Vorwort: Die folgenden Ausführungen befassen sich mit einem mathematischen Thema, das aber nicht wie im Mathe-Buch, sondern verständlich erläutert vorgestellt wird. Da der Verfasser aus der Elektrotechnik kommt, hat er zum Teil andere Sichtweisen auf die Dinge. Vielleicht hilft gerade diese Besonderheit beim Verständnis der beschriebenen Mathematik.

Achtung: Trotz aller Sorgfalt bei der Erarbeitung kann keine Garantie für die Richtigkeit der Inhalte übernommen werden.

Eine Bemerkung zuvor: Zahlensysteme – abgesehen von solchen Varianten wie den komplexen Zahlen – funktionieren letztlich alle gleich. Deshalb greife ich zu Beginn der Erläuterungen immer wieder auf die Dezimalzahlen zurück. Wenn man sich deren Funktion verdeutlicht, hat man in der Regel auch schon das andere/neue Zahlenformat verstanden. Probieren wir es aus …

Inhaltsübersicht binäre bzw. duale Zahlen

Grundlagen der binären bzw. dualen Zahlen

Zum Verständnis vorab die Grundregeln der Dezimalzahlen

1. Jede Dezimal*ziffer* kann 10 verschiedene Zustände annehmen:

   0, 1, 2, 3, ... , 9

2. Die Basisdes Dezimalzahlensystems ist entsprechend die Zahl 10.

3. Die höchstwertigste Ziffer ist die erste Ziffer der Dezimalzahl. Die niederwertigste Ziffer der Dezimalzahl ist die letzte Ziffer.

4. Die Wertigkeit einer zum Beispiel fünfstelligen und ganzzahligen Dezimalzahl VWXYZ berechnet sich wie folgt:

Dezimalzahl:	**V**	**W**	**X**	**Y**	**Z**
Ziffernnummer:	4	3	2	1	0

   Wertigkeit: $V \cdot 10^4 + W \cdot 10^3 + X \cdot 10^2 + Y \cdot 10^1 + Z \cdot 10^0$

5. Handelt es sich um eine rationale Dezimalzahl V,WXYZ mit Kommastellen, sieht das Ziffernschema zum Beispiel wie folgt aus:

Dezimalzahl:	**V,**	**W**	**X**	**Y**	**Z**
Ziffernnummer:	0	-1	-2	-3	-4

   Wertigkeit: $V \cdot 10^0 + W \cdot 10^{-1} + X \cdot 10^{-2} + Y \cdot 10^{-3} + Z \cdot 10^{-4}$
   $= V \cdot 1 + W \cdot 0{,}1 + X \cdot 0{,}01 + Y \cdot 0{,}001 + Z \cdot 0{,}0001$

Nun die Grundregeln der Binär- bzw. Dualzahlen

1. Jede Binär*ziffer* kann 2 verschiedene Zustände annehmen:

   0 und 1

2. Die Basis des Binär- bzw. Dualzahlensystems ist entsprechend die Zahl 2.

3. Die Wertigkeit einer zum Beispiel achtstelligen und ganzzahligen Binärzahl IJKLMNOP berechnet sich wie folgt:

Binär- bzw. Dualzahl:	**I**	**J**	**K**	**L**	**M**	**N**	**O**	**P**
Ziffernnummer:	7	6	5	4	3	2	1	0

   Wertigkeit: $I \cdot 2^7 + J \cdot 2^6 + K \cdot 2^5 + L \cdot 2^4 + M \cdot 2^3 + N \cdot 2^2 + O \cdot 2^1 + P \cdot 2^0$
   $= I \cdot 128 + J \cdot 64 + K \cdot 32 + L \cdot 16 + M \cdot 8 + N \cdot 4 + O \cdot 2 + P \cdot 1$

4. Die höchstwertigste Ziffer – im Beispiel gekennzeichnet durch den Buchstaben I – bezeichnet man als **MSB = most significant bit**. Die niederwertigste Ziffer – im Beispiel gekennzeichnet durch den Buchstaben P – bezeichnet man als **LSB = last significant bit**.

