# コンピュータ構成要素

・USB
Type-A,Type-Bの1.0,2.0では４本。3.0では９本。

・主記憶とキャッシュメモリの紐付け方法

ダイレクトマップ -> ハッシュ演算によって、主記憶のブロックとキャッシュメモリのブロックが1対1対応

フルアソシエイティブ -> 主記憶のブロックをキャッシュメモリのどのブロックにも格納可能

セットアソシエイティブ -> キャッシュメモリのブロックを連続した一定数ごとにまとめた″セット″を用意し、主記憶のアドレスから対応する″セット″が一位に定まり、そのセット内のブロックならどこでも格納できる方式。

ハミング符号は自動訂正機能がある(作った組み合わせから誤りを特定可能、水平パリティチェックは1bitまで)

メモリインターリープ -> 主記憶を複数の独立したグループに分けて、各グループに交互にアクセスすることで、主記憶へのアクセスの高速化

パイプライン処理 -> 処理をいくつかの段階に分けて、同時並行的に流れ作業のように実行する方式

CPUにおけるパイプライン
**フェッチ**（命令を読み取る）
**デコード**（命令を解釈する）
**実行**（命令を実行する）
**メモリアクセス**（データ読み書き）
**書き戻し**（結果をレジスタに書く）

**パイプラインハザード**（CPU分野の用語）
- 例：分岐命令で次の命令が予測外 → パイプラインがストール（停止）

CISC(Complex Instruction Set Computer )
CISCは「複雑な命令セット」を持ち、1命令で色々な処理を行える。　・　でも命令ごとに処理時間がバラバラで、ハードウェアも複雑になりがち。
CISCはRISCの逆で、
命令セットは多く
命令実行時間は命令によってバラバラ
実装はハードが複雑でソフトは簡単め

RISC(Reduced Instruction Set Computer)
RISCは命令をできるだけ単純化し、**1クロックで実行できる命令を基本にする。**
複雑な処理は「単純な命令を組み合わせて」ソフトウェア側（コンパイラなど）で実現する。

1. **シンプルな命令セット**
    - 命令の種類を減らし、形式を統一する。
    - 各命令がほぼ同じ時間（クロック数）で実行可能。
2. **ロード／ストア型**
    - メモリに直接アクセスするのは「ロード（読み出し）」と「ストア（書き込み）」だけ。
    - 演算命令はレジスタ間で行う。
3. **パイプライン処理に最適化**
    - 命令が規則的で単純なので、パイプライン化して並列実行しやすい。
4. **ハードウェアを単純化**
    - 命令デコーダなどが簡単になり、電力効率がよい。

Write back ライトバック
CPUから書き込み命令があった時に、キャッシュメモリにのみ書き込む。キャッシュからデータが追い出されたら主記憶へ書き込む。

Write through ライトスルー
CPUから書き込み命令があったら、主記憶、キャッシュどちらにも書き込む。データの生合成が取れるが遅い。

メモリ系まとめ

## 1. SRAM（Static RAM）

- **方式**：フリップフロップ回路でデータを保持（リフレッシュ不要）
- **速度**：速い
- **揮発性**：あり（電源切ると消える）
- **コスト**：高い、集積度低い
- **用途**：CPUのキャッシュメモリ（L1, L2, L3）

## 2. DRAM（Dynamic RAM）

- **方式**：コンデンサに電荷を蓄えて保持（リフレッシュ必要）
- **速度**：遅い（SRAMより）
- **揮発性**：あり
- **コスト**：安い、集積度高い
- **用途**：主記憶（メインメモリ）

## 3. ROM（Read Only Memory）

- **方式**：基本的に読み出し専用（書き換え不可 or 制限あり）
- **揮発性**：なし（電源切っても内容保持）
- **用途**：ファームウェアやBIOS格納

👉 種類

- **MASK ROM**：工場出荷時に書き込み固定
- **PROM**：ユーザーが一度だけ書き込み可能
- **EPROM**：紫外線で消去、再書き込み可能
- **EEPROM**：電気的に消去・書き込み可能（遅い）
- **フラッシュメモリ**：EEPROMをブロック単位で高速書き換え可能 → USBメモリ、

SSD に利用

### 4. キャッシュメモリ

- **方式**：SRAM を利用
- **特徴**：CPU と DRAM の速度差を埋めるための超高速メモリ
- **階層**：L1（CPU コア内）、L2、L3 と多段階

