電気工学通論Ⅱ（１１）

坂井　修一
東京大学大学院 情報理工学系研究科 電子情報学専攻
東京大学　工学部　電子情報工学科

- 講義の概要と予定
- キャッシュと仮想記憶

はじめに

- 本講義の目的
  - ディジタル回路とコンピュータアーキテクチャの基礎を学習する
- 時間・場所
  - 火曜日 8:30 - 10:15、工学部2号館213
- ホームページ（ダウンロード可能）
  - url:　http://www.mtl.t.u-tokyo.ac.jp/~sakai/tsuron2/
- 教科書
  - 坂井修一『論理回路入門』（培風館）
  - 坂井修一　『コンピュータアーキテクチャ』（コロナ社）

講義の概要と予定（１／２）

１．ディジタル回路入門

ディジタルとアナログ、２進数、補数表現、四則演算

２．　論理演算

論理演算とは、３つの基本論理演算、NORとNAND、完備性、
ディジタル回路の表現、ブール代数、標準形

３．組み合わせ回路の構成法

真理値表と組合せ回路、組合せ回路の簡単化

４．組合せ回路の実例

加算器、補数、減算器、ＡＬＵ、デコーダ、セレクタなど

５．フリップフロップ

ＦＦ、ＳＲラッチ、Ｄラッチ、非同期と同期、SR-FF、D-FF、
マスタスレーブ形とエッジトリガ形, JK-FF, レジスタ

６．基本的な順序回路

７．一般的な順序回路の作り方

講義の概要と予定（２／２）

８．コンピュータアーキテクチャ入門

計算のサイクル、主記憶装置、メモリの構成と分類、レジスタファイル、命令、命令実行の仕組み、実行サイクル、算術論理演算命令、シーケンサ、条件分岐命令

９．命令セットアーキテクチャ

操作とオペランド、命令の表現形式、アセンブリ言語、命令セット、算術論理演算命令、データ移動命令、分岐命令、アドレシング、サブルーチン、RISCとCISC

10．パイプライン

11．キャッシュと仮想記憶

12．入出力と周辺装置

周辺装置、ディスプレイ、二次記憶装置、ハードウェアインタフェース、割り込みとポーリング、アービタ、DMA、例外処理

休講：　X月Y日　　試験：Z月W日

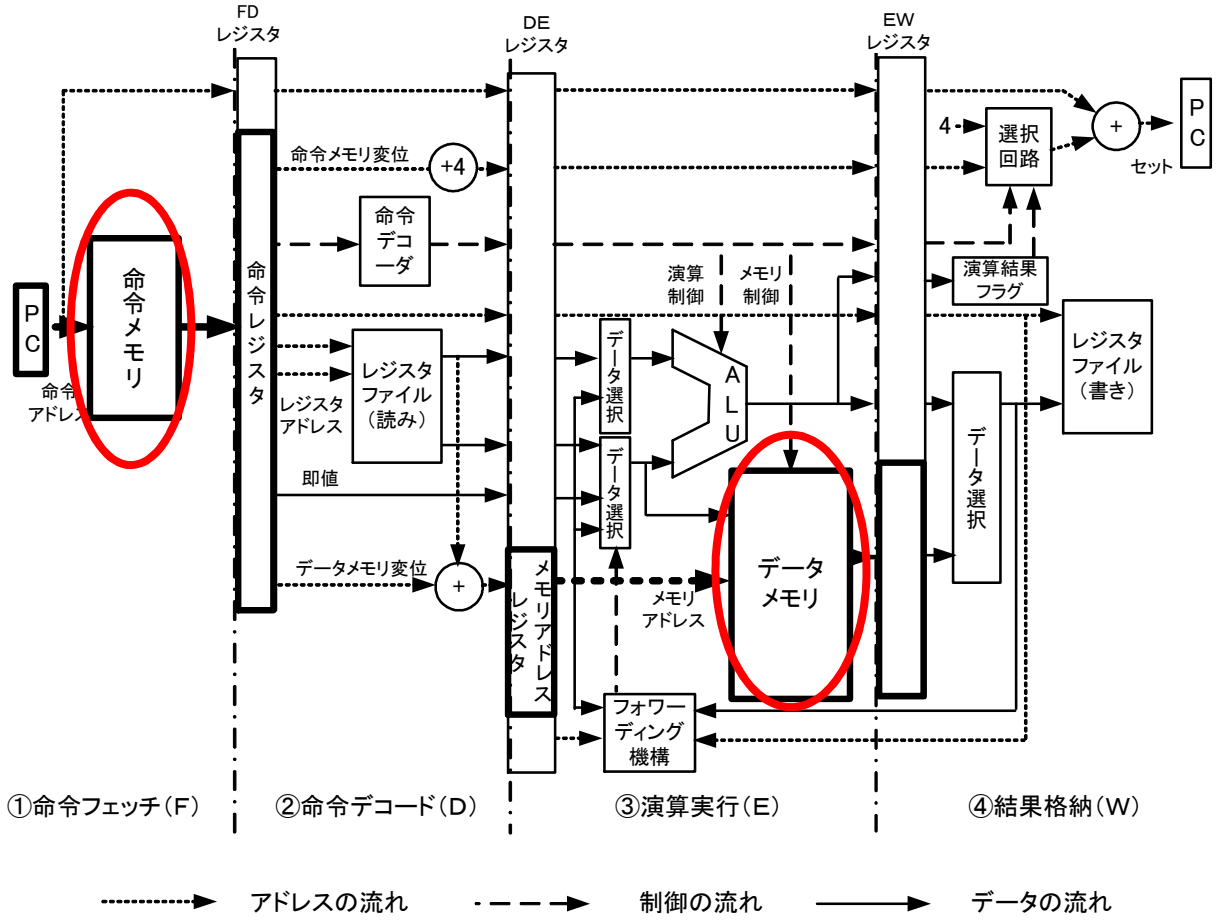
成績：出席（小テスト）＋試験　　４年生：追試・レポート無し

１１．キャッシュと仮想記憶

- 内容
  - 記憶階層
    - 命令パイプラインとメモリ
    - 記憶階層と局所性
  - キャッシュ
    - キャッシュとはなにか
    - ライトスルーとライトバック
    - ダイレクトマップ型キャッシュ
    - キャッシュミス
    - フルアソシアティブ型キャッシュ
    - セットアソシアティブ型キャッシュ
    - キャッシュの入ったCPU
    - キャッシュの性能
  - 仮想記憶
    - 仮想記憶とはなにか
    - 仮想記憶の構成
  - 透過性と互換性

命令パイプラインとメモリ

- 問題： はたしてメモリは1クロックで読み書きできるか？



命令メモリ、データメモリは
ＣＰＵチップの外に置かれる
↓
アクセスに時間がかかる

メモリの理想と現実

- 理想
  - 無限大の容量
  - 無限小のアクセス時間
  - 単純なアドレシング
- 現実
  - 大容量と高速アクセスは両立しない
    - 容量が大きくなるとアクセス時間も大きくなる

