IT Infrastrukturen – Rechnerstrukturen

Einführungsvorlesung

Prof. Dr.-Ing. Sebastian Schlesinger
Professur für Wirtschaftsinformatik
(Security and Embedded Systems Engineering)

Dieser Foliensatz basiert auf den Folien zu „Rechnerorganisation“ von Prof. Dr. Ben Juurlink (TU Berlin) und Prof. Paula Herber, WWU Münster

Überblick

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law

- Organisatorisches
- Lernziele und Themen-Überblick
- Einführung

Überblick

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law

- Organisatorisches
- Lernziele und Themen-Überblick
- Einführung

Ansprechpartner

Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

Vorlesung

- Prof. Dr. Sebastian Schlesinger
  Raum 5.2001
  sebastian.schlesinger@hwr-berlin.de
  MS Teams

Ablauf der Veranstaltung

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law

- Vorlesungen und Übungen finden grundsätzlich hybrid statt
- kurzfristige Änderungen werden über Moodle Announcements bekannt gegeben

Prüfung

- Zwei Moodle-Tests während der Vorlesung
- Ergibt 34 Punkte (ganzes Modul Infrastrukturen ergibt 100 Punkte)

Organisation

Fragen?

Überblick

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law

- Organisatorisches
- Lernziele und Themen-Überblick
- Einführung

Lernziele

- Zahlendarstellungen im Rechner und Mikroalgorithmen zur Umsetzung von arithmetischen Operationen verstehen und anwenden können.
- Den Aufbau digitaler Systeme und den Entwurfsprozess von Hardware zu verstehen und anhand von Beispielen nachvollziehen zu können.
- Die Übersetzung eines Programmes von einer höheren Programmiersprache (z. B. C) in eine Maschinensprache und deren Ausführung von einem digitalen System verstehen.
- Assemblerprogramme nachvollziehen und selbst schreiben können.
- Die logischen Abläufe zur Bearbeitung von Maschinenbefehlen in einem Prozessor auf Register-Transfer-Ebene verstehen.

Themen

1. Rechnerarithmetik (Zahlendarstellungen, arithmetische Operationen)
2. Grundlagen der Digitaltechnik (Gatter, Flipflops, Register)
3. Maschinenbefehle (Assemblersprache, Kontrollfluss, Adressierungsarten)
4. Aufbau und Funktionsweise eines einfachen Prozessors
5. Fließbandverarbeitung (Pipelining, Pipelinekonflikte und ihre Lösungen) (optional)
6. Ggf. Ausblick Weiterentwicklung Rechnerarchitekturen und SoC Designs / Architekturen für spezielle Use Cases (z.B. FPGAs, GPUs)

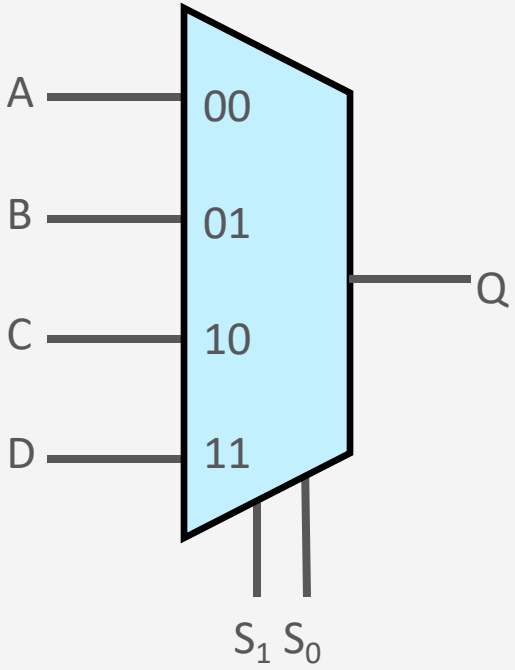
Thema 1: Rechnerarithmetik

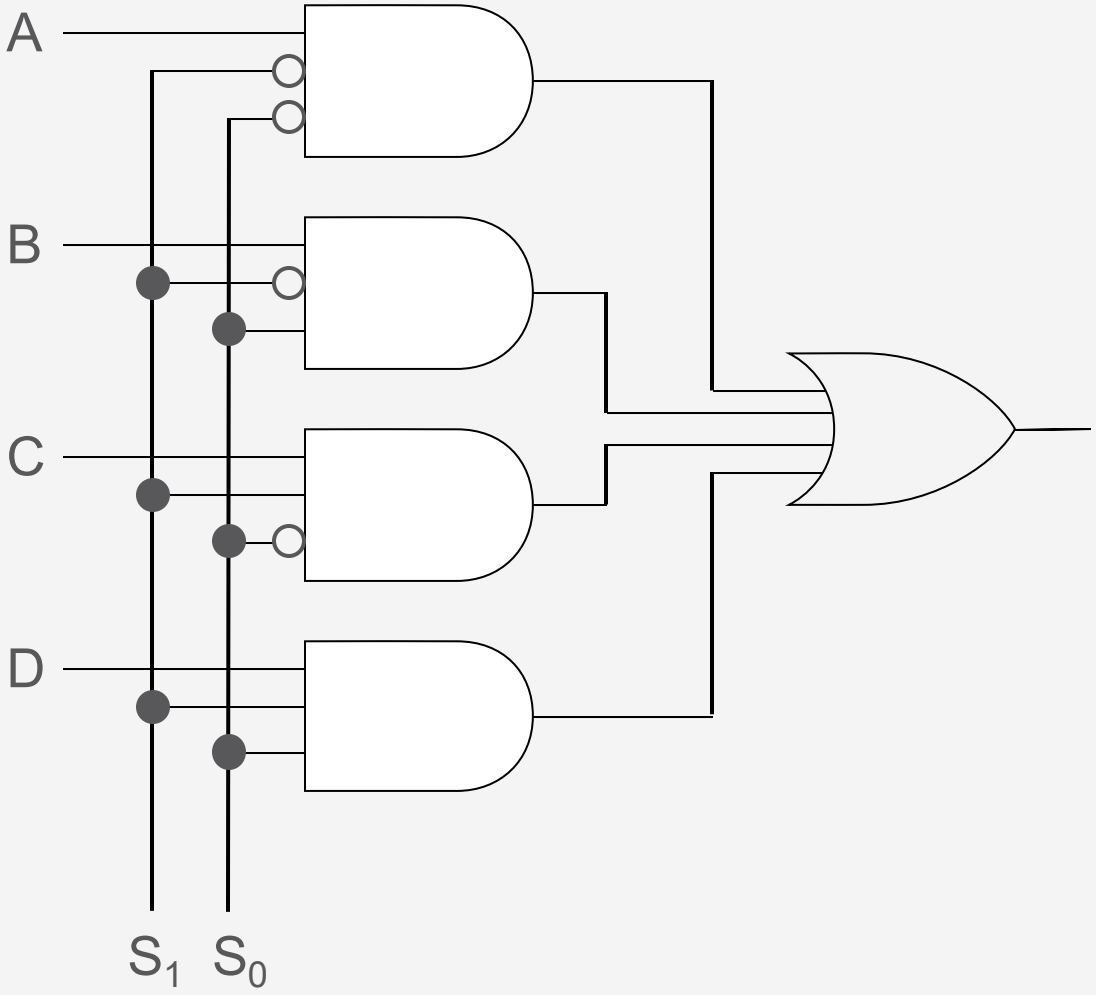
$$2435 = 2 \cdot 10^3 + 4 \cdot 10^2 + 3 \cdot 10^1 + 5 \cdot 10^0$$