5. Über die dargestellte Summation der Wertigkeiten der einzelnen Binär- bzw. Dualziffern kann man eine beliebige Binär- bzw. Dualzahl in eine Dezimalzahl umrechnen:

Binär- bzw. Dualzahl:	**0**	**1**	**1**	**0**	**1**	**0**	**1**	**0**
Ziffernnummer:	7	6	5	4	3	2	1	0

Wertigkeit:
$$0\cdot 2^7+1\cdot 2^6+1\cdot 2^5+0\cdot 2^4+1\cdot 2^3+0\cdot 2^2+1\cdot 2^1+0\cdot 2^0$$
$$=0\cdot 128+1\cdot 64+1\cdot 32+0\cdot 16+1\cdot 8+0\cdot 4+1\cdot 2+0\cdot 1$$
$$=106_{dezimal}$$

6. Achtung! Bei Bedarf kann die Binär- bzw. Dualzahl auch eine „Kommastelle" bekommen. Das passiert, wenn man die Wertigkeit der einzelnen Binärziffern ganz einfach in den negativen Bereich ausdehnt:

Binär- bzw. Dualzahl:	**I**	**J**	**K**	**L,**	**M**	**N**	**O**	**P**
Ziffernnummer:	3	2	1	0	-1	-2	-3	-4

Wertigkeit:
$$I\cdot 2^3+J\cdot 2^2+K\cdot 2^1+L\cdot 2^0+M\cdot 2^{-1}+N\cdot 2^{-2}+O\cdot 2^{-3}+P\cdot 2^{-4}$$
$$=I\cdot 8+J\cdot 4+K\cdot 2+L\cdot 1+M\cdot 0{,}5+N\cdot 0{,}25+O\cdot 0{,}125+P\cdot 0{,}0625$$

Ein Beispiel für eine Binärzahl mit „Kommastelle":

Binär- bzw. Dualzahl:	**0**	**1**	**1**	**0,**	**1**	**0**	**1**	**0**
Ziffernnummer:	3	2	1	0	-1	-2	-3	-4

Wertigkeit:
$$0\cdot 2^3+1\cdot 2^2+1\cdot 2^1+0\cdot 2^0+1\cdot 2^{-1}+0\cdot 2^{-2}+1\cdot 2^{-3}+0\cdot 2^{-4}$$
$$=0\cdot 8+1\cdot 4+1\cdot 2+0\cdot 1+1\cdot 0{,}5+0\cdot 0{,}25+1\cdot 0{,}125+0\cdot 0{,}0625$$
$$=6{,}625_{dezimal}$$

Umrechnung von Dezimalzahlen in Binär- bzw. Dualzahlen

Variante 1: Manuelles Zerlegen der Dezimalzahl in eine Summe einzelner Zweierpotenzen

Beispiel 1:

$$47_{dezimal}=\mathbf{32}+15$$
$$15=\mathbf{8}+7$$
$$7=\mathbf{4}+3$$
$$3=\mathbf{2}+\mathbf{1}$$

$$47_{dezimal}=32+8+4+2+1$$

$$47_{dezimal}=0\cdot 2^7+0\cdot 2^6+1\cdot 2^5+0\cdot 2^4+1\cdot 2^3+1\cdot 2^2+1\cdot 2^1+1\cdot 2^0$$
$$47_{dezimal}=00101111_{binär}$$

Beispiel 2:

$$229_{dezimal}=\mathbf{128}+101$$
$$101=\mathbf{64}+37$$
$$37=\mathbf{32}+5$$
$$5=\mathbf{4}+\mathbf{1}$$

$$229_{dezimal}=128+64+32+4+1$$

$$229_{dezimal}=1\cdot 2^7+1\cdot 2^6+1\cdot 2^5+0\cdot 2^4+0\cdot 2^3+1\cdot 2^2+0\cdot 2^1+1\cdot 2^0$$

$229_{dezimal} = 11100101_{binär}$

Variante 2: Wiederholte Division durch 2 mit Festhalten des Divisionsrestes

Beispiel 1:

$47_{dezimal}: 2 = 23$	Rest 1	Der Rest der ersten Division durch 2 entspricht dem LSB der entstehenden Binärzahl.
$23: 2 = 11$	Rest 1	
$11: 2 = 5$	Rest 1	
$5: 2 = 2$	Rest 1	
$2: 2 = 1$	Rest 0	
$1: 2 = 0$	Rest 1	Der Rest der letzten Division durch 2 entspricht dem MSB der entstehenden Binärzahl.

