第1回・情報学基礎

コンピュータの誕生と情報のデジタル化

平石 拓　　京都橘大学 工学部 専任講師

京都橘大学
KYOTO TACHIBANA UNIVERSITY

講義スケジュール 全10回

- #01 **コンピュータの誕生と情報のデジタル化**
- #02 n進法
- #03 集合
- #04 演習：離散数学基礎(1)
- #05 命題と論理
- #06 帰納法
- #07 演習：離散数学基礎(2)
- #08 グラフ理論
- #09 木，有限オートマトンと正規表現
- #10 演習：離散数学基礎(3)

00

授業科目の概要

授業科目の概要

●コンピュータの歴史，および仕組みの概要を学ぶ

- ✓ コンピュータにどのように情報が保存されるか，どのようにプログラムが動作するかの概観を学ぶ.

●情報技術，プログラミングを学ぶうえで必要となる数学知識（主に離散数学）を学ぶ

- ✓ n進法
- ✓ 集合と命題論理
- ✓ 数学的帰納法
- ✓ グラフ理論
- ✓ 正規表現と有限オートマトン

●キーワード

●デジタル化 離散数学 n進法 集合 命題論理 帰納法 グラフ理論 正規表現

今回の内容

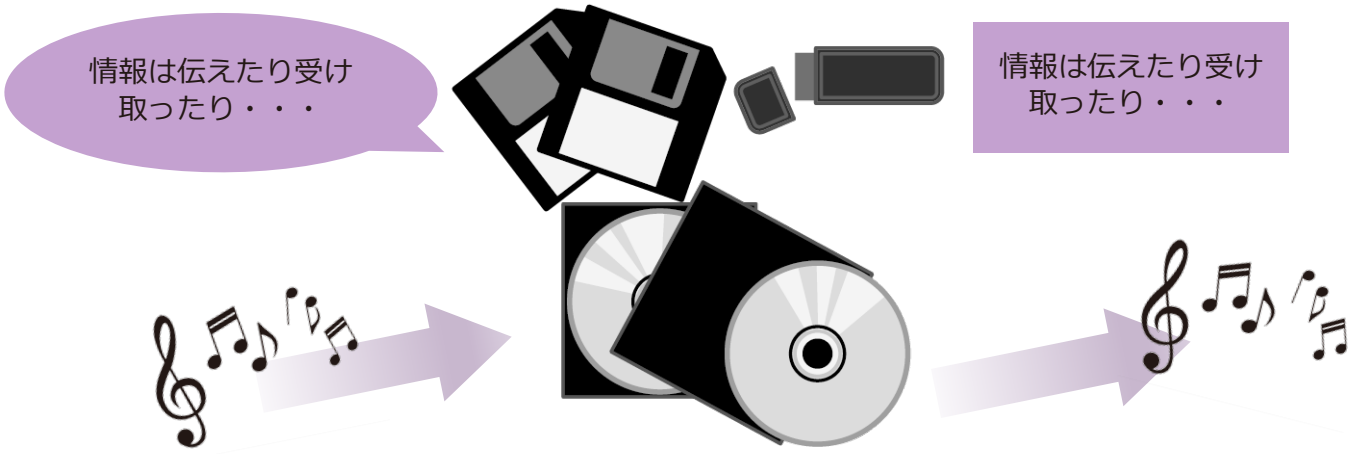
1. コンピュータの歴史
2. コンピュータと2進数
   - コンピュータの仕組みの基本：トランジスタ，IC，LSIとコンンピュータ
   - コンピュータの性能の急速な進歩（ムーアの法則）
3. 文字・画像・音声のデジタル化
   - ビット列（0と1）による情報の表現（文字・音声・画像）
   - 情報の圧縮技術

01

コンピュータの歴史

情報とコンピュータ

- 情報は伝えたり受け取ったりするだけでなく、「記録」して同じ内容を何度も伝えられるようになることが重要
- 情報をすばやく記録したり、分類したり、並べ替えたり、比較したり、結びつけたり、表示したりするような技術をひとまとめにした装置が「コンピュータ」



足し算を実現する方法
～交通量調査で使われるカウンター～

- カウンター（数取器）
  - 年に１回程度、交差点などで、車の台数を調査する人を見かけることがある
  - 調査員は右図のようなカウンター（数取器（かずとりき）と呼ばれる）を用い、ボタンを押すと数字が１増える

足し算を実現する方法
～数取器で足し算器が作れる？～

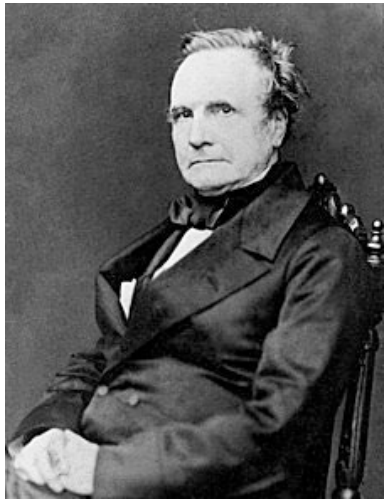
数取器の仕組み

- 数取器はボタンを押した回数だけ、数字が増える
- １の位のボタン、10の位のボタン、100の位のボタンを作れば、１回のボタンを押すたびに、10ずつ増やしたり、100ずつ増やしたりできる数取器も作れる？
- このような仕組みを用いると機械式の足し算器が作れる
- 昭和30年代（1960年）ぐらいまでは、機械式の計算機が日本でも売られていた
- シャープ（当時は早川電機）が1964年3月に発表したわが国最初の電卓CS-10Aは、世界で最初のオールトランジスタ型の電卓といわれており、
  価格は１台53万5000円だった

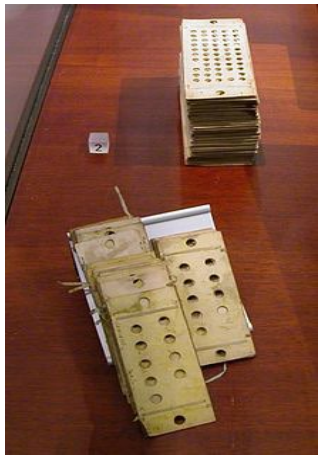
(参考) 情報処理学会, コンピュータ博物館 "CS-10A", http://museum.ipsj.or.jp/heritage/CS-10A.html

バベッジの解析機関

- チャールズ・バベッジ（1791-1871）
  - イギリスの数学者
  - 「コンピュータの父」と呼ばれる
  - パンチカードでプログラムを組むことが可能
  - 入力、演算、記憶、出力からなる系列でプログラムを作成
  - 機械は未完成



https://en.wikipedia.org/wiki/Charles_Babbage

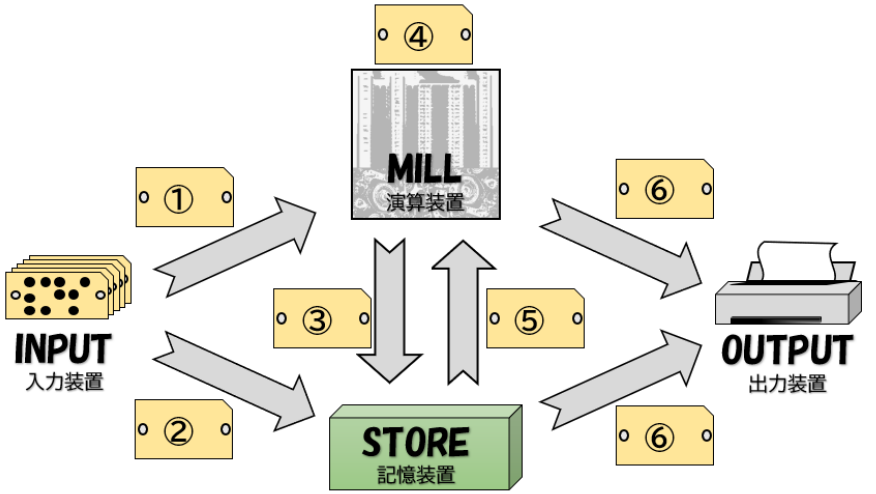


パンチカード

https://en.wikipedia.org/wiki/Analytical_engine

1枚のパンチカードでいずれかの処理をそれぞれ行うことが可能

① 数を演算装置へ入力する
② 数を記憶装置へ入力する
③ 数を演算装置から記憶装置へ転送する
④ 演算装置に処理を命令する
⑤ 数を記憶装置から演算装置へ転送する
⑥ 記憶装置または演算装置の数を出力する

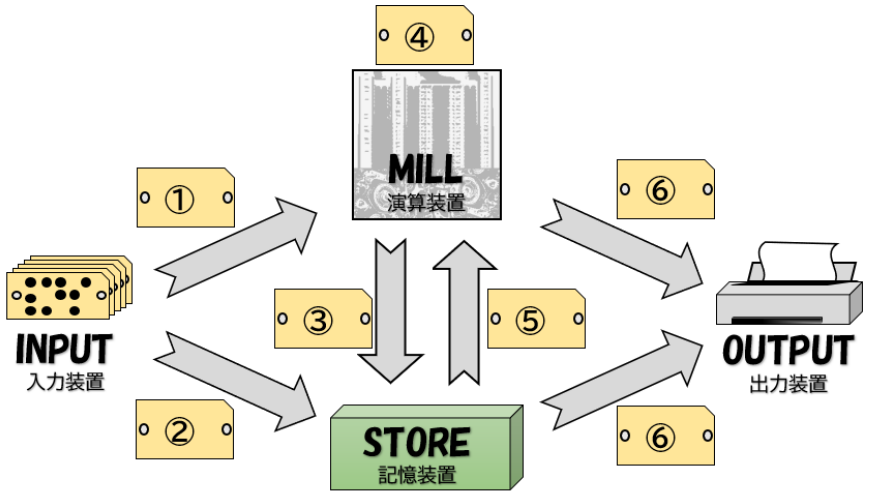


解析機関のプログラム

- Y=2X+3を計算するプログラム
  - 数を記憶装置 X へ入力
  - 数 X を演算装置 R へ入力
  - 演算装置で R の値を 2 倍に
  - 演算装置に 3 を加算
  - 演算装置 R の数を出力
- Z=X+1を計算するプログラム？
  - 数を記憶装置 X へ入力
  - 数 X を演算装置 R へ入力
  - 演算装置に 1 を加算
  - 演算装置 R の数を出力

1 枚のパンチカードでいずれかの処理をそれぞれ行うことが可能

① 数を演算装置へ入力する
② 数を記憶装置へ入力する
③ 数を演算装置から記憶装置へ転送する
④ 演算装置に処理を命令する
⑤ 数を記憶装置から演算装置へ転送する
⑥ 記憶装置または演算装置の数を出力する



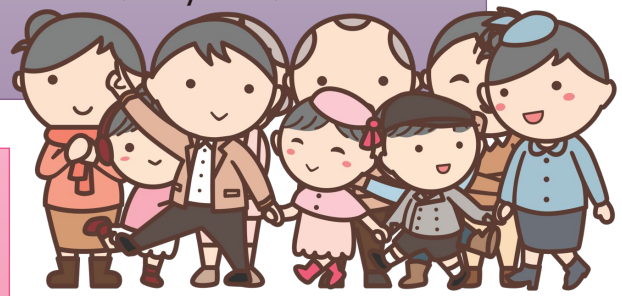
ここでは、演算装置Rの値の２倍を計算したり、Rの値に定数（1とか3）を加えたりできるものと仮定している

