# Organisatorisches (1)

- **Institut für Verteilte Systeme / AG Echtzeitsysteme und Kommunikation**
  - Ort: Gebäude 29, Etage 4
  - nett@ivs.cs.uni-magdeburg.de
  - Sekretariat: Frau Duckstein,  Tel. 67-18345
- **Web-Adresse**

  http://ivs.cs.uni-magdeburg.de/EuK
  - Rubrik „Lehrveranstaltungen“
  - Folien der Vorlesung im Web
  - Übungsaufgaben (nur im Web, immer nach der Vorlesung für die kommende Übung)
  - Mitteilungen
  - Literaturhinweise
- **Vorlesungsskript** in Kürze bei Coppenrath und Boeser erhältlich

# Organisatorisches (2)

- **Übungen**
  - **Übungsleiter** Deutscher, Hoffmann, Rutsch, Breß, Klonz
    - Mail-Kontakt: Jan.Hoffmann@ovgu.de
  - **Aufgabenzettel** wöchentlich im Web
  - **Übungsgruppeneinteilung** per Anmeldung diese Woche im Web
  - **Übungsbeginn** 8. April, theoretisch nach der zweiten Vorlesung, die aber ausfällt
  - **Anmeldungen bis** spätestens Freitag, 3. April

- **Vorausetzungen für Klausurteilnahme**
  - Mind. 66% aller Übungsaufgaben vorbereiten und votieren
  - Zulassungen zur Klausur werden nach der letzten Übung bekannt gegeben

- **Dringender Rat: Vorlesungen und Übungen regelmäßig besuchen, Quellen nutzen und in Übungen diskutieren.**

# Rechnersysteme

## Einordnung:

### Unterschiedliche Abstraktionsebenen eines Rechners:

**5. Anwendungsebene** (Anwendungsprogramme wie Word, Excel)

**4. Ebene der höheren Programmiersprachen** (C, $C^{++}$, Java ....)

**3. Assembler-Ebene** (symbolische Darstellung von Maschinenbefehlen)

**2. (Betriebssystem-Ebene)** (Programm in Maschinensprache, verwaltet die Betriebsmittel (z.B. Speicher)

**1. Maschinensprache-Ebene** (SW/HW-Schnittstelle eines Rechners, definiert seine Architektur)

**0. Hardware (Gatter)-Ebene** (definiert den digitalen, physikalischen Rechner, Konstruktionsplan)

### Definitionen:

Unter einem *$L_i$-Compiler(Übersetzer)* versteht man ein Maschinenprogramm, das ein Programm $P_i$ der Sprache $L_i$ in ein äquivalentes Programm $P_j$ der Sprache $L_j$ transformiert.

Unter einem *$L_i$-Interpreter* versteht man ein Maschinenprogramm, das ein Programm $P_i$ der Sprache $L_i$ Anweisung für Anweisung auf dem physikalischen Rechner ausführt. (Es wird also im Unterschied zum Compiler kein zu $P_i$ äquivalentes Programm erzeugt und abgespeichert.)

# Rechnersysteme

**Inhalt:**

- Wie werden Programme in einem Rechner ausgeführt aus der Sicht von Ebene 2 (Assembler-Programmierer)
  - welche Möglichkeiten gibt es, Befehlsfolgen vom Hauptspeicher in die CPU zu bringen und wie werden sie dann ausgeführt
  - welche Adressierungstechniken gibt es für Hauptspeicherzellen und CPU-Register

- Struktur und Funktionsweise der CPU
  - Adresspfad
  - Datenpfad
  - Kontrolleinheit (Befehlsdekodierung)
  - RISC-Konzept

