

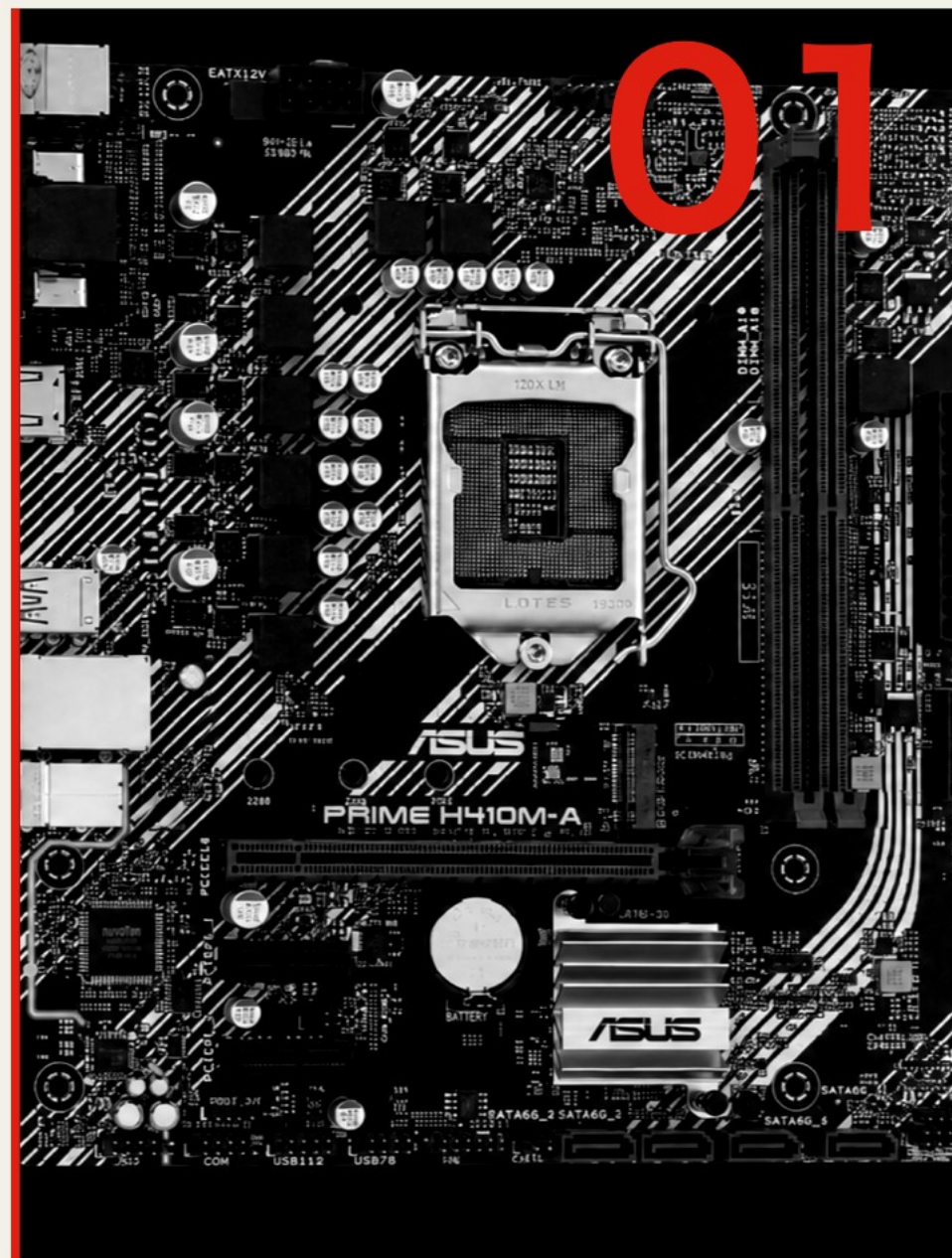
KYUSHU UNIVERSITY · SYS-101

コンピュータ システム入門

箱の中を、役割から読む

担当教員 Olivia Chen Toyama C2LAB

2026年9月 LECTURE 1





例 | スマートフォン、腕時計、カメラなど



LOOK INSIDE / BEFORE THE LABELS

?

「魔法の箱」の中で、 何が起きている？

複雑に見えるのは、違う役割をもつ部品が、一つの仕事を協力して進めているからです。

魔法の箱を、部品と役割に分けて 読めるようになる

01

観察する

形、位置、接続、印字から手掛かりを集める。

02

役割を分ける

入力・記憶・制御・演算・出力の役割で整理する。

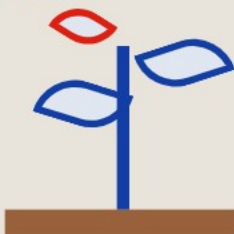
03

協力を追う

一つの情報が部品の間をどう移動するか説明する。

研究の問いを、データで確かめる

EXAMPLE / PLANT GROWTH



研究の問い

温度が違くと、
植物の育ち方は変わる？



01 / MEASURE

温度を測り、
写真を撮る



02 / RECORD

毎日のデータ
を
残す



03 / COMPARE

条件ごとの違
いを
比べる



04 / EXPLAIN

結果をグラフで
伝える

「なんとなくそう見える」から **データを根拠に、答えを考える。**

この授業で学ぶ基礎

必要な装置と処理を、自分の目的から考える。

これは

プログラミングの授業 ではありません。

コードを書く前に、 **コードが動く「下の世界」を理解する授業です。**

三日間で、部品から未来まで登る

DAY 1・9/7

構成と情報表現

DAY 2・9/8

記憶とアーキテクチャ

DAY 3・9/9

OS と未来の計算

DAY 1・9/7

構成と情報表現

10:00-17:00

AM

10:00-12:30

10:00-11:30

LECTURE 1

コンピュータの基本構成

11:45-12:30

OBSERVATION +
PRACTICE

実物観察と二つの演習

PM

13:30-17:00

13:30-15:00

LECTURE 2

情報のデジタル表現+ミニ体験
15分

15:15-16:15

LECTURE 3

デジタル回路の基礎 (1)

