オペレーティングシステム(OS)

柴山　潔

11. 仮想メモリ (1)

- 領域の管理
- 仮想メモリとは？

OSによるメモリ管理

- メモリ（＝メインメモリ＋ファイル装置）の空間的管理　＝ メモリ領域の管理

(A) メインメモリ（実メモリ）の管理

（例）プロセスの割り付け，アクセス権限のチェック（＝メモリ保護）

(B) 仮想メモリの管理

（例）メインメモリとファイル装置との対応付け

(C) ファイルとファイル装置の管理

（例）ファイル割り付け，ファイル保護

領域管理の実際

■ 領域（＝そこでの保持対象）の実際

(A) メインメモリ上のプログラム or プロセス
- 動的（＝実行時）にしか決定できない要素や属性がほとんど

(B) ファイル装置上のファイル
- 静的（＝実行前）にしか決定できない要素や属性がほとんど

■ 具体的な機能

(a) 割り付け（＝アロケーション(allocation)）：ソフトウェア（例：プロセス，プログラム，ファイル）が使用するメモリ領域を確保，使用可能に
- OS（＝領域管理者）が，メモリ装置（特に，メインメモリとファイル装置）上で，ソフトウェアを保持

(b) 解放（＝リリース(release)）：確保・使用しているメモリ領域を未使用（状態）に
- OS（＝領域管理者）に返却 or ゴミ集め（＝未使用メモリ領域を収集&割り付け可能状態に，ガーベジコレクション(garbage collection)）の対象へ

領域管理(割り付け)方式の分類 (1) —固定長/可変長—

■ 対象:(A)のメインメモリ上のプログラム or プロセス
➢ 割り付け単位(サイズ)を固定長/可変長のどちらにするか?

(A-a) 固定長領域割り付け :割り付け単位が一定・固定

(A-b) 可変長領域割り付け :割り付け単位が自由・可変

領域管理(割り付け)方式の分類 (2) —連続/不連続—

- 対象:**(B)**のファイル装置上のファイル
  - 割り付け単位を連続/不連続のどちらにするか?

(B-a) 連続領域割り付け:論理的割り付け単位を物理的に(=ハードウェア装置上で)連続に割り付け

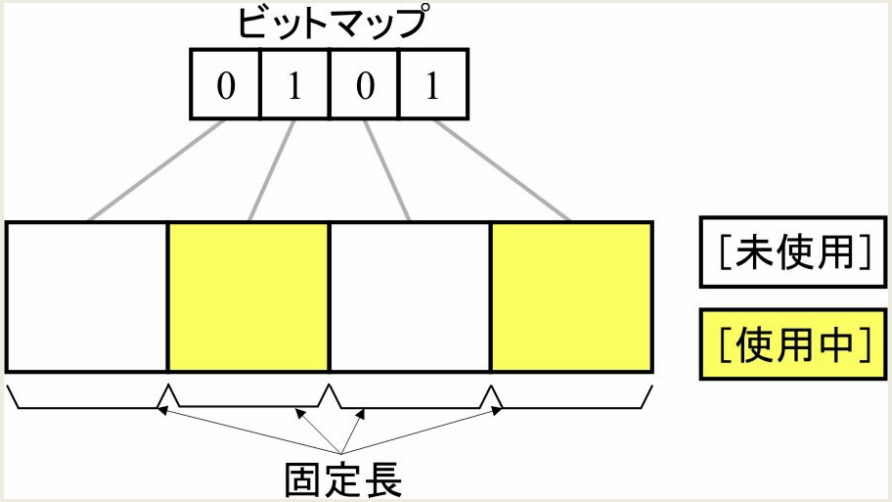
(B-b) 不連続領域割り付け:論理的割り付け単位を物理的に(=ハードウェア装置上で)不連続に割り付け

