

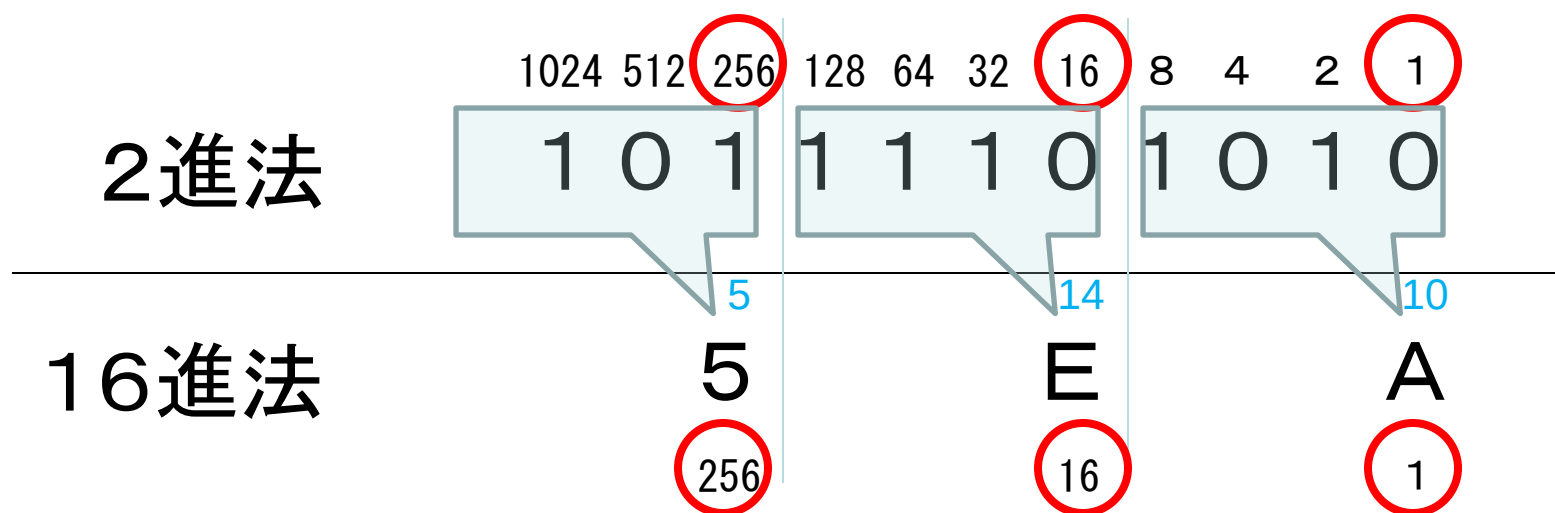
文字のデジタル化

情報の科学 第32回授業

07情報のデジタル化

対応ファイル: 13exp30.xls (前々回)

< 復習 > 2進法と16進法



2進数と16進数では、繰り上がりのタイミングが同じ！
→ 2進4ケタをそのまま16進に変換して表せられる！

< 復習 > 2進法と16進法

2進法

01011101

上位4ビット

5

13

下位4ビット

16進法

5 D

- このように、2進8文字は、16進2文字で表すことができる！！
- 私たち人間にとっては、2進法よりも16進法の方が扱いやすい！！

＜復習＞情報の量

- コンピュータでは、0と1の電気信号に情報を変換、すなわち2進法で処理をしている。
- 2進法の数1ケタを「1 bit (ビット)」とし、情報の量の単位とする。

<復習>「2進法」と「場合の数」

1ケタにつき
0 or 1の
2パターン

2進

1 1 1 1 1

場合の数

$2 \times 2 \times 2 \times 2 \times 2$

通り

5bitの情報量



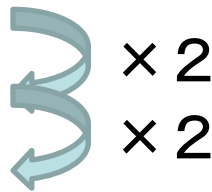
2⁵

2進法での
ケタ数と
同じ

2進法での「ケタ数」が情報の量(bit数)と考えて良い

2進法と場合の数

bit	場合の数
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024



「一つ上のセル」を2倍することをコピーしていけばよい

例) (L5の場所) $\cdots = L4 * 2$

L5を「コピー」、
L6からL28まで「貼り付け」

これらのことから

5bitの情報量では、32 通り
のものが区別でき、

512通りのものを区別するには 9 bit
100通りのものを区別するには 7 bit
の情報量が必要であることがわかる

文字コード

一つひとつの文字に割り当てられた「背番号」
この番号をもとに、文字を表示したり区別する。

$2^7 = 128$ 通り

1963年 7bit・・・ASCIIコード(英数字記号・制御記号)

