# Chapter 04

# Computer Codes

Computer Fundamentals - Pradeep K. Sinha & Priti Sinha

---

# Learning Objectives

### In this chapter you will learn about:

§ Computer data

§ Computer codes: representation of data in binary

§ Most commonly used computer codes

§ Collating sequence

# Data Types

§ **Numeric Data** consists of only numbers 0, 1, 2, ..., 9

§ **Alphabetic Data** consists of only the letters A, B, C, ..., Z, in both uppercase and lowercase, and blank character

§ **Alphanumeric Data** is a string of symbols where a symbol may be one of the letters A, B, C, ..., Z, in either uppercase or lowercase, or one of the digits 0, 1, 2, ..., 9, or a special character, such as + - * / , . ( ) = etc.

# Computer Codes

§ Computer codes are used for internal representation of data in computers

§ As computers use binary numbers for internal data representation, computer codes use binary coding schemes

§ In binary coding, every symbol that appears in the data is represented by a group of bits

§ The group of bits used to represent a symbol is called a **byte**

*(Continued on next slide)*

2

# Computer Codes

*(Continued from previous slide..)*

§ As most modern coding schemes use 8 bits to represent a symbol, the term byte is often used to mean a group of 8 bits

§ Commonly used computer codes are BCD, EBCDIC, and ASCII

# BCD

§ BCD stands for **B**inary **C**oded **D**ecimal

§ It is one of the early computer codes

§ It uses 6 bits to represent a symbol

§ It can represent 64 ($2^6$) different characters

## Coding of Alphabetic and Numeric Characters in BCD

| Char | BCD Code | | Octal |
|------|------|-------|-------|
| | Zone | Digit | |
| A | 11 | 0001 | 61 |
| B | 11 | 0010 | 62 |
| C | 11 | 0011 | 63 |
| D | 11 | 0100 | 64 |
| E | 11 | 0101 | 65 |
| F | 11 | 0110 | 66 |
| G | 11 | 0111 | 67 |
| H | 11 | 1000 | 70 |
| I | 11 | 1001 | 71 |
| J | 10 | 0001 | 41 |
| K | 10 | 0010 | 42 |
| L | 10 | 0011 | 43 |
| M | 10 | 0100 | 44 |

| Char | BCD Code | | Octal |
|------|------|-------|-------|
| | Zone | Digit | |
| N | 10 | 0101 | 45 |
| O | 10 | 0110 | 46 |
| P | 10 | 0111 | 47 |
| Q | 10 | 1000 | 50 |
| R | 10 | 1001 | 51 |
| S | 01 | 0010 | 22 |
| T | 01 | 0011 | 23 |
| U | 01 | 0100 | 24 |
| V | 01 | 0101 | 25 |
| W | 01 | 0110 | 26 |
| X | 01 | 0111 | 27 |
| Y | 01 | 1000 | 30 |
| Z | 01 | 1001 | 31 |

*(Continued on next slide)*

---

## Coding of Alphabetic and Numeric Characters in BCD

*(Continued from previous slide..)*

| Character | BCD Code | | Octal Equivalent |
|-----------|------|-------|------------------|
| | Zone | Digit | |
| 1 | 00 | 0001 | 01 |
| 2 | 00 | 0010 | 02 |
| 3 | 00 | 0011 | 03 |
| 4 | 00 | 0100 | 04 |
| 5 | 00 | 0101 | 05 |
| 6 | 00 | 0110 | 06 |
| 7 | 00 | 0111 | 07 |
| 8 | 00 | 1000 | 10 |
| 9 | 00 | 1001 | 11 |
| 0 | 00 | 1010 | 12 |

4

# BCD Coding Scheme (Example 1)

**Example**

Show the binary digits used to record the word BASE in BCD

**Solution:**

B = 110010 in BCD binary notation
A = 110001 in BCD binary notation
S = 010010 in BCD binary notation
E = 110101 in BCD binary notation