固定長領域割り付け —例：ビットマップ(bitmap)法—

- 固定長（サイズ）の領域ごとに，1ビットフラグ（論理値，"0"：空き/未使用，"1"：割り付け済み/使用中）を充てて管理

［特徴］

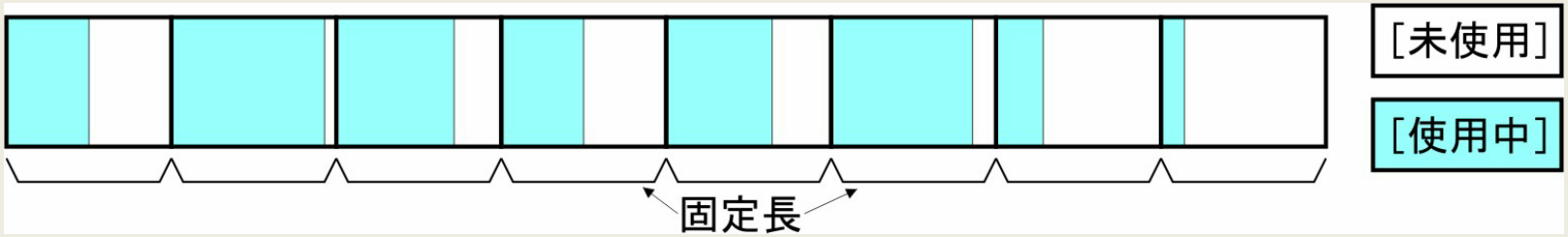
- ○ 管理が簡単 → 割り付け/解放時間が一定&高速 → リアルタイムOS（＝即時に/一定時間内に応答･処理するリアルタイムシステム専用OS）向き
- × 対象全体が可変長領域の管理では，複雑な領域分割や管理が必要 → 最適化が難
- × 内部フラグメンテーション（→参考3.3）が不可避



（参考3.3）

内部フラグメンテーション(fragmentation)

- 領域の割り付け/解放を繰り返すうちに，各領域の内部に「割り付け不可能な未使用領域の断片/小片」という多数の無駄領域が発生（＝フラグメンテーション(fragmentation)，領域の断片化や小片化）

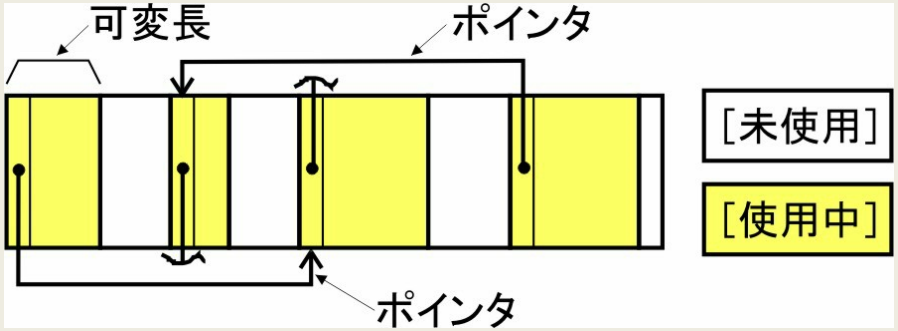


可変長領域割り付け —例：リスト(list)法—

- 可変長の領域をそれに付加したヘッダ（＝先頭部分の見出し情報，領域サイズ&次連結領域へのポインタで構成）で管理

[特徴]

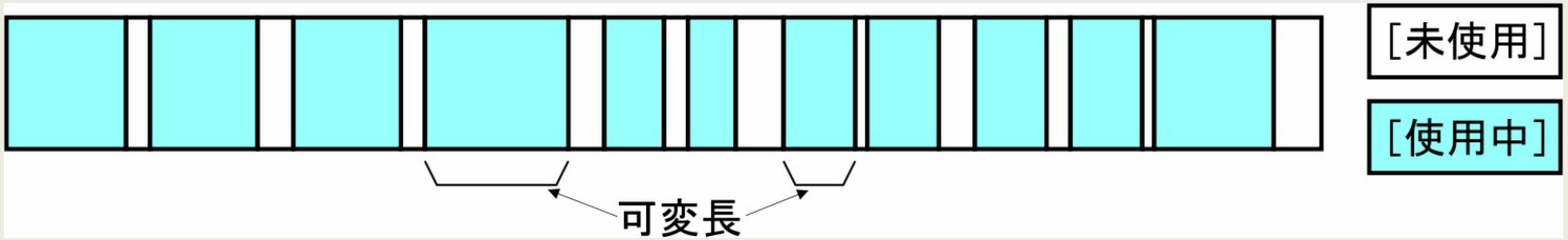
- ○ 対象が固定長/可変長領域のどちらでも管理方法が同じ
- × 割り付け/解放処理が複雑 → 遅い
- × 外部フラグメンテーション（→参考3.4）が不可避



