Computer Systems
Lecture 4

Overview

- Data, Information and Knowledge
- Data codes – numeric and character
- Binary number system
- Base n notation
- Conversion of binary numbers to decimal
- Conversion of decimal numbers to binary
- Hexadecimal notation

Data, Information and Knowledge

- Data – raw facts, figures, measurements, …
  - 1.00001, 1.00000010, 2.0000101, 3.0000102, 5.000..
- Information
  - data organized into useful representation
  - 1.000, 1.000, 2.000, 3.000, 5.000, …
- Knowledge
  - application of reasoned analysis of information
  - ‘data are in increasing order’, ‘data can be derived based on Fibonacci sequencing’, etc

Measurement of Information

- In 1948, motivated by the problem of efficiently transmitting information over a noisy communication channel, Claude Shannon introduced a revolutionary new probabilistic way of thinking about communication and simultaneously created the first truly mathematical theory of **entropy**.
- His ideas created two main lines of development: information theory and coding theory

- Entropy of an event X is related to p(X)

Information entropy

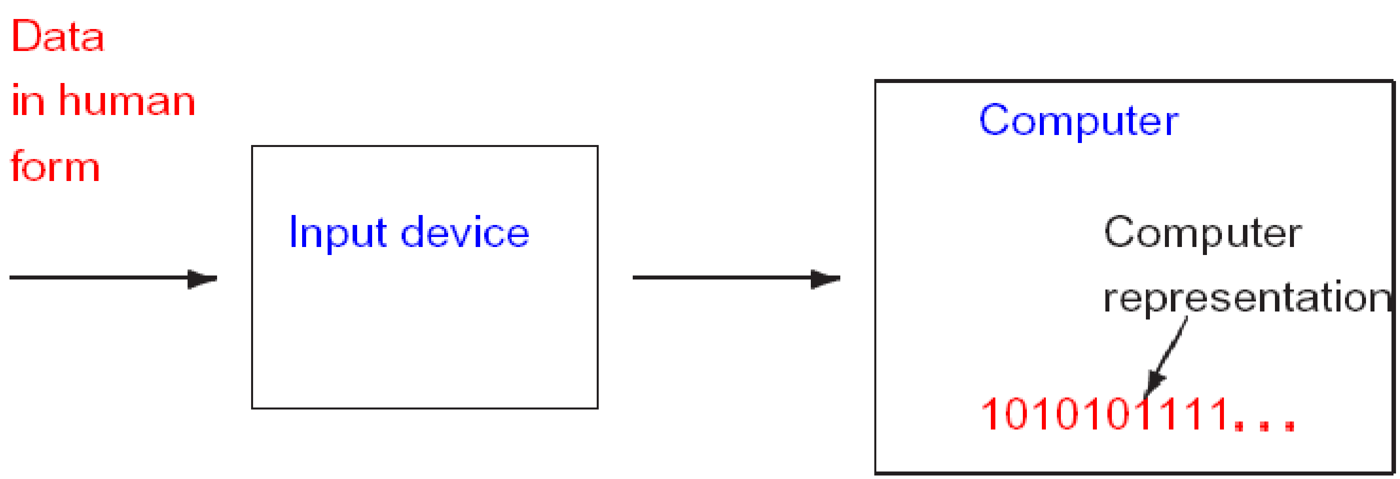
- Which one carries more information?
  - ‘There is a terror attack scheduled today at 11pm in LA.’
  - ‘Today’s weather is fine.’

Data codes – numeric and character

- Within a computer the **binary number system** is a system of choice, both for data storage and for internal processing of operations.
  - Based on the two-state, ON/OFF technology.
- We as human beings don't choose normally to work with binary form. We use language, images, sounds…numbers usually represented in **decimal number system**.

Data codes (cont.)

- Thus, one needs to convert words, numbers, images, sounds into a binary form and back.



Alphanumeric data

- The majority of the data originally comes in the form of letters in alphabet, numbers and punctuation (**alphanumeric data)**.
  - They are represented in computers by binary numbers.
  - Later we will see how.

Bit

- A bit is the most **basic unit of information** possible: it contains the information necessary to distinguish two alternatives (**1** or **0**, **YES** or **NOT**, etc.).

- We think of these alternatives as being numbers **0** and **1**.
- Two alternatives are represented by **two-state** elements (memory cell is charged, or not).