16:15-17:00

PRACTICE

演習1：論理回路のアトリエ

休憩・昼休みを含む。進行により時刻は調整する場合がある。

01

何が変わり、 何が残ったのか

現在の形だけを見る前に、コンピュータの変化をたどる。

外形

01

形はどう変わった？

部屋 → 机 → 手のひら

仕事の変更

02

仕事はどう切り替える？

配線 → プログラム

共通点

03

どんな役割が残った？

入力・記憶・制御・演算・出力

部屋から机へ、そして手のひらへ



STEP 5 / 5 ・ 現在へ

2000年代～ スマートフォン

いつでも持ち歩く

通信・写真・地図など、
多くの機能を一台に。

小型化・集積化 多くの回路を小さなチップにまとめる → 使う人と場所が広がった

戻る

1 ENIAC

2 IBM 1401

3 Intel 4004

4 IBM PC

5 スマートフォン

次へ

[発展学習：コンピュータの歴史](#) ↗

仕事を変える方法が、変わった

ENIAC 初期の方式



開発の目的

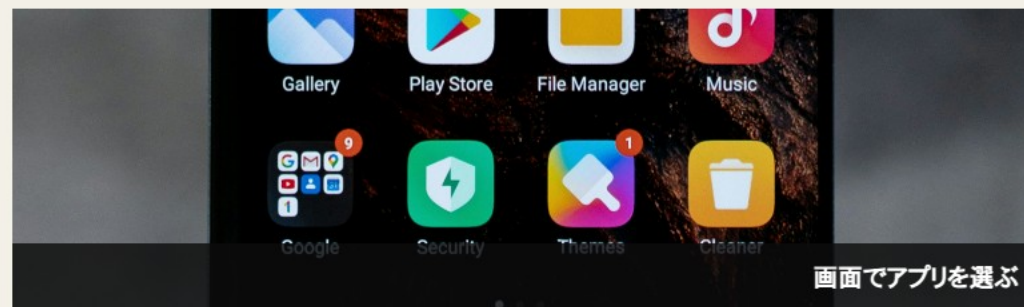
砲弾の軌道を計算し、射表を作る。

配線・スイッチを設定し直す

別の問題も扱えるが、準備に手間がかかる。

プログラムは、処理の手順を表す命令の集まりである。

スマートフォン 現在の方式



同じ本体で

複数のアプリを実行する。

実行するプログラムを切り替える

本体を配線し直さず、処理する仕事を変える。

外形が違っても、五つの機能がある



部品名を覚える前に、受け取る・記憶する・指示する・計算する・伝える、という機能を見る。

五つの機能は、プログラムに従って動く

CAMERA

MAP

MUSIC

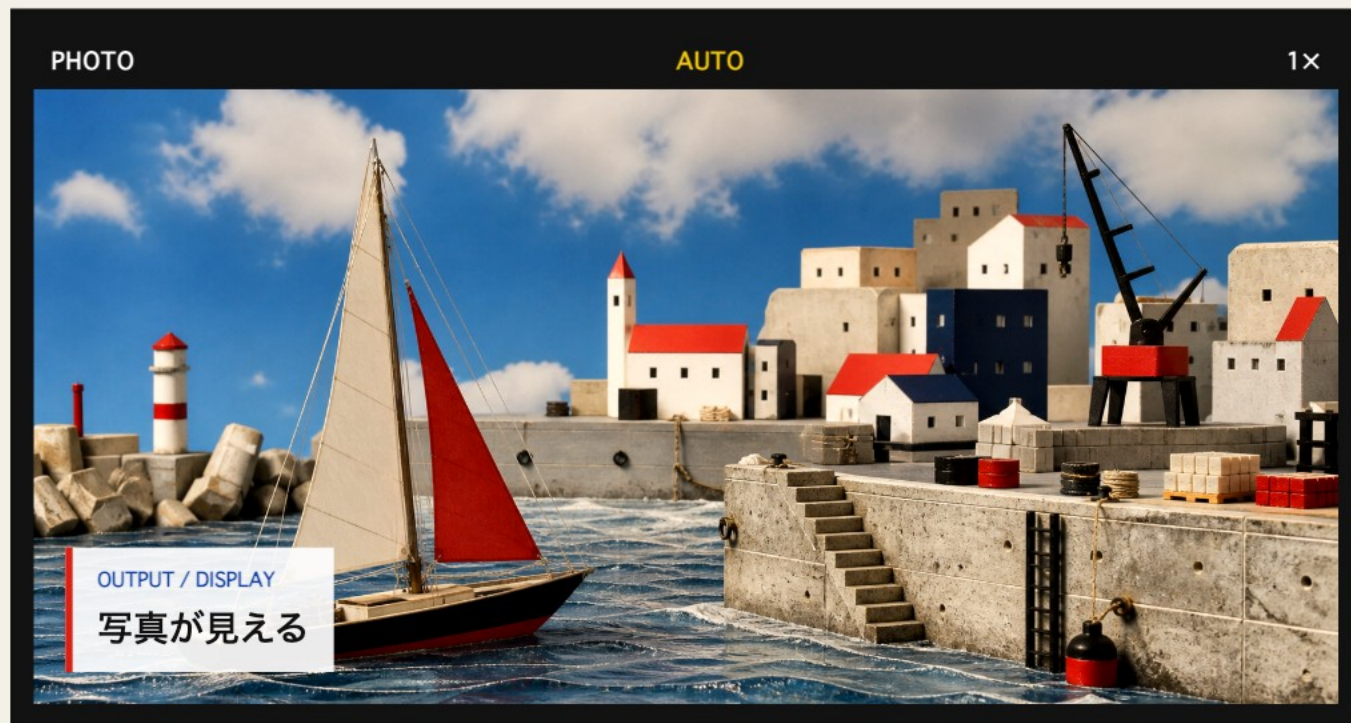


FIVE FUNCTIONS

受け取る → 記憶する →
指示する・計算する → 伝える

プログラムは、五つの機能を「何を・どの順に」使うか決める。

一枚の写真で、五つの機能が協力する



STEP 5 / 5

出力：完成した写真を見せる

画面の画素が光り、写真として見える。

1 入力

2 記憶

3 制御・演算

4 出力

一枚の写真に、五つの機能が参加した。

戻る

1 撮影前

2 感光

3 一時保存

4 画像処理

5 表示

次へ

利用者は、五つの機能を直接操作しない

あなたがすること

撮影ボタンを押す

アプリが頼むこと

写真を撮り、保存する

OS(基本ソフト)の役割

カメラ・メモリ・ファイルを管理する

ハードウェアがすること

五つの機能を使って処理する

必要な操作だけを見せ、細かな手順を下の層に任せる。

コンピュータとは

入力	記憶	制御	演算	出力
----	----	----	----	----

を、プログラムに従って行う装置。

外形や用途が違って、この五つの機能をプログラムが結び付ける。

同じコンピュータを、五つの層で見る

05	アプリ	利用者の仕事を実現する
04	OS	資源と実行を管理する
03	アーキテクチャ	命令と機械の約束を定める
02	論理	0 と 1 で判断と記憶を作る
01	物理	部品、配線、電気で実体を作る

