★0　オリエンテーション

★0.1　この科目について

この科目は，以下の 2 つの合併開講科目です (☆ 1).

- 数理情報学科の 2007 年度以降入学生にとっての「計算機システム II」
- 数理情報学科の 2006 年度以前入学生にとっての「計算機アーキテクチャ」

☆ 1) 2008 年度からのカリキュラム改革にともない，2007 年度以前とは内容が大幅に変わっています．2008 年度からは,「計算機アーキテクチャ概論」半分,「オペレーティングシステム概論」半分て感じです.

講義概要

この科目の前半では，計算機システムの「建築学」,「建築様式」（architecture）を考えます．PC やゲーム機は言うにおよばず，携帯電話，自動車，炊飯器など，巷には複雑な計算や機械制御のための演算装置（プロセッサ）を内蔵した機器があふれていますが，これらのプロセッサはみな共通の構成原理に基づいて設計されています．前半の目標は，その構成原理を知り，これらのプロセッサが何をしているのか理解することです．特に PC などの計算機の中核をなすプロセッサである CPU をとりあげて，その動作のしくみを学びます．また，プロセッサに命令を与えるための言葉であるアセンブリ言語の初歩についても学びます．

後半は，オペレーティングシステム (OS) の役割・仕組みを考えます．Windows, Linux, Mac OS X といった OS の名前を聞くと，それらの OS と共に提供されるユーザインタフェースの部分，デスクトップ画面や操作性のことをついつい思い浮かべがちですが，それらは単なる見た目に過ぎません．後半の目標は，OS の中核部分の地味だけれど大切な機能について理解することです．

到達目標

人がゲームしたりプログラム実行したりしてる裏で CPU や OS が何をしてるのか，イメージが湧くようになるといいですね．身近な計算機が 1 秒間でどれだけのことができるのかを見積もってみて「イマドキの計算機はすごいねぇ」と思ったり，単なる画面の見た目ではない OS の機能について思いをはせたりできるかもしれません．

系統的履修

「情報処理の基礎／情報処理システム I」と「計算機システム I／情報処理システム II」のつづきです．

成績評価の方法

平常点 30%(☆ 2)+ 定期試験 70%.

☆ 2) ReLS（龍谷大学 e-Learning システム）を利用して毎週行なう予定の小テストの得点を含む

テキストと参考文献

テキストは特に指定していません．参考書についてはウェブシラバスやこの科目の web ページを参照してください．

いろいろ

- 真剣に授業に参加している人の邪魔をする行為（おしゃべり，途中入退室など）は禁止.
- 大学の授業は，授業時間の他にも自学自習することを前提に作られています．龍谷大学の講義科目の場合，自習時間は講義時間の 2 倍とされています．授業時間以外にも勉強することが必要です.
- 介護体験などやむを得ない理由で欠席した場合，有効な欠席届をすみやかに高橋に手渡してください (☆ 3).

☆ 3) 提出が遅れると受け取らないことがあります.

★ 0.2 ReLS 上での小テストについて

ReLS（龍谷大学 e-Learning システム）を利用して授業時間外に毎週小テストを行なう予定です．毎回の授業終了後，指定の締切日時までに受けてください．ReLS のサイトを利用可能であれば，夜中に自宅で受けたりすることも可能です．ReLS へのアクセス法は↓を参照してください．

注意

- 講義資料などを参照しながら解答して構いませんが，自力で解答しなければいけません（不正を行なった場合はしかるべき処置をします．履修要項参照）
- どのような理由で欠席した場合でも，ReLS 上の小テストは原則として免除になりません．自習して解答してください．

★ 0.3 アクセス

この科目に関するウェブサイト

(1) 高橋のウェブページ (☆ 4)

`http://www-tlab.math.ryukoku.ac.jp/wiki/`

から「時間割」→「CompSysII」 とたどると，この科目のページにたどりつけます．

☆ 4) 高橋のページにたどりつくには，この URL をブラウザに直接入力するかわりに，理工学部や数理情報学科のウェブサイトからたどったり検索したりする手もありますね．`www.math.ryukoku.ac.jp`

(2) 龍谷大学 e-Learning システム (ReLS) 上のこの科目のページ

このサイト上で毎週小テストを行なう予定です．

このページにたどりつくには，上記のこの科目のページからリンクをたどるのがはやいでしょう (☆ 5)．途中で，全学認証のユーザ名とパスワードでのログインが求められます．

