

# 計算機入門

田村直之

(神戸大学工学部情報知能工学科)

E-mail: [tamura@kobe-u.ac.jp](mailto:tamura@kobe-u.ac.jp)

WWW: <http://bach.seg.kobe-u.ac.jp/tamura.html>

# 1 はじめに

## 1.1 計算機の用途

パソコンを中心に

- インターネット (電子メール, WWW)
- ワープロ・DTP
- 表計算・グラフ作成
- データベース・CD-ROM
- 図形処理・グラフィックス
- ホビー・音楽・ゲーム

その他

- 制御・ロボット・自動車・電化製品
- オンラインシステム・銀行・みどりの窓口・ホームトレード
- 通信・POS 端末
- コンピュータグラフィックス・仮想現実感
- 人工知能・機械翻訳
- 文房具・電子手帳・電子辞書

## 1.2 計算機とは

大量のデータ (情報) を, 人間の指示した手順 (プログラム) にしたがって, 高速に**処理**および**通信**する機能を持った機械

- **ハードウェア** (hardware)  
コンピュータの機械装置そのもの
- **ソフトウェア** (software)  
ハードウェアに対する用語で, 計算機を利用するためのプログラムの総称

## 1.3 計算機の種類

- ゲーム機
- 携帯情報機器, PDA (Personal Digital Assistants)
- パソコン (パーソナルコンピュータ)  
デスクトップ, ノートブック
- ワークステーション

- 大型コンピュータ
- スーパーコンピュータ
- マイコン (マイクロコンピュータ)

#### 1.4 計算機と社会

- アルビン・トフラー「第三の波」  
農業革命→産業革命→情報革命
- アラン・ケイ “ダイナブック”  
メディアとしてのコンピュータ  
エージェントとしてのコンピュータ

## 2 計算機の歴史

### 2.1 電子計算機以前

- パスカルの計算機 (1642 年) 加減算
- ライプニッツの計算機 (1673 年) 加減乗除算
- バベージの階差機関 (1820 年) 階差方程式
- バベージの解析機関 (1833 年)  
歯車式 (未完成), 現代の計算機と同様の構成, パンチカードによるプログラム
- ホレリスのパンチカード式統計機 (1887 年)  
国勢調査, IBM の前身
- ハーバード大学 リレー計算機 MARK-I (1944 年)  
バベージの夢の実現, 約 3,000 個の電磁リレー, 23 桁乗算が約 5 秒

### 2.2 電子計算機の誕生

- ペンシルバニア大学 ENIAC (1946 年)  
世界最初のプログラム可能な電子計算機  
18,000 本以上の真空管, スイッチと配線によるプログラム, 10 進数による計算, 10 桁加算が  $200\mu$  秒
- フォン・ノイマンの提案 (1945 年)  
計算機の設計方針についての提案 (ノイマン方式, フォン・ノイマン型計算機)
  - 2 進数による計算
  - 5 つの基本構成要素 (演算装置, 制御装置, 記憶装置, 入力装置, 出力装置)
  - プログラム内蔵方式
  - 汎用
- ケンブリッジ大学 EDSAC (1949 年)  
ソフトウェアでの重要な概念 (アセンブラ, サブルーチン, ライブラリ, マクロ)

#### 計算機でよく使用する量の単位

単位	読み	値
T	テラ	$10^{12} = 1,000,000,000,000$
G	ギガ	$10^9 = 1,000,000,000$
M	メガ	$10^6 = 1,000,000$
K, k	キロ	$10^3 = 1,000$
m	ミリ	$10^{-3} = 0.001$
$\mu$	マイクロ	$10^{-6} = 0.000001$
n	ナノ	$10^{-9} = 0.000000001$
p	ピコ	$10^{-12} = 0.000000000001$

## 2.3 計算機の世代

- **第1世代, 真空管の時代 (1951–1957)**  
ユニバック社 UNIVAC I, IBM 650, 磁気ドラム
- **第2世代, トランジスタの時代 (1958–1963)**  
ユニバック社 USSC, IBM 1401 (IBM の成功), 磁気コア
- **第3世代, IC の時代 (1964–1970)**  
IBM 360 シリーズ, オペレーティングシステム
- **ミニコンピュータの誕生 (1965 年)**  
DEC 社 PDP-8
- **第3.5世代, LSI の時代 (1970–1980 年頃)**  
IBM 370 シリーズ, IC メモリー, RAS 機能
- **マイクロプロセッサの誕生 (1971 年)**  
インテル社 4004, 8008 (1972 年)
- **スーパーコンピュータの誕生 (1972 年)**  
クレイ社 CRAY-1, 150MFLOPS
- **第4世代, VLSI の時代 (現代)**
- **パソコンの誕生 (1977 年)**  
アップル社 Apple II, IBM PC (1981 年), Macintosh (1984 年), PC 9801 (1982 年)
- **第5世代計算機の研究の開始 (1982 年)**  
新世代コンピュータ技術開発機構

### IC (Integrated Circuit, 集積回路)

集積度	名称	
数十	小規模集積回路	SSI (Small Scale Integrated circuit)
数百	中規模集積回路	MSI (Middle Scale Integrated circuit)
数千 ~ 数万	大規模集積回路	LSI (Large Scale Integrated circuit)
数十万	超 LSI	VLSI (Very Large Scale Integrated circuit)

