

# 岡山大学科学先取りコース

2009年10月24日

コンピュータはどのように1+2を計算しているのだろうか

(数字や文字やプログラムがどのように2進数で表され、  
コンピュータ内部で処理されているかを学びます。)

宇部工業高等専門学校 電気工学科 岡村好庸

- 1 この計算は正しいでしょうか
- 2 あなたのパソコンで実習しましょう
- 3 コンピュータはどのように1と2を足すのか
- 4 おまけ コンピュータシミュレーションと物理



この計算は正しいでしょうか？

$$1+1=10$$

!!!コンピュータが扱う言語は0と1だけ!!!

0	0	0	0	0	0	0	1
---	---	---	---	---	---	---	---

1

0	0	0	0	0	0	1	1
---	---	---	---	---	---	---	---

3

[128, 64, 32, 16, 8, 4, 2, 1]

0	0	0	0	0	1	1	1
---	---	---	---	---	---	---	---

7

[128, 64, 32, 16, 8, 4, 2, 1]

0	0	1	1	1	1	1	0
---	---	---	---	---	---	---	---

62

[128, 64, 32, 16, 8, 4, 2, 1]

1	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---

128

[ $2^7$ ,  $2^6$ ,  $2^5$ ,  $2^4$ ,  $2^3$ ,  $2^2$ ,  $2^1$ ,  $2^0$ ]

0	0	0	0	0	0	0	1
---	---	---	---	---	---	---	---

+

0	0	0	0	0	0	0	1
---	---	---	---	---	---	---	---

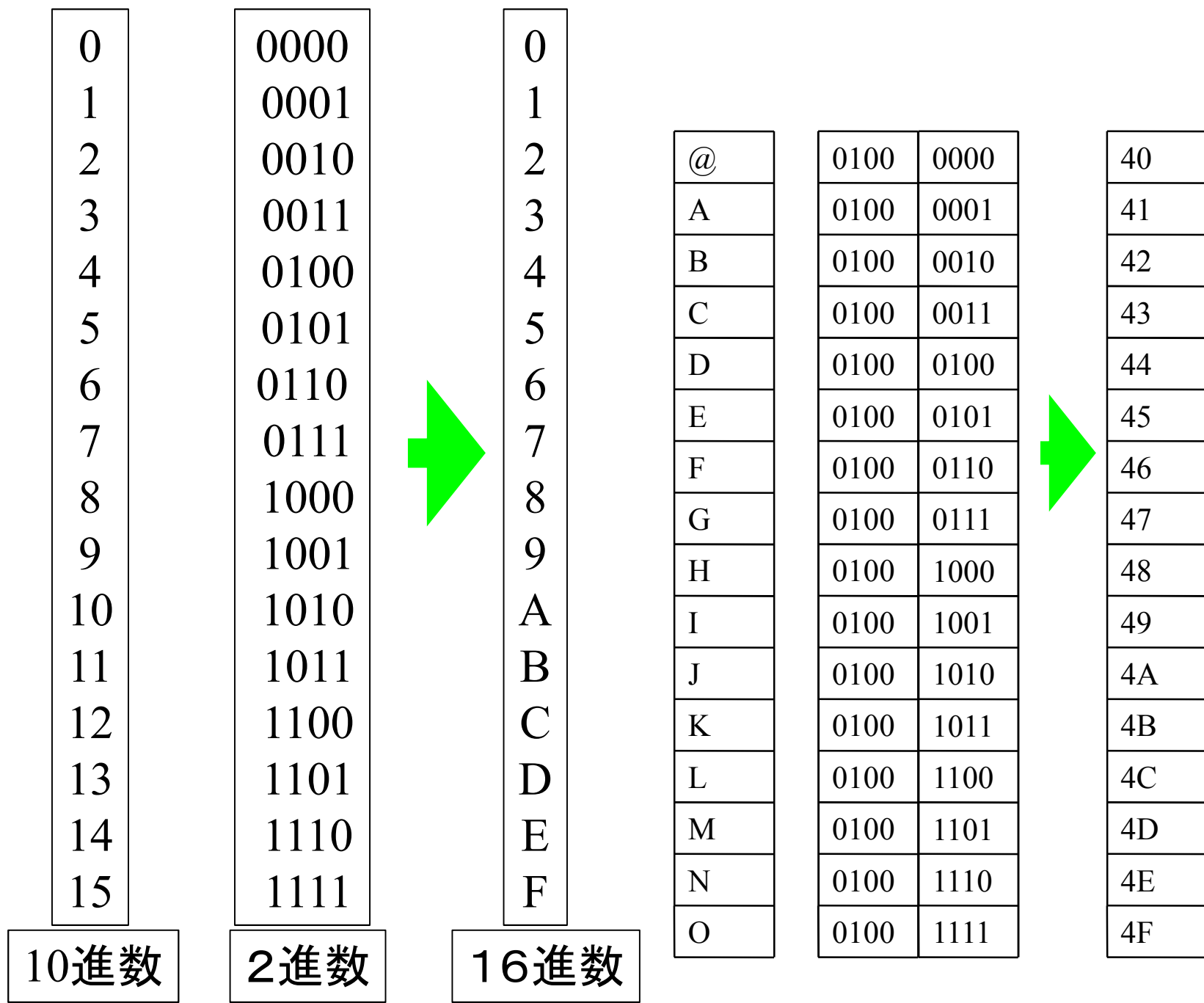
||

0	0	0	0	0	0	1	0
---	---	---	---	---	---	---	---

# ASCIIコード

@	0100	0000
A	0100	0001
B	0100	0010
C	0100	0011
D	0100	0100
E	0100	0101
F	0100	0110
G	0100	0111
H	0100	1000
I	0100	1001
J	0100	1010
K	0100	1011
L	0100	1100
M	0100	1101
N	0100	1110
O	0100	1111

ASCIIとは、  
1963年にアメリカ規格協会 (ANSI)が定めた、  
情報交換用の文字コードの体系。  
1967年に国際標準化機構(ISO)で定められた  
情報交換用符号の  
国際規格「ISO 646」とほぼ同じもの。  
7ビットで表現され、  
128種類のローマ字、数字、記号、  
制御コードで構成されている。  
実際にはコンピュータは1文字を  
8ビット(1バイト)で表現するため、  
256種類の文字を扱うことができる。



