はじめに

- 今回と次回の2回に分けて、メモリアーキテクチャについて講義します。

- まず、最初のテーマに対する教科書の対応範囲は、以下の通りです。
  - 7.1.1項(a)（p.222）
  - 7.1.2項（p.224～226）
  - 7.1.3項（p.226～229）

メモリの機能

- メモリ（特にメインメモリ）の機能については、第1回目の講義で解説しました。
  - いまさら付け加えることはありませんが、一応、教科書の7.1.1項(a)（p.222）の内容を再確認しておいてください。

メモリ装置の分類

- 教科書の7.1.2項(a)（p.224～225）を読んでください。
  - 教科書では、ICメモリ（半導体メモリ）だけでなく、広く記憶装置について、その特徴の観点から分類しています。
  - 特に頑張って暗記するような事項ではありませんが、逆に、当たり前に知っていて当然の内容です。そのつもりで理解しておいてください。
  - 本講義では、補足（追加説明）する点についてのみ、次ページ以降でコメントします。

- 揮発性
  - 電源の供給を止めると記憶内容を“忘れて”しまうメモリ装置を「揮発性のある（volatile）メモリ」と言います。
    - “忘れた”状態とは、全ビットの記憶内容がすべて1または0のどちらかになってしまう状態です。
  - 電源の供給を止めても記憶内容が失われないメモリ装置は「不揮発性の（non-volatile）メモリ」と言います。

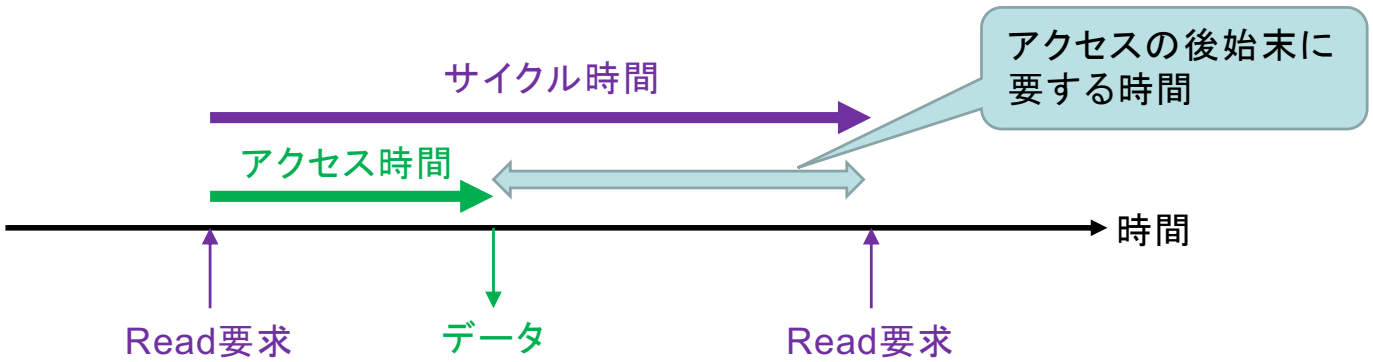
- RAM
  - もともと、半導体メモリはランダムアクセス（random access）が可能なメモリです。
    - 「ランダムアクセスが可能」とは、「アクセスに要する時間が、記憶場所によらずに一定時間でアクセスできる」ことを指します。
      - 例えば、何番地のデータでも、同じ時間で読み出すことができます。
      - 紙テープや磁気テープの場合は、途中から読み出したい場合は、最初の部分を読み飛ばさなければならず、読み出したい場所が後の方になるほど、読み飛ばしに時間がかかります。このように、記憶されている順にアクセスする性質を持つメモリ装置をシーケンシャルアクセス（sequential access）メモリと言います。
    - つまり、半導体メモリはRAM（Random Access Memory）なのです。

- ところが、読み出し専用のメモリがROM（Read-Only Memory）と呼ばれるようになっても、書き込みもできる半導体メモリには特別な名称が与えられず、RAMと呼ばれたままの状態が続いています。
  - そのため、読み出し専用のメモリはROM、読み出しも書き込みもできるメモリはRAM、という妙な術語の使い分けがされています。
  - もちろん、ROMもランダムアクセス可能なメモリです。

