INTRODUCTION TO NUMBER THEORY

FARIZA RAKYMZHANKYZY,

SENIOR-LECTURE

NUMBER THEORY

- The part of mathematics devoted to the study of the set of integers and their properties is known as Number Theory. In this lecture we will develop some of the important concepts of Number Theory including many of those used in computer science.

NUMBER THEORY

- In this lecture we introduce several important applications of number theory. In particular, we will use number theory to generate pseudorandom numbers, to assign memory locations to computer files, and to find check digits used to detect errors in various kinds of identification numbers. We also introduce the subject of cryptography. Number theory plays an essentially role both in classical cryptography, first used thousands of years ago, and modern cryptography, which plays an essential role in electronic communication.

NUMBER THEORY

- We will show how the ideas we develop can be used in cryptographical protocols, introducing protocols for sharing keys and for sending signed messages. Number theory, once considered the purest of subjects, has become an essential tool in providing computer and Internet security.

DIVISIBILITY AND MODULAR ARITHMETIC

- The ideas that we will develop here are based on the notion of divisibility. Division of an integer by a positive integer produces a quotient and a remainder. Working with these remainders leads to modular arithmetic, which plays an important role in mathematics and which is used throughout computer science. We will discuss some important applications of modular arithmetic later, including generating pseudorandom numbers, assigning computer memory locations to files, constructing check digits, and encrypting messages.

DIVISION

- When one integer is divided by a second nonzero integer, the quotient may or may not be an integer. For example, $\frac{12}{3} = 4$ is an integer, whereas $\frac{11}{4} = 2.75$ is not. This leads to Definition 1.

DIVISION

- **DEFINITION 1.** If a and b are integers with $a \neq 0$, we say that a divides b if there is an integer c such that $b = ac$, or equivalently, if b/a is an integer. When a divides b we say that a is a *factor* or *divisor* of b, and that b is a *multiple* of a. The notation $a|b$ denotes that a divides b. We write $a \nmid b$ when a does not divide b.
- **Remark:** We can express $a|b$ using quantifiers as $\exists c(ac = b)$, where the universe of discourse is the set of integers.

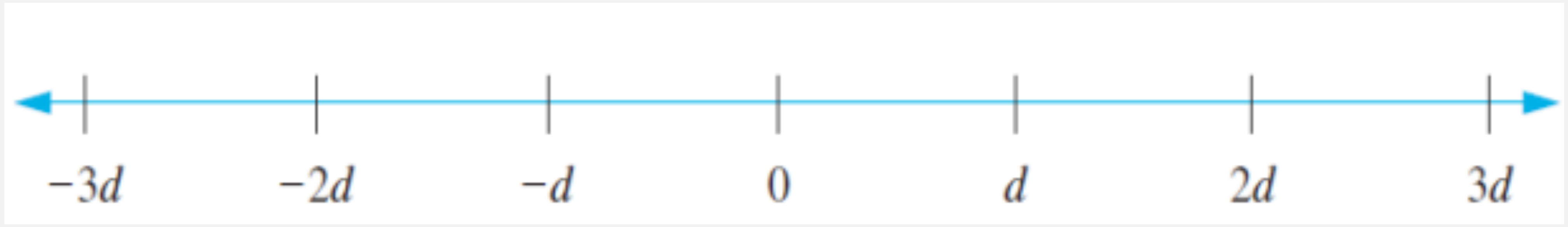
EXAMPLES

- Determine whether $3|7$ and whether $3|12$.
- Let n and d be positive integers. How many positive integers not exceeding n are divisible by d?

EXAMPLES

- Let n and d be positive integers. How many positive integers not exceeding n are divisible by d?

In Figure, a number line indicates which integers are divisible by the positive integer d.



DIVISION

THEOREM 1. Let $a, b,$ and c be integers, where $a \neq 0$. Then

- (i) if $a|b$ and $a|c$, then $a|(b + c)$;
- (ii) if $a|b$, then $a|bc$ for all integers c;
- (iii) if $a|b$ and $b|c$, then $a|c$.

DIVISION

- **COROLLARY 1**. If $a, b,$ and c are integers, where $a \neq 0$, such that $a|b$ and $a|c$, then $a|(mb + nc)$ whenever m and n are integers.

THE DIVISION ALGORITHM

- When an integer is divided by a positive integer, there is a quotient and a remainder, as the division algorithm shows.