10進数

2進数

16進数



# ASCIIコードとJISコード表

## 下位ビット

下位ビット

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
1	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
2	SP	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	☐	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	☐	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	☐
8																
9																
A		。	「	」	、	・	ヲ	アイ	ウ	エ	オ	ヤ	ユ	ヨ	ツ	
B	ー	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
C	夕	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
D	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	”	°
E																
F																

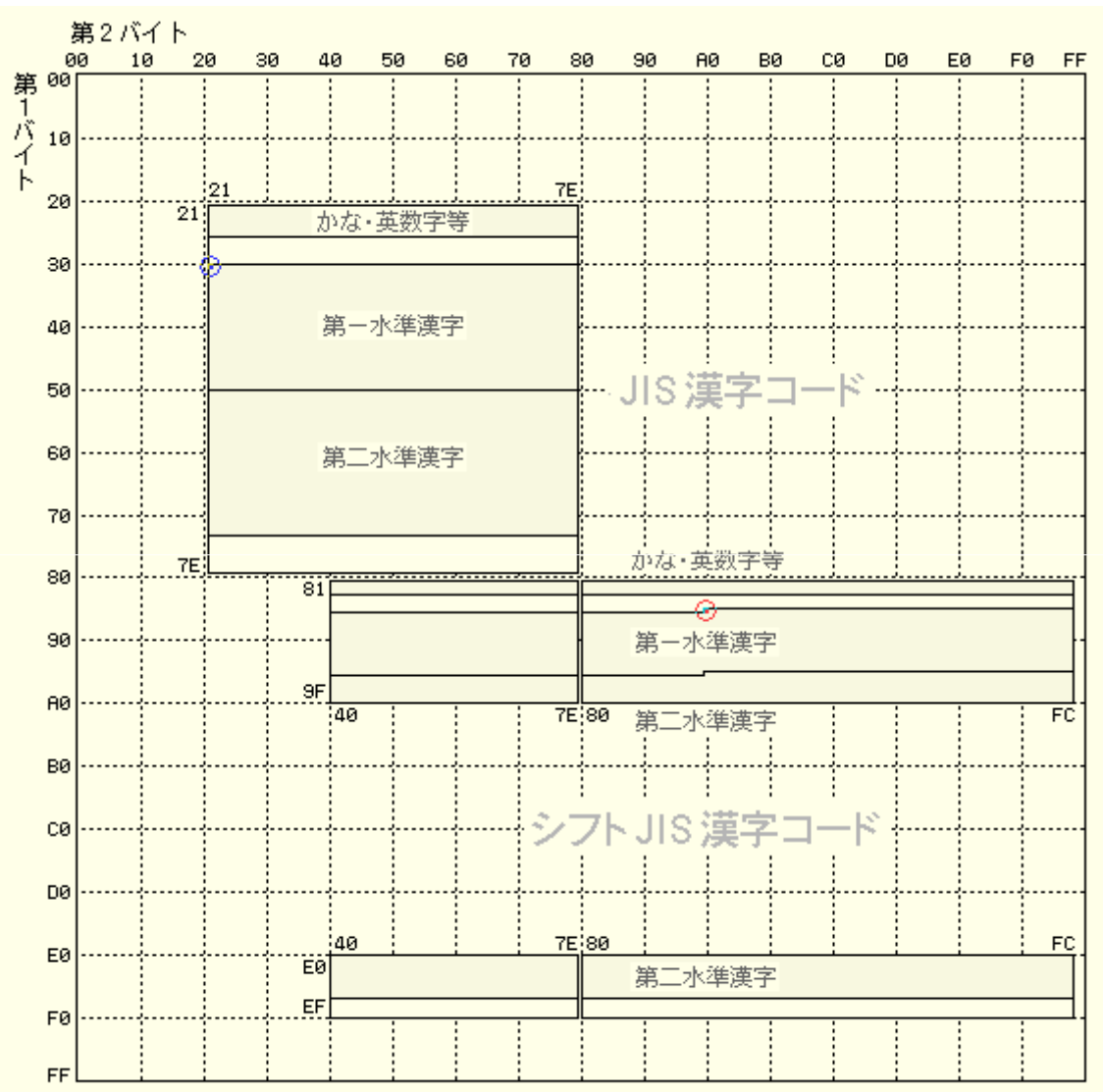
4A 49 53 34 41 3B 7A 25 33 21 3C 25 49 → JIS4A;z%3!<%I

## JIS 漢字コード (JIS X 0208)

2 バイトを使って、第1バイト、第2バイトともに16進数表記で21～7Eの、それぞれ94文字、全体では94×94＝8,836文字を表すことができる領域に、漢字、かな、英数字、記号など6,879文字を割り当てています。

## シフトJIS漢字コード

パソコンの内部で使用されている文字コードで、JIS漢字コード（JIS X0208）を移動（シフト）させたものです。





参照 <http://www.infonet.co.jp/ueyama/ip/binary/shiftjis.html>

## JIS 漢字コードからシフトJIS漢字コードへ

4A 49 53 34 41 3B 7A 25 33 21 3C 25 49

これをASCIIコード(1バイト)で読めば 「JIS4A;z%3!<%I」

4A 49 53 1B 24 42 34 41 3B 7A 25 33 21 3C 25 49 1B 28 42

これをJIS 漢字コードで読めば 「JIS 漢字コード」

4A 49 53 8A BF 8E 9A 83 52 81 5B 83 68

これをシフトJIS 漢字コードで読めば 「JIS 漢字コード」

これら ○ は1バイトでは読めない

このプログラムはコンパイルエラーとなった。なぜ？

```
#include <stdio.h>
main(){
int a, b, c;
a=1;
b=2;
c=a+b;
printf("c=%d¥n",c); }
add.c
```