$$1011_B = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = (8 + 0 + 2 + 1)_D = 11_D$$

$$b_{31}b_{30}\ldots b_1 b_0 = -b_{31}2^{31} + b_{30}2^{30} + \ldots + b_1 2^1 + b_0 2^0$$

Thema 2: Digitaltechnik





Thema 3: Assembler

High-level language program (in C)

```
void swap(int v[], int k)
{
    int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

C compiler

Assembly language program (for MIPS)

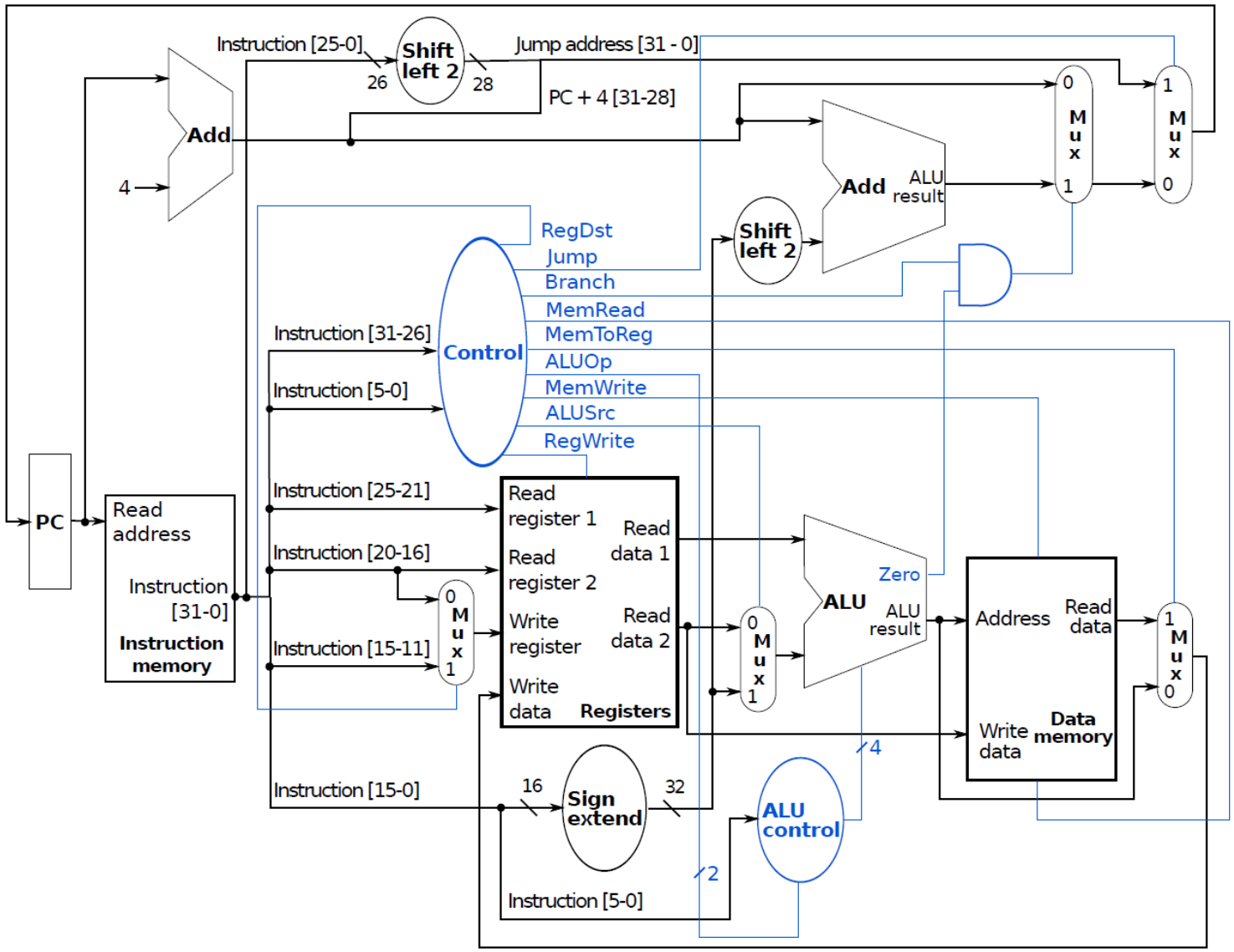
```
swap:
  muli $2,$5,4
  add  $2,$4,$2
  lw   $15,0($2)
  lw   $16,4($2)
  sw   $16,0($2)
  sw   $15,4($2)
  jr   $31
```

assembler

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

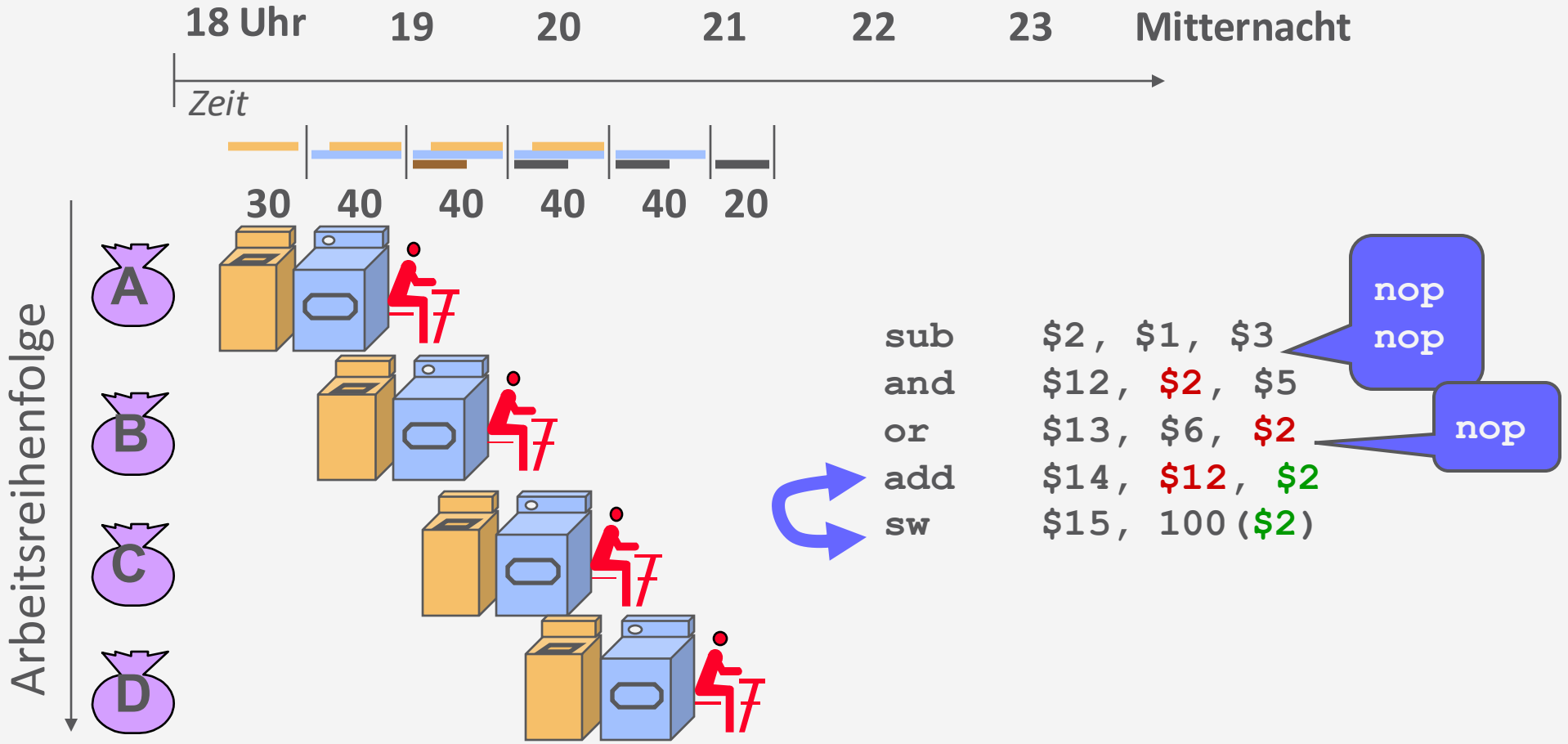
Binary machine language program (for MIPS)

Thema 4: Prozessorentwurf



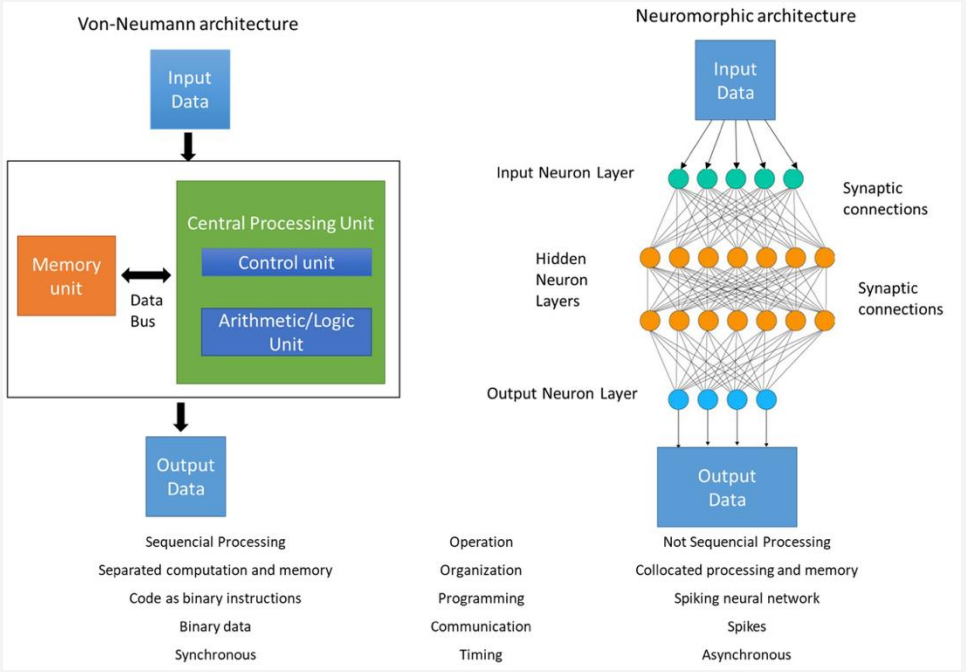
Thema 5: Pipelining - optional

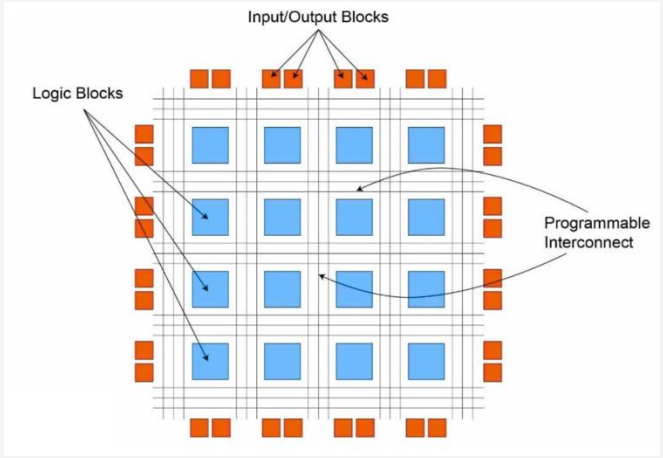
Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

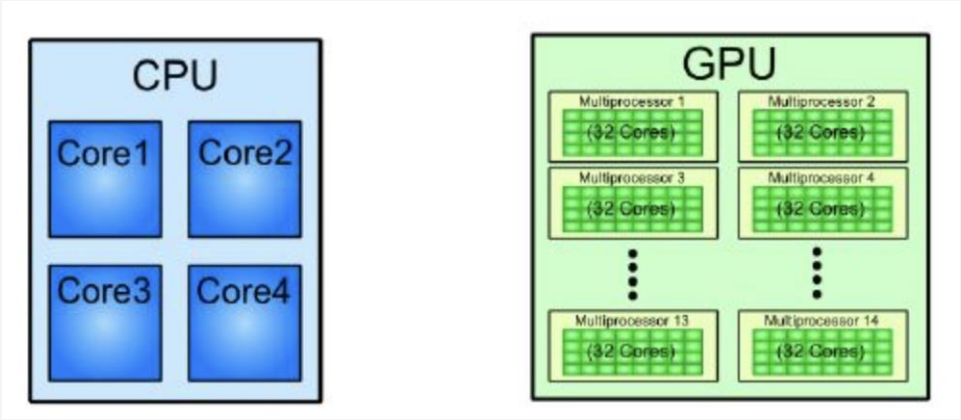


Thema 6: Weiterentwicklung von Architekturen und spezielle SoC

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law







Literatur

