IT

Zahlensysteme

Zahlendarstellung in einem Stellenwertcode (jede Stelle hat einen bestimmten Wert)
Def. *Code*: Eindeutige Abbildungsvorschrift für die Abbildung eines Zeichen-Vorrates in einem anderen Zeichenvorrat.

Dezimalsystem

Die Basis (b) des Dezimalsystems ist b=10
Die Ziffern (a) des Dezimalsystems sind $a_n..a_0 = 0...9$
Die höchste Ziffer ist „b-1=9“

Beispiel:
3465 = 3*1000 + 4*100 + 6*10 + 5*1
$= 3*10^3 + 4*10^2 + 6*10^1 + 5*10^0$
z = $a_3 *10^3 + a_2 *10^2 + a_1 *10^1 + a_0 *10^0$

Allgemeine Schreibweise

$$z = a_n \cdot b^n + a_{n-1} \cdot b^{n-1} + ...a_1 \cdot b^1 + a_0 \cdot b^0$$

$$\sum_{i=n}^{i=0} a_i \cdot b^i$$

Binär- oder Dualsystem

Die Basis (b) des Binärsystems ist b=2
Die Ziffern (a) des Binärsystems sind $a_n..a_0 = 0{,}1$
Die höchste Ziffer ist 1

Beispiel:
10011b = 19d
(b für binär) (d für dezimal)
$= 1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 1*2^0$
= 1*16 + 0*8 + 0*4 + 1*2 + 1*1
= 19d

Allgemeine Schreibweise

$$z = a_n \cdot 2^n + a_{n-1} \cdot 2^{n-1} + ...a_1 \cdot 2^1 + a_0 \cdot 2^0$$

$$\sum_{i=n}^{i=0} a_i \cdot 2^i$$

Hexadezimal- oder Sedezimalzahlen

Die Basis (b) des Hexadezimalsytems ist b=16
Die Ziffern (a) des Hexadezimalsytems sind :

Hex.	0	1	2	3	...	8	9	A	B	C	D	E	F
Dez.	0	1	2	3	...	8	9	10	11	12	13	14	15

Allgemeine Schreibweise

$$z = a_n \cdot 16^n + a_{n-1} \cdot 16^{n-1} + ...a_1 \cdot 16^1 + a_0 \cdot 16^0$$

$$\sum_{i=n}^{i=0} a_i \cdot 16^i$$

Die höchste Ziffer ist b-1 = Fh (15d)

Konvertierungsverfahren

Dezimal → Binär

70d → 1000110b

1. Methode

64	32	16	8	4	2	1	
(2^6)	(2^5)	(2^4)	(2^3)	(2^2)	(2^1)	(2^0)	=> Potenzen von 2 (binär)
1	0	0	0	1	1	0	=> Kommt in der 70 vor (mit 64 anfangen)

170d → 10101010 b

256	128	64	32	16	8	4	2	1	=> Potenzen von 2 (binär)
	1	0	1	0	1	0	1	0	=> Kommt in der 170 vor

2. Methode

70d

70 : 2 = 35 Rest 0
35 : 2 = 17 Rest 1
17 : 2 = 8 Rest 1
8 : 2 = 4 Rest 0
4 : 2 = 2 Rest 0
2 : 2 = 1 Rest 0
1 : 2 = 0 Rest 1

70d = 1000110b

!!Binäre Zahl von unten lesen!!!!!!!!

...
3. Ziffer
2. Ziffer
1. Ziffer

170d

170 : 2 = 85 Rest 0
85 : 2 = 42 Rest 1
42 : 2 = 21 Rest 0
21 : 2 = 10 Rest 1
10 : 2 = 5 Rest 0
5 : 2 = 2 Rest 1
2 : 2 = 1 Rest 0
1 : 2 = 0 Rest 1

170d = 10101010b

Merke: Bei der 2. Methode immer durch die Basis des Zahlensystems dividieren.