半角スペース	20
全角スペース	81 40

シフトJIS漢字コード

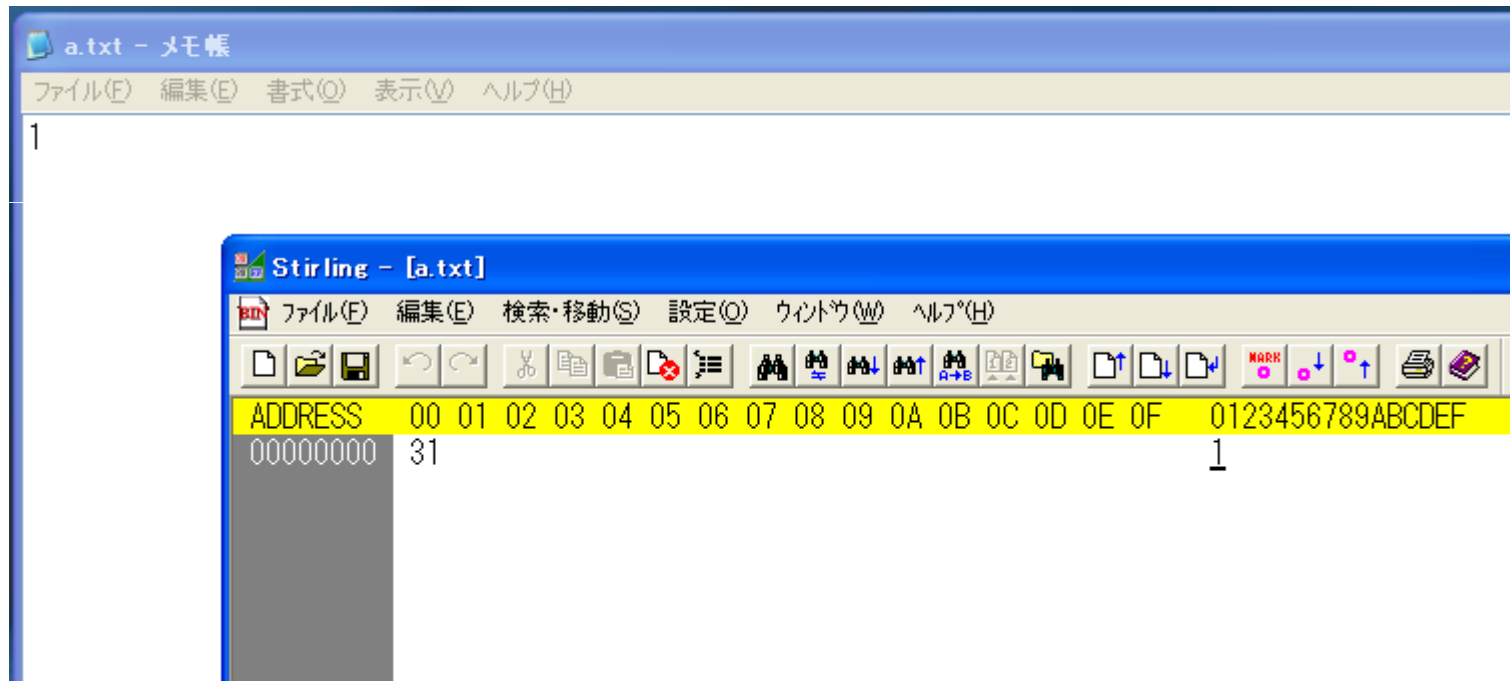
あなたのパソコンで実習しましょう

目的は文字とコードの対応関係を調べることです

## 使うソフトウェア

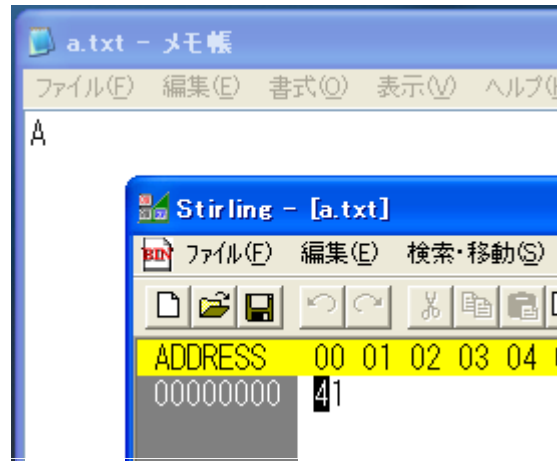
- 1 メモ帳(テキストエディタ) 
- 2 スターリング[Stirling] (バイナリエディタ) 

- .1 メモ帳とスターリングを開く。
- .2 メモ帳に半角英数文字「1」を書いてa.txtという名前で保存する。
- .3 スターリングでファイルa.txtを開く。

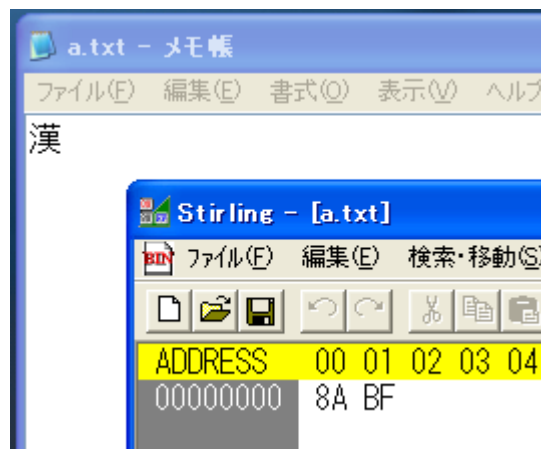


ASCIIコード表へ

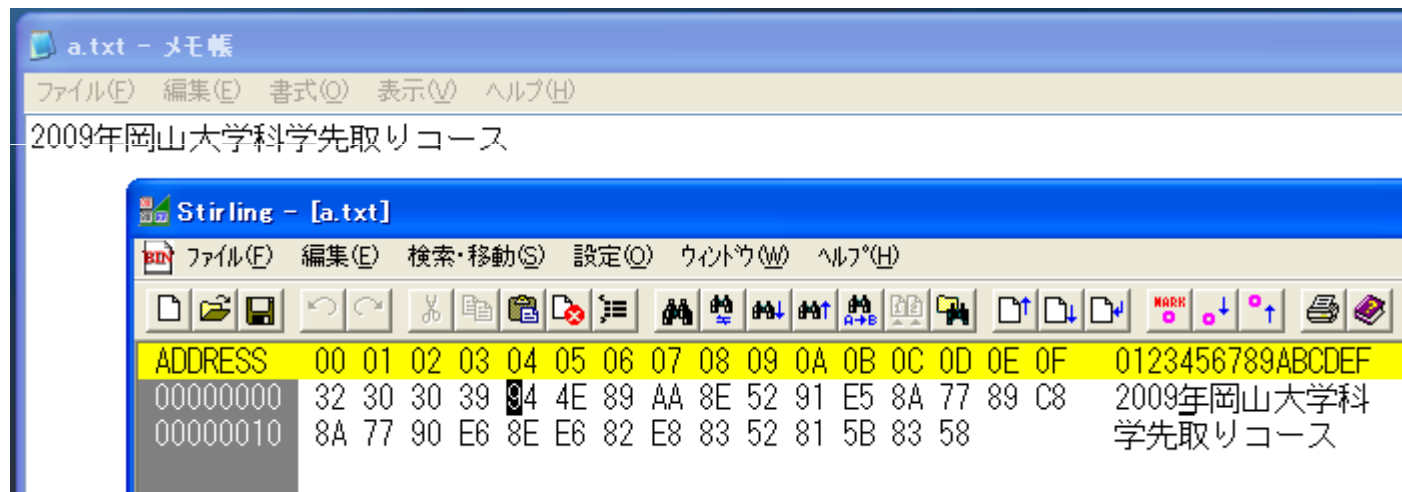
- 1 メモ帳に半角英数文字「A」を書いて上書き保存する。
- 2 スターリングでコードを見る。