THEOREM 2. (THE DIVISION ALGORITHM) Let a be an integer and d a positive integer. Then there are unique integers q and r, with $0 \leq r < d$, such that $a = dq + r$.

THE DIVISION ALGORITHM

DEFINITION 2. In the equality given in the division algorithm, d is called the divisor, a is called the dividend, q is called the quotient, and r is called the remainder. This notation is used to express the quotient and remainder:

- $q = a \ \boldsymbol{div}\ d, \qquad r = a \ \boldsymbol{mod}\ d.$

THE DIVISION ALGORITHM

- **Remark:** Note that both $a\ \boldsymbol{div}\ d$ and $a\ \boldsymbol{mod}\ d$ for a fixed d are functions on the set of integers.
- Furthermore, when a is an integer and d is a positive integer, we have

$$q = a\ \boldsymbol{div}\ d = [a/d] \text{ and } r = a\ \boldsymbol{mod}\ d = a - dq.$$

EXAMPLES

- What are the quotient and remainder when 101 is divided by 11?
- What are the quotient and remainder when -11 is divided by 3?

THE DIVISION ALGORITHM

- **Remark:** A programming language may have one, or possibly two, operators for modular arithmetic, denoted by mod (in BASIC, Maple, Mathematica, EXCEL, and SQL), % (in C, C++, Java, and Python), rem (in Ada and Lisp), or something else. Be careful when using them, because for $a < 0$, some of these operators return $a - m([a/m] + 1)$ instead of $a\,\boldsymbol{mod}\,m = a - m[a/m]$. Also, unlike $a\,\boldsymbol{mod}\,m$, some of these operators are defined when $m < 0$, and even when $m = 0$.

MODULAR ARITHMETIC

- In some situations we care only about the remainder of an integer when it is divided by some specified positive integer. For instance, when we ask what time it will be (on a 24-hour clock) 50 hours from now, we care only about the remainder when 50 plus the current hour is divided by 24.

MODULAR ARITHMETIC

- Because we are often interested only in remainders, we have special notations for them. We have already introduced the notation $a\ \boldsymbol{mod}\ m$ to represent the remainder when an integer a is divided by the positive integer m. We now introduce a different, but related, notation that indicates that two integers have the same remainder when they are divided by the positive integer m.

MODULAR ARITHMETIC

- **DEFINITION 3.** If a and b are integers and m is a positive integer, then a is congruent to b modulo m if m divides $a - b$. We use the notation $a \equiv b (mod\ m)$ to indicate that a is congruent to b modulo m. We say that $a \equiv b (mod\ m)$ is a congruence and that m is its modulus (plural moduli). If a and b are not congruent modulo m, we write

$$a \not\equiv b (mod\ m).$$

MODULAR ARITHMETIC

- Although both notations $a \equiv b(mod\ m)$ and $a\ \boldsymbol{mod}\ m = b$ include "mod," they represent fundamentally different concepts. The first represents a relation on the set of integers, whereas the second represents a function.
- However, the relation $a \equiv b(mod\ m)$ and the $\boldsymbol{mod}\ m$ function are closely related, as described in Theorem 3.

MODULAR ARITHMETIC

- **THEOREM 3.** Let a and b be integers, and let m be a positive integer. Then $a \equiv b (mod\ m)$ if and only if $a\ \boldsymbol{mod}\ m = b\ \boldsymbol{mod}\ m$.

EXAMPLE

- Determine whether 17 is congruent to 5 modulo 6
- and whether 24 and 14 are congruent modulo 6.

MODULAR ARITHMETIC

- The great German mathematician Karl Friedrich Gauss developed the concept of congruences at the end of the eighteenth century.
- The notion of congruences has played an important role in the development of number theory. Theorem 4 provides a useful way to work with congruences.

MODULAR ARITHMETIC

THEOREM 4. Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that

$$a = b + km.$$

MODULAR ARITHMETIC

- The set of all integers congruent to an integer a modulo m is called the congruence class of a modulo m.