- Microsoft とビル・ゲイツ
- Apple とスティーブ・ジョブス

### 3 2進数

#### 3.1 年齢当てゲーム

表 1								表 2							
1	3	5	7	9	11	13	15	2	3	6	7	10	11	14	15
17	19	21	23	25	27	29	31	18	19	22	23	26	27	30	31
33	35	37	39	41	43	45	47	34	35	38	39	42	43	46	47
49	51	53	55	57	59	61	63	50	51	54	55	58	59	62	63

表 3								表 4							
4	5	6	7	12	13	14	15	8	9	10	11	12	13	14	15
20	21	22	23	28	29	30	31	24	25	26	27	28	29	30	31
36	37	38	39	44	45	46	47	40	41	42	43	44	45	46	47
52	53	54	55	60	61	62	63	56	57	58	59	60	61	62	63

表 5								表 6							
16	17	18	19	20	21	22	23	32	33	34	35	36	37	38	39
24	25	26	27	28	29	30	31	40	41	42	43	44	45	46	47
48	49	50	51	52	53	54	55	48	49	50	51	52	53	54	55
56	57	58	59	60	61	62	63	56	57	58	59	60	61	62	63

#### 3.2 2進数の表記方法

10進数	2進数	10進数	2進数
0	0	16	10000
1	1	17	10001
2	10	18	10010
3	11	19	10011
4	100	20	10100
5	101	21	10101
6	110	22	10110
7	111	23	10111
8	1000	24	11000
9	1001	25	11001
10	1010	26	11010
11	1011	27	11011
12	1100	28	11100
13	1101	29	11101
14	1110	30	11110
15	1111	31	11111

- 2進数から10進数への変換  
2進数の各桁の位が下から順に1, 2, 4, 8, 16, 32, ...であることを利用して計算する。
- 10進数から2進数への変換 (その1)  
10進数を, 商が0になるまで次々と2で割っていく。そのとき得られた余りを逆順に並べたものが, 2進数に変換した結果である。

- 10 進数から 2 進数への変換 (その 2)

10 進数から, 1, 2, 4, 8, 16, 32, ... のうちで, できるだけ大きい数を引くことを, 結果が 0 になるまで繰り返す. 引いた数を位の数とみなして, 対応する桁を 1, それ以外の桁を 0 としたものが, 2 進数に変換した結果である.

- 年齢当てゲームの解説

年齢当てゲームの表  $i$  が, 2 進数で下から  $i$  桁目が 1 の数ばかりを並べた表であることを利用している.

### 3.3 2 進数の加算

- $1 + 1$  が桁上がりをして  $10_2$  となることに注意すれば, 10 進数での筆算の方法と同様にして計算できる.
- 桁数を固定した 2 進数の加算: 2 進数で桁数を 4 桁に固定してみる. 4 桁未満の数は 0 を上位に付け加えて 4 桁にする. 5 桁以上の数は, 下位 4 桁だけを考え, 4 桁目より上位の桁はオーバーフロー(桁あふれ)として無視する. 桁数を固定した計算での結果は  $\equiv$  で表すことにする.

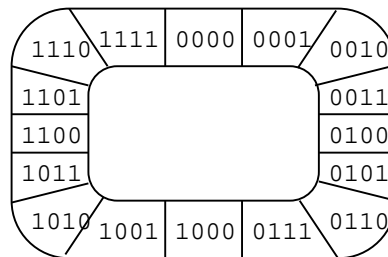


図 1: 2 進数 4 桁のリング

### 3.4 補数と 2 進数の減算

$1111_2$  を加えた結果は, いつも元の数から 1(すなわち  $0001_2$ ) を引いた値に一致する. 同様に,  $1110_2$  を加えた結果は, 元の数から 2(すなわち  $0010_2$ ) を引いた値に一致し,  $1101_2$  を加えた結果は, 元の数から 3(すなわち  $0011_2$ ) を引いた値に一致する.

$0001_2$  と  $1111_2$ ,  $0010_2$  と  $1110_2$ ,  $0011_2$  と  $1101_2$  などの 2 数について, 一方を他方の補数という.

$0001_2$	↔	補数	↔	$1111_2$
$0010_2$	↔	補数	↔	$1110_2$
$0011_2$	↔	補数	↔	$1101_2$
$0100_2$	↔	補数	↔	$1100_2$
$0101_2$	↔	補数	↔	$1011_2$
$0110_2$	↔	補数	↔	$1010_2$
$0111_2$	↔	補数	↔	$1001_2$

0000<sub>2</sub> と 1000<sub>2</sub> の補数は、それぞれ自分自身である。

$$\begin{array}{ccc} & \text{補数} & \\ & \longleftrightarrow & \\ 0000_2 & & 0000_2 \\ & \text{補数} & \\ & \longleftrightarrow & \\ 1000_2 & & 1000_2 \end{array}$$