Rechnerorganisation und -entwurf
Die Hardware/ Software-Schnittstelle
David A. Patterson and John L. Hennessy
Übersetzt von Arndt Bode, Wolfgang Karl und Theo Ungerer

Wir verwenden die MIPS Edition (3. bis 5. Auflage)
nicht die RISC V Edition

Literaturübersicht (Kapitel 1 – 3)

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law

Themen der Vorlesung	**Kapitel in Patterson & Hennessy: Rechnerorganisation und Rechnerentwurf. 5. Auflage, De Gruyter, 2016.**
Thema 1: Zahlendarstellung und Rechnerarithmetik	Kapitel 2.4: Vorzeichenbehaftete und nicht vorzeichenbehaftete Zahlen Kapitel 3: Rechnerarithmetik
Thema 2: Digitaltechnik	Anhang B: The Basics of Logic Design in Patterson & Henessy nicht erschöpfend behandelt → ergänzend: Chapter 3 in Null, Linda und Julia Lobur: Essentials of Computer Organization and Architecture. Jones & Bartlett Learning, Burlington, MA, 4. Auflage, 2015. http://ccftp.scu.edu.cn:8090/Download/545716da-b3aa-4aae-840e-3a5b9646c6dc.pdf
Thema 3: Befehle, die Sprache des Rechners	Kapitel 2: Befehle, die Sprache des Rechners Anhang A: Assemblers, Linkers, and the SPIM Simulator