- 1 メモ帳に全角文字「漢」を書いて上書き保存する。
- 2 スターリングでコードを見る。



- 1 メモ帳に「2009年岡山大学科学先取りコース」と書いて上書き保存する。(2009は半角文字、他は全角文字)
- 2 スターリングでコードを見る。



おまけ もしワードがインストールされているのなら

.スターリングがa.txtを表示しているままで

- .1 ワードを開いて「2009年岡山大学科学先取りコース」と書いて名前を付けて保存する。(2009は半角文字、他は全角文字)
- .2 スターリングでワード文書を開く
- .3 コードを比較する。

Stirling - 2009岡山大科学先取りコース.doc

ファイル(F) 編集(E) 検索・移動(S) 設定(O) ウィンドウ(W) ヘルプ(H)


2009岡山大科学先取りコース.doc

ADDRESS	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
00003440	02	00	00	00	98	00	00	00	03	00	00	00	C0	00	00	00	.....タ...
00003450	04	00	00	00	CC	00	00	00	05	00	00	00	DC	00	00	00	....7.....7...
00003460	06	00	00	00	E8	00	00	00	07	00	00	00	F4	00	00	00	.....
00003470	08	00	00	00	08	01	00	00	09	00	00	00	18	01	00	00	.....
00003480	12	00	00	00	24	01	00	00	0A	00	00	00	40	01	00	00	....\$......@...
00003490	0C	00	00	00	4C	01	00	00	0D	00	00	00	58	01	00	00	....L.....X...
000034A0	0E	00	00	00	64	01	00	00	0F	00	00	00	6C	01	00	00	....d.....l...
000034B0	10	00	00	00	74	01	00	00	13	00	00	00	7C	01	00	00	....t.....l...
000034C0	02	00	00	00	A4	03	00	00	1E	00	00	00	1D	00	00	00	.....
000034D0	32	30	30	39	89	AA	8E	52	91	E5	8A	77	89	C8	8A	77	2009岡山大科学
000034E0	90	E6	8E	E6	82	E8	83	52	81	5B	83	58	00	00	00	00	先取りコース....
000034F0	1E	00	00	00	01	00	00	00	00	30	30	39	1E	00	00	00	.....009....
00003500	08	00	00	00	20	61	6B	61	64	6F	75	00	1E	00	00	00	.... akadou....
00003510	01	00	00	00	00	61	6B	61	1E	00	00	00	01	00	00	00	.... aka.....
00003520	00	61	6B	61	1E	00	00	00	0B	00	00	00	4E	6F	72	6D	.aka.....Norm
00003530	61	6C	3E	64	6E	74	00	77	1E	00	00	00	08	00	00	00	al det ...

a.txt

ADDRESS	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
00000000	32	30	30	39	94	4E	89	AA	8E	52	91	E5	8A	77	89	C8	2009年岡山大科学
00000010	8A	77	90	E6	8E	E6	82	E8	83	52	81	5B	83	58			学先取りコース





コンピュータはどのように  
1と2を足すのか

```
#include <stdio.h>
```

```
main(){
```

```
int a, b, c;
```

```
a=1;
```

```
b=2;
```

```
c=a+b;
```

```
printf("c=%d\n",c); }
```

add.c

```
extern int printf (__const  
/* Read for  
extern int scanf (__const
```

整数

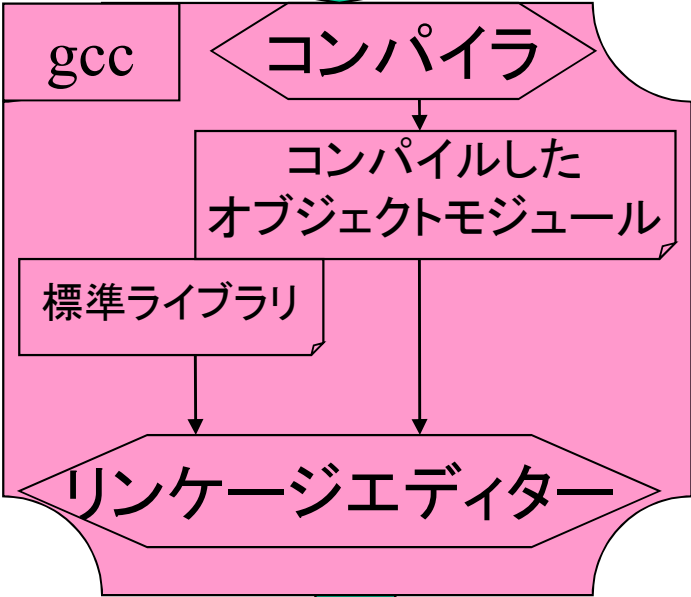
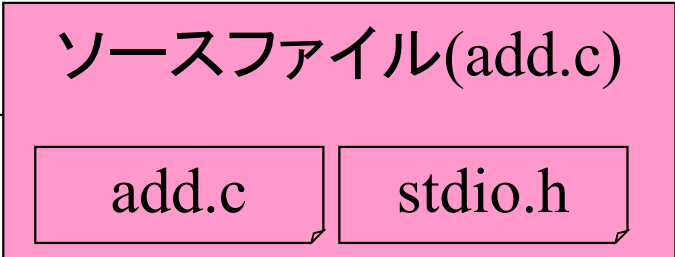
a	000000000000000001
b	000000000000000010
c	000000000000000011