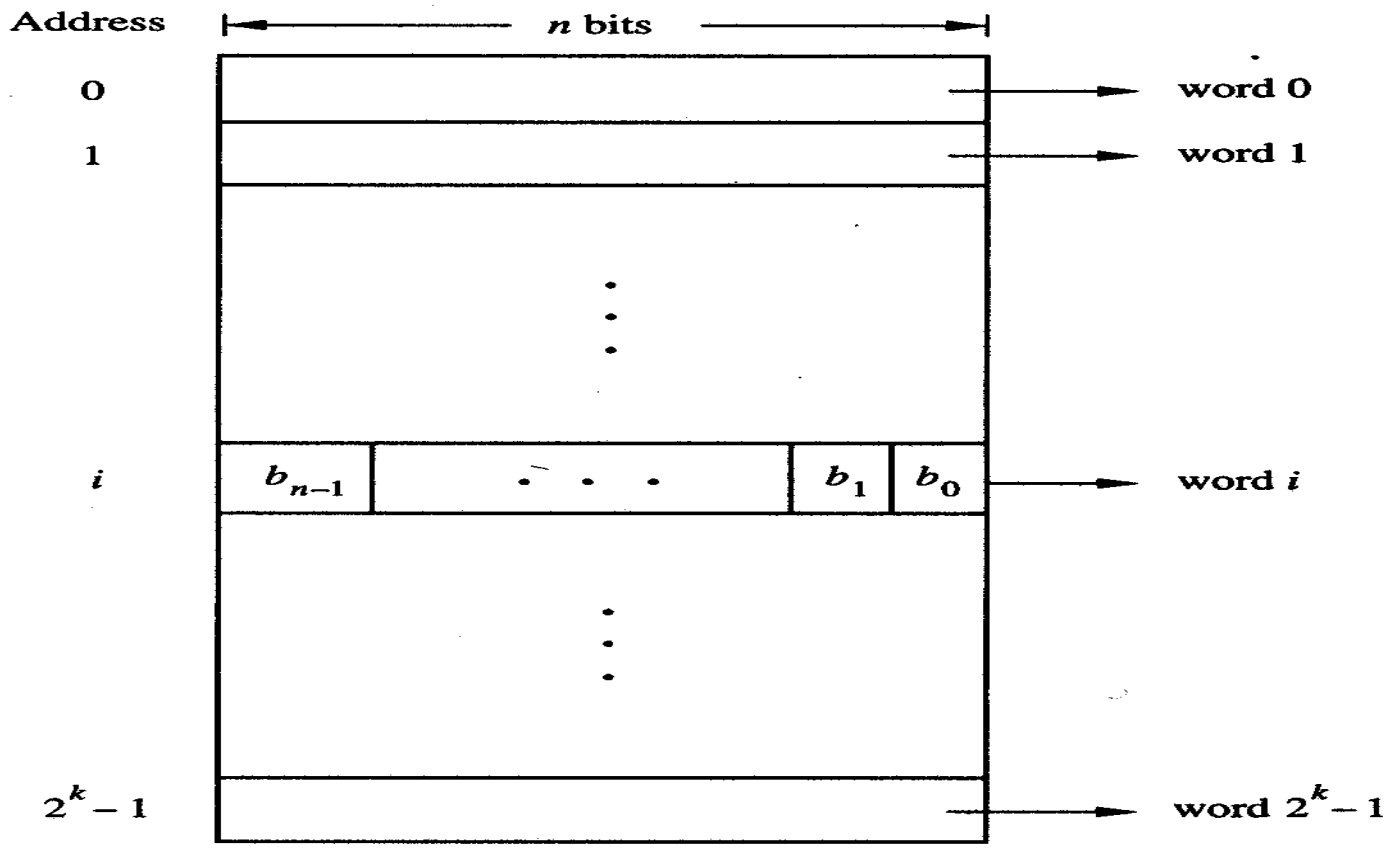
- Struktur und Funktionsweise des Hauptspeichers
  - Aufbau eines Speichers
  - Entwurf einer Speicherhierarchie
    - Cache
    - virtuelles Speicherkonzept

- Parallelrechner
  - Kommunikationsmodelle
  - Verbindungsnetzwerke
  - Taxonomie

