計算機システムの基礎（第5回配布）

第2章データ表現

（1）数値データ（numeric data)
　　整数(integer), 浮動小数点数(floating point number)等
（2）非数値データ（nonnumeric data）
　　文字コード、論理データ、制御コードなど

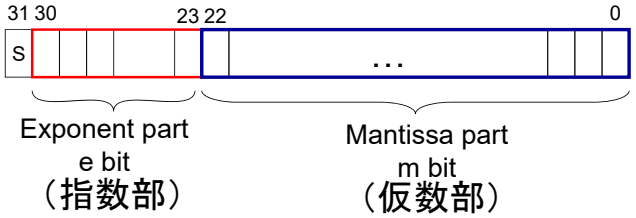
担当：　福井大学　大学院工学研究科
情報・メディア工学専攻
森　眞一郎　(moris@u-fukui.ac.jp)

4. 浮動小数点数

- 浮動小数点表現の基本概念
  - 正規化表現　[$(-1)^S \times M \times 2^E$形式]
    (科学記数法[有効桁数が重要])
  - 整数部1桁（0以外の数値）＋小数部k桁　（以下はk=5の例）

1.00111×2^4　　-1.10100×2^3　　1.01011×2^{-120}

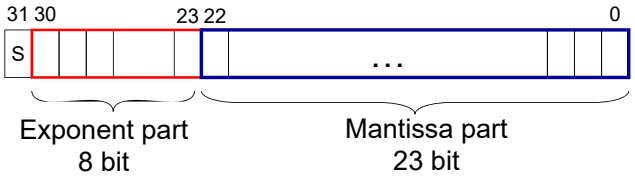
32bit浮動小数点表現の例



4. 浮動小数点数

- IEEE754　32ビット（単精度）浮動小数点表現

$(-1)^S \times 1.F \times 2^{E-127}$



E，Fともに（見かけ上）符号無し整数として操作可能

4. 浮動小数点数

- IEEE　32ビット浮動小数点表現

$(-1)^S \times 1.F \times 2^{E-127}$

指数部 E（10進数）	バイアス	指数表現（e）	
11111111(255)		128	無限大，非数
11111110(254)		127	
⋮		⋮	
10000000(128)	−127	1	通常数の表現は この範囲を使用
⋮		⋮	
00000001(1)		−126	
00000000(0)		−127	ゼロ，非正規化数

図 5.4　指数部の表現（通称：下駄履き表現）

実数の表現（2進数表記）

- IEEE　32ビット浮動小数点表現

$(-1)^S \times 1.F \times 2^{E-127}$

「固定小数点表現」に対する利点・欠点は？
- ⇒ 表現可能な実数の範囲（ダイナミックレンジ）が広い
- ⇒ 有効数字の桁数が少ない
- ⇒ 加減算では毎回桁合わせが必要

「下駄履き表現」の利点は？
- ⇒ 浮動小数点表現された実数の大小比較が
  指数部Eを符号無し整数とみなした
  比較で可能になる。

実数の表現（2進数表記）

- IEEE　32ビット浮動小数点表現

$(-1)^S \times 1.F \times 2^{E-127}$

eの値	数の表現
$e = 128$	$F = 0$のとき　$(-1)^S \infty$ $F \neq 0$のとき　非数NaN
$e = 127 \sim -126$	$(-1)^S (1.F) \times 2^e$
$e = -127$	$F = 0$のとき　$(-1)^S 0$ $F \neq 0$のとき　$(-1)^S 2^{-126} (0.F)$　非正規化数

図 5.5　IEEE 形式浮動小数点数の表現

(32bit)

実数の表現（2進数表記）

- IEEE　32ビット浮動小数点表現

$(-1)^S \times 1.F \times 2^{E-127}$

正規化数として表現できる
- (絶対値が)最も小さな値 ….. 1.0×2^{-126}
- (絶対値が)最も大きな値 ….. $(2.0 - 2^{-23}) \times 2^{127}$

非正規化数を用い表現できる
- (絶対値が)最も小さな値 ….. 1.0×2^{-149}
- (絶対値が)最も大きな値 ….. $(1.0 - 2^{-23}) \times 2^{127}$

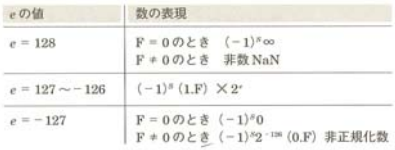
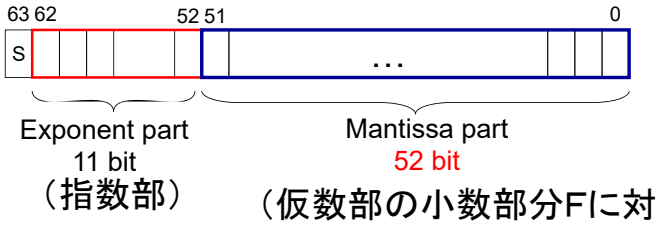