（参考3.4）

外部フラグメンテーション(fragmentation)

- 領域の割り付け/解放を繰り返すうちに，割り付け可能な未使用領域 が小片化/断片化
  - 外部フラグメンテーションが発生 → 不連続で効率の悪い割り付けに → 領域管理（特に，割り付けと解放）が遅
  - 外部フラグメンテーションの解消 →「小片化/断片化した領域の収集&詰め直し」（＝デフラグ(de-fragmentation)）機能が必須



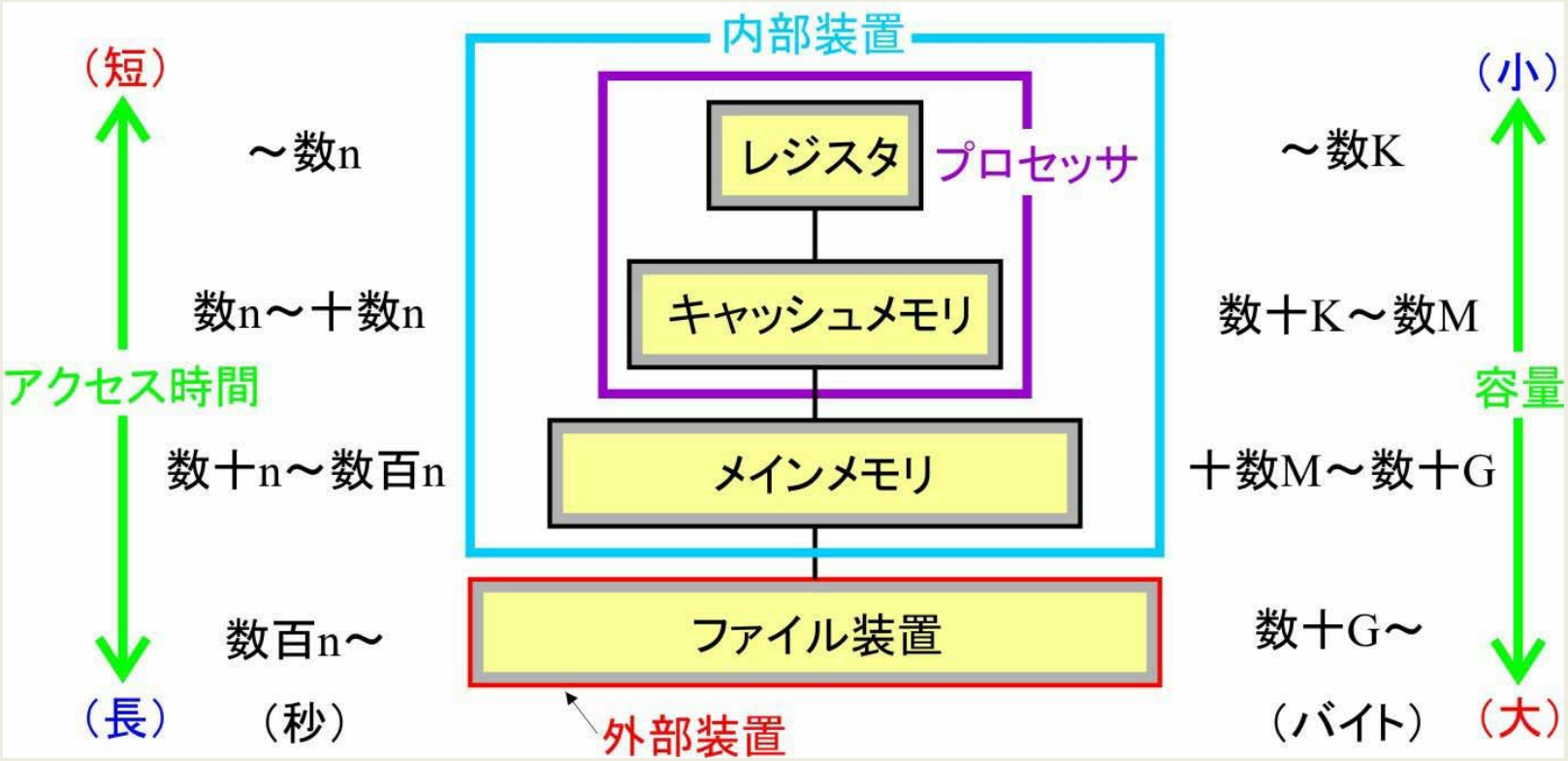
メモリ階層

［定義3.1］ **メモリ階層**

- 容量とアクセス時間とのトレードオフ（両立しない関係）によって識別できるメモリ機能やそれを実現するメモリ装置の種類のそれぞれ

  - メモリ階層間のトレードオフ（→主として，容量とアクセス時間）によって種々のメモリ装置の種類と機能や特性を実現
  - メモリ装置の構成や設定時に，「トレードオフとなる容量とアクセス時間の各特性によるメモリ装置の使い分け（適材適所を図る）指標」として利用

主要なメモリ階層（図）



主要なメモリ階層

(A) メインメモリ：容量とアクセス時間のいずれの指標でもバランスのとれた性能を示す中心的なメモリ階層

(B) ファイル装置：容量の指標でメインメモリよりも優，アクセス時間の指標でメインメモリよりも劣

(C) レジスタ，キャッシュメモリ：どちらのメモリ階層もプロセッサ内に実装 → アクセス時間の指標でメインメモリよりも相対的に優，容量の指標でメインメモリよりも相対的に劣

メインメモリ（main memory; 主メモリ）