$x$  の補数 (正確には  $x$  の 2 の補数) は、 $16 - x$  で求めることができる。あるいは、 $x$  の各桁について、0 を 1 に 1 を 0 に反転させた数 (これを 1 の補数と呼ぶことがある) に 1 を加えて求めてもよい。

計算機では、減算を「引く数の補数を加える」という方法で行っている。

$$x - y \equiv x + y \text{ の補数}$$

### 3.5 2進数の乗算, 除算

乗算・除算は、10進数の筆算と同様にしてできる。

- 右に 0 を付け加えると 2 倍
- 右の 1 桁を取り除くと 2 分の 1 倍

### 3.6 16進数

10 進数	2 進数	16 進数	10 進数	2 進数	16 進数
0	0	0	11	1011	B
1	1	1	12	1100	C
2	10	2	13	1101	D
3	11	3	14	1110	E
4	100	4	15	1111	F
5	101	5	16	10000	10
6	110	6	17	10001	11
7	111	7	18	10010	12
8	1000	8	19	10011	13
9	1001	9	20	10100	14
10	1010	A	...	...	...

- 16 進数と 2 進数の間の変換方法  
16 進数の 1 桁が 2 進数の 4 桁にちょうど対応する。
- 16 進数から 10 進数への変換方法  
16 進数の各桁の位が下から順に 1, 16, 256, 4096, 65536, ... であることを利用して計算する。
- 10 進数から 16 進数への変換方法  
10 進数を、商が 0 になるまで次々と 16 で割っていく。そのとき得られた余りを 16 進数に直して逆順に並べたものが、16 進数に変換した結果である。

### 3.7 ビットとバイト

- ビット (bit): 2 進数 1 桁のこと
- バイト (byte): 8 ビットのこと

### 3.8 文字と文字コード

文字は、数にコード化され(文字コード), そのコードが2進数で表現されている。文字コードには規格がある(JISなど)。

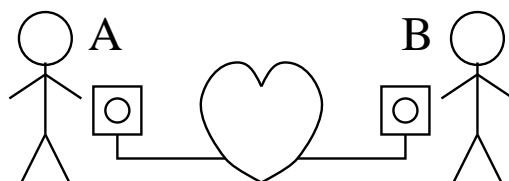
- 英数字カナ: 1文字が1バイト
- 漢字: 1文字が2バイト (JISコード, シフトJISコード, EUCコード)  
「漢」はJISでは $3441_{16}$ , シフトJISでは $8ABF_{16}$

### 3.9 数の表現

- 整数
- 浮動小数点数

## 4 論理回路

### 4.1 お見合い



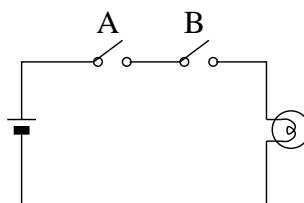
$A$  = 「A さんが B 君を好きだ」

$B$  = 「B 君が A さんを好きだ」

- 両思いの回路 (AND 回路)

$A \wedge B$  (論理積)

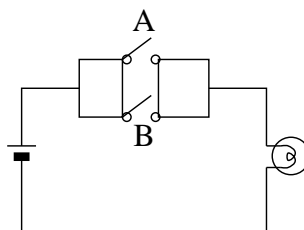
なる論理式で表し「A かつ B」とか「A AND B」と読む。



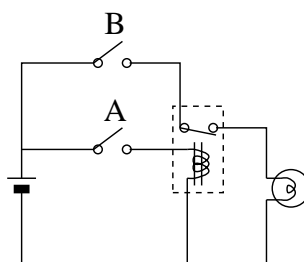
- どちらか思いの回路 (OR 回路)

$A \vee B$  (論理和)

なる論理式で表し「A または B」とか「A OR B」と読む。



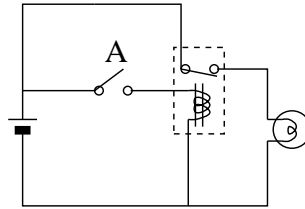
- B 君の片思いの回路 (NOT 回路)



この一部の「Aでない」を

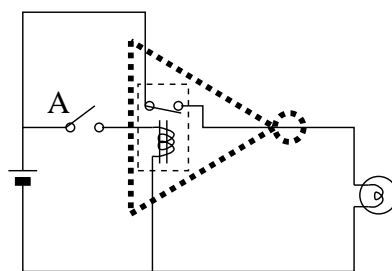
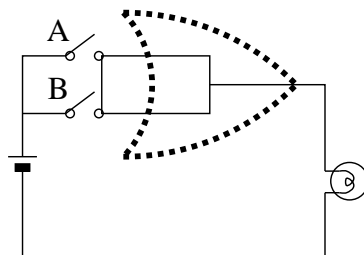
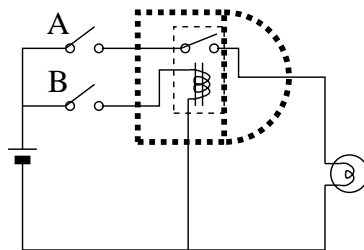
$\bar{A}$  (否定)

なる論理式で表し「Aでない」とか「NOT A」と読む。

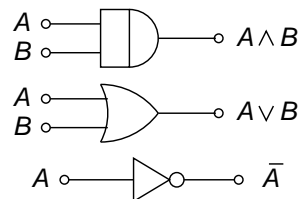


## 4.2 論理素子と論理回路

AND回路, OR回路, NOT回路を書き直す。

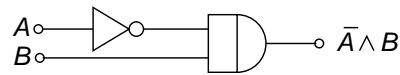


これらを入力と出力の関係だけに着目して, 次のように記号化する (論理素子).