- **パンチカードの内容を変える**ことで**異なる演算**を実行可能

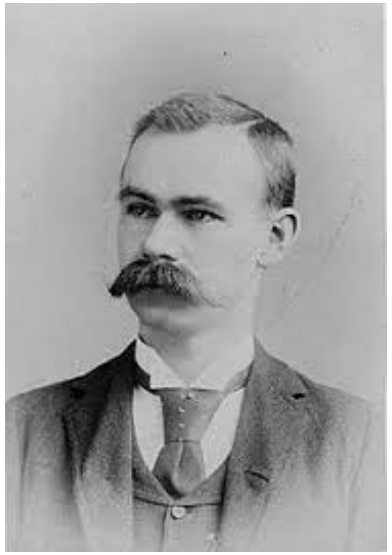
ホレリスの人口統計機

- ハーマン・ホレリス（1860-1929）
  - アメリカの発明家
  - 人口統計機を開発

子どもは何人ですか？
a)　0-2人 b) 3人 c) 8-20人 d) 20人以上
宗教は何ですか？
a) キリスト教 b)イスラム教 c) 仏教
d) その他



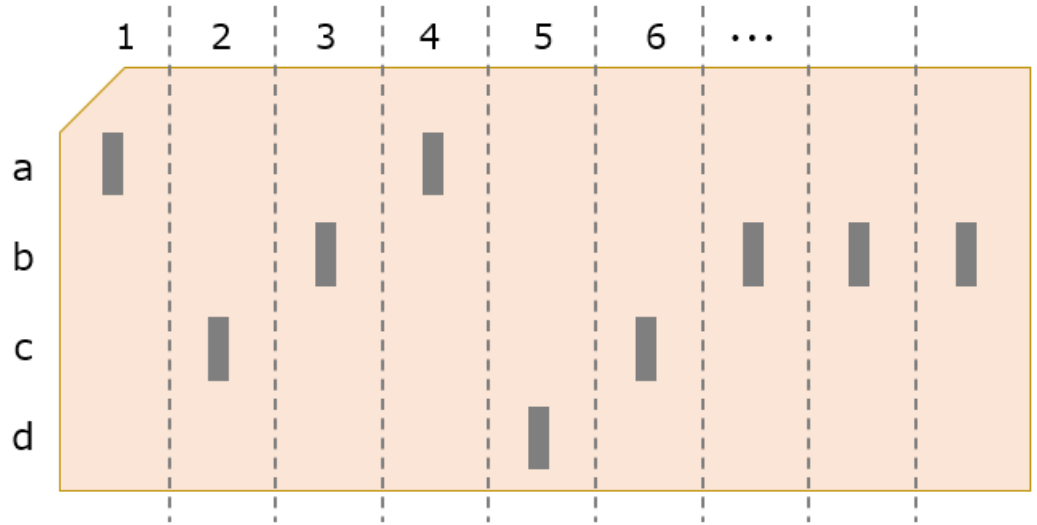
- 市民の総数は？
- 子どもが0-2人の人の数は？
- キリスト教徒で子供が3人の人の数は？

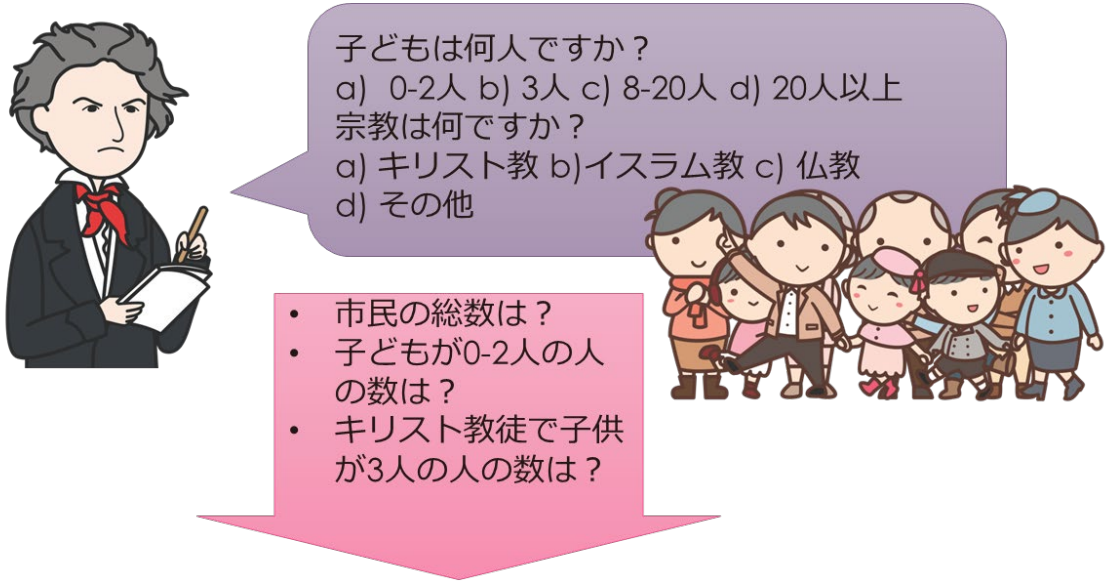


https://en.wikipedia.org/wiki/Herman_Hollerith

ホレリスの人口統計機

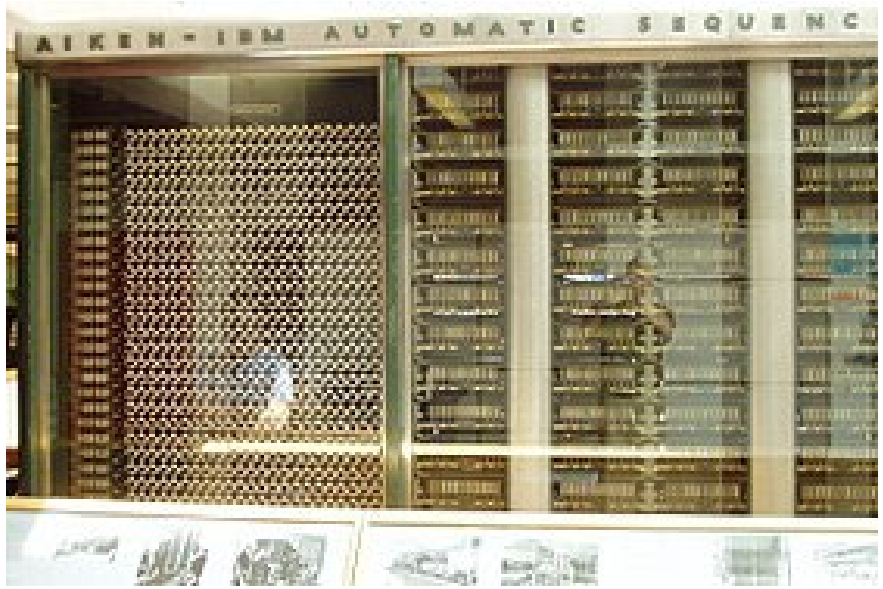
- １ドル紙幣大のパンチカードに各人の回答を打ち込むよう提唱
- 下図は１番の問題の答はａ，２番の問題の答はc，…
  であることを表している





電磁式コンピュータMARK-1

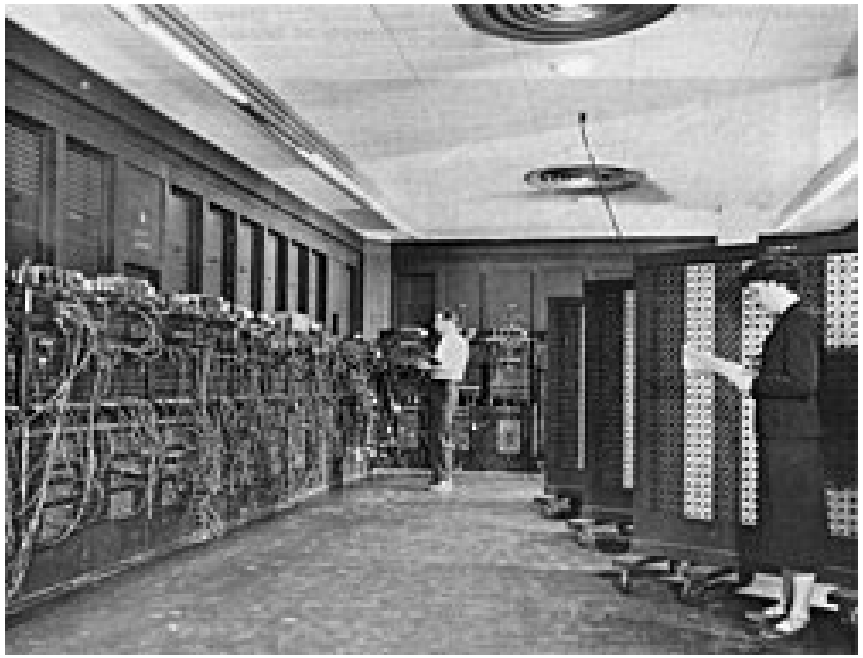
- ハーバード大学とIBM の共同研究により、アメリカ海軍が1944年に巨大な電磁式コンピュータMARK-1を開発
- モデルはバベッジの解析機関
- スイッチ、リレー、歯車式の計算装置、クラッチなと765,000個の電気機械部品と数百kmの電線を使って作られた
- 全長16m、高さ2.4m、奥行き約60cm、重量約4.5t
- MARK-1は約３秒で10桁の数２つの掛け算を実行することができた



https://ja.wikipedia.org/wiki/Harvard_Mark_I

真空管式計算機

- **ENIAC**
  - アメリカ陸軍の弾道研究所での砲撃射表の計算を第一の目的として設計された電子計算機、1946年（第２次世界大戦後）完成
  - 18000個の真空管
  - １秒間に500演算
  - 空調装置も必要