So the binary digits

110010   110001   010010   110101
  B        A        S        E

will record the word BASE in BCD

---

# BCD Coding Scheme (Example 2)

*Example*

Using octal notation, show BCD coding for the word DIGIT

**Solution:**

D = 64 in BCD octal notation
I = 71 in BCD octal notation
G = 67 in BCD octal notation
I = 71 in BCD octal notation
T = 23 in BCD octal notation

Hence, BCD coding for the word DIGIT in octal notation will be

64  71  67  71  23
D   I   G   I   T

5

# EBCDIC

§ EBCDIC stands for **E**xtended **B**inary **C**oded **D**ecimal **I**nterchange **C**ode

§ It uses 8 bits to represent a symbol

§ It can represent 256 ($2^8$) different characters

# Coding of Alphabetic and Numeric Characters in EBCDIC

| Char | EBCDIC Code | | Hex |
|------|-------|------|-----|
|      | Digit | Zone |     |
| A | 1100 | 0001 | C1 |
| B | 1100 | 0010 | C2 |
| C | 1100 | 0011 | C3 |
| D | 1100 | 0100 | C4 |
| E | 1100 | 0101 | C5 |
| F | 1100 | 0110 | C6 |
| G | 1100 | 0111 | C7 |
| H | 1100 | 1000 | C8 |
| I | 1100 | 1001 | C9 |
| J | 1101 | 0001 | D1 |
| K | 1101 | 0010 | D2 |
| L | 1101 | 0011 | D3 |
| M | 1101 | 0100 | D4 |

| Char | EBCDIC Code | | Hex |
|------|-------|------|-----|
|      | Digit | Zone |     |
| N | 1101 | 0101 | D5 |
| O | 1101 | 0110 | D6 |
| P | 1101 | 0111 | D7 |
| Q | 1101 | 1000 | D8 |
| R | 1101 | 1001 | D9 |
| S | 1110 | 0010 | E2 |
| T | 1110 | 0011 | E3 |
| U | 1110 | 0100 | E4 |
| V | 1110 | 0101 | E5 |
| W | 1110 | 0110 | E6 |
| X | 1110 | 0111 | E7 |
| Y | 1110 | 1000 | E8 |
| Z | 1110 | 1001 | E9 |

*(Continued on next slide)*

6

## Coding of Alphabetic and Numeric Characters in EBCDIC

*(Continued from previous slide..)*

| Character | EBCDIC Code | | Hexadecimal Equivalent |
|---|---|---|---|
| | Digit | Zone | |
| 0 | 1111 | 0000 | F0 |
| 1 | 1111 | 0001 | F1 |
| 2 | 1111 | 0010 | F2 |
| 3 | 1111 | 0011 | F3 |
| 4 | 1111 | 0100 | F4 |
| 5 | 1111 | 0101 | F5 |
| 6 | 1111 | 0110 | F6 |
| 7 | 1111 | 0111 | F7 |
| 8 | 1111 | 1000 | F8 |
| 9 | 1111 | 1001 | F9 |

## Zoned Decimal Numbers

§ Zoned decimal numbers are used to represent numeric values (positive, negative, or unsigned) in EBCDIC

§ A sign indicator (C for plus, D for minus, and F for unsigned) is used in the zone position of the rightmost digit

§ Zones for all other digits remain as F, the zone value for numeric characters in EBCDIC

§ In zoned format, there is only one digit per byte

# Examples Zoned Decimal Numbers

| Numeric Value | EBCDIC | Sign Indicator |
|---|---|---|
| 345 | F3F4F5 | F for unsigned |
| +345 | F3F4C5 | C for positive |
| -345 | F3F4D5 | D for negative |

# Packed Decimal Numbers

§ Packed decimal numbers are formed from zoned decimal numbers in the following manner:

Step 1: The zone half and the digit half of the rightmost byte are reversed

Step 2: All remaining zones are dropped out

§ Packed decimal format requires fewer number of bytes than zoned decimal format for representing a number

§ Numbers represented in packed decimal format can be used for arithmetic operations

8

## Examples of Conversion of Zoned Decimal Numbers to Packed Decimal Format

| Numeric Value | EBCDIC | Sign Indicator |
|---|---|---|
| 345 | F3F4F5 | 345F |
| +345 | F3F4C5 | 345C |
| -345 | F3F4D5 | 345D |
| 3456 | F3F4F5F6 | 03456F |

## EBCDIC Coding Scheme

**Example**

Using binary notation, write EBCDIC coding for the word BIT. How many bytes are required for this representation?

**Solution:**

B = 1100 0010 in EBCDIC binary notation
I = 1100 1001 in EBCDIC binary notation
T = 1110 0011 in EBCDIC binary notation

Hence, EBCDIC coding for the word BIT in binary notation will be

11000010    11001001    11100011
    B            I            T

3 bytes will be required for this representation because each letter requires 1 byte (or 8 bits)

9

# ASCII

§ ASCII stands for **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange.

§ ASCII is of two types – ASCII-7 and ASCII-8

§ ASCII-7 uses 7 bits to represent a symbol and can represent 128 ($2^7$) different characters

§ ASCII-8 uses 8 bits to represent a symbol and can represent 256 ($2^8$) different characters

§ First 128 characters in ASCII-7 and ASCII-8 are same

# Coding of Numeric and Alphabetic Characters in ASCII

| Character | ASCII-7 / ASCII-8 | | Hexadecimal Equivalent |
|---|---|---|---|
| | Zone | Digit | |
| 0 | 0011 | 0000 | 30 |
| 1 | 0011 | 0001 | 31 |
| 2 | 0011 | 0010 | 32 |
| 3 | 0011 | 0011 | 33 |
| 4 | 0011 | 0100 | 34 |
| 5 | 0011 | 0101 | 35 |
| 6 | 0011 | 0110 | 36 |
| 7 | 0011 | 0111 | 37 |
| 8 | 0011 | 1000 | 38 |
| 9 | 0011 | 1001 | 39 |

*(Continued on next slide)*

10

## Coding of Numeric and Alphabetic Characters in ASCII

*(Continued from previous slide..)*

| Character | ASCII-7 / ASCII-8 | | Hexadecimal Equivalent |
|-----------|------|-------|------------------------|
| | Zone | Digit | |
| A | 0100 | 0001 | 41 |
| B | 0100 | 0010 | 42 |
| C | 0100 | 0011 | 43 |
| D | 0100 | 0100 | 44 |
| E | 0100 | 0101 | 45 |
| F | 0100 | 0110 | 46 |
| G | 0100 | 0111 | 47 |
| H | 0100 | 1000 | 48 |
| I | 0100 | 1001 | 49 |
| J | 0100 | 1010 | 4A |
| K | 0100 | 1011 | 4B |
| L | 0100 | 1100 | 4C |
| M | 0100 | 1101 | 4D |

*(Continued on next slide)*

## Coding of Numeric and Alphabetic Characters in ASCII

*(Continued from previous slide..)*

| Character | ASCII-7 / ASCII-8 | | Hexadecimal Equivalent |
|-----------|------|-------|------------------------|
| | Zone | Digit | |
| N | 0100 | 1110 | 4E |
| O | 0100 | 1111 | 4F |
| P | 0101 | 0000 | 50 |
| Q | 0101 | 0001 | 51 |
| R | 0101 | 0010 | 52 |
| S | 0101 | 0011 | 53 |
| T | 0101 | 0100 | 54 |
| U | 0101 | 0101 | 55 |
| V | 0101 | 0110 | 56 |
| W | 0101 | 0111 | 57 |
| X | 0101 | 1000 | 58 |
| Y | 0101 | 1001 | 59 |
| Z | 0101 | 1010 | 5A |

11

# ASCII-7 Coding Scheme

**Example**

Write binary coding for the word BOY in ASCII-7. How many bytes are required for this representation?

