GRUNDLAGEN

Computerinterne Datendarstellung - Zahlensysteme

Computerinterne Datendarstellung

relevante Informationseinheiten

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- binäre Zustände
- ganze Zahlen (un/signed)
- gebrochene Zahlen (Fest-/Gleitkomma)
- Zeichenketten, Grafiken
- Maschinenbefehle

Die physische Repräsentation erfolgt durch *Gruppen von n Bits*.

Bits	Anzahl Kombinationen
1	2
2	4
⋮	⋮
n	2^n

b_{n-1}	b_{n-2}	$\cdots$	b_4	b_3	b_2	b_1	b_0
MSB							LSB

MSB Most Significand Bit

LSB Least Significand Bit

Fragen:

- Ein Prozessor hat 12 Adressleitungen. Wie viele Adressen können adressiert werden?
- Wie viele Zahlen kann ich mit 8 Bit darstellen?
- Welches ist die kleinste und welches die größte darstellbare Zahl mit 8 Bit?
  - ☞ max. 256 Kombinationen, aber größte ist 255 (Null, siehe Dezimalsystem)

physische Datendarstellung

im Speicher

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- i.d.R. fixe Anzahl Bits/Informationseinheit
  - meist ganze Vielfache der Größe einer Speicherzelle
- entsprechen oft charakteristischen Hochsprach-Datentypen

Name	Bitzahl	Hochsprach-Datentyp
Byte	8	`char`
Word	16	`short, int`
Double Word	32	`int, long, float`

- geeignete Codierung zur 1:1 Zuordnung nötig

  Bitmuster $\Longleftrightarrow$ Bedeutung

Weitere Bezeichnungen:

- 4 Bit = Tetrade, auch Nibble, Halbbyte
- Word = 16 Bit = 2 Byte: High-Byte | Low-Byte

Bitmuster – Bedeutung:

- Bitmuster ist einfach Anordnung von Bit …
- Bedeutung ergibt sich aus Verwendung:
  0100 0001
    Zahl: 41 Hexadezimal, 65 Dezimal
    ASCII: Buchstabe „A“
    Maschinencode 8086: Befehl „INC CX“

<u>Im folgenden Bedeutungen:</u>

Vorzeichenlose Ganzzahlen

unsigned integer

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- natürliches Zahlensystem eines Rechners
- basiert auf Positionszahlensystem

Binärzahl

$$B = b_{n-1}b_{n-2}\dots b_1 b_0$$

Bitkombination

$$Wert(B) = b_{n-1} * 2^{n-1} + b_{n-2} * 2^{n-2} + \dots + b_1 * 2^1 + b_0 * 2^0$$

Berechnung des Wertes

mit 2 als Basis des Zahlensystem und 2^i Wert der binären Ziffer b_i

Hexadezimalzahl

$$H = h_{n-1}\dots h_1 h_0$$

$$Wert(H) = h_{n-1} * 16^{n-1} + \dots + h_1 * 16^1 + h_0 * 16^0$$

mit 16 als Basis und 16^i Wert der hexadezimalen Ziffer h_i

Alles wie beim Dezimalsystem:

Tausender 10^3	Hunderter 10^2	Zehner 10^1	Einer 10^0

Vorzeichenbehaftete Ganzzahlen

signed integer

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Zweierkomplement

$$B = b_{n-1}b_{n-2}\dots b_1 b_0$$

$$Wert(B) = -b_{n-1} * 2^{n-1} + b_{n-2} * 2^{n-2} + \dots + b_1 * 2^1 + b_0 * 2^0$$

MSB = Vorzeichen

alternativ: Vorzeichen-Betrags-Darstellung

$$B = \underline{b_{n-1}}\ \underline{b_{n-2}\dots b_1 b_0}$$

mit b_{n-1} als Vorzeichen und $b_{n-2}\dots b_1 b_0$ als Betrag

Nicht eineindeutig! Kaum verwendet

Zweierkomplement

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Anmerkungen

- Eigenschaften basieren auf Zahlen mit endlicher Stellenzahl
- Darstellung hängt von Stellenzahl *n* ab
  (-2 mit 8 Bit $\Rightarrow$ 0FEH, mit 16 Bit $\Rightarrow$ 0FFFEH)
  - s. Vorzeichenrichtige Erweiterung
- Most Significand Bit MSB *verkörpert* Vorzeichen (1 -> negativ)
- Zweierkomplement ermöglicht einfache Realisierung von Addition und Subtraktion

$$a - b \Rightarrow a + (-b)$$

Bildung des Zweierkomplementes

a) $ZK(Z) = EK(Z) + 1$, mit $EK(Z) = \overline{Z}$
b) $ZK(Z) = 2^n - Z$
c) Z rechts beginnend einschließlich erstes 1-Bit kopieren, Rest invertieren