Literaturübersicht (Kapitel 4 – 6)

Hochschule für Wirtschaft und Recht Berlin
Berlin School of Economics and Law

Themen der Vorlesung	**Kapitel in Patterson & Hennessy: Rechnerorganisation und Rechnerentwurf. 5. Auflage, De Gruyter, 2016.**
Thema 4: Eintaktprozessor	Kapitel 4: Der Prozessor (Kapitel 4.1 – 4.4)
Thema 5: Pipelining	Kapitel 4: Der Prozessor (Kapitel 4.5 – 4.11)

Überblick

- Organisatorisches
- Lernziele und Themen-Überblick
- Einführung
  - Die Computerrevolution
  - Klassen von Computersystemen
  - Hardware- und Software-Ebenen
  - Von einer Hochsprache zur Sprache der Hardware
  - Klassische Computerkomponenten
  - Abstraktion

Die Computerrevolution

- Computer haben die Welt verändert.
- Vor 50 - 60 Jahren waren folgende Anwendungen Science Fiction:
  - Laptops und Handys
  - World Wide Web (WWW, Web)
  - Digitales Fernsehen/Kamera
  - moderne Navigationstechnologie und autonome Fahrzeuge
  - Forschung am menschlichen Genom
  - Viren-Simulation
  - …
- Diese und andere Anwendungen sind nur möglich Dank der Entwicklung der Rechnertechnologie

Klassen von Computersystemen