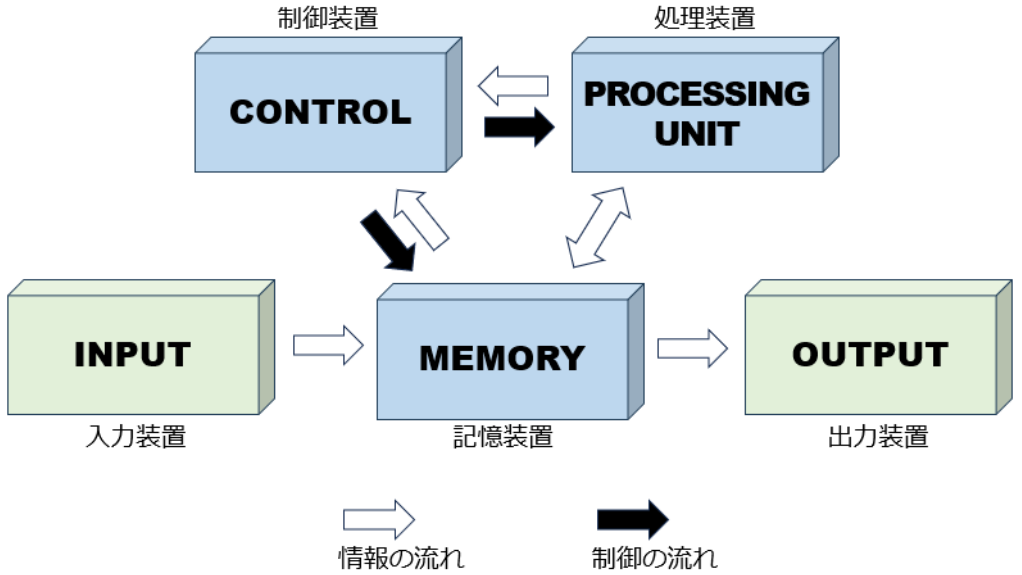


https://ja.wikipedia.org/wiki/ENIAC

コンピュータは記憶装置付きの計算ロボット

（現在の）コンピュータは、**入力装置、処理装置、制御装置、記憶装置、出力装置**からなる情報処理装置

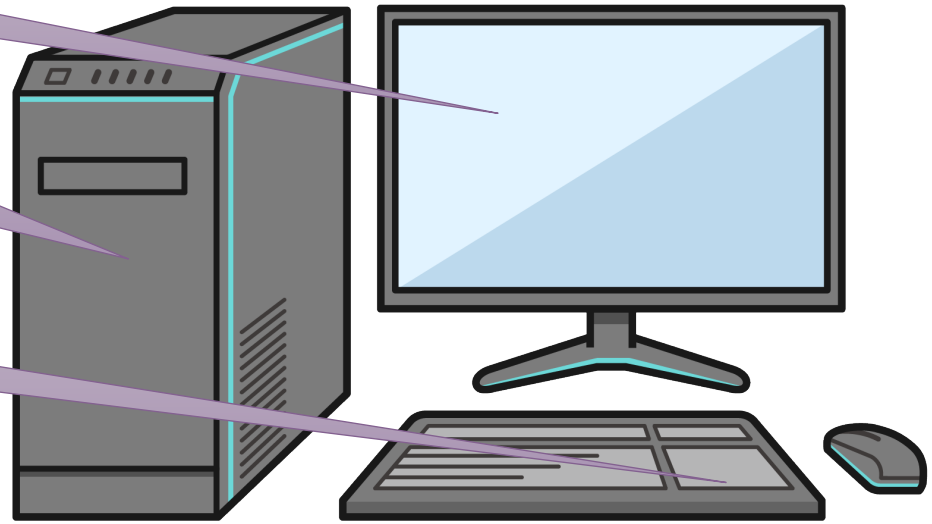
データだけでなく、処理内容（プログラム）も記憶装置に保存されることがポイント



出力装置（モニタ，プリンタ）

処理装置・制御装置（CPU）
記憶装置（メモリ，ディスク）

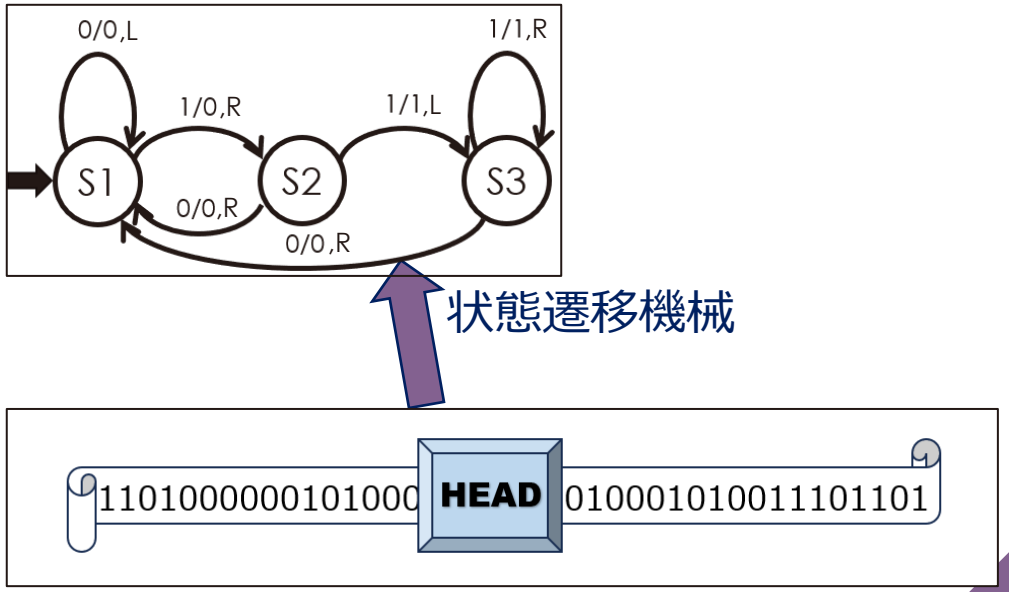
入力装置（キーボード，マウス）



チューリング機械

- イギリスの数学者のアラン・チューリング（1912-1954）が考えた現在のコンピュータの原型
- 記号（0/1）の列が記録されている無限の長さのテープと、テープの記号を1つずつ読み書きできるヘッドを備えただけの単純な機械で、現在のコンピュータでできる計算はすべてできること、コンピュータのプログラムをテープの上に書き込めること、などを**数学的に**示した
- 入力（テープの初期値）、処理装置（テープの読み書き装置）、制御装置（チューリング機械を動かす装置）、記憶装置（テープ）、出力（テープの最終結果）からなる情報処理装置でコンピュータが動く

https://en.wikipedia.org/wiki/Alan_Turing

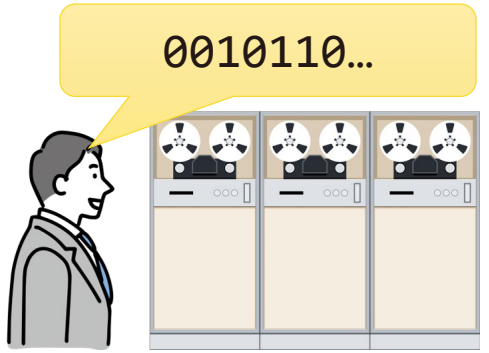


プログラム内蔵方式

- ジョン・フォン・ノイマン（1903-1957）
  - 米国プリンストン大学の数学者
  - 現在の大多数のコンピュータで採用されている
    プログラム内蔵方式の仕組み
    （**ノイマン型コンピュータ**）を考案

ノイマンの提唱

1. 記憶装置にしまっておける形に命令を符号化する方法を開発すること
   - ノイマンは0と1の組み合わせを提案



2. 仕事に必要なあらゆる情報に加え，命令も記憶装置に保存

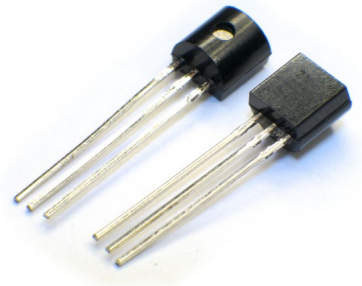
3. プログラムを実行する際は，命令をパンチカードから読み取るのではなく記憶装置から持ってくる

→ **プログラム内蔵方式**

02

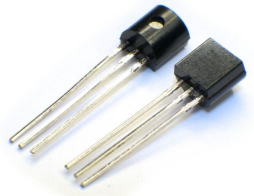
コンピュータと2進数

トランジスタの発明



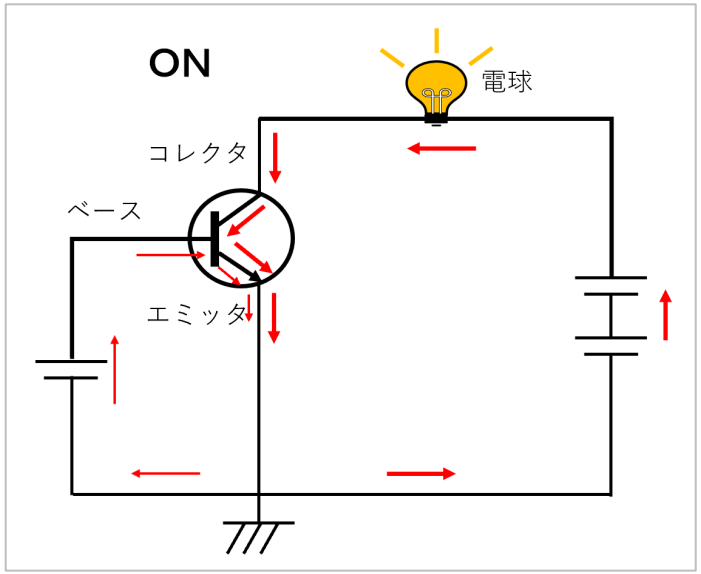
- **トランジスタ（英: transistor）**
  - 電子回路において、**信号を増幅またはスイッチング（ON/OFFを切り替え）**することができる半導体素子
  - 1947年、ベル研究所の理論物理学者ジョン・バーディーンと実験物理学者ウォルター・ブラッテンが、高純度のゲルマニウム単結晶に、きわめて近づけて立てた2本の針の片方に電流を流すと、もう片方に大きな電流が流れるという現象を発見した。これが最初のトランジスタである点接触型トランジスタの発見
  - 固体物理学部門のリーダーだったウィリアム・ショックレーは、この現象を増幅に利用した。この研究は、固体による増幅素子の発明として、1948年6月30日に3人の連名で発表された。この3人は、この功績により1956年のノーベル物理学賞を受賞
  - トランジスタには、増幅作用とスイッチング作用を持ち、スイッチング作用はメカニカルなリレースイッチの代わりに利用されるようになり、トランジスタはコンピュータの主要部品として使われるようになった

トランジスタのスイッチング作用



- トランジスタとは**電気的に**ON/OFFがコントロールできるスイッチのようなデバイスである
- トランジスタの持つ３つの端子のうち、「ベース」という端子に（0.7V以上の）電圧をかけて少量の電流が流れるような状態を作ると「コレクタ」という端子から「エミッタ」という端子へ電流が流れる状態になる

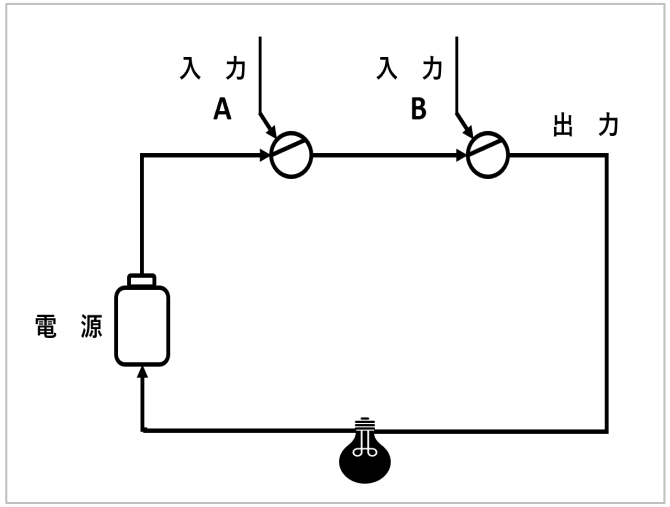
ベースに電気が流れるとスイッチON
ベースに電気が流れないとスイッチOFF



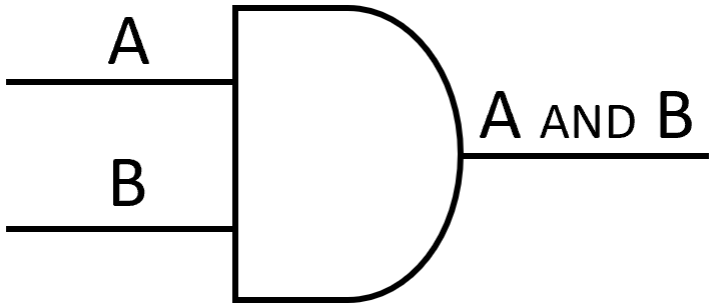
https://sagara-works.jp/research-and-development/electronics/transistor-basic/transistor-switch/