今日はどの層を開けるのか

05	アプリ	使う
04	OS	Day 3
03	アーキテクチャ	Day 2
02	論理	午後：0 / 1 と論理ゲート
01	物理	午前：部品と役割

まず「見える部品」を読み、そこから「見えない論理」へ進む。

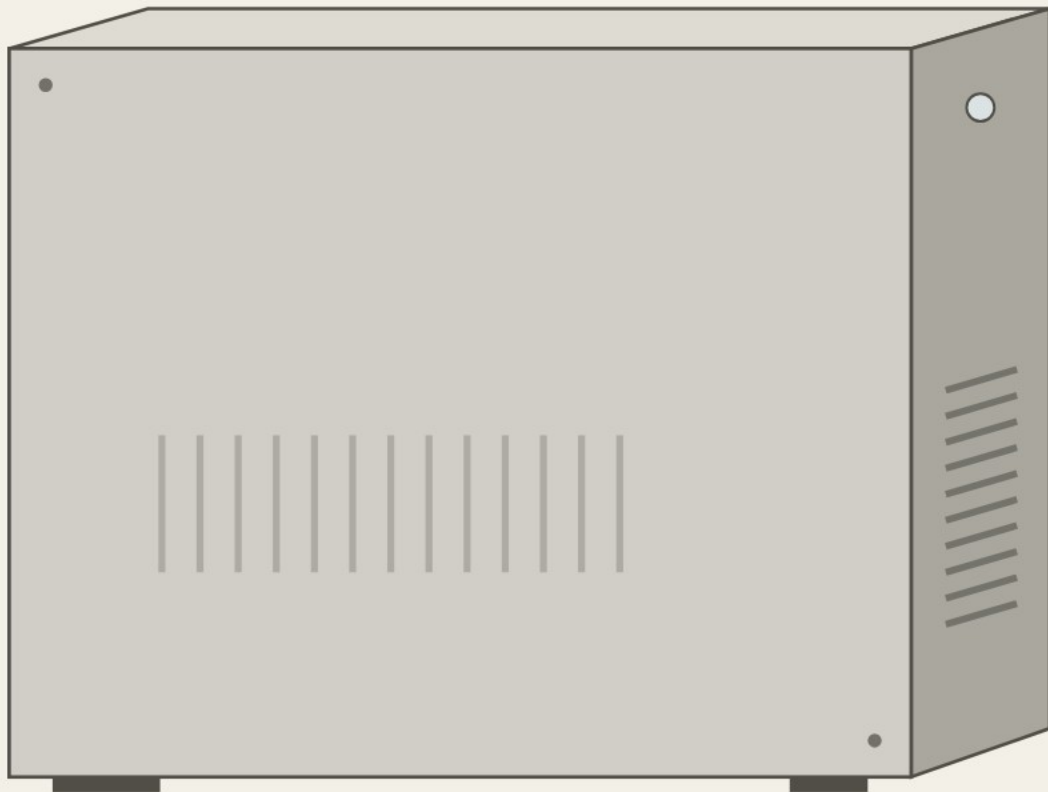
PART 2 / READ THE PHYSICAL SYSTEM

PCの中を 読む

写真図で中を観察し、模式図で部品のつながりを読む。



この箱の中には、何がある？



見えない内部を、想像する

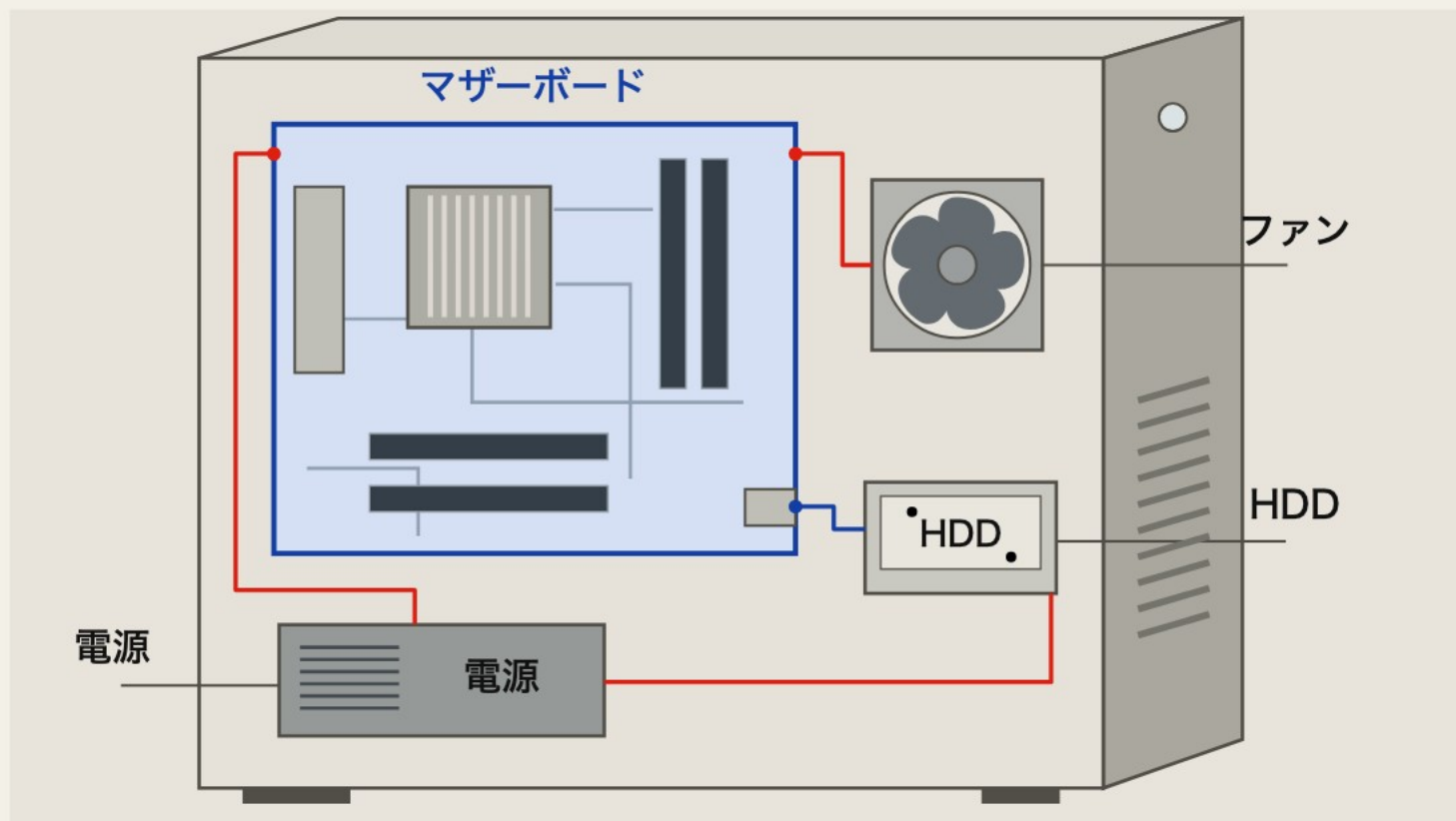
保存するもの。

冷やすもの。

電力を供給するもの。

それぞれの部品は、
どのようにつながっているのだろう。

ケースを開き、見た目から接続図へ



MOTHERBOARD

これが、マザーボード

CPU や RAM を取り付け、部品
の間に信号と電力の通り道を作る
基板である。

— 信号の接続 — 電力の供給

次は基板を取り出し、接続の役割を順番に見る。

戻る

1 ケース

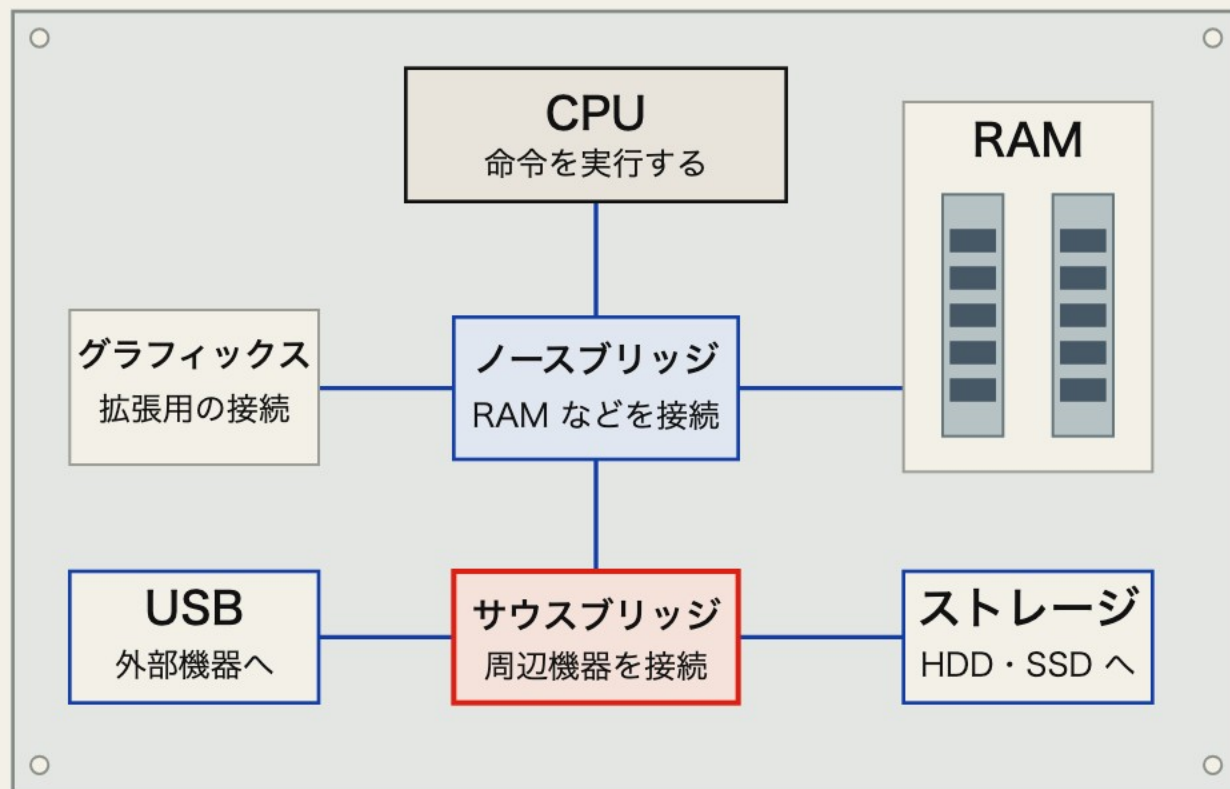
2 中を見る

3 接続を見る

4 基板の役割

次へ

マザーボードは、部品をつなぐ基板



READ THE BOARD

周辺の接続をまとめる

HDD・SSD や USB などの接続を受け持ち、ノースブリッジを介して CPU につながる。

サウスブリッジと呼ばれていた。

部品配置と配線を簡略化した模式図。特定の製品を再現したものではない。

戻る

1 基板

2 CPU

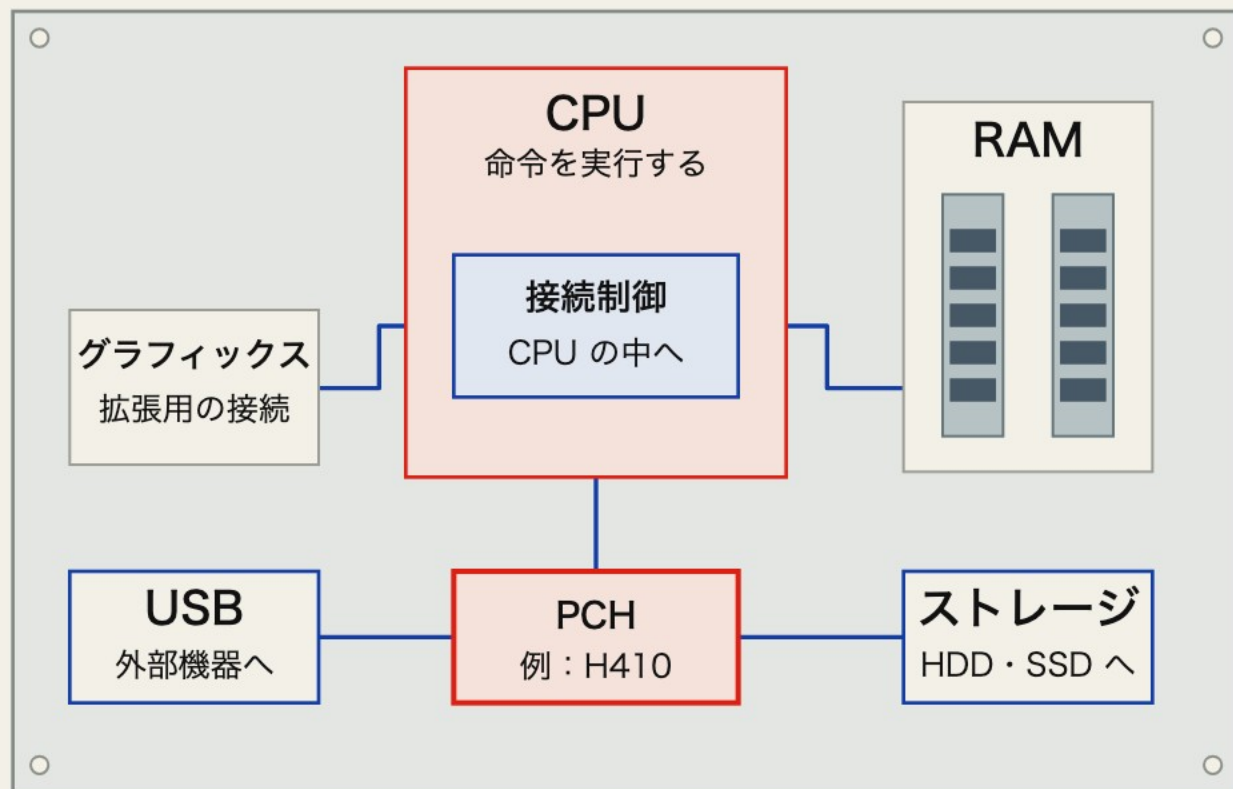