ASCIIコード

下位ビット

上位ビット

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	000		SH	SX	EX	ET	EQ	AK	BL	BS	HT	LF	HM	CL	CR	SO	SI
10	001	DE	D1	D2	D3	D4	NK	SN	EB	CN	EM	SB	EC	→	←	↑	↓
20	010		!	"	#	\$	%	&	'	(	)	*	+	,	.	/	
30	011	0	1	2	3	4	5	6	7	8	9	:	<	=	>	?	
40	100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	101	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	^	_
60	110		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	111	p	q	r	s	t	u	v	w	x	y	z	{		}	—	

文字コード

一つひとつの文字に割り当てられた「背番号」
この番号をもとに、文字を表示したり区別する。

$2^7 = 128$ 通り

1963年 7bit・・・ASCIIコード(英数字記号・制御記号)

1969年 1Byte(8bit)・・・JIS X 0201<半角>

$2^8 = 256$ 通り

(アスキーコード+カナ文字)

JIS X 0201 (拡張ASCIIコード)

下位ビット

上
位
ビ
ット

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
	0000																
10	0001	DE	D1	D2	D3	D4	NK	SN	EB	CN	EM	SB	EC	→	←	↑	↓
20	0010		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
30	0011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	0100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	0101	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	^	_
60	0110		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	0111	p	q	r	s	t	u	v	w	x	y	z	{		}	—	
80	1000																
90	1001																
A0	1010		。	「	」	、	・	ヲ	ア	イ	ウ	エ	オ	ヤ	ユ	ヨ	ツ
B0	1011	一	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
C0	1100	タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
D0	1101	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	ゝ	。°
E0	1110																
F0	1111																

拡張部分

「漢字」について

常用漢字

・・・日常の使用に必要なものとして定められた漢字

小学校(1006字)

中学校(1130字) 計2136字(2010年)

JIS第一水準(2965字)、第二水準(3390字)

・・・常用漢字も含め、「日本工業規格」で決められた、
コンピュータ等で良く使われる漢字。

文字コード

一つひとつの文字に割り当てられた「背番号」
この番号をもとに、文字を表示したり区別する。

$2^7 = 128$ 通り

1963年 7bit・・・ASCIIコード(英数字記号・制御記号)

1969年 1Byte(8bit)・・・JIS X 0201<半角>

$2^8 = 256$ 通り

(アスキーコード+カナ文字)

1976年 2Byte(16bit)・・・JIS X 0208<全角>

(漢字など)

$2^{16} = 65536$ 通り

1993年 ……Unicode(UCS-2)<多国語処理>

現在 4Byte(32bit)・・・Unicode(UCS-4)、UTF-8

※文字コードが違えば、全く違う「文字や記号」が表示される！
(いわゆる「文字化け」)

