodell**

Register und Registersätze (2/2)

Typische Spezialregister

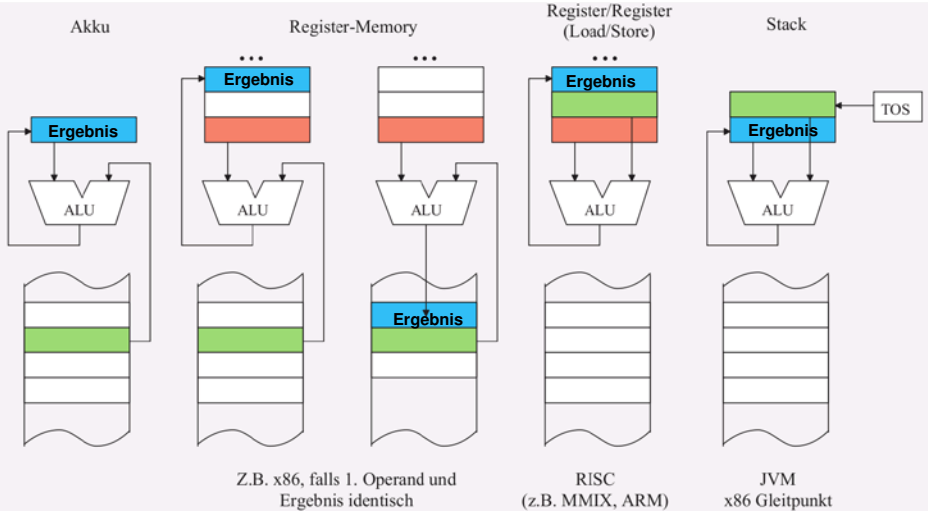
- Befehlszähler
- Stackpointer
- Statusregister (kann z. B. anzeigen, ob bei der letzten Operation ein Überlauf aufgetreten ist, oder ob das Ergebnis negativ war etc.)
- Indexregister (für Adressrechnungen)

Operanden und Ergebnis (1/6)

- ISAs unterscheiden nach Zugriff auf Register und Speicherinhalte
- Aufgabe: Werte aus zwei Speicherzellen addieren und in dritter Zelle speichern

  C := A + B;

Operanden und Ergebnis (2/6)

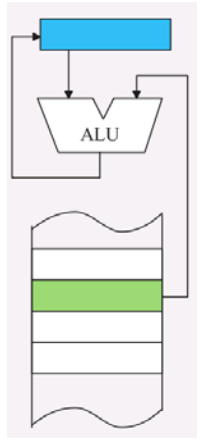


Operanden und Ergebnis (3/6)

C := A + B

mit **Akku**:

```
LOAD  A
ADD   B
STORE C
```



nutzt implizit den Akku:

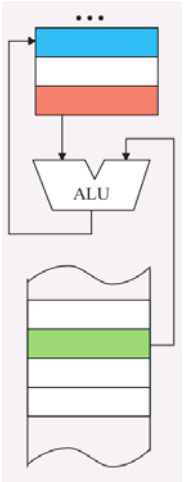
- LOAD A = lade A in Akku
- ADD B = addiere B zum Wert in Akku (Ergebnis im Akku)
- STORE C = schreibe Akku-Inhalt nach C

Operanden und Ergebnis (4/6)

C := A + B

mit **Register-Memory**:

```
LOAD  R1,A
ADD   R3,R1,B
STORE R3,C
```



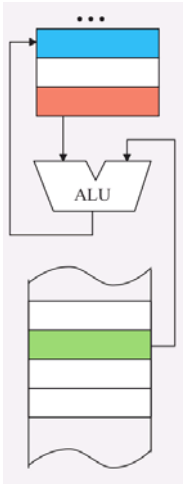
nennt alle Register explizit (R1, R3)
Argumente können Register oder Speicherstellen sein (z. B. R1, B)

Operanden und Ergebnis (5/6)

C := A + B

mit **Register-Register (Load/Store)**:

```
LOAD   R1,A
LOAD   R2,B
ADD    R3,R1,R2
STORE  R3,C
```



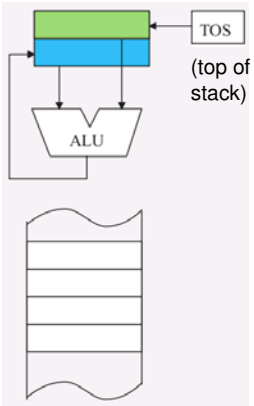
- nennt alle Register explizit (R1, R3)
- Argumente für Operationen können nur Register sein (R1, R2)
- Zugriff auf Speicher nur durch LOAD, STORE

Operanden und Ergebnis (6/6)

C := A + B

mit **Stack**:

```
PUSH A
PUSH B
ADD
POP  C
```



- keine Register
- Argumente auf Stack, dann Operation
- Operation implizit (oberste Werte auf Stack)
- Zugriff auf Speicher nur durch PUSH, POP

Vorschau 14.10.2010

- Ein-, Zwei-, Drei-Adress-Befehle
- Stack-Maschinen, Java
- Befehlsarten (Gleitkomma, Sprünge, atomare Operationen)