

コンピュータ科学II

担当：武田敦志 <takeda@cs.tohoku-gakuin.ac.jp>
http://takeda.cs.tohoku-gakuin.ac.jp/

データ表現(1)

- コンピュータにおけるデータ表現
コンピュータで扱う**全てのデータ**は
2進数の整数値で表現する

整数：100 → データ：1100100₂

小数：2.5
小数：0.625 × 2³ } データ：0.101₂ × 11₂
※本当は少し異なる表現をする

文字：A → データ：01000001₂

IPアドレス：192.168.0.254
→ データ：11000000101010000000000001111111₂

データ表現(2)

■データの長さ

データの長さを『bit』で表す
n桁の2進数整数値の長さは『n bit』となる

7 = 111₂ → 3bit

100 = 1100100₂ → 7bit

通常は決まった長さのデータを使う
(例えば 8bit)

7 = 0000111₂ → 8bit

100 = 01100100₂ → 8bit

2進数の計算(1)

■計算問題 – 加算

(1) 197 + 246

(2) 101101₂ + 1101110₂

(3) 2時間42分 + 1時間56分

2進数の計算(2)

■10進数の加算

(1) $197 + 246$

$$\begin{array}{r}
 197 \\
 + 246 \\
 \hline
 13
 \end{array}
 \rightarrow
 \begin{array}{r}
 13 \div 10 \text{の商} \\
 1 \\
 197 \\
 + 246 \\
 \hline
 143 \\
 13 \div 10 \text{の余り}
 \end{array}
 \rightarrow
 \begin{array}{r}
 14 \div 10 \text{の商} \\
 1 \\
 197 \\
 + 246 \\
 \hline
 443 \\
 14 \div 10 \text{の余り}
 \end{array}$$

page 5

2進数の計算(3)

■60進数の加算

(3) 2時間42分 + 1時間56分

$$\begin{array}{r}
 2 \text{ 時間 } 42 \text{ 分} \\
 + 1 \text{ 時間 } 56 \text{ 分} \\
 \hline
 98 \text{ 分}
 \end{array}
 \rightarrow
 \begin{array}{r}
 98 \div 60 \text{の商} \\
 1 \text{ 時間} \\
 2 \text{ 時間 } 42 \text{ 分} \\
 + 1 \text{ 時間 } 56 \text{ 分} \\
 \hline
 4 \text{ 時間 } 38 \text{ 分} \\
 98 \div 60 \text{の余り}
 \end{array}$$

page 6

2進数の計算(4)

■2進数の加算

(2) $101101_2 + 1101110_2$

$$\begin{array}{r}
 101101 \\
 + 1101110 \\
 \hline
 211
 \end{array}
 \rightarrow
 \begin{array}{r}
 2 \div 2 \text{の商} \\
 1 \\
 101101 \\
 + 1101110 \\
 \hline
 3011 \\
 2 \div 2 \text{の余り}
 \end{array}
 \rightarrow
 \begin{array}{r}
 1 \\
 101101 \\
 + 1101110 \\
 \hline
 11011
 \end{array}$$

page 7

2進数の計算(5)

■加算・乗算・除算

計算方法は10進数の計算方法と同じ

基数が変わっても計算方法は変わらない

乗算： 46×27

$$\begin{array}{r}
 46 \\
 \times 27 \\
 \hline
 322 \\
 92 \\
 \hline
 1242
 \end{array}$$

除算： $106 \div 4$

$$\begin{array}{r}
 26 \\
 4 \overline{) 106} \\
 \underline{8} \\
 26 \\
 \underline{24} \\
 2
 \end{array}$$

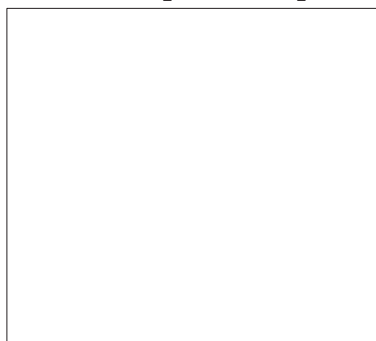
page 8

2進数の計算(6)

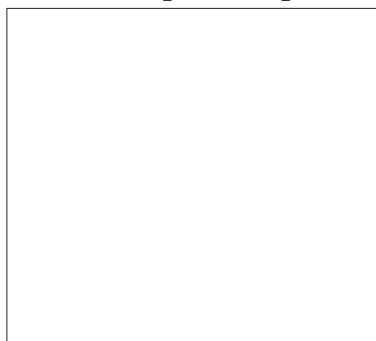
■計算問題 – 乗算・除算

次の式を計算せよ

(1) $10110_2 \times 1011_2$



(2) $11010_2 \div 101_2$



page 9

2の補数表現(1)