文字コード表

☆JISコード表の例

最初の3ケタが左側の数字に、最後の1ケタの数が上の数字に対応している。

→ 最初の3ケタで「行」を探し、残りの1ケタを横に探せばよい。

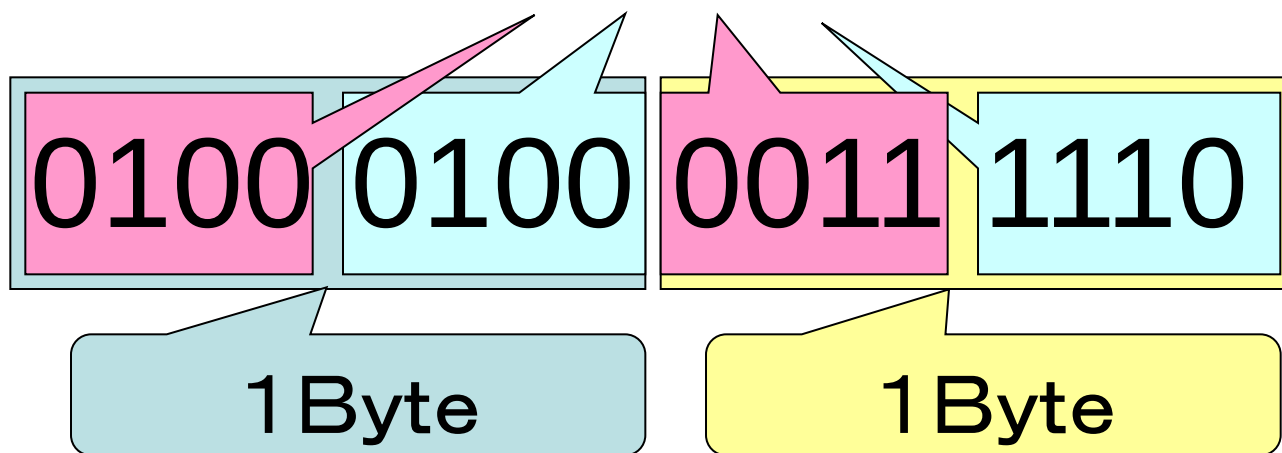
例) 443E

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4370	註	耐	鑄	駐	檣	瀦	猪	苧	著	貯	丁	兆	凋	喋	寵	
4420		帖	帳	庀	弔	張	彫	徵	懲	挑	暢	朝	潮	牒	町	眺
4430	聽	脹	腸	蝶	調	謀	超	跳	銚	長	頂	鳥	勅	抄	直	朕
4440	沈	珍	賃	鎮	陳	津	墜	椎	槌	追	鎚	痛	通	塚	拇	搥
4450	槻	佃	漬	柘	辻	薦	綴	鐸	椿	潰	坪	壺	孀	紬	爪	吊
4460	釣	鶴	亭	低	停	偵	剃	貞	呈	堤	定	帝	底	庭	廷	弟
4470	梯	抵	挺	埶	梯	江	碛	埶	租	絰	甌	訂	諱	陪	準	

漢字の文字コード

直

443E



漢字1文字について、2Byteの情報量で
区別している

「半角」と「全角」

K

日本語
「OFF」

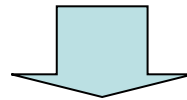
004B

K

日本語
「ON」

234B

日本語入力を「オン」にすると、
「半角（直接：1Byte）入力」から
「全角（日本語変換：2Byte）入力」モードへ



見た目はほとんど同じでも、番号が違う！！
→ コンピュータ内では、「全く違う文字」として認識される！！
普段から「半角」と「全角」の違いに気をつけよう！！

16進数と文字コード

塞 翁 が 馬

3A49

3227

242C

474F

0011101001001001 0011001000100111 0010010000101100 0100011101001111

フォント

文字の「形(=グリフ)」を集めた「種類」
大きさや色情報等を含めて呼ぶこともある

「明朝体(みんちょうたい)」

「ゴシック体」

「行書体(ぎょうしょたい)」

プロポーショナルフォント

☆プロポーショナルフォント

文字本来の形に合わせ、横幅をバランスよく変えたもの。

(「i」や「j」に注目)

例) MSPゴシック abcdefghijklmnopqrstuvwxyz

☆等幅フォント

文字の形に関係なく、1つひとつの文字を同じ幅にしたもの。

例) MSゴシック abcdefghijklmnopqrstuvwxyz

☆プロポーショナルフォント利用時に行頭をあわせたい場合は、「タブ」や「インデント」といった機能を使うとよい。

文字の形を記録するしくみ

ビットマップフォント



アウトラインフォント

成瀬
高校

「点の集まり」として記録

負荷が小さいため、機械への組み込みとして良く使われる

大きくするとギザギザが目立つ

複雑な関数などの計算式として記録

大きくしても再計算し滑らかに表示できる為、画面表示や印刷など広く使われている

都度計算するので比較的負荷が高い
(高性能化に伴い現在のPCではほぼ無視できる)

文字をデジタル化する方法

- キーボード
入力したキーに対応する文字コードを送る
- 手書き入力
ペンタブレット、タブレットPCなど
手書きで入力した「文字」を認識してコード化
- OCR
イメージスキャナ等から画像を文字として認識
- 音声入力
音声から文字を認識

文字をデジタル化するメリット

- 簡単に検索ができるようになる
- 編集がしやすくなる
- 保存が容易になる
- 劣化しにくい
- ネットワークを通じて瞬時に送受信できる
.....など

まとめ

- コンピュータでは、文字1つ1つに「背番号」のような「文字コード」をつけて処理しており、これを一覧にしたのが文字コード表である。
- 英数記号カナ文字を1バイトで割り振った「半角」と、漢字などを含めた文字を2バイトで割り振った「全角」がある。
- 文字コード表には歴史的・国際的な理由から多くの種類があり、送信側と受信側が合っていないと「文字化け」がおこることがある。