- プロセッサと対になる主要な内部装置　＝ 内部メモリ, 1次メモリ
  - 生きているプロセス（＝プロセッサが実行中の命令や使用中のデータ）を保持

- 仮想メモリ機構の直接の対象となる物理的・実際のメモリそのもの（＝実メモリ）であり，OSがプロセッサやユーザプログラムなどから隠ぺい

- プロセッサが直接アクセス可
  - → ファイル装置と相対的に，格納機能よりもアクセス機能を重視，小容量でも高速性を必要

◆ 現代の代表的なメインメモリはICメモリの(D)RAM((Dynamic) Random Access Memory)で構成

ファイル装置

- ユーザが直接に操作(例:生成/消去/併合/分割/編集など)するファイルを格納する外部装置
  ＝ 補助メモリ, 外部メモリ, 2次メモリ
  - プロセッサが使用中でない(＝まだプロセスになっていない)プログラムやデータも格納

  - 仮想メモリ機構では，メインメモリのバックアップ(退避)用メモリとして機能し，論理的・仮想のメモリ(＝仮想メモリ)を構成するのに使用

  - プロセッサからは外部装置/周辺装置という位置付けで，プロセッサからはメインメモリをいったん経由する間接アクセス
    → メインメモリと相対的に，アクセス機能よりも格納機能を重視，低速でも大容量を必要

◆ 現代の代表的なファイル装置はハードディスクドライブ(hard disk drive; HDD)装置

参照局所性 (1)

［定義3.2］ **参照局所性**

- あるプログラム（実際には，マシン命令列）がアクセス/参照する命令 やデータのアドレス（番地，格納場所）は一部あるいは特定の個所に集中 ＝「参照局所性がある」，「参照局所性が高い」