Dezimal → Hex

259d → 103h

1. Methode

4096	256	16	1	=> Potenzen von 16 (hex)
(16^3)	(16^2)	(16^1)	(16^0)	
	1	0	3	=> Kommt in der 259 vor (mit 256 anfangen)

18380d →

4096	256	16	1	=> Potenzen von 16 (hex)
(16^3)	(16^2)	(16^1)	(16^0)	
4	7	C	C	=> Kommt in der 259 vor (mit 4096 anfangen)

Am besten nach folgendem System:

18380 : 4096 = 4,48... also 4x 4096 → 4 *4096 = 16384
18380-16384= 1996 Restbetrag (mit diesem Wert weiterrechnen)→
1996 : 256 = 7,79... also 7x 256 → 7*256 = 1792
1996 - 1792 = 204 Restbetrag (mit diesem Wert weiterrechnen)→
204 : 16 = 12,75 also 12x 16 <u>!!! 12d ist = Ch !!!!</u> also Cx 16 → 12*16 = 192
204 - 192 = 12 (mit diesem Wert weiterrechnen)→
12 : 1 = 12 also 12x 1 ... also Cx 1 → <u>**47CCh**</u>

2. Methode

259d

259 : 16 = 16 Rest 3	3. Ziffer
16 : 16 = 1 Rest 0	2. Ziffer
1 : 16 = 0 Rest 1	1. Ziffer

259d = 103h

18380d

118380 : 16 = 1148 Rest C (12d)
1148 : 16 = 71 Rest C (12d)
71 : 16 = 4 Rest 7
4 : 16 = 0 Rest 4

18380d = 47CCh

Merke: Bei der 2. Methode immer durch die Basis des Zahlensystems dividieren.

Binär → Hex

1100111111100b → 19FCh

Die binäre Zahl in „vierer-Päckchen“ aufteilen:

Exponenten der Basis 2^b $(2^3, 2^2, 2^1, 2^0 = 8,4,2,1,)$	1	8 4 2 1	8 4 2 1	8 4 2 1
Binär-Zahl	1	1 0 0 1	1 1 1 1	1 1 0 0
Ergebnis der Addition, das den Hex Wert ergibt	1	9	15	12
Addition der Exponenten der binären 1-Werte	1	8+1	8+4+2+1	8+4
Hex Zahl	1	9	F	C

Hex → Binär

8BF3h → 1000101111110011b

Hex Zahl	8	B (11d)	F (15d)	3
Exponenten der Basis 2^b $(2^3, 2^2, 2^1, 2^0 = 8,4,2,1,)$	8 4 2 1	8 4 2 1	8 4 2 1	8 4 2 1
Welche addierten Exponenten ergeben als Summe die Hex Zahl ?	8	8+2+1	8+4+2+1	2+1
Umwandeln in binär, wobei nicht addierte Expon. = 0 Und adierte Expon. = 1	1 0 0 0	1 0 1 1	1 1 1 1	0 0 1 1
Binär-Zahl	1000101111110011b			

Binäre Arithmetik

Addition

	Summe	Übertrag
0 + 0	0	0
0 + 1	1	0
1 + 0	1	0
1 + 1	0	1
1 + 1 + 1	1	1

Bei der Addition immer nach der Tabelle gehen.

Beispiel:

```
   1 0 1 1
+  1 0 1 1
_1__1_1____ Übertrag
 1 0 1 1 0
```

Üt= Übertrag

```
1+1          = 0 Üt 1
1+1+1(Üt.) = 1 Üt 1
0+0+1(Üt)  = 1
1+1          = 0 Üt 1
Üt 1         = 1
```

Subtraktion

1. Methode

	Differenz	Entleihung (borrow)
0 - 0	0	0
0 - 1	1	1
1 - 0	1	0
1 - 1	0	0

Bei der Subtraktion immer nach der Tabelle gehen.