（メモ）メモリの階層化

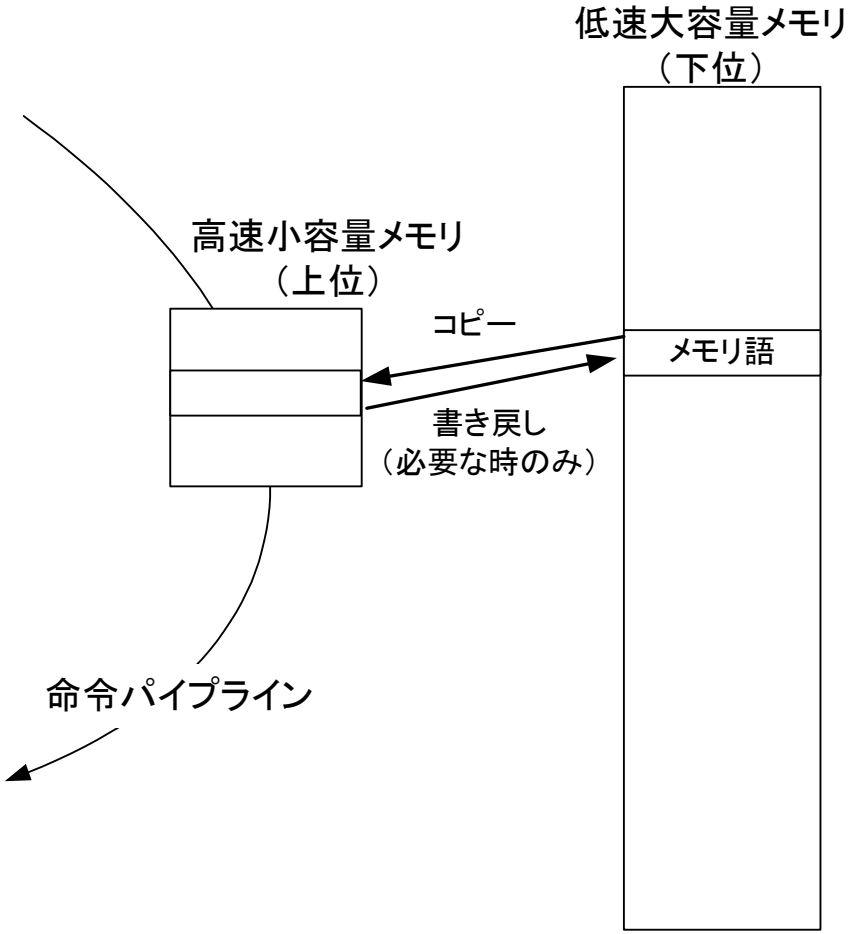
- アクセス・ギャップ
  - CPUの命令実行速度とメモリのアクセス時間のギャップの増大
    - メモリ素子技術とCPU素子技術の差異 （DRAMセルと、フリップフロップ）
    - チップ内遅延時間 （CPUは局所性利用、メモリは √n）
    - チップ内外のアクセス時間差 （チップ内 1GHz、チップ外 133MHz）
    - メモリの大規模化とアドレスTree, マルチプレクサ Tree
    - 例： CPU 1GHzクロック、1クロックに最大4命令実行

      メモリ SDRAM 133MHzクロック、アクセス時間6クロック

      メモリ DDR 266MHzクロック、アクセス時間10クロック

（DDR=Double Data Rate クロックの上げエッジと下げエッジの両方を用いる）

記憶階層と局所性



- なぜ記憶階層が有効か？
  命令やデータに局所性があるから
  - 空間局所性
    - あるメモリ語が参照されたときに、その語の近くの語が引き続き参照される性質
  - 時間局所性
    - あるメモリ語が参照されたとき、その語が時間をおかずに再び参照される性質

メモ：時間的局所性と空間的局所性

- 時間的局所性(Temporal Locality)
  - 一度アクセスしたアドレスに、近い将来再びアクセスすること
  - ループ内のスカラ変数、大域変数など
  - ループ構造、再帰構造の命令アクセス

- 空間的局所性(Spacial Locality)
  - ある場所にアクセスすると、近い将来その近傍の場所にアクセスすること
  - ベクトル、マトリクスのアクセス
  - 逐次実行中の命令アクセス
  - 入れ子構造のローカルデータ

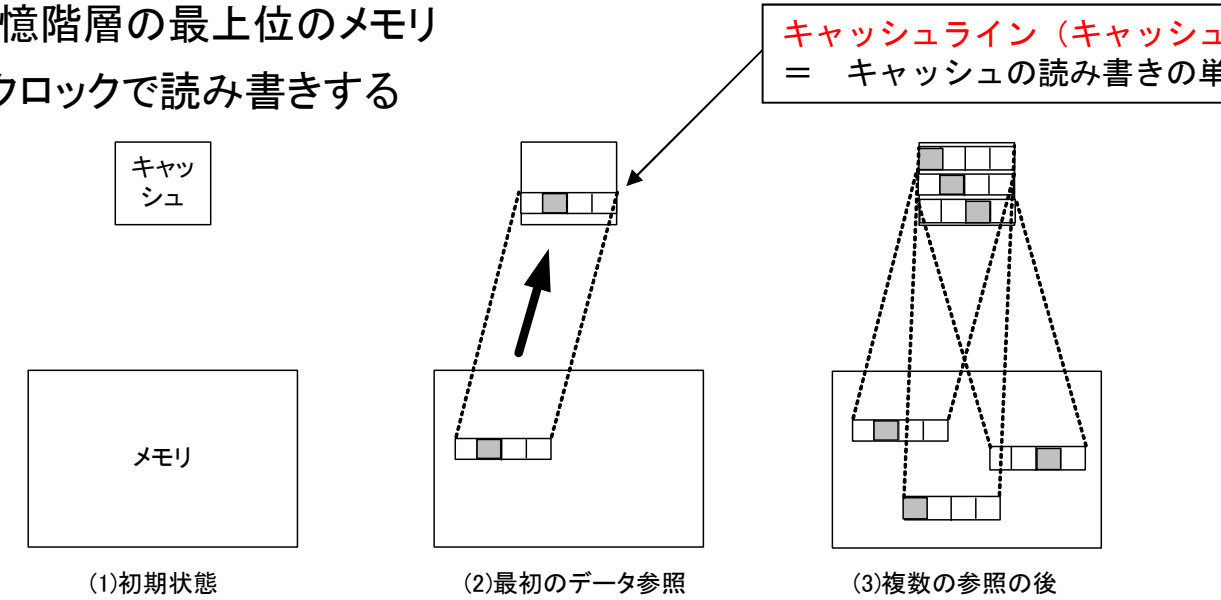
- 一般のプログラムではこの両者が混合して出現
- 全アドレス空間中に、局所性を持つ場所がN個出現
  - 一般に、Nは2から5位（強くプログラム依存）

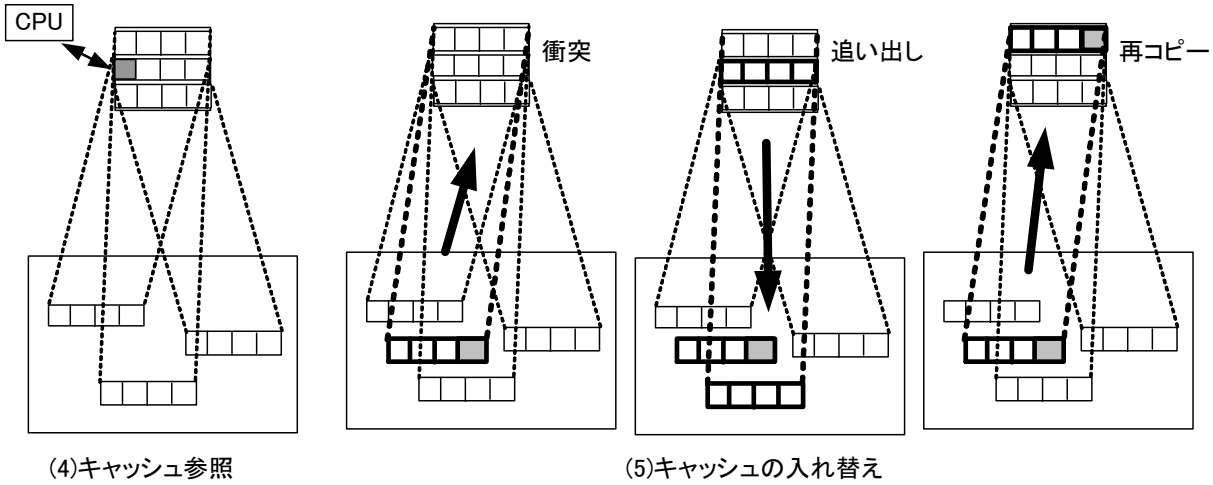
記憶階層と機械語プログラム

- ◆ 記憶階層を陽に見せる方式
  - ・ 各階層のメモリのそれぞれにアドレスを付け、コピーや書き戻しもロード命令・ストア命令で実現する
  - ×アセンブリ言語のプログラマが記憶階層を意識しなくてはならず、いつも最良のメモリの利用法を考えてプログラムをしなくてはならない。
  - ×命令セットが同じでもメモリの階層構成が変化したり、各階層のメモリの大きさが変化したりするたびにプログラムを作り直さなくてはならない。
- ◆ 記憶階層を見せない方式
  - ・ 見かけ上は、高速で大容量のメモリが1つだけあるものとして機械語のプログラムを書き、ハードウェアの機構でどの階層のメモリをどう使って局所性を活かすかを決める
  - ◎単純に命令セットだけを意識して機械語プログラムを書いておけば、効率や安全性はハードウェアが勝手に面倒を見てくれる。この性質を**透過性(transparency)**と呼ぶ。