☆ 5) 龍大ポータルサイトからも行けますが，だいぶ遠回りです

はじめてこのページへ行こうとすると，ログインした後に,「このコースへ登録するためには「登録キー」というワンタイムパスワードが必要です」といわれるでしょう.「登録キー」は，初回の授業でお教えします．

高橋の連絡先など

- 研究室: 1–508（または 1–602）`e-mail: takataka[at]math.ryukoku.ac.jp`
- 高橋の 2012 年度後期の週間スケジュールは以下の通りです．この科目に関する質問等がある場合は，以下の OfficeHour の時間帯に訪問してください．最新の情報はウェブ上および研究室の前に掲出してあります．

	月	火	水	木	金
1					Vision
2			(学科会議)		
昼休み				OfficeHour	
3				OfficeHour	
4	AProg		(教授会)	SJE	CompSysII
5	AProg		(教授会)	SJE	

(ほげ) — あったりなかったりなイベント

ここに示したもの以外に，会議・セミナ・出張等が不定期であります．

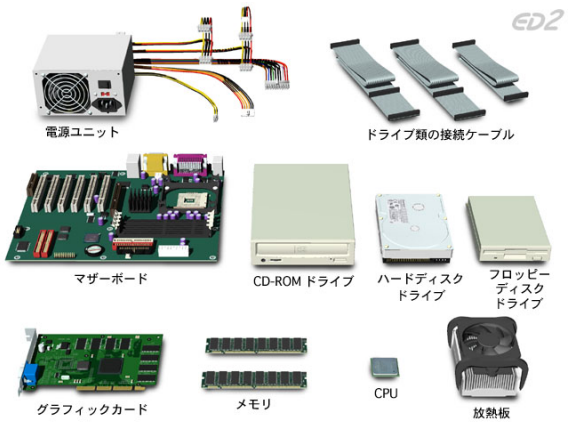
★1　そこにもここにも CPU

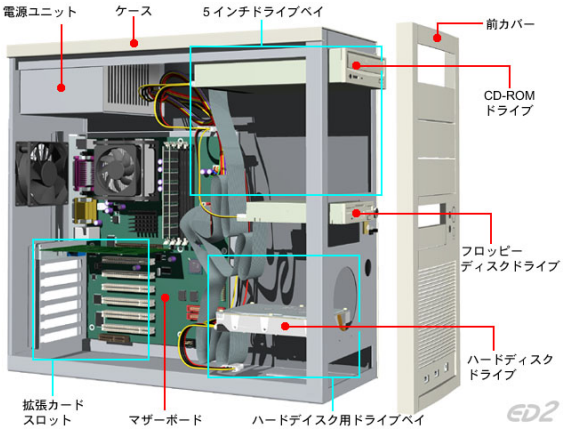
現代では，PC に限らず様々な機器に CPU（MPU またはプロセッサともいう）が組み込まれている．

PC: パーソナルコンピュータ
CPU: Central Processing Unit, MPU: Micro-

PC の中身

PC の機能の中心的役割を担うのは，CPU である．これがマザーボード上の回路を介してメモリやグラフィックカード，ハードディスクドライブなどとデータをやりとりすることで，様々な情報処理が行われる．



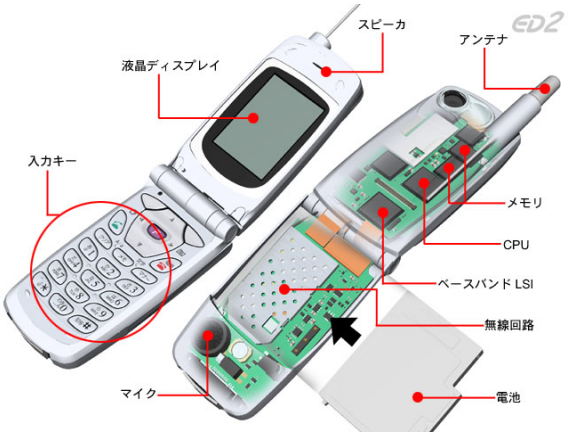


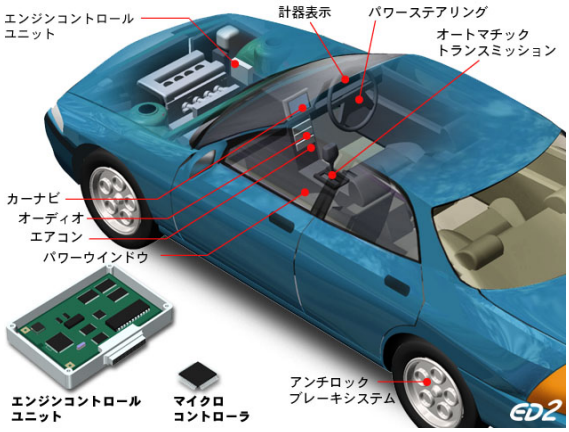
携帯電話，自動車の中にあるプロセッサ

携帯電話にも CPU が搭載されており，アドレス帳の管理，着信メロディの再生など様々な情報処理を行っている．これらは 32bit(☆ 6) の高性能プロセッサが主流で，Java プログラムの実行など PC なみの高度な情報処理が行えるようになっている．また，大容量のメモリも搭載されている．