半導体メモリ素子の性能指標

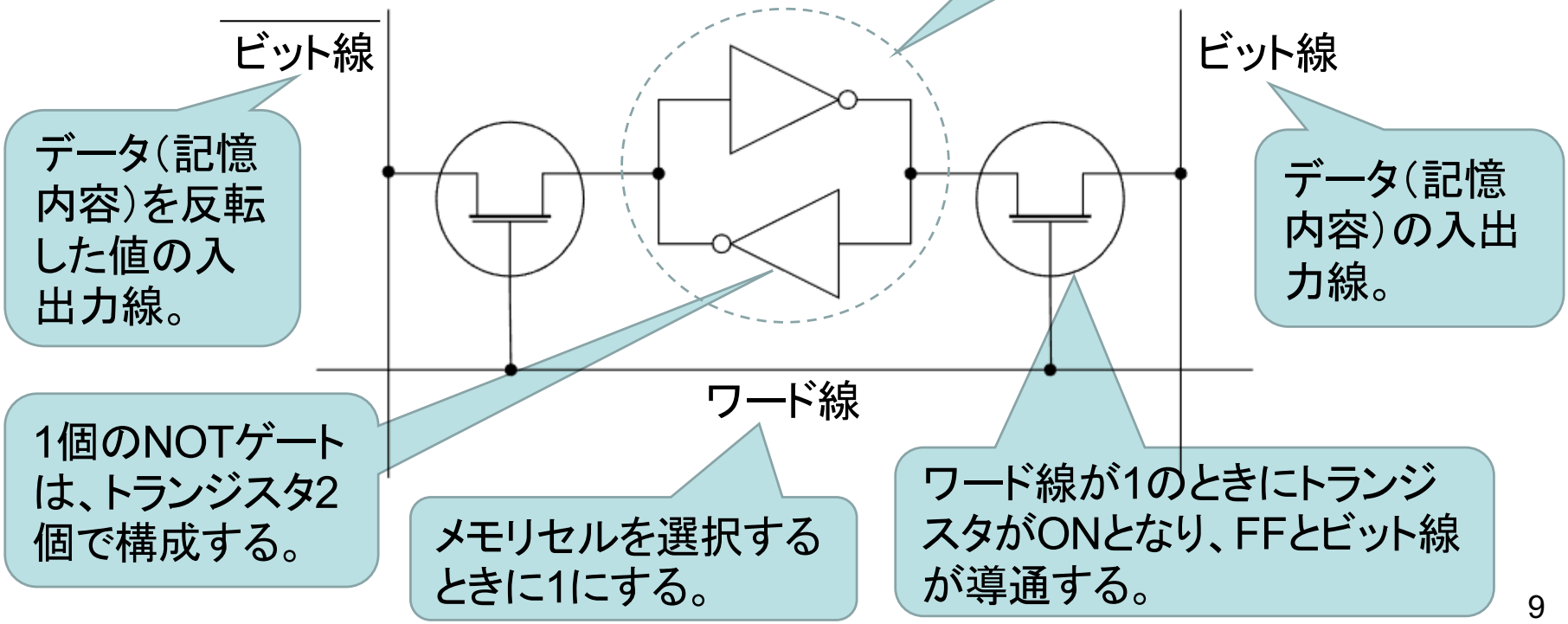
- 容量
  - たくさん記憶できた方が良いに決まっています。
    - そのためには、LSIチップ上で、1ビットを記憶するセル（cell）の面積が小さい方がよい、ということになります。
  - アドレスビットとの関係で、大抵は、容量は、2のべき乗のビット数になっています。

- 速度
  - メモリアクセスに要する時間は短い方が望ましい。
  - 以下の2種類の時間を考慮しなければなりません。
    - アクセス時間
      - Read要求を行ってから、データ（記憶内容）が出力されるまでの時間。または、Write要求を行ってから、記憶内容の更新が完了するまでの時間。
    - サイクル時間
      - アクセス（ReadまたはWrite）要求を行ってから、次のアクセス要求を行えるようになるまでの時間。

