

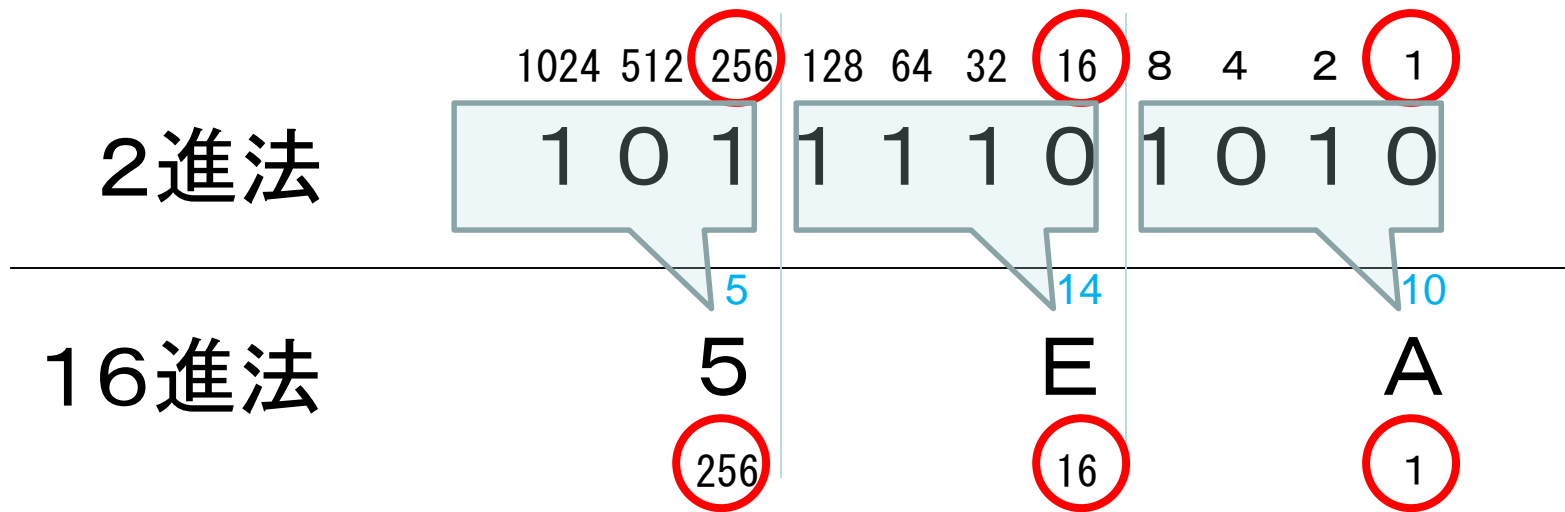
# 文字のデジタル化

情報の科学 第31回授業

06コンピュータによる情報の表現

対応ファイル: 17exp29.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

0101 1101

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。

# 「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



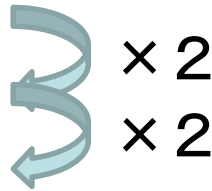
5  
2

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)・・・ =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」  
文字と文字コードの対応関係を文字コード体系という。

$2^7 = 128$ 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

7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	{		}	~	

4B



# 文字コード

一つひとつの文字に割り当てられた「背番号」  
文字と文字コードの対応関係を文字コード体系という。

$2^7 = 128$ 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(JIS)」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」  
文字と文字コードの対応関係を文字コード体系という。

$2^7 = 128$ 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>

$2^{16} = 65536$ 通り

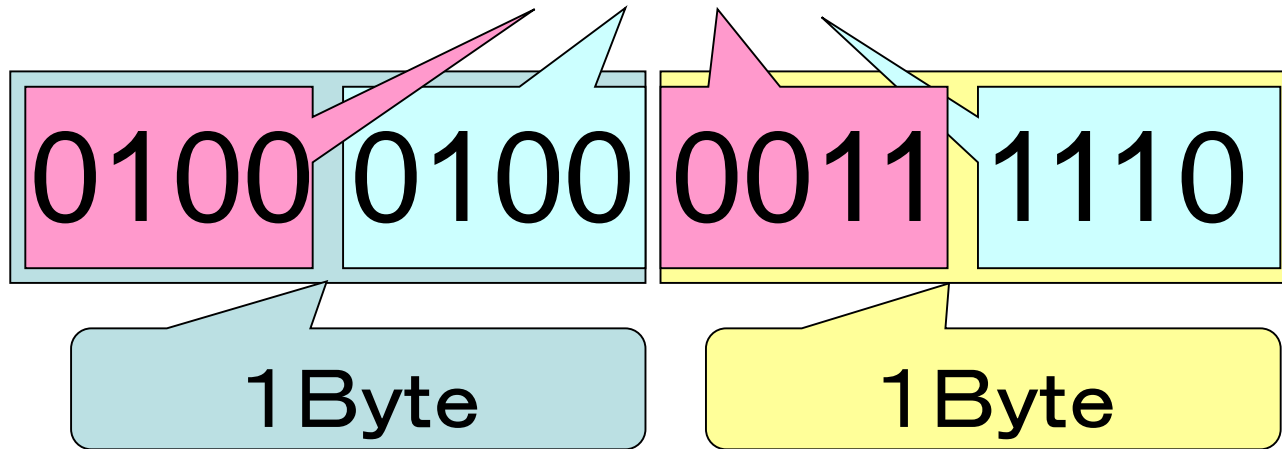
(漢字など)



# 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

# エンコード(符号化)

- 情報を0と1の文字列にすることをエンコード(符号化)という。
- 特に、文字を0と1の文字列に当てはめる時、さまざまな方式(エンコーディング方式)が使われた。
  - JIS X 0208 をもとに、いくつかの組織が独自の文字コード体系を作った。
  - ISO-2022-JP、Shift\_JIS、EUC-JP など

# 文字コード

一つひとつの文字に割り当てられた「背番号」  
文字と文字コードの対応関係を文字コード体系という。

$2^7 = 128$ 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



# 「半角」と「全角」

K

日本語  
「OFF」

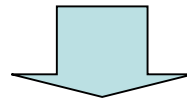
004B

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

001101001001001 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て、横幅をバランスよく変えたもの。

(「i」や「j」に注目)

例) MS **P**ゴシック abcdefg**h**ijklmnopqrstuvwxyz

## ☆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 abcdefg**h**ijklmnopqrstuvwxyz

☆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)

# 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、これらの対応関係を文字コード体系という。
- 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。
- 文字コード体系には歴史的・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。