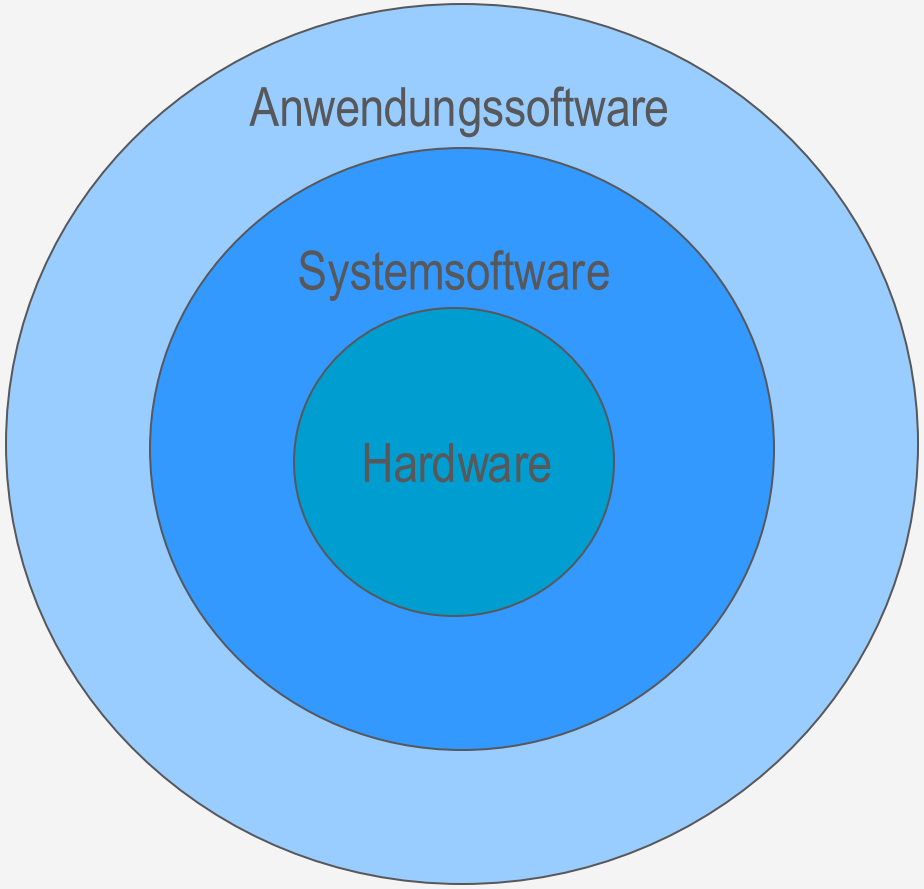
- Arbeitsplatzrechner (*desktop computers*) (100 M/Jahr)
  - der auf/unter/neben dem Schreibtisch
  - gute Leistungen zu akzeptablen Preisen
  - führen Software von Drittanbietern aus
- Server (5 M/Jahr)
  - bewältigen große Lasten (Anwendungen aus dem technisch/wissenschaftlichen Bereich, Verarbeitung vieler kleiner Jobs (Web Server))
  - gleiche Technologie wie Arbeitsplatzrechner, jedoch höheres Maß an Erweiterbarkeit
- Eingebettete Rechner (*embedded computers*) (1000 M/Jahr)
  - größte Klasse und größte Bandbreite an Anwendungen und Leistungen
  - in Waschmaschinen und Autos, Flugzeugen, Satelliten, Fahrkartenautomaten, Fabriken, …

Hinter einem Programm

- Vereinfachte Darstellung der Hardware (HW) und Software (SW) als hierarchische Ebenen in Form von konzentrischen Kreisen

- Komplexe Anwendungen bestehen häufig aus mehreren SW-Ebenen

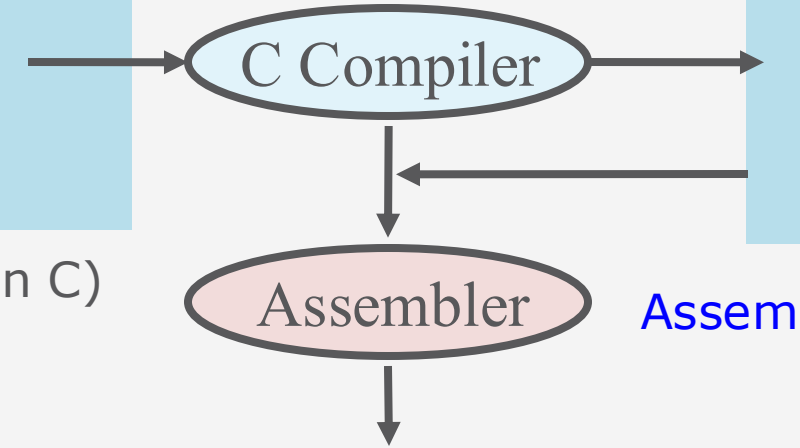
- Systemsoftware: SW, die die Verbindung zur Hardware herstellt
  (z. B. Betriebssysteme,
  Compiler und Assembler)



Von Hochsprache zu Maschinensprache


