集中講義令和8年2月24日～26日

コンピュータシステム入門

Day 1: デジタル回路の基礎 (1)

講師　陳　オリビア　（大学院システム情報科学研究院）

TA　GPT-5 Thinking (OPEN AI)

Gemini 2.5 pro (Google)

CONTENTS

今日のスケジュール

	時間帯	モジュール	タイプ
午前中	10:00 ~ 11:30	イントロダクションとコンピュータの基本構成	講義
	11:30 ~ 11:45	Coffee Break	
	11:45 ~ 12:30	コンピューターの分解	Demo
	12:30 ~ 13:30	Lunch Break	
午後	13:30 ~ 14:30	情報のデジタル表現	講義
	14:30 ~ 15:00	「デジタルの自分」を分解してみよう	演習
	15:00 ~ 15:15	Coffee Break	
	15:15 ~ 16:30	デジタル回路の基礎（１）	講義
	16:30 ~ 17:00	論理ゲートで遊ぼう	演習

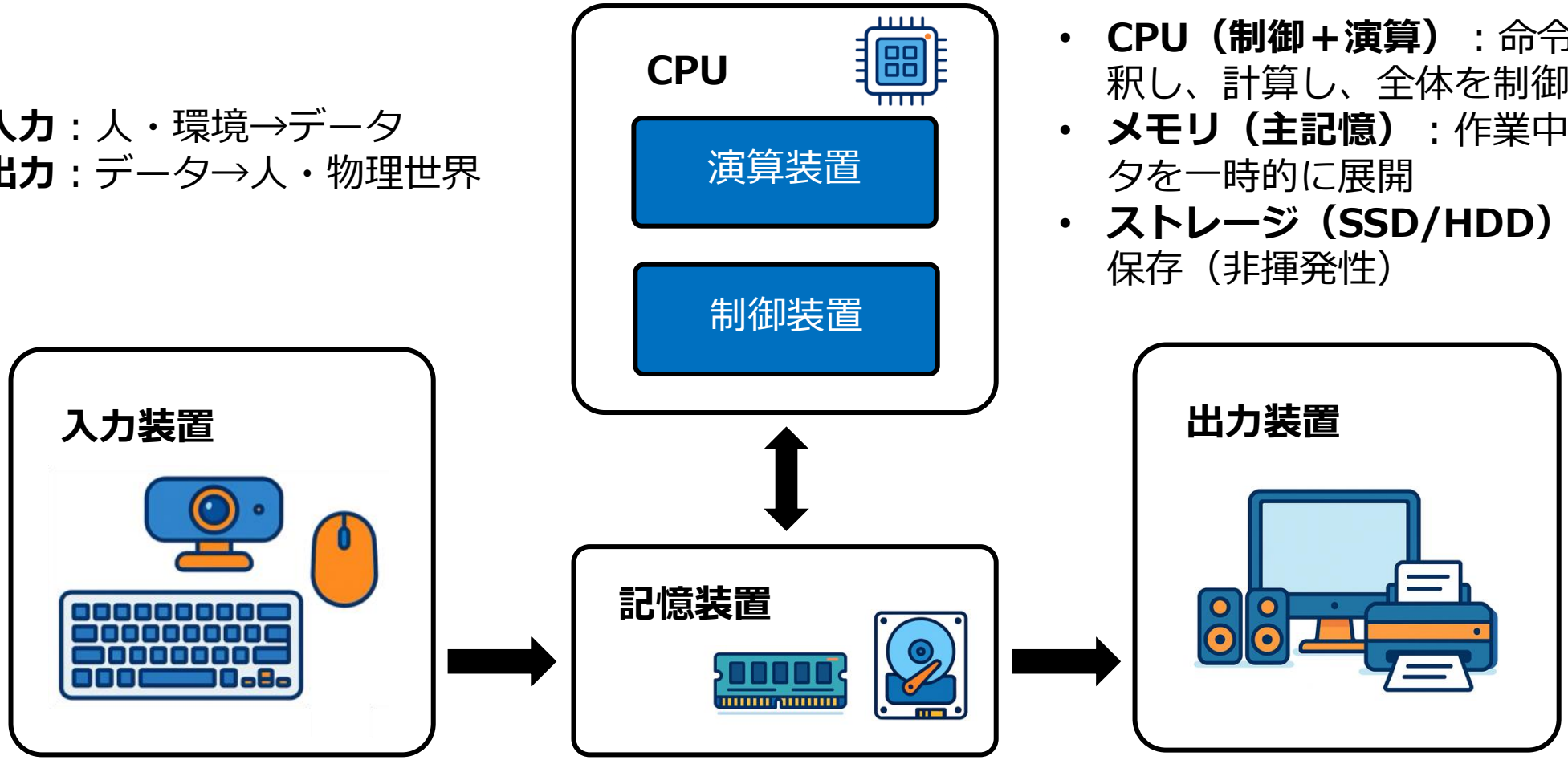
CONTENTS

モジュール３：デジタル回路の基礎（１）

CONTENTS

前回の復習：コンピュータの５大装置

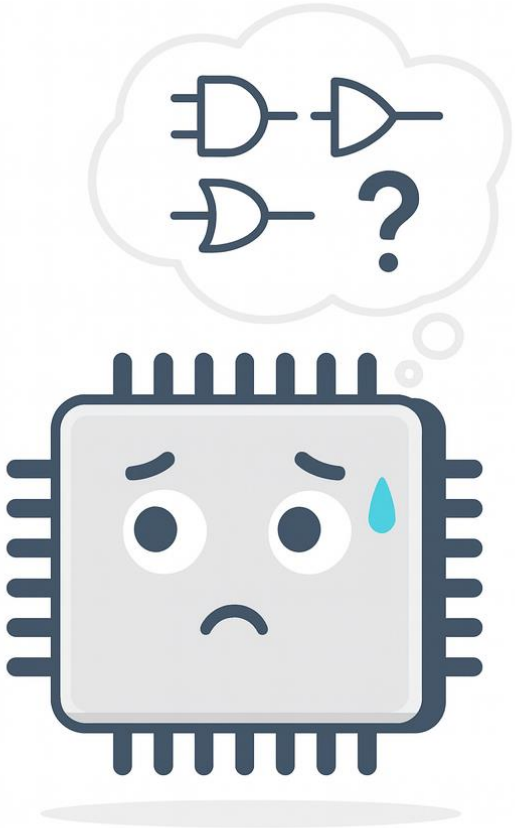
- **入力**：人・環境→データ
- **出力**：データ→人・物理世界



- **CPU（制御＋演算）**：命令を解釈し、計算し、全体を制御
- **メモリ（主記憶）**：作業中のデータを一時的に展開
- **ストレージ（SSD/HDD）**：長期保存（非揮発性）

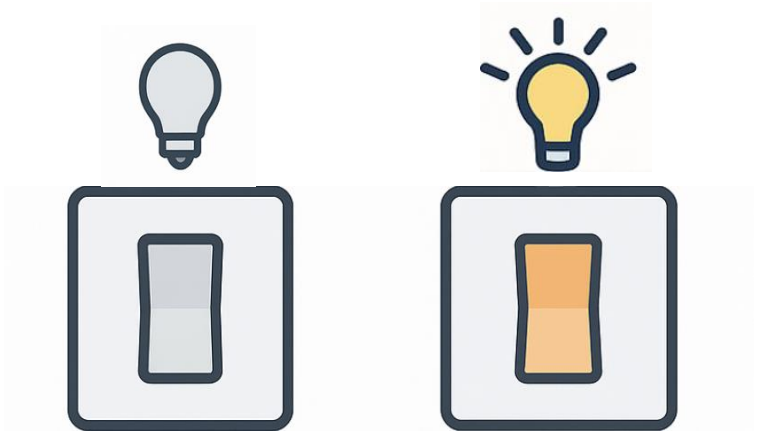
核心的な問い

```
 0101
+001 =
------
    ?
```