$47_{dezimal} = 101111_{binär}$

Typischer Weise wird die Stellenzahl des Ergebnisses auf Vielfache von 4 oder 8 aufgefüllt, ohne dass sich der Wert des Ergebnisses ändert – also mit Nullen, sofern es sich um eine positive Binärzahl handelt.

$47_{dezimal} = 00101111_{binär}$

Beispiel 2:

$229_{dezimal}: 2 = 114$	Rest 1	Der Rest der ersten Division durch 2 entspricht dem LSB der entstehenden Binärzahl.
$114: 2 = 57$	Rest 0	
$57: 2 = 28$	Rest 1	
$28: 2 = 14$	Rest 0	
$14: 2 = 7$	Rest 0	
$7: 2 = 3$	Rest 1	
$3: 2 = 1$	Rest 1	
$1: 2 = 0$	Rest 1	Der Rest der letzten Division durch 2 entspricht dem MSB der entstehenden Binärzahl.

$229_{dezimal} = 11100101_{binär}$

Typischer Weise wird die Stellenzahl des Ergebnisses auf Vielfache von 4 oder 8 aufgefüllt, ohne dass sich der Wert des Ergebnisses ändert – also mit Nullen, sofern es sich um eine positive Binärzahl handelt. Das ist bei diesem Ergebnis nicht erforderlich.

Zahlenbereiche von Binär- bzw. Dualzahlen ohne Vorzeichen

- 4 Bit
  größte positive Binär- bzw. Dualzahl: $1111_{binär} = 15_{dezimal}$
  kleinste positive Binär- bzw. Dualzahl: $0000_{binär} = 0_{dezimal}$

- 8 Bit
  größte positive Binär- bzw. Dualzahl: $1111\ 1111_{binär} = 255_{dezimal}$
  kleinste positive Binär- bzw. Dualzahl: $0000\ 0000_{binär} = 0_{dezimal}$

- 16 Bit
  größte positive Binär- bzw. Dualzahl: $1111\ 1111\ 1111\ 1111_{binär} = 65535_{dezimal}$
  kleinste positive Binär- bzw. Dualzahl: $0000\ 0000\ 0000\ 0000_{binär} = 0_{dezimal}$

- …

Zahlenbereiche von Binär- bzw. Dualzahlen mit Vorzeichen

Für Binär- bzw. Dualzahlen gibt es zwei verschiedene Betrachtungsweisen:

1. Die Binär- bzw. Dualzahlen sind positive Zahlen ohne Vorzeichen – so, wie die Zahlen bis jetzt vorgestellt wurden:

   $00101111_{binär} = 47_{dezimal}$
   $11100101_{binär} = 229_{dezimal}$

2. Oder aber die Binärzahlen werden als vorzeichenbehaftete Binärzahlen betrachtet: Das höchstwertige Bit (MSB = most significant bit) wird zum **Vorzeichenbit**. 0 heißt positives Vorzeichen, 1 heißt negatives Vorzeichen:

   $\mathbf{0}0101111_{binär} > 0$
   $\mathbf{1}1100101_{binär} < 0$

Welche Betrachtungsweise benutzt wird, hängt von der jeweiligen Problemstellung ab. Treten nur positive Zahlenwerte auf, kann auf die Vorzeicheninformation verzichtet werden. Sind negative Zahlen nicht ausschließbar, muss mit vorzeichenbehafteten Binärzahlen gearbeitet werden. In der Prozessor-Programmierung hat man die Wahl zwischen Befehlen, die Zahlenwerte immer als positive Größe auffassen, und anderen Befehlen, die Zahlenwerte vorzeichenrichtig verarbeiten.