# Adressierung und Befehlsfolgen (1)

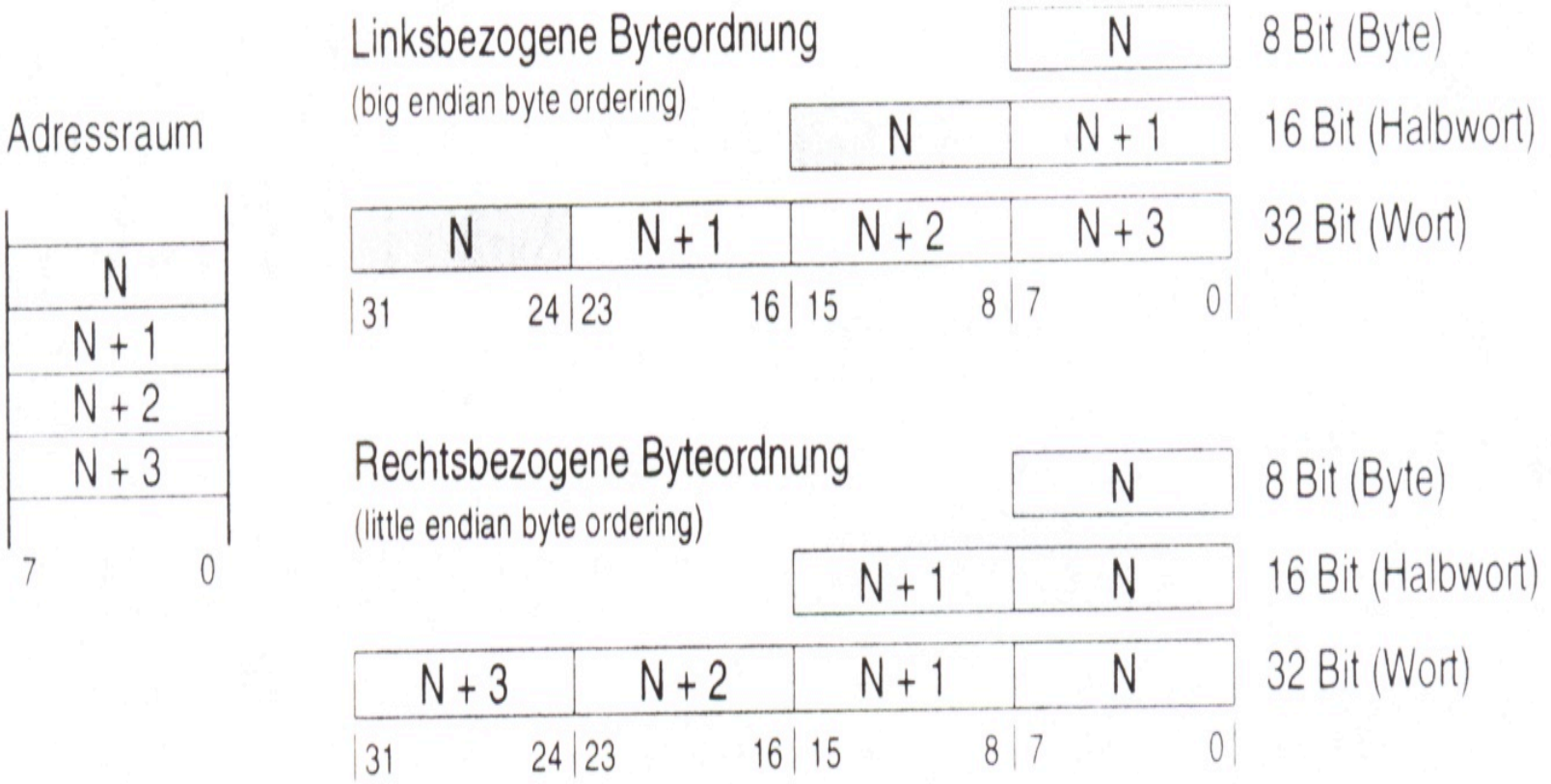
## 1.1 Speicherbelegung

