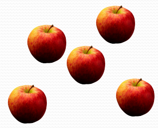
コンピュータ基礎(5)

7 章 情報の表現と基礎理論

数の表現（書き方）

- 「数」と「数の書き方」をわけて考える
  - 「数の書き方」と，「数そのものの性質」は別のもの
    例：13 は素数・・・”13”という書き方とは無関係



5

67

- ここでは書き方（表現方法）について考える

基数法とは？

- 位取り基数法（基数法）
  - 67 = $6\times10^1+7\times10^0$ のような数の表現方法
  - 10 の部分は10でなくても良い．これを　　という．
  - 基数を 2 にしたものを二進数と言う．
    - 二進数の 1 けたを　　，8 けたを　　と呼ぶ．
- 基数法の利点
  - 大きな数でも短く表現出来る（文字数が少ない）
  - ある数字の表現は，ただ 1 通りしかない

日本のそろばん：数の表現が一通り

基数法の性質

- 表現出来る数の範囲（基数を r とする）
  - n けたの r 進数で表せる数は 0 から r^n-1 である
- けたずらし（シフト）
  - r 進数の各けたを左に 1 けたずらすと r 倍になる
    例：10進数で 123 → 1230 にすると10倍
    同様に，2進数で 101 → 1010 にすると2倍
  - r 進数の各けたを右に1けたずらすと 1/r 倍になる
    例：10進数で 1230 → 123 にすると1/10倍
    同様に，2進数で 1010 → 101 にすると½倍
  - この性質を使うと，一番下のけたの値を調べることが出来る
    例：3456 / 10 = 345，余り 6 となる
    同様に，1101 を ½ 倍すると，110　余り 1

基数変換の方法

- 2 進数を10進数に変換
  - 例：1101 = $1\times2^3+1\times2^2+0\times2^1+1\times2^0$ = 13
- 10進数を2進数に変換
  - 2で割った余りを調べ，下の桁から順に並べる
    13 / 2 = 6 余り 1
    6　/ 2 = 3 余り 0
    3　/ 2 = 1 余り 1
    1　/ 2 = 0 余り 1
    となるので，13 は二進数では 1101 になる
  - これは，前のスライドの「2 進数を右に 1 けたシフトすると ½ になる」という性質を使っている
    1101(13) を 2 で割ると 110(6)，余り 1 になる

16進数と8進数

- 10進数と2進数の間の変換は，面倒くさい
  - でも，2進数での表記は，長すぎる
  → 2進数を短く表現出来ないか？
- 2進数を 4 桁ずつ区切って表示する
  - 例：10110111 → 1011 と 0111 に分け，
    それぞれに記号を割り当てて表示しよう！
  - 4 桁の 2 進数は 0~15 の 16 通りなので，
    数字（0~9）では足りない
    → A~F を 1010 $(10)_{10}$ ~ 1111 $(15)_{10}$ に割り当て．
    10110111 は B7 と表現出来る
- これは，　　進数である．
  - 2 進数を 4 桁ずつ区切っているので，2^4 = 16
  - B7 = B(11)$\times16^1+7\times16^0$ = 183

例題

- 10進数の 83 について
  - 8bit の 2進数で表せ.
  - 2桁の16進数で表わせ.

- 2進数の 01101011 について
  - 10進数で表せ.
  - 16進数で表わせ.

- 16進数の 7D について
  - 8bit の2進数で表せ.
  - 10進数で表せ.

負の数の表現

- 8bit(1byte)の２進数は，0~255 を表現出来る
  - 0と正の値しか表現出来ない
  - マイナス(-) の記号を表現出来ないか？
- アイディア１：絶対値表現（　　による方法）

符号ビット
0 なら +
1 なら -

1 0 1 1 0 1 1 0　10進数の-54

数の絶対値

- 問題点
  - 0の表現が二通りできてしまう
    00000000 ・・・+0
    10000000 ・・・ -0

負の数の表現

- アイディア2：　　表現

0 0 1 1 0 1 1 0　10進数の54

0と1を反転

1 1 0 0 1 0 0 1　10進数の-54

符号ビット
0 なら +
1 なら -

- 問題点
  - 0の表現が二通りできてしまう
    00000000 ・・・+0
    11111111 ・・・ -0

負の数の表現

重要!!