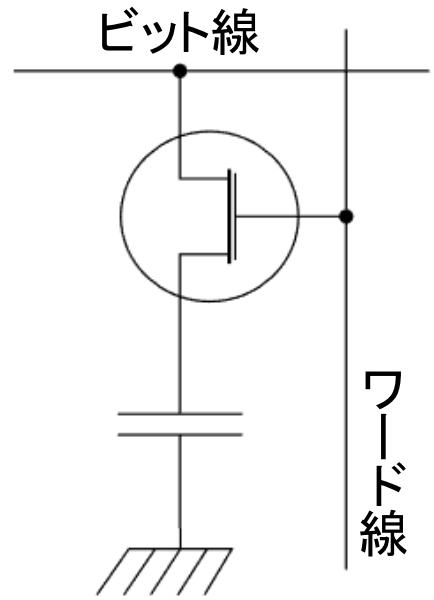


SRAMとDRAM

- SRAM（Static RAM）
  - フリップフロップ（FF）で記憶。
  - メモリセル1個に計6個のトランジスタを要する。



- DRAM（Dynamic RAM）
  - コンデンサの充電/放電で1/0を記憶。
  - メモリセル1個分の面積はトランジスタ1個分＋α。
    - α：コンデンサの面積はトランジスタよりも小さい。
    - SRAMよりもメモリセルの面積が小さく、SRAMより大容量化が図れる。
  - データを読み出すと放電してしまう。つまり、記憶内容が消えてしまうので、読み出した値を再び書き込む作業が必要。
    - その時間が「半導体メモリの性能指標」で述べた「アクセスの後始末に要する時間」にあたる。
    - SRAMにはこのような時間は不要で、したがって、SRAMの方が高速にアクセスできる。



- また、一定時間が経過すると自然放電により記憶データが消えてしまう。
  - そのため、定期的に記憶データを読み出して、再度書き込みを行う必要があり、この動作を「リフレッシュ（refresh）」という。
  - このようなリフレッシュが必要なメモリをダイナミックメモリと呼び、これがDRAMの由来でもある。
  - 通常、アドレスを巡回するようにして、リフレッシュ動作を行う。
    - 現在のコンピュータでは、メインメモリにDRAMを用いている。そのため、リフレッシュを行う専用のメモリコントローラも設置している。
  - SRAMにはリフレッシュは必要ない。

- 以上をまとめると、
  - 速度はSRAMの方が速い。
  - 容量はDRAMの方が大きい。
    - 1セルに要する面積がDRAMの方が小さいので、LSIの同一の面積で比較すると、DRAMの方が多くのビットを記憶できる（つまり、集積度が高い）。
- 「速くて大容量のメモリが欲しい！」のですが、現実には、以上のように速度と容量が両立しません。

- ハードディスク装置（HDD: Hard Disk Drive）だともっと大容量になりますが、速度はもっと遅くなります。
  - 磁性体でビットの値を記憶します。
  - 円盤を回転させて読み書きしますので、半導体メモリで電荷が移動する速度とは比べものになりません。

- 一般に、容量が大きいものは遅く、速いものは小容量です。しかし、欲しいのは「速くて大容量のメモリ」です。
いったい、どうすれは良いのでしょうか？

メモリ階層

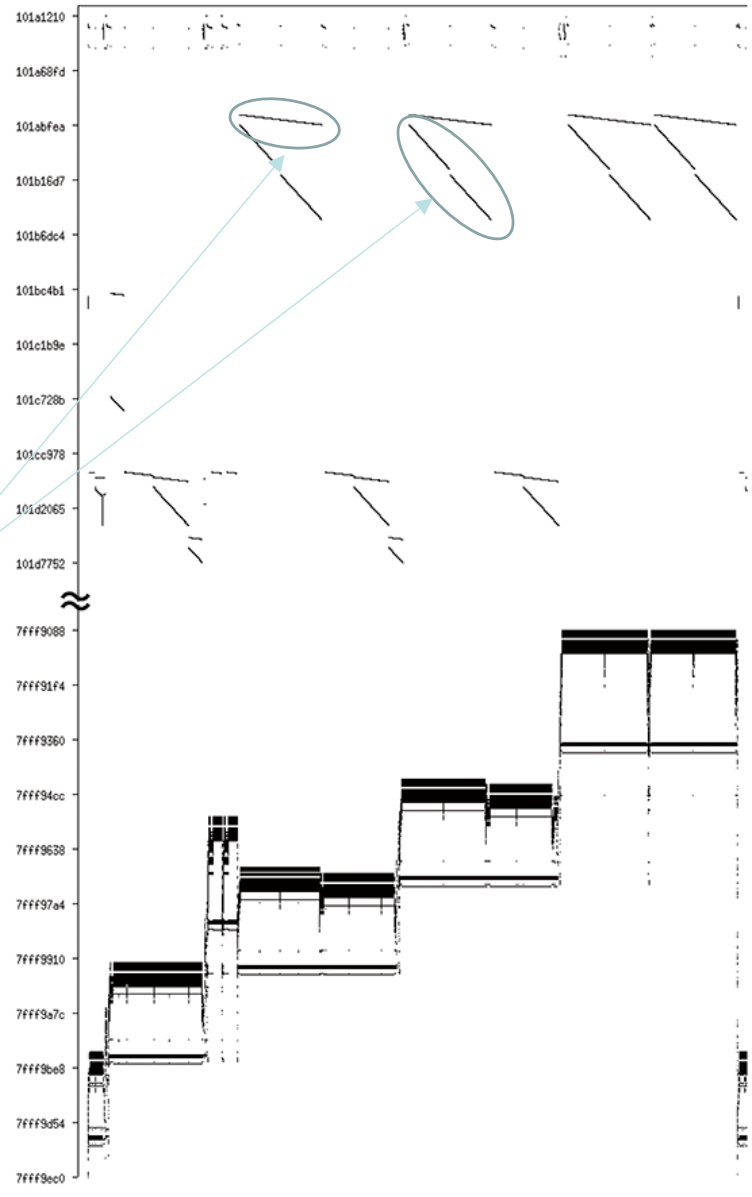
- 繰り返します。容量が大きい記憶装置は速度が遅く、速度が速い記憶装置は小容量です。
- この状況で、速くて大容量のメモリシステムを実現しなければなりません。

