2. Grundlagen der digitalen Schaltungstechnik

Will man Digitalschaltungen z.B. zu Codieren oder in Rechenschaltungen einsetzen, so benötigt man Kenntnisse über die dort benutzten Zahlensysteme und die auf sie anzuwendenden Rechenoperationen.

2.1. Zahlensysteme

Jedes Zahlensystem besteht aus der Basis B und den Ziffern. Die Anzahl der Ziffern ergibt sich aus der Basis. Die größte Ziffer entspricht der Basis minus 1. Wird die größte Ziffer überschritten, entsteht ein Übertrag für den nächsthöheren Stellenwert.

Jede natürliche Zahl Z lässt sich eindeutig in einer Potenzreihe zur Basis B zerlegen:

$Z = a_n.B^n + a_{n-1}.B^{n-1} + ...a_1B^1 + a_0.B^0$

$$Z = \sum_{n=0}^{k} a_n.B^n$$

Das am meisten verwendete Zahlensystem ist das **Dezimalsystem.** Zur Auswahl stehen dazu zehn (10) verschiedene Ziffern, 0 bis 9. Das Dezimalsystem, auch **Zehnersystem** genannt, verwendet daher die **Basis** 10.

z.B.: 347_{10} ist gleich: 3 Hunderter + 4 Zehner + 7 Einer

Was hier etwas an die Schulzeit erinnert, ist eine Betrachtung nach **Stellenwerten**. Mathematischer ausgedrückt:

$347_{10} = 3.10^2 + 4.10^1 + 7.10^0 = 3.100 + 4.10 + 7.1 = 300 + 40 + 7$

Dabei wird jede Ziffer mit ihrem Stellenwert multipliziert. Im oberen Beispiel ist 7 die **niederwertigste** und 3 die **höchstwertigste** Stelle.

Besondere Bedeutung hat in der Informatik und Digitaltechnik das **Binärsystem**. Das Binärsystem, auch **Dualsystem** oder **Zweiersystem** genannt, verwendet die Basis B = 2, d.h. es gibt zwei (2) verschiedene Werte, nämlich Null (0) und Eins (1).

Das duale Zahlensystem ist entstanden, weil man in der elektronischen Datenverarbeitung nur zwei Zustände unterscheiden kann. Die Zustände werden üblicherweise mit 0 und 1 abgekürzt. Weitere gebräuchliche Ausdrucksweisen sind. L/H, low/high, Off/ON..

Die kleinste Informationseinheit (0 oder 1) wird als **Bit** (*binary digit*) bezeichnet. Eine Gruppe von Bits wird in der Informationstechnik als **Wort**, die Anzahl der Bits in einem Wort als **Wortbreite** bezeichnet.

Die am häufigsten verwendete Wortbreite ist 8 (8-Bit-Wort). Das 8-Bit-Wort wird **Byte** (genau genommen Oktett) genannt. (große Bedeutung bei Textdarstellung ASCII-Zeichen), 16-Bit-Wörter hauptsächlich bei Mikrocontroller Anwendungen, 32 (bzw.64) Bit-Wortbreite bei Computern.

Umrechnung vom Binär- ins Dezimalsystem:

Was bedeutet nun etwa die Binärzahl 00111000? Die Vorgangsweise ist genauso wie oben bei Dezimalzahlen vorgehen. Wichtig zu wissen ist auch hier: Ganz links ist die höchstwertigste (MSB Most Significant Bit) und ganz rechts die niederwertigste Stelle (Least significant Bit). Um den Wert in Dezimalform zu erhalten, werden die einzelnen Stellenwerte addiert.

$0 * 2^0 = 0$
$0 * 2^1 = 0$
$0 * 2^2 = 0$
$1 * 2^3 = 8$
$1 * 2^4 = 16$
$1 * 2^5 = 32$
$0 * 2^6 = 0$
$0 * 2^7 = 0$

$= 56$

Die folgende Tabelle soll dieses Prinzip noch einmal veranschaulichen:

128	64	32	16	8	4	2	1
0	0	1	1	1	0	0	0

In der oberen Zeile steht der ausmultiplizierte Stellenwert in Dezimalschreibweise (z.B. $2^5 = 32$), darunter die Werte der einzelnen Stellen aus dem oberen Beispiel. Um zu einem dezimalen Ergebnis zu gelangen, brauchen nur die Stellen addiert werden, die auf Eins gesetzt sind.

Umrechnung vom Dezimal- ins Binärsystem:

Um eine Dezimal z.B.: 90_{10} in eine Dualzahl umzuwandeln gibt es zwei gebräuchliche Verfahren:

Bei der **Subtraktionsmethode** geht man vom größten dualen Stellenwert aus, der in der Dezimalzahl enthalten ist. Um diesen wird die Dezimalzahl verringert. Vom Rest wird nun versucht, den nächstniederen dualen Stellenwert abzuziehen. Immer wenn eine Subtraktion stattfindet, wird eine 1 notiert. Ist der Stellenwert nicht enthalten, wird eine 0 zum Ergebnis geschrieben. Die so erhaltene Abfolge von 1 und 0 ergibt die Dualzahl.

Bei der **Divisionsmethode** wird die Dezimalzahl ganzzahlig fortlaufend durch zwei geteilt und der Rest notiert. Die bei den einzelnen Divisionen gewonnenen Reste ergeben, mit dem zuletzt erhaltenen Rest beginnend die gewünschte Dualzahl.