3 RAM

4 ノースブリッジ

5 サウスブリッジ

次へ

接続機能の一部は、CPU の中へ



FUNCTIONS MOVE

周辺接続は、PCH が担当

H410 では、ストレージや USB をつなぐ回路が、CPU とは別のチップに残っている。

今日の基板にはノースブリッジはないが、H410 チップセットがある。

Intel H410 世代との比較。PCH = Platform Controller Hub。

戻る

1 従来の配置

2 CPU に統合

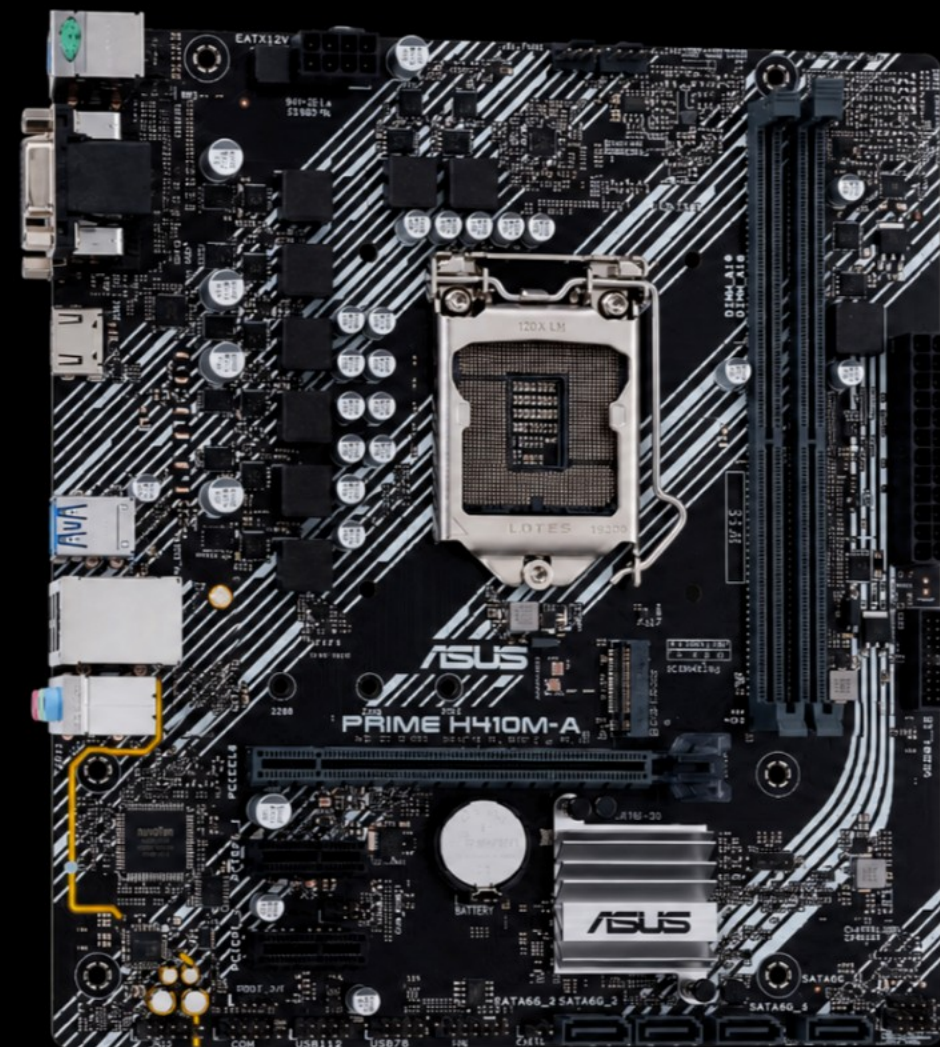
3 PCH に注目

次へ

PHYSICAL ACTIVITY / PASS IT AROUND

H410 マザーボードを 手に取って観察する

画面は別製品の参考写真です。手元の基板では、配置の違いも確かめてください。



参考写真 | ASUS PRIME H410M-A

図で見た部品を、実物の形から確かめる



01

印字

数字や規格名は何を示す？

02

端子

何が、どの向きでつながる？

03

スロット

長さや切り欠きが違うのはなぜ？

04

固定具

どの部品を動かないようにする？

H410 基板の参考例で、接続先を確かめる



CPU 側につながる

- 1 CPU ソケット**
CPU を取り付ける場所。写真は未装着。
- 2 RAM スロット**
装着した RAM は、CPU 内の回路につながる。

PCH 側につながる

- 3 H410 チップセット**
銀色の放熱板の下にある。
- 4 SATA 端子**
HDD や SSD をケーブルで接続する。
- 5 USB 端子の例**
外部機器との接続を担う。

