Lernzettel

Betriebssysteme und Rechnerarchitektur: Von-Neumann, Prozessor, Arbeitsspeicher, Prozesse, Scheduling

Universität:	Technische Universität Berlin
Kurs/Modul:	Einführung in die Informatik
Erstellungsdatum:	September 19, 2025

Lernzettel: Betriebssysteme und Rechnerarchitektur
(Von-Neumann, Prozessor, Arbeitsspeicher, Prozesse, Scheduling)

(1) Von-Neumann-Architektur.
Die Grundidee ist, dass Programme und Daten im selben Speicher abgelegt sind und von der CPU über Busse eingelesen werden. Typische Bestandteile:
- CPU (Rechenwerk, Steuerwerk, Registersatz)
- Speicher (RAM) als Primärspeicher
- Ein-/Ausgabe-Systeme (I/O)
- Busse (Adress-, Daten-, Steuerbus)

$$\text{CPU} \overset{\text{Busse}}{\longleftrightarrow} \text{Speicher} \overset{\text{I/O}}{\longleftrightarrow} \text{Peripherie}$$

Fetch-Decode-Execute-Zyklus.

$$\text{Fetch: } \text{IR} \leftarrow \text{Mem}[\text{PC}]; \quad \text{PC} \leftarrow \text{PC} + 1$$

$$\text{Decode: } \text{opcode}, \text{operands} \leftarrow \text{IR}$$

$$\text{Execute: } \text{perform}(\text{opcode}, \text{operands})$$

Bottleneck.

$$\text{Bottleneck} \approx \frac{\text{Speicherbandbreite}}{\text{CPU-Takt}} \quad \text{(vereinfacht)}$$

(2) Prozessor (CPU).
- Rechenwerk (ALU) führt arithmetische und logische Operationen aus.
- Steuerwerk (Control Unit) interpretiert Befehle und koordiniert Datenfluss.
- Registersatz: PC (Program Counter), IR (Instruction Register), allgemeine Register R0, R1, ... , FLAGS.
- Befehlssatz (ISA) definiert, welche Befehle existieren und wie sie ausgeführt werden.
- Execute-Einheit führt Operationen aus, ggf. mit Pipelines zur Überlappung von Fetch/Decode/Execute.

$$\text{ALU}(\text{Op}, \text{Operand}_1, \text{Operand}_2) \rightarrow \text{Result}$$

$$\text{Registersatz} = \{\text{PC}, \text{IR}, \text{R0}, \ldots, \text{R}_{n-1}, \text{FLAGS}\}$$

$$\text{Instruktionszyklus} : \text{Fetch} \rightarrow \text{Decode} \rightarrow \text{Execute}$$

Pipelining (optional).

$$\text{Pipelining} \Rightarrow \text{Überlappung von Fetch/Decode/Execute}$$

(3) Arbeitsspeicher (RAM).
- Volatilität: RAM ist flüchtig; Inhalte gehen bei Ausschalten verloren.
- Speicherkonzepte: Adressierung durch Adressbus, Datentransfer durch Datenbus.
- Speicherhierarchie: Register $<$ Cache $<$ RAM $<$ Festplatte.

- Zugriffskosten variieren stark (L1/L2 Cache vs RAM).

$$T_{\text{Zugriff}} \approx T_{\text{Cache}} \text{ oder } T_{\text{RAM}}$$

Speicherhierarchie (Kurzform).
Register (schnell) → Cache (sehr schnell) → RAM (langsamer) → Festplatte (langsam)

(4) Prozesse.
- Ein Prozess ist eine Ausführungseinheit mit eigenem Adressraum und Zustand.
- Prozesszustände: Neu (Ready), Bereit, Laufend (Running), Blockiert (Waiting), Beendet.
- Kontextwechsel (Context Switch) kostet Zeit:

$$T_{\text{Context}} = T_{\text{speichern}} + T_{\text{wiederherstellen}} + T_{\text{Wechsel}}$$

- Interner vs externer Fragmentierung: Relevanz in Speichermanagement.

(5) Scheduling.
- Ziel: fair, effizient, kurze Wartezeiten, gute Reaktionszeit.
- Preemptive Scheduling: Unterbrechung laufender Prozesse; wichtig für Multitasking.
- Algorithmen (Beispiele):
- Round-Robin (RR) mit Zeitscheibe τ:

$$\text{Zeitquantum } \tau;\ \text{Kontextwechsel nach jeder Zeitscheibe}$$

- Shortest Job First (SJF): kleinstes nächstes CPU-Budget; optimal in der Erwartung für Durchsatz.
- Priority Scheduling: Berücksichtigt Prioritäten; ggf. Preemption.
- Multilevel Queue / Multilevel Feedback Queue: mehrere Ebenen mit unterschiedlicher Policies.
- Kennzahlen: Durchsatz, mittlere Wartezeit, Reaktionszeit, Ausnutzungsgrad.
- Scheduling-Entscheidungen beeinflussen die Performance von Anwendungen und das Reaktionsverhalten.

Zusammenfassung (Kernpunkte).
- Von-Neumann-Architektur: gemeinsamer Speicher für Code und Daten; Fetch-Decode-Execute-Zyklus.
- Prozessorarchitektur: Rechenwerk, Steuerwerk, Registers, ISA, Pipeline als Optimierungsmöglichkeit.
- Arbeitsspeicher: volatile, hierarchisiert; schnelle Register/Cache vor RAM.
- Prozesse und Scheduling: Kontextwechsel, Zustandsdiagramm, verschiedene Scheduling-Algorithmen zur CPU-Zuteilung.