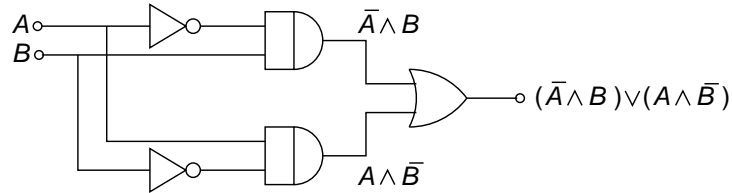
以上のような論理素子を使って論理回路を書き表す。

- 例 1: B 君の片思いを表す論理回路

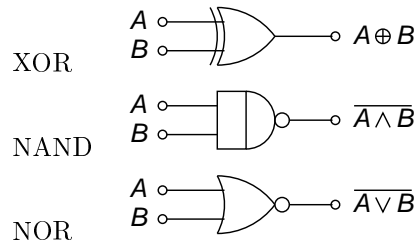


- 例 2: 片思いを表す論理回路

片思いを論理式で表すと  $(\bar{A} \wedge B) \vee (A \wedge \bar{B})$  だから、



XOR(Exclusive OR, 排他的論理和) 回路と呼ばれ、 $A \oplus B$  という論理式で表される。



### 4.3 真理値表

A	B	$A \wedge B$	A	B	$A \vee B$	A	B	$A \oplus B$
0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1	1
1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0

A	$\bar{A}$
0	1
1	0

### 4.4 論理式のいくつかの性質

- AND について

$$A \wedge B = B \wedge A \quad A \wedge (B \wedge C) = (A \wedge B) \wedge C$$

$$A \wedge A = A \quad A \wedge 0 = 0 \quad A \wedge 1 = A$$

- OR について

$$A \vee B = B \vee A \quad A \vee (B \vee C) = (A \vee B) \vee C$$

$$A \vee A = A \quad A \vee 0 = A \quad A \vee 1 = 1$$

- XOR について

$$A \oplus B = B \oplus A \quad A \oplus (B \oplus C) = (A \oplus B) \oplus C$$

$$A \oplus A = 0 \quad A \oplus 0 = A \quad A \oplus 1 = \bar{A}$$

- NOT について

$$\overline{\overline{A}} = A \text{ (2重否定)} \quad A \wedge \overline{A} = 0 \quad A \vee \overline{A} = 1$$

- 分配法則

$$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C) \quad A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$$

- ド・モルガンの法則

$$\overline{A \wedge B} = \overline{A} \vee \overline{B} \quad \overline{A \vee B} = \overline{A} \wedge \overline{B}$$

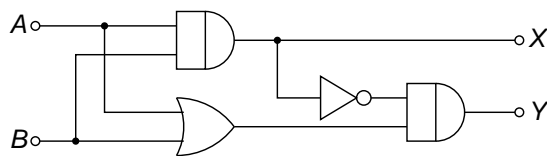
#### 4.5 論理式と真理値表

- 論理式 → 真理値表
- 真理値表 → 論理式

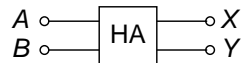
#### 4.6 加算回路

2つの1ビットの2進数  $A$  と  $B$  の和を求める。和の上位1ビットを  $X$ ，下位1ビットを  $Y$  とする。

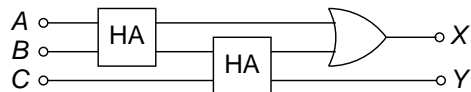
$A$	$B$	$X$	$Y$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



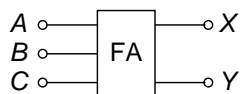
この回路は半加算回路 (Half Adder) と呼ばれる。



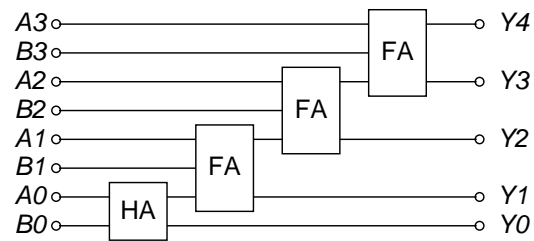
3つの1ビットの2進数  $A$ ， $B$ ， $C$  の和は次の回路で求められる。



これは全加算回路 (Full Adder) と呼ばれる。



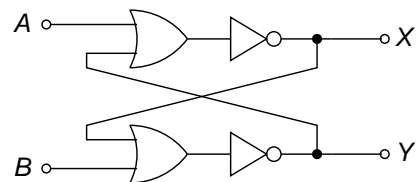
4ビットの2進数  $A_3A_2A_1A_0$  と  $B_3B_2B_1B_0$  の和  $Y_4Y_3Y_2Y_1Y_0$  は、次の回路で求められる。