stdio.h

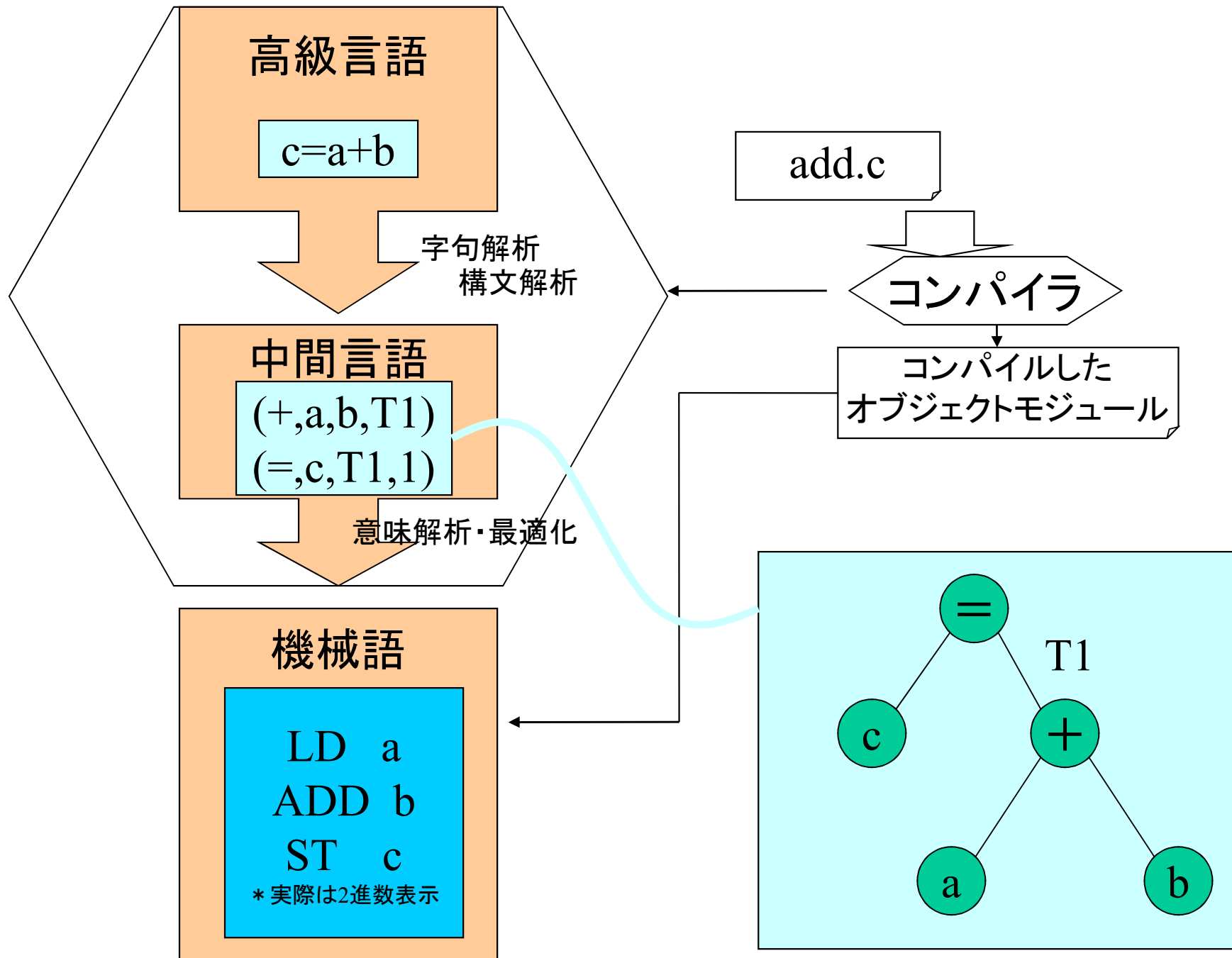
符号付の固定小数点数に変換する  
(負数は2の補数で表わす)

\*語: コンピュータごとに決めた処理の基本単位

```
% ls
% vi add.c
% ls
add.c
% gcc add.c -o add
% ls
add add.c
% add
c=3
%
```