Binary number system

- A sequence of bits (binary digits) represents a number in the binary notation
- Example: the sequence

0	0	0	1	0	1	1	0

represents the number 22: $10110_2 = 22_{10}$

Binary number system

- As in decimal system a binary digit (1 or 0) represents some value depending on where it is written in the number:

…1024	512	256	128	64	32	16	8	4	2	1
1	0	1	1	0	0	1	0	0	1	1

$$1024 + 256 + 128 + 16 + 2 + 1 = 1427.$$

- Therefore, 10110010011 is a binary representation of 1427 (decimal).

Binary number system

$$0_2 = 0_{10}$$
$$1_2 = 1_{10}$$
$$10_2 = 1 \times 2^1 + 0 \times 2^0 = 2 + 0 = 2_{10}$$
$$11_2 = 1 \times 2^1 + 1 \times 2^0 = 2 + 1 = 3_{10}$$
$$\vdots$$
$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^0 =$$
$$8 + 4 + 1 = 13_{10}$$
$$\vdots$$

Base n notation, n=10,2,16,…

- Base 10 notation, or *decimal*. 10 digits are in use.

$$\underline{a_1 a_2, \ldots a_k}_{10} =$$

$$a_k \times 10^0 + a_{k-1} \times 10^1 + \ldots + a_1 \times 10^{k-1}.$$

$$23_{10} = 2 \times 10^1 + 3 \times 10^0$$

$$171_{10} = 1 \times 10^2 + 7 \times 10^1 + 1 \times 10^0$$

- Base 2 notation, or *binary*. 2 digits are in use.

$$\underline{a_1 a_2, \ldots a_k}_2 = a_k \times 2^0 + a_{k-1} \times 2^1 + \ldots + a_1 \times 2^{k-1}$$

$$23_{10} = 10111_2 = 1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 1\times 2^1 + 1\times 2^0$$

$$171_{10} = 10101011_2$$

Conversion of binary numbers to decimal

- Shown, actually, on previous two slides.
- It is done by
  - multiplying the weighting value by the associated digit (bit) and
  - adding the results.

- It is easy to implement.

Exercises

- Convert the following binary numbers into decimal.
  - 10000001
  - 11111111

Conversion of decimal numbers to binary

- It can be done in a similar way. For example:

  2397_{10} = 2*1000 + 3*100+9*10+7*1 =

  10*1111101000+11*1100100+

  1001*1010+111*1= 100101011101_2

- Requires binary arithmetic.
- Easy to implement on a computer, but difficult for a human.

More human-friendly method for converting decimal into binary

- **Repeatedly divide** the decimal number **by 2**
- **Write down the remainder** from each stage (1 or 0) from right to left

Integer remainder

- If a and d are integers, with d non-zero, then
  - a remainder is an integer r such that $a = qd + r$ for some integer q, and
  - $0 \leq |r| < |d|$.
- The number q is called the *quotient*.

Conversion by division

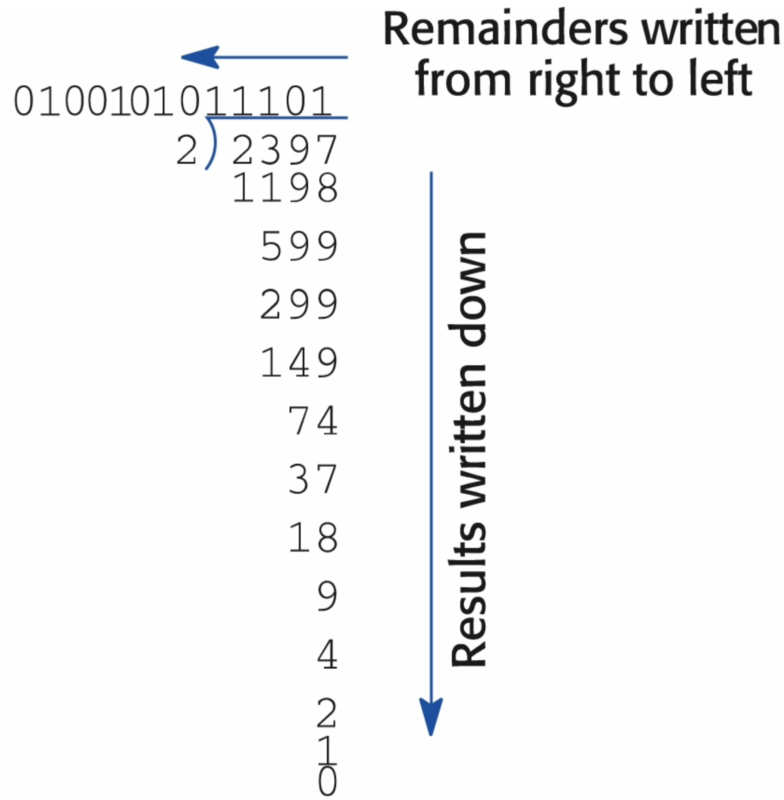


Fig. 2.9 Conversion of decimal numbers to binary by repeated division.