図 5.5　IEEE 形式浮動小数点数の表現

実数の表現（2進数表記）

- IEEE　64ビット浮動小数点表現

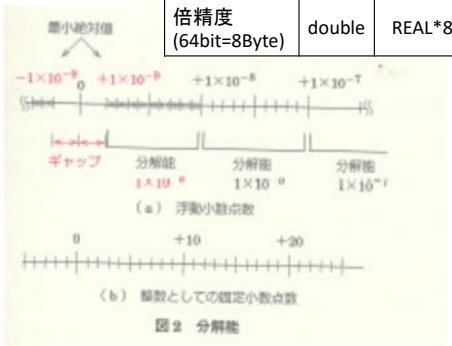
$(-1)^S \times 1.F \times 2^{E-1023}$



4. 浮動小数点数

- プログラム言語での宣言

	C言語	FORTRAN	（正規化数として）表現可能な		仮数部の有効桁数
			絶対値最大の数	絶対値最小の数	
単精度 (32bit=4Byte)	float	REAL	$(2.0-2^{-23})\times 2^{127}$	1.0×2^{-126}	7
倍精度 (64bit=8Byte)	double	REAL*8	$(2.0-2^{-52})\times 2^{1023}$	1.0×2^{-1022}	15

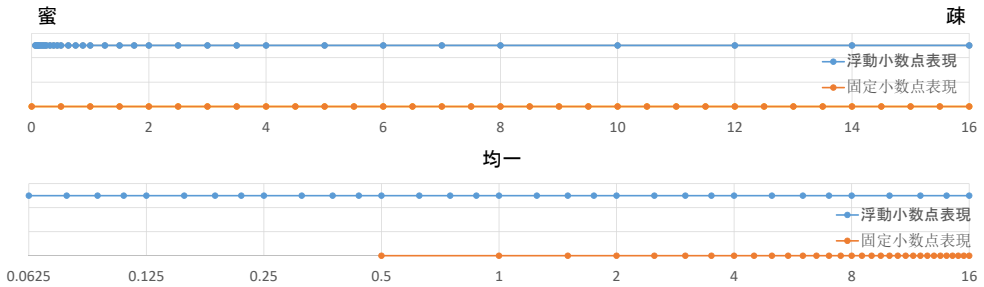


$2^{23} = 2^3 \times 2^{10} \times 2^{10} = 8 \times 1024 \times 1024$

$2^{52} = 2^2 \times 2^{50} = 4 \times (1024)^5$

『補足資料』　同一ビット数の　浮動小数点表現と固定小数点表現で
表現できる数値の関係（一例） [2進数表現Version]

（教科書　P.053の図2は「雰囲気」はわかりますが厳密には不正確です）

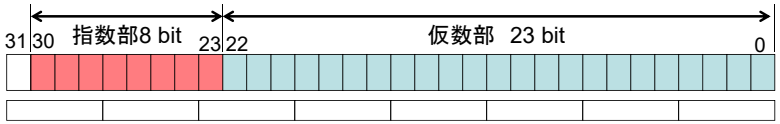


（固定小数点表現では　0　も表現できるが、横軸対数メモリのグラフでは表現できないので省略してあります）

これらは5bit の例ですが、厳密には右端の16（33個目）は表現できませんが見やすくするために追加しています

- IEEE　32ビット浮動小数点表現　(具体例)

$(-1)^S \times 1.F \times 2^{E-127}$



$-0.125 = (-1)^1 \times 0.001_{(2)} = (-1)^1 \times 1.0_{(2)} \times 2^{-3} = (-1)^1 \times 1.0_{(2)} \times 2^{124-127}$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B		E		0		0		0		0		0		0																	

3		F		4		0		0		0		0		0	
0	0	1	1	1	1	1	1	0	1	0	0	0	0	0	0

$(-1)^0 \times 1.\ 1_{(2)} \times 2^{126-127} = 1.\ 1_{(2)} \times 2^{-1} = 0.11_{(2)} = 0.5+0.25 = 0.75$

$(= 11_{(2)} \times 2^{-2} = 3 \times 0.25 = 0.75)$ と考えてもOK

1は　$0.125 \times 8 = 0.125 \times 2^3$

2.5は　$5 \times 0.5 = 5 \times 2^{-1}$

- IEEE　32ビット浮動小数点表現(具体例)

$(-1)^S \times 1.F \times 2^{E-127}$

$0.75_{(10)} = 0.5_{(10)} + 0.25_{(10)} = 0.11_{(2)} = (-1)^0 \times 1.1 \times 2^{126-127}$