一方，1 台の自動車の中には，近頃では数十個以上ものプロセッサが搭載されている．その中には，エンジンやトランスミッションを制御するための高性能なものもあれば，計器表示やパワーウィンドウの制御などを個別に行う安価な（8bit や 16bit の）ものもある．

☆ 6) 「32bit のプロセッサ」，「8bit のプロセッサ」という表現の意味については，次回以降考えていきます．ただし，厳密な定義があるわけではありません．

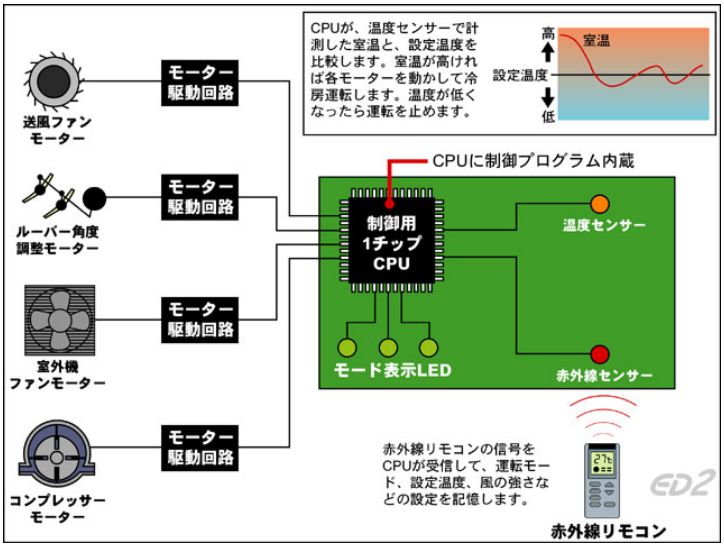




この資料中の「ED2」というロゴのある画像は，教育目的での利用のために提供されている「情報機器と情報社会のしくみ素材集」のものです．
http://kyoiku-gakka.u-sacred-heart.ac.jp/jyouhou-kiki/

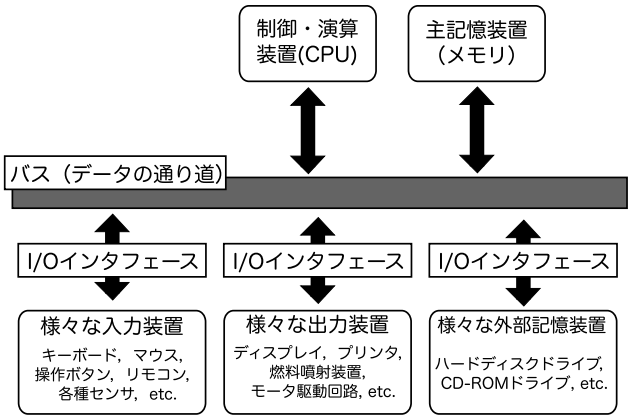
1 チップマイコン

家電製品の場合，1 チップマイコンと呼ばれるプロセッサを搭載しているものが多い．これは，プロセッサチップの中にメモリや様々な制御回路もひとまとめに内蔵したもの．家電製品に限らず，世の中には 1 チップマイコンを搭載した機器があふれている．



★ 1.1 情報機器に共通のしくみ

CPU が，メモリに記憶されている命令にしたがって，四則演算や各装置間でのデータの転送などを実行する (☆ 7).