Z:Zahl, EK: Einerkomplement

Warum ZKPL?

```
  5      0101       5         0101
 -3     -0011    +(-3)       +1101
  2      0010       2        10010
```

Subtraktion technisch komplizierter als Addition, deshalb nur Addition.
Und: Einsparung SUB-Modul | Schaltkreis | Transistoren

Demo: Zahl 6 nach -6 konvertieren, 4 Bit: 0110

```
a) EK:  1001     b) 2^4  10000     c)  0110
       +0001            - 0110         !  ←
   -7:  1010             01010         1010
```

Übung: Konvertieren der Zahlen

1 nach -1 → 4 Bit: 1111 → 8 Bit: 1111 1111 (FFh)

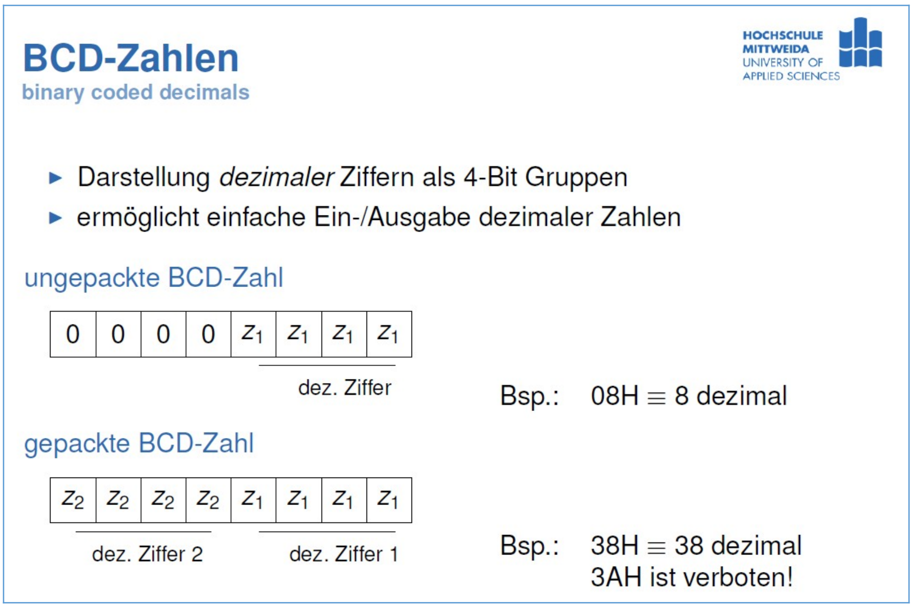
24d (8 Bit) binär 00011000 (18h) → 1110 1000 (E8h)

127d (8 Bit) 0111 1111 (7Fh) → 1000 0000 (80h)
(größte positive Zahl)

128d (8 Bit) 1000 0000 (80h) → 1000 0000 (80h)
(positiv / negativ)

136d (8 Bit) 1000 1000 = Lt. Definition ZKPL negative Zahl, Vorzeichen = 1 !
136d ja: als vorzeichenlose Ganzzahl, im 8Bit-ZKPL nicht möglich.

Zahlenbereich 8 Bit: -128 → 0 → +127 <u>ODER</u> 0 → 255
Bitkombination ist immer korrekt, Wertung liegt beim „Betrachter“



!!! BCD-Zahlen sind Bitkombinationen, Hex-Zahlen, KEINE Dezimalzahlen.
Es sind nur die Hex-Zahlen erlaubt, die einer Dez-Zahl entsprechen !!!

<u>Darstellung alphanumerischer Zeichen</u>

- Eindeutige Zuordnung:
  Bitkombination ←→ Zeichen (Ziffer, Buchstabe, Sonderzeichen, Steuerzeichen…)

ASCII - American Standard Code for Information Interchange

- 7-Bit-Code
- 00-1F: Steuerzeichen
- 20-7F: Ziffern, Buchstabe, Sonderzeichen
  20h – Leerzeichen
  30h – „0“, 31h – „1“ … 39 – „9“
  41h – „A“, 61h – „a“
- Landesspezifische Erweiterung: Nutzung des 8. Bit

Uni-Code – universeller Code

- Code mit variabler Länge, meist 16 Bit
- ISO 10646: Universal Coded Character Set
- Darstellung aller sinnvollen Schriftzeichen oder Textelement aller bekannten Schriftkulturen und Zeichensysteme

Logische Operationen

NOT (NICHT) (Einerkomplement) $Y = \overline{X}$	AND (UND) Y = X1 ^ X2	OR (ODER) Y = X1 v X2	XOR (Exclusiv-ODER, Antivalenz) Y = X1 (+) X2
0 -> 1 1 -> 0	0 v 0 -> 0 0 1 0 1 0 0 1 1 1	0 ^ 0 -> 0 0 1 1 1 0 1 1 1 1	0 (+) 0 -> 0 0 1 1 1 0 1 1 1 0