AND回路

- 入力信号Aと入力信号Bが共にやってきたときだけ、電流が流れ、電球が灯る
- 電流が流れるときを１，流れないときを０とすると、AとBの値が右図のように**共に１**になったときだけ、出力が１になる
- このような回路を**AND回路**と呼ぶ

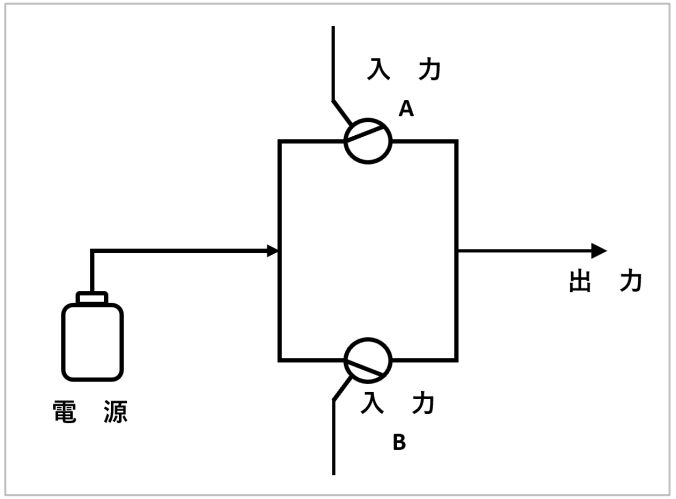


A	B	出力
1	1	1
1	0	0
0	1	0
0	0	0

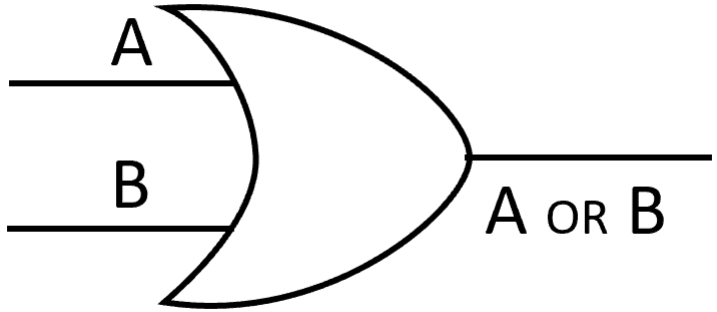


OR回路

- 入力信号Aと入力信号Bのいずれかがやってきたとき、電流が流れ、電球が灯る
- 電流が流れるときを１，流れないときを０とすると、AとBの値が右図のように**少なくとも一方が１**になったときだけ、出力が１になる
- このような回路を**OR回路**と呼ぶ

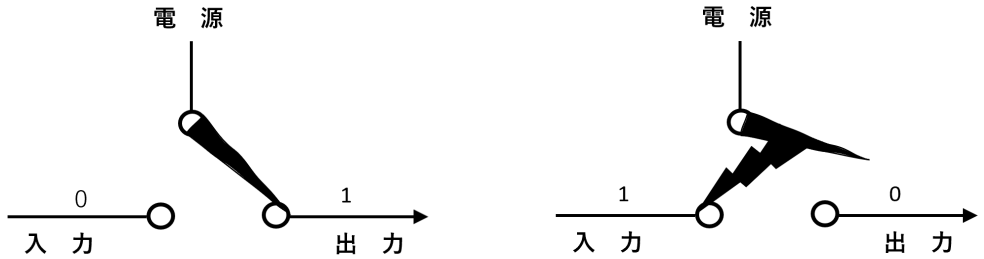


A	B	出力
1	1	1
1	0	1
0	1	1
0	0	0

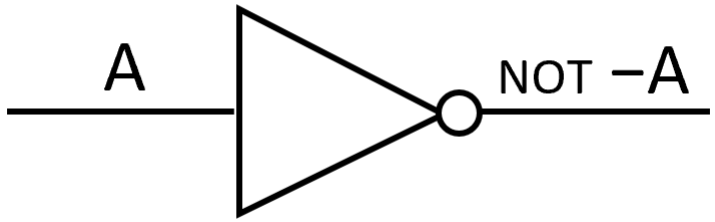


NOT回路（否定回路、反転回路）

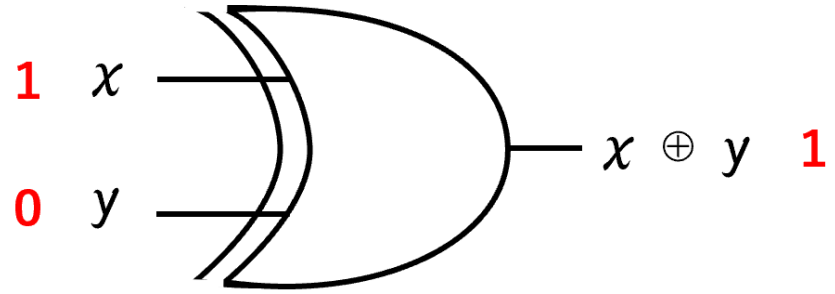
- 入力信号Aがやってくると出力に電流が流れず、入力信号Aがやってこないと出力に電流が流れ、電球が灯る
- 電流が流れるときを１，流れないときを０とすると、Aの値と出力が反対になる。このような回路をNOT回路（否定回路、反転回路）と呼ぶ

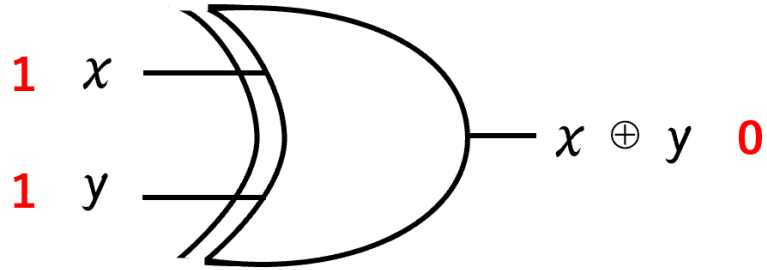


A	出力
1	0
0	1



XOR回路（排他的論理和回路）





排他的論理和の真理値表

入力		出力
x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

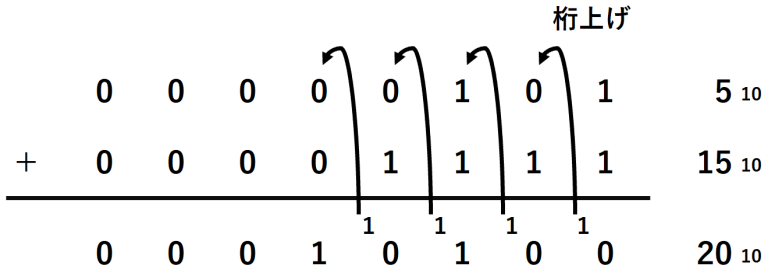
排他的論理和 ⊕ は,
入力の 1 の数が
　奇数のとき、出力が 1
　偶数のとき、出力が 0
となる関数

2 進数の加算

- ２進数の加算は基本的には 10 進数の加算と同様で、最下位の桁から順に同じ桁同士の加算を行い、桁あふれが起こった場合は、上位桁にあふれ分を加えることで加算の演算を行う。
- 5　$=8*0+4*1+2*0+1*1=(0101)_2$
- $15=8*1+4*1+2*1+1*1=(1111)_2$
- $20=(10100)_2$

16 8 4 2 1

8 4 2 1



2進数の加算

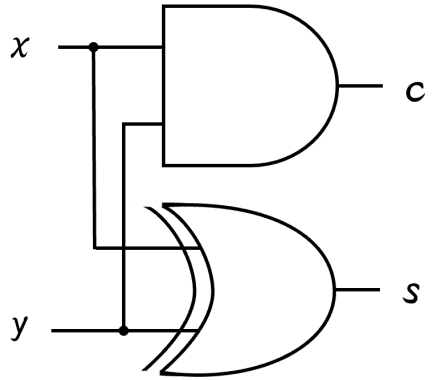
2進数	10進数
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

半加算器

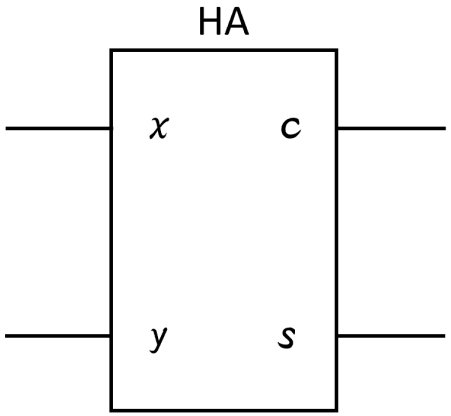
- 論理和OR（∨），論理積AND（·），否定NOT（¬），排他的論理和XOR（⊕）を組み合わせて、2 進数に対する加減算などの算術演算を実現することができる
- 例えば 1 桁の 2 進数 x と y の和を 2 ビット列 “c s”（2×c+s が演算結果）と表すとすると、c （上位への桁上げ）や s （その桁の和）の値は右上の表のように表される
  - c の値が 1 になるのは x も y も共に 1 のときに限るので、c = x·y と表すことができる
  - 同様に、s の値が 1 になるのは x と y のいずれか一方のみが 1 のときに限るので s = x⊕y と表すことができる
- このように，下位ビットからの桁上げを考慮しない1ビットの加算器を **半加算器**（half adder: HA）と呼ぶ

半加算器の真理値表

x	y	c	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0



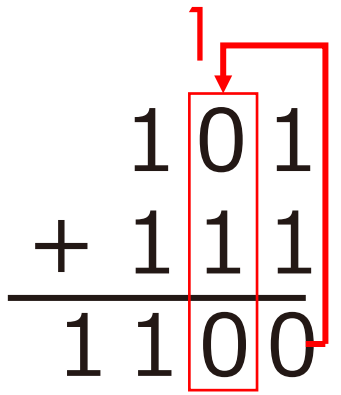
論理ゲートを用いた半加算器の実現



半加算器の論理記号

全加算器

- **下位桁からの桁上げを考慮**した場合の 1 ビットの加算器を 全加算器（full adder: FA）と呼ぶ。下位桁からの桁上げを変数 z で表し、半加算器と同様に 1 ビットの 2 進数 x と y および下位桁からの桁上げ z との和を 2c + s で表すと、c や s の値は右下の表のようになる。
- c の値が 1 になるのは、x, y, z の値のうちの 2 つ以上が 1 のときに限るので
  $c = (x \cdot y) \vee (y \cdot z) \vee (z \cdot x)$
  と論理式で表すことができる。
- 同様に s の値が 1 になるのは、x と y と z の値のうちの奇数個が 1 であるときに限るので
  $s = (\neg x \cdot \neg y \cdot z) \vee (\neg x \cdot y \cdot \neg z) \vee (\neg x \cdot \neg y \cdot z) \vee (x \cdot y \cdot z)$
  $= x \oplus y \oplus z$
  と論理式で表すことができる。