キャッシュとは何か

- キャッシュ
  - 記憶階層の最上位のメモリ
  - 1クロックで読み書きする

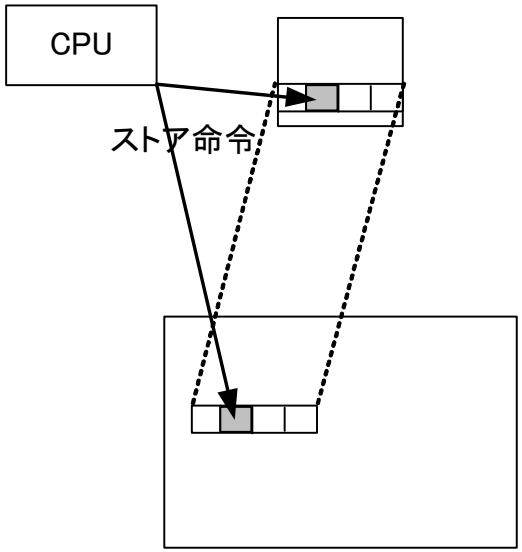




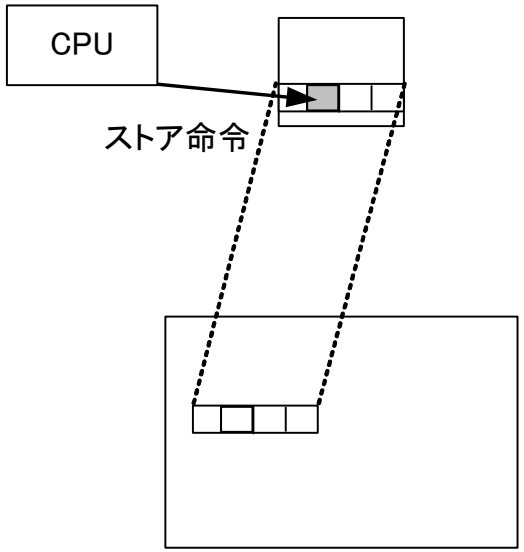
キャッシュの分類

- メモリ書き込み方式による分類
  - ライトスルー
  - ライトバック
- 連想度による分類
  - ダイレクトマップ
  - フルアソシアティブ
  - セットアソシアティブ

