CDT Zusammenfassung

René Bernhardsgrütter, 08.04.2012

# Basics und Darstellung von Zahlen

Bit 🡸 binary digit: Eine Ziffer die nur 2 mögliche Werte annehmen kann.  
Wenn das System kein Gedächtnis hat, dann spricht man von einem **kombinatorischen** System  
Für N Eingänge hat man 2 mögliche Eingangskombinationen.

|  |  |  |
| --- | --- | --- |
| **NOT** | Y= !A |  |
| **AND** | Y = A & B |  |
| **OR** | Y = A # B |  |
| **XOR** | Y = A $ B  Wenn ein Eingang immer auf 1, dann ist das Gatter ein Inverter. |  |

# Zahlensysteme

|  |  |  |
| --- | --- | --- |
| Name | Basis | Wertebereich |
| Dezimalsystem | 10 | 0-9 |
| Hexadezimalsystem | 16 | 0-15 => 0-F |
| Binärsystem | 2 | 0-1 |

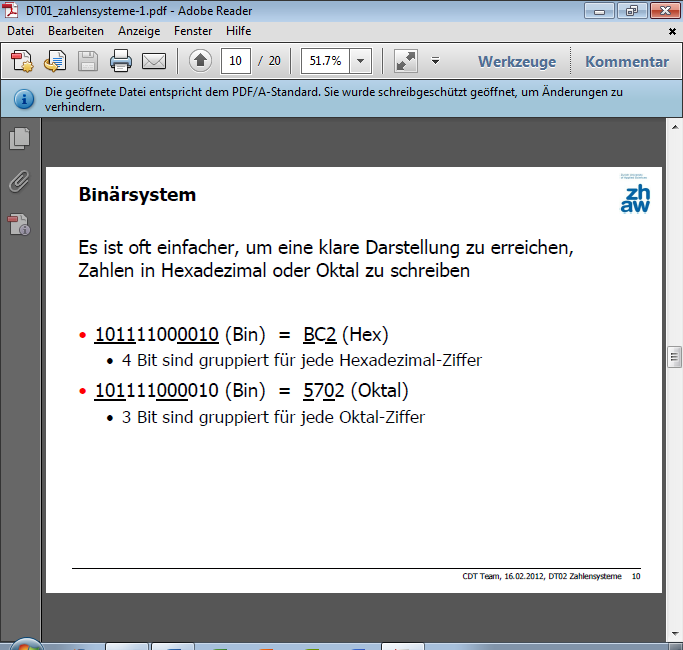
Anzahl darstellbare Zahlen immer: Bei n Ziffern => Basisn

Binärsystem: **MSB (Most Significant Bit d.h. Bit mit höchster Wertigkeit)** ist die am weitesten links stehende Ziffer. **LSB (Least Significant Bit d.h Bit mit niedrigster Wertigkeit)** ist die am weitesten rechts stehende Ziffer.

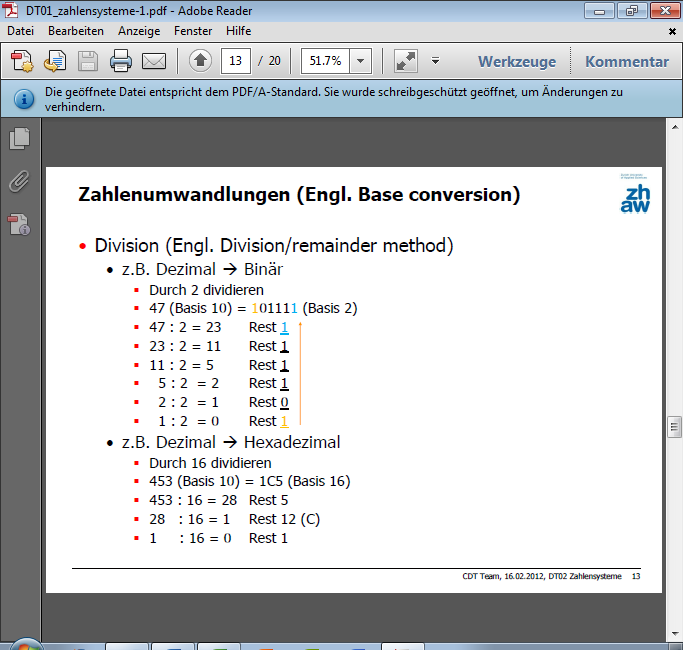
Nibble = 4 Bits  
Byte = 8 Bits = 2 Nibbles  
Word = mehr als 8 Bits, oft eine Gruppe von 2 Bytes (= 16 Bit)  
double Word = oft eine Gruppe von 32 Bits (= 4 Bytes)

