Integers

Because $N > N_1 > N_2 \ldots$, continuing this sequence will eventually produce a quotient $N_{m-1} = 1$ (except for the decimal integers 0 and 1, whose binary equivalents are 0 and 1, respectively) and a remainder R_{m-2}, which is 0 or 1. Then

$$N = (1 * 2^{m-1}) + (R_{m-2} * 2^{m-2}) + \ldots + (R_2 * 2^2) + (R_1 * 2^1) + R_0$$

which is the binary form of N. Hence, we convert from base 10 to base 2 by repeated divisions by 2. The remainders and the final quotient, 1, give us, in order of increasing significance, the binary digits of N.

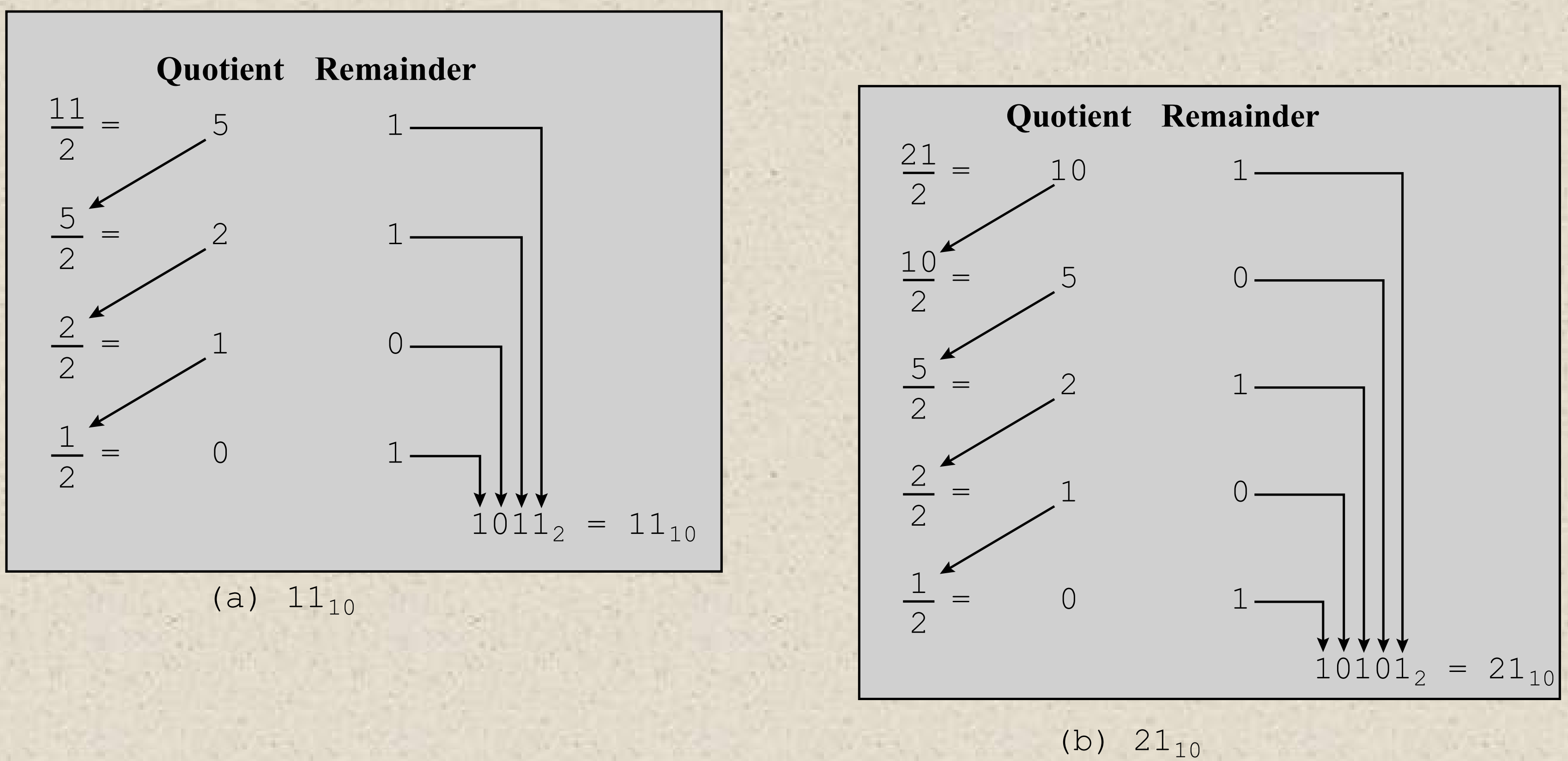

Figure 9.1 Examples of Converting from Decimal Notation to Binary Notation for Integers