ライトスルー方式とライトバック方式



(1)ライトスルー方式

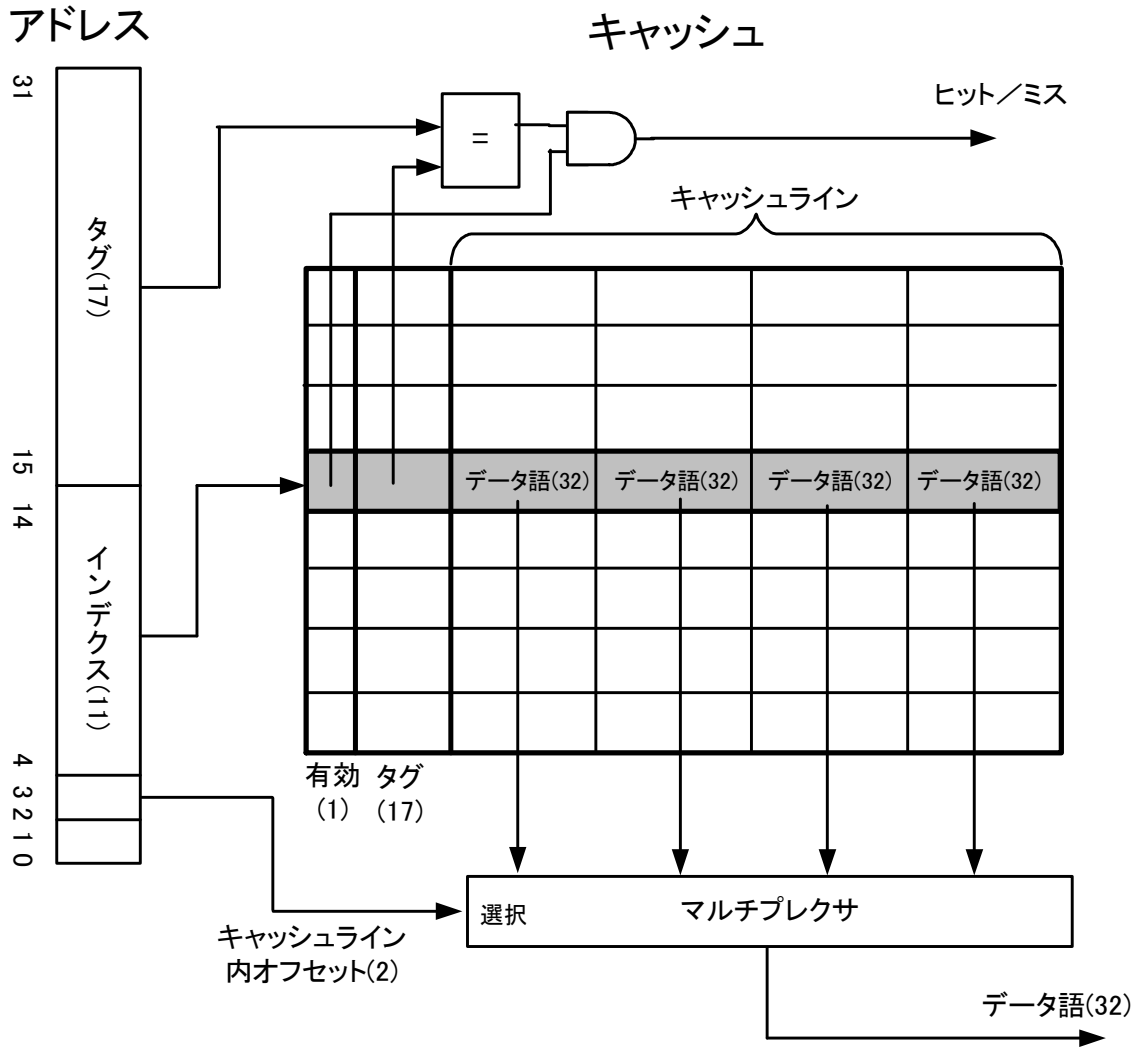


(2)ライトバック方式

表 5.1 ライトスルー方式とライトバック方式

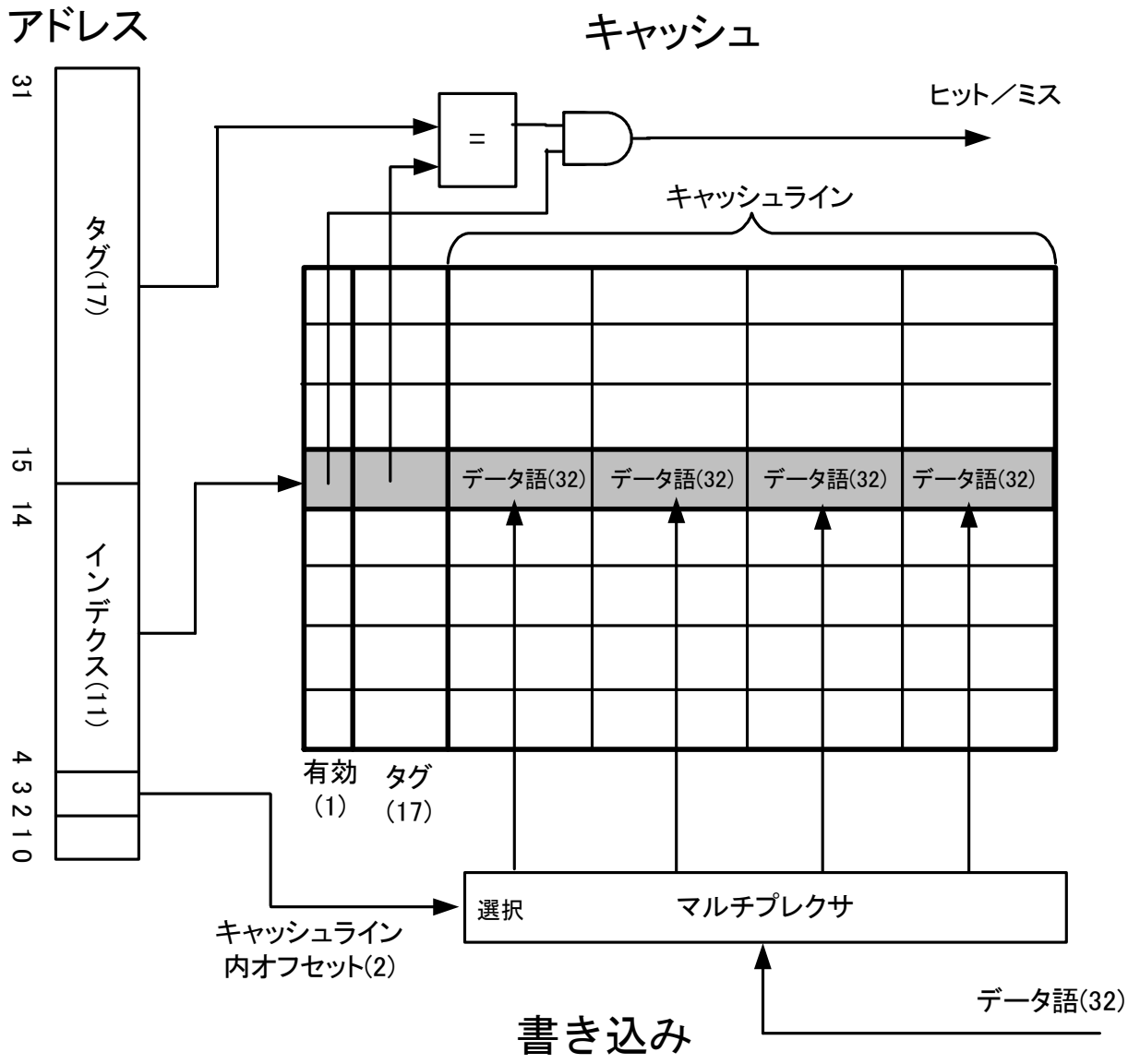
	ライトスルー方式	ライトバック方式
メモリアクセス	ストア命令の実行時	キャッシュライン追い出しの時
ライト命令の実行速度	ライトバッファの速度	キャッシュの速度
キャッシュライン書き戻し	不要	キャッシュライン追い出しの時
実装	単純	複雑

ダイレクトマップ型キャッシュ（1）



読み出し

ダイレクトマップ型キャッシュ（2）



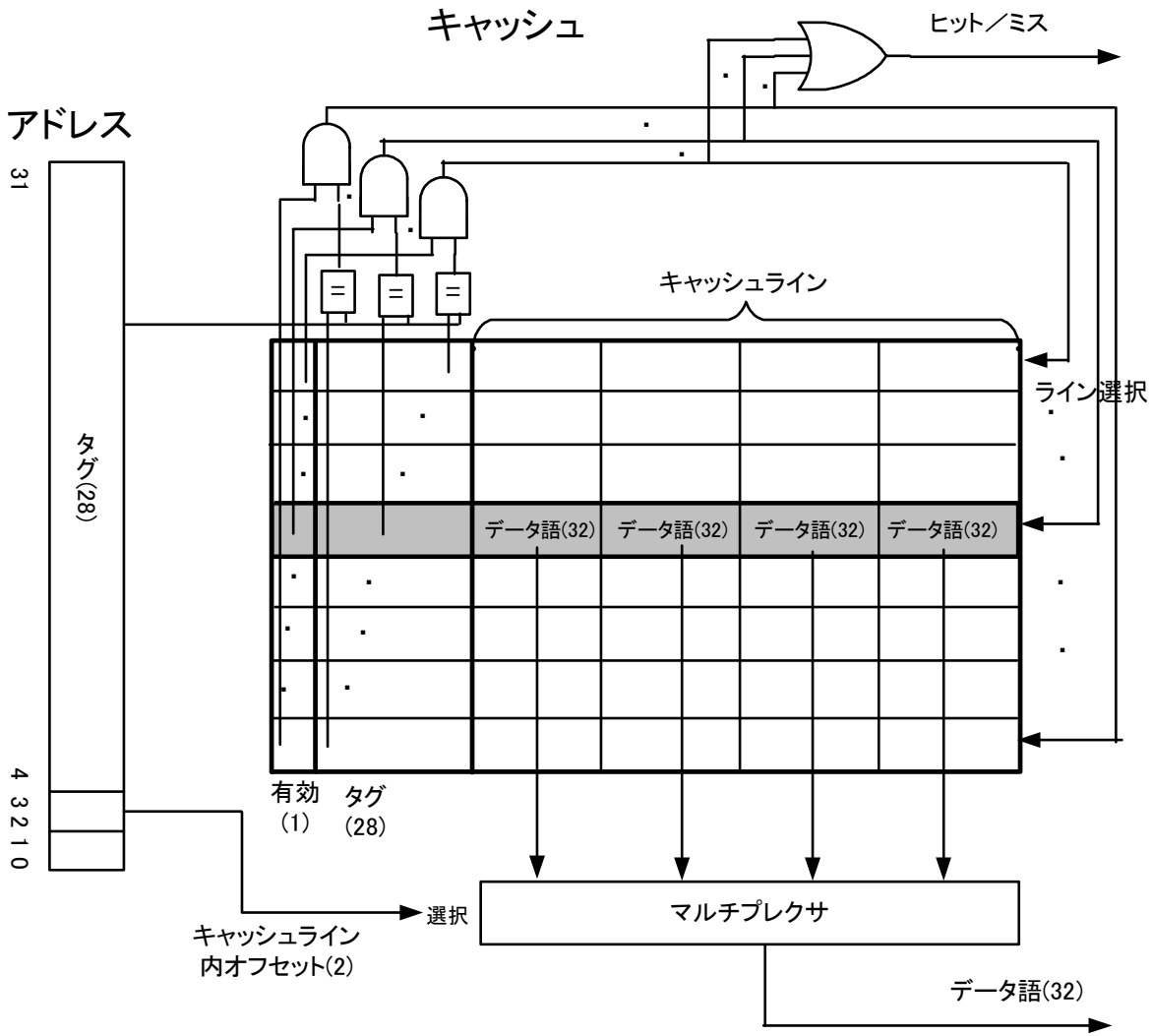
キャッシュミス

キャッシュミス　＝　３つのＣ

①初期参照ミス(compulsory miss, cold start miss)
　　キャッシュラインを最初にアクセスするときに起こるミス
②競合性ミス(conflict miss, collision miss)
　　同じインデクスをもつ異なるキャッシュラインにアクセスすることで起こるミス
③容量性ミス(capacity miss)
　　キャッシュしたいライン数がキャッシュ容量を上回ることで起こるミス