THEOREM 5. Let m be a positive integer. If $a \equiv b (mod\ m)$ and

$c \equiv d (mod\ m)$, then

$$a + c \equiv b + d (mod\ m) \quad and \quad ac \equiv bd (mod\ m).$$

EXAMPLE

- Because $7 \equiv 2 (mod\ 5)$ and $11 \equiv 1 (mod\ 5)$, it follows from Theorem 5 that

$$18 = 7 + 11 \equiv 2 + 1 = 3 (mod\ 5),$$

and that

$$77 = 7 \cdot 11 \equiv 2 \cdot\ 1 = 2 (mod\ 5).$$

ARITHMETIC MODULO m

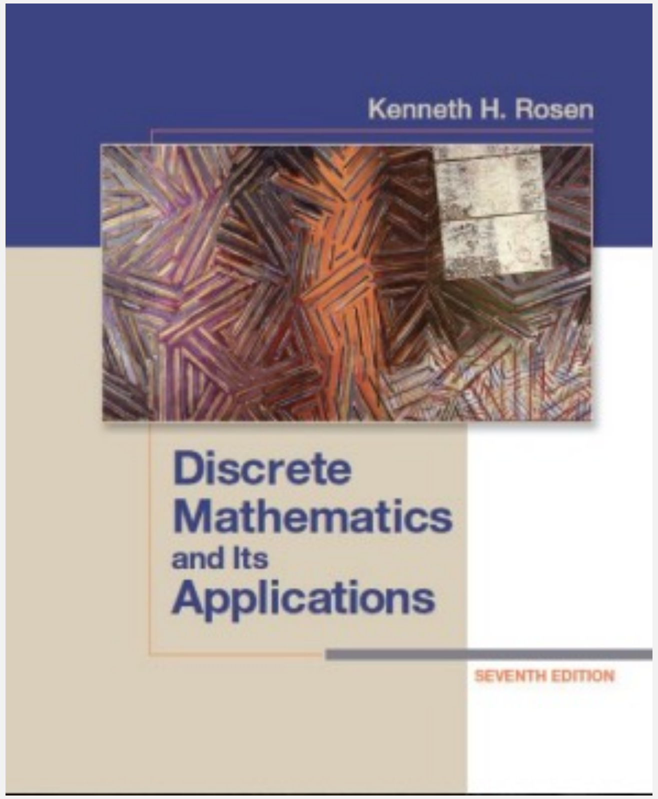
- We can define arithmetic operations on $\mathbb{Z}_m$, the set of nonnegative integers less than m, that is, the set $\{0, 1, \ldots, m - 1\}$. In particular, we define addition of these integers, denoted by $+_m$ by

$$a +_m b = (a + b)\boldsymbol{mod}\, m,$$

where the addition on the right-hand side of this equation is the ordinary addition of integers, and we define multiplication of these integers, denoted by $\cdot_m$ by

$$a \cdot_m b = a \cdot b\, \boldsymbol{mod}\, m$$

HOMEWORK: EXERCISES 6, 10, 12, 14, 22, 24 ON PP. 244-245;


Kenneth H. Rosen

Discrete Mathematics and Its Applications

SEVENTH EDITION

INTEGER REPRESENTATIONS AND ALGORITHMS

- Integers can be expressed using any integer greater than one as a base, as we will show. Although we commonly use decimal (base 10), representations, binary (base 2), octal (base 8), and hexadecimal (base 16) representations are often used, especially in computer science. Given a base b and an integer n, we will show how to construct the base b representation of this integer. We will also explain how to quickly covert between binary and octal and between binary and hexadecimal notations.

INTEGER REPRESENTATIONS AND ALGORITHMS

- The term algorithm originally referred to procedures for performing arithmetic operations using the decimal representations of integers. These algorithms, adapted for use with binary representations, are the basis for computer arithmetic. They provide good illustrations of the concept of an algorithm and the complexity of algorithms. For these reasons, they will be discussed here.

REPRESENTATIONS OF INTEGERS

- In everyday life we use decimal notation to express integers. For example, 965 is used to denote

$$9 \cdot 10^2 + 6 \cdot 10 + 5$$

However, it is often convenient to use bases other than 10.

REPRESENTATIONS OF INTEGERS

- In particular, computers usually use binary notation (with 2 as the base) when carrying out arithmetic, and octal (base 8) or hexadecimal (base 16) notation when expressing characters, such as letters or digits. In fact, we can use any integer greater than 1 as the base when expressing integers. This is stated in Theorem 1.

REPRESENTATIONS OF INTEGERS

- **THEOREM 1.** Let b be an integer greater than 1. Then if n is a positive integer, it can be expressed uniquely in the form

$$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0,$$

where k is a nonnegative integer, $a_0, a_1, \ldots, a_k$ are nonnegative integers less than b, and $a_k \neq 0$.