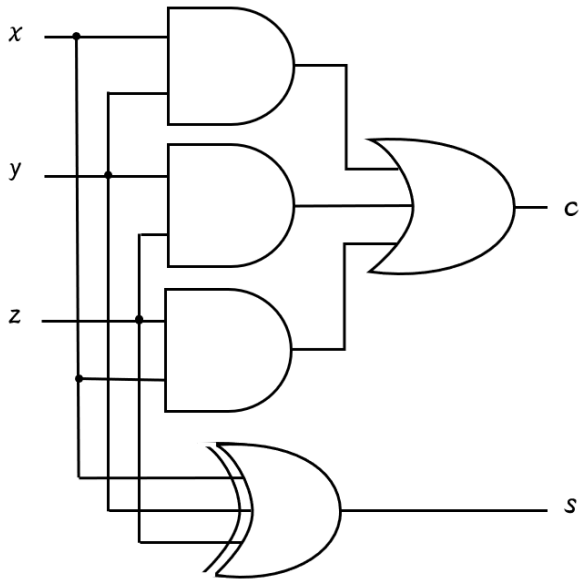


全加算器の真理値表

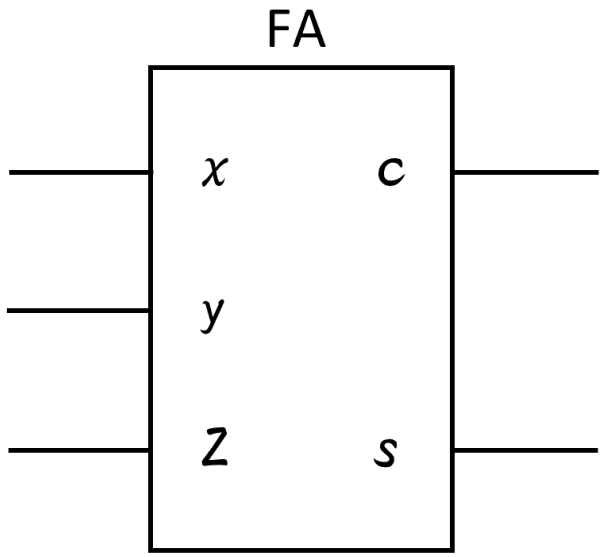
x	y	z	c	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

全加算器

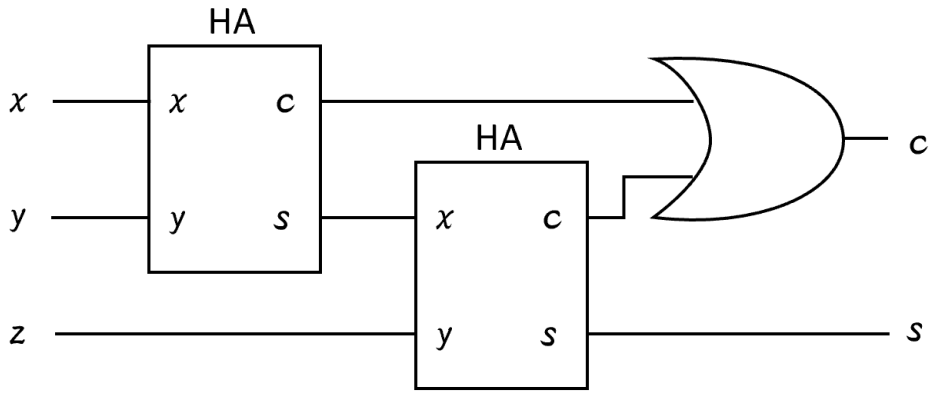
- 半加算器と同様、全加算器も論理演算記号を用いて左下の図のように実現でき、全加算器そのものを，下中央の図のような論理演算記号を用いて表現することもある。
- さらに，全加算器は半加算器を用いて，右下の図のように作ることもできる



論理ゲートを用いた全加算器の実現



全加算器の論理記号

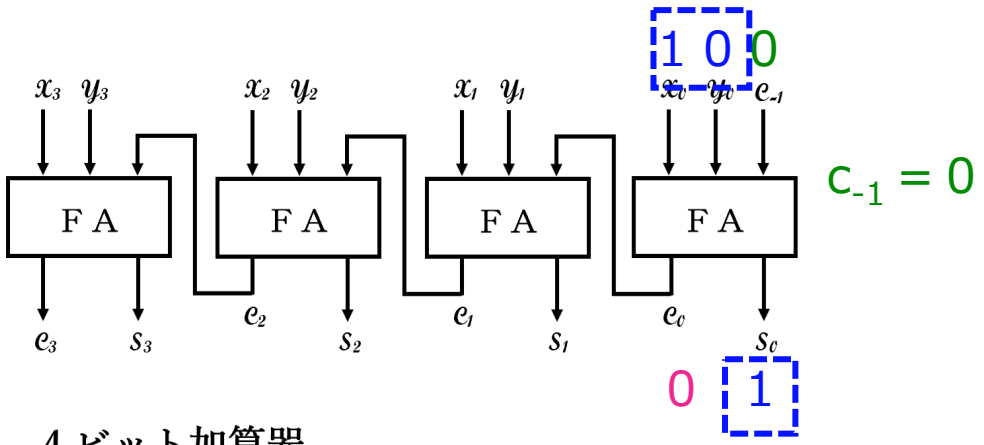


半加算器と論理ゲートを用いた全加算器の実現

４ビットの全加算器

- 全加算器は下位桁からの桁上げを考慮して加算を行うことができるので、全加算器を図のように接続すると、複数ビット（bit）で表現されている 2 進数の加算演算を実現することができる。

$(x_3,x_2,x_1,x_0)=(1,0,1,1)$ (11)
$(y_3,y_2,y_1,y_0)=(1,0,1,0)$ (10)



４ビット加算器

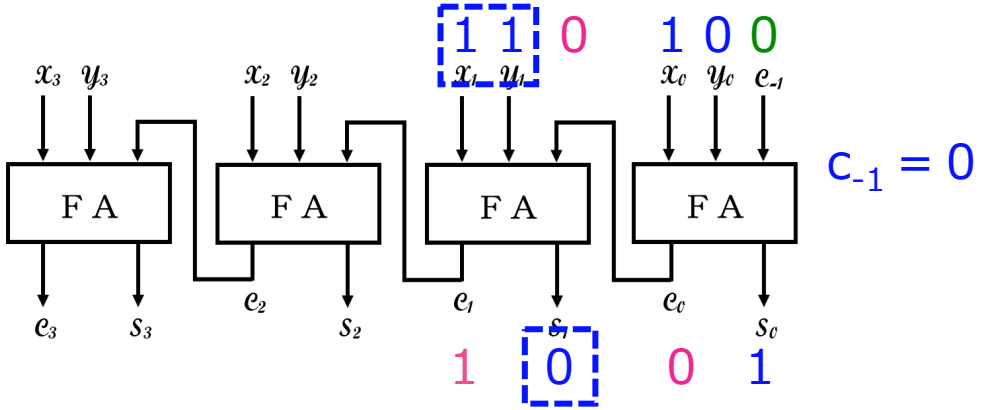
```
    1 0 1 1   (11)
+   1 0 1 0   (10)
  1 0 1 0 1   (21)
```

c_3 $s_3s_2s_1s_0$

４ビットの全加算器

- 全加算器は下位桁からの桁上げを考慮して加算を行うことができるので、全加算器を図のように接続すると、複数ビット（bit）で表現されている 2 進数の加算演算を実現することができる。

$(x_3,x_2,x_1,x_0)=(1,0,1,1)$ (11)
$(y_3,y_2,y_1,y_0)=(1,0,1,0)$ (10)



４ビット加算器

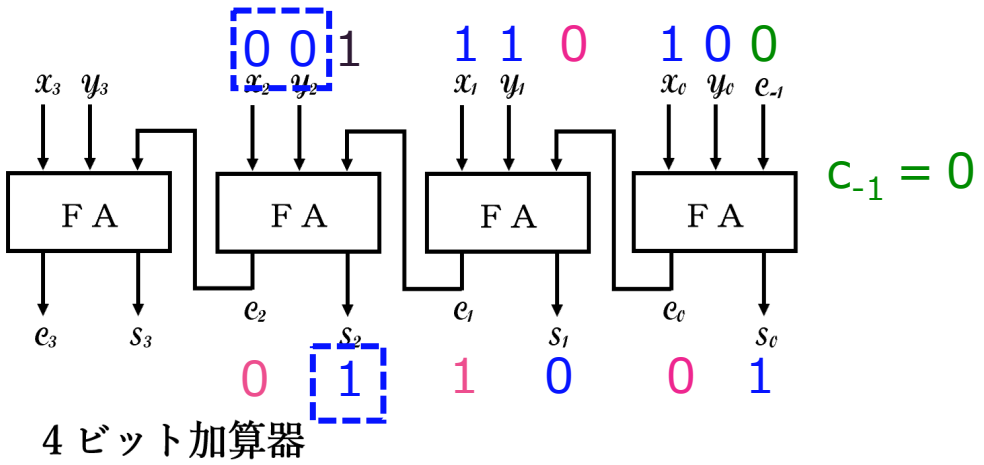
```
    1 0 1 1    (11)
+   1 0 1 0    (10)
  1 0 1 0 1    (21)
```

c_3 $s_3s_2s_1s_0$

４ビットの全加算器

- 全加算器は下位桁からの桁上げを考慮して加算を行うことができるので、全加算器を図のように接続すると、複数ビット（bit）で表現されている 2 進数の加算演算を実現することができる。

$(x_3,x_2,x_1,x_0)=(1,0,1,1)$ (11)
$(y_3,y_2,y_1,y_0)=(1,0,1,0)$ (10)