メモ　競合性ミスはフルアソシアティブ型キャッシュでは起こらない

フルアソシアティブ型キャッシュ



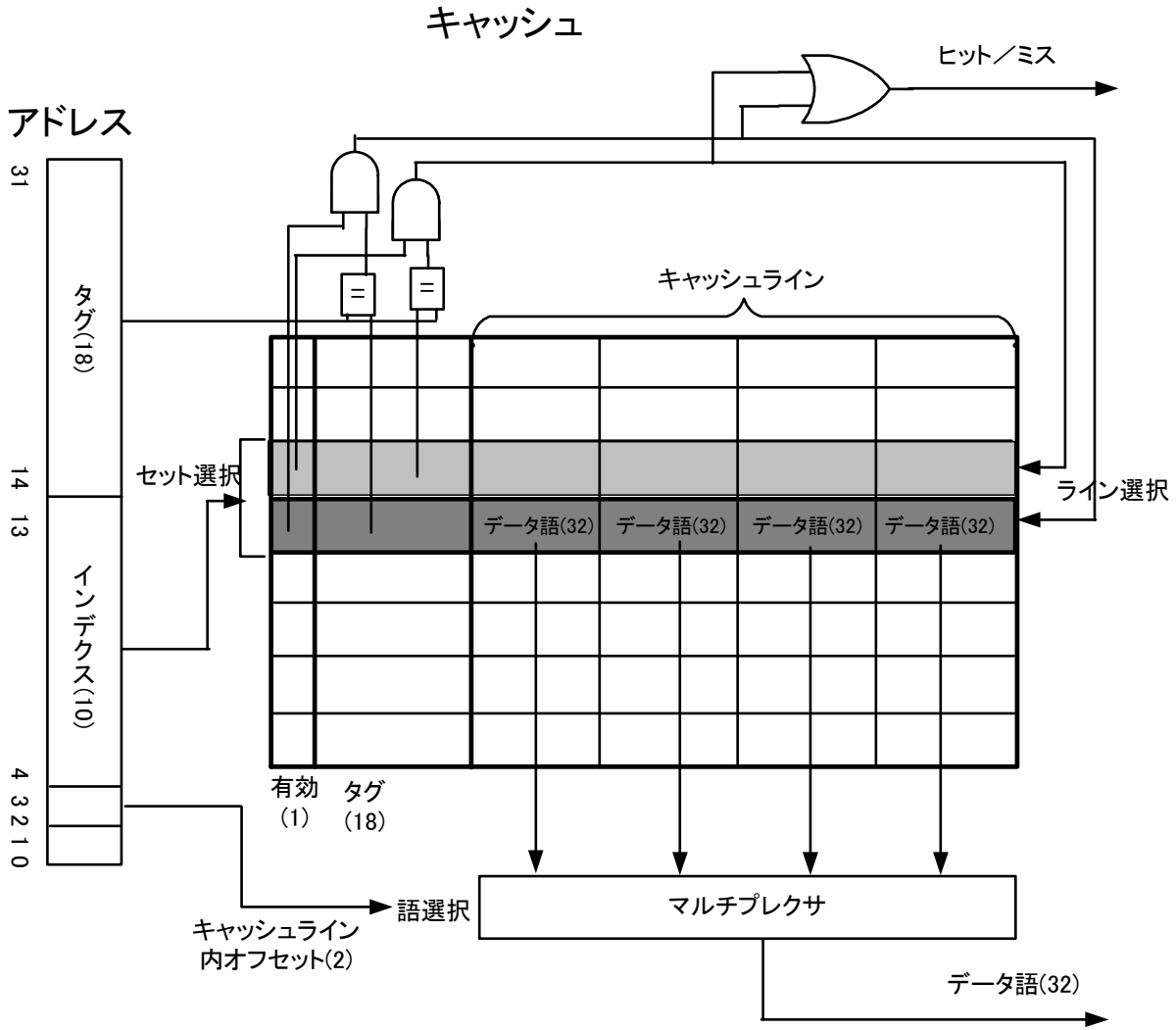
$L = S \times A$

L：ライン数
S：セット数
A：連想度

◆フルアソシアティブ型
$A = L$

◆ダイレクトマップ型
$A = 1$

セットアソシアティブ型



キャッシュ方式の比較

表 5.2　キャッシュの3つの型

	ダイレクトマップ	セットアソシアティブ	フルアソシアティブ
連想度	1	A(2、4など)	＝ライン数
セット数	＝ライン数	S	1
ハードウェア	◎	○	×
ゲート遅延	◎	○	×
競合性ミス	×	○	◎