Beispiel:

```
     11 11
   1  0  0  1
-  0  1  1  1
 ____________
         1  0
```

2. Methode : *Komplementäraddition*

```
Bsp1: 26D    =>    11010b
      -11D   =>   - 1011b
      ____        _______
       15D   =>     1111b
```

} Einfach umgerecnnet, ist so aber micht möglich, erst das 2er Komplement bilden (siehe Kasten)

```
1. Schritt     11010b
              -01011b  ausfüllen der leeren Stellen
               ---------
2. Schritt     11010b
              +10100b Einerkomplement -> invertieren
              +      1b Zweierkomplement -> 1 addieren
               ---------
3. Schritt     11010b addieren
              +10101b
               _______
              [1]01111b
```

Ergebnis liegt nicht als Komplement Zahl vor sondern ist das korrekte Ergebnis (Da erste Ziffer die 1 ist)

> *Bildung des 2er Komplements:*
>
> Das Komplement einer Binärzahl erhält man , indem man jede 0 durch eine 1 und jede 1 durch eine 0 ersetzt (invertieren). Dann zusätzlich eine 1 addieren.
>
> ```
>  1011b => 2er Kompl. = 0101b
>   ||||
>   0100b   (In Additions-Tabelle
> +    1b      nachschauen)
> ________
>   0101b
> ```

Bsp 2 :

11D	=>	1011b	Einfach umgerechnet,
-26D	=>	- 11010b	ist so aber micht möglich,
_______		________	erst das 2er Komplement
-15D	=>	11110001b	bilden (siehe Kasten)

> *Vorrsicht:*
>
> Der Taschenrechner zeigt als Ergebnis 1111110001 an, richtig sind aber 8 Stellen, wobei die 8. entweder 1 für negatives Ergebnis oder 0 für ein positives Ergebnis ist.
>
> Also wenn beim 3. Schritt die 1. Ziffer (also der letze Übertrag) 1 ist ist dies das korrekte Ergebnis.
> Wenn der letzte Übertrag 0 ist muss wieder invertiert und + 1 gerechnet werden.
> Dieses Ergebnis wird dann vor das erste gehängt (!!nicht mehr als 8 Stellen insgesamt).
> Am besten in die Tabellen schauen und auf den Zettel ☺ siehe Anhang-

1. Schritt 01011b ausfüllen der leeren Stellen
-11010b
----

2. Schritt 01011b
\+ 00101b Einerkomplement -> invertieren
\+ 1b
----

3. Schritt 01011b addieren
+00110b
_1_1__
$\boxed{0}$10001b

Ergebnis liegt als Komplement vor und ist negativ, also wieder invertieren und + 1

4. Schritt 01110b Einerkomplement vom Ergebnis (10001b invertiert)
+00001b Zweierkomplement -> 1 addieren
____
1111b => vor erstes Ergebnis schieben => <u>11110001b</u>

Ps: Vorzeichenbehaftete Dualzahlen nennen sich in den gängigen Programmiersprachen Signierte Dualzahlen (signed Integer)

Anhang: Subtraktion durch Komplementäraddition in Dezimalsystem

Normalschreibweise		*Komplementäradition*
9D	=>	9D
- 2D	=>	+ 8D Zehnerkomplement von 2
_______		_______
7D	=>	$\boxed{1}$7D Keine Komplementzahl (weil 1 als Übertrag, diese fällt weg)
8D	=>	8D
- 9D	=>	+ 1D Zehnerkomplement von 9
_______		_______
-1D	=>	$\boxed{0}$9D 9 Ergebnis ist eine Komplementzahl (weil 0 als Übertrag)
		also wieder Koplement von 9 zur Basis10 ist 1 also <u>-1.</u>

Multiplikation

	Differenz
0 * 0	0
0 * 1	0
1 * 0	0
1 * 1	1

Bei der Multiplikation immer nach der Tabelle gehen.

Beispiel:
10010000 * 1001

```
______________
   10010000
    00000000
     00000000
      10010000
----------------------
    1010010000
```

Auf die richtige Schreib-
weise achten.
Schräg versetzt und dann
addieren

Division

Beispiel:

10010000 : 1001

```
______________
   10010000
   00000000
   00000000
   10010000
---------1------------
 100100000
```

Auf die richtige Schreib-
weise achten.
direkt untereinander und
dann addieren.