- ほとんどの命令やデータは空間的参照局所性と時間的参照局所性とを併示
- 実行プログラムや適用問題ごとに参照局所性の傾向や性質は相違

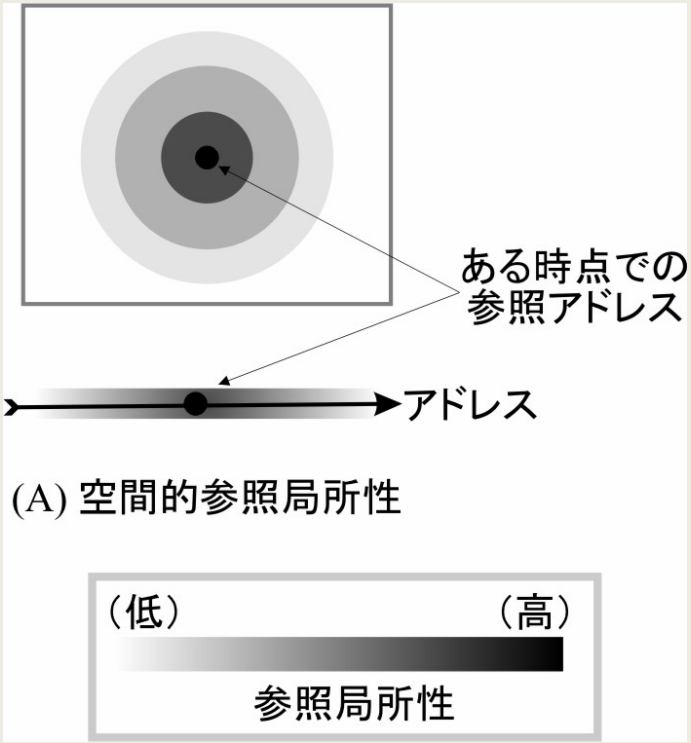
■ **OS**によるメモリ管理機能の設計

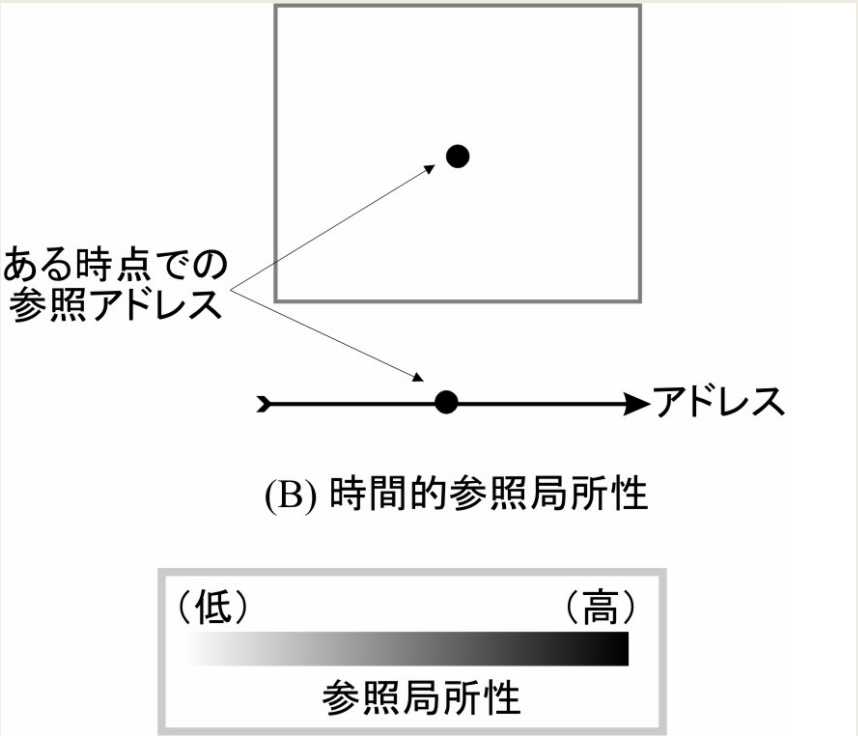
- 「メモリ階層の管理における“参照局所性”の活用」が要点
- OSは，隣接メモリ階層の特性を利用すれば，参照局所性が高いプログラム/プロセスを，当該メモリ階層において空間的/時間的に効率良く管理・処理できるようになり，結果として，当該メモリ階層そのものの機能を改善可

参照局所性 (2) —局所性を示す対象による分類—

(A) **空間的参照局所性**: 一度アクセスしたアドレスに近接する（格納場所が近い）アドレスは近いうちにアクセスする可能性が高

(B) **時間的参照局所性**: 一度アクセスしたアドレスそのものは近いうちに（ 例：同じプログラムの実行中に）またアクセスする可能性が高





アドレス空間

［定義3.3］ **アドレス空間**

- メモリにおけるアクセス可能な（＝アクセス対象となる）アドレス
  - 「アドレスが振ってある領域/空間」という意味

アドレス空間の大きさ ＝ メモリ領域のサイズ ＝ 容量

メモリ階層によるメインメモリ性能の改善
ー代表例：仮想メモリ（概要）ー

- 隣接メモリ階層であるファイル装置を利用して，メインメモリ性能の空間的改善を目標
- プロセッサから見えるメインメモリのアドレス空間（定義3.3）という空間的制限を撤廃