REPRESENTATIONS OF INTEGERS

- The representation of n given in Theorem 1 is called the **base *b* expansion of *n***. The base b expansion of n is denoted by

$$(a_k a_{k-1} \ldots a_1 a_0)_b .$$

For instance, $(245)_8$ represents $2 \cdot 8^2 + 4 \cdot 8 + 5$.

- Typically, the subscript 10 is omitted for base 10 expansions of integers because base 10, or decimal expansions are commonly used to represent integers.

REPRESENTATIONS OF INTEGERS

BINARY EXPANSIONS

- Choosing 2 as the base gives binary expansions of integers. In binary notation each digit is either a 0 or a 1. In other words, the binary expansion of an integer is just a bit string. Binary expansions (and related expansions that are variants of binary expansions) are used by computers to represent and do arithmetic with integers.

EXAMPLE

- What is the decimal expansion of the integer that has

$$(1\ 0101\ 1111)_2$$

as its binary expansion?

REPRESENTATIONS OF INTEGERS

OCTAL AND HEXADECIMAL EXPANSIONS

- Among the most important bases in computer science are base 2, base 8, and base 16. Base 8 expansions are called **octal** expansions and base 16 expansions are **hexadecimal** expansions.

EXAMPLES

- What is the decimal expansion of the number with octal expansion $(7016)_8$?

Sixteen different digits are required for hexadecimal expansions. Usually, the hexadecimal digits used are $0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E$, and F, where the letters A through F represent the digits corresponding to the numbers 10 through 15 (in decimal notation).

- What is the decimal expansion of the number with hexadecimal expansion $(2AE0B)_{16}$?

REPRESENTATIONS OF INTEGERS

BASE CONVERSION

- We will now describe an algorithm for constructing the base b expansion of an integer n. First, divide n by b to obtain a quotient and remainder, that is,

$$n = bq_0 + a_0, \quad 0 \leq a_0 < b.$$

- The remainder, a_0, is the rightmost digit in the base b expansion of n. Next, divide q_0 by b to obtain

$$q_0 = bq_1 + a_1, \quad 0 \leq a_1 < b.$$

REPRESENTATIONS OF INTEGERS

BASE CONVERSION

- We see that a_1 is the second digit from the right in the base b expansion of n. Continue this process, successively dividing the quotients by b, obtaining additional base b digits as the remainders. This process terminates when we obtain a quotient equal to zero. It produces the base b digits of n from the right to the left.

EXAMPLE

- Find the octal expansion of $(12345)_{10}$.
- Find the binary expansion of $(241)_{10}$.