実行

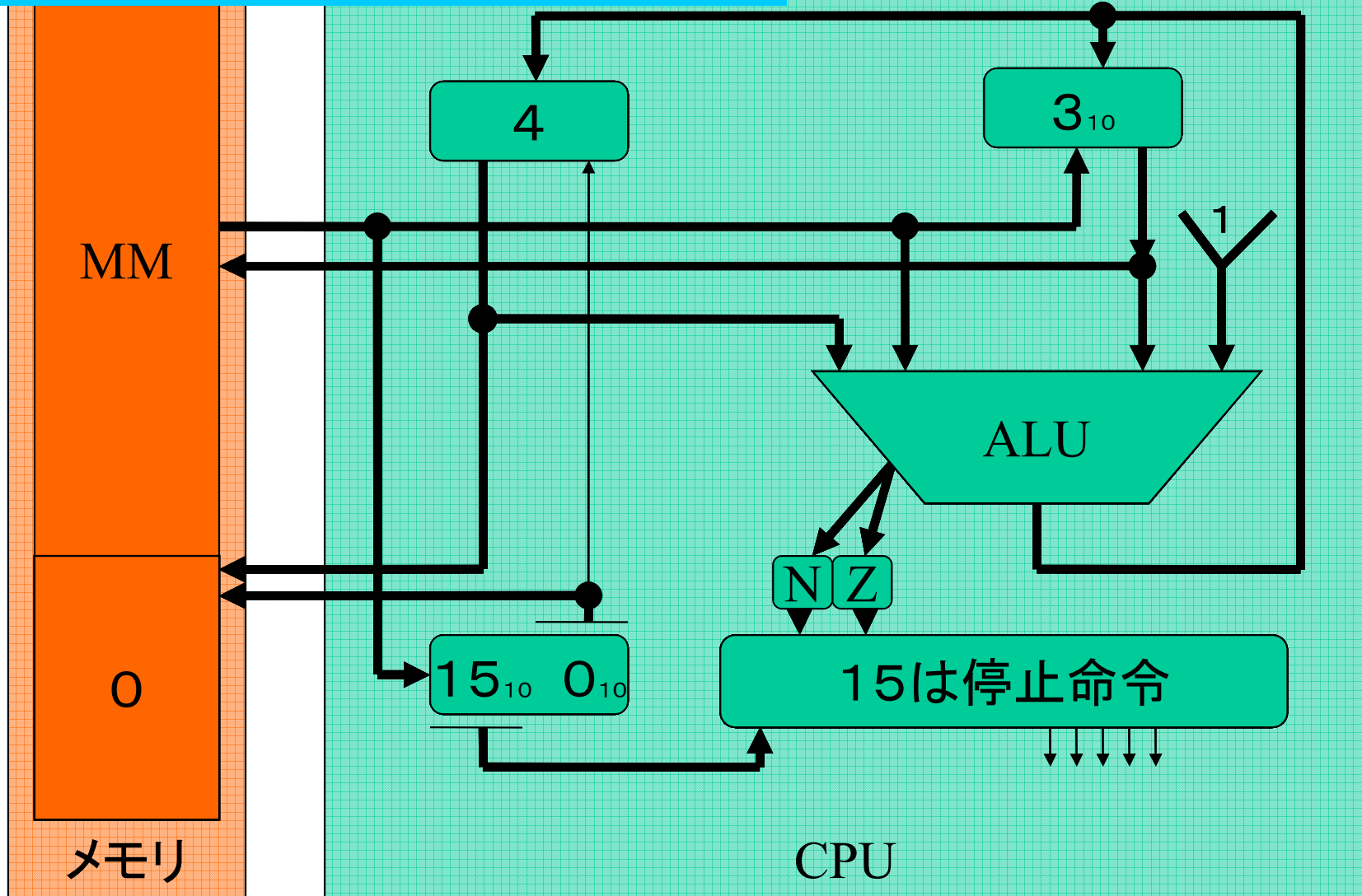


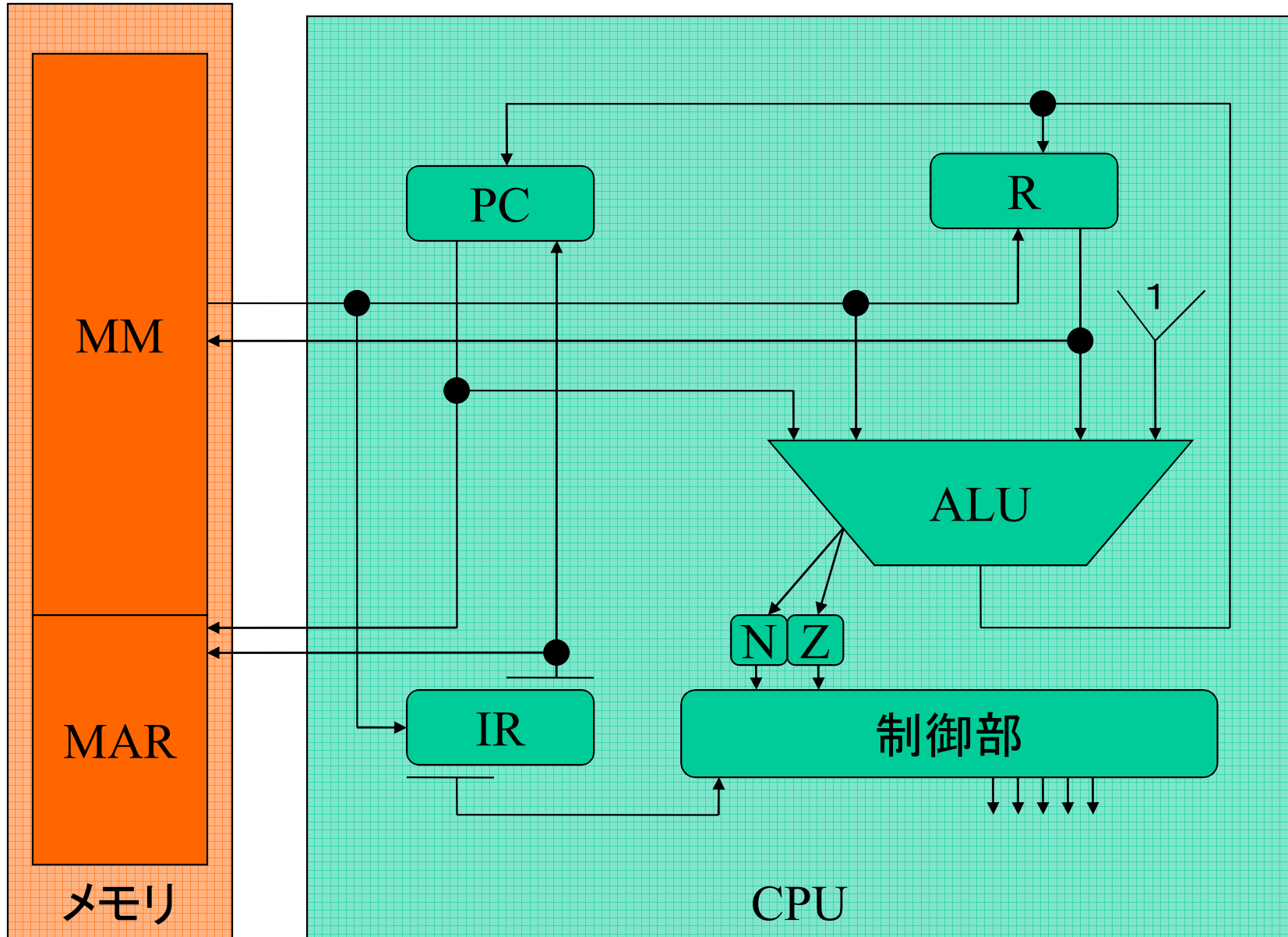
# 機械語表示(16ビット)

	アドレス	(命令)	データ
テキスト セグメント	0000	0000	0000000000100 (LD 4 <sub>10</sub> )
	0001	0010	0000000000101 (ADD 5 <sub>10</sub> )
	0010	0001	0000000000110 (ST 6 <sub>10</sub> )
	0011	1111	0000000000000 (HLT)
データ セグメント	0100		000000000000000001
	0101		000000000000000010
	0110		000000000000000000