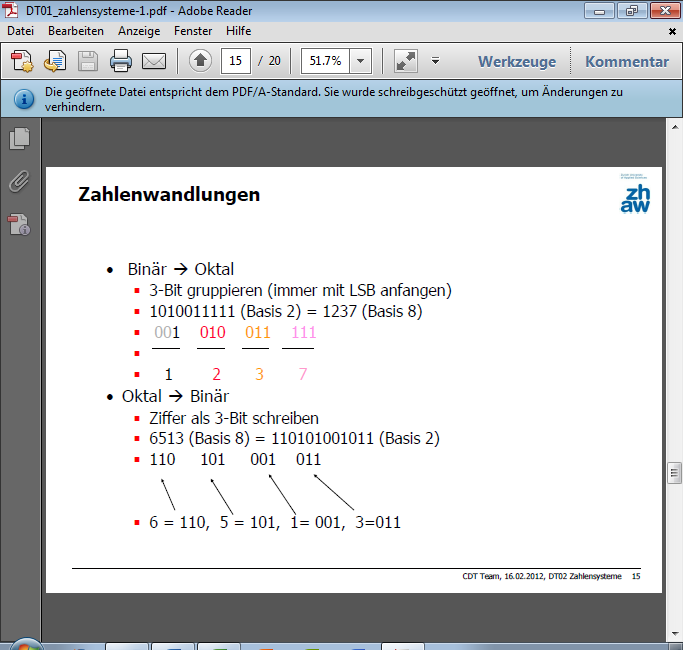
## Umrechnung

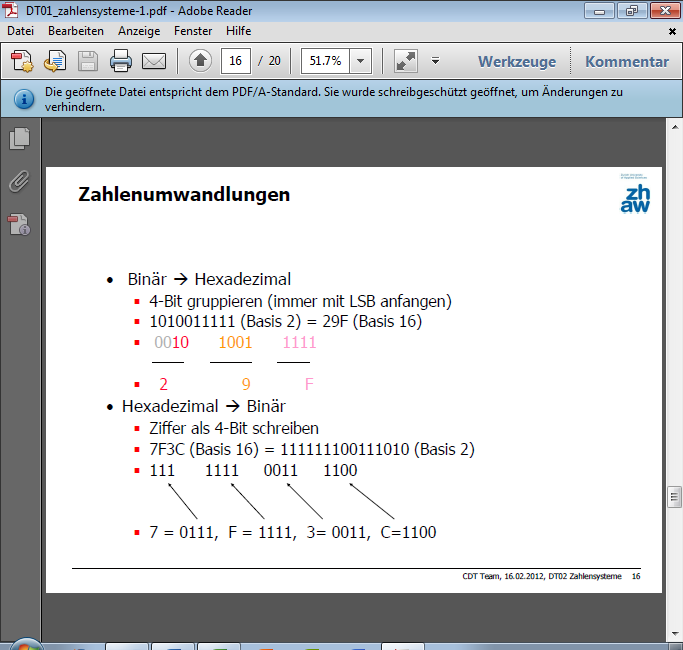
**Wenn nicht von einem hohen System (Basis > 2), dann immer über das Binärsystem, welches als Zwischenschritt dient, zum Zielsystem.**



Zuverlässige und merkbare Methode für alle Systeme:







# Schaltalgebra

Die Prioritätsreihenfolge sieht folgendermassen aus:

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | ( ) | Klammern |  |
| 2. | ! | Negation | NOT |
| 3. | & | Konjunktion | AND |
| 4. | # | Disjunktion | OR |

## Axiome

X1, X2, …, Xn sind jeweils die verfügbaren Eingänge.

OR: Falls X1 = X2 = … = Xn = 0, dann ist die Schaltung = 0, sonst = 1.

AND: Falls X1 = X2 = … = Xn = 1, dann ist die Schaltung = 1, sonst = 0.

XOR: 1 $ 1 = 0, 0 $ 0 = 0, 0 $ 1 = 1, 1 $ 0 = 1  
🡺 Wenn Eingänge verschieden, Ausgang = 1

## Theoreme

Verknüpfung mit 0: 0 & X = 0  
0 # X = X

Verknüpfung mit 1: 1 & X = X  
1 # X = 1

Verknüpfung mit sich selbst: X & X = X  
X # X = X  
X & !X = 0  
X # !X = 1

Kommuntativgesetzte: X1 & X2 = X2 & X1  
X1 # X2 = X2 # X1

Assoziativgesetze: (X1 & X2) & X3 = X1 & (X2 & X3)  
(X1 # X2) # X3 = X1 # (X2 # X3)  
Die Klammern haben höchste Priorität (zuerst ausgeführt)

Distributivgesetze: (X1 # X2) & X3 = (X1 & X3) # (X2 & X3)  
(X1 & X2) # X3 = (X1 # X3) & (X2 # X3)