☆ 7) この図では 1 本のバスに全ての装置が接続されているが，最近の PC などでは，独立したバスを複数用意して（例えば CPU とメモリ間専用など）性能向上をはかっているものも多い．

I/O: Input/Output, 入力/出力

★ 1.2 なぜ計算機アーキテクチャを学ぶのか

- 消極的理由: 単位が必要だから.「情報」と名のつく理系学科を卒業したと言い張るためには計算機アーキテクチャは常識かもしれへんから.
- 積極的理由: 世の中のいろんなものにコンピュータが組み込まれてる → やぱしその仕組み知りたいやんか

「でもソフトウェア技術者をめざすなら CPU の構成なんか知らんでも C とか Java と かでプログラム書けたらそれでええんちゃう？」

ソフトウェア技術者がみな PC 上で動作するプログラムを開発しているわけではなく，様々な機器に組み込まれた CPU のためのソフトウェアを開発する技術者（組み込みソフトウェア技術者と呼んだりする）もたくさんいる．C や Java などの「高級な」言語を用いる PC 上でのソフトウェア開発と違い，組み込み系エンジニアはよりハードウェアのレベルに近い立場で開発することになるため，計算機アーキテクチャを理解していることが求められる．また，PC 上でのソフトウェア開発の場合でも，アーキテクチャを理解していることでより優れた（CPU の性能を生かして高速に計算できるとか，メモリを節約できるとかそういう）ソフトウェアを作ることができる．

★2　論理回路から CPU のおおまかなしくみへ

Q1. $(6)_{10}$(☆8) と $(3)_{10}$ を 6bit の 2 進数で表しなさい
Q2. 2 の補数表現を用いて，$(-3)_{10}$ を 6bit の 2 進数で表しなさい．
Q3. 6bit の 2 進数で表現できる最大と最小の数はそれぞれいくつか．ただし負数には 2 の補数表現を用いるものとする．
Q4. $6-3$ の演算を 6bit の 2 進数で行う過程を説明しなさい．
Q5. りんご，みかん，マンゴー，ドリアンの 4 つを区別するためにそれぞれに 2 進数を割り当てたい．最小何 bit の 2 進数にすればよいか．

☆8) $(6)_{10}$ は 10 進数の 6 のこと．複数の進法が混じって紛らわしいときに使うことがある．

★2.1　加算器

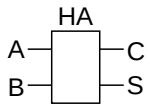
「情報処理の基礎」や「計算機システム I」で学んだように，**AND ゲート**や**OR ゲート**などの**論理素子**を組み合わせると，1bit の 2 進数同士の加算を行う論理回路（**半加算器**）を構成することができる．半加算器は，下位桁からの**桁上げ**入力も考慮した**全加算器**の部品となる (☆9)．全加算器を複数つなぎあわせると，複数桁から成る 2 進数同士を加算する回路を構成できる．

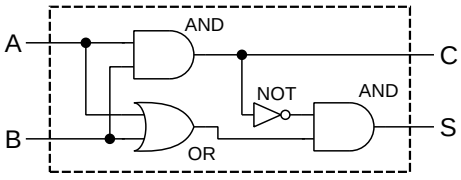
半加算器: Half Adder
全加算器: Full Adder

☆9) 半加算器 2 つを組み合わせるかわりに，AND や OR を直接組み合わせて全加算器を構成することもできる．

半加算器

入力—A と B
出力—A と B の和 S と桁上げ出力 C



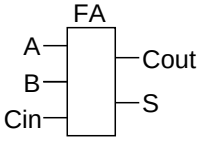


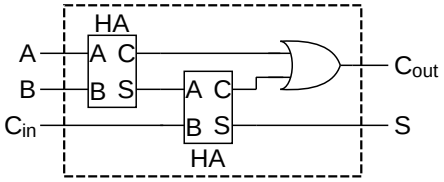
真理値表

A	B	S	C
0	0		
0	1		
1	0		
1	1		

全加算器

入力—A，B と下位桁からの桁上げ入力 C_{in}
出力—A，B の和 S と桁上げ出力 C_{out}

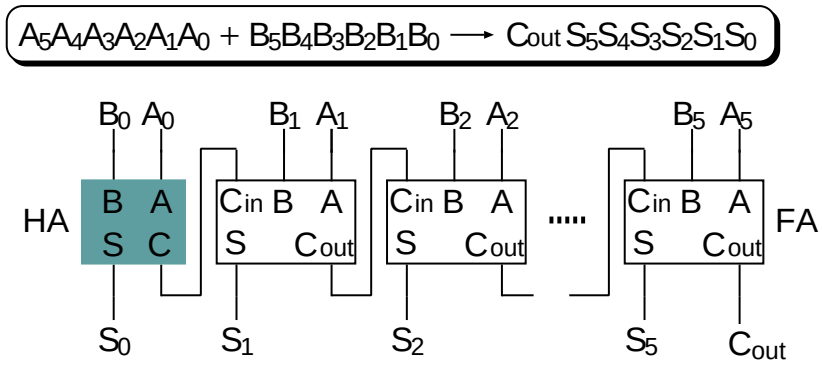




真理値表

A	B	C_{in}	S	C_{out}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

複数桁の加算器（6bit の加算器の例）



［発展］ 左図のような回路構成では，A0+B0 の桁上げを使って A1+B1 を計算→その桁上げを使って A2+B2 を計算→…というように計算が進む．そのため，桁数が多くなると信号が何段もの論理回路を通過しなければならず，計算にかかる時間が長くなる．実用的な加算器ではこのような問題を避ける工夫がなされていることが多い．