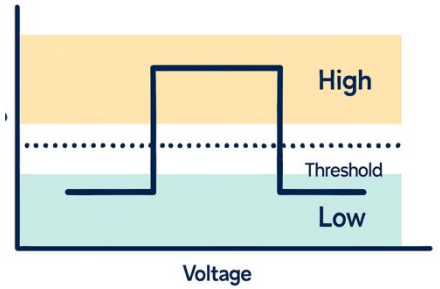


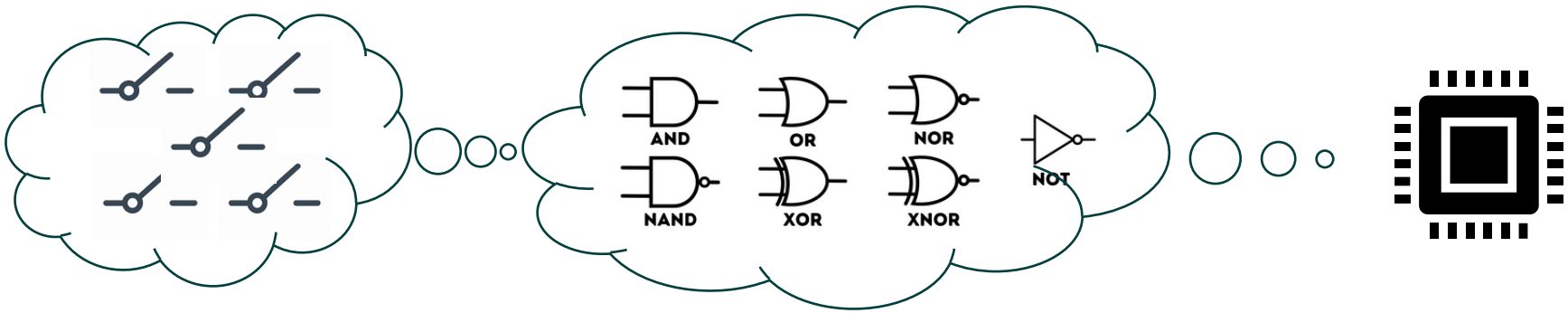
再び「スイッチ」の話へ

デジタルは **2値**：ON(1) / OFF(0)



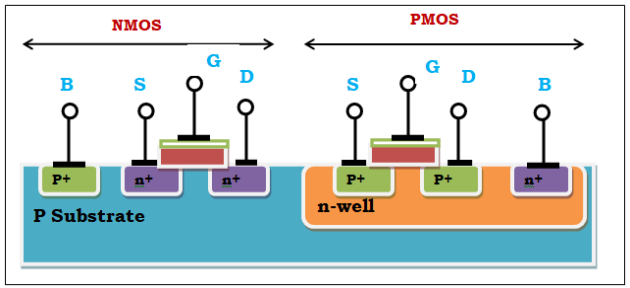
電気的には **High / Low の電圧**
（しきい値より上=1、下=0）





スイッチ　論理（デジタル）回路　CPU

CPU のスイッチはどう進化してきた？



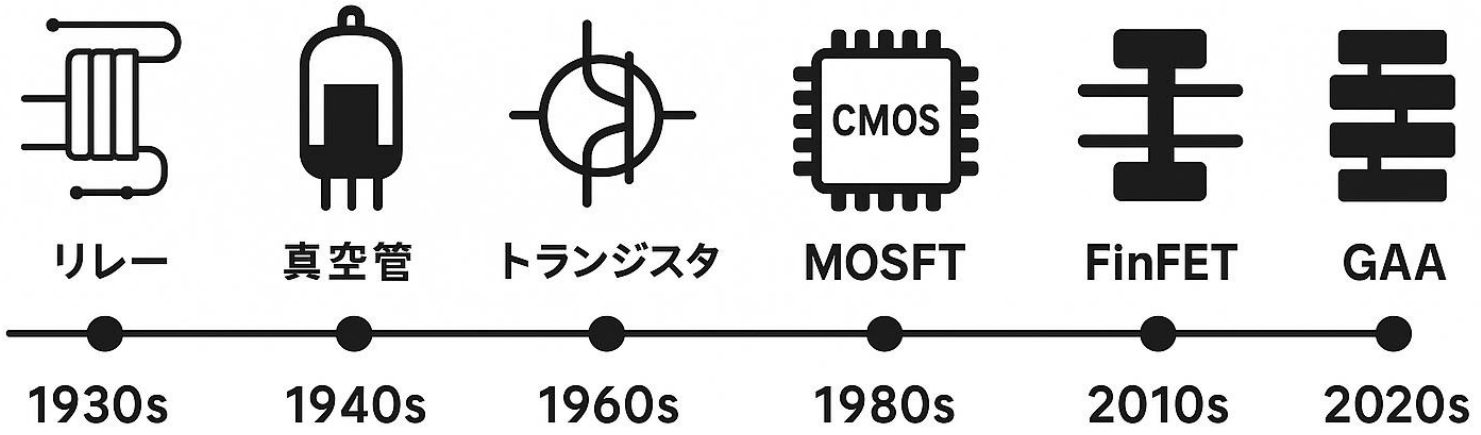
サイズ⇩

消費電力⇩

速度⇧

安定性⇧

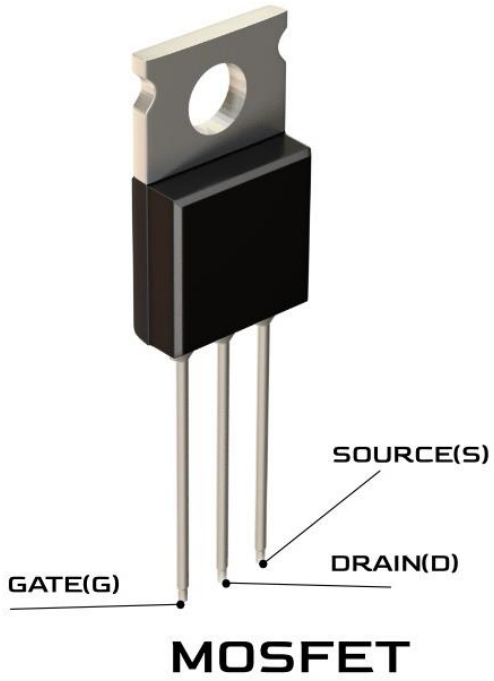
集積度⇧

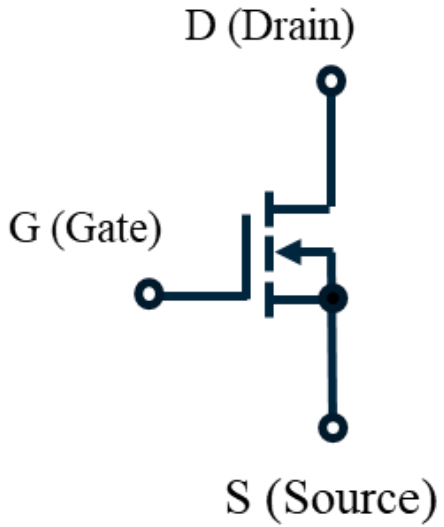


スイッチの進化

トランジスタ = コンピュータを支える「電子スイッチ」

- トランジスタは複数種類ある（Biploar Transistors, MOSFETs, IGBTs）
- 集積回路（CPU/メモリ）の主役は MOSFET（CMOS）