Vereinfachungsgesetze: X1 # (X1 & X2) = X1  
X1 & (X1 # X2) = X1  
X1 # (!X1 & X2) = X1 # X2  
X1 & (!X1 # X2) = X1 & X2

Gesetze von de Morgan (Inversionsgesetze):  
!(X1 & X2 & ... & Xn) = !X1 # !X2 # ... # !Xn  
!(X1 # X2 # ... # Xn) = !X1 & !X2 & ... & !Xn

Satz von Shannon: !F (X1, X2, ... Xn, &, #) = F (!X1, !X2, ... ,!Xn, #, &)

# DNF: Disjunktive Normalform

Man verwendet die DNF, um von einem Ausgangszustand zu einem Zielzustand, der sogenannten Endlösung, zu kommen. Man möchte herausfinden, welche Kombinationen zur Lösung führen Dazu macht man die DNF.

Die Zwischenlösungen der einzelnen Verknüpfungen heissen **Minterme**. Solche die 1 ergeben heissen **gute Minterme**, die anderen heissen **schlechte Minterme**.

Für n Variablen (Eingänge) gibt es 2n Minterme.

K ist der Ausgang. K ist die OR-Verknüpfung **aller** guten Minterme. **K ist die disjunktive Normalform**. Sieht etwa so aus:

**K = (..&..&..&) # (..&..&..&)…**

Vorgehen:

1. Wahrheitstabelle machen und gute Minterme heraussuchen.
2. Diese mit den Gattern (also den logischen Verknüpfungen) aufschreiben.
3. Die Aussage vereinfachen, mit dem De Morgan Theorem oder den andere Regeln.  
   Darauf achten, dass wenn möglich nur Gleiche Gattertypen verwendet werden (dafür wird meistens NOT gebraucht).

# KNF: Konjunktive Normalform

Statt für K = 1, wie bei der DNF, schreibt man bei der KNF die Funktionen für K = 0. Man invertiert dafür das K. Dies kann besser sein, wenn man so weniger schreiben oder rechnen muss.

**Man muss die Maxterme verknüpfen: Ein Maxterm ist eine OR-Verknüpfung, welche die Eingangsgrössen entweder in direkter oder in negierter Form enthält.**

**Jeder Maxterm entspricht einer Zeile der WT, wobei die Variable in negierter Form einzusetzen ist, falls in der WT an der entsprechenden Stelle eine 1 steht, und in direkter Form, falls eine 0 steht.**

**!K = (..&..&..&) # (..&..&..&)…**

Dann das De Morgan Theorem anwenden:

**K = (..# ..#..) & (..# ..# ..) & (..#..#)**

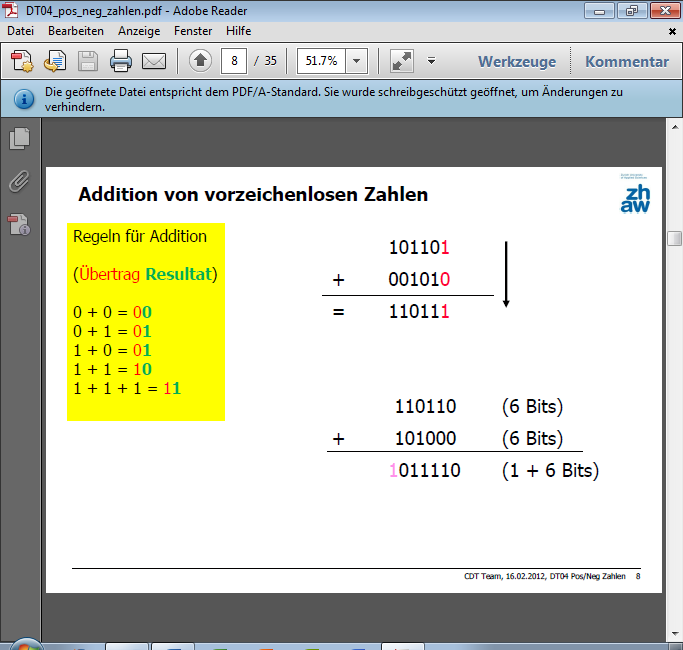
Die konjunktive Normalform ist die AND Verknüpfung **aller** Maxterme, die zu den Zeilen mit K = 0 gehören.

# Signed und unsigned numbers

Bei einer **vorzeichenlosen Operation** zeigt das **Carry-Flag** (CF) eine Bereichsverletzung (Übertrag oder Unterlauf).

Bei einer **vorzeichenbehafteten Operation** zeigt das **Overflow-Flag** (CF **XOR** (Übertrag zum MSB des Resultats)) eine Bereichsverletzung (Über- oder Unterlauf).

