1 aus n Code – Wörter von n Länge lauter 0, ausgenommen der Stelle entsprechend dem Zifferwert der darzustellenden Zahl ist 1, von rechts bei 0 angefangen abzuzählen. $7_{10} = 0010000000$

BCD – Binary Coded Decimal ist eine Mischform des binären und hexadezimalen Systems. Dabei wird jede Dezimalstelle durch vier, der binären Darstellung entsprechenden, Binärstellen verwendet und aneinandergehängt. Da jedoch vier binärstellen mehr als nur die 10 Zahlen von 0 bis 9 Darstellen können, werden $1010_2 = 10_{10}$ bis $1111_2 = 15_2$ nicht verwendet. Bei einer Addition wird in Viererpaketen von rechts nach links vorgegangen. Wird dabei der Wert 9 überschritten, wird zusätzlich noch $6_{10} = 0110_2$ dazu addiert. Der Übertrag, welcher die theoretische 5. Stelle füllt, wird dann auf die erste Stelle des nächsten Paketes dazu addiert:

$$1001011001111_{BCD} = 1001_2\ 0110_2\ 0111_2 = 9_{10}\ 6_{10}\ 7_{10} = 967_{10}$$

Binärsystem – Zahlensystem in Basis 2. Beinhaltet Zeichen $\{0,1\}$. Computer arbeiten grundsätzlich in binär, da zwei Zustände durch ***AUS*** und ***AN*** dargestellt werden kann.

Umrechnung zu Dezimalzahl: Der Wert ist Additiv umzurechnen: $11010_2 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 24_{10}$

Umrechnung zu Oktalzahl: Die Binärzahl kann in Dreierpaketen, von rechts ausgehend, geschrieben werden. Ergibt es beim vordersten Paket weniger als 3 Stellen, werden entsprechend 0 vorangestellt. Jedes Paket kann dann einzeln mit der Standardmethode in die Zahlen von 0 bis 7 umgerechnet werden und in der gleichen Reihenfolge wie die Pakete aneinandergereiht werden: $11010101_2 = (0)11_2\ 010_2\ 101_2 = 3_8\ 2_8\ 5_8 = 325_8$

Umrechnung zu Hexadezimalzahl: Analog zur Umrechnung in einen Oktalwert, einfach mit einer Paketgrösse von 4, in die Zahlen $\{0, F\}$: $1101101_2 = 0110_2\ 1101_2 = 6_{16}\ D_{16} = 6D_{16}$.

Dezimalsystem – Zahlensystem in Basis 10, beinhaltet die Zeichen $\{0,1,2,3,4,5,6,7,8,9\}$. Im westlichen Kulturkreis gebräuchliches Zahlensystem.

Umrechnung zu Binärsystem: Die Zahl wird wiederholend durch 2 geteilt, bis die Zahl 0 erreicht wird. Dabei wird jeweils der Rest der Division vorne an der Zahl Binärzahl angehängt.

$$43 \div 2 = 21 \quad Rest{:}\,1 \quad 1$$
$$21 \div 2 = 10 \quad Rest{:}\,1 \quad 11$$
$$10 \div 2 = 5 \quad Rest{:}\,0 \quad 011$$
$$5 \div 2 = 2 \quad Rest{:}\,1 \quad 1011$$
$$2 \div 2 = 1 \quad Rest{:}\,0 \quad 01011$$
$$1 \div 2 = 0 \quad Rest{:}\,1 \quad 101011$$
$$43_{10} = 101011_2$$

Umrechnung zu Oktal- und Hexadezimalsystem: Die Dezimalzahl zu Binärzahl umwandeln und dann zu der entsprechenden Zahl umrechnen, nach der Anleitung in **Binärsystem**.

Einerkomplement – Zur Darstellung von negativen Werten als binäre Zahlen werden alle Stellen einfach invertiert. $-13_{10} = -01101_2 \mathrel{\hat{=}} 10010_2$ Da damit nur $i = \pm 2^{n-1}$ Werte angezeigt werden können und dafür n Stellen benötigt werden, ist die Methode sehr fehleranfällig. Ebenfalls gibt es 2 Darstellungen für 0. Besser ist das **Zweierkomplement**.

Exess binary – Siehe **Exzessdarstellung**

Exzessdarstellung – Bei der Exzessdarstellung wird der Nullpunkt transponiert, von -2^{n-1} bis $2^{n-1} - 1$, n ist die Bitlänge.

Fehler – Bei Datenübertragungen sollte die Korrektheit von Daten gewährleistet werden können. Bei einer fehlerhaften Übertragung kann entweder eine erneute Übertragung vom Sender angefordert werden oder durch den Empfänger korrigiert werden. Das reine Erkennen kann durch ein einfaches **Paritätsbit** erreicht werden, das zweite durch Algorithmen wie den **Hammingcode**. Die auf diesem Blatt beschriebenen Methoden erlauben die Erkennung, respektive Korrektur von 1-Bit-Fehler. Falls mehr als ein Bit fehlerhaft ist, funktioniert es nicht mehr. Methoden, welche mehrere Fehler unterstützen benötigen weitere Paritätsbits.