独立したノースブリッジはない。
接続の仕事は、CPU と H410 に分担されている。

コンピュータの五大装置



先ほどの五つの機能を、入力・記憶・制御・演算・出力の五大装置として整理する。

記憶装置は、データと命令を置く



いま使う場所と、長く残す場所。速さと役割が違う。

制御装置は、次に何をするか決める



命令を読み、必要な装置へ「いま動く」という制御信号を送る。

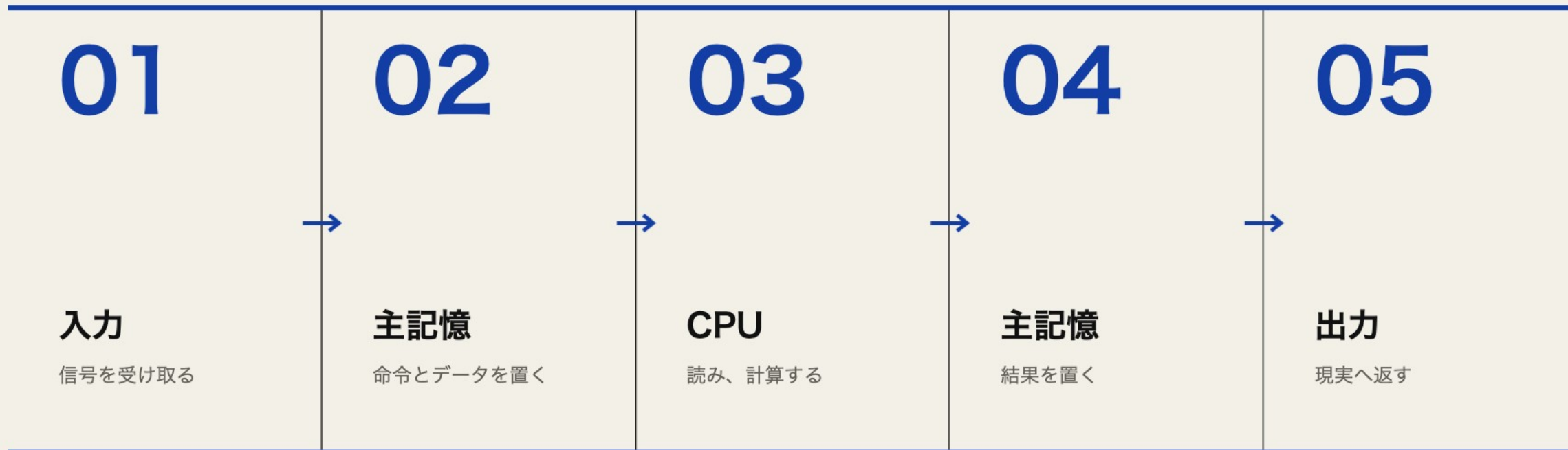
演算装置は、計算と比較を行う



03 制御と 04 演算は、CPU の主要な構成要素である。

演算装置に戻る

データは、どの部品を通して移動するのか



データは一方向に流れ続けるのではなく、CPU と主記憶の間を何度も往復する。

制御信号とデータの流は、何が違うのか

CONTROL · RED

いつ、何を動かすか

読む、書く、計算する、送るという動作の開始と順序を指定する。

制御装置 → 各装置

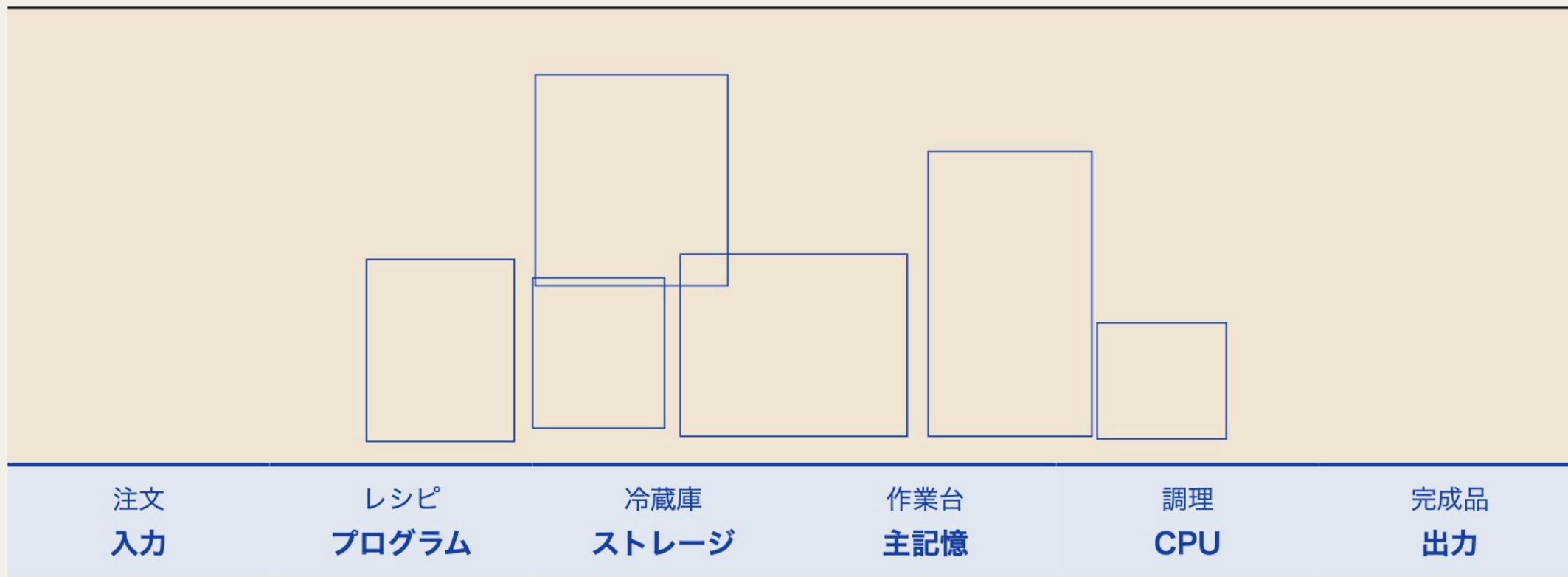
DATA · BLUE

何进行处理するか

文字、数値、画像、音声、命令など、処理と記憶の対象を運ぶ。

入力 ↔ 記憶 ↔ CPU ↔ 出力

一つの注文を、料理に変える



同じ役割を、部品の名前で読む 入力を受け、命令とデータを使って処理し、結果を出す。

[戻る](#)[1 注文](#)[2 手順](#)[3 準備](#)[4 調理](#)[5 提供](#)[6 部品へ](#)[次へ](#)

手順を変えると、同じ厨房で別の料理になる

同じ材料

野菜

レシピ A

切る → 盛り付ける

サラダ

レシピ B

切る → 煮る

スープ

コンピュータでは

プログラム

が手順を指定する。

CPU

が命令を実行する。

材料＝データ、レシピ＝プログラム。**処理する部品と、処理の手順は別である。**

保存する場所と、作業する場所を分ける

冷蔵庫 ストレージ あとで使うものを 保存しておく。	作業台 主記憶 (RAM) いま使う命令とデータを 一時的に置く。	調理する担当 CPU 命令に従って 計算・判断する。

次は、この三つの役割を実際のコンピュータで確かめる。

次は、処理の中心を詳しく見る



02 記憶・03 制御・04 演算に注目する。まずは、CPU から。

CPU は、命令を実行する

制御

次の動きを指示する

命令を読み、必要な回路へ
合図を出す。

演算

計算・比較する

足す、引く、大小を比べる。
条件に応じて処理を進める。

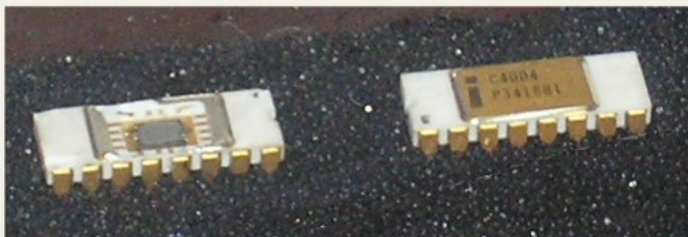
CPU 内部にも記憶はある

ただし容量は限られる。大きなプログラムやデータを扱うには、**主記憶 (RAM)** が必要になる。

CPU は、どう進化してきたのか

1971

Intel 4004



内部（左）／外観（右）

CPU を1チップに

命令を実行する回路を、
小さな半導体チップに集積する。

電卓などへ組み込む

1981

IBM PC に Intel 8088 を採用



8088 の外観

個人のコンピュータへ

CPU にメモリや入出力を組み合わせ、
さまざまなソフトを動かす。

机上の PC を動かす中心に

2000年代～

PC にもマルチコアが広がる



コア＝命令を実行する単位

複数のコアで分担する

分けられる仕事を同時に進め、
処理できる量を増やす。

例：2005 年の Pentium D

小さくする → 広く使う → 複数のコアで分担する CPU の役割を保ちながら、実現する方法が変わってきた。

代表的な転機を抜粋。写真は同一縮尺ではない。8088 写真：Jud McCranie / CC BY-SA 4.0

一つの CPU の中には、何が入っている？

外から見える部品



CPU パッケージ

回路を保護し、
マザーボードにつなぐ。

写真：Core i7-940（外観例）

ふたの下を見ると



回路を作り込んだ部分

小さなシリコン片に、
多数の回路が入っている。

ダイの中を、役割で分ける



4 コア・1 枚のダイを例にした模式図。

製品により構成は異なる。

一つの CPU に、複数のコアが入る。 コアは、一つひとつが命令を実行する単位である。

CPU は写真で観察する。写真：Rainer Knäpper / Free Art License（表示サイズのみ変更）

コア数と GHz から、何が分かる？

読み方を考える架空の仕様

実在製品の比較ではない

	CPU A	CPU B
コア数	4 コア 	8 コア 
動作周波数	3.6 GHz	3.0 GHz

この二つの数字だけで、
「どちらが速い」と決められる？

01 コア数：仕事を分担する単位の数

分けられる仕事では、複数のコアが働ける。
8 コアでも、すべての仕事が 2 倍速くなるわけではない。

02 GHz：回路を動かすクロックの頻度

1 GHz = 1 秒間に 10 億回の周期。
1 周期で進む仕事の量は、CPU や処理によって違う。

用途・世代を確認し、実測性能を見る。

周波数やコア数だけでは、CPU の速さは決まらない。

PC を選ぶとき

文書・Web

操作への応答、価格とのバランス

動画の書き出し

使うソフトでの処理時間、並列処理の効果

次は、命令とデータを置く記憶装置



02 記憶装置に注目する。主記憶と補助記憶の役割を分けて見る。

記憶装置は、二つの役割に分かれる

MAIN MEMORY / RAM

主記憶

いま使うものを広げる

実行中の命令と、作業中のデータ。

電源を切ると内容は失われる。

使うときに

← **読み込む**

保存する →

変更を残すときに

SECONDARY STORAGE / SSD・HDD

補助記憶

あとで使うものを残す

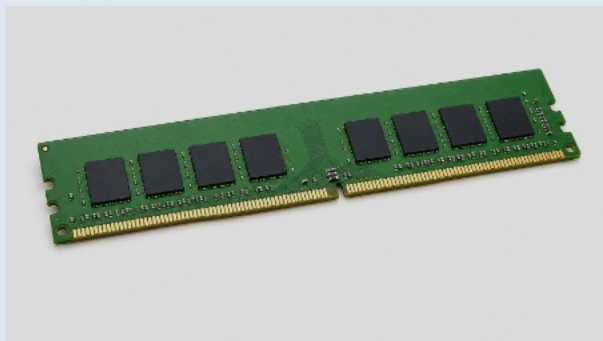
アプリや、文書・写真などのファイル。

電源を切っても内容が残る。

補助記憶はストレージとも呼ぶ。CPU は、主記憶に読み込まれた命令とデータを使う。

「いま使う」と「残しておく」は違う

主記憶 RAM



いま使う命令とデータ

アプリの実行中に使う場所。

電源を切ると、内容は失われる。

補助記憶 ストレージ



あとで使うファイル

OS・アプリ・写真などを保存する場所。

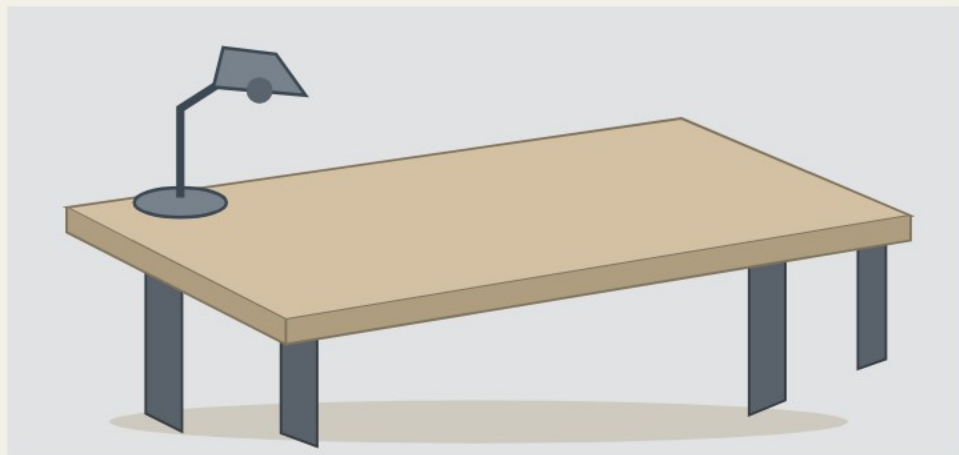
電源を切っても、内容を保持する。

容量の GB（ギガバイト）は、置けるデータの量を表す。

主記憶は机の上、ストレージは本棚

机の上 = いま使う場所

主記憶 (RAM)