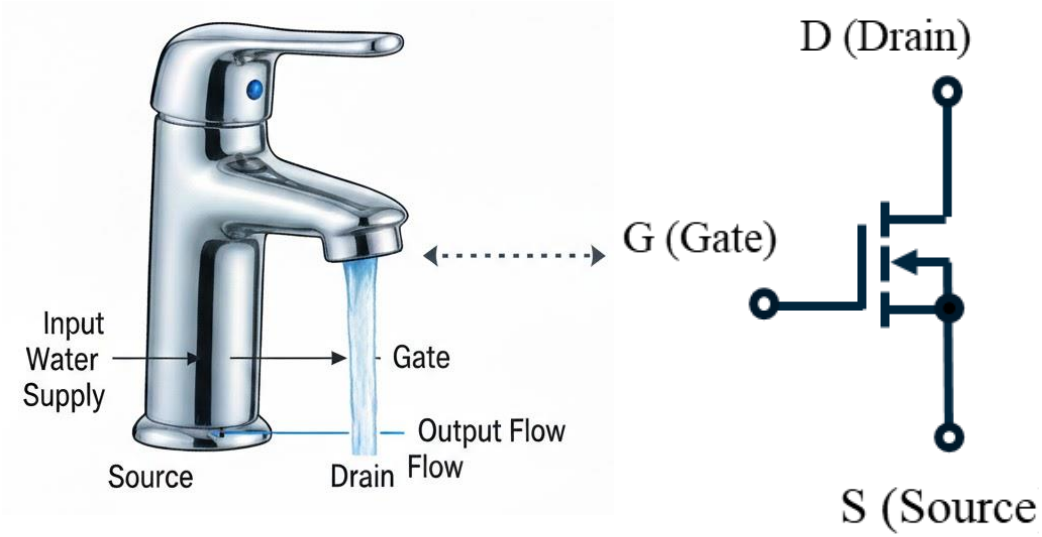




- 電圧が電流を制御する：ゲート電圧で「流す／止める」が決まる

トランジスタ = コンピュータを支える「電子スイッチ」

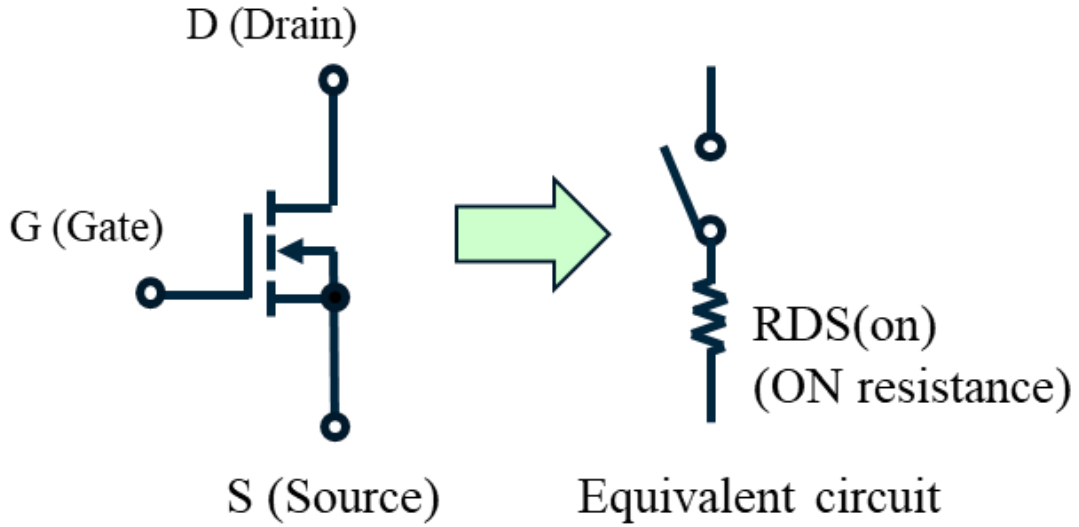
- トランジスタは複数種類ある（Biploar Transistors, MOSFETs, IGBTs）
- 集積回路（CPU/メモリ）の主役は MOSFET（CMOS）



- 電圧が電流を制御する：ゲート電圧で「流す／止める」が決まる

トランジスタ = コンピュータを支える「電子スイッチ」

- **トランジスタは複数種類ある（Biploar Transistors, MOSFETs, IGBTs）**
- **集積回路（CPU/メモリ）の主役は MOSFET（CMOS）**



- 電圧が電流を制御する：ゲート電圧で「流す／止める」が決まる
- ON / OFF を 0 / 1 とみなす：これがデジタル回路の基本
- たくさん組み合わせると論理になる：論理回路・メモリ・CPU/GPUへ

ムーアの法則

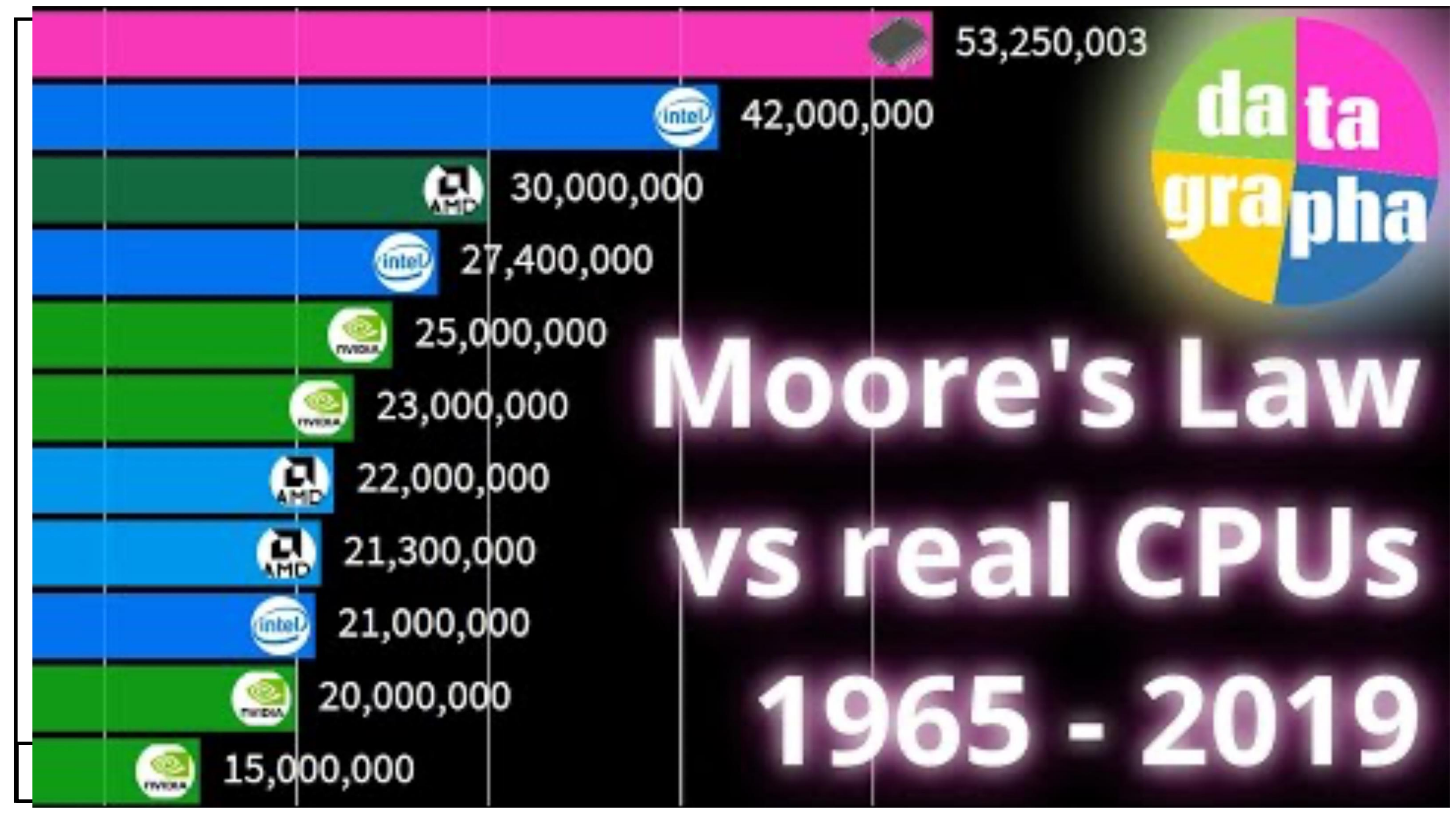
Gordon Moore
(1929-2023)
Intel Co-founder

ムーアの法則

- **1～2年ごとに**、同程度のコストで**チップ上のトランジスタ数が約2倍**になるという経験則
- 結果：**計算あたりのコストが下がり**、小型化・省電力・高機能化が進む
- 数式イメージ：$N(t) = N_0 \cdot 2^{t/T}$　（$T \approx 2$ 年）

よくある誤解

- **物理法則ではなく産業トレンド**
- 近年は**電力の壁**や**製造コスト**で“周波数の倍増”は止まったが、密度・エネルギ効率は新構造（FinFET/GAA）や先進パケジング技術（Chiplet/3D, HBM）で改善を継続


53,250,003
42,000,000
30,000,000
27,400,000
25,000,000
23,000,000
22,000,000
21,300,000
21,000,000
20,000,000
15,000,000
data graph a
Moore's Law
vs real CPUs
1965 - 2019