■コンピュータにおける除算

加算回路を使って除算する

正の整数を除算 → 負の整数を加算

$35 - 13 = 22$ $35 + (-13) = 22$

負の整数をどのように表現するか？

コンピュータにおける負数の表現方法 = 2の補数表現

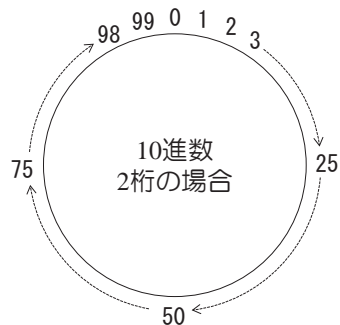
page 10

2の補数表現(2)

■2の補数表現の考え方

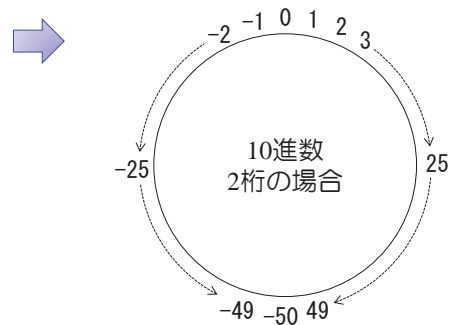
(1) 桁数を設定

円形の数直線ができる



(2) 負数の領域を設定

数値の大きい領域に対し
逆方向に負数を割り振る

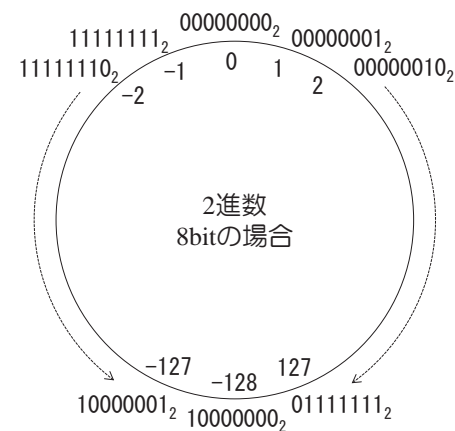


page 11

2の補数表現(3)

■負の数の割り当て

2進数8ビットの場合の数直線



page 12

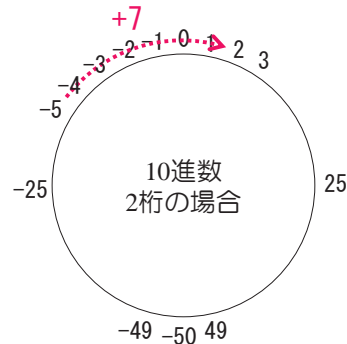
2の補数表現(4)

- 2の補数表現を使った除算
加算回路を用いて計算する

$$7 - 5 = 2 \quad \Rightarrow \quad 7 + (-5) = 2$$

$$\begin{array}{r} 00000111 \\ +) 11111011 \\ \hline 100000010 \end{array}$$

最後の繰り上がりは
結果から消える



page 13

2の補数表現(5)

- 負数表現の計算方法

10進数2桁の場合

値	表現
-3	97
-2	98
-1	99
0	00
1	01
2	02

2進数8bitの場合

値	表現
-3	11111101 ₂
-2	11111110 ₂
-1	11111111 ₂
0	00000000 ₂
1	00000001 ₂
2	00000010 ₂

表現 = 100 - 値の絶対値
(例: 97 = 100 - 03)

表現 = 100000000₂ - 値の絶対値
(例: 11111101₂ = 100000000₂ - 00000011₂)

page 14

2の補数表現(6)

- 負数表現の計算方法 (繰り下がりなし)

10進数2桁の場合

$$\begin{aligned} \text{負数表現} &= 100 - \text{値の絶対値} \\ &= 99 - \text{値の絶対値} + 1 \end{aligned}$$

(例: 97 = 99 - 03 + 1)

2進数8bitの場合

$$\begin{aligned} \text{負数表現} &= 100000000_2 - \text{値の絶対値} \\ &= 11111111_2 - \text{値の絶対値} + 1 \end{aligned}$$

ビットの反転で計算できる

(例: 11111101₂ = 11111111₂ - 00000011₂ + 00000001)

page 15

2の補数表現(7)