```
void swap(int v[], int k)
{
    int temp;
    temp = v[k];
    v[k] = v[k+1];
    v[k+1] = temp;
}
```

Hochsprachenprogramm (in C)




```
swap:
  muli $2,$5,4
  add  $2,$4,$2
  lw   $15,0($2)
  lw   $16,4($2)
  sw   $16,0($2)
  sw   $15,4($2)
  jr   $31
```

Assemblerprogramm für MIPS

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000…
```

Binärer Maschinencode für MIPS

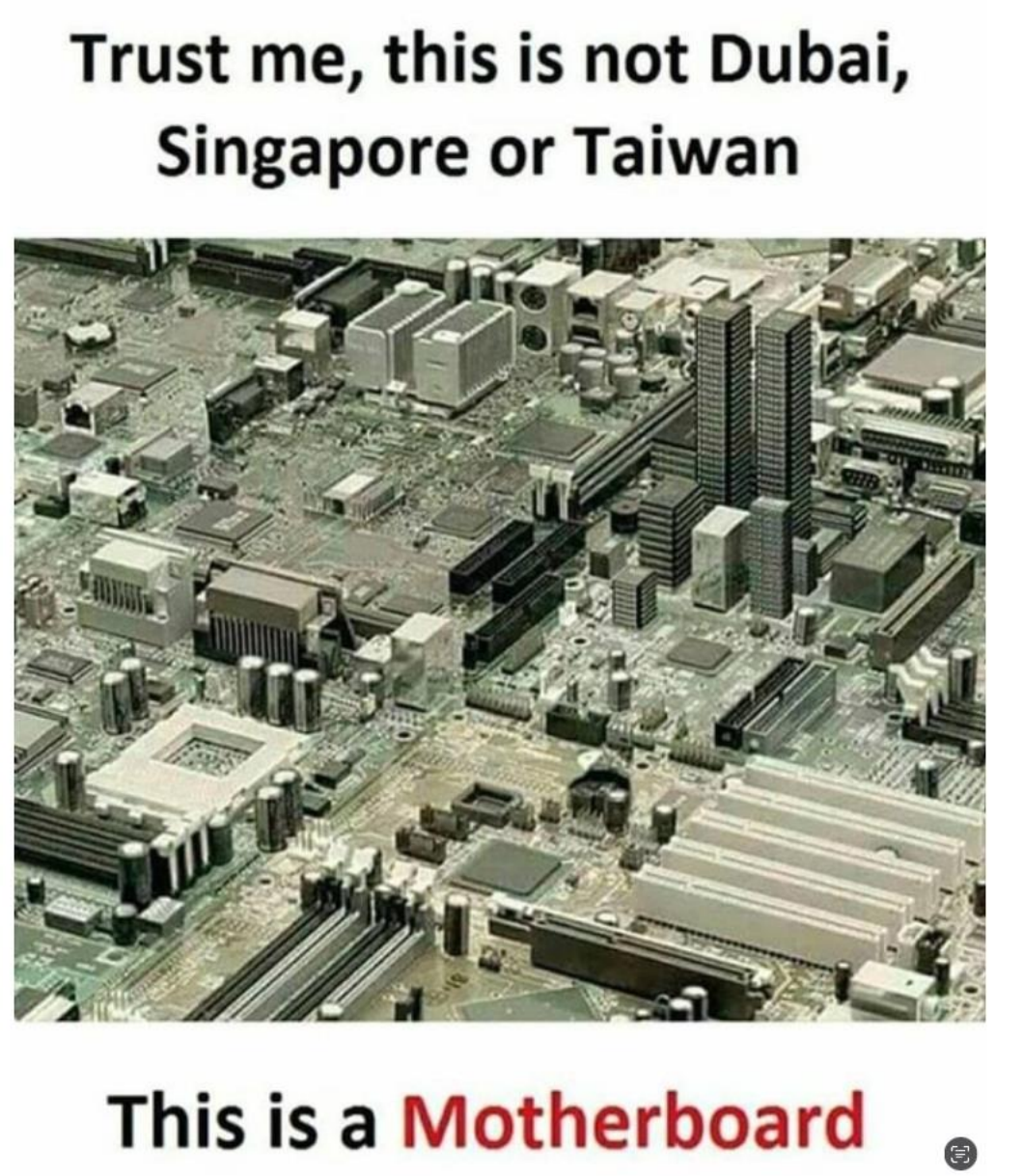
Computerkomponenten

Die 5 klassischen Komponenten eines Computers

- Eingabegeräte (Maus, Tastatur, …)
- Ausgabegeräte (Bildschirm, Drucker, …)
- Speicher
  - Intern: DRAM, SRAM [flüchtig]
  - Extern: Festplatte, CD [nicht flüchtig]
- Datenpfad
  - führt Operationen aus
  - die Muskeln eines Prozessors
- Leitwerk / Steuerung
  - sendet Signale, welche die Operationen bestimmen
  - Gehirn eines Prozessors

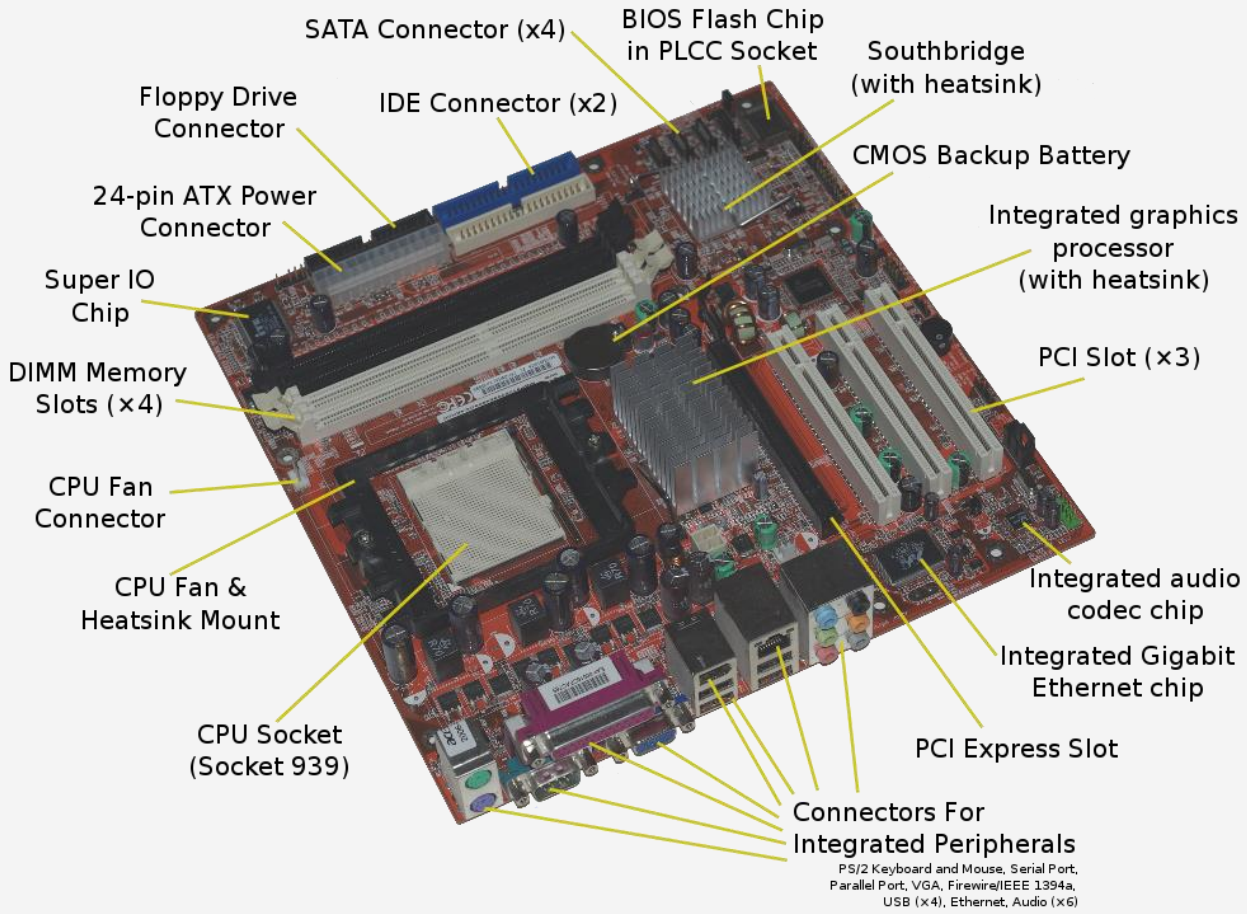
} Prozessor oder *Central Processing Unit* (CPU)