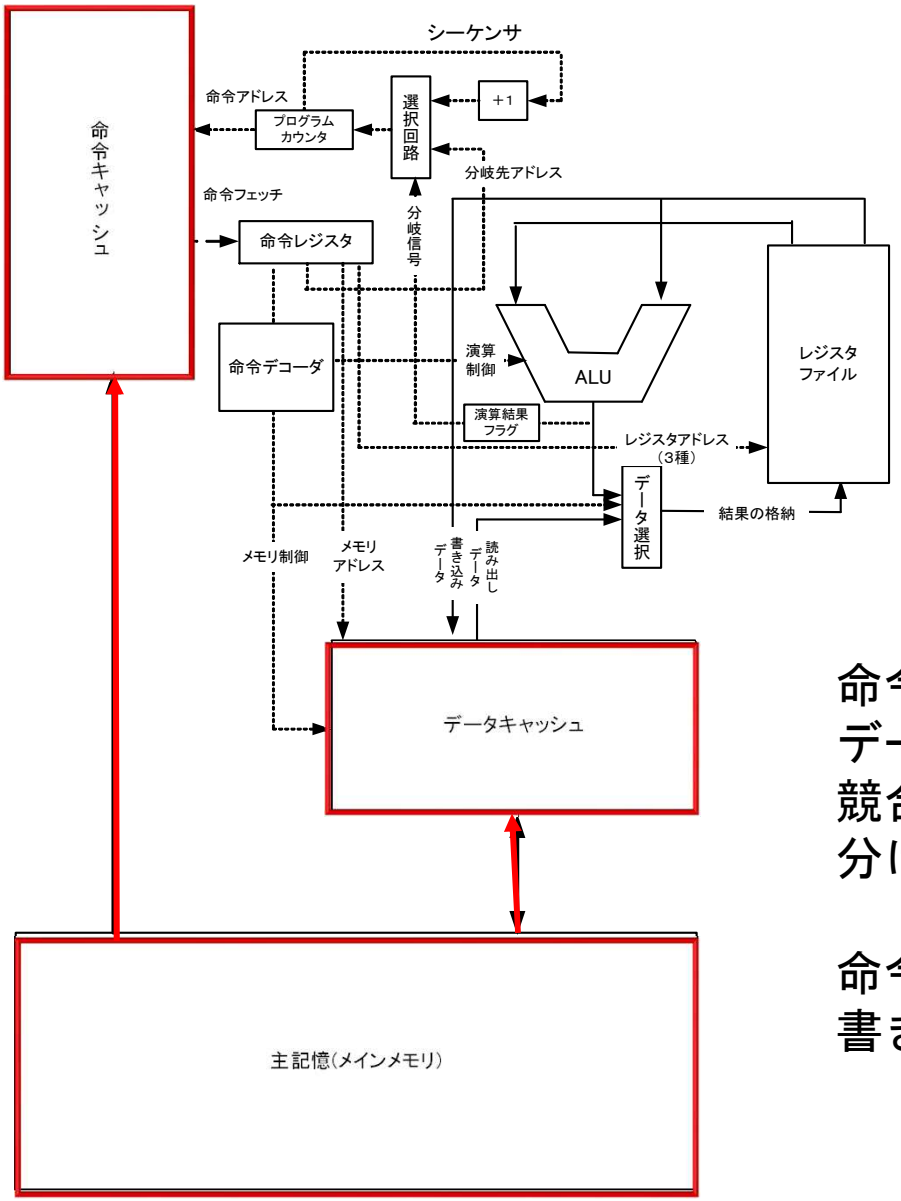
- キャッシュラインの交換

  追い出すラインをどう決めるか？

  - ランダム
  - LRU （Least Recently Used）

    使われていない時間が最も長い(Least Recent Used)ラインを追い出しの対象とする

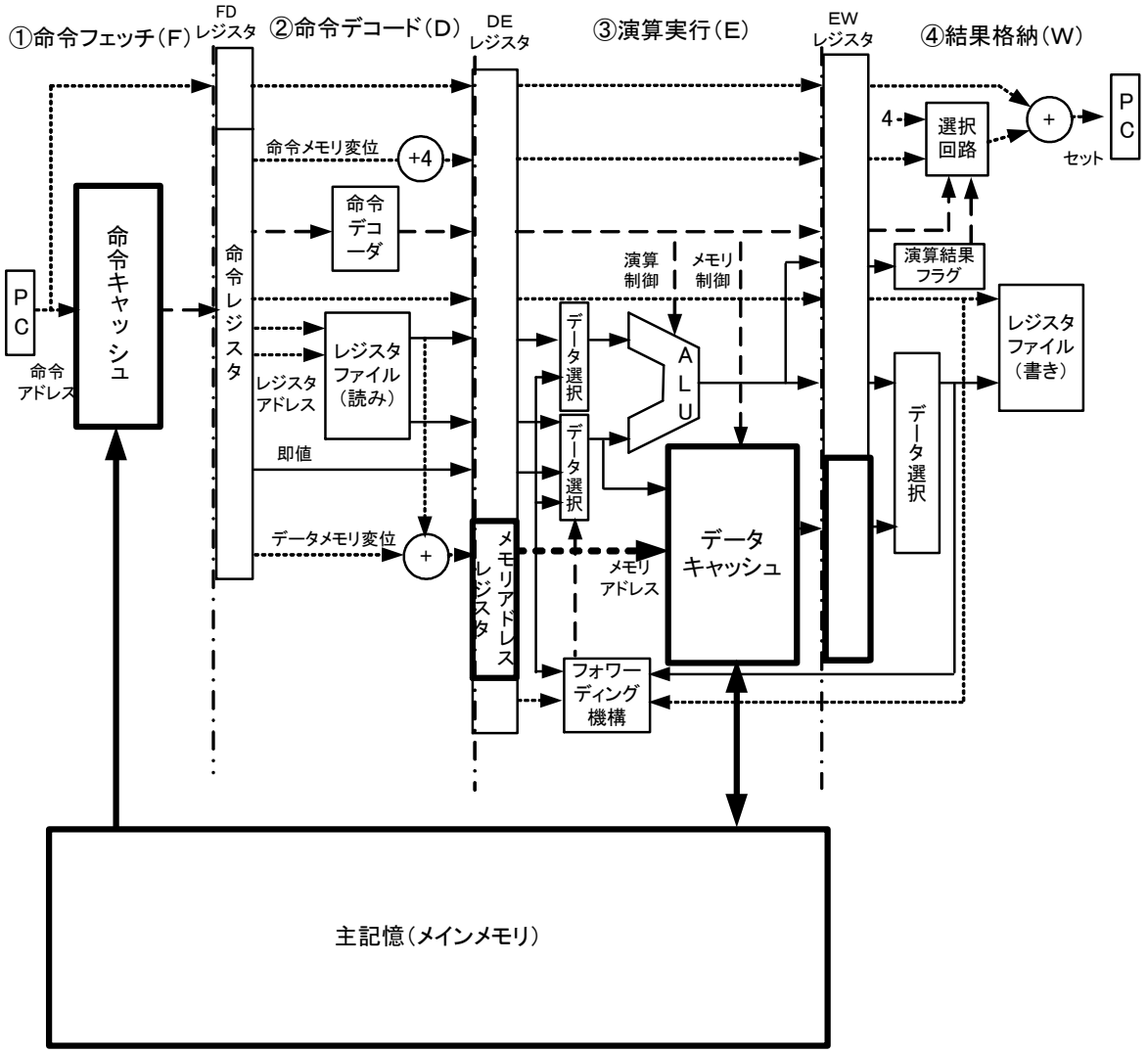
キャッシュの入ったCPU



命令キャッシュと
データキャッシュは
競合を避けるために
分ける。

命令キャッシュは
書き戻し機構が不要

キャッシュの入った命令パイプライン



アドレスの流れ　制御の流れ　データの流れ

キャッシュの性能

- $T_p = N \cdot (1 + r_{ls} \cdot r_{miss} \cdot t_{mstall}) / C$
  - T_p:プログラム実行時間、
  - C [Hz] : CPUのクロック速度
  - N : プログラムで実行される命令の数
  - r_{ls}: ロード・ストア命令の割合
  - r_{miss}: ロード・ストア命令ごとのキャッシュミス率
  - t_{mstall} : 1回のミスによるストール時間(ミスペナルティ、miss penalty)

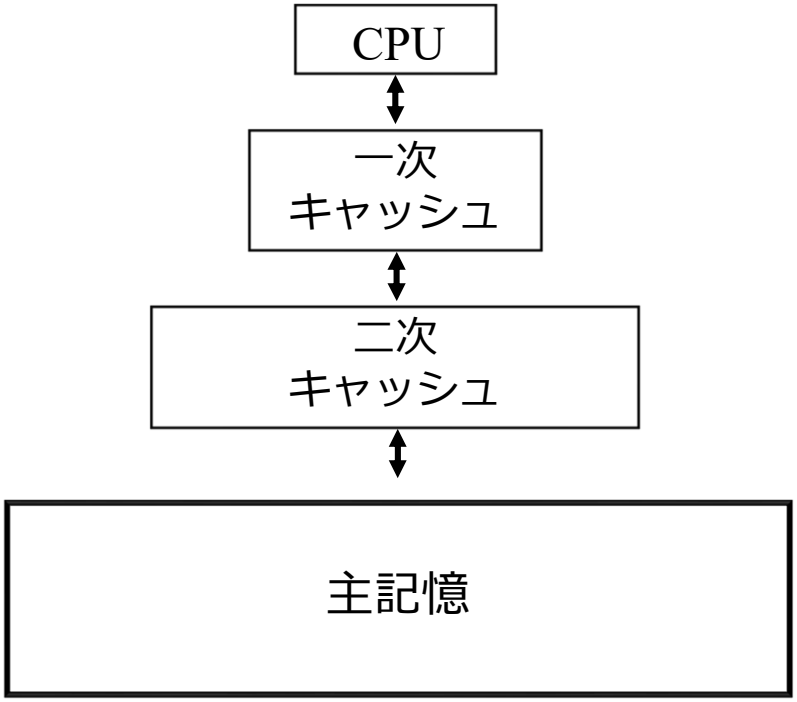
表 5.4　キャッシュミス率とミスペナルティの例（解答）

	ミス率	ミスペナルティ	実行時間相対値
事例 1	0	—	1
事例 2	0.05	10	1.15
事例 3	0.05	50	1.75
事例 4	0.5	10	2.5
事例 5	0.5	50	8.5