Das sind die relevanten Stellen, der letzte Übertrag fällt raus, Ergebnis : 10000

Grundlagen der Digitaltechnik

Lösungsmöglichkeiten

- Digitale Verknüpfungsglieder (*Logikbausteine*)
- Verbindungsprogrammierte Steuerung (z.B. mehrere Schalter hintereinander
- Speicherprogrammierbare Steuerung *SPS*)
- Pc inclusive Programm (*IPC => Industrie-Pc*)
- Micro-Controller (μC)
- Programmierbare Logicbausteine (*PLD*) z.B. *GAL*, *PAL* ...

Boolesche Algebra

Wahrheitstabelle von ODER

A	**B**	**Q**
0	0	0
0	1	1
1	0	1
1	1	1

$$Q = \overline{A} \bullet (UND) B + (ODER) A \bullet \overline{B} + A \bullet B = A + B$$

Rechengesetzte

1. Kommutativgesetze

$A \bullet B = B \bullet A \qquad A + B = B + A$

2. Assoziativgesetze

$$A \bullet (B \bullet C) = (A \bullet B) \bullet C = A \bullet B \bullet C$$
$$A + (B + C) = (A + B) + C = A + B + C$$

3. Distributivgesetze

$$A \bullet (B + C) = (A \bullet B) + (A \bullet C)$$
$$A + (B \bullet C) = (A + B) \bullet (A + C)$$

=> Die Rechenarten sind gleichrangig (k.P.vor- Rechn.)

4. *Absorptionsgesetze*

A + A = A	A ● A = A
A + 0 = A	A ● 0 = 0
A + 1 = 1	A ● 1 = A

5. *Gesetz der doppelten Verneinung*

$$\overline{\overline{A}} = A$$

6. *De Morgansche Identitäten*

$$\overline{A \bullet B} = \overline{A} + \overline{B}$$

$$\overline{A + B} = \overline{A} \bullet \overline{B}$$

7. Zusatz

$A + \overline{A} = 1 \qquad A \bullet \overline{A} = 0$

Verknüpfungsgleichungen

Beispiel Antivalenz

1. Disjunktive Normalform

Die einzelnen Variablen werden durch UND verknüpft , die Teilaussagen durch ODER

a) $y = (\overline{A} \bullet B) + (A \bullet \overline{B})$

b) $y = (\overline{A} \bullet \overline{B}) + (A \bullet B)$

2. Konjunktive Normalform

Die Einzelnen Variablen werden durch ODER verknüpft, die Teilaussagen durch UND.

a) $y = (A + B) \bullet (\overline{A} + \overline{B})$

b) $y = (A + \overline{B}) \bullet (\overline{A} + B)$

Aufgabe 1 (siehe Zettel auf nachfolgender Seite)

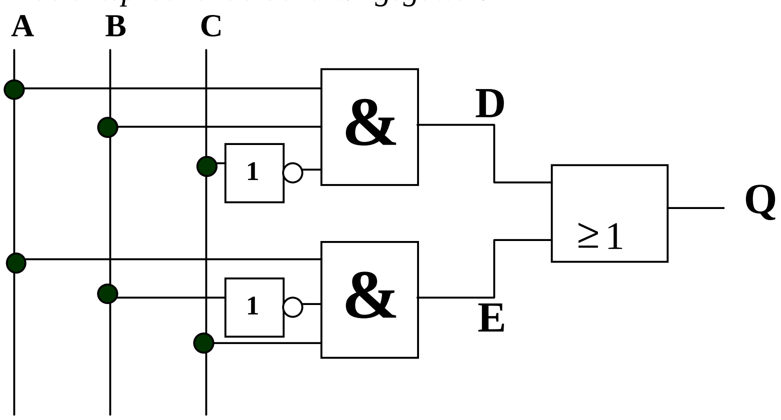
A Geld	**B** Kaffee	**C** Tee		**D** Kaffee Abgabe	**E** Tee Abgabe	**Q** Wasser in Becher
0	0	0		0	0	0
0	0	1		0	0	0
0	1	0		0	0	0
0	1	1		0	0	0
1	0	0		0	0	0
1	0	1		0	1	1
1	1	0		1	0	1
1	1	1		0	0	0