### 5. VRAM（Video RAM）

- **方式**：DRAM 系
- **用途**：ディスプレイのフレームバッファ（画面表示用）
- 今は GDDR（Graphics DDR）として GPU 用に進化

### 6. フラッシュメモリ

- **方式**：EEPROM の一種（ブロック単位で書き換え可能）
- **揮発性**：なし
- **用途**：SSD、USB メモリ、SD カード、スマホのストレージ

| 種類 | 記憶方式 | 揮発性 | 速度 | コスト | 主な用途 |
|---|---|---|---|---|---|
| **SRAM** | フリップフロップ | あり | ◎ 高速 | × 高価 | CPU キャッシュ |
| **DRAM** | コンデンサ | あり | ○ 中速 | ○ 安価 | 主記憶（メインメモリ） |
| **ROM** | 固定 | なし | △ 遅い | ○ | BIOS, 組込機器 |
| **EPROM** | 紫外線消去型 | なし | △ | ○ | 開発用（古い） |
| **EEPROM** | 電気消去型 | なし | △ 遅い | ○ | IC カード, 設定保存 |
| **フラッシュ** | ブロック消去型 | なし | ○ 速い | ○ | SSD, USB, スマホ |
| **VRAM** | DRAM 派生 | あり | ○ | ○ | グラフィック用 |

HDD のサーチ時間

HDD のデータにアクセスするのにかかる時間は、だいたい次の合計：

### アクセス時間 = シークタイム + 回転待ち時間（サーチタイム）+ 転送時間

- **シークタイム**：ヘッド移動（数ミリ秒〜十数ミリ秒）

- **回転待ち時間（サーチタイム）**：回転速度依存（7200rpmなら平均約4.2ms）ディスクが1回転する時間の半分
- **転送時間**：実際にデータを読み書きする時間（ナノ秒〜マイクロ秒レベルで短い）1トラックが1回転。

ベクトルコンピュータでは、浮動小数点をよく使うので、

CRT(ブラウン管) ディスプレイ
ブラウン管を使ったディスプレイ。奥行きもあり、広い設置面積が必要。消費電力大。
電子ビームが発光体に衝突して生じる光を使っている。

液晶ディスプレイ
電圧によって液晶を制御し、バックライトもしくは外部からの光を取り込むことで表示する仕組みのディスプレイ。

有機ELディスプレイ
有機化合物 (発光素子) に電圧を加えることで発光する仕組みを利用する。バックライトが不要で理論上は省電力。

プラズマディスプレイ
プラズマ放電による発光を利用する。高電圧が必要。

プロセッサのアーキテクチャ
SISD 単一の命令で単一のデータを処理
SIMD 単一の命令で複数のデータを処理
MISD 複数の命令で単一のデータを処理
MIMD 複数の命令で複数のデータを処理

アドレス指定方式

直接アドレス方式
アドレス部の値をそのまま有効アドレスに

間接アドレス指定
アドレス部の値で主記憶上のアドレスを指定し、そのアドレスに格納されている値を有効アドレスとする。

指標アドレス指定
アドレス部の値にインデックスレジスタの値を加えたもの

基底アドレス指定
アドレス部の値にベースレジスタの値を加えたもの

相対アドレス指定
アドレス部の値にプログラムカウンタの値を加えたもの

即値アドレス指定

アドレス部の値を有効アドレスではなく、そのまま演算対象データとする。

ヘテロジニアスマルチプロセッサ
1つのチップに　異なる種類のCPUコアを積む

ホモジニアスマルチプロセッサ
1つのチップに 同じ種類のCPUコアを積む

システムバス

コンピュータ内部で **CPU・メモリ・入出力装置（I/O）** をつなぐ共通の通信路のこと
いわば **「部品同士のデータをやり取りするための道路」** みたいな存在

VLIW (Very Long Instruction Word)
プログラムのコンパイル時に依存関係の多数の命令を1つの複合命令にまとめる

**従来の「分離メモリ方式」**

- **CPU用メインメモリ（DRAM）** と **GPU用専用メモリ（VRAM, GDDR）** が別々
- データをCPUで処理 → GPUで使うときは「メインメモリ →VRAM」にコピーする必要がある
- 高性能GPU（NVIDIA GeForce, AMD Radeon などPC向け）はこの方式

**ユニファイドメモリ方式**

- CPUとGPUが **同じメモリ空間を使う**
- データコピー不要で、同じデータを両方が直接参照できる
- スマホや組込み機器、Apple M1/M2チップ、APU（AMD）などが代表例