**Solution:**

B = 1000010 in ASCII-7 binary notation
O = 1001111 in ASCII-7 binary notation
Y = 1011001 in ASCII-7 binary notation

Hence, binary coding for the word BOY in ASCII-7 will be

| 1000010 | 1001111 | 1011001 |
|---------|---------|---------|
| B | O | Y |

Since each character in ASCII-7 requires one byte for its representation and there are 3 characters in the word BOY, 3 bytes will be required for this representation

# ASCII-8 Coding Scheme

**Example**

Write binary coding for the word SKY in ASCII-8. How many bytes are required for this representation?

**Solution:**

S = 01010011 in ASCII-8 binary notation
K = 01001011 in ASCII-8 binary notation
Y = 01011001 in ASCII-8 binary notation

Hence, binary coding for the word SKY in ASCII-8 will be

| 01010011 | 01001011 | 01011001 |
|----------|----------|----------|
| S | K | Y |

Since each character in ASCII-8 requires one byte for its representation and there are 3 characters in the word SKY, 3 bytes will be required for this representation

# Unicode

- § **Why Unicode:**
  - § No single encoding system supports all languages
  - § Different encoding systems conflict

- § **Unicode features:**
  - § Provides a consistent way of encoding multilingual plain text
  - § Defines codes for characters used in all major languages of the world
  - § Defines codes for special characters, mathematical symbols, technical symbols, and diacritics

---

# Unicode

- § **Unicode features (continued):**
  - § Capacity to encode as many as a million characters
  - § Assigns each character a unique numeric value and name
  - § Reserves a part of the code space for private use
  - § Affords simplicity and consistency of ASCII, even corresponding characters have same code
  - § Specifies an algorithm for the presentation of text with bi-directional behavior
- § **Encoding Forms**
  - § UTF-8, UTF-16, UTF-32

# Collating Sequence

§ Collating sequence defines the assigned ordering among the characters used by a computer

§ Collating sequence may vary, depending on the type of computer code used by a particular computer

§ In most computers, collating sequences follow the following rules:

1. Letters are considered in alphabetic order
   (A < B < C ... < Z)

2. Digits are considered in numeric order
   (0 < 1 < 2 ... < 9)

---

# Sorting in EBCDIC

**Example**

Suppose a computer uses EBCDIC as its internal representation of characters. In which order will this computer sort the strings 23, A1, 1A?

**Solution:**

In EBCDIC, numeric characters are treated to be greater than alphabetic characters. Hence, in the said computer, numeric characters will be placed after alphabetic characters and the given string will be treated as:

A1 < 1A < 23

Therefore, the sorted sequence will be: A1, 1A, 23.

14

# Sorting in ASCII

**Example**

Suppose a computer uses ASCII for its internal representation of characters.  In which order will this computer sort the strings 23, A1, 1A, a2, 2a, aA, and Aa?

**Solution:**

In ASCII, numeric characters are treated to be less than alphabetic characters.  Hence, in the said computer, numeric characters will be placed before alphabetic characters and the given string will be treated as:

1A < 23 < 2a < A1 < Aa < a2 < aA

Therefore, the sorted sequence will be: 1A, 23, 2a, A1, Aa, a2, and aA

---

# Key Words/Phrases

- § Alphabetic data
- § Alphanumeric data
- § American Standard Code for Information Interchange (ASCII)
- § Binary Coded Decimal (BCD) code
- § Byte
- § Collating sequence
- § Computer codes
- § Control characters
- § Extended Binary-Coded Decimal Interchange Code (EBCDIC)
- § Hexadecimal equivalent
- § Numeric data
- § Octal equivalent
- § Packed decimal numbers
- § Unicode
- § Zoned decimal numbers

15