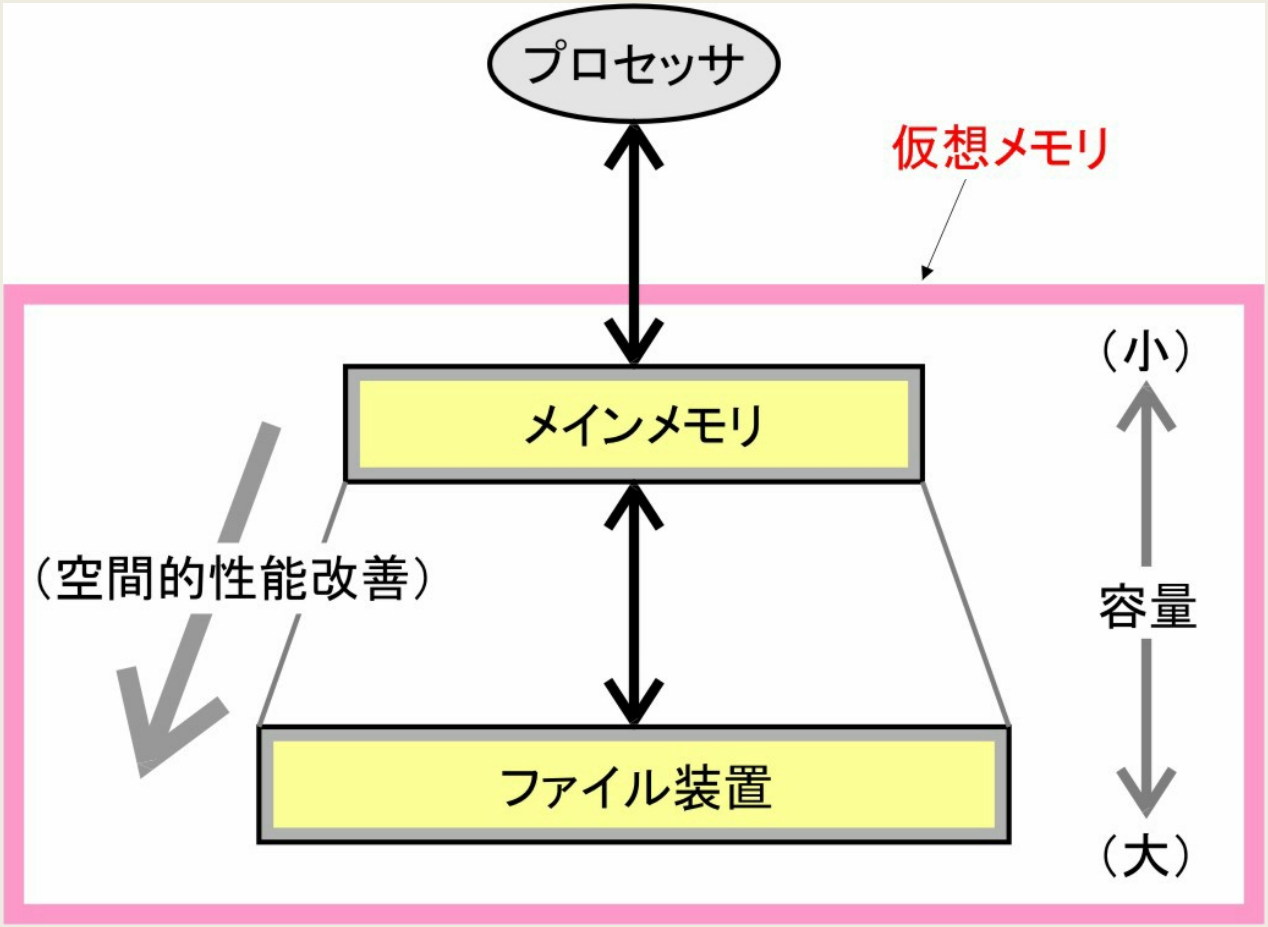
■ 仕組みの概要

- ファイル装置をメインメモリのバックアップメモリとして利用
- マシン命令（正確には，マシン命令中のオペランド）で指定するメインメモリのアドレス空間を実際の（＝物理的な）メインメモリ容量とは独立に

↓

➢ プロセッサには，「一定容量で広い仮想的なメモリのアドレス空間（＝仮想アドレス空間）」を見せかける

メモリ階層によるメインメモリ性能の改善
ー代表例：仮想メモリ（図説）ー



仮想メモリの原理 (1)

(A) 実メモリ(real memory):メインメモリというハードウェア装置に付けてあるアドレス(＝実アドレス)で指定するアドレス空間
＝ 実際の(＝物理的な)メインメモリのアドレス空間

