2.2.1 Operatoren

Das Gleichheitszeichen ist in C ein Zuweisungsoperator, d.h. man weist damit einer Variablen einen Wert zu.

- Zuweisung eines bestimmten Zahlenwertes beim Programmstart ("Initialisierung")

```
char c = 5;   // in der Variablen c wird der Startwert 5 gespeichert
```

- Hochzählen einer Zahl

```
c = c + 5;    // c wird um 5 erhöht
```

- Zuweisung eines Rechenergebnisses

```
c = a / b;    // Das Ergebnis von a durch b wird der Variablen c zugewiesen
```

Arithmetisch Rechenoperatoren

Operator	Beschreibung	Beispiele
+	Addition	x = 34 + 45; x = x + 3;
+=	Addition mit Zuweisung	x += 3; wie x = x + 3
++	Inkrement	x++; wie x = x + 1;
-	Subtraktion	x = 45 – 34; x = x – 3;
-	Multiplikation mit -1	x = -x;
-=	Subtraktion mit Zuweisung	x -= 3; wie x = x - 3;
--	Dekrement	x --; wie x = x - 1;
*	Multiplikation	x = 3 * 5; x = x * 5;
*=	Multiplikation mit Zuweisung	x *=5; wie x = x * 5;
/	Division	x = 5 / 7; x = x / 7;
/=	Division mit Zuweisung	x /= 7; wie x = x / 7;
%	Modulo Division für ganze Zahlen, ergibt den Rest der Division	x = 14 % 4; // Ergebnis 2

Vergleichsoperatoren

Das Ergebnis von Vergleichen ist ein Wahrheitswert, also **wahr** oder **falsch**. Der Wert „falsch“ wird in C mit dem Zahlenwert 0 dargestellt. Der Wert „wahr“ mit einem Zahlenwert ≠ 0. Vergleichsoperatoren dienen hauptsächlich zur Formulierung von Fallunterscheidungen bei Programmverzweigungen und Abbruchbedingungen von Programmschleifen.

Operator	Beschreibung	Beispiele
>	größer als	while (x > 5)
>=	größer gleich	if (x >=0)
<	kleiner als	while (a < b)
<=	kleiner gleich	while (y <= 4.6)
==	gleich	if (a == 0)
!=	ungleich	if (a !=0)

Logische Operatoren

Müssen komplexere Fallunterscheidungen getroffen werden, können Wahrheitswerte durch logische Operatoren verknüpft werden. Das Ergebnis ist wieder ein Wahrheitswert.

Operator	Beschreibung	Beispiele
&&	logisches UND	if ((a > 5) && (Taster == 1))
||	logisches ODER	while ((a ==0) || (Taster == 0))
!	logisches NICHT	if (!Taster)

Bitweise Operatoren

Bitweise Verknüpfungen von Variablen, Maskierung, Verschieben

Operator	Beschreibung	Beispiele

&	bitweise logisches UND	x = 0x03 & 0x06; // Ergebnis: 0x02
		0000 0011 ≙ 0x03
		0000 0110 ≙ 0x06
		0000 0010 ≙ 0x02

|	bitweise logisches ODER	x = a | 0x0F; // niederwertige 4 Bits eins setzen

^	bitweises EXOR	x = a ^ 0x0F // für a=0x55 wird x=0x5A
		EXOR mit 0 lässt das Bit unverändert, EXOR mit1 invertiert das Bit.

>>	nach rechts schieben	x = x >> 1; // um 1 Bit nach rechts schieben
		// um 1 Bit nach rechts schieben
		Bei unsigned wird in die oberste Bitstelle eine 0 hinein geschoben, bei signed wird eine 1 (Vorzeichenerweiterung!)

<<	nach links schieben	x = x << 2; // um 2 Bits nach links schieben

~	bitweise Negation	P2_0 = ~P2_0; // Alle Portbits invertieren
		x = ~x; // alle Bits invertieren

Prioritäten

Bei der Auswertung von Rechenausdrücke oder Abfragen mit mehreren Operatoren ist die Operatorreihenfolge zu beachten.

Priorität	Operatorvorrang (von links nach rechts)
1	() []
2	! ~ ++ --
3	* / %
4	+ -
5	<< >>
6	< <= > >=
7	== !=
8	& ^ |
9	&& ||
10	=
	+= -= *= /= %= &= ^= |= <<= >>=

Bsp.: Dargestellt ist die Reihenfolge der Abarbeitung einer komplexen if-Abfrage.