- 多数決回路
- 階段のスイッチ
- 7セグメント表示

#### 4.7 記憶回路

1ビットを記憶するのに次のようなフリップフロップ (Flip Flop) 回路が使われる (スタティック RAM)。



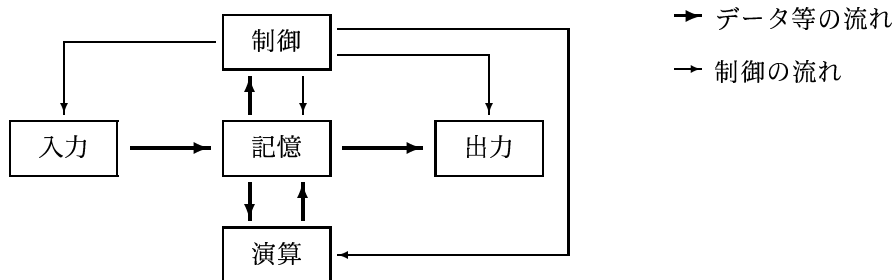
- S-R型フリップフロップ

## 5 ハードウェア (hardware)

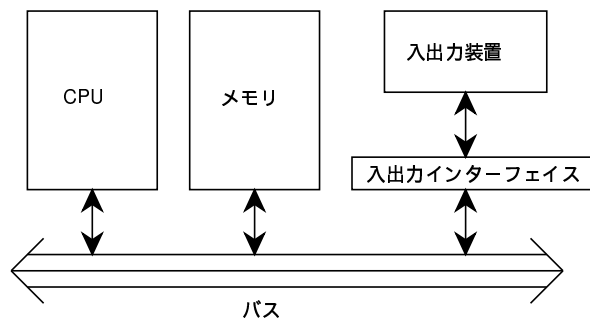
コンピュータの機械装置そのもの

### 5.1 コンピュータの5大機能

- 入力機能: 外部からデータをコンピュータ内部に取り込む機能
- 記憶機能: データやプログラムを記憶する機能
- 演算機能: データに対して計算を行う機能
- 出力機能: 結果をコンピュータ外部に出力する機能
- 制御機能: 全体がうまく働くように制御する機能



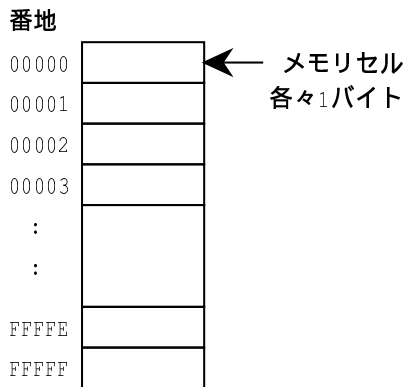
### 5.2 パソコンのハードウェア構成



- CPU (Central Processing Unit) 中央処理装置
- メモリ (memory) 記憶装置
- 入出力装置 (I/O)
- 入出力インターフェイス (I/O Interface)
- バス (bus)

### 5.3 メモリ (Memory)

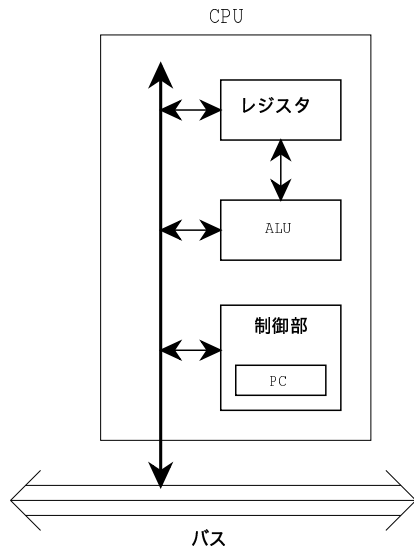
記憶機能を持つ



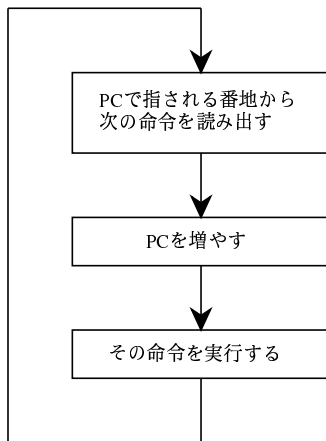
- **メモリセル** (memory cell)
- **番地, アドレス** (address)
- **メモリの種類**
  - **ICメモリ** (半導体メモリ)
  - **RAM** (Random Access Memory)
  - **ROM** (Read Only Memory)
  - フラッシュメモリ

## 5.4 CPU (中央処理装置)

制御機能と演算機能を持つ



- **レジスタ** (register)
- **ALU** (Arithmetic Logic Unit) 算術論理演算装置
- **PC** (Program Counter) プログラムカウンタ



CPU の制御機能

## 5.5 入出力装置

- 入力装置:

- キーボード
- マウス
- スキャナ
- デジタル・カメラ

- 出力装置:

- ディスプレイ (CRT, 液晶)
- プリンタ (バブルジェット, レーザー)
- スピーカ

- 補助記憶装置:

- フロッピーディスク (3.5 インチ), MO
- ハードディスク (固定ディスク)
- CD-ROM, CD-R, CD-RW

- その他:

- モデム
- ネットワーク

## 6 プログラミング言語

### 6.1 機械語とアセンブリ言語

- **機械語** (machine language): 計算機がハードウェアにより直接実行できる。2進数で表現。計算機の種類ごとに異なる (下は Z80 マイクロプロセッサの例)。

番地	メモリ	
0000	3A	} 1000番地の内容を Aレジスタにコピー
0001	00	
0002	10	
0003	47	} Aレジスタの内容を Bレジスタにコピー
0004	3A	
0005	01	} 1001番地の内容を Aレジスタにコピー
0006	10	
0007	80	} AレジスタとBレジスタの和を Aレジスタに求める
0008	32	
0009	02	} Aレジスタの内容を 1002番地にコピー
000A	10	
:		
1000		
1001		
1002		

- **アセンブリ言語** (assembly language): 機械語命令と一対一に対応。命令の略号であるニーモニックを使って表現。計算機の種類ごとに異なる (下は Z80 マイクロプロセッサの例)。

```
LD    A,(X)
LD    B,A
LD    A,(Y)
ADD   A,B
LD    (Z),A
:
X DS  1
Y DS  1
Z DS  1
```

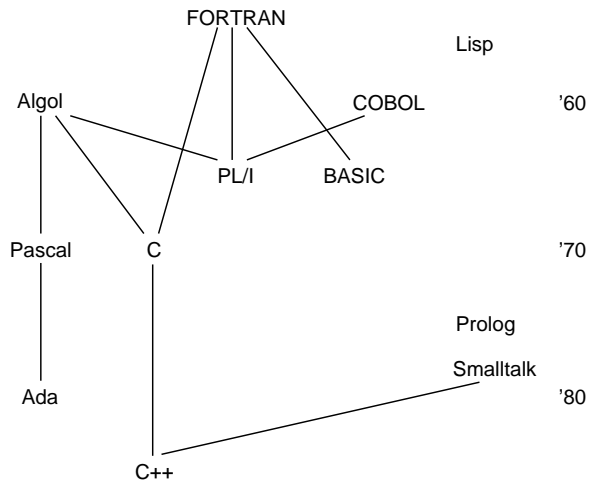
### 6.2 高水準言語, 高級言語

- **高水準言語, 高級言語** (high-level language):  
1954年, FORTRAN (FORmula TRANslation). 数式に似た形で命令を記述。

$$Z = X + Y$$

- 高水準言語の利点  
記述性, 可読性, 保守性, 互換性, 開発効率

- 高水準言語の種類



…型

手続き型, 非手続き型 (関数型, 論理型), オブジェクト指向

…向, …用

科学技術計算向, 事務計算向, 人工知能向

### 6.3 高水準言語の実現方式

- コンパイラ方式

いったん高水準言語のプログラムを機械語に翻訳したのち実行する。翻訳を行うプログラムをコンパイラ (compiler) と呼ぶ。

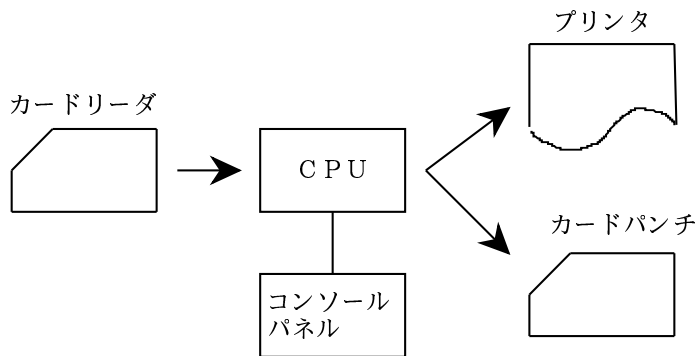
- インタプリタ方式

高水準言語のプログラムをメモリに記憶し, 各命令を実行のたびに解釈する。解釈実行を行うプログラムをインタプリタ (interpreter) と呼ぶ。

## 7 オペレーティング・システム

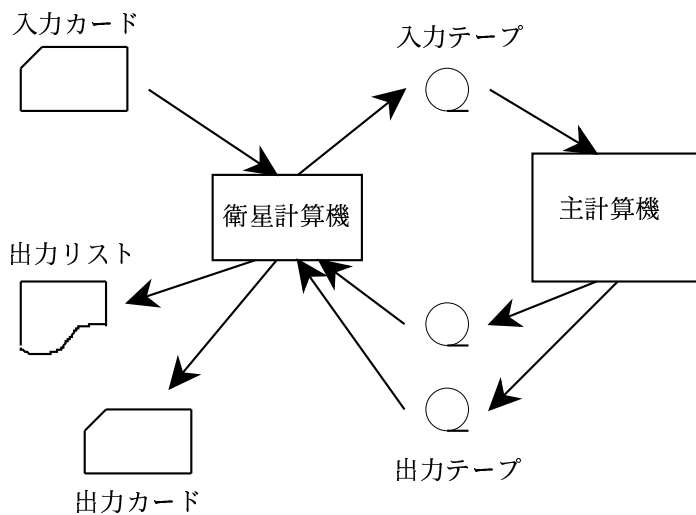
- オペレーティング・システム (OS, Operating System):  
標準的な定義はない  
「計算機資源 (ハードウェア, ソフトウェア, 人) の有効な利用を目的としたソフトウェアシステム」
- ユーザから見ると, 計算機システム=ハードウェア+OS