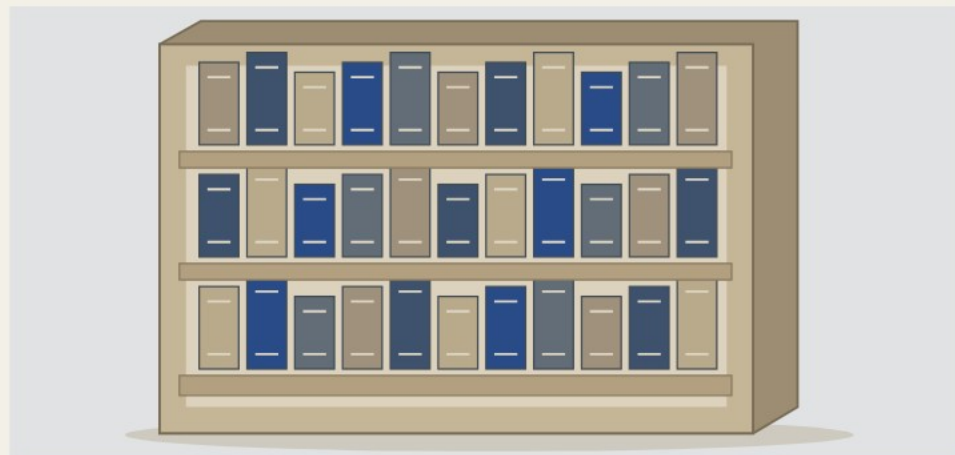


すぐ使う作業場所。広さには限りがある。

電源 OFF : 内容は失われる。

本棚 = あとで使うものを置く場所

補助記憶 (ストレージ)



取り出す手間はあるが、多くの本を置ける。

電源 OFF でも、保存した内容は残る。

このたとえでは、**消灯すると机の上だけを空にする**。本棚の本は、そのまま残す。

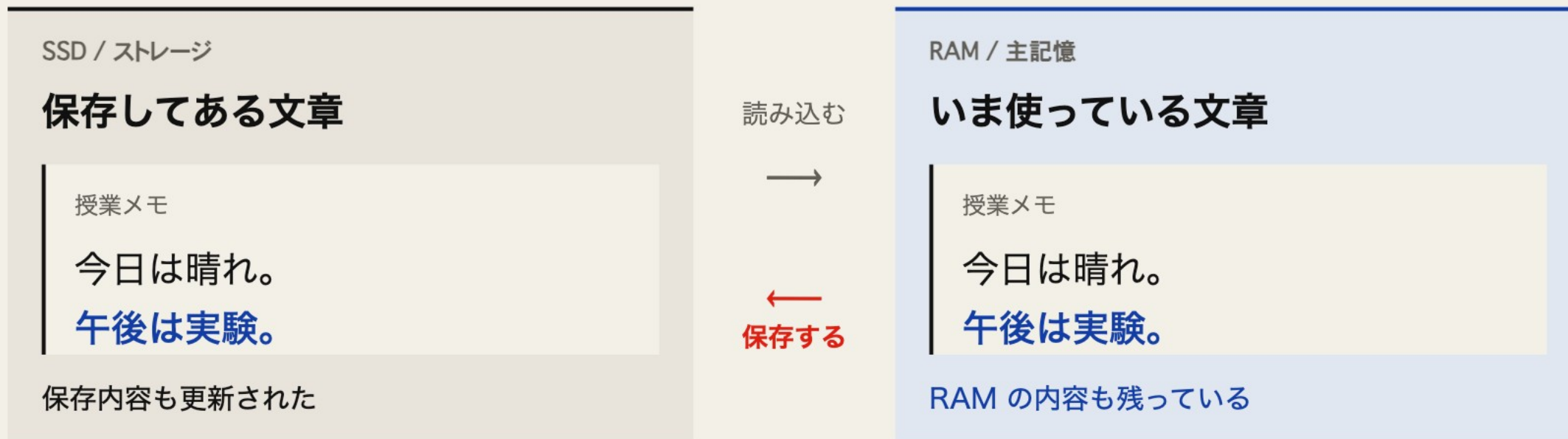
戻る

1 作業中

2 消灯 (電源 OFF)

次へ

同じ文章を、読み込み・編集・保存で追う



CPU

アプリの命令に従い、RAM 上のデータを使って処理する。

変更を SSD に書き込む。これで、あとから読み直せる。

戻る

1 保存済み

2 開く

3 編集

4 保存

次へ

CPU・主記憶・ストレージを組み合わせる

CPU 内：レジスタ＝演算中の値



CPU 内：少量を高速に保持

ストレージ：大容量・電源断後も保持

速度・容量・容量あたりの価格が異なる。一種類で全部をまかなわず、役割を分ける。

これが、メインメモリ (RAM)



