CHAPTER 4

The Fundamental Operations

> Thus calculation can be seen as the basis and foundation of all the arts.
>
> —Adam Ries, *Book of Calculation*

> And you, poor creature, you are completely useless. Look at me. Everyone needs me.
>
> —Aesop, "The Fir and the Blackberry Bush"

> There is one small prerequisite for mastering the mathemagic tricks in this chapter—you need to know the multiplication tables through 10 . . . backward and forward.
>
> —Arthur Benjamin, Michael B. Shermer, *Mathemagics*

THE FUNDAMENTAL BUILDING BLOCKS OF any software package for computer arithmetic are the functions that carry out the basic operations of addition, subtraction, multiplication, and division. The efficiency of the entire package hangs on the last two of these, and for that reason great care must be taken in the selection and implementation of the associated algorithms. Fortunately, volume 2 of Donald Knuth's classic *The Art of Computer Programming* contains most of what we need for this portion of the FLINT/C functions.

In anticipation of their representation to come, the functions developed in the following sections use the operations `cpy_l()`, which copies one `CLINT` object to another in the sense of an allocation, and `cmp_l()`, which makes a comparison of the sizes of two `CLINT` values. For a more precise description see Section 7.4 and Chapter 8.

Let us mention at this point that for the sake of clarity, in this chapter the functions for the fundamental arithmetic operations are developed all of a piece, while in Chapter 5 it will prove practical to split some of the functions into their respective "core" operations and from there develop additional steps such as the elimination of leading zeros and the handling of overflow and underflow, where, however, the syntax and semantics of the functions are kept intact. For an understanding of the relations described in this chapter this is irrelevant, so that for now we can forget about these more difficult issues.

4.1 Addition and Subtraction

> This notion of "further counting" means, "to the integer n_1 add the integer n_2," and the integer s at which one arrives by this further counting is called "the result of addition" or the "sum of n_1 and n_2" and is represented by $n_1 + n_2$.
>
> —Leopold Kronecker, *On the Idea of Number*

Since addition and subtraction are in principle the same operation with differing signs, the underlying algorithms are equivalent, and we can deal with them together in this section. We consider operands a and b with representations

$$a := (a_{m-1}a_{m-2}\dots a_0)_B = \sum_{i=0}^{m-1} a_i B^i, \qquad 0 \le a_i < B,$$

$$b := (b_{n-1}b_{n-2}\dots b_0)_B = \sum_{i=0}^{n-1} b_i B^i, \qquad 0 \le b_i < B,$$

where we assume $a \ge b$. For addition this condition represents no restriction, since it can always be achieved by interchanging the two summands. For subtraction it means that the difference is positive or zero and therefore can be represented as a `CLINT` object without reduction modulo $(N_{\max} + 1)$.

Addition consists essentially of the following steps.

Algorithm for the addition $a + b$

1. Set $i \leftarrow 0$ and $c \leftarrow 0$.
2. Set $t \leftarrow a_i + b_i + c$, $s_i \leftarrow t \bmod B$, and $c \leftarrow \lfloor t/B \rfloor$.
3. Set $i \leftarrow i + 1$; if $i \le n - 1$, go to step 2.
4. Set $t \leftarrow a_i + c$, $s_i \leftarrow t \bmod B$, and $c \leftarrow \lfloor t/B \rfloor$.
5. Set $i \leftarrow i + 1$; if $i \le m - 1$, go to step 4.
6. Set $s_m \leftarrow c$.
7. Output $s = (s_m s_{m-1} \dots s_0)_B$.

The digits of the summands, together with the carry, are added in step 2, with the less-significant part stored as a digit of the sum, while the more-significant part is carried to the next digit. If the most-significant digit of one of the summands is reached, then in step 4 any remaining digits of the other summand are added to any remaining carries one after the other. Until the last summand digit is processed, the less-significant part is stored as a digit of the sum, and the more-significant part is used as a carry to the next digit. Finally, if there is a