Fliesskommazahlen – Zahlenformat mit flexibler Kommastelle, Funktion ähnlich der Wissenschaftlichen Schreibweise. Nach Norm **IEEE 754** gibt es 2 verschiedene Genauigkeiten: 32 und 64 Bits. Dabei ist die Bitaufteilung wie folgt: 1 Bit für Vorzeichen: $\mathbf{1 = -}$ $\mathbf{0 = +}$; 8 Bits für Exponenten und 23 Bits für die Mantisse, oder 1 Bit für Vorzeichen: $\mathbf{1 = -}$ $\mathbf{0 = +}$; 11 Bits für Exponenten und 52 Bits für die Mantisse. Der Exponent ist dabei in **Exzessdarstellung**, verschoben um 127, respektive 1023 um die Exponenten von -126 bis 127, respektive -1022 bis 1023 zu unterstützen. Die Berechnung des Exponenten ist wie folgt: $B_{10} = Int(\log_2 Z)$, die Mantisse: $Z \div 2^B = M$. Da die Mantisse über ein verstecktes Bit (hidden Bit) verfügt, wird die die 1 vor dem Komma, respektive die vorderste 1 bei der binären Darstellung weggelassen. $132_{10} = \underbrace{0_{\pm}}_{V}\ \underbrace{7_{10}}_{B}\ \underbrace{1.03125_{10}}_{M} =$

$$0\ 10000110\ (1)01010001111010111000001_2 =$$
$$01000011001010001111010111000001_2 = 4328F5C1_{16}$$

Floating Point Number – Siehe **Fliesskommazahl**

Hamming Abstand – Anzahl Bits die sich von einem zu anderen beliebigen Wert ändern. Desto höher der Abstand, desto höher die Redundanz und Fehlerbehebungspotential.

Hammingcode – Paritätsbits-verfahren welche Fehlerkorrektur erlaubt. Dazu Raster erstellen, bei jeder Stelle, von Links bei 1 beginnend, jede Potenz von Basis 2 als Paritätsbit kennzeichnen. Leere Stellen mit Datenbits ausfüllen. Die Paritätsbits sind **0** falls vom Paritätsbit nach links startend jede Gruppe an n Bits mit jeweils n Bits Lücke zusammen eine gerade Anzahl an **1** vorkommt, sonst eine **1**. Dabei ist n die Potenz der Paritätsbits. Oder man addiert die binären Zeilennummern von Spalten, welche eine 1 enthalten ohne Übertrag. Die resultierende Zahl wird dann in die Paritätsbits eingefüllt.

12	11	10	9	8	7	6	5	4	3	2	1
1	0	1	1	1	0	0	1	0	1	0	1
	0		1		0		1		1		1
	0	1			0	0			1	0	
1					0	0	1	0			
1	0	1	1	1							

$$12_{10} + 10_{10} + 9_{10} + 5_{10} + 3_{10} = \begin{array}{r} 1100 \\ 1010 \\ 1001 \\ 0101 \\ \underline{0011} \\ 1001 \end{array}$$

Der Empfänger zählt alle Zeilen zusammen (ohne Übertrag). Falls das Resultat 0 ist, war die Übertragung fehlerfrei. Falls nicht, ist das Resultat die Zeilennummer mit dem Fehler.

Hexadezimalsystem – Zahlensystem in Basis 16. Beinhaltet Zeichen $\{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$, dabei stehen die Buchstaben A_{16} bis F_{16} für die Zahlenwerte 10_{10} bis 15_{10}. Das System wird, wie auch das Oktalsystem, da stets4 Binärstellen in eine hexadezimalstelle zu überführen sind, zur leserfreundlichen Darstellung von Maschinenausgaben verwendet. Hexadezimal ist dabei weiterverbreitet als Oktal. Zur Kennzeichnung einer hexadezimalen Zahl wird anstelle dem Subskript oft ein *0x* der Zahl vorangestellt: $0x1A5 = 1A5_{16}$

Umrechnung zu Binärzahl: Jede Stelle wird einzeln in ein Viererpacket in Basis 2 umgerechnet und dann in selbiger Reihenfolge aneinandergereiht. Jede Ziffer wird wiederholend durch 2 geteilt bis der Wert 0 erreicht wird. Dabei wird jeweils der Rest der Division vorne an der Zahl Binärzahl angehängt. Falls dabei weniger als 4 Stellen resultieren, sind dementsprechend 0 voranzufügen. [Beispiel: B]

$B \div 2 = 5 \quad Rest: 1 \quad 1$

$5 \div 2 = 2 \quad Rest: 1 \quad 11$

$2 \div 2 = 0 \quad Rest: 0 \quad 011$

$1 \div 2 = 0 \quad Rest: 1 \quad 1011$

$AD_{16} = A_{16}\, D_{16} = 1010_2\, 1101_{16} = 10101101_{16}$

Umrechnung zu Oktalzahl: Die Zahl zuerst in eine Binärzahl umwandeln und dann die unter Binärzahlen beschriebene Methode verwenden.