$c_{-1} = 0$

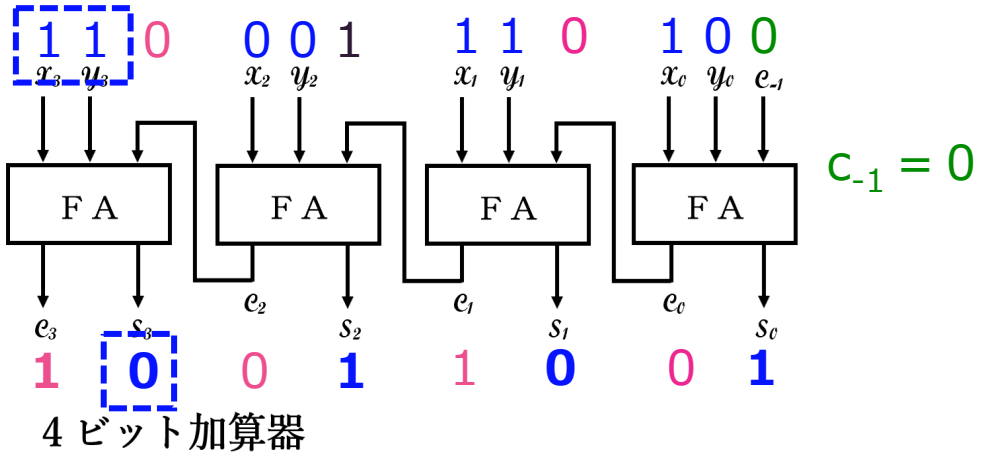
```
      1
    1 0 1 1   (11)
+   1 0 1 0   (10)
  1 0 1 0 1   (21)
```

c_3 $s_3s_2s_1s_0$

４ビットの全加算器

- 全加算器は下位桁からの桁上げを考慮して加算を行うことができるので、全加算器を図のように接続すると、複数ビット（bit）で表現されている 2 進数の加算演算を実現することができる。

$(x_3,x_2,x_1,x_0)=(1,0,1,1)$ (11)
$(y_3,y_2,y_1,y_0)=(1,0,1,0)$ (10)



$c_{-1} = 0$

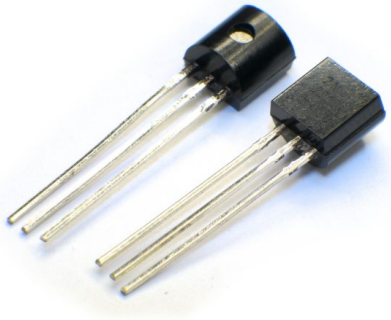
```
    1 0 1 1   (11)
+   1 0 1 0   (10)
  1 0 1 0 1   (21)
```

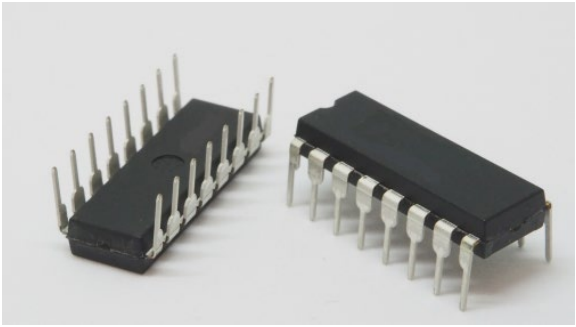
c_3 $s_3s_2s_1s_0$

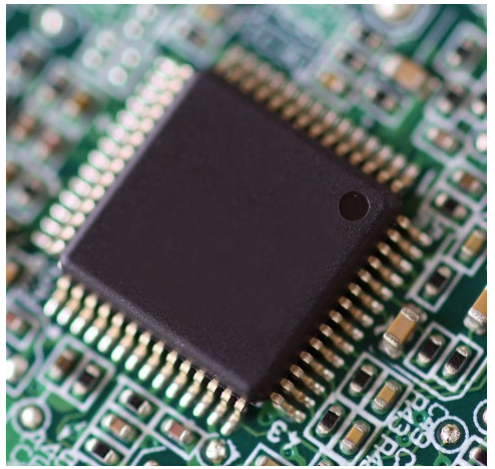
→ こうした話が「コンピュータの世界は0と1」のベース

トランジスタやIC,LSIの出現

- 増幅器として開発された「トランジスタ」がスイッチの役割を果たし、大きさ、価格ともに信じられないほど小さくなった
  - トランジスタ
  - IC（集積回路）
  - LSI（大規模集積回路）
    - トランジスタなど1000素子以上
  - VLSI（超LSI）
    - 10万素子以上

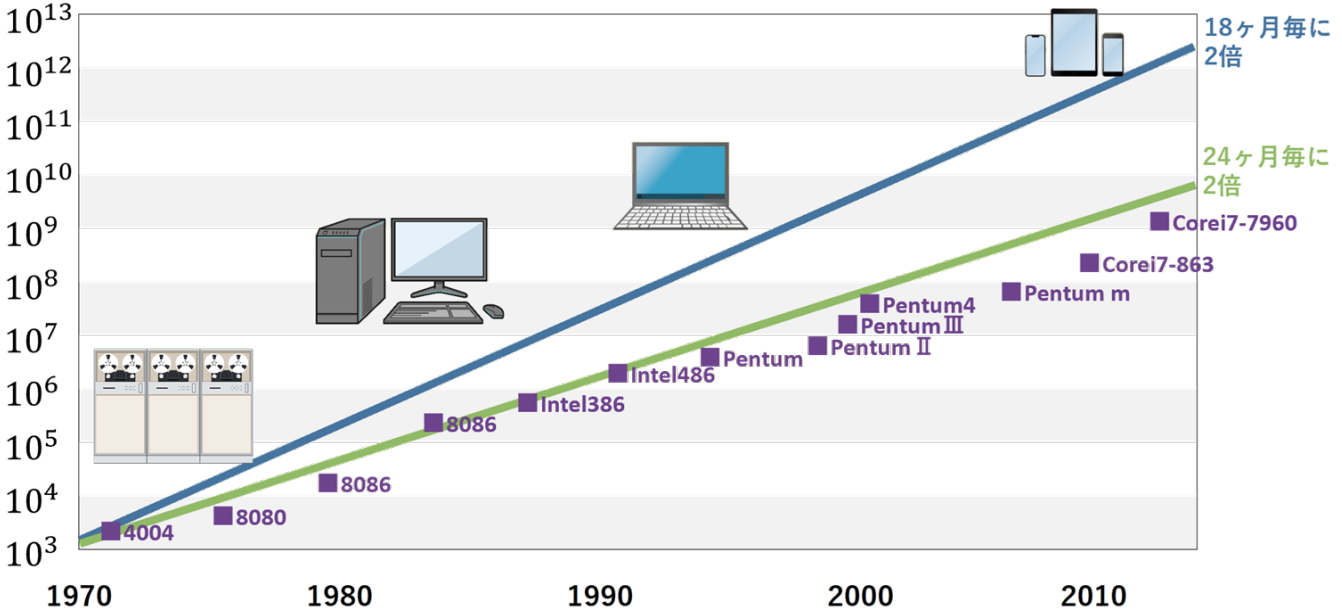






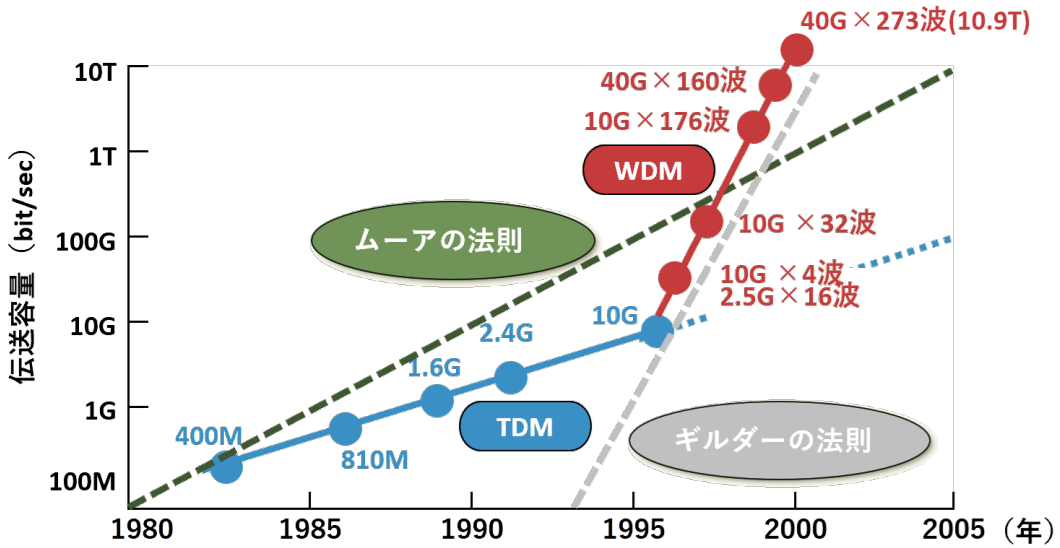
ムーアの法則

- 世界最大の半導体メーカーIntel社の創設者の一人Gordon Moore博士が1965年に提唱した「半導体の集積密度は18～24ヶ月で倍増する」という経験則（18ヶ月で2倍なら10年で約100倍高速になる）
  - **回路が1/kに小さくなると、動作速度がk倍速くなる**



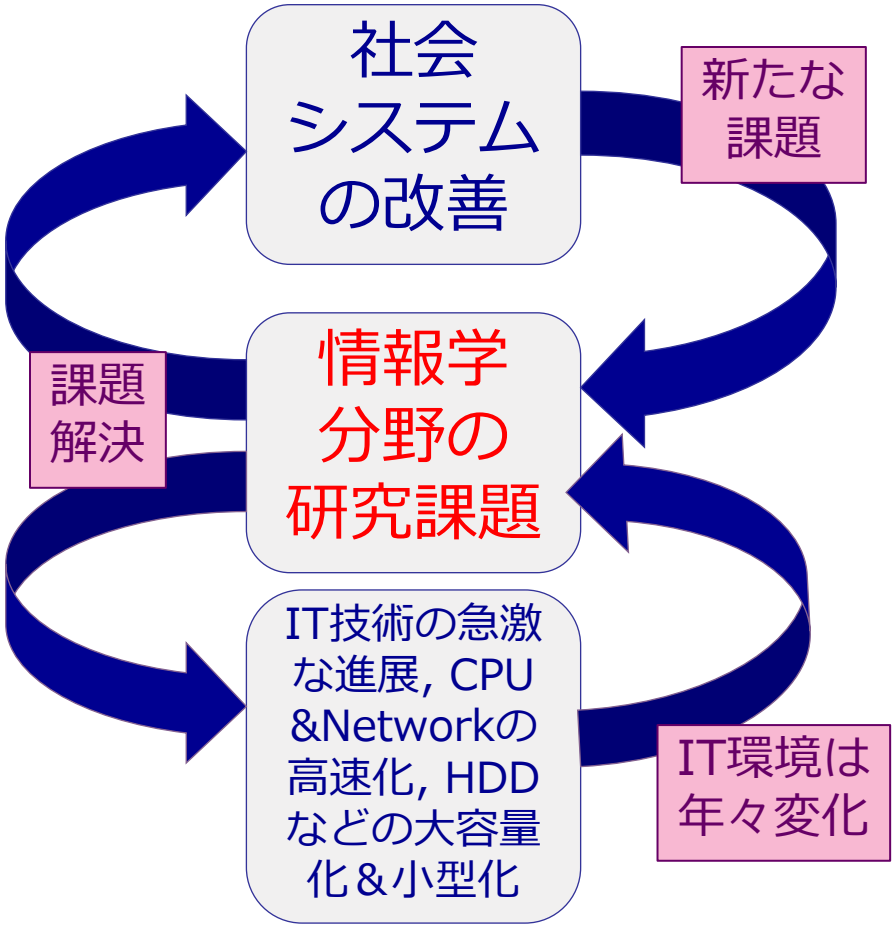
ギルダーの法則

- アメリカの経済学者ジョージ・ギルダー（George Gilder）が2000年に自著「テレコズム」にて提唱した「通信網の帯域幅は6ヶ月で2倍になる」という**ネットワークのスピードアップ**に関する法則
  - ギルダーの法則に従うと10年で100万倍のペースとなるが、実際は10年で1000倍くらい（1年で2倍のペース）で、イーサネットは、ムーアの法則（10年で100倍）程度のペースでしか速くなっていない。