Inneres eines PC



Inneres eines PCs

Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law

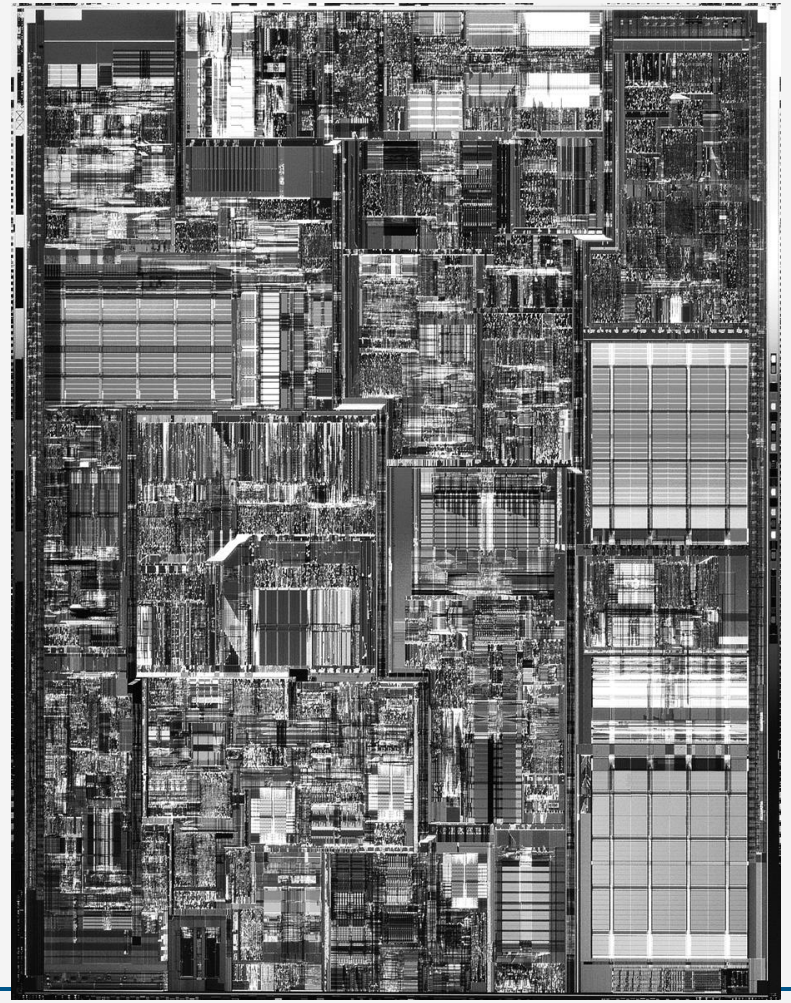


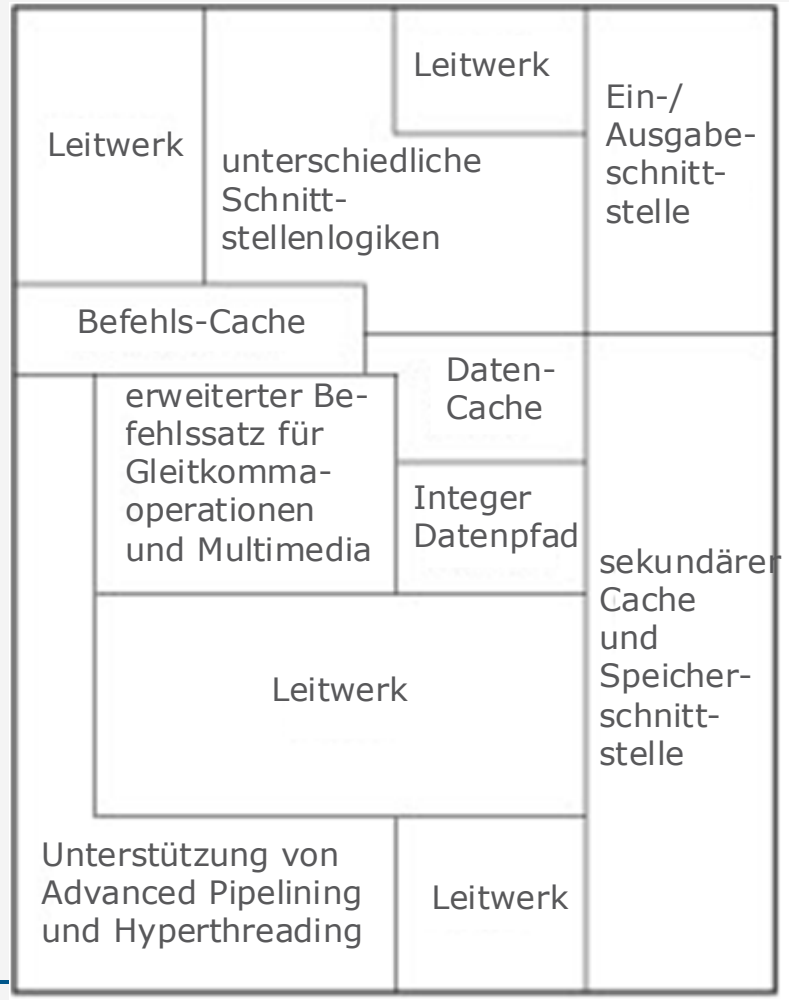
Hauptplatine (motherboard)