Umrechnung zu Dezimalzahl: Analog zur Methode beschrieben unter **Zahlensysteme**:

$$AD_{16} = A_{16} \cdot 16^1 + D_{16} \cdot 16^0 = 10_{10} \cdot 16^1 + 13_{10} \cdot 16^0 = 160_{10} + 13_{10} = 173_{10}$$

Logische Operatoren – Operationen für binären Input

AND: Falls beide Inputs **1**, dann Output **1** $I_1 \& I_2 = O$

OR: Falls mindestens ein Input **1**, dann Output **1** $I_1 \mid I_2 = O$

XOR: Falls genau ein Input **1**, dann Output **1** $(I_1 \& \neg I_2)|(\neg I_1 \& I_2) = O$

NOT: Umkehrung des Inputs $\neg I = O$

NAND: Umkehrung des Outputs von **AND**

NOR: Umkehrung des Outputs von **OR**

Offset Binary – Siehe **Exzessdarstellung**

Oktalsystem – **Zahlensystem** in Basis 8. Beinhaltet die Zeichen $\{0,1,2,3,4,5,6,7\}$. Da eine Stelle in Oktal genau mit drei Stellen in Binär übereinstimmt und sich deshalb zur leserfreundlicheren Darstellung von Maschinenausgaben eignet.

Umrechnung zu Binärzahl: Jede Stelle wird einzeln in ein Dreierpacket in Basis 2 umgerechnet und dann in selbiger Reihenfolge aneinandergereiht. Jede Ziffer wird wiederholend durch 2 geteilt bis der Wert 0 erreicht wird. Dabei wird jeweils der Rest der Division vorne an der Zahl Binärzahl angehängt. Falls dabei weniger als 3 Stellen resultieren, sind dementsprechend 0 voranzufügen. [Beispiel: 6]

$6 \div 2 = 3 \quad Rest: 0 \quad 0$

$3 \div 2 = 1 \quad Rest: 1 \quad 10$

$1 \div 2 = 0 \quad Rest: 1 \quad 110$

$573_8 = 5_8\, 7_8\, 3_8 = 101_2\, 111_2\, 011_2 = 101111011_2$

Umrechnung zu Dezimalzahl: Analog der Methode beschrieben unter **Zahlensysteme**:

$$573_8 = 5 \cdot 8^2 + 7 \cdot 8^1 + 3 \cdot 8^0 = 320_{10} + 56_{10} + 3_{10} = 379_{10}$$

Umrechnung zu Hexadezimalzahl: Die Zahl zur Binärzahl umrechnen und dann wie unter Binärsystem beschrieben zur hexadezimalzahl umwandeln.

Paritätsbit – Dem Wort angehängtes Bit zur Fehlererkennung. Erhöhung des **Hamming Abstandes** um 1.

Even Parity: Falls die Anzahl der **1** in Wort gerade, dann wird eine **0** angehängt, sonst eine **1**.

Odd Parity: Falls die Anzahl der **1** in Wort gerade, dann wird eine **1** angehängt, sonst eine **0**.

Vorzeichen – Negative Zahlen in binärer Schreibweise darzustellen ohne ein + oder – zu verwenden, kann gemacht werden, in dem man ein Vorzeichenbit der Zahl voranstellt, welche bei negativen Werten **1** beträgt, bei positiven **0**. Bei dieser Darstellung ist jedoch keine Arithmetik möglich. Daher gibt es das **Einerkomplement.**

Zahlensysteme – Wir beschränken uns auf Stellenwertsysteme, römische Zahlen sind zum Beispiel nicht eine Darstellung in einem Stellenwertsystem. Zahlenwerte können in verschiedenen Basen angegeben werden. In der Informatik (und im Modul) sind meist die folgenden verwendet: **Binärsystem**, **Oktalsystem**, **Dezimalsystem** und **Hexadezimalsystem**. Spezifische Umrechnung unter diesen Punkten. Zahlensysteme sind allgemein so aufgebaut: Der Ziffernwert jeder Stelle muss mit dem Potenzwert von Basis des Zahlensystems und dem Exponenten der abgezählten Position von rechts bei Null beginnend multipliziert werden. Die Basis des Systems wird oft als Subskript der Zahl angehängt.

$$xyz_n = x \cdot n^2 + y \cdot n^1 + z \cdot n^0$$

Diese Berechnungsweise kann auf alle Zahlensysteme aller Basen angewendet werden.

Zweierkomplement – Anders als beim Einerkomplement wird nach der Inversion der negativen Zahl eine 1 dazu addiert. Dies löst das Problem der redundanten 0.