3F40 0000

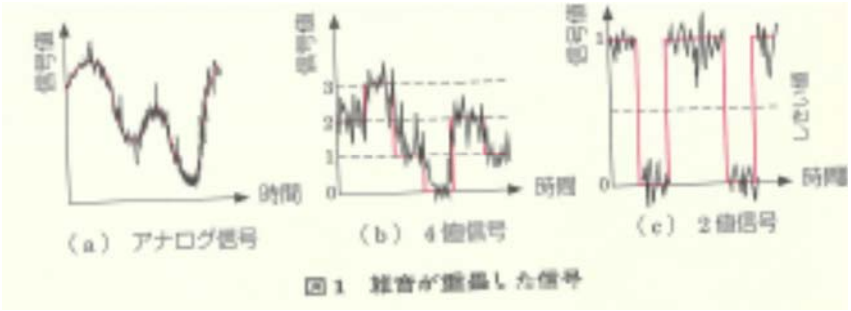
$5.0_{(10)} = 101.0_{(2)} = (-1)^0 \times 1.01 \times 2^{129-127}$

40A0 0000

5. データ表現の長所と問題点

2進数が用いられている理由

信頼性、経済性
情報理論的側面 …. 本当は e進数がベスト！
演算時の回路設計の容易さ



5. データ表現の長所と問題点

有限桁の問題

[1]丸め誤差

$1.01 \times 2^5 + 1.00 \times 2^2 = 1.011 \times 2^5$
だけど….
計算前後 で 有効桁数を
変えることはできない！

(a)絶対値の小さい値に丸める
(b)絶対値の大きい値に丸める
(c)切り捨てて丸める
(d)四捨五入を行ない丸める
…..

例えば先週でてきた 循環小数の
打ち切り誤差も この部類

$0.725d = 0.101\dot{1}10\dot{0}$

[2]桁落ち

絶対値がほぼ等しく符号が同じで
丸め誤差を持つ数字同士の減算

$1.001 \times 2^5 - 1.000 \times 2^5 = 1.000 \times 2^2$
だけど….
右辺は正規化のために1.000と
かかれているが**少数点以下の3桁**
は便宜上埋めただけで、誤差を含
んでいる

[3]情報落ち

絶対値が大きな数Aと絶対値が小さな数Bの加減算の結果Bが無視されてしまう。

有効桁数3桁の場合
$(1.01 \times 2^5 + 1.00 \times 2^1)$
$-1.01 \times 2^5 = 1.00 \times 2^1$
とはならず 結果は 0 になってしまう。

( )内を計算した時点で B に相当する部分が丸められてしまう。

6. コード

計算機内での文字コード

ASCIIコード
半角英数、記号、制御文字を表現する
7bitコード
（1Byteに合わせて8bitに拡張して扱うことが多い[半角文字]）

漢字コード
日本語文字を表現するための2バイト(16bit)コードJIS7(JIS)、JIS8（SJIS)、EUC(Extended Unix Code)等の複数の符号化方式が混在して使われています。
（メール等では、従来はJIS7を使用することを推奨）

UNICODE
世界中の言語を1つのコード体系で表現しようと試みたプロジェクト （UTF8など）

6. コード

その他の文字コード

2進化10進コード
（BCD: Binary Corded Dicimal)

10進数1桁 を 2進数4桁 を用いて表すコード

10進数	BCDコード
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

グレイコード

隣り合うコードで0と1が異なる場所が
つねに1箇所だけとなるように符合を割り当てたコード

10進数	グレイコード（例）
0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

6. コード

教科書P.70は符号の構成方など複数の間違いがあります。

誤り検出・訂正が可能なコード

パリティ符号

元の符号に、パリティ情報（1の数の偶奇に関する情報）を付加して、1ビットの誤りを検出可能にした符号

10進数	BCDコード	偶パリティ付BCDコード
0	0000	00000
1	0001	00011
2	0010	00101
3	0011	00110
4	0100	01001
5	0101	01010
6	0110	01100
7	0111	01111
8	1000	10001
9	1001	10010

ハミングコード

元の符号に、付加情報を追加して、1ビットの誤り訂正（Error Correction)を可能にした符号

(4,7)符号の例

7	6	5	4	3	2	1	How to calculate parity bits
D_7	D_6	D_5	P_4	D_3	P_2	P_1	This represents the full codeword
D_7	D_6	D_5	P_4				P_4 – Even parity of D_7 D_6 D_5
D_7	D_6			D_3	P_2		P_2 – Even parity of D_7 D_6 D_3
D_7		D_5		D_3		P_1	P_1 – Even parity of D_7 D_5 D_3

(2,5)符号の例