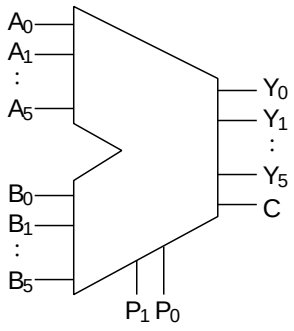
★ 2.2 算術論理演算ユニット

算術論理演算ユニット (ALU): 加算や減算などの算術演算や，ビット反転，論理積 (AND) などの論理演算を 行うハードウェア．乗除算や平方根計算などの機能を備えたものもある．

ALU では，演算に用いる数だけでなく演算の種類も入力として指定できるようになっている．CPU の中では ALU が加減算や乗算などの演算を行っている．

ALU: Arithmetical Logical Unit

●単純な ALU の例



A, B: 6bit の入力，Y: 6bit の出力，C: 桁上げ出力
P: 実行させたい演算の種類を表す入力

P は，例えば以下のように設定される

P_1	P_0	演算の種類
0	0	加算: $A + B \to Y$
0	1	減算: $A - B \to Y$
1	0	ビット毎の OR: A OR B $\to Y$
1	1	ビット毎の AND: A AND B $\to Y$

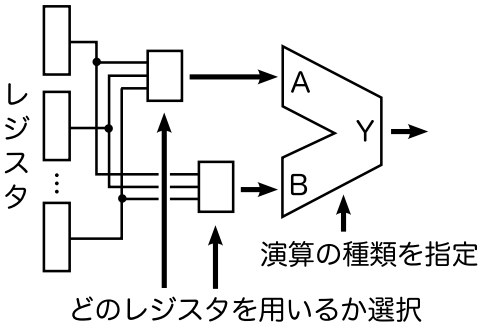
Q6. 上記の ALU で，A が 000011，B が 000010 のときに P を 00 とする (☆ 10) と Y はいくつになるか．P が 01 のときはどうか．

Q7. 上記の ALU に機能を追加して，8 種類の演算を実行できるようにしたとする．このとき，P には最低何 bit 必要か．命令の種類数が 6 の場合はどうか．

☆ 10) ここでは，「P を 01 にする」は，$P_1 = 0, P_0 = 1$ という意味とする．

★ 2.3 レジスタ

CPU の中には，ALU への入力などを一時的に保持したり演算結果を書き込むための記憶領域が用意されている．これを**レジスタ**という．レジスタは複数用意されるのが普通で，その場合，どのレジスタの内容を ALU に入力するかを指定するための選択回路が必要となる（同様に，演算結果をどのレジスタに書き込むかの指定も必要（図では省略している））．



Q8. 上図のような ALU と 4 つのレジスタが接続された CPU を考える．ALU への入力の片方（例えば A）にどのレジスタを用いるか指定するためには，最低何 bit の 2 進数を用いればよいか．レジスタの数が 6 の場合はどうか．