- 2の補数表現を使った減算

負数表現は「ビットの反転」と「加算」で計算できる

減算 = 「正の整数」と「負の整数」の加算



2の補数表現を使った(2進数の)減算は
加算回路のみを用いて計算することができる

➡ 加算回路があれば、四則演算のすべてが可能

page 16

2の補数表現(8)

■演習問題

『-9』を2の補数表現(2進数8bit)で示せ

『17 + (-9)』を2の補数表現(2進数8bit)を使って計算せよ

page 17

整数の計算(1)

■コンピュータにおける除算

加算回路を使って除算する

正の整数を除算 → 負の整数を加算

$$35 - 13 = 22 \quad \rightarrow \quad 35 + (-13) = 22$$

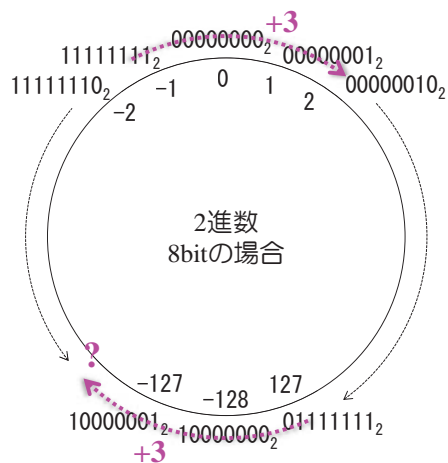
負の整数をどのように表現するか？

コンピュータにおける負数の表現方法 = 2の補数表現

page 18

整数の計算(2)

■2の補数表現



円形の数直線を使って計算



整数の桁数は決まっている
不連続な部分がある

page 19

整数の計算(3)

■桁あふれ

表現できる値の範囲を超えて計算すると
間違った計算結果となる

2進数8bit (2の補数表現) の場合

$$127 + 1 = 01111111_2 + 00000001_2 = 10000000_2 = -128$$

$$127 + 2 = 01111111_2 + 00000010_2 = 10000001_2 = -127$$

$$127 + 3 = 01111111_2 + 00000011_2 = 10000010_2 = -126$$

page 20

整数の計算(4)

■整数計算を行う場合の注意

整数で表現できる値の**最小値**・**最大値**を意識する

最小値～**最大値**の中での計算であれば正しい結果となる

8ビット整数： 16ビット整数：
最小値 = -128 最小値 = -32768
最大値 = 127 最大値 = 32767

32ビット整数：
最小値 = -2147483648
最大値 = 2147483647

page 21

整数の計算(5)

■桁あふれを発生させるプログラム

プログラム (int型は32bit)

```
#include <stdio.h>

int main()
{
    int a = 1;
    int i;

    for(i = 0; i < 13; i++){
        printf("10の%d乗 = %d\n", i, a);
        a = a * 10;
    }
}
```

実行結果

```
10の 0乗 = 1
10の 1乗 = 10
10の 2乗 = 100
10の 3乗 = 1000
10の 4乗 = 10000
10の 5乗 = 100000
10の 6乗 = 1000000
10の 7乗 = 10000000
10の 8乗 = 100000000
10の 9乗 = 1000000000
10の10乗 = 1410065408
10の11乗 = 1215752192
10の12乗 = -727379968
```

page 22

練習問題

■2進数8bit (2の補数表現) の環境で、以下の計算せよ

(1) $120 + 80$

(2) 100×4

page 23

コンピュータにおける小数の表現(1)

■仮数と指数

仮数と指数を使って数値を表現することができる

例：光の速さ (秒速約30万km)

300,000,000 m/s



$\underbrace{3.0}_{\text{仮数}} \times 10^{\underbrace{8}_{\text{指数}}} \text{ m/s}$

page 24

コンピュータにおける小数の表現(2)

■浮動小数点数 (IEEE754)

コンピュータでは、小数を仮数+指数で表現する

例：35.375

$$100011.011_2 \quad \rightarrow \quad \underbrace{1.00011011_2}_{\text{仮数の小数点以下}} \times \underbrace{2^5}_{\text{指数}}$$

単精度 浮動小数点数 (float)

$$\underbrace{01000010000011011000000000000000}_{\text{指数+127}} \quad \underbrace{10110000000000000000}_{\text{仮数の小数点以下}}$$

コンピュータにおける小数の表現(3)

■浮動小数点数の最大値と最小値 (単精度浮動小数点数の場合)