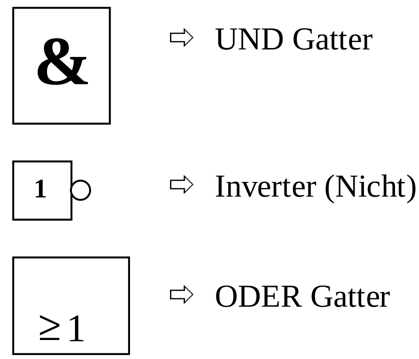
Die Anzahl der Kombination (2^n) ist abhängig von der Anzahl der Variablen , in diesem Fall(Variablen A,B und C also 3) $2^3 = 8$ Kombinationsmöglichkeiten (Zeilen in der Tabelle)

$D = A \wedge B \wedge \overline{C}$ $\qquad$ $E = A \wedge \overline{B} \wedge C$ $\qquad$ $Q = A \wedge B \wedge \overline{C} \vee A \wedge \overline{B} \wedge C$

$\boxed{\begin{array}{l} \wedge\,(\bullet) = UND \\ \vee\,(+) = ODER \\ \overline{} = NICHT \end{array}}$

Das entsprechende Schaltungsgatter:

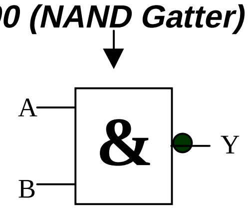




Aufgabe 3: (Pascal Programm das die Wertetabelle der UND bzw ODER Funktion ausgibt)

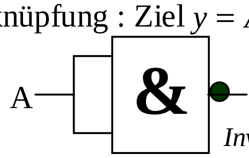
```
PROGRAM xyz;
USES crt;
VAR i,j : boolean;

BEGIN
  CLRSCR;
  FOR j:= false TO true DO
        FOR i:= false TO true DO
                WRITELN (j : 6 , i : 6 , j AND i :6 ) // XOR statt AND für oder Funktion
END.
```

Aufbau einer Schaltung ausschließlich mit dem Baustein 7400 (NAND Gatter)



1) **Nicht** Verknüpfung : Ziel $y = \overline{A}$ Wir haben $y = \overline{A \bullet B}$

$y = \overline{A \bullet A}$

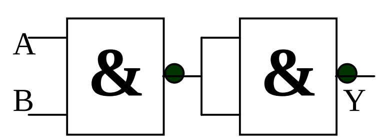


Inverter (Parallelgeschaltete Eingänge, aus A wird B)

$y = \overline{A}$ (Adsorptionsgesetz $A \bullet A = A$)

2) **UND** Verknüpfung : Ziel $y = A \bullet B$ Wir haben $y = \overline{A \bullet B}$

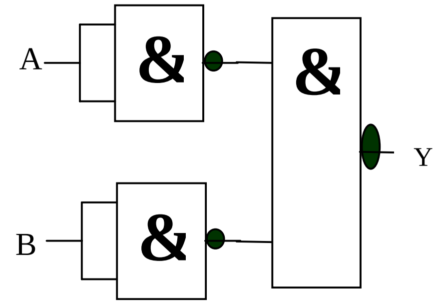
$y = \overline{\overline{A \bullet B}}$ (Doppelte Verneinung => positiv, also $y = A \bullet B$



In Reihe geschaltet => positiv

3) **ODER** Verknüpfung : Ziel $y = A + B$ Wir haben $y = \overline{A \bullet B}$

$y = \overline{\overline{A + B}}$ (Doppelte Verneinung) => $y = \overline{\bar{A} \bullet \bar{B}}$



2 Inverter für eine Schaltung