CONTENTS

モジュール３：デジタル回路の基礎（１）

計算の前に、まず「論理」

ブール代数（論理代数）：「１」と「０」だけでの演算を実現する

英国の数学者ジョージ・ブール（1815～1864）が考案

- **「１」と「０」 → 「真」と「偽」**
- 「論理和」「論理積」「論理否定」　（参考：四則演算は加減乗除）

OR　　AND　　NOT

命題１：猫は動物の一種	・・・	真　（１）
命題２：今年は昭和50年	・・・	偽　（０）

AND：命題が２つとも真なら、真
OR：どちらか一方が真なら、真
NOT：１つの命題を対象とし、真と偽を逆に

論理演算① AND（論理積）

命題A：期末試験に合格する

命題B：レポートを提出する

結果：単位取得（論理積）

命題A	命題B	結果
偽	偽	偽
真	偽	偽
偽	真	偽
真	真	真

真理値表

A	B	A・B
0	0	0
1	0	0
0	1	0
1	1	1

論理演算② OR（論理和）

命題A：ICカードを使う

命題B：現金を使う

結果：買い物できる（論理和）

命題A	命題B	結果
偽	偽	偽
真	偽	真
偽	真	真
真	真	真

真理値表

A	B	A＋B
0	0	0
1	0	1
0	1	1
1	1	1

論理演算③ NOT（論理否定）

命題A：ドアがロックされている

結果：中に入る（論理否定）

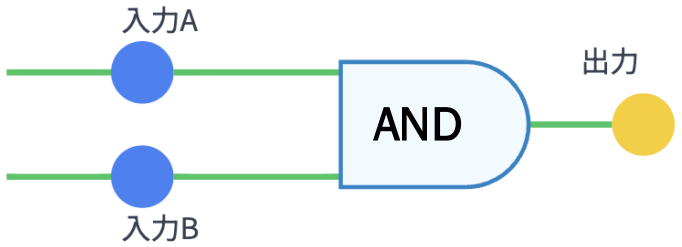
命題A	結果
偽	真
真	偽

真理値表

A	¬A/A’
0	1
1	0

論理から物理へ：「論理ゲート」

ブール論理の各演算を、トランジスタを組み合わせて物理的に実現した回路を論理ゲート

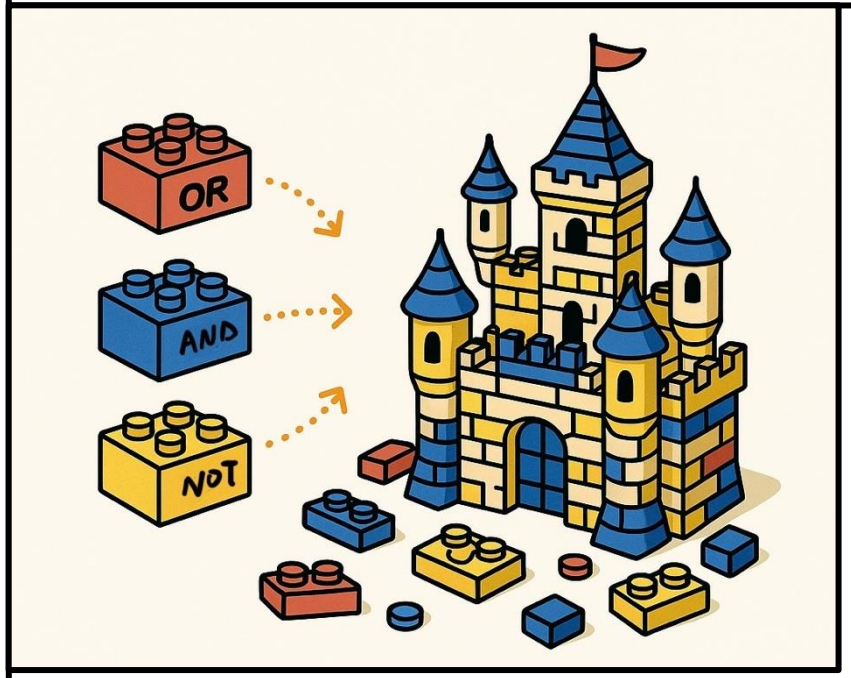


論理ゲートの詳細

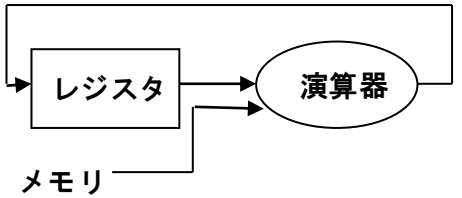
CONTENTS

モジュール３：デジタル回路の基礎（１）

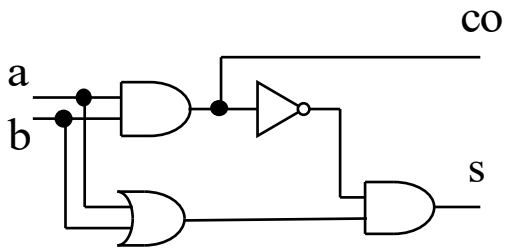
組み合わせの威力



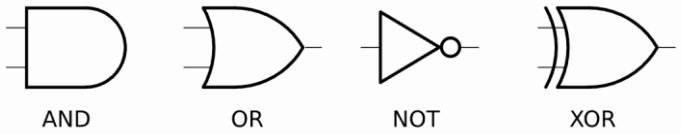
CPU



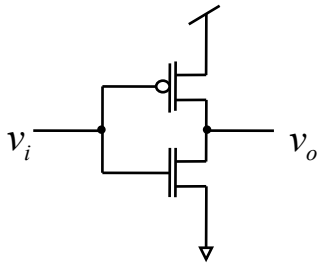
Circuit



Logic Gate



Transistor



小さなゲートから大きな計算へ

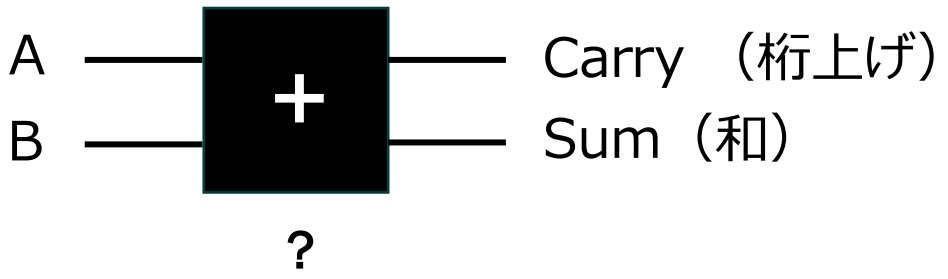
目標：2進数1桁の足し算

10進数の足し算

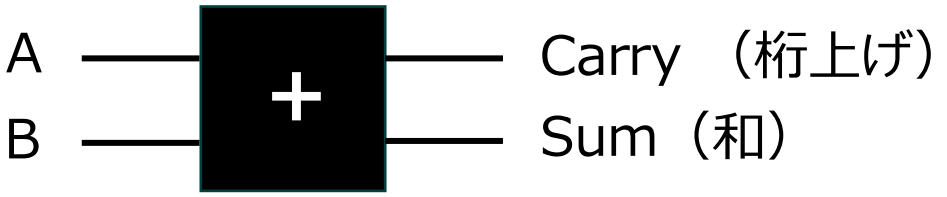
```
  4        9
+ 5      + 5
---      ---
 0 9      1 4
```