指数の表現範囲：-126 ~ 127

仮数の表現範囲：1 ~ (2 - 2⁻²³)

最小値：1.0 × 2⁻¹²⁶ ≐ 1.175494 × 10⁻³⁸

$$\underbrace{00000001}_{\text{指数}} \underbrace{000000000000000000000000}_{\text{仮数の小数点以下}}$$

最大値：(2 - 2⁻²³) × 2¹²⁷ ≐ 3.402823 × 10³⁸

$$\underbrace{01111110}_{\text{指数}} \underbrace{111111111111111111111111}_{\text{仮数の小数点以下}}$$

コンピュータにおける小数の表現(4)

■2進数の小数 (小数点以下の計算)

2進数から10進数への変換

$$0.1_2 = 1/2$$

$$0.01_2 = 1/4$$

$$0.001_2 = 1/8$$

$$\rightarrow \quad 0.011_2 = 3/8 = 0.375$$

10進数から2進数への変換

$$0.375$$

$$= 375/1000$$

$$= 101110111_2 / 11111010000_2$$

$$= 0.011_2$$

(工夫をすれば簡単に計算可能)

$$\begin{array}{r} \\ 1111101000_2 \overline{) 10111011.000_2} \\ \underline{11111010 _2} \\ 01111101 _2 \\ \underline{1111101 _2} \\ 0_2 \end{array}$$

練習問題

■下記の小数を2進数で表現せよ

(1) 0.625

(2) 9.4375

少数計算に発生する誤差(1)

■2進数の無限小数

10進数と2進数では、無限小数となる値に違いがある

(例)

$$\begin{aligned} 0.1 &= 1/10 \\ &= 0.00011001100\dots_2 \end{aligned}$$

$$\begin{array}{r} 0.0001100110011\dots_2 \\ 1010_2 \overline{) 1.0000000000000_2} \\ \underline{1010_2} \\ 1100_2 \\ \underline{1010_2} \\ 10000_2 \\ \underline{1010_2} \\ 1100_2 \\ \underline{1010_2} \\ 10\dots_2 \end{array}$$

page 29

少数計算に発生する誤差(2)

■丸め誤差

10進数では有限でも、2進数だと無限小数になるものがある

⇔ コンピュータ表現できる小数の長さは有限

(例) 2進数 (小数点以下8bit) で表現する場合

$$\begin{array}{ccc} 0.1 = 0.00011001100\dots_2 & \Rightarrow & 0.00011010_2 = 0.1015625 \\ \text{入力} & & \text{内部表現} \quad \text{出力} \end{array}$$

$$\begin{array}{ccc} 0.2 = 0.00110011000\dots_2 & \Rightarrow & 0.00110011_2 = 0.19921875 \\ \text{入力} & & \text{内部表現} \quad \text{出力} \end{array}$$

page 30

少数計算に発生する誤差(3)

■丸め誤差を確認するプログラム

プログラム

(float型的小数点以下は23bit)
(double型的小数点以下は52bit)

```
#include <stdio.h>

int main()
{
    float a = 1.1;
    double b = 1.1;

    printf("a = %.20f\n", a);
    printf("b = %.20f\n", b);
}
```

実行結果

```
a = 1.10000002384185791016
b = 1.10000000000000008882
```

page 31

少数計算に発生する誤差(4)

■丸め誤差が発生する計算

コンピュータ内的小数の桁数は有限

⇒ 近似値で計算している場合がある

(例) 2進数 (小数点以下8bit) で表現

『(0.2 - 0.1) × 10』を計算する

$$0.2 \Rightarrow 0.00110011_2$$

$$0.1 \Rightarrow 0.00011010_2$$

$$0.2 - 0.1 = 0.00011001_2 = 0.09765625$$

$$(0.2 - 0.1) \times 10 = 0.9765625$$

page 32

少数計算に発生する誤差(5)

■計算誤差が発生するプログラム

プログラム

```
#include <stdio.h>

int main()
{
    float a = 1.0 + 2.0 / 1000.0;
    float b = 1.0 + 1.0 / 1000.0;
    float c = (a - b) * 1000.0;

    printf("2.0 - 1.0 = %.20f¥n", c);
}
```

実行結果

```
(0.002 - 0.001) * 1000 = 0.99992752075195312500
```

練習問題

■次の数値・計算を2進数(小数点以下8bit)で表現せよ

(1) 0.7

(2) $0.7 - 0.5$

(3) $(0.7 / 4 - 0.5 / 4) \times 4$