(B) 仮想メモリ(virtual memory):マシン命令中のメモリオペランドをもとにして生成するアドレス(＝仮想アドレス)で指定するアドレス空間
＝ 仮想の(＝論理的な)メモリのアドレス空間

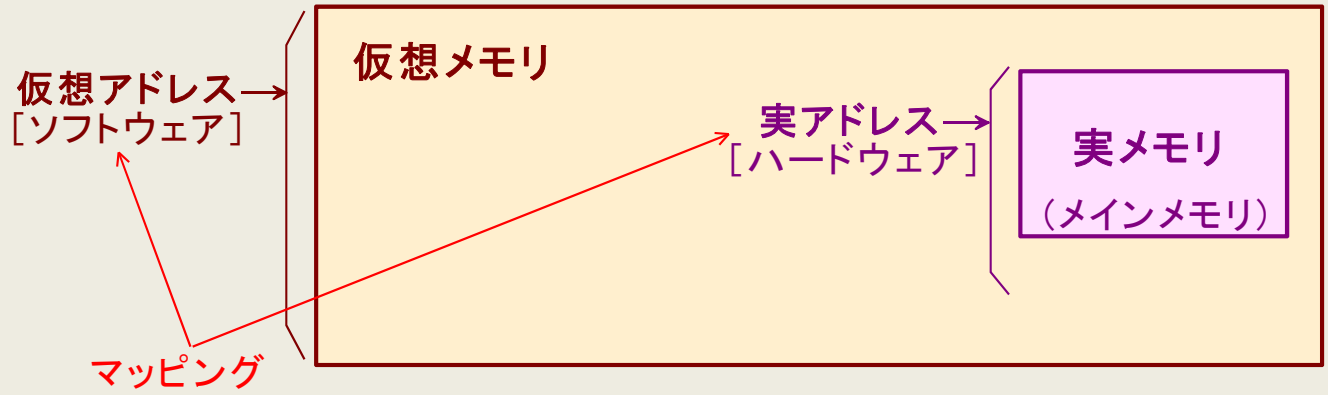
とを独立して構成&相互に対応付け(＝マッピング(mapping))で実現

- マシン命令中のオペランドで指定できるメモリのアドレス空間を仮想的に拡大&一定サイズに固定可

仮想メモリの原理 (2)

(A) 実アドレス：実装しているメインメモリの物理的アドレス ＝ 実メモリのアドレス

(B) 仮想アドレス：マシン命令中のオペランドから生成する論理的アドレス ＝ 仮想メモリのアドレス



仮想メモリの効果

- マシン命令の使用者である (a) プロセッサ：プログラム/プロセスを実行；(b) OS ：プロセスを管理；(c) コンパイラ：マシン命令を生成；からは，

> 実メモリ/実アドレス空間ではなく，一定で巨大なサイズの仮想メモリ/仮想アドレス空間が見える

- メインメモリ（＝実メモリ）の多種多様な性能仕様（特に，容量）の相違を隠ぺい，統一した性能仕様（特に，容量）のメモリ領域やアドレス空間を，ハードウェア（プロセッサ）&ソフトウェア（OS自身，コンパイラ，ユーザプログラムなど）に提示/提供可

■ 仮想メモリの実現

(1) メインメモリ（＝実メモリ）の利用効率が良
(2) メインメモリ（＝実メモリ）サイズによる制約を事実上撤廃可

OSから見る仮想メモリ －仮想メモリ（概要）－

- **OS**から見る（メイン）メモリは“一定サイズに固定&巨大サイズの（仮想）メモリ”
  - **OS**によるメモリ管理とは“仮想メモリ管理”

■ **OS**に与える具体的な効果

- 一定サイズ（例：Windowsなどの32ビット版パソコンOSでは，4G≒2^{32}バイト）で広い仮想メモリ空間の連続領域をプロセス割り付け可 → 実アドレス空間の外部フラグメンテーションを防止可&（メインメモリへの）プロセス割り付けでの実アドレス空間サイズの考慮不要
- プログラムやデータの論理的な意味に配慮したプロセス割り付け/プロセス管理/ファイル管理が可能
- プログラムを動的にメインメモリ（＝実メモリ）上でリロケーション（relocation；再配置＝場所を変えての保持し直し）可能 → プロセス割り付けが容易