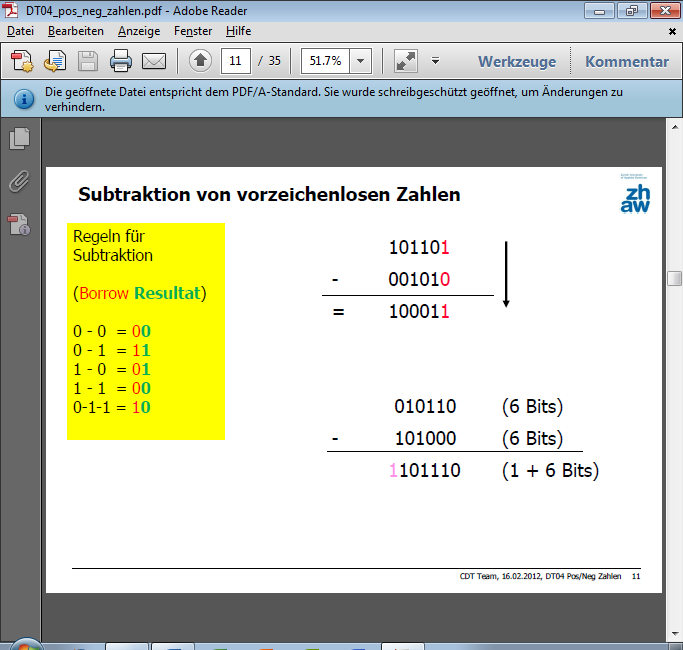
## Addition von vorzeichenlosen Zahlen:



**Carry-Bit = Übertrag** 🡺 Nicht genug Bits vorhanden, um die Zahl korrekt darzustellen!

## Subtraktion von vorzeichenlosen Zahlen:

Hier muss man die Zahlen wirklich subtrahieren, nicht wie bei den vorzeichenbehafteten Zahlen (dort kann man sie addieren)!



## Darstellung vorzeichenbehafteter Zahlen

Bei der Addition, Subtraktion kann ein Überlauf entstehen (Engl. **Overflow Bit = Überlauf**).

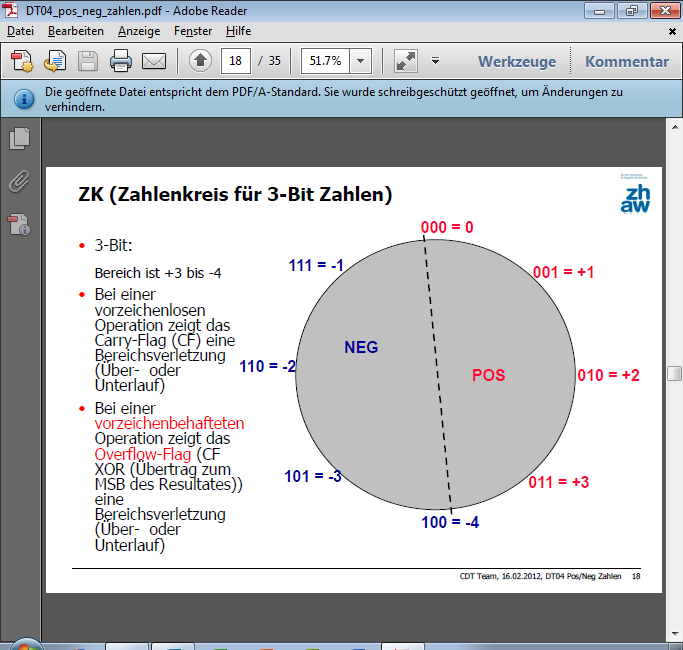
Bei vorzeichenbehafteten Zahlen ist das MSB (=Most Significant Bit, hier das linkste) für die Darstellung des Vorzeichens reserviert:

* 0 = Positive Zahl
* 1 = Negative Zahl

**Einerkomplement = Zahl invertieren.**

**Zweierkomplement = Zahlen invertieren und + 1 addieren. Immer für die Subtraktion verwenden!**

**Die Differenz zweier Zahlen ist die Addition des Zweierkomplements des Subtrahenden zum Minuenden.**



Negative Zahlen = negativer Wert der ersten Stelle (MSB) plus die Summe der restlichen Stellen als positive Werte. So gehen keine Werte verloren (durch doppeltes Null).

# Karunugh-Diagramm

Vereinfachung bis 4 Eingänge (nachher kompliziert). Immer gute 2K-Minterme (=Gruppen) bilden, wo Y=1 ist. Möglichst grosse Gruppen.

# Gray-Code

In drei Schritten zum Gray-Code aus Binärcode:

* X1: Dualzahl im **Binärcode**
* X2: **Rechts-Shift** der Dualzahl um 1 Bit
* X3: Modulo-2-Addition (**XOR-Verknüpfung**) von X1 und X2; dies ist die gewünschte Zahl im Graycode.