### Hauptspeicheradressen



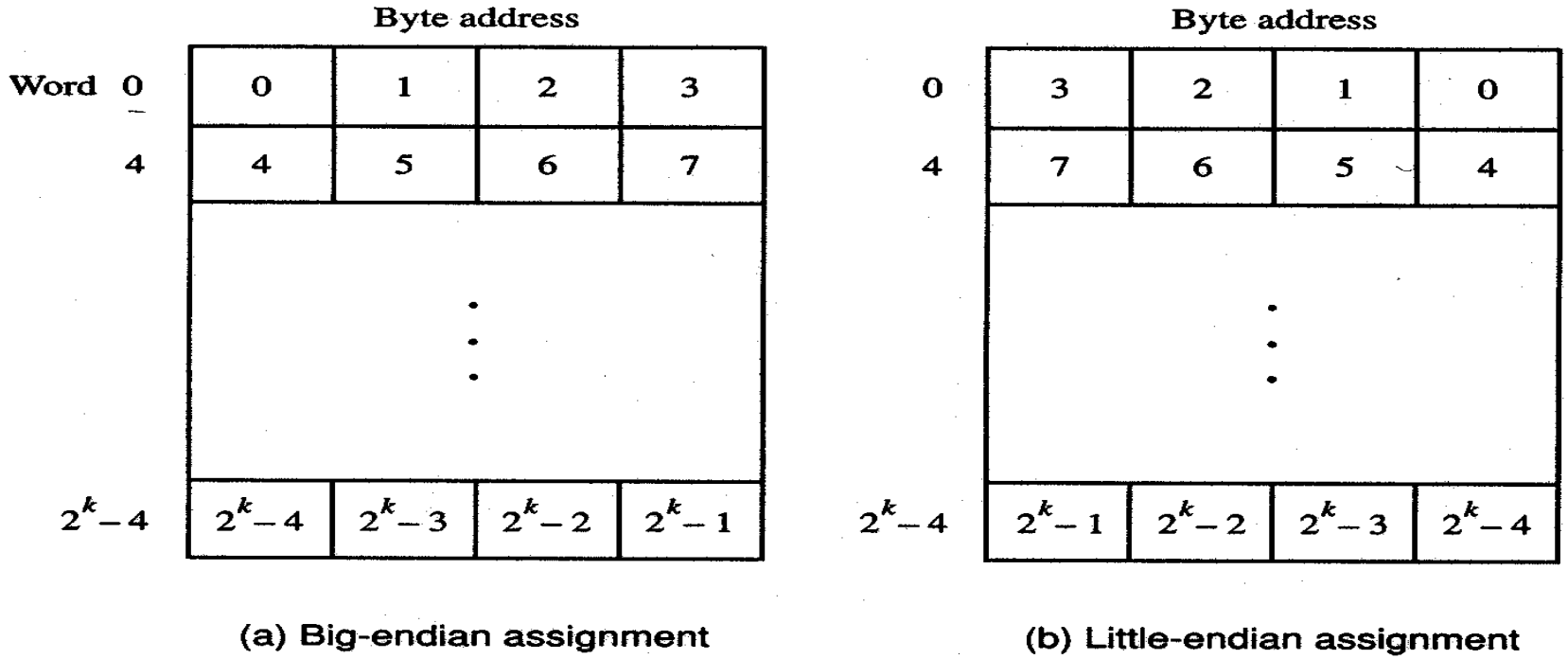
# Adressierung und Befehlsfolgen (2a)