Problem: Der Wert einer negativen Binärzahl lässt sich nicht wie bei positiven Binärzahlen ermitteln. Mit Hilfe der **Zweierkomplementbildung** wird eine negative Binärzahl in die zugehörige positive Binärzahl umgewandelt (d. h. das Vorzeichen wird gewechselt). Von der entstandenen positiven Binärzahl kann man dann wie bisher den Wert errechnen. Damit ist bekannt, welchen Wert die negative Binärzahl hat.

Schritte bei der Bildung des Zweierkomplements sind:

1. Binärzahl invertieren = negieren, d. h. aus 0 wird 1 und aus 1 wird 0
2. Addition von 1 auf die letzte Stelle der negierten Binärzahl

Ausgangszahl: $\mathbf{1}1100101_{binär} = ???$

Frage: Welchen (Dezimal-)Wert hat die negative Binärzahl $11100101_{binär}$?

Negation: $00011010_{binär}$
Addition von 1: $\mathbf{0}0011011_{binär} = 27$

Antwort: Die negative Binärzahl $\mathbf{1}1100101_{binär}$ hat den (Dezimal-)Wert -27.

Ausgangszahl: $\mathbf{0}0101111_{binär} = +47_{dezimal}$

Frage: Welche Binärzahl gehört zur (Dezimal-)Zahl $-47_{dezimal}$?

Negation: $11010000_{binär}$
Addition von 1: $\mathbf{1}1010001_{binär} = -47_{dezimal}$

Antwort: Die zu $-47_{dezimal}$ gehörende Binärzahl lautet $\mathbf{1}1010001_{binär}$.

Es ergibt sich folgendes Bild für vorzeichenbehaftete Binär- bzw. Dualzahlen (beispielhaft für 8 Bit dargestellt):

$$\begin{aligned}
+127_{dezimal} &= 0111\ 1111_{binär}\\
&\ \ .\\
&\ \ .\\
&\ \ .\\
+4_{dezimal} &= 0000\ 0100_{binär}\\
+3_{dezimal} &= 0000\ 0011_{binär}\\
+2_{dezimal} &= 0000\ 0010_{binär}\\
+1_{dezimal} &= 0000\ 0001_{binär}\\
0_{dezimal} &= 0000\ 0000_{binär}\\
-1_{dezimal} &= 1111\ 1111_{binär}\\
-2_{dezimal} &= 1111\ 1110_{binär}\\
-3_{dezimal} &= 1111\ 1101_{binär}\\
-4_{dezimal} &= 1111\ 1100_{binär}\\
&\ \ .\\
&\ \ .\\
&\ \ .\\
-128_{dezimal} &= 1000\ 0000_{binär}
\end{aligned}$$

Was sagt uns das? Ganz einfach: **Positive** Binär- bzw. Dualzahlen kann man bei Bedarf beliebig mit Nullen auf 8 oder 16 oder 32 oder X Bit auffüllen. Am Wert der aufgefüllten Zahl ändert sich dabei nichts.

Handelt es sich um eine **negative** Binär- bzw. Dualzahl – sprich das erste Bit = Vorzeichenbit ist gleich 1, dann muss man mit Einsen auffüllen, damit das Vorzeichenbit auch nach dem Auffüllen noch gleich 1 ist. Am Wert der negativen aufgefüllten Zahl ändert sich dabei auch nichts. Klingt komisch, funktioniert aber so.

Gut nachvollziehen kann man das an $-1_{dezimal} = 1111\ 1111_{binär}$. Ob man da 8 oder 16 oder 32 oder X binäre Einsen dastehen hat – sie entsprechen immer dem Wert $-1_{dezimal}$.