IT技術の急激な進展と情報学分野における研究課題の変遷

- 数年～10年のオーダーでCPUのスピードやハードディスクの記憶容量、ネットワークの速度が100倍になっている。IT機器のサイズは小さく安くなっている。
- 30年で100万倍程度コストパーフォーマンスが向上するので、扱う問題が日々変化している。
- IT技術の進展に伴い、新しい情報学分野の研究課題が生じ、研究成果が社会システムの改善に寄与し、社会システムが変化している。
- 単一のテーマを追い求めた研究開発から、最近はベースとなるIT技術を発展させ、新たな社会課題解決につながる研究開発へ
（of ICT から by ICT へ）



03

文字・画像・音声のデジタル化

ビット数と表現可能な情報の数

- 例えば４つの状態を 0 と 1 だけを使って表現するなら 2 bit あれば可能である。４つの記号 ♥ ♣ ♠ ♦ を 0 と 1 だけを使って表現するには、以下のような対応表を作ればよい。

♥	♣	♠	♦
00	**01**	**10**	**11**

- ８つの状態を 0 と 1 だけを使って表現するなら、 3 bit あれば可能である。例えば、明日の天気を表すのに以下のような対応をさせてみる。

快晴	晴れ	曇り	雨	雷雨	雪	大雪	台風
000	**001**	**010**	**011**	**100**	**101**	**110**	**111**

- アルファベットは26文字あるので、26状態を表現するには 5 bit あればよい。

A	B	C	…	Z
00000	**00001**	**00010**	…	**11001**

bit 数と情報の数

bit 数	表現可能な情報の数
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256

デジタル化は世界中が同じルールを使用しないと意味がない

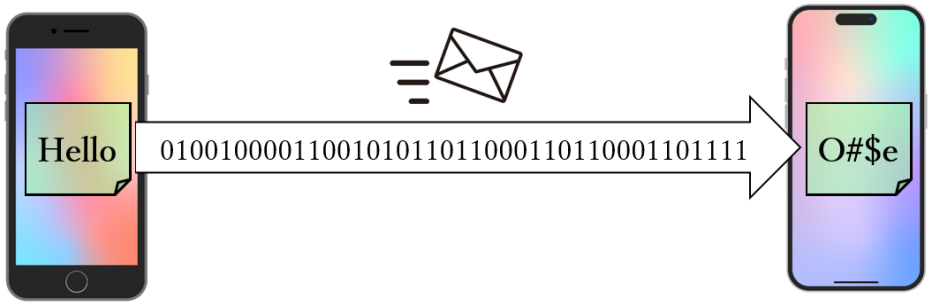
- Hello の 5 文字を世界共通のルールでデジタル化すると以下の 40bit の情報

  0100100001100101011011000110110001101111

- これならコンピューターで扱えるし、通信もできる。

H	e	l	o
01001000	01100101	01101100	01101111

- 受信した相手がこんな一見無意味に思える 40bit の01列を正しい元 の５文字に戻すためには、お互いに共通の変換ルールが必要になる



ASCIIコード（世界共通の英数文字表現ルール）

- **英数文字を表現するのに必要なビット数**
  - アルファベット大文字・・・・　26種類
  - アルファベット小文字・・・・　26種類
  - 数字・・・・・・・・・・・・・　10種類
  - よく使う記号・・・・・・・・・　30種類程度
  - 合計・・・・・・・・・・・・・　92種類程度
- **92 種類の状態を表すのにいったい何 bit 必要であろうか？**
  - 2 の n 乗を計算してみる
  - $2^4 = 16$　$2^5 = 32$　$2^6 = 64$　$2^7 = 128$　$2^8 = 256$
  - 7 bit あれば表現できることになる。7 bit は奇数で扱いにくいため、もう 1 bit 加えて 8 bit で 1 文字を表すことにした　**→ 8 bit = 1 byte = 英数字1文字**

ASCIIコード
（世界共通の英数文字表現ルール）

1 byte＝8 bit ＝ 英数字１文字

ASCIIコード表

スペース	00100000	0	00110000	@	01000000	P	01010000	`	01100000	p	01110000
!	00100001	1	00110001	A	01000001	Q	01010001	a	01100001	q	01110001
"	00100010	2	00110010	B	01000010	R	01010010	b	01100010	r	01110010
#	00100011	3	00110011	C	01000011	S	01010011	c	01100011	s	01110011
$	00100100	4	00110100	D	01000100	T	01010100	d	01100100	t	01110100
%	00100101	5	00110101	E	01000101	U	01010101	e	01100101	u	01110101
&	00100110	6	00110110	F	01000110	V	01010110	f	01100110	v	01110110
'	00100111	7	00110111	G	01000111	W	01010111	g	01100111	w	01110111
(	00101000	8	00111000	H	01001000	X	01011000	h	01101000	x	01111000
)	00101001	9	00111001	I	01001001	Y	01011001	i	01101001	y	01111001
*	00101010	:	00111010	J	01001010	Z	01011010	j	01101010	z	01111010
+	00101011	;	00111011	K	01001011	[	01011011	k	01101011	{	01111011
,	00101100	<	00111100	L	01001100	¥	01011100	l	01101100	|	01111100
-	00101101	=	00111101	M	01001101	]	01011101	m	01101101	}	01111101
.	00101110	>	00111110	N	01001110	^	01011110	n	01101110	~	01111110
/	00101111	?	00111111	O	01001111	_	01011111	o	01101111		

漢字や各国の文字のコード化

漢字の文字コード化

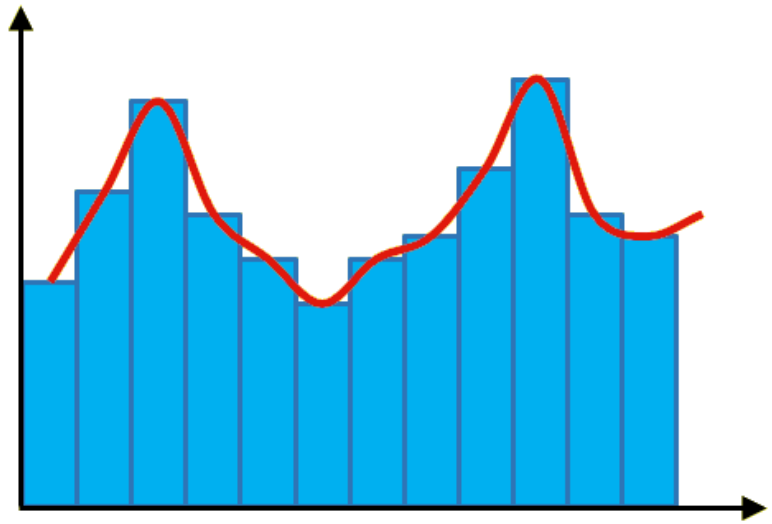
- 漢字は個数が多いので２バイト（16ビット）で１文字を表現
- **JISコード**：日本語用の文字コード
  - 英数字などの１バイト文字と漢字の２バイト文字を混在させている。制御文字で１バイト文字と２バイト文字を切り替え
- **シフトJISコード**：日本語用の文字コード
  - 制御文字なしで１バイト文字と２バイト文字を混在させる日本語用文字コード
- 文字は漢字10,000文字（原稿用紙25枚）でも、20,000バイト（20Kバイト、160Kビット＝160,000ビット）

世界各国の文字のコード化

- **Unicode**：世界共通の文字コード（1文字あたり1～6バイト：文字種によって異なる）

音楽情報のデジタル化

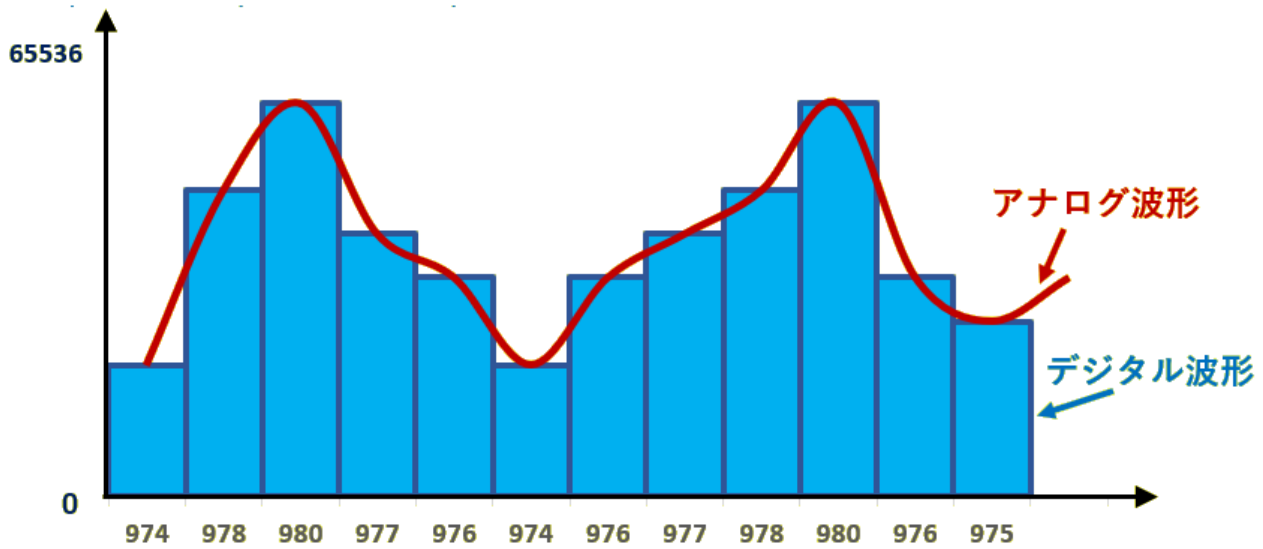
- 音楽情報の周波数
  - 幾つかのsin波やcos波で合成
  - 音楽CDの場合、最大22,050Hz
- **サンプリング定理**
  - アナログ信号のデジタル化
  - 最大周波数の２倍の個数(サンプル数)の棒グラフでサンプリング →原音にほぼ忠実に再生可能
  - CDや電話の音声のデジタル化



音楽情報のデジタル化

- 最大周波数　22,050Hz（サンプル数：44,100個）
- 音の強さを16ビット（65,536段階）で表現
- 16ビット×44,100個＝705,600ビット/秒 (705.6Kビット/秒)

