the LSB of the divisor be denoted X. We will assume that there are n bits in the divisor and $2n$ bits in the dividend. Let $i = 0$.

2. Compare X and Y. If $X \geq Y$, the quotient bit is 1: perform $X - Y$. If $X < Y$, the quotient bit is 0.
3. Set $i = i + 1$. If $i \geq n$, stop. Otherwise, shift Y one bit to the right and go to step 2.

For the purposes of illustration, this procedure assumed the division of integers. If the divisor is greater than the dividend, the quotient is 0, and if the divisor is 0, the procedure should be stopped since dividing by 0 results in an error.

As we can see from these examples, multiplication and division operations can be reduced to repeated shift and addition (or subtraction). If the hardware can perform shift, add, and subtract operations, it can be programmed to perform multiplication and division as well. Older digital systems used such measures to reduce hardware costs. With the advances in digital hardware technology, it is now possible to implement these and more complex operations in an economical manner. (See the references at the end of this chapter for books that describe procedures for multiplication and division that are more elegant than the ones described here.)

Shifting Generally, shifting a base r number left by one position (and inserting a 0 into the vacant LSD position) is equivalent to multiplying the number by r. Shifting the number right by one position (inserting a 0 into the vacant MSD position) generally is equivalent to dividing the number by r.

In binary system, each left shift multiplies the number by 2, and each right shift divides the number by 2, as shown in Example 1.21.

Example 1.21

	Binary	Decimal
N	01011.11	$11\frac{3}{4}$
2 * N	10111.1 $\boxed{0}$ ↙Insert	$23\frac{1}{2}$
N ÷ 2	Insert↘ $\boxed{0}$0101.11 $\boxed{1}$ ↖Discard	$5\frac{3}{4}$ (Inaccurate, since only two-bit accuracy is retained.)

If the MSB of an n-bit number is not 0, shifting it left would result in a number larger than the magnitude that can be accommodated in n bits—and the 1 shifted out of the MSB position cannot be discarded. If nonzero bits shifted out of the LSB position during a right shift are discarded, the accuracy is lost. Later in this chapter, we will discuss shifting in further detail.

1.4.2 Octal Arithmetic

Table 1.4 shows the octal addition and multiplication tables. The examples that follow illustrate the four arithmetic operations in octal and their similarity to decimal arithmetic. (Table 1.4 can be used to look up the result at each stage in the arithmetic.) An alternate method is used in the following examples. The operation is first performed in decimal and then the result is converted into octal, before proceeding to the next stage, as shown in the scratchpad.

Table 1.4 Octal Arithmetic

(a) Addition

		A							
	A + B	0	1	2	3	4	5	6	7
	0	0	1	2	3	4	5	6	7
	1	1	2	3	4	5	6	7	10
	2	2	3	4	5	6	7	10	11
B	3	3	4	5	6	7	10	11	12
	4	4	5	6	7	10	11	12	13
	5	5	6	7	10	11	12	13	14
	6	6	7	10	11	12	13	14	15
	7	7	10	11	12	13	14	15	16

(b) Multiplication

		A							
	A × B	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0	0
	1	0	1	2	3	4	5	6	7
	2	0	2	4	6	10	12	14	16
B	3	0	3	6	11	14	17	22	25
	4	0	4	10	14	20	24	30	34
	5	0	5	12	17	24	31	36	43
	6	0	6	14	22	30	36	44	52
	7	0	7	16	25	34	43	52	61