- 4 Bit
  größte positive Binär- bzw. Dualzahl: $\mathbf{0}111_{binär} = +7_{dezimal}$
  kleinste negative Binär- bzw. Dualzahl: $\mathbf{1}000_{binär} = -8_{dezimal}$

- 8 Bit
  größte positive Binär- bzw. Dualzahl: $\mathbf{0}111\ 1111_{binär} = +127_{dezimal}$
  kleinste negative Binär- bzw. Dualzahl: $\mathbf{1}000\ 0000_{binär} = -128_{dezimal}$

- 16 Bit
  größte positive Binär- bzw. Dualzahl: $\mathbf{0}111\ 1111\ 1111\ 1111_{binär} = +32767_{dezimal}$
  kleinste negative Binär- bzw. Dualzahl: $\mathbf{1}000\ 0000\ 0000\ 0000_{binär} = -32768_{dezimal}$

- …

Rechnen mit Binär- bzw. Dualzahlen

Wer das aus welchen Gründen auch immer unbedingt sehen möchte – ich habe es mal beispielhaft vorgenommen.

Addition ohne Vorzeichen (das Rechenprinzip entspricht der schriftlichen Addition von Dezimalzahlen):

```
Binärzahlen              Dezimalzahlen

  0110 1010 =             106
+ 0011 0101 =            + 53
-----------------------------
  1001 1111 =             159

  0111 0011 =             115
+ 0001 1111 =            + 31
-----------------------------
  1001 0010 =             146
```

Addition mit Vorzeichen (das Rechenprinzip entspricht der schriftlichen Addition von Dezimalzahlen):

```
Binärzahlen              Dezimalzahlen

  0110 1010 =             106
+ 1010 0110 =          + (-90)
-------------------------------
  0001 0000 =              16

  0011 0110 =              54
+ 1000 1100 =          +(-116)
-------------------------------
  1100 0010 =             -62
```

Subtraktion ohne Vorzeichen (das Rechenprinzip entspricht der schriftlichen Subtraktion von Dezimalzahlen):

```
Binärzahlen              Dezimalzahlen

  0110 1010 =             106
- 0011 0101 =            - 53
-----------------------------
  0011 0101 =              53

  0111 0011 =             115
- 0001 1111 =            - 31
-----------------------------
  0101 0100 =              84
```

Subtraktion mit Vorzeichen (das Rechenprinzip entspricht der schriftlichen Subtraktion von Dezimalzahlen):

```
Binärzahlen              Dezimalzahlen

  0001 1010 =              26
- 1010 0110 =          - (-90)
-------------------------------
  0111 0100 =             116

  1001 1011 =           (-101)
- 1000 1100 =          -(-116)
-------------------------------
  0000 1111 =              15
```

Multiplikation (das Rechenprinzip entspricht der schriftlichen Multiplikation von Dezimalzahlen):

Binärzahlen

```
10110111 * 01110010 =
           0
+     10110111
+      10110111
+       10110111
+               0
+                0
+          10110111
+                  0
    1000100101
+       110011011110
    101000101111110
```

Dezimalzahlen

```
183  * 114
    183
+    183
+     732
    20862
```

Division ohne Kommastelle (das Rechenprinzip entspricht der schriftlichen Division von Dezimalzahlen):

Dezimal:

```
 20862              : 114        = 183
-114
 0946
 -912
  0342
  -342
     0
```

Binär:

```
 101000101111110 : 01110010 = 10110111
-01110010
 0011000011
  -01110010
   010100011
   -01110010
    0011000111
     -01110010
      010101011
      -01110010
       001110010
       -01110010
               0
```

Division mit Kommastelle (das Rechenprinzip entspricht der schriftlichen Division von Dezimalzahlen):

Dezimal:

```
 15    : 4    = 3,75
-12
  30
 -28
   20
  -20
    0
```

Binär: $1111 : 100 = 11{,}11 = 1*2^1+1*2^0+1*2^{-1}+1*2^{-2} = 3{,}75$

$$\begin{array}{r}
1111 \\
\underline{-100} \\
0111 \\
\underline{-100} \\
0110 \\
\underline{-100} \\
0100 \\
\underline{-100} \\
0
\end{array}$$