REPRESENTATIONS OF INTEGERS

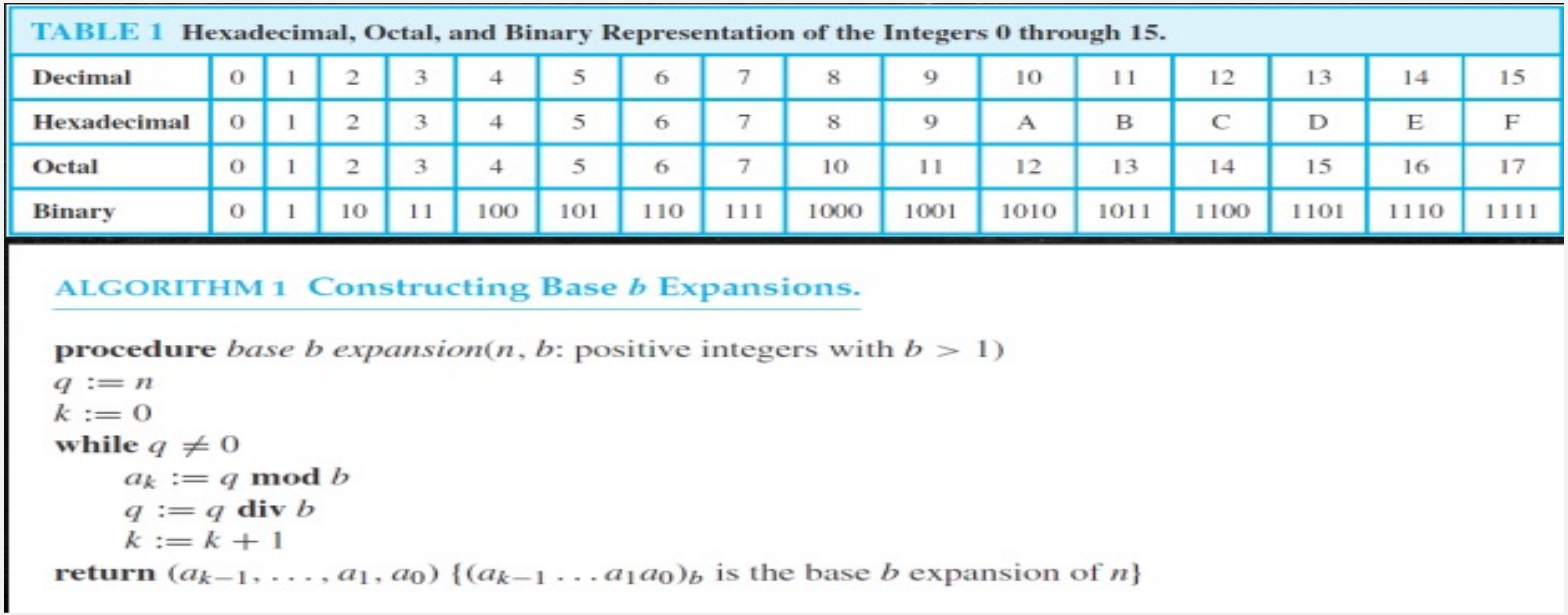


TABLE 1 Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Octal	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

ALGORITHM 1 Constructing Base b Expansions.

```
procedure base b expansion($n$, $b$: positive integers with $b > 1$)
$q := n$
$k := 0$
while $q \neq 0$
    $a_k := q$ mod $b$
    $q := q$ div $b$
    $k := k + 1$
return $(a_{k-1}, \ldots, a_1, a_0)$ {$(a_{k-1} \ldots a_1 a_0)_b$ is the base $b$ expansion of $n$}
```

REPRESENTATIONS OF INTEGERS

Algorithms for Integer Operations

- The algorithms for performing operations with integers using their binary expansions are extremely important in computer arithmetic. We will describe algorithms for the addition and the multiplication of two integers expressed in binary notation.

REPRESENTATIONS OF INTEGERS

- Throughout this discussion, suppose that the binary expansions of a and b are

$$a = (a_{n-1} \dots a_1 a_0)_2, \quad b = (b_{n-1} \dots b_1 b_0)_2$$

so that a and b each have n bits (putting bits equal to 0 at the beginning of one of these expansions if necessary).

- We will measure the complexity of algorithms for integer arithmetic in terms of the number of bits in these numbers.

REPRESENTATIONS OF INTEGERS

ADDITION ALGORITHM

- Consider the problem of adding two integers in binary notation. To add a and b, first add their rightmost bits. This gives $a_0 + b_0 = c_0 \cdot 2 + s_0$, where s_0 is the rightmost bit in the binary expansion of $a + b$ and c_0 is the carry, which is either 0 or 1.
- Then add the next pair of bits and the carry, $a_1 + b_1 + c_0 = c_1 \cdot 2 + s_1$, where s_1 is the next bit (from the right) in the binary expansion of $a + b$, and c_1 is the carry.

REPRESENTATIONS OF INTEGERS

ADDITION ALGORITHM

- Continue this process, adding the corresponding bits in the two binary expansions and the carry, to determine the next bit from the right in the binary expansion of $a + b$.
- At the last stage, add a_{n-1}, b_{n-1}, and c_{n-2} to obtain $c_{n-1} \cdot 2 + s_{n-1}$. The leading bit of the sum is $s_n = c_{n-1}$. This procedure produces the binary expansion of the sum, namely, $a + b = (s_n s_{n-1} \dots s_1 s_0)_2$

EXAMPLE

- Add $a = (1110)_2$ and $b = (1011)_2$.

REPRESENTATIONS OF INTEGERS

MULTIPLICATION ALGORITHM

- Next, consider the multiplication of two n-bit integers a and b. The conventional algorithm (used when multiplying with pencil and paper) works as follows. Using the distributive law, we see that

$$ab = a(b_0 2^0 + b_1 2^1 + \cdots + b_{n-1} 2^{n-1}) =$$
$$a(b_0 2^0) + a(b_2 2^1) + \cdots + a(b_{n-1} 2^{n-1}).$$

- We can compute ab using this equation.

EXAMPLE

- Find the product of $a = (110)_2$ and $b = (101)_2$.

HOMEWORK: EXERCISES 2, 4, 6, 8, 22, 32 ON PP. 255-256;