1 秒に原稿用紙100枚分以上の文字の伝送と同じ



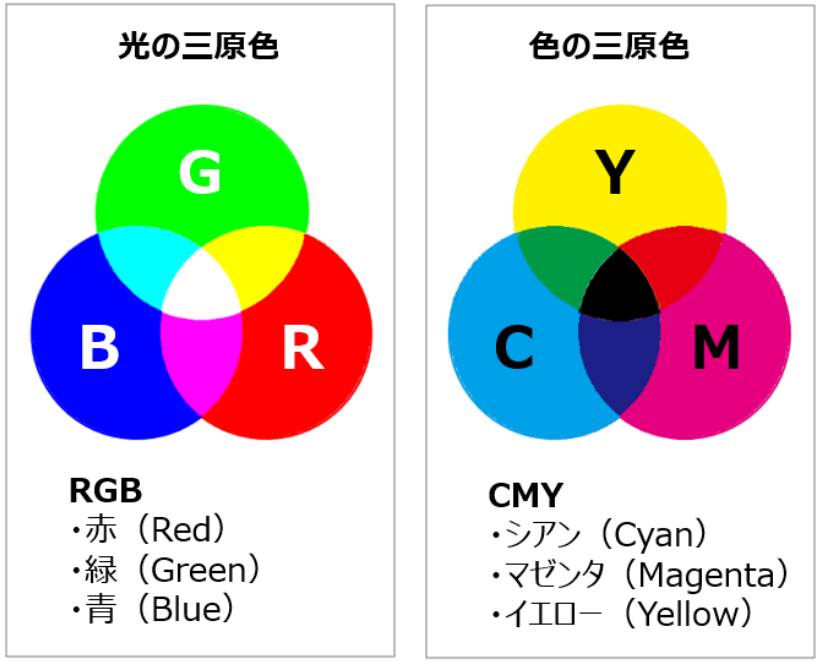
デジカメの写真のデジタル化

写真のデジタル化

- デジタルカメラで撮った写真をどんどん拡大していくと、小さな点が見える。 デジタルデータの画像は画素と呼ばれる小さな点の集合体で、 0 と 1 で表されるデータとして表現
- 実際の絵や風景などをデジタル画像では数値で表している

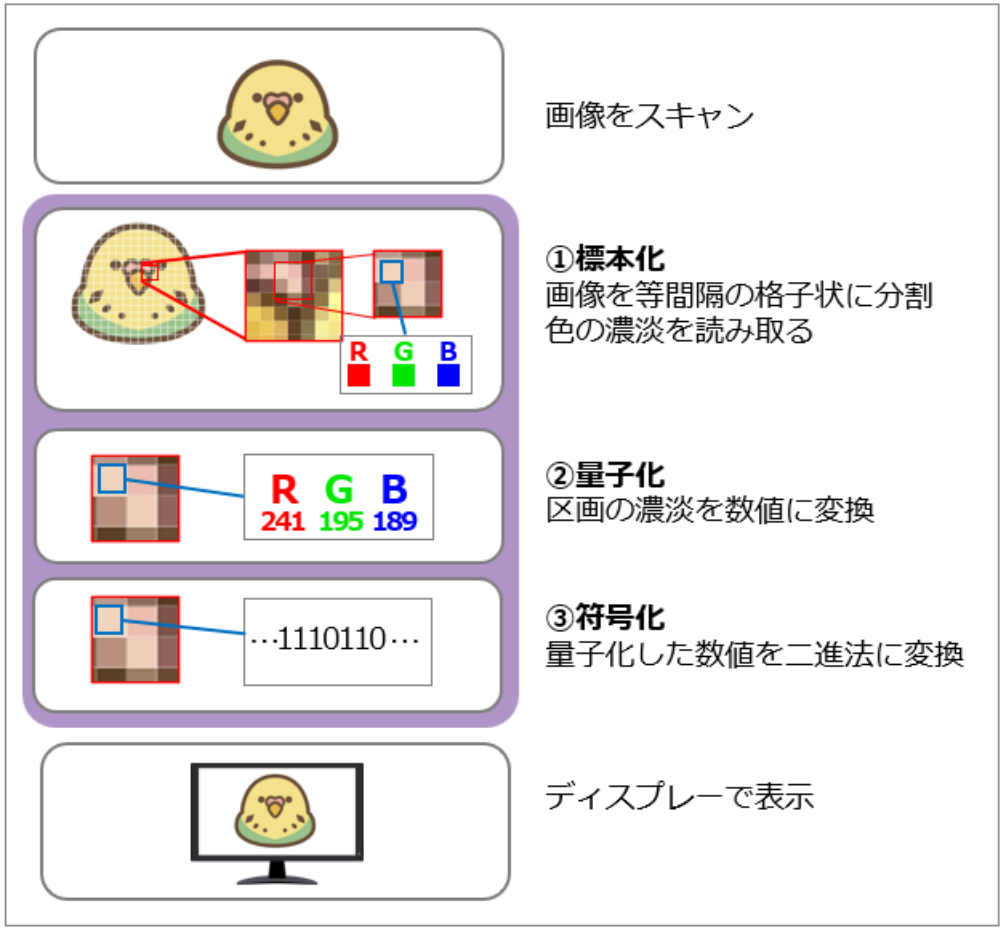
光の三原色と色の三原色

- ディスプレイ上にある一つ一つの画素は光の三原色（赤、緑、青）の値によって表される（光の三原色は加法混色と呼ばれる）
- 画像をプリントアウトする際は、色の三原色（シアン（青）、マゼンダ（赤）、イエロー（黄））を用いた減法混色で表現される



画像のデジタル化

- 画像のデジタル化はまず画像を等間隔の格子状に分割（標本化）して色の濃淡を読み取る
- 次に各区画の濃淡を数値に変換（量子化）
- 最後に量子化した数値を２進法に変換（符号化）して、スキャンした画像をディスプレイで表示できるようにする

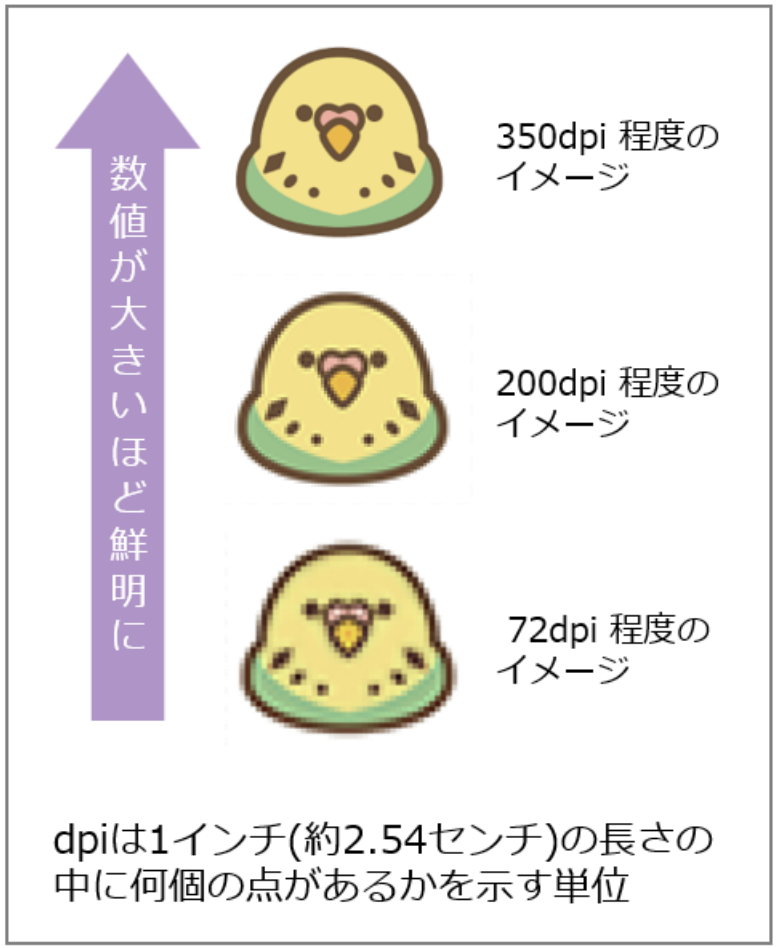


画像のデジタル化

画像のデジタル化

解像度：dpi (dots per inch)

- 画像を画面に表示したり、紙に印刷したりする時に画素の密度を表す値
- 解像度が高いほど滑らかに、低いほど粗い画像になる
- **dpi**は１インチ（約2.54センチ）の長さに何個の点（ピクセル）があるかを表す単位
  - dpiの値が大きいほど解像度が高い



解像度が変化すると

高画質の画像のサイズ

- RGB３色256階調（約1,600万色表現可能)
  - 赤、緑、青それぞれ１バイト必要（１ピクセル３バイト）
- PCの画面：1,920×1080ピクセル表示（2,073,600ピクセル）

→ 2,073,600×3 ＝6,220,800バイト

≒6.2Mバイト必要

画像の圧縮の手法
ーエントロピー符号化（その１）ー

例1：**ランレングス符号化**

- ビット列としての符号
  213800000000000365381111111111456456000000000000000023
  → 2138A01136538A110456456A01423
  目印 A と記号 x と繰り返し回数 yy → Axyy
- 音声の無音部分は０の連続系列
- 空や壁の色は同じ色の連続の可能性が高い

画像の圧縮の手法
ーエントロピー符号化（その２）ー

例２：**統計的符号化**

- モールス信号

  E　・　　使用頻度が高い場合短く

  Q　ーー・ー　　使用頻度が低い場合長い

- よく使う色を短い符号に割り当て，あまり利用しない色を長い符号に割り当てる

  赤　赤　赤　赤　黄　緑　黄　赤　赤

  00　00　00　00　01　10　01　00　00

  → 18ビット

  0　0　0　0　10　11　10　0　0

  → 12ビット

画像の圧縮の手法
ーエントロピー符号化（その３）ー

例３： **CLUT (Color Look Up Table) 符号化**

- RGB３色256階調(1,600万色)で表現
  → 各ピクセル毎に24ビット必要
  n ピクセル　→　 24n ビット
- もし1,600万色のうち64色程度しか使わないなら
  → 64×24　どの色を使うかの表（ CLUT ）
  n ピクセル　→　 6n ビット
  合計： 64×24＋ 6n ビット　<< 24n ビット
- 2m 色使う場合
  → 2m× 24＋ m×n ビット <> 24n ビット

画像の圧縮の手法-II
―情報源符号化―

- 例１：**差分符号化**
  - 70,68,69,71,73,72,70,...
    - 音声の大きさが小幅で変動する場合
  - 70,-2,+1,+2,+2,-1,-2,...のように表現
    - 音声の大きさを表すのに必要な情報量を減らせる
  - 連続したデータ間に大きな違いがない場合に有効
  - データの内容がランダムに変化する場合あまり有効でない
- その他の圧縮技術（詳細は省略）
  - 離散余弦変換
  - ベクトル量子化

静止画像圧縮技術：JPEG

JPEG (Joint Photographic Experts Group)

- 前述のような圧縮技術を組み合わせた静止画像圧縮技術
- デジタルカメラなどに広く用いられている画像圧縮の一手法
- JPEGの圧縮技術を用いると、1/20程度に静止画像を圧縮することも可能

まとめ

1. コンピュータの歴史
2. コンピュータと2進数
   - コンピュータの仕組みの基本：トランジスタ，IC，LSIとコンンピュータ
   - コンピュータの性能の急速な進歩（ムーアの法則）
3. 文字・画像・音声のデジタル化
   - ビット列（0と1）による情報の表現（文字・音声・画像）
   - 情報の圧縮技術