- アイディア3：　　表現

0 0 1 1 0 1 1 0　10進数の54

0と1を反転

1 1 0 0 1 0 0 1

1を足す

1 1 0 0 1 0 1 0　10進数の-54

符号ビット
0 なら +
1 なら -

もしも繰り上がりのために桁数が増えたら，その桁は単に捨てる！

- 利点
  - ０の表現が一通りしかない
  - <u>負の数をそのまま加算すると正しい答えが得られる</u>

例題(1)

- -21 を，8bit の２の補数で表現せよ.
  - 21 は，２進数で 00010101
    反転して 11101010
    1を加えて 11101011　となる.

- 34 - 21 を，２の補数を用いて計算せよ.
  - 34 は，２進数で 00100010
  - これに，上で求めた -21 の２の補数を加える
    　　00100010
    ＋　11101011
    　100001101 一番上の桁をとって 00001101
  - これは，10進数では 13 になる.

例題(2)

- -113 を，8bit の２の補数で表現せよ.

- 57 - 31 を，２の補数を用いて計算せよ.

0の２の補数を計算してみる

- ０の２の補数を求める
  - 00000000
    これを反転して 11111111
    これに1を加えて 100000000
  - 最上位桁をすてると 00000000
- よって，0の表現は１通りしか存在しない
- n桁の２の補数による値の表現範囲
  - $-2^{(n-1)}$ から $2^{(n-1)}-1$ まで
  - 例：8bit の場合，-128 から 127 まで(256通り)

なぜ引き算ができるのか？(1)

- 10進数で 845 - 268 を計算することを考える
  - 繰り下がり計算が面倒！999から引くなら楽なのに！
- なんとかして 999 を使ってみる
  - 845 – 268
    = 845 + (999 – 268) – 999
    = 845 + (999 – 268) – 1000 + 1
    = 845 + (999 – 268 + 1) – 1000
    とすると計算できる.
- 999 から 268 を引算するのは，各桁の反転に相当
  - 0←→9, 1←→8, 2←→7, 3←→6, 4←→5 で置換
  - なので，上の括弧内の計算(999-268+1)は
    各桁を反転して１を足すことに相当する

なぜ引き算ができるのか？(2)

- ２進数の反転計算
  - 11111111 から引くことと同じ.
- ２の補数
  - 反転してから１を加えるので，100000000 から引くことに相当する
  - 桁あふれは無視するので，100000000 を足したり引いたりするのは，何もしないのと同じ.
- つまり，２の補数とは
  - 事前に引き算をしておいた値のようなもの.

小数の表現(1)

- 桁ずらし（シフト演算）について
  - 10進数の 123 を右に１桁ずらすと 12.3 となる.
    この値は 123 の 1/10 である.
  - 同様に，２進数の 1101 を右に一桁ずらすと
    110.1 となる．これは 1101 $(13)_{10}$ の ½ である.
    つまり，6.5 である.
  - これを，　　　　　と呼ぶ.
- 基数による解釈
  - 110.1 は $1\times2^2+1\times2^1+0\times2^0+1\times2^{-1} = 6.5$
  - 小数点以下の各桁の重みは，0.5, 0.25, 0.125, ..
- 割り切れない数について
  - 例えば，0.2 は２進数では循環小数になる
    0.00110011001100…

例題

- 2進数の固定小数， 110.011 について
  - 10進数で表わせ.
- 10進数の小数，7.825 について
  - 2進数で表せ.

- 非常に大きい数の表現
  - 例：光の速度　3×10^8 =300000000 [m/s]
  - このように，桁をずらす桁数（指数）を使うことで非常に大きい数や小さな数を表現することが出来る
  - C言語では，float や double で使われている
    - float：単精度浮動小数点数
    - double：倍精度浮動小数点数
- ２進数では
  - １桁ずらすと，２倍したことになるので，
    $a \times 2^b$ のような表現になる
  - a を　　　，bを　　　と呼ぶ.