2進数の足し算

```
  0        1
+ 1      + 1
---      ---
 0 1      1 0
```

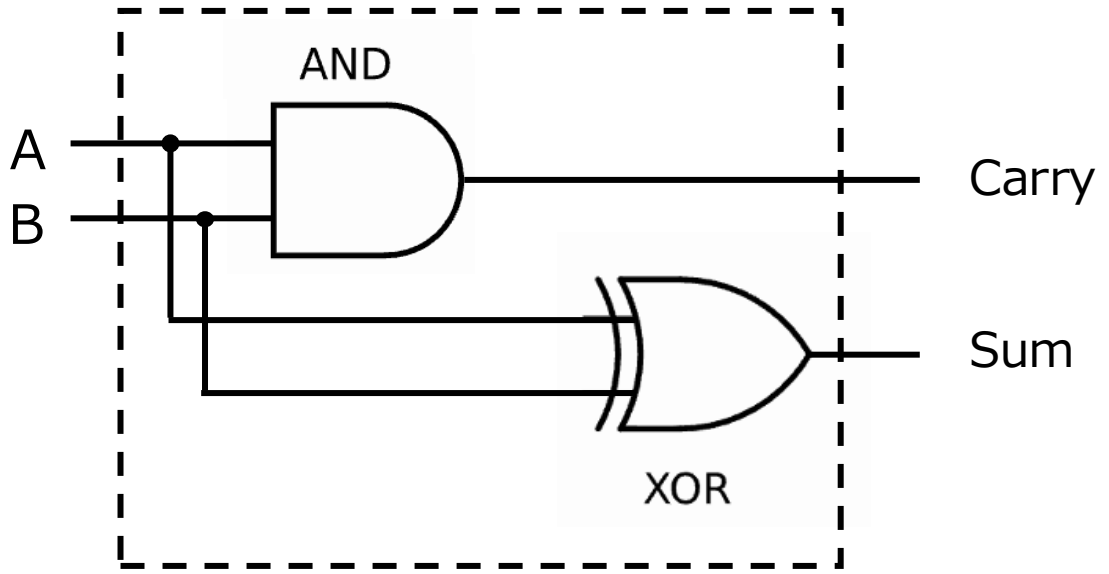


足し算を分解して考える



A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Carry = A・B　　Sum = A ⊕ B



半加算器　Half Adder

下の桁からの繰り上がりを入れられない！

全加算器 (Full Adder) の回路構成

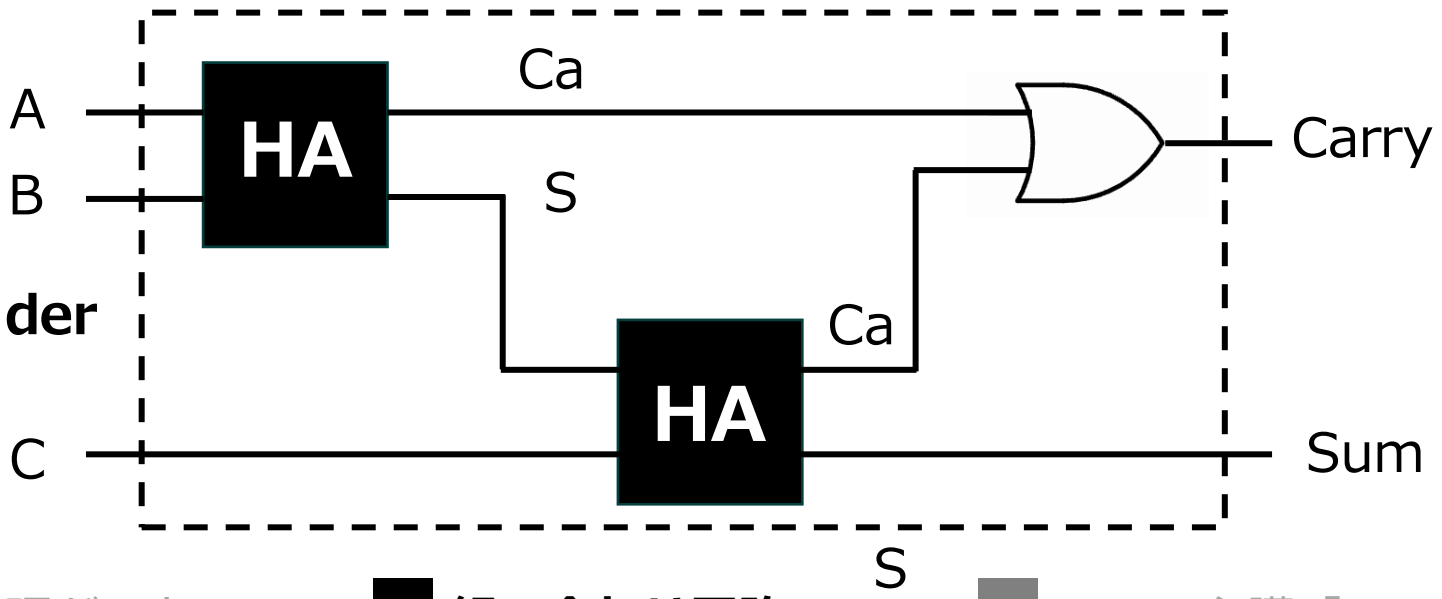
10進数の足し算

```
  4        9
+ 5      + 5
-----    -----
  0 9      1 4
+   7    +   7
-------  -------
  1 6      2 1
```

2進数の足し算

```
  0        0        1        1
+ 0      + 1      + 0      + 1
-----    -----    -----    -----
  0 0      0 1      0 1      1 0
+   1    +   0    +   1    +   1
-------  -------  -------  -------
  0 1      0 1      1 0      1 1
```

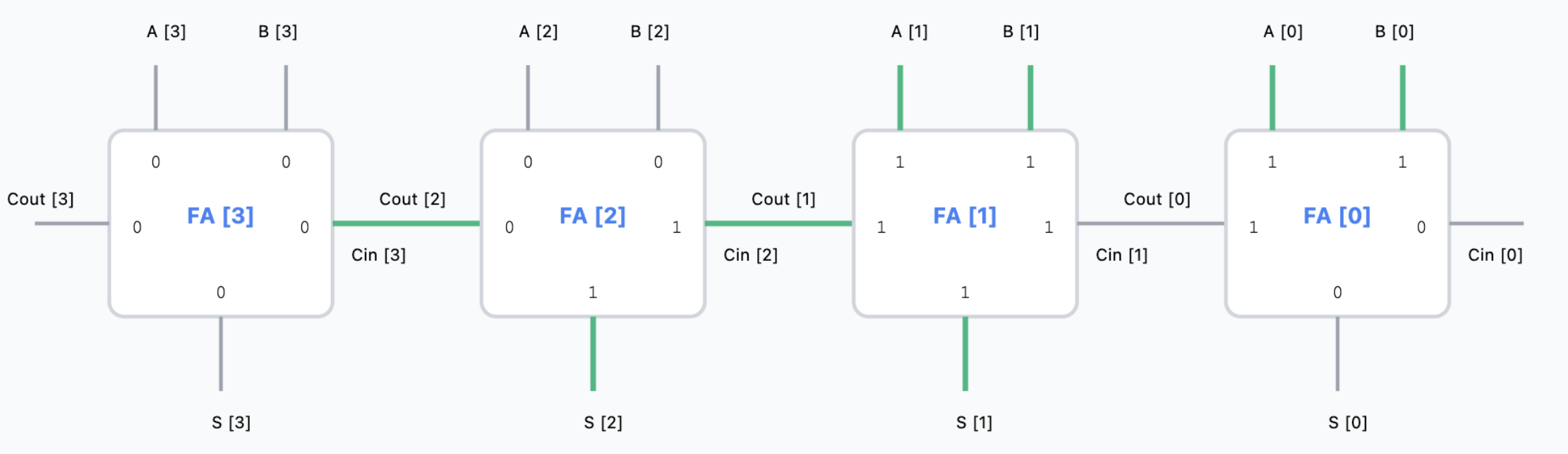
{Ca, S} = A + B + C



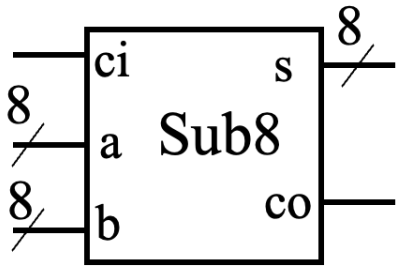
全半加算器　Full Adder

複数ビットの加算器へ

リップルキャリー加算器 (Ripple Carry Adder: RCA)