- そこで、まず、目を付けたのが、「参照の局所性（locality of references, referential locality）」という、プログラム実行時に一般的に観察される傾向です。
  - あくまでもそういう傾向が見られる、というだけのことです。必ず備えているという性質や原理でもなければ、規則や制約条件でもありません。

参照の局所性

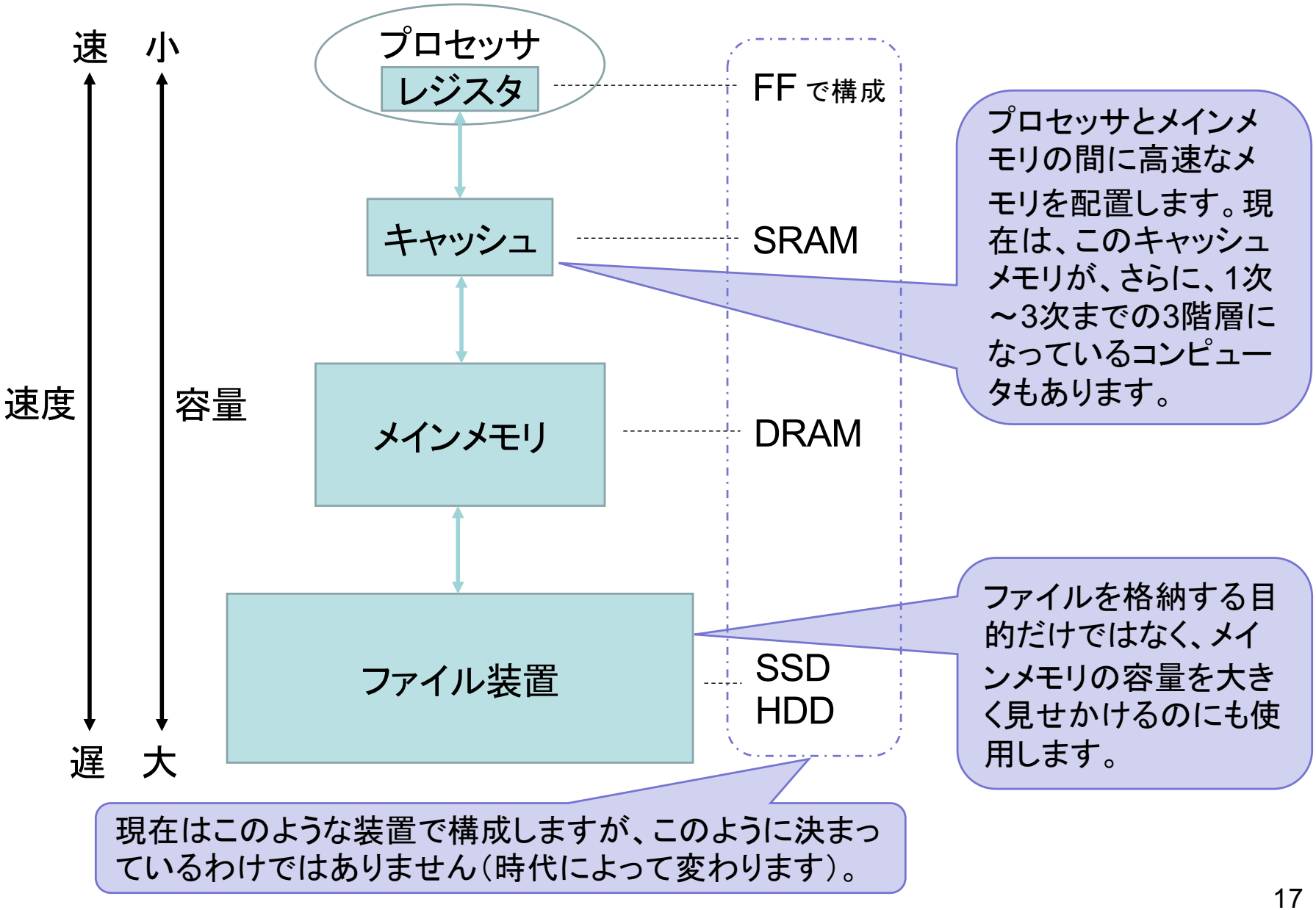
- プログラムでメインメモリにアクセスする場合の一般的かつ特徴的なある傾向で、
  - 一度アクセスされたロケーション（アドレス）は、近い将来に再びアクセスされる可能性が高い。（時間的局所性）
  - 一度アクセスされたロケーション（アドレス）の近傍は、近い将来にアクセスされる可能性が高い。（空間的局所性）
- 以上の傾向が見られるかどうかはプログラムの作り方によりますが、程度の違いはあっても、大抵は上記の傾向が見られます。
  - 逆に、参照の局所性が全くないプログラムを作る方が困難です。
  - なお、この参照の局所性を利用してコンピュータシステムを設計していますので、プログラムの参照の局所性を高めることは、そのプログラムの実行時間を短縮することにつながります。

