Inhaltsverzeichnis

1
Arithmetik

Die meisten Rechner arbeiten digital (von lat. *digitus* = der Finger). Das heißt, dass eine Speicherzelle eine endliche Anzahl von Zuständen annehmen kann. In der Praxis ist eine Speicherzelle **binär**, das heißt, dass sie nur zwei Zustände annehmen kann: Spannung vorhanden, Spannung nicht vorhanden; 0 oder 1. In diesem Kapitel besprechen wir, wie man ganze Zahlen mithilfe von **Bits** repräsentiert und wie man die gewohnten Rechenoperationen (Addition, Subtraktion) durch Bitoperationen ausdrückt.

Wir diskutieren zuerst, wie man mittels **Stellenwertsystemen** die natürlichen Zahlen durch Folgen von Ziffern (aus einem endlichen Ziffernvorrat) darstellt. Im täglichen Leben verwenden wir zehn Ziffern, der Rechner verwendet zwei. Beim Programmieren ist es manchmal praktisch, sechzehn oder nur acht Ziffern zu verwenden. Eine binäre Ziffer nennt man **Bit**. Mit einer Folge von n Bits kann man dann 2^n Zahlen darstellen. Aus praktischen Gründen rechnet ein Prozessor immer mit Bitfolgen fest vorgegebener Länge. Dies rührt daher, dass die Schaltkreise, die die Bitfolgen verarbeiten eine feste Größe haben und sich nicht ändern lassen. Daher müssen wir verstehen, wie wir die gewohnten arithmetischen Operationen sinnvoll auf diese Zahlen fester Länge anpassen. Dies erreichen wir durch die **Moduloarithmetik**.

1.1 Stellenwertsysteme

Zahlen schreibt man gewöhnlich als Folge von **Ziffern**. Dadurch, dass man nur einen endlichen Vorrat von Ziffern hat, muss man Ziffern zu Folgen zusammensetzen, um jede natürliche Zahl darstellen zu können. Da es auf die Gestalt der Ziffer nicht ankommt, sondern allein entscheidend ist, dass man nur endlich viele Ziffern hat, wollen wir im folgenden die Menge der Ziffern mit der Menge $\{0, \ldots, \beta - 1\}$ gleichsetzen, wobei $\beta > 0$ die Anzahl der vorhandenen Ziffern ist und **Basis** des Stellen-