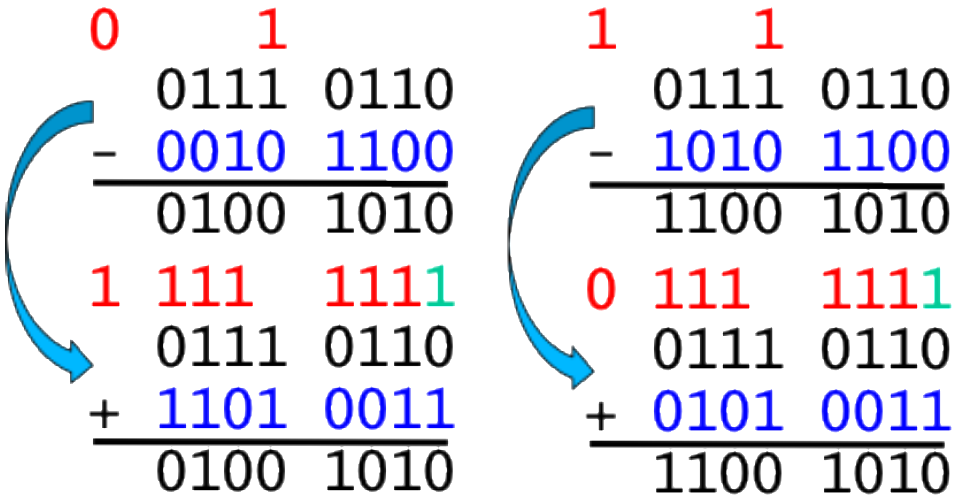


加減算器の実現



減算 = 負の数の加算

$$a - b = a + (-b)$$
$$= a + (\overline{b} + 1)$$



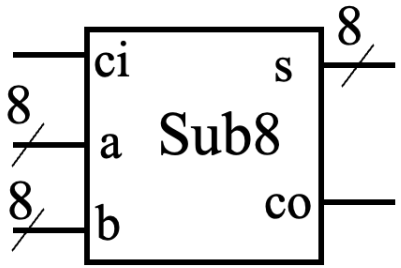
2の補数：**d** を加算して **0** になる値→ 負の数 **–d**

正の数	1	2	3	4
d	00000001	00000010	00000011	00000100
$\sim d$	11111110	11111101	11111100	11111011
$-d$	11111111	11111110	11111101	11111100
負の数	–1	–2	–3	–4

加算器の桁上げ（キャリー）入力 = 1

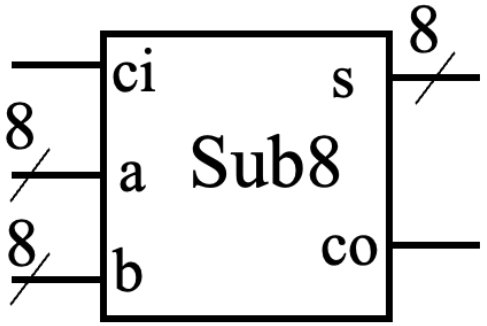
加算器のキャリー出力の反転
= 減算器の桁借り（ボロー）出力

加減算器の実現



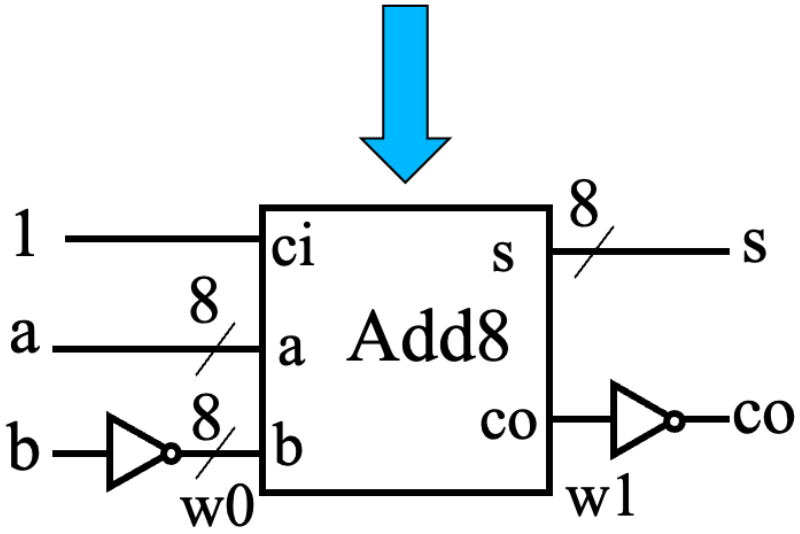
減算 ＝ 負の数の加算

$$a - b = a + (-b)$$
$$= a + (\overline{b} + 1)$$



2の補数：**d** を加算して **0** になる値→ 負の数 **–d**

正の数	1	2	3	4
d	00000001	00000010	00000011	00000100
$\sim d$	11111110	11111101	11111100	11111011
$-d$	11111111	11111110	11111101	11111100
負の数	–1	–2	–3	–4