## Linksbündige und rechtsbündige Byteordnung für Einheiten der Breiten 8, 16 und 32 Bit



# Adressierung und Befehlsfolgen (2)

**Beispiele für Byte-Adressierung**



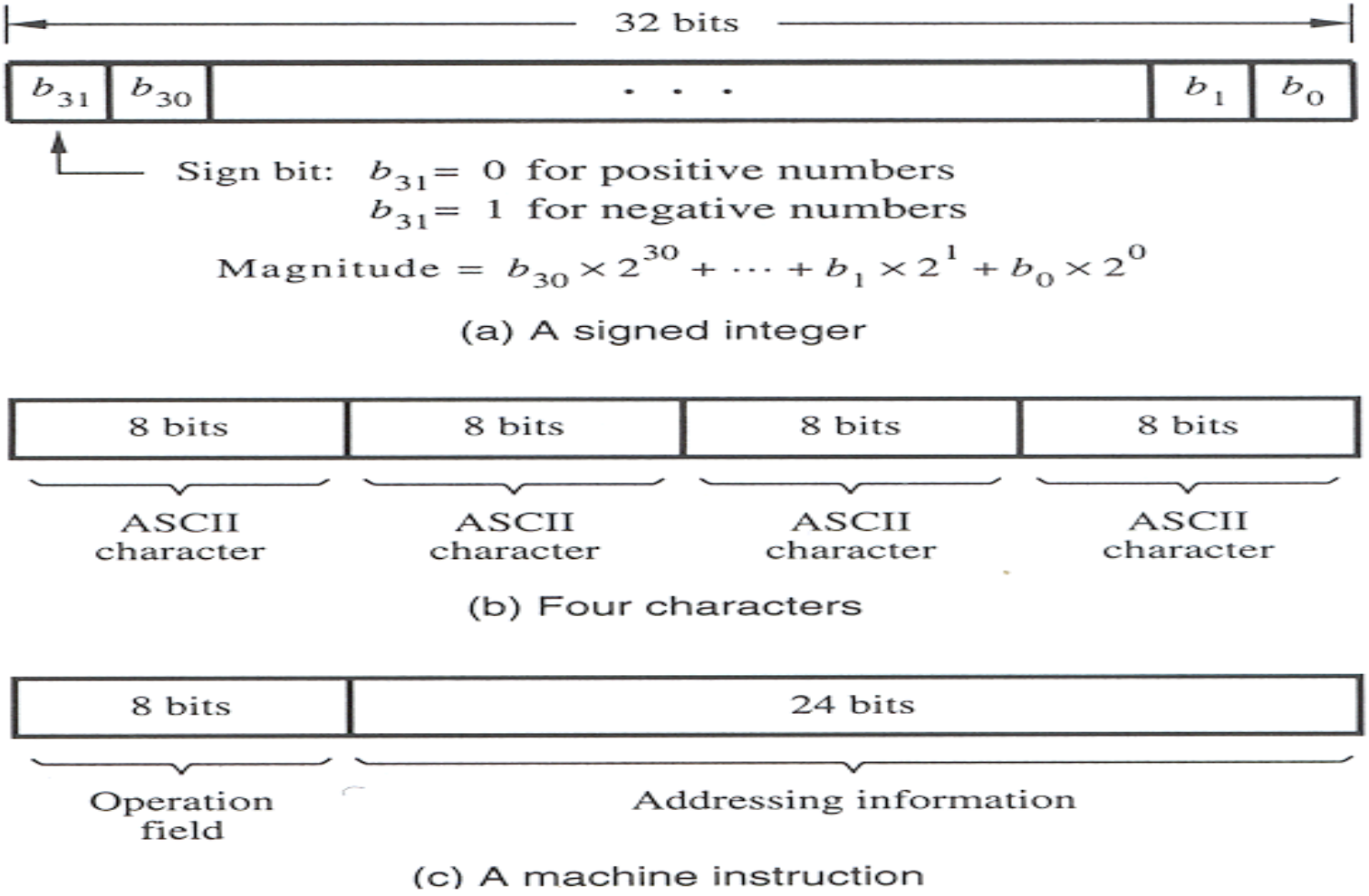

**Ein Wort beschreibt eine Adresse oder den Inhalt (Wert) einer Adresse!**

**Inhalte werden unterteilt in:**

- Befehle
- Operanden
  - Zahlen
  - Zeichen

# Adressierung und Befehlsfolgen (3)

**Beispiele für die unterschiedlichen Inhalte in einem 32-bit Wort**



# Adressierung und Befehlsfolgen (4a)

**Datentransfer zwischen Hauptspeicher und CPU-Registern**

- Speicherplätze (Adressen) werden durch Namen (Variable) identifiziert
- Der Inhalt (Wert) eines Speicherplatzes A wird durch [A] bezeichnet

**Ausführen von logisch/arithmetischen Operationen auf Daten**

Beispiel: C <----- [A] + [B]
durch

- 3-Adress-Befehle
- 2-Adress-Befehle
- 1-Adress-Befehle