- 右の図は、C言語で書いたあるプログラムを実行したときのメモリアクセスの様子を示したものです。
  - 横軸が時刻です。
  - 縦軸はメモリのアドレスです。
  - アクセスした場所（アドレス）に点を打ちました。
  - 上半分はデータ領域（セグメント）へのアクセスの様子です。斜めの線は配列のアクセスによるものです。空間的局所性が示されています。
  - 下半分はスタックフレーム領域（セグメント）へのアクセスの様子です。関数内で定義したオート変数（その場所は、関数呼び出し毎に変化します）に頻繁にアクセスしている様子が読み取れます。



メモリ階層（つづき）

- 参照の局所性を利用して、大容量化と高速化を両立します。
  - 頻繁にアクセスするデータ（命令語）は、速くて小容量のメモリに記憶します。
  - あまりアクセスしないデータ（命令語）は、遅くて大容量のメモリに記憶します。
  - 以上のように、速くて小容量のメモリと遅くて大容量のメモリとを組み合わせて、メモリ階層（memory hierarchy）を構成します。


速
小
速度
容量
遅
大
プロセッサ
レジスタ
キャッシュ
メインメモリ
ファイル装置
FF で構成
SRAM
DRAM
SSD
HDD
プロセッサとメインメモリの間に高速なメモリを配置します。現在は、このキャッシュメモリが、さらに、1次～3次までの3階層になっているコンピュータもあります。
ファイルを格納する目的だけではなく、メインメモリの容量を大きく見せかけるのにも使用します。
現在はこのような装置で構成しますが、このように決まっているわけではありません（時代によって変わります）。

- メインメモリの高速化
  - ⇨キャッシュ（cache）メモリ
    - 頻繁に使用する命令・データは小容量の高速メモリに！
    - 基本的に、ハードウェア機構のみで制御します。
      - OSは初期設定ぐらいは行いますが、OS自身もキャッシュのお世話になります（キャッシュによって高速化されています）。
      - cacheという単語の意味は「隠し場所」。つまり、ソフトウェアからは見えない（制御できない）のです。

- メインメモリの大容量化
  - ⇨仮想メモリ（virtual memory / virtual storage）
    - 実行中のプログラム（プロセス）の全ての情報をメインメモリに置くのが基本ですが、実際には、その大きなプロセス空間の情報をファイル装置に記憶して、そのうち使用する領域のみをメインメモリに置きます。
    - ハードウェア機構とOSで協調して制御します。