Rechnen mit Dualzahlen
Addition, Subtraktion, Multiplikation und Division von Dualzahlen erfolgt grundsätzlich nach den gleichen Regeln wie bei den Dezimalzahlen. In der Computertechnik wird die Subtraktion überwiegend durch die Addition des Komplements der abzuziehenden Zahl vorgenommen.

Besonders wichtig ist in der Informatik und Digitaltechnik neben dem Binärsystem auch das **Hexadezimalsystem** (**Sedezimalsystem**). Das Hexadezimalsystem verwendet die Basis 16, d.h. es gibt 16 verschiedene Ziffern, 0 bis 9 und zusätzlich die Buchstaben A bis F .

Mit dem Hexadezimalsystem können auf einfachere und kürzere Weise Binärzahlen notiert werden. Mit einer vierstelligen Binärzahl (auch als **Halbbyte** oder **Nibble** bezeichnet) lassen sich 16

(2^4 = 16) verschiedene Zahlen darstellen, und zwar 0 bis 15 (die Null zählt mit!). Da das Hexadezimalsystem die Basis 16 (= 2^4) verwendet, reicht eine (!) Hexadezimalzahl aus, um vier Bits (Binärziffern) darzustellen. Mit zwei Hexadezimalzahlen kann ein Byte (8 Bits) angeschrieben werden.
Um eindeutig darauf hinzuweisen, dass es sich um eine Hexadezimalzahl handelt, kann ebenso wie in anderen Zahlensystemen die Basis tiefgestellt dazu geschrieben werden, z.B. $3F_{16}$ (= 63_{10} dezimal) oder 93_{16} (= 147_{10} dezimal).
Es sind aber auch andere Schreibweisen üblich:
a) Vorangestelltes **0x** (*Prefix*), z.B. 0x93. Diese Notation wird in Programmiersprachen mit C-ähnlicher-Syntax verwendet.
b) Nachgestelltes **h** (*Postfix*), z.B. 93h. Letztere Schreibweise ist besonders in der Technik gebräuchlich.

Umrechnung vom Dezimal- ins Hexadezimalsystem:

Die Umrechnung funktioniert ähnlich der Umrechnung von Dezimal- zu Binärzahlen. Nun muss aber, statt durch 2, durch 16 dividiert werden. Die Reste werden genauso von rechts nach links angeschrieben und geben, wenn das Ergebnis der Ganzzahlendivision 0 ist, das Endergebnis.

Beispiel: Die Dezimalzahl 304 soll in eine Hexadezimalzahl umgewandelt werden.

1. 304 dividiert durch 16, gibt 19, **kein Rest**, dh. **0** (Null) anschreiben.
2. 19 dividiert durch 16, gibt 1, **3 Rest**, dh. **3** anschreiben.
3. 1 dividiert durch 16, gibt 0, **1 Rest**, dh. **1** anschreiben.

Endergebnis: 130_{16}, das entspricht der Dezimalzahl 304_{10}.

Umrechnung vom Hexadezimal- ins Dezimalsystem:

Die Umrechnung vom Hexadezimal- ins Dezimalsystem kann genauso wie oben von Binär nach Dezimal demonstriert, erfolgen. Die einzelnen Ziffern werden mit dem jeweiligen Stellenwert (16^n, wobei n = 0, 1, 2, ...) multipliziert und die jeweiligen Ergebnisse aufsummiert. Das folgende Beispiel demonstriert dies anhand der Hexadezimalzahl 130_{16}:

$$
\begin{aligned}
0 * 16^0 &= 0 \\
3 * 16^1 &= 48 \\
1 * 16^2 &= 256 \\
\hline
&= 304
\end{aligned}
$$

Diese Antwort hätte in der Praxis natürlich auch ein wissenschaftlicher Taschenrechner geliefert. Es reicht dazu sogar der **Windows-Rechner** (den Sie nur auf die wissenschaftliche Ansicht umstellen müssen) oder unter Linux Programme wie z.B. **KCalc**.

Dezimalsystem			**Dualsystem**								**Hexdezimalsystem**		
10^2	10^1	10^0	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	16^2	16^1	16^0
		0	0	0	0	0	0	0	0	0	0	0	0
		1	0	0	0	0	0	0	0	1	0	0	1
		2	0	0	0	0	0	0	1	0	0	0	2
		3	0	0	0	0	0	0	1	1	0	0	3
		4	0	0	0	0	0	1	0	0	0	0	4
		5	0	0	0	0	0	1	0	1	0	0	5
		6	0	0	0	0	0	1	1	0	0	0	6
		7	0	0	0	0	0	1	1	1	0	0	7
		8	0	0	0	0	1	0	0	0	0	0	8
		9	0	0	0	0	1	0	0	1	0	0	9
	1	0	0	0	0	0	1	0	1	0	0	0	A
	1	1	0	0	0	0	1	0	1	1	0	0	B
	1	2	0	0	0	0	1	1	0	0	0	0	C
	1	3	0	0	0	0	1	1	0	1	0	0	D
	1	4	0	0	0	0	1	1	1	0	0	0	E
	1	5	0	0	0	0	1	1	1	1	0	0	F
	1	6	0	0	0	1	0	0	0	0	0	1	0
	1	7	0	0	0	1	0	0	0	1	0	1	1
	.	.	.	.	.	.	.	.	.	.	.	.	.
	4	7	0	0	1	0	1	1	1	1	0	2	F
	.	.	.	.	.	.	.	.	.	.	.	.	.
	8	2	0	1	0	1	0	0	1	0	0	5	2
	.	.	.	.	.	.	.	.	.	.	.	.	.
	9	9	0	1	1	0	0	0	1	1	0	6	3
1	0	0	0	1	1	0	0	1	0	0	0	6	4