浮動小数点数の規格

- IEEE754 として規格化されている

s	指数部(e)	仮数部(m)

  - float :全体で32bit
    - 符号ビット:1bit, 指数部:8bit, 仮数部:23bit の計32bit
  - double : 全体で64bit
    - 符号ビット:1bit, 指数部:11bit, 仮数部:52bit の計64bit
- 値は，以下の式で計算できる
  - float : $(-1)^s \times 2^{e-127} \times (1+m)$
  - double : $(-1)^s \times 2^{e-1023} \times (1+m)$
  - s が 1 なら負の数である．

10進数の表現

- コンピュータは，2進数と10進数を変換している
  - 人に計算結果を見せるため(2→10)
  - プログラムをコンパイルするとき(10→2)
- 10進数も，なんらかの方法で表現する必要がある
- 10進数の一桁は， 2 進数の4bit で表すことが出来る．
  - パック10進数という．23 なら “0010 0011”
  - ２進数でも，１６進数でもないことに注意！！
- 10進数の一桁を，8bitで表すこともある．
  - ゾーン10進数という．

文字コード

- 文字を２進数で表すには？
  - a~z なら26 通り
    大文字/小文字に，10個の数字を加えて62 通り
    記号 ! “ # $ % & ‘ ( ) + - * / = ^ ~ ; : { } < > ?
  - ・・などを考えると，キリがいい所で 8bit にしよう．
    - 8bitなら，256 種類の文字が使える．
- 文字１つ１つに，数値を割り当てる．
  - 文字‘0’ には 48, 文字‘A’ は 65, というふうに．
  - 0番から31番は特殊な用途に使われている．
    改行記号，１文字消去，などなど．
- C言語の char 型変数は，8bit の変数．

ASCII コード/ JISコード

- コード
  - 7bit にアルファベット・数字・記号を入れたもの
  - 現在は，ほとんどのコンピュータで使われている
- コード(JIS X0201)
  - 残った半分にカタカナを入れたもの（半角カナ）
  - 濁点も１文字
  - ひらがな，漢字は表現できない

上位4bit

下位4bit

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		DE	SP	0	@	P		p				ー	タ	ミ		
1	SH	D1	!	1	A	Q	a	q			。	ア	チ	ム		円
2	SX	D2	”	2	B	R	b	r			「	イ	ツ	メ		年
3	EX	D3	#	3	C	S	c	s			」	ウ	テ	モ		月
4	ET	D4	$	4	D	T	d	t			、	エ	ト	ヤ		日
5	EQ	NK	%	5	E	U	e	u			・	オ	ナ	ユ		時
6	AK	SN	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		分
7	BL	EB	’	7	G	W	g	w			ァ	キ	ヌ	ラ		秒
8	BS	CN	(	8	H	X	h	x			ィ	ク	ネ	リ		
9	HT	EM	)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A	LF	SB	*	:	J	Z	j	z			ェ	コ	ハ	レ		
B	HM	EC	+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C	CL	→	,	<	L	¥	l	|			ャ	シ	フ	ワ		
D	CR	←	−	=	M	]	m	}			ュ	ス	ヘ	ン		
E	SO	↑	.	>	N	^	n	~			ョ	セ	ホ	゛		
F	SI	↓	/	?	O	_	o				ッ	ソ	マ	゜		

ASCIIコード

例題

- 次の４文字 “STAR” を，ASCII コードで表わせ．
  - １６進数では □□ □□ □□ □□
- 次の16進数は，どのような文字か．
  - 54 6F 77 6E

漢字コード

- 漢字は数千種類も存在する
  - 8bit では表現出来ない・・16bit（２バイト）使う
- 数種類の漢字コードが使われている
  - JIS漢字コード・・通信での標準的な規格
  - Shift-JISコード (Windows での標準)
  - EUCコード (UNIX でよく使われてきた)

  コードが違うと，文字化けの原因に．
- 他の言語（韓国，タイ，・・）の文字は？
  - Unicode が策定され，普及し始めている
  - 多言語の文字を扱うことが出来る