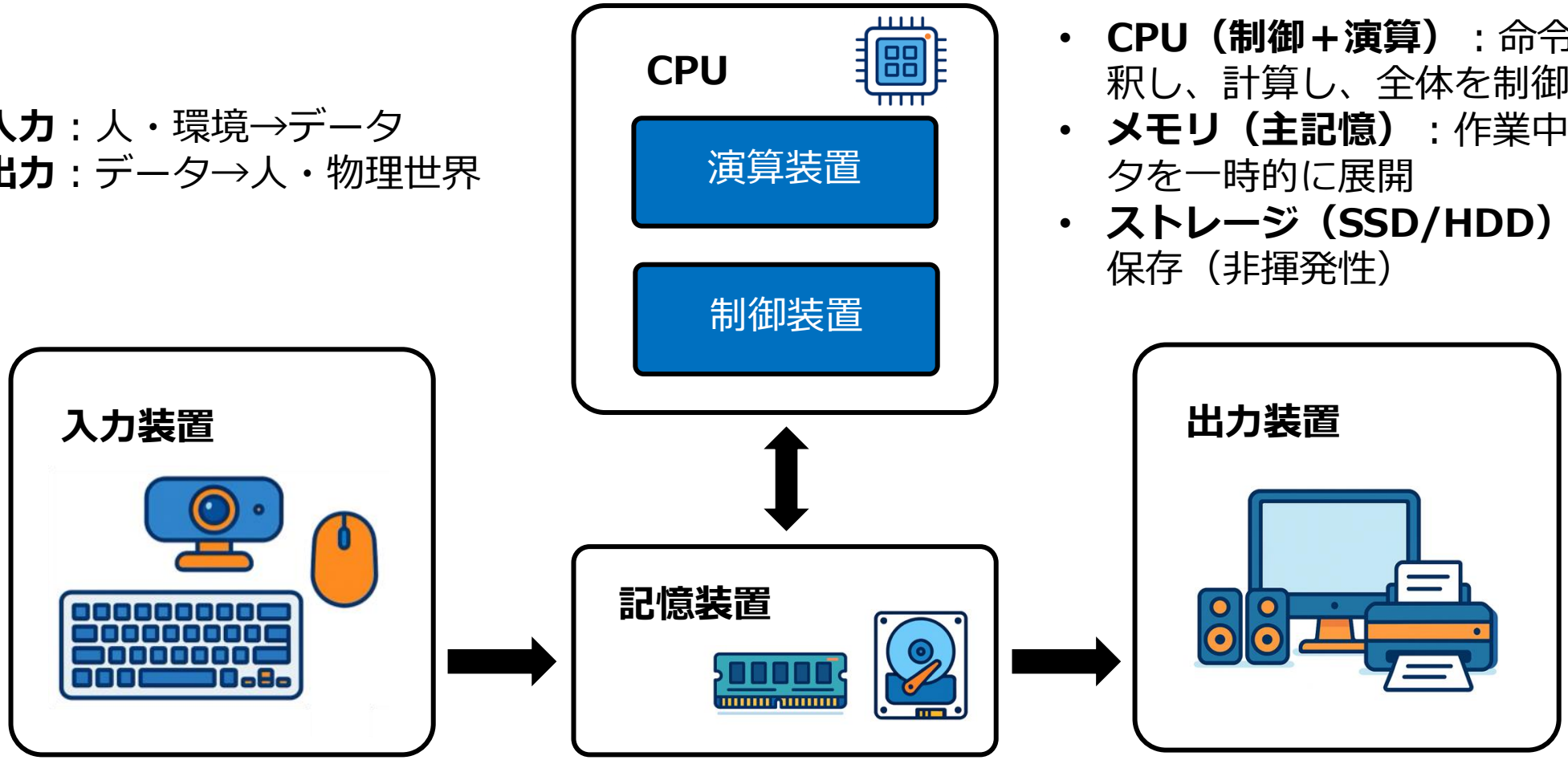


CONTENTS

モジュール３：デジタル回路の基礎（１）

前回の復習：コンピュータの５大装置

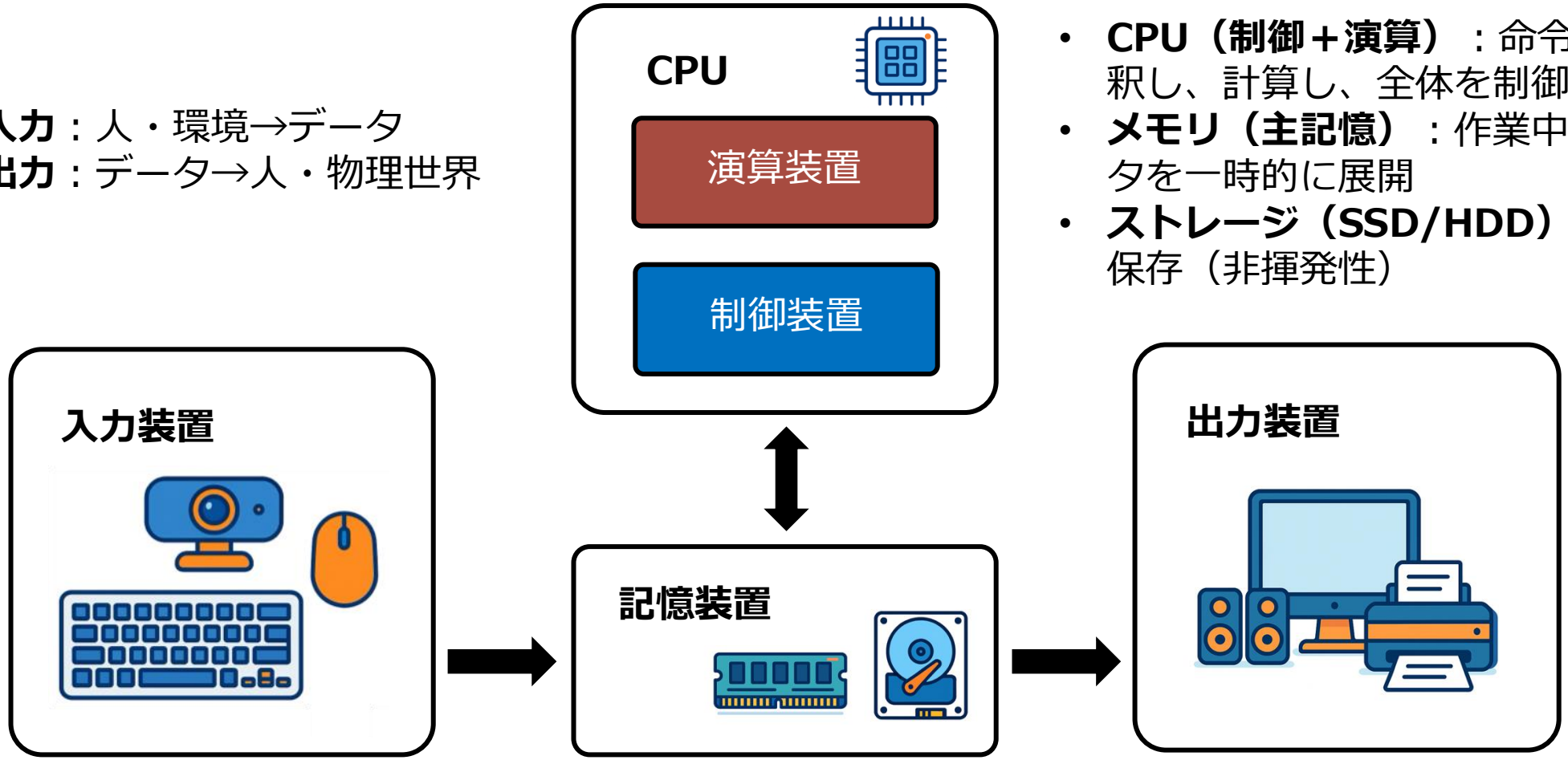
- **入力**：人・環境→データ
- **出力**：データ→人・物理世界



- **CPU（制御＋演算）**：命令を解釈し、計算し、全体を制御
- **メモリ（主記憶）**：作業中のデータを一時的に展開
- **ストレージ（SSD/HDD）**：長期保存（非揮発性）

前回の復習：コンピュータの５大装置

- **入力**：人・環境→データ
- **出力**：データ→人・物理世界

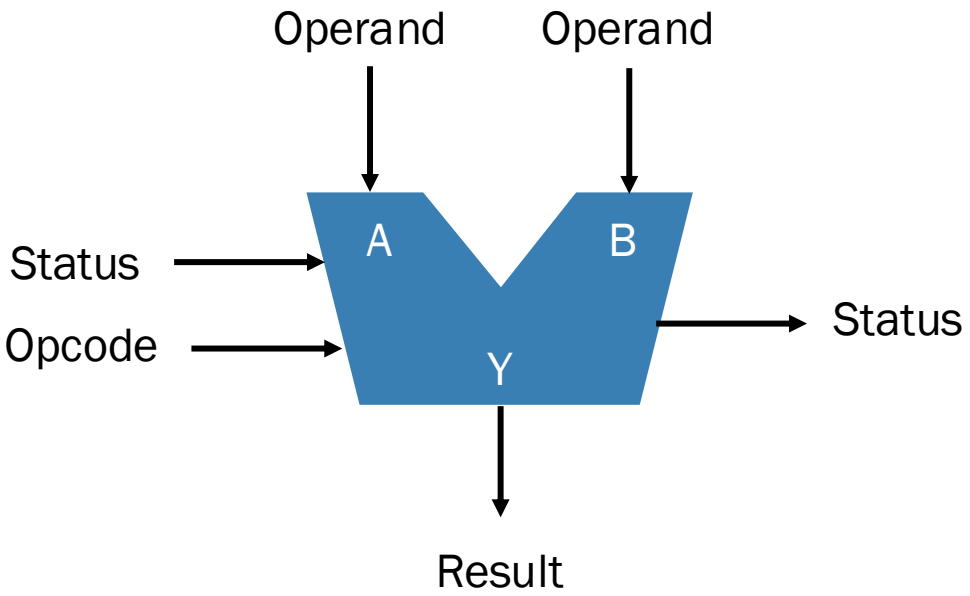


- **CPU（制御＋演算）**：命令を解釈し、計算し、全体を制御
- **メモリ（主記憶）**：作業中のデータを一時的に展開
- **ストレージ（SSD/HDD）**：長期保存（非揮発性）

CPUの計算センター「ALU」

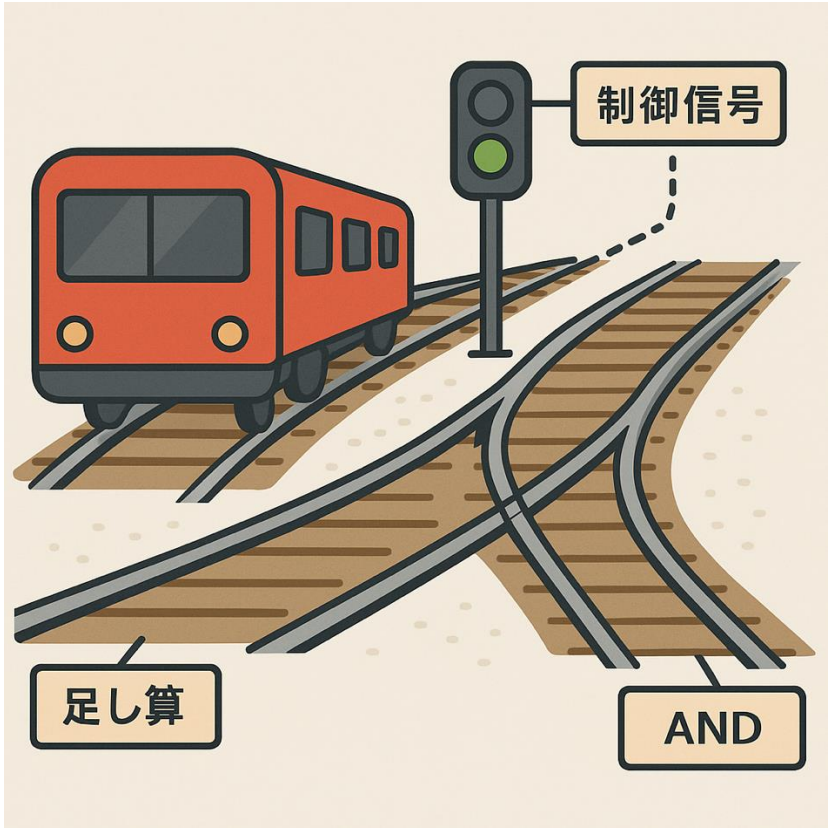
ALU(**a**rithmetic **l**ogic **u**nit、算術論理演算装置)