MEMORY CHIPS

いま使う命令とデータを保持

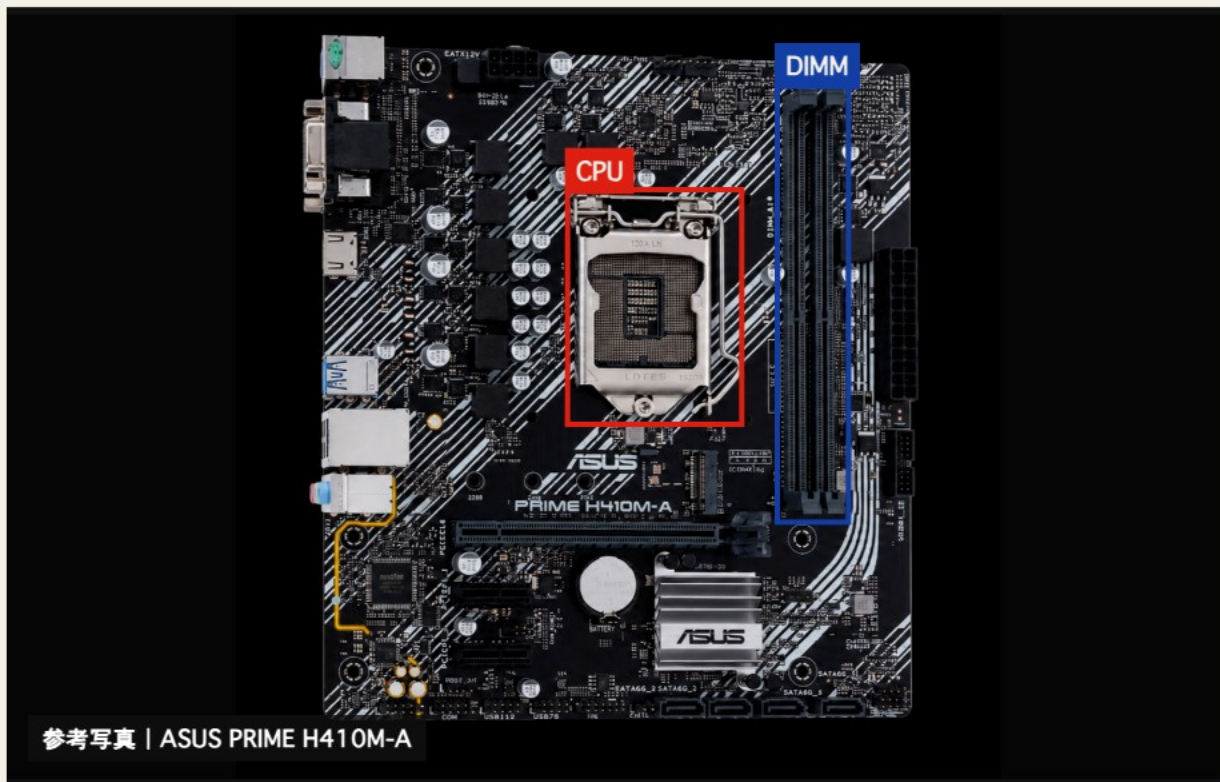
GOLD CONTACTS

スロットを通して CPU 側と接続

POWER OFF

電源を切ると内容は失われる

主記憶は、DIMM スロットに取り付ける



01 二本の長い差し込み口

写真の基板では CPU ソケットの右側。位置は製品で異なる。

02 切り欠きの向きを合わせる

規格が違う RAM は同じスロットに入らない。

03 CPU が何度も読み書きする

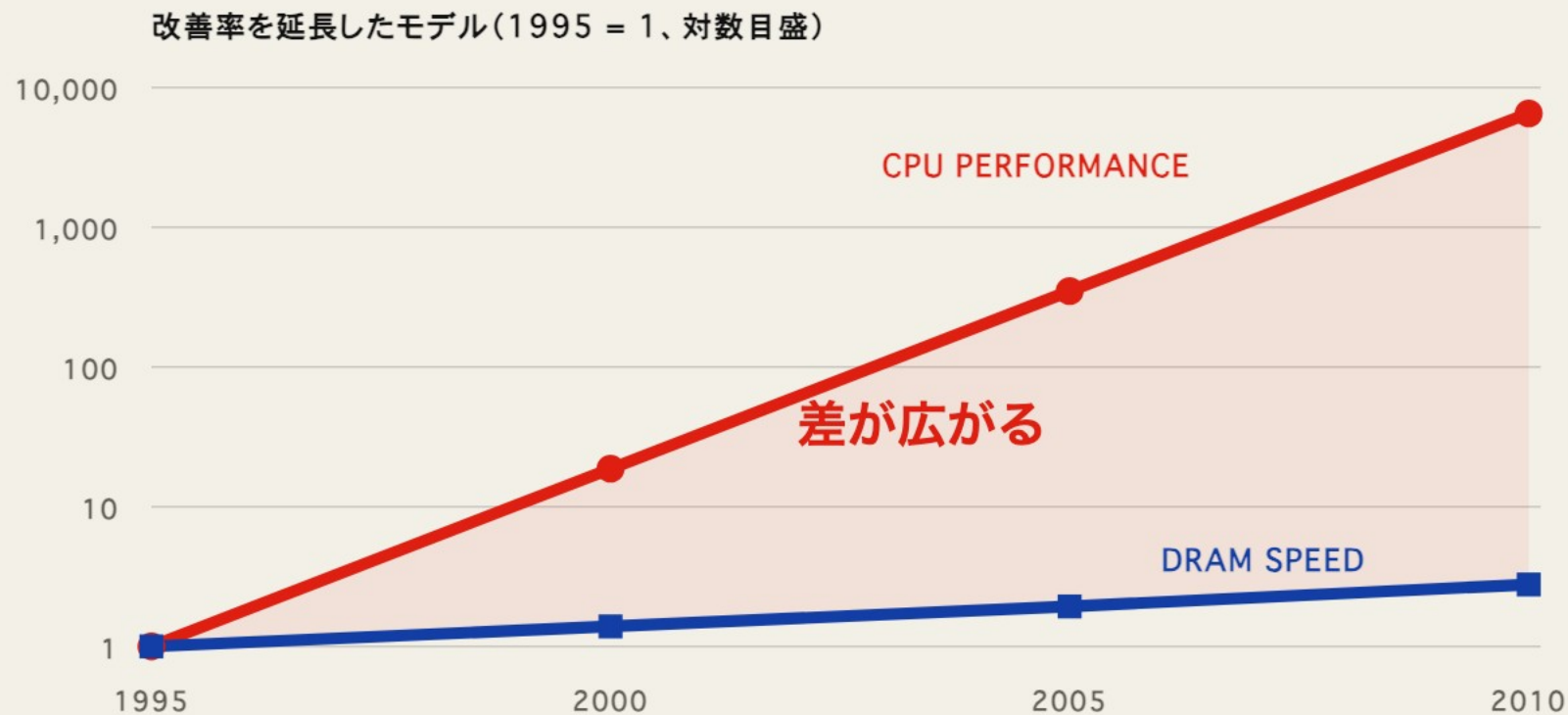
だから主記憶は CPU の近くに置かれる。

アプリを開くと、RAM に何が入る？



CPU は、RAM に置かれた命令とデータを何度も読み書きする。

CPU と RAM の速度差：メモリの壁



平均待ち(CPU周期)

2000 1.52

2005 8.25

2010 98.8

MEMORY WALL

計算できても、
データを待つ止まる。

Wulf & McKee(1995)のモデル: CPU性能 +80%/年、DRAM速度 +7%/年。実測推移ではない。

次に見るのは、ストレージ

CPU 内：レジスタ＝演算中の値



CPU 内：少量を高速に保持

ストレージ：大容量・電源断後も保持

HDD と SSD は、速さと使いどころが違う

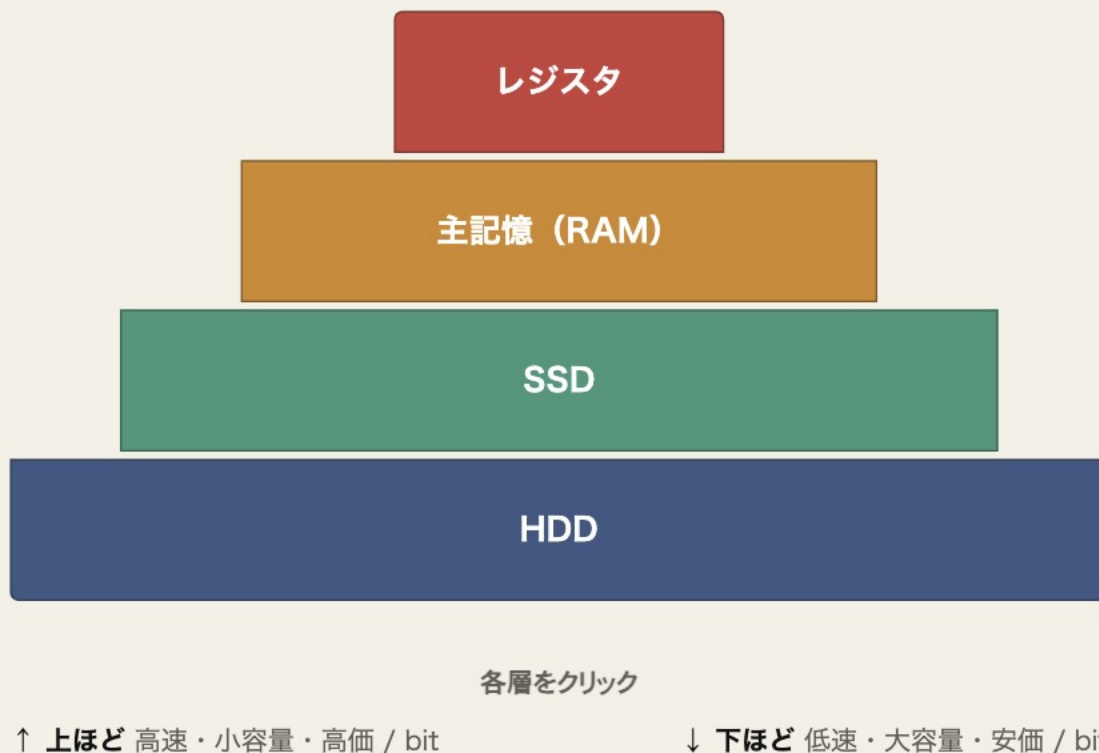
比較する点	HDD	SSD	
	円盤とヘッドを動かす	半導体へ電氣的に記録する	
連続読出し 大きなデータを 順に読む速さ	220 MB/s BarraCuda Pro の製品例	560 MB/s 870 EVO / SATA	7,450 MB/s 990 PRO / PCIe 4.0
細かい読出し	位置合わせ・回転待ちがある。 離れた場所を読むと待ちが増える。	機械的な待ちがない。 起動やアプリの読み込みに強い。	
価格と容量	容量あたり安い傾向。 同じ予算で大量に保存しやすい。	同じ容量では高めの傾向。 速いアクセスを重視する。	
衝撃・音	動作中の衝撃に弱い。回転音がある。	衝撃に比較的強い。駆動音がない。	
使い分け	大量の写真・動画などの保存	OS・アプリ・頻繁に使うファイル	

数値は製品別の公称最大値の例で、実測比較ではない。MB/s は 1 秒に転送する MB 数。

SATA・PCIe は接続方式の名前。速度は製品・接続方式・読み方によって変わる。

[内部のしくみに戻る](#)

