Lernzettel

Von-Neumann-Architektur, Prinzipien, sichere Rechnerarchitektur und Mehrzyklus-Implementierung

Universität: Technische Universität Berlin
Kurs/Modul: Rechnerorganisation
Erstellungsdatum: September 20, 2025

Lernzettel: Von-Neumann-Architektur, Prinzipien, sichere Rechnerarchitektur und Mehrzyklus-Implementierung

(1) Grundprinzipien der Von-Neumann-Architektur. Die Von-Neumann-Architektur basiert auf einem gemeinsamen Speicher für Programm und Daten sowie einer zentralen Verarbeitungseinheit. Der Flaschenhals dieser Architektur ist der gemeinsame Bus (Von-Neumann-Bottleneck).

(2) Aufbau der Verarbeitungseinheit. Eine typische Von-Neumann-Architektur umfasst:

- Rechenwerk (ALU) und Steuerwerk (CU)
- Speicher (RAM/ROM) mit Adress- und Datenspeicher
- Registersatz: PC, IR, MAR, MDR, Allgemeinregister
- Bus-System: Adressbus, Datenbus, Kontrollbus
- Ein-/Ausgabe-System

(3) Register-Transfer-Ebene (RT-Ebene) der Befehlsbearbeitung.

- Befehlszyklus (Fetch-Decode-Execute) in RT-Operanden.
- Fetch-Schritte (je Zeile ein Display-Schritt):

$$\text{MAR} \leftarrow \text{PC}$$

$$\text{MDR} \leftarrow \text{Memory[MAR]}$$

$$\text{IR} \leftarrow \text{MDR}$$

$$\text{PC} \leftarrow \text{PC} + 1$$

- Decode/Prepare:

$$\text{Opcode} \leftarrow \text{IR}_{\text{Opcode}}$$

$$\text{Operand} \leftarrow \text{IR}_{\text{Operand}}$$

- Execute:

$$\text{ALU}_{\text{op}}(\text{Operands}) \rightarrow \text{Result}$$

- Typische Befehle (LOAD, STORE, ADD, JUMP) werden durch Mikrooperationen umgesetzt.

(4) Sichere Rechnerarchitektur: Grundprinzipien.

- Privilegierte und unprivilegierte Ausführungsebenen (User/Supervisor)
- Speicherverwaltung (MMU, Paging, TLB) zum Schutz von Programmen
- Interrupt- und Ausnahmebehandlung sowie Fehler-Logging
- I/O-Sicherheit: kontrollierter Direktzugriff (DMA-Schutz, I/O-Privilegs)
- Zugriffskontrollen, Context- und Policy-Based Security

(5) Mehrzyklus-Implementierung.

- In einem Mehrzyklus-Design wird eine Befehlsausführung in mehrere Taktzyklen unterteilt, statt in einen einzigen Zyklus zu fallen.
- Vorteile: geringere Maximaltaktfrequenz pro Zyklus, flexiblere Nutzung von Ressourcen, bessere Ausnutzung von Speicherzugriffen.
- Nachteile: längere durchschnittliche Befehlsdauer, komplexere Steuereinheit.
- Typische Zyklenbestandteile:

$$\text{Fetch} \Rightarrow \text{Decode} \Rightarrow \text{Execute}$$

$$T_{\text{instruction}} = N \cdot T_{\text{cycle}}$$

  wobei N die Anzahl der Zyklen pro Befehl ist.

(6) Beispielhafter Ablauf einer LOAD-Anweisung.

- Fetch-Schritte (je Zeile):

$$\text{MAR} \leftarrow \text{PC}$$

$$\text{MDR} \leftarrow \text{Memory[MAR]}$$

$$\text{IR} \leftarrow \text{MDR}$$

$$\text{PC} \leftarrow \text{PC} + 1$$

- Decode:

$$\text{Opcode} \leftarrow \text{IR}_{\text{Opcode}}$$

$$\text{Addr} \leftarrow \text{IR}_{\text{Address}}$$

- Execute-/Memory-Zyklus:

$$\text{MAR} \leftarrow \text{Addr}$$

$$\text{MDR} \leftarrow \text{Memory[MAR]}$$

$$\text{Register[Dest]} \leftarrow \text{MDR}$$

(7) Zusammenfassung und Ausblick.

- Von-Neumann-Architektur: gemeinsamer Speicher, zentrale Verarbeitungseinheit
- RT-Ebene: klare Abfolge Fetch-Decode-Execute
- Sicherheitsaspekte: Privilegien, MMU, Schutzmechanismen
- Mehrzyklus-Implementierung: Flexibilität vs. Befehlsdauer
- Relevante Verknüpfungen zur Speicherhierarchie (Cache, TLB, virtuelle Adressen)