### 7.1 第1世代計算機のOS



- オープンショップ方式 (時間貸し)

### 7.2 第2世代計算機のOS

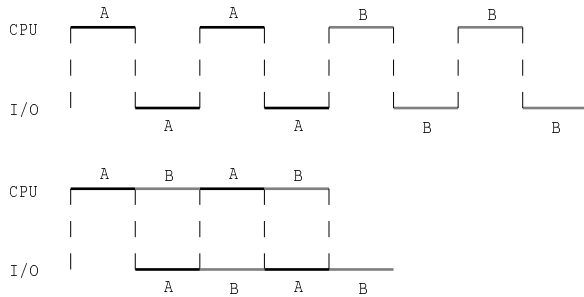


- バッチ処理 (一括処理)
- モニタの誕生 (OS の芽生え)

### 7.3 多重プログラミング

- 多数のプロセス (タスク) の並行実行  
入力プロセス, 出力プロセス, 実行プロセスの並行実行

- 割り込み処理
- スケジューリング



#### 7.4 第3世代計算機のOS

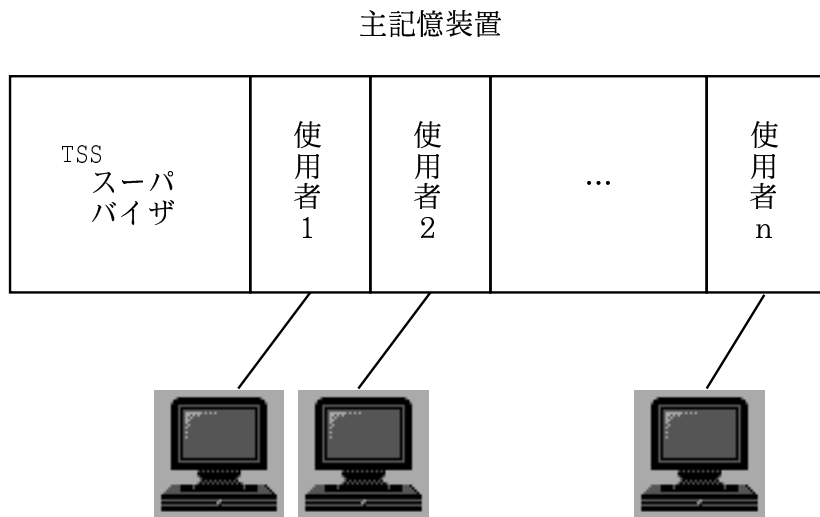
- 1965年 IBMシステム/360, OS/360  
F.P.Brooks, Jr. 著「ソフトウェア開発の神話」
- 汎用大規模OS  
事務計算と科学技術計算, 大型・中型・小型



- タスク管理
- ジョブ管理
- データ管理
- オンライン (リアルタイム) 処理
- リモートバッチ処理
- タムシェアリング処理 (×)

## 7.5 タイムシェアリング (時分割) システム

- 1963年 MIT, CTSS
- マルチユーザ (複数台の端末)



## 7.6 仮想記憶方式

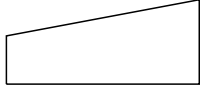


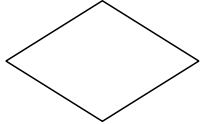
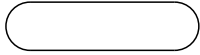
- 一次元仮想記憶方式  
あたかも大きな一次元のメモリ空間があるかのように見せる
- 二次元仮想記憶方式  
二次元アドレス空間, セグメント

## 8 フローチャート

処理手順の流れを図式化したもので、プログラムの設計や説明のために用いられる。

### 8.1 フローチャートの記号

JIS で規定されている。

意味	記号
入力	
出力	
処理	
判断	
端子	

### 8.2 フローチャートの例

- 西暦  $Y$  年が「うるう年」か「平年」かを判断する。

$Y \bmod 4 = 0$	$Y \bmod 100 = 0$	$Y \bmod 400 = 0$	
×	×	×	平年
○	×	×	うるう年
○	○	×	平年
○	○	○	うるう年

- 入力された数  $n$  の 2 進数表現を下の桁から出力する。
- 一山崩しゲーム  
人間と計算機が交互に 1 から 3 個の石を取る。全部取った方が勝ち。(繰り返し, ループ)
- 入力された数  $n$  の 2 進数表現を上 の桁から出力する。

## 9 人工知能

### 9.1 人工知能とは?

- 人工知能 (Artificial Intelligence, AI)
- マコーダック 『コンピュータは考える』

人工知能とは、すなわち人間性の真髄，理性的能力を複製する試みである。
- 「考える機械」
- 人工知能研究者の立場
  - 認知心理学的立場  
人間の知能の解明を目的として人工知能を作成する
  - 知識工学的立場  
人間の持っている知識を工学的な目的でコンピュータに載せる
  - 例え: 鳥の研究，飛行機の開発

