

### 1 主記憶装置の構成

- 主記憶装置 — プログラムやデータを記憶するための番地のつけられた記憶セル (語) の集合.
- ビット (*bit*, *b*) — データの最小単位,  $2^{16} = 65,536 = 64K$ ,  $2^{32} = 4,294,967,296 = 4G$
- 最上位ビット (*Most Significant Bit*, MSB), 最下位ビット (*Least Significant Bit*, LSB) — 1 語の左端・右端のビット
- 語長 — 記憶セルの大きさ, バイト (*byte*, *B*) — 1 文字を格納するための単位, 通常は 8 ビット
- パリティビット (*parity bit*) — 信頼性向上のために付加されたパリティチェック用ビット.
- オクテット (*octet*) — byte が必ずしも 8 ビットを表すとは限らないので, ネットワークの世界では 8 ビットをオクテットと呼ぶ.

### 2 数の表現

#### 2.1 整数 (integer), 自然数

1. 符号 + 絶対値表現 (*signed magnitude representation*) — 符号に 1 ビット割り当てる.
2. 1 の補数表現 (*one's complement representation*) — 絶対値を 2 進数で表現し, 負の数であれば 0 と 1 を反転.
3. 2 の補数表現 (*two's complement representation*) — 1 の補数表現に 1 を加える.
4. *n* 増し表現 (*n-biased representation*) — 正のみ

[練習問題] 上記の整数表現法を比較せよ.

#### 2.2 実数 (real number) — IEEE, URR

$(-1)^s \times m \times b^e$  ただし, *m*: 仮数 (*mantissa*, *significand*), *b*: 基数 (*base*), *e*: 指数 (*exponent*).

1. 固定小数点表示 (*fixed point number*) — 整数もこの表現法の 1 種
2. 浮動小数点表示 (*floating point number*)  
仮数の MSB が 1 である浮動小数点表現を正規化 (*normalized*) 浮動小数点表現と言う. (下線部は 2 進数)  
 $15 = \underline{1.111} \times 2^{11}$ ,  $-129 = -\underline{1.0000001} \times 2^{111}$ ,  
 $0.125 = \underline{1.0} \times 2^{-11}$ ,  $0.015625 = \underline{1.0} \times 2^{-110}$

浮動小数点表現の標準規格: IEEE 754 floating-point standard

32 ビット表現

1 ビット	8 ビット	23 ビット
符号 <i>s</i>	指数 <i>e</i>	仮数 <i>m</i>

は  $(-1)^s \times (1 + m) \times 2^{(e-127)}$  を表現.

$$\left\{ \begin{array}{l} e = 255, m \neq 0: \text{非数}(Nan) \\ e = 255, m = 0: (-1)^s \times \infty \\ 0 < e < 255: (-1)^s \times 2^{e-127} \times \underline{1.m} \\ e = 0, m \neq 0: (-1)^s \times 2^{-126} \times \underline{0.m} \\ e = 0, m = 0: (-1)^s \times 0 \end{array} \right.$$

64 ビット表現

1 ビット	11 ビット	52 ビット
符号	指数	仮数

は  $(-1)^s \times (1 + m) \times 2^{(e-1023)}$  を表現.

### 2.3 IEEE 754 での例外処理

- 桁あふれ (*overflow*): 演算結果が決められたビット数で表現できず.
- アンダーフロー (*underflow*): 値の絶対値が小さすぎて表現できず.

### 3 情報交換用符号

- ASCII 符号 (American Standard Code for Information Interchange) — 7 ビット符号
- EBCDIC 符号 (Extended Binary Coded Decimal Interchange Code) — 8 ビット符号
- JIS8 ビット符号 — ASCII 符号 + 半角かな文字
- JIS 漢文字符 — 2 バイト使用 JIS X 0208 ('83) ASCII 符号とは Shift-In/Shift-Out の escape sequence (ESC + 2 バイト) により区別.
- EUC-JP 符号 (Extended Unix Code) — 8 ビット. MSB セットで 2 バイト符号. ISO-2022. (G0: ASCII, G1: JIS X 0208-1983, JIS X 208-1990).
- ISO-2022-JP 符号 — MIME 文字セット. ESC ( B ⇒ ASCII, ESC ( J ⇒ JIS X 0201-1976 (ローマ字), ESC \$ @ ⇒ JIS X 0208-1978 (7 ビット × 2), ESC \$ B ⇒ JIS X 0208-1983 (7 ビット × 2).
- シフト JIS 漢文字符 — 2 バイト (MS-DOS 用)
- UNICODE — 2 バイト (Windows NT, ISO10646)

### 4 バイト順序 (Byte Order) .... 要注意

“big-end-first”

バイト番号			
0	1	2	3

*Big-Endian*

例: IBM370, HP, SUN, PC/RT, MC, News, SGI

“little-end-first”

バイト番号			
3	2	1	0

*Little-Endian*

例: Vax, DECstation, 80x86

#### 4.1 ネットワークバイト順序 (Network Byte Order) ..... Big Endian

htons()/htonl()/ntohs()/ntohl()

### 5 抽象データ型

1. カプセル化 — データ構造と演算を一体化, クラス / インスタンス
2. クラス — クラス / サブクラス
3. 継承 (インヘリタンス)
  - オブジェクト指向 — 1,2,3