- DIMM = Dual In-line Memory Module
- PCI = Peripheral Component Interconnect
- IDE = Integrated Drive Electronics (bus for hard disk drives)
- SATA = Serial Advanced Technology Attachment
- CPU = Cenral Processing Unit

Inneres eines Prozessorchips (Pentium 4)

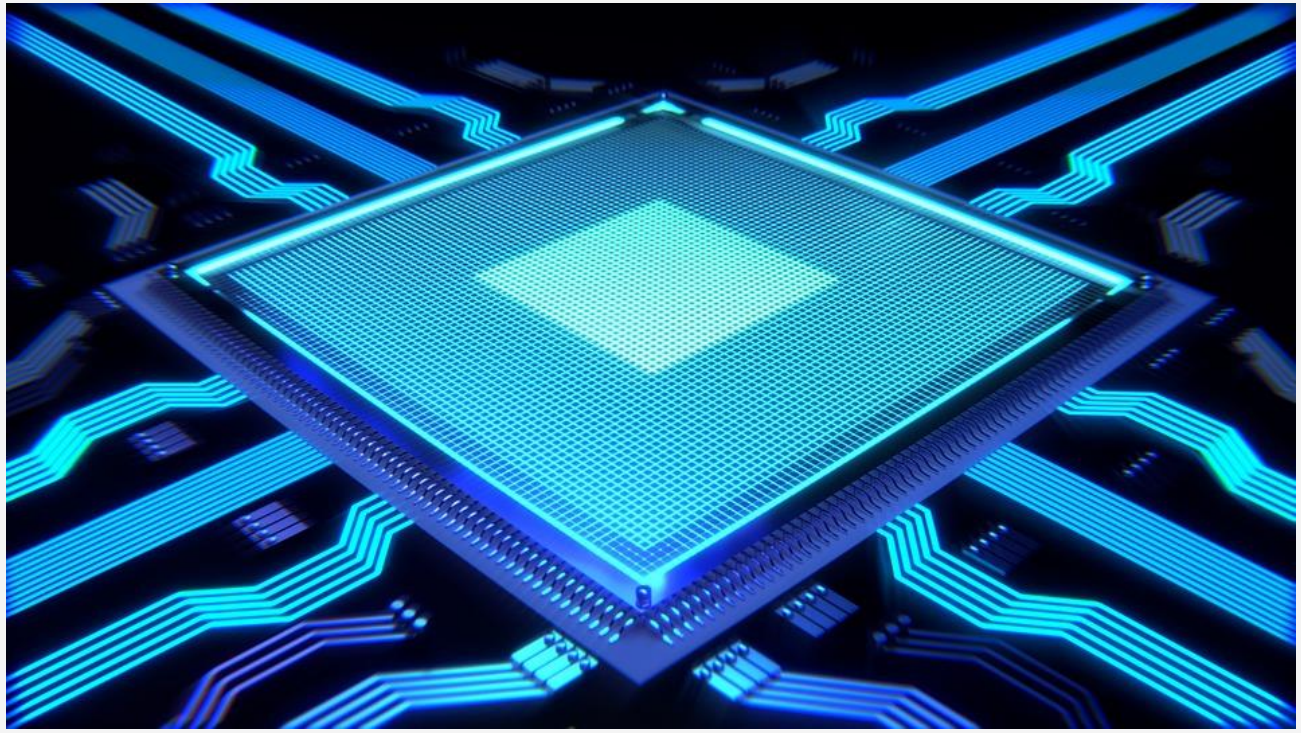
Hochschule für
Wirtschaft und Recht Berlin
Berlin School of Economics and Law





Fokus

- Unser Fokus: Der Prozessor (CPU = *Central Processing Unit*)
  - Datenpfad
  - Steuerwerk
- Milliarden von Transistoren
- Unmöglich zu verstehen, wenn man die Transistoren einzeln betrachtet
- Wir benötigen Abstraktionen auf viele Ebenen.



Abstraktion

- Abstraktion: Details der unteren Ebenen eines (Computer-) Systems werden vorübergehend ausgeblendet, um die Entwicklung komplexer Systeme zu erleichtern.
- Beispiele:
  - Schaltkreise (Logische Gatter AND, OR, NOT statt einzelner Transistoren)
  - Befehlssatzarchitektur (digitaler Rechner = Satz von Befehlen, den er ausführen kann)
  - Programmabstraktion (Funktionen, Klassen, Objekte)
  - Datenabstraktion (Listen, Warteschlangen)

Befehlssatzarchitektur (*Instruction Set Architecture*)

- Eine sehr wichtige Abstraktion!
- Vorteile
  - ✓ einfach verständliche Schnittstelle zwischen HW und SW
  - ✓ Standardisierung von Befehlen, Bitfolgen, u.s.w.
  - ✓ Programmierer / Compiler muss nicht die darunter liegende Digitaltechnik kennen
  - ✓ verschiedene Implementierungen eines Befehlssatzes möglich
- Nachteil: verhindert manchmal Innovationen

Zusammenfassung

- Klassen von Computersystemen: *Arbeitsplatzrechner*, *Server*, *eingebettete Rechner*
  - eingebettete Rechner höchste Anzahl, Arbeitsplatzrechner größter Umsatz
- klassische Komponenten eines Computers:
  - Eingabegeräte (*input devices*), Ausgabegeräte (*output devices*), Speicher, Datenpfad, Leitwerk/Steuerung (*control*)

- Unser Fokus ist der Prozessor (CPU = *Central Processing Unit,* umfasst Datenpfad und Leitwerk/Steuerung)
- Wir brauchen *Abstraktion* um den Prozessor verstehen zu können!