$r_{ls} = 0.3$の場合

メモ

- キャッシュが複数レベルのものも存在

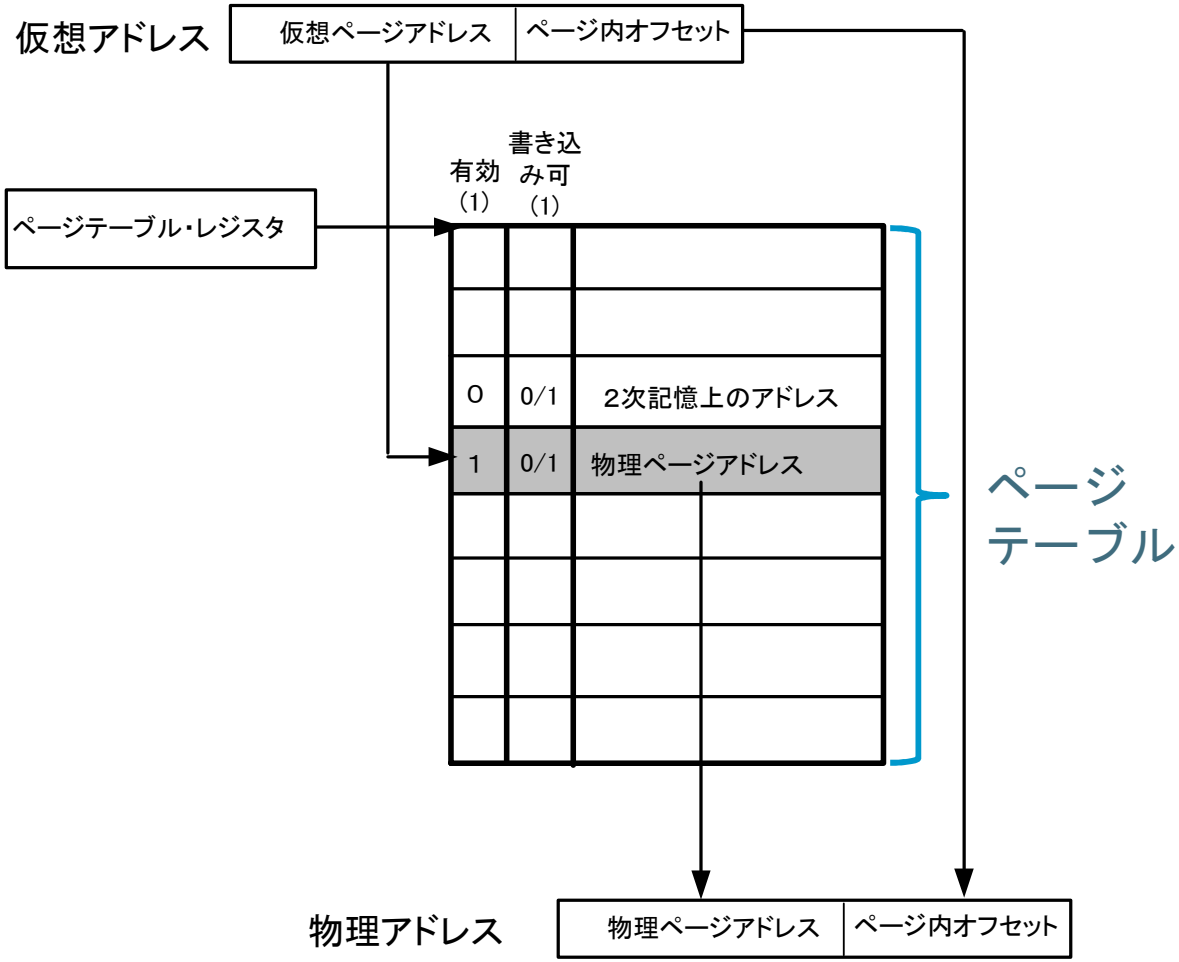


Intel core i7の場合
一次（1L)キャッシュ　32KB
二次（2L）キャッシュ　256KB
3次（3L)キャッシュ　8MB

仮想記憶とはなにか

- 仮想記憶　＝　主記憶以下の階層化
  - 主記憶と二次記憶のメモリ階層を「巨大な主記憶」として使えるように透過性をもたせたもの
    - 主記憶よりも大きな容量のメモリがあるものとしてプログラムが書けるようになる
    - 複数のプログラムが一つの物理記憶を安全に分かち合って使えるようになる
- 仮想記憶の動作原理
  - アドレス変換
    - 仮想アドレス(virtual address) ⇒ 物理アドレス(physical address)
  - スワップ
    - 2次記憶(HDD)のデータ　⇔　主記憶のデータ

仮想記憶の構成



- ページ
  - データ移動の単位。通常数KB =>最近は数MBのものも(ラージページ)
- アドレス変換
  - ページテーブル
- スワップ
  - フルアソシタティブ、ライトバック