### 9.2 人工知能の歴史

時代	出来事
紀元前 850 年頃	ホーマーの「イリアド」に現れる青銅の人間タロス
1818 年	メアリー・シェリー 小説「フランケンシュタイン」
1833 年	バベッジ 解析機関を設計
1915 年	アイザック・アシモフ 小説「わたしはロボット」
1946 年	エッカートとモークリー 電子計算機 ENIAC を開発
1950 年	チューリング チューリングテストを提案
1950 年	シャノン 論文「チェスをする機械」
1953 年	手塚治虫 アニメ「鉄腕アトム」
1956 年	サイモンとニューウェル ロジックセオリストを開発
1956 年	ダートマス会議 人工知能の提唱
1957 年	イリアック IV がイリアック組曲を作曲
1959 年	サミュエル チェッカープログラムを開発
1960 年	サイモンとニューウェル 一般問題解決プログラム GPS を開発
1961 年	スレイグル 記号積分プログラム SAINT を開発
1962 年	マッカーシー 人工知能向きプログラミング言語 Lisp を設計
1965 年	ロビンソン 導出原理を発明
1971 年	ニルソンとラファエル 計画立案プログラム STRIPS を開発
1971 年	ウィノグラード 自然言語で質疑応答できる SHRDLU を開発
1971 年	ファイゲンバーム 化学構造式決定する DENDRAL の開発
1974 年	コワルスキー 人工知能向きプログラミング言語 Prolog の提唱
1976 年	ショートリフ 抗生物質投与の助言をする MYCIN の開発
1977 年	ファイゲンバーム 知識工学の提案
1980 年	トフラール 著書「第三の波」
1982 年	第五世代コンピュータプロジェクトの開始

- ウィンストンによる時代わけ

先史時代 (1960 年頃以前),  
 黎明期 (1960 年頃から 1965 年頃),  
 暗黒時代 (1965 年頃から 1970 年頃),  
 ルネッサンス期 (1970 年頃から 1975 年頃),  
 協力期 (1975 年頃から 1980 年頃),  
 企業家の時代 (1980 年頃から)

### 9.3 第五世代コンピュータとは?

- 通産省のプロジェクト 昭和 57 年から 10 年間  
 新世代コンピュータ技術開発機構 (ICOT)
- 素子・アーキテクチャの違い  
 真空管 → トランジスタ → IC → LSI → 超 LSI → 並列計算機
- 処理の違い  
 数値処理 → データ処理 → 知識処理

### 9.4 人工知能 何ができるのか

- 「宣教師と人喰い土人」のパズル

3 人の宣教師と 3 人の人喰い土人が川の左岸にいる。1 隻の 2 人乗りボートを使って、右岸に渡るにはどうすれば良いか。ただし土人の人数が宣教師の人数よりも多くなると、宣教師は喰われてしまう。

- 一般問題解決プログラム (GPS)  
 記号化・定式化された問題を探索型推論により解決する
- 状態を記号で表す
- 可能な動作を規則として表す

- 記号積分 (SAINT)

数式処理

$$\int \frac{x^4}{(1-x^2)^{5/2}} dx = \arcsin x + \frac{1}{3} \tan^3 \arcsin x - \tan \arcsin x$$

探索型推論で解決できる。単純にしらみつぶしに探索するのではダメ。副問題の難しさを見積って、どれを解くのかを決める。

- 自然言語理解 (積木の世界, SHRDLU)

入力文: 赤い積木を黄色い積木の上に置け

↓

日本語解析プログラム

↓  
内部表現: goal(上下 (赤い積木, 黄色い積木))

↓  
**問題解決プログラム**

↓  
動作: 白い積木を黄色い積木の上から取りました  
白い積木を, テーブル上に置きました  
青い積木を赤い積木の上から取りました  
青い積木を, テーブル上に置きました  
赤い積木をテーブル上から取りました  
赤い積木を黄色い積木の上に積みました

- 分野

- ゲーム  
チェス, チェッカー, オセロ, 将棋, 囲碁
- 数式処理  
MACSYMA, REDUCE, Mathematica
- 機械翻訳
- 定理の証明
- ロボット
- 画像理解
- 音声理解
- エキスパートシステム (専門家システム)  
**DENDRAL** 質量分析スペクトルから化学構造式を定める  
**MYCIN** 細菌感染に対する抗生物質投与の助言を行う  
**PROSPECTOR** 地質学的データから鉱脈を発見する

- 能力

記憶, 推論, 類推, 学習, 発見

## 9.5 将来我々の生活がどのようなになるのか

- トフラー 『第三の波』

- 農業革命**

定住, 生活の余裕

生産者 = 消費者, 教会

- 産業革命**

単純肉体労働からの開放, 物質文明の発達

生産者 ≠ 消費者, 工場

エネルギー危機, 環境問題

## 情報革命

単純頭脳労働からの開放

生産者 = 消費者, 家庭

- 人工知能は『第三の波』を加速する

多様なニーズに応える

弱者のための社会 (医療, 教育)

精神的な満足を得るために利用