★ 2.4 CPU の大まかなしくみ

ALU，レジスタ，演算の種類や使用するレジスタを選択する回路の 3 つを要素と考えれば，CPU の大まかなしくみを理解することができる．

★肝腎のところなので，詳しくは講義で説明します．

★3　CPU のあれこれ

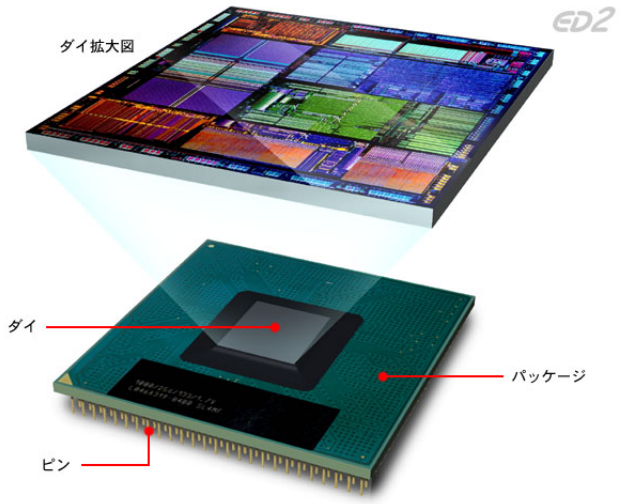
● CPU の構造

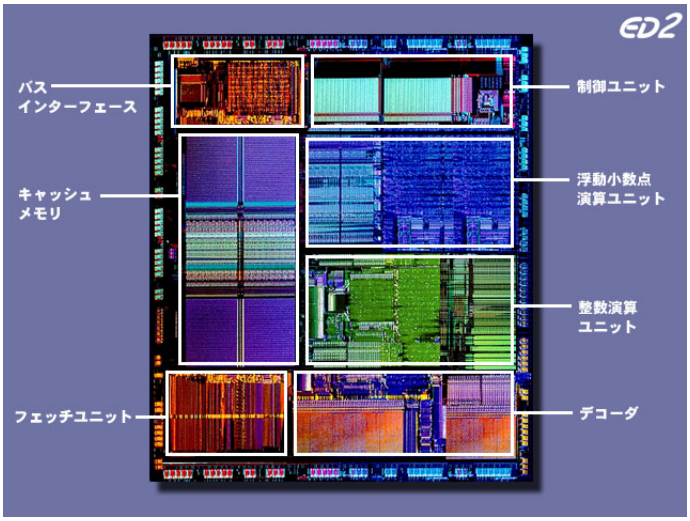
現在一般的な CPU は，ダイと呼ばれる小さく（指先にのる位のサイズ）薄いシリコン基盤の表面に，トランジスタを用いた回路を形成したものとなっている．回路は印刷技術を応用した方法で形成されている．その最小描画寸法（プロセスルール）は，2011 年現在で 45〜32nm (1nm = 10^{-9} m = 10 億分の 1 m) 程度となっている．

一つのプロセッサには，トランジスタなどの素子が数百万個から数億個集積されている．

参考（wikipedia 「数量の比較」より）：

- 髪の太さ: 0.1mm 弱 $= 1 \times 10^{-4}$ m 弱
- ヒト赤血球の直径: $6-8\mu$m $= (6-8) \times 10^{-6}$m
- HIV ウイルスの大きさ: 90nm $= 90 \times 10^{-9}$m





● 動作周波数

CPU を含めて現在一般的な論理回路では，回路内の論理素子は外部から与えられる周期的な信号（クロック信号）に同期して動作する（タイミングをあわせて 0/1 が変化する）．その動作の周波数は，数 GHz（1GHz は 10 億 Hz）程度に達している．ただし，この授業でこれから学ぶことだが，「動作周波数が高い＝性能が高い」とは限らないことに注意．

● CPU の進歩

例えば，インテル社のいくつかの CPU（多くは電卓や PC に搭載されてきた）のトランジスタ数と動作周波数の推移は次のようになっている (☆ 11)．

年	名称	トランジスタ数	プロセスルール	動作周波数
1971	4004(☆ 12)	2,300		数百 kHz
1978	8086	29,000		数 MHz
1985	i386	275,000	1.5-1μm	数十 MHz
1993	Pentium	3,100,000	0.8-0.25μm	60MHz 以上
2000	Pentium4	42,000,000	130-65nm	1GHz 以上
2006	Core 2 Duo	291,000,000	65-45nm	1-3GHz 程度

集積回路の進歩に関しては，「ムーアの法則」が有名．
ムーアの法則： インテル社の G.E.Moore が 1965 年に発表した，「半導体集積回路に集積されるトランジスタ数は 18 から 24ヶ月で倍増する」という経験則 (☆ 13)．何十年も成り立ってきた（ねずみ算式にものすごい勢いで集積度が上がり続けてきた）が，そろそろ転換点を迎えつつあるかもしれない．

☆ 11) CPU よりも DRAM 等の方が同じ時代により高い集積度を実現していることが多い．これは，DRAM 等の方が構造が単純だからである．

☆ 12) 世界初のマイクロプロセッサ

Q9 現在の PC 向け CPU の例として，インテル社の「Core i7」について，左記の値がそれぞれどれ位かウェブや文献で調べてみよう．

☆ 13) 例えば 2 年で倍なら，10 年で $2^5 = 32$ 倍，20 年で $2^{10} = 1024$ 倍，30 年で $2^{15} = 32768$ 倍．