ページフォールト

ページフォールト

ページが主記憶に入っていない状態

＝　ページテーブルで有効ビットが0のとき

ページフォールトが起こったとき

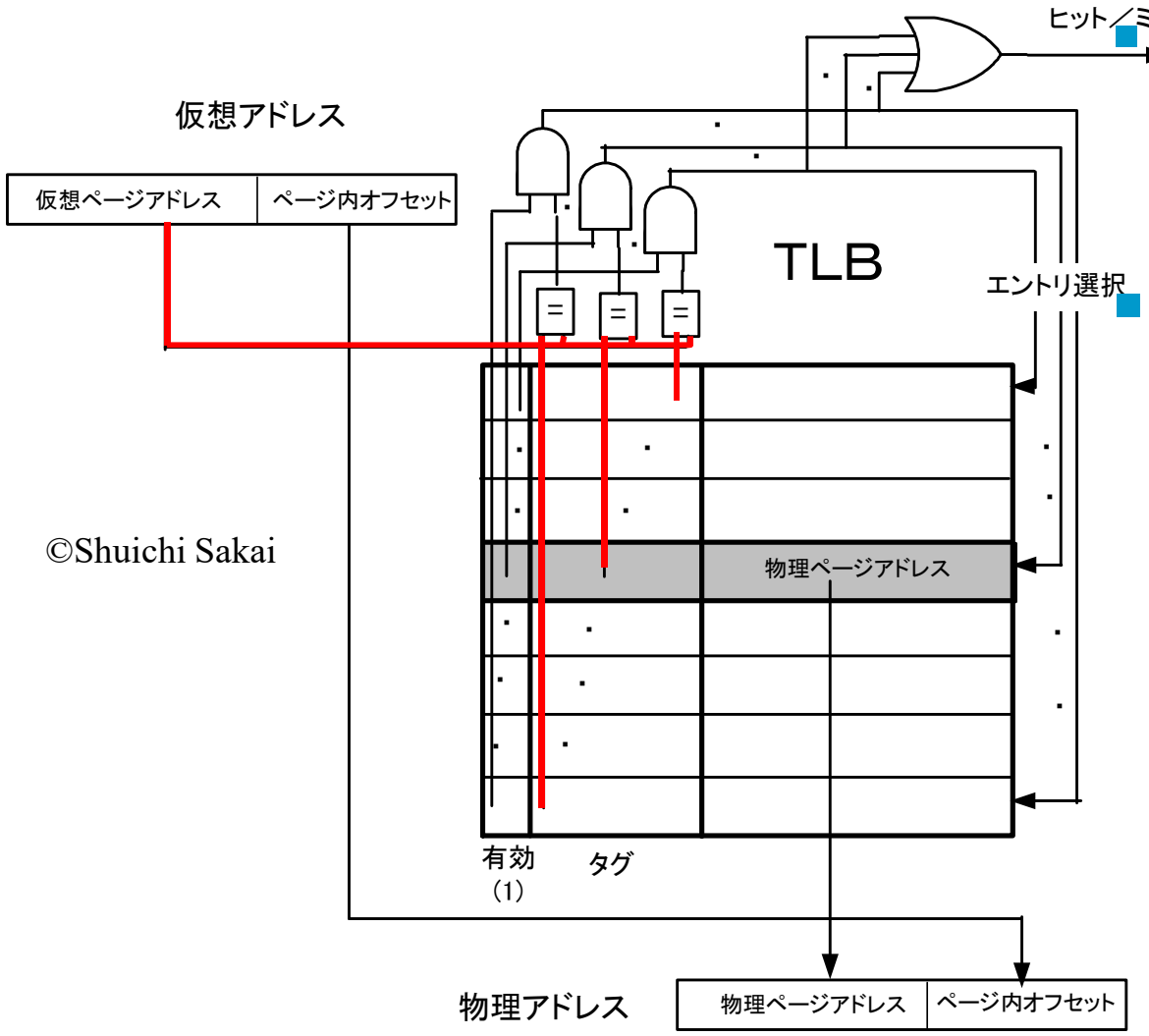
（1） CPUの処理を中断する　⇒　例外処理

（2） テーブルのエントリに入っている二次記憶上のアドレスから主記憶の空いている場所にページをコピー

（空いている場所がなければ追い出す）

（3） ページテーブルのエントリに物理ページアドレスを書き込み、有効ビットを1にする

TLB



TLB

Translation Lookaside Buffer

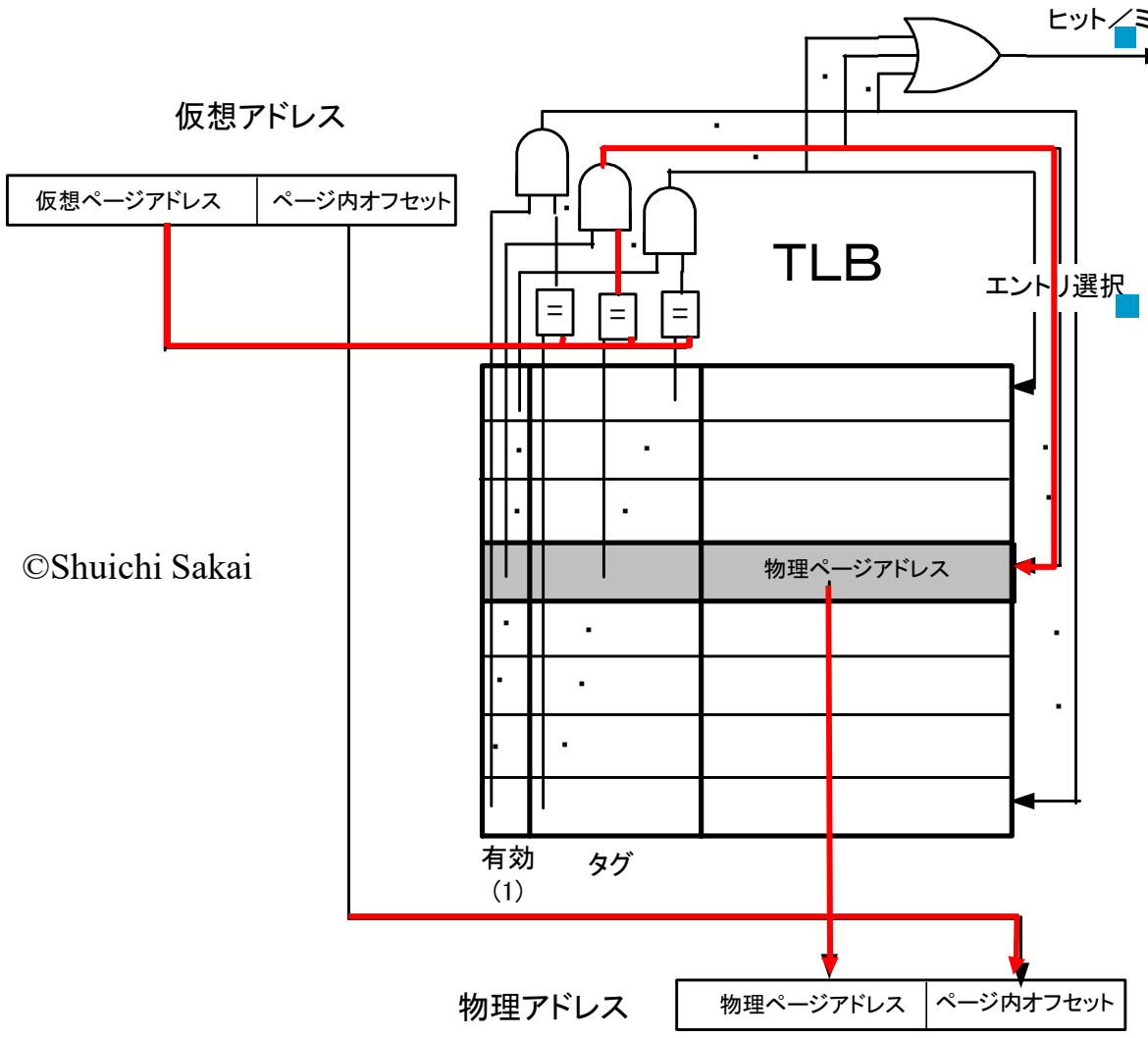
＝ページテーブルのキャッシュ

フルアソシアティブ方式

動作

- (1) メモリアクセスが起こると、仮想アドレスをタグとして、TLBが参照される。
- (2) TLBがヒットすると、該当する物理ページアドレスが取り出され、ページ内オフセットとあわせて物理アドレスが作られる。
- (3)TLBがミスすると、ページテーブルが参照され、TLBの空いているエントリに、現在参照している仮想ページアドレスに対応する物理ページアドレスが入れられる。TLBが空いていない場合は、LRUなどのやりかたでエントリがひとつあけられる。

TLB



TLB

Translation Lookaside Buffer

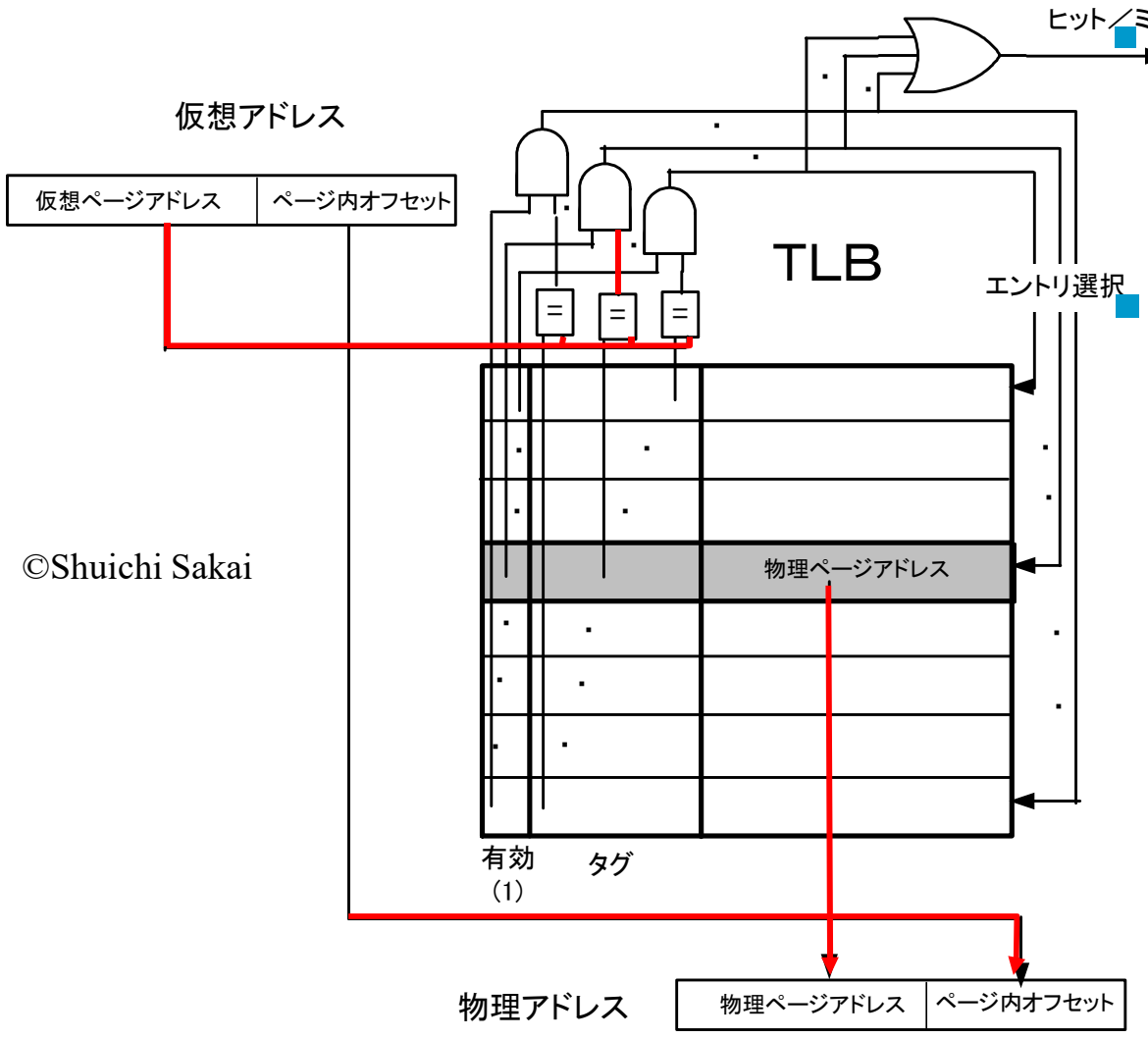
＝ページテーブルのキャッシュ

フルアソシアティブ方式

動作

- (1) メモリアクセスが起こると、仮想アドレスをタグとして、TLBが参照される。
- (2) TLBがヒットすると、該当する物理ページアドレスが取り出され、ページ内オフセットとあわせて物理アドレスが作られる。
- (3)TLBがミスすると、ページテーブルが参照され、TLBの空いているエントリに、現在参照している仮想ページアドレスに対応する物理ページアドレスが入れられる。TLBが空いていない場合は、LRUなどのやりかたでエントリがひとつあけられる。

TLB



TLB

Translation Lookaside Buffer