Exercises

- Convert the following decimal numbers into binary.
  - 1023
  - 998

Binary vs. decimal notation

- Decimal notation is more compact than binary.
- Decimal is more convenient for humans.

- Binary is more “convenient” for computers due to two-state, ON/OFF technology employed.

Hexadecimal notation

- Decimal notation uses 10 as a base.
- Binary notation uses 2 as a base.

- Every number can be used as a base.

- Hexadecimal notation uses 16 as a base.

Hexadecimal notation (cont.)

- Why hexadecimal, or base 16, number notation?
  - Convenient shorthand for binary numbers.
  - Every hexadecimal digit exactly represents 4 binary bits.

Base n notation, n=10,2,16,…(cont.)

- Base 16 notation, or *hexadecimal*. 16 digits are in use:

$0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.$

$\underline{a_1 a_2, \ldots a_k}_{16} =$

$a_k \times 16^0 + a_{k-1} \times 16^1 + \ldots + a_1 \times 16^{k-1}.$

$23_{10} = 1 \times 16^1 + 7 \times 16^1 = 17_{16}$

$171_{10} = 10 \times 16^1 + 11 \times 16^0 = AB_{16}$

Hexadecimal and binary

- Any hexadecimal digit represents some 4-digit pattern (binary number) and vice versa:

0	=	0000_2	8	=	1000_2
1	=	0001_2	9	=	1001_2
2	=	0010_2	A	=	1010_2
3	=	0011_2	B	=	1011_2
4	=	0100_2	C	=	1100_2
5	=	0101_2	D	=	1101_2
6	=	0110_2	E	=	1110_2
7	=	0111_2	F	=	1111_2

Translation from binary to hexadecimal

- Any binary number can be translated into hexadecimal form,
  - first breaking it into 4-digits groups (from right to left) and
  - translating each group into one hexadecimal digit:

$$10011110_2 = 9E_{16}$$

Translation from hexadecimal to binary

Translate every hexadecimal digit into the 4-digit binary pattern.

Example:

$$AA3E = 1010101000111110_2$$

Translation from hexadecimal to decimal

- Similar to the translation from binary to decimal:

  $2AE_{16}$

  $= 2*16^2 + 10*16 + 14*1$

  $= 2*256 + 10*16 + 14*1$

  $= 512+160+14$

  $= 672+14$

  $= 686_{10}$

Exercises

- Translate the following hex number into decimal.
  - 03AB
  - 0FEE2

Translation from decimal to hexadecimal

- As in the case of binary system:
  - **Repeatedly divide** the decimal number **by 16**
  - **Write down the remainder** from each stage (0,1,…,15) from right to left.

- One should replace the remainders 10, 11, 12, 13, 14, 15 with the corresponding hexadecimal digits A,B,C,D,E,F.

Exercise

- Convert the following into Hex.
  - 255
  - 128

How data are represented in a computer – Summary

- **ASCII Table (7-bit)**
  (ASCII = American Standard Code for Information Interchange)

Decimal	Octal	Hex	Binary	Value
---	---		--- ---	---
048	060	030	00110000	**0**
049	061	031	00110001	**1**
050	062	032	00110010	**2**
056	070	038	00111000	**8**
057	071	039	00111001	**9**
065	101	041	01000001	**A**
066	102	042	01000010	**B**
097	141	061	01100001	**a**
098	142	062	01100010	**b**

How are programs represented in a computer?

Q&A

- Define ‘bit’.
- Decimal notation is more compact than binary. (T or F).
- Binary is more “convenient” for computers due to ?
- Every number can be used as a base. (T or F).
- Hexadecimal notation uses ? as a base.
- Octal notation uses ? as a base.

Q&A

- Why do we use hexadecimal, or base 16, number notation?
- How many bits are required to encode ASCII?

Readings

- [Wil06] Chapter 2, section 2.8.