記憶階層は、異なる性質の記憶を重ねる



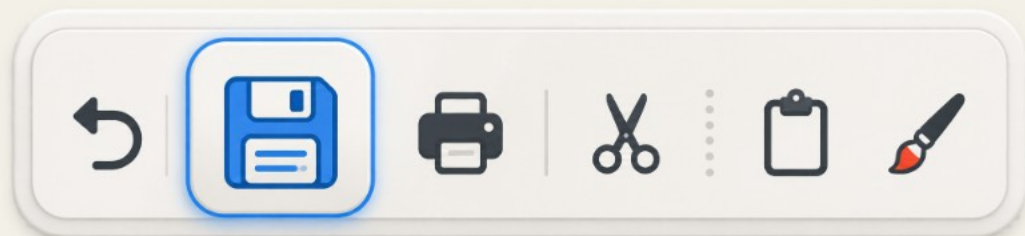
SELECT A LEVEL

左の階層を一つ選ぶ

容量、待ち時間、場所、
電源を切った後を比較する。

ピラミッドの幅は容量を表す。数値は代表的な目安であり、製品や世代によって異なる。

保存ボタンの形を、見たことがある？



REVEAL

フロッピー
ディスク

媒体が変わっても、その形は保存の記号として残っている。



最後に見るのは、入力装置と出力装置



I/O は、コンピュータ内部と外部の境界である。

入力と出力は、コンピュータから見た向き

INPUT

入力装置

操作をデータにして渡す



キーボード

キーを押す



マウス

動かす・クリックする

内部へ



コンピュータ

CPU・記憶装置
プログラム

外部へ



OUTPUT

出力装置

結果を人に伝える



ディスプレイ

文字や画像を映す



スピーカー

音を鳴らす



プリンター

紙に印刷する



身近な問い

タッチ画面は、入力？ 出力？

触れた位置を検出 → 入力

文字や画像を表示 → 出力

戻る

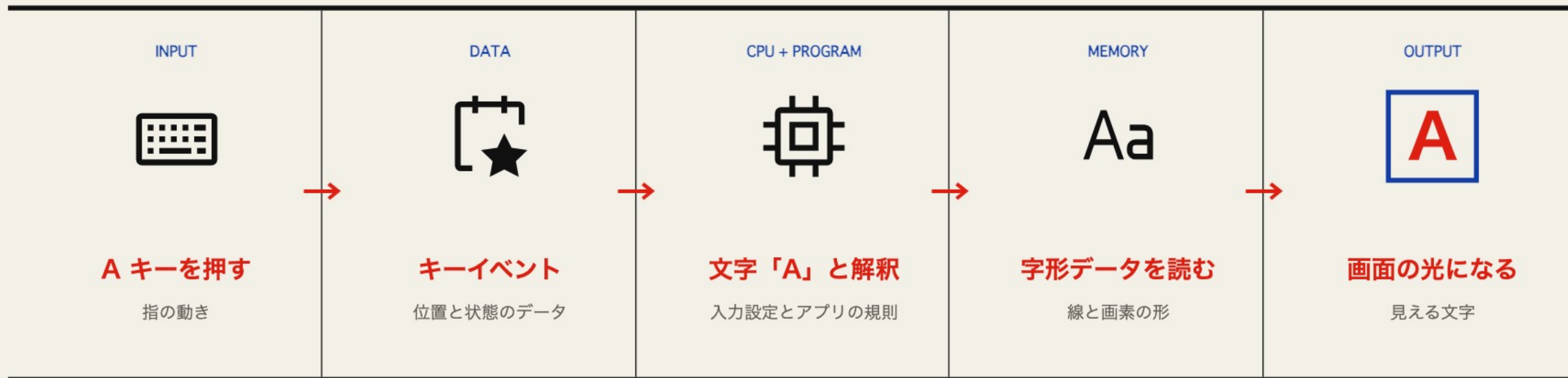
1 身近な入出力

2 タッチ画面の答え

次へ

「A」は、そのまま画面へ移動するわけではない

入力から出力まで、段階ごとに表現が変わる。



01 入力、02 記憶、03 制御、04 演算、05 出力が、一つの処理でつながる。

センサーも、入力装置の仲間

人の操作だけでなく、**光・音・温度・距離**もデータとして取り込める。

01 スマホのカメラ



測るもの

光

取り込むデータの例

画像データ

02 マイク



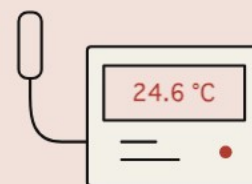
測るもの

音

取り込むデータの例

音声データ

03 温度センサー



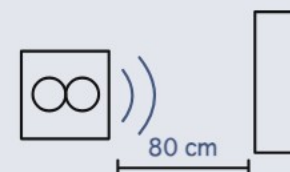
測るもの

温度

取り込むデータの例

24.6 °C

04 距離センサー



測るもの

距離

取り込むデータの例

80 cm

何を調べたい？

物体までの距離を測る。ロボットにも使われる。

調べたいことに合わせて、測る装置を選ぶ。

戻る

1 カメラ

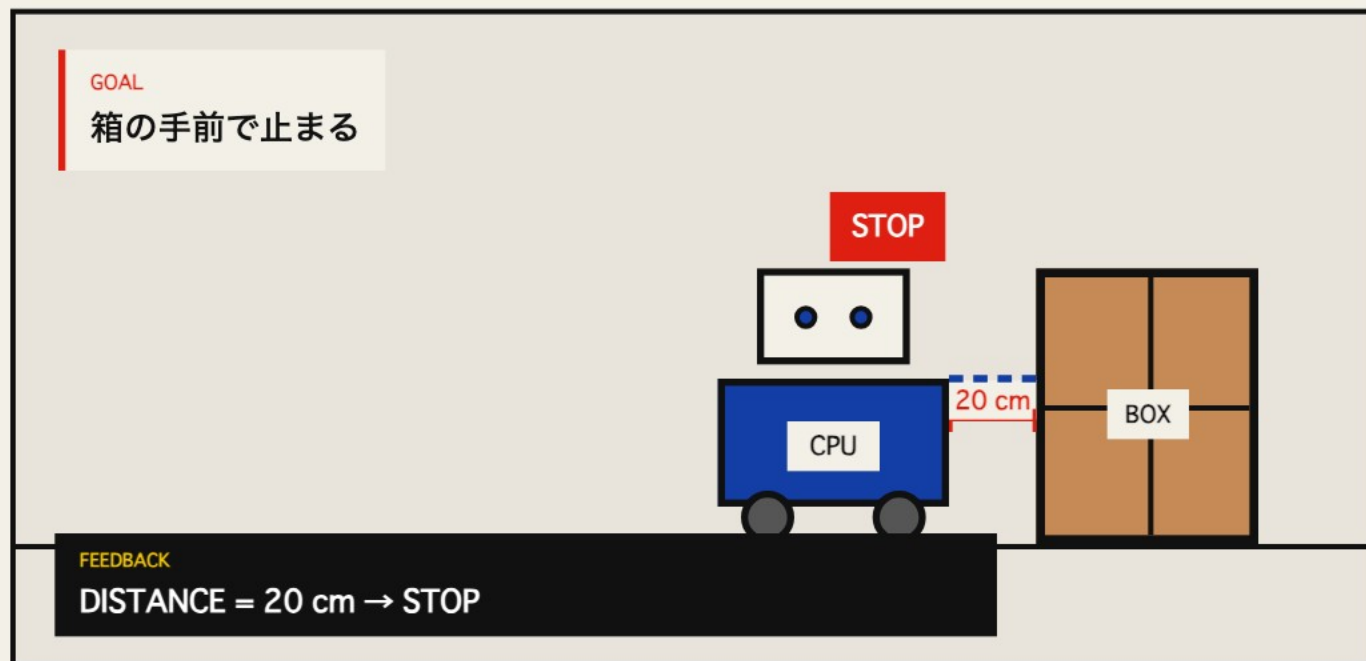
2 マイク

3 温度

4 距離

次へ

ロボットで、入力と出力をつなぐ



STEP 5 / 5

測り直して、止まる

距離をもう一度測り、20 cm でモーターを止める。

入力装置 **距離センサー**

判断する部分 **CPU + プログラム**

出力装置 **モーター**

結果を測り直して動きを調整する。これがフィードバック。

入力：測る



制御・演算：判断する



出力：動かす



結果を、もう一度測る

戻る

1 目標

2 入力

3 判断

4 出力

5 測り直す

次へ

用途が違えば、優先する部品も変わる

同じ予算なら、どの部品を優先する？ 理由を一つ添える。

A

資料もブラウザも、
同時に開いて作業する

いま使うデータが多い

RAM の容量

同時に使うデータを置く
容量を確保する。

B

写真や動画を、
大量に残しておきたい

保存するデータが多い

ストレージの容量

写真や動画が収まる
容量を確保する。

C

計算処理にかかる
時間を短くしたい

処理の仕事量が多い

用途に合う CPU

処理が CPU の性能に
制限されているか確認する。

部品名だけでなく、用途 → 必要な働き → 選ぶ理由の順で説明する。

戻る

1 まず考える

2 考え方を確認

次へ

実物で確かめ、用途に合わせて選ぶ

01 マザーボード探検



CPU と RAM の場所を探し、
形や端子を根拠に説明する。

[マザーボードの演習を開く ↗](#)

02 PC 組み立て

用途を決める

資料作成 / 写真保存 / 計算

部品を選ぶ

CPU + RAM + ストレージ

理由を説明する

予算と必要な働きを結び付ける

用途と予算を決め、
選んだ構成の理由を一つ説明する。

[PC 組み立てシミュレーターを開く ↗](#)

この全体を、どうやって 0 と 1 で表す？

次は、文字・数値・画像・音を、0 と 1 で表す方法を扱います。