- 計算をまとめて行うCPUの「計算センター」
- 算術演算：足し算、引き算
- 論理演算：AND、OR など



回路の切り替え役「マルチプレクサ」

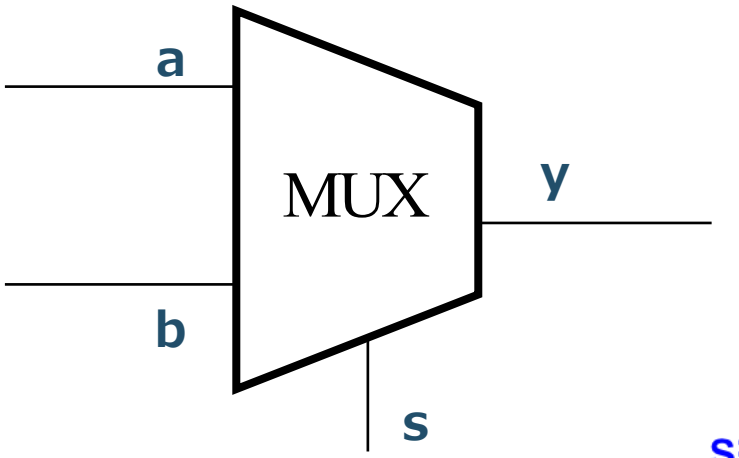
計算の切り替えスイッチ：マルチプレクサ（MUX）



- 「制御信号」で計算を選択
- 例：足し算？ それとも AND？

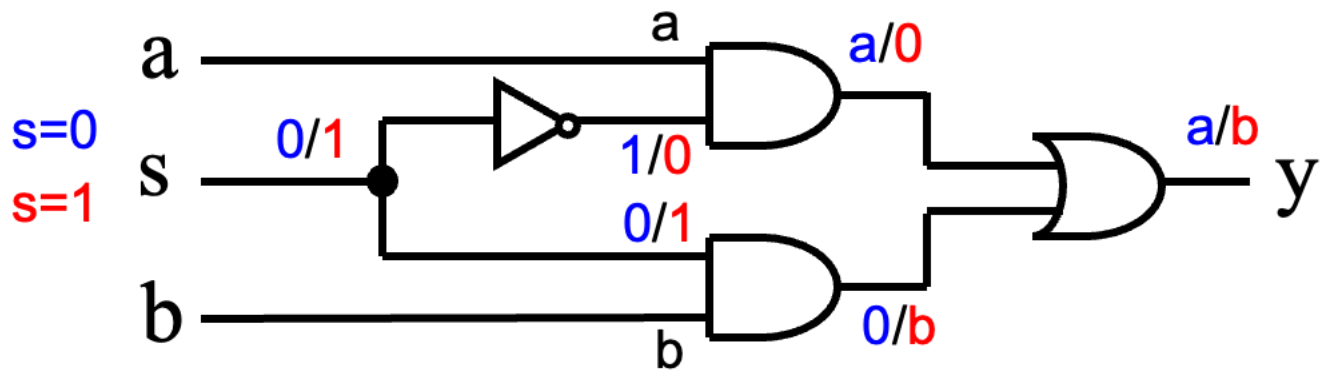
回路の切り替え役「マルチプレクサ」

計算の切り替えスイッチ：マルチプレクサ（MUX）



s	y
0	a
1	b

1bitマルチプレクサの回路図



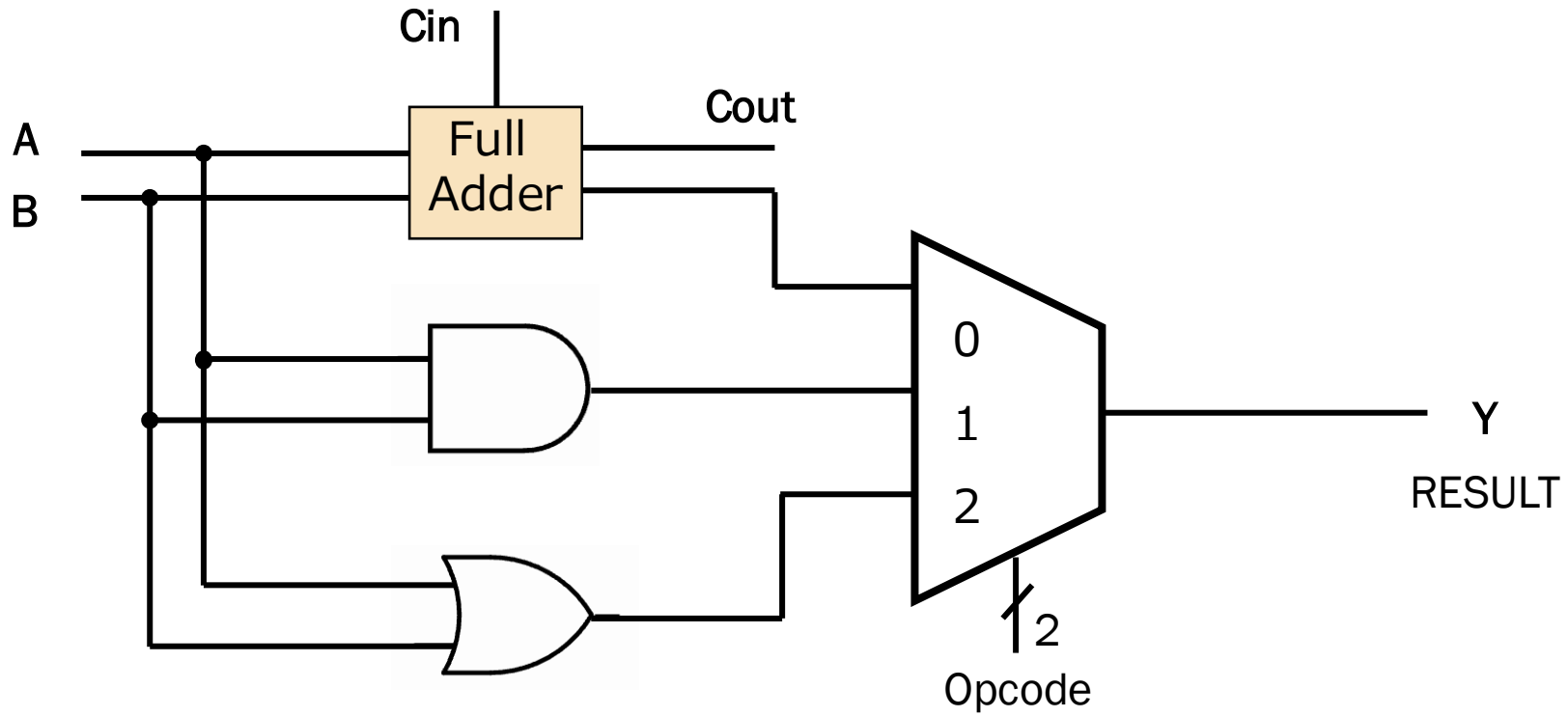
2つの1bitの入力の中から1つを選択して出力

ALUの内部構造

1-bit ALUを設計してみる
機能：１ビットの入力AとBを、選択入力Sによる、AND、OR、足し算

ALUの内部構造

1-bit ALUを設計してみる
機能：１ビットの入力AとBを、選択入力Sによる、AND、OR、足し算



次回予告

次回：コンピュータはどうやってその結果を『記憶』している