アドレス 命令(HLT) データ

0011 1111 000000000000





# ノイマン型コンピュータ [フォン・ノイマン(1903-1957)]

## 1. プログラム内臓方式

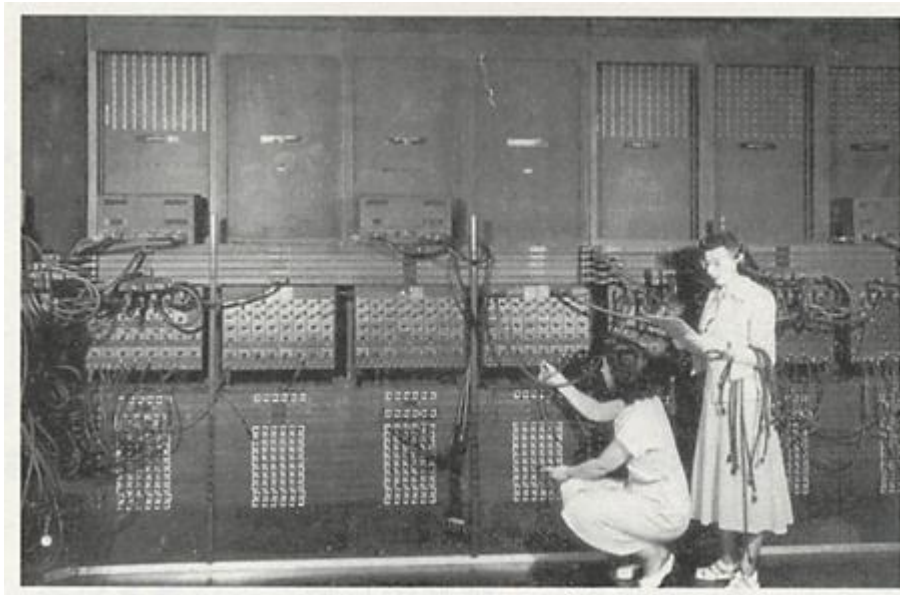
プログラムがMMIに格納されている

## 2. 逐次制御方式

MMから命令が1つずつ取り出されて実行

## 3. 2進数の採用

データのON・OFF表現



ENIAC(1946)では10進数を用い、配線をつなぎかえる方法により計算を行った。(US Army photo)



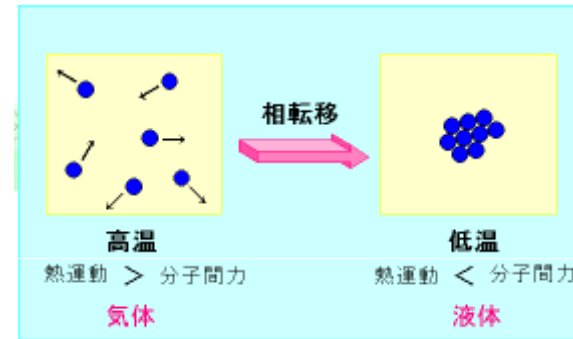
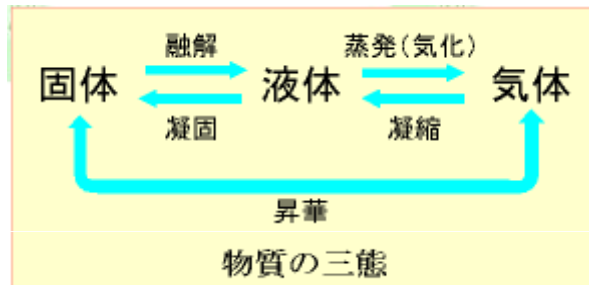


# コンピュータシミュレーションと物理

相転移を乱数を用いてコンピュータ上で実現する

相転移とは、温度や圧力などの変化により生じる

## 1 物質の三態の間の状態変化 (蒸発・凝縮・融解・凝固・昇華など)



図参照、<http://piano.chem.yamaguchi-u.ac.jp/kiho/ken/jiyuudo/souteni.html>

## 2 秩序 - 無秩序へと転移する現象 (磁気相転移、常伝導から超伝導状態への転移など)

# 磁気相転移を説明するモデルについて

Ising Model (1925 E.Isingが提案)

$$E = - \sum_{(i,j)} J_{i,j} \sigma_i \sigma_j$$

1次元(1925 E. Ising) 相転移を示さない

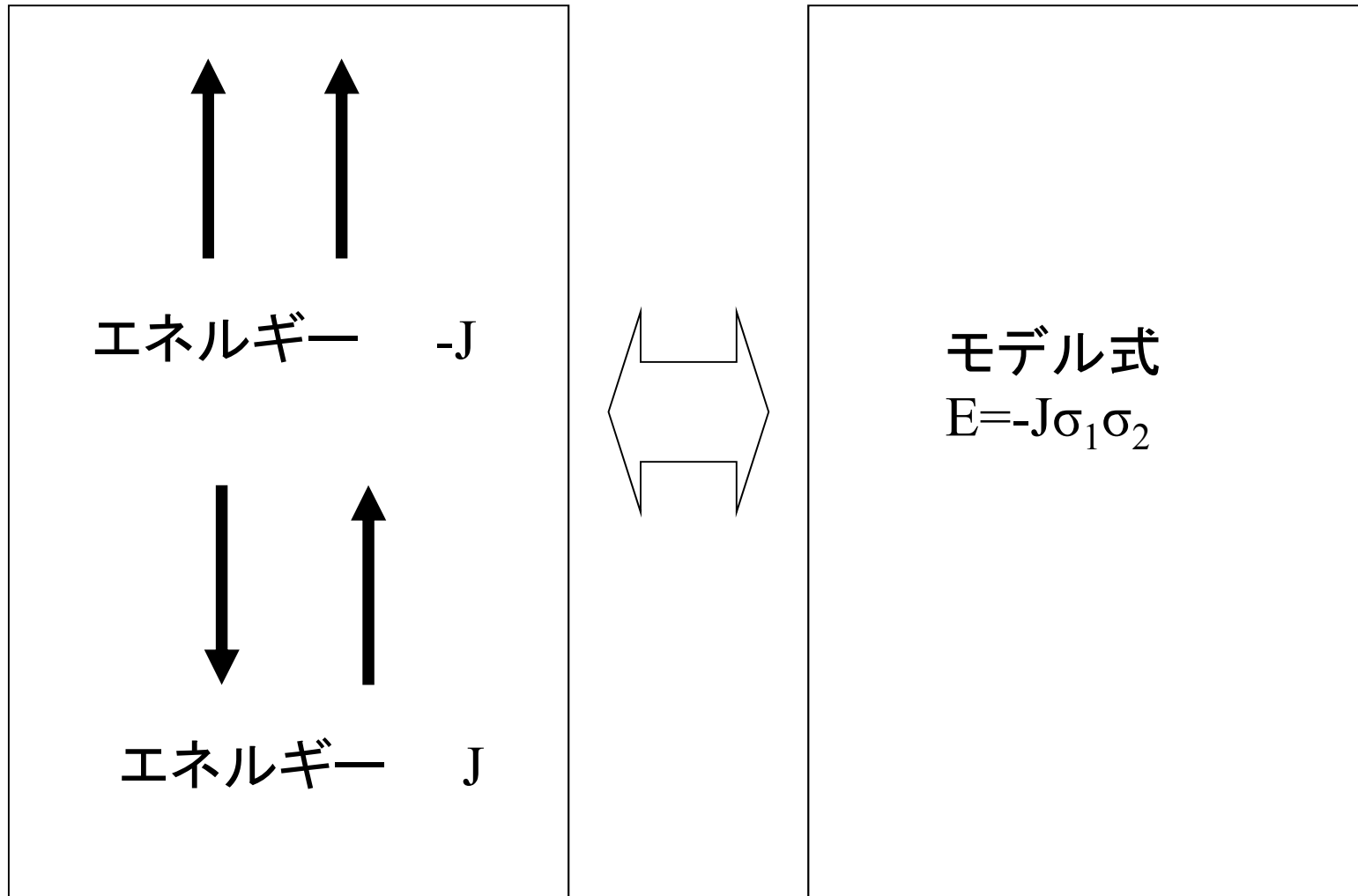
2次元(1944 L.Onsager)

多体系の厳密解で相転移を示す(相転移研究の支柱)ために非常に重要

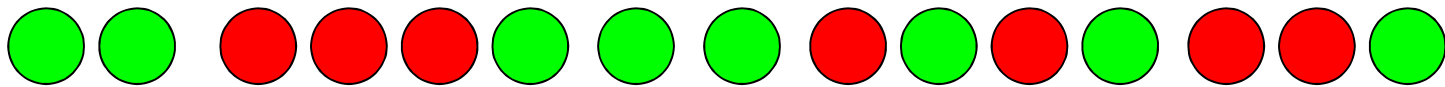
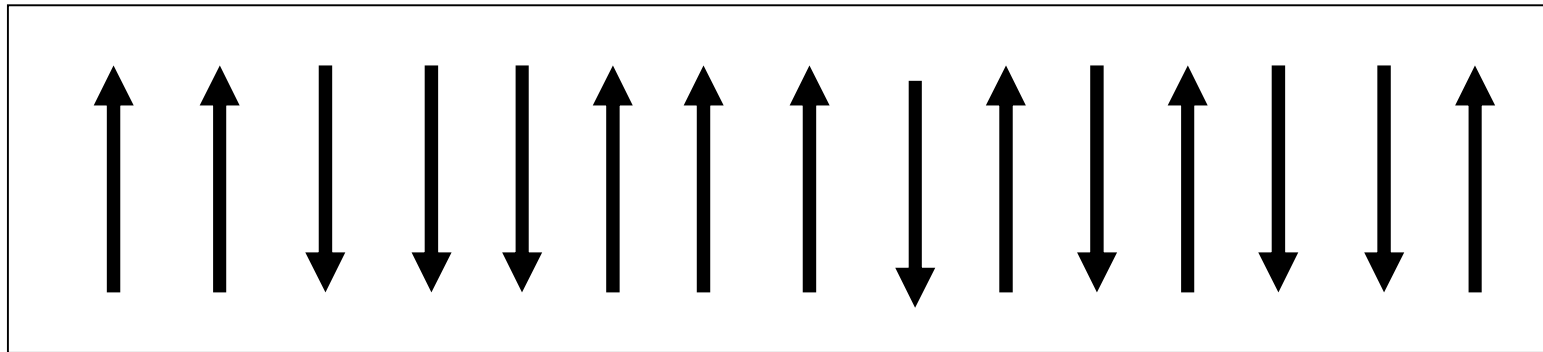
自発磁化  $m = [1 - 1/\sinh^4(J/k_B T)]^{1/8}$

3次元(いまだ厳密解は求められていない)

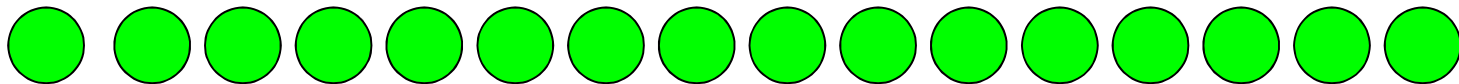
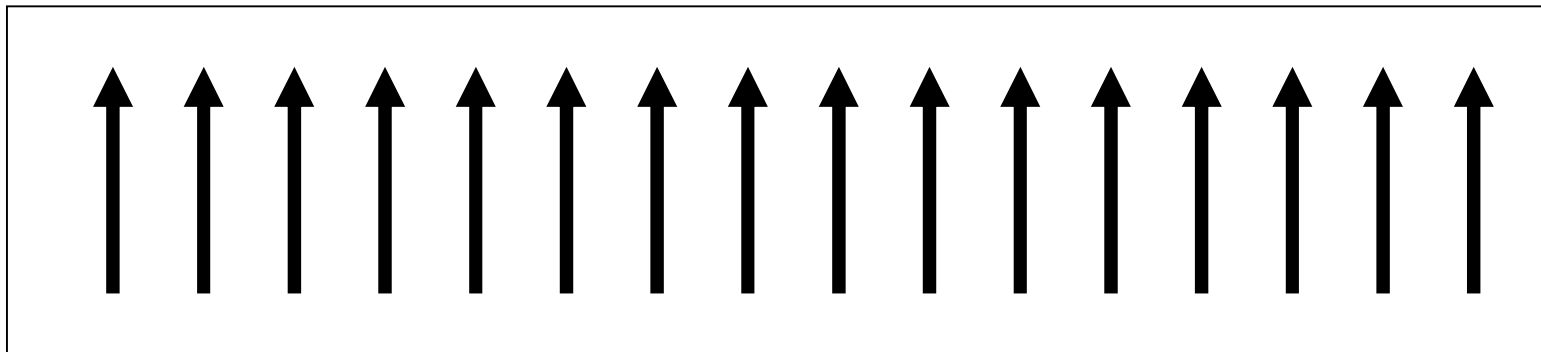
磁性体———温度が下がると磁石になる性質がある



温度が高いと



温度が低いと



# Ising Model