Fractions

For the fractional part, recall that in binary notation, a number with a value between 0 and 1 is represented by

$$0.b_{-1}b_{-2}b_{-3}\ldots \qquad b_i = 0 \text{ or } 1$$

and has the value

$$(b_{-1} * 2^{-1}) + (b_{-2} * 2^{-2}) + (b_{-3} * 2^{-3})\ldots$$

This can be rewritten as

$$2^{-1} * (b_{-1} + 2^{-1} * (b_{-2} + 2^{-1} * (b_{-3} + \ldots)\ldots))$$

Suppose we want to convert the number F $(0 < F < 1)$ from decimal to binary notation. We know that F can be expressed in the form

$$F = 2^{-1} * (b_{-1} + 2^{-1} * (b_{-2} + 2^{-1} * (b_{-3} + \ldots)\ldots))$$

If we multiply F by 2, we obtain,

$$2 * F = b_{-1} + 2^{-1} * (b_{-2} + 2^{-1} * (b_{-3} + \ldots)\ldots)$$

Continued . . .

Fractions

From this equation, we see that the integer part of $(2 * F)$, *which must be* either 0 or 1 because $0 < F < 1$, *is simply* b_{-1}. *So we can say* $(2 * F) = b_{-1} + F_1$, where $0 < F_1 < 1$ *and where*

$$F_1 = 2\text{-}1 * (b_{-2} + 2^{-1} * (b_{-3} + 2^{-1} * (b_{-4} + . . .) . . .))$$

To find b_{-2}, we repeat the process.
At each step, the fractional part of the number from the previous step is multiplied by 2. The digit to the left of the decimal point in the product will be 0 or 1 and contributes to the binary representation, starting with the most significant digit. The fractional part of the product is used as the multiplicand in the next step.

Figure 9.2

Examples of Converting from Decimal Notation To Binary Notation For Fractions

Product	Integer Part	0.110011_2
$0.81 \times 2 = 1.62$	1	
$0.62 \times 2 = 1.24$	1	
$0.24 \times 2 = 0.48$	0	
$0.48 \times 2 = 0.96$	0	
$0.96 \times 2 = 1.92$	1	
$0.92 \times 2 = 1.84$	1	

(a) $0.81_{10} = 0.110011_2$ (approximately)

Product	Integer Part	0.01_2
$0.25 \times 2 = 0.5$	0	
$0.5 \times 2 = 1.0$	1	

(b) $0.25_{10} = 0.01_2$ (exactly)

Hexadecimal Notation

- Binary digits are grouped into sets of four bits, called a *nibble*
- Each possible combination of four binary digits is given a symbol, as follows:

0000 = 0	0100 = 4	1000 = 8	1100 = C
0001 = 1	0101 = 5	1001 = 9	1101 = D
0010 = 2	0110 = 6	1010 = A	1110 = E
0011 = 3	0111 = 7	1011 = B	1111 = F

- Because 16 symbols are used, the notation is called *hexadecimal* and the 16 symbols are the *hexadecimal digits*
- Thus

$$2C_{16} = (2_{16} * 16^1) + (C_{16} * 16^0)$$

$$= (2_{10} * 16^1) + (12_{10} * 16^0) = 44$$

Table 9.3

Decimal, Binary, and Hexadecimal

Decimal (base 10)	Binary (base 2)	Hexadecimal (base 16)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	0001 0000	10
17	0001 0001	11
18	0001 0010	12
31	0001 1111	1F
100	0110 0100	64
255	1111 1111	FF
256	0001 0000 0000	100

Hexadecimal Notation

Not only used for representing integers but also as a concise notation for representing any sequence of binary digits

Reasons for using hexadecimal notation are:

- It is more compact than binary notation
- In most computers, binary data occupy some multiple of 4 bits, and hence some multiple of a single hexadecimal digit
- It is extremely easy to convert between binary and hexadecimal notation

Summary

Chapter 9

Number Systems

- The decimal system
- Positional number systems
- The binary system
- Converting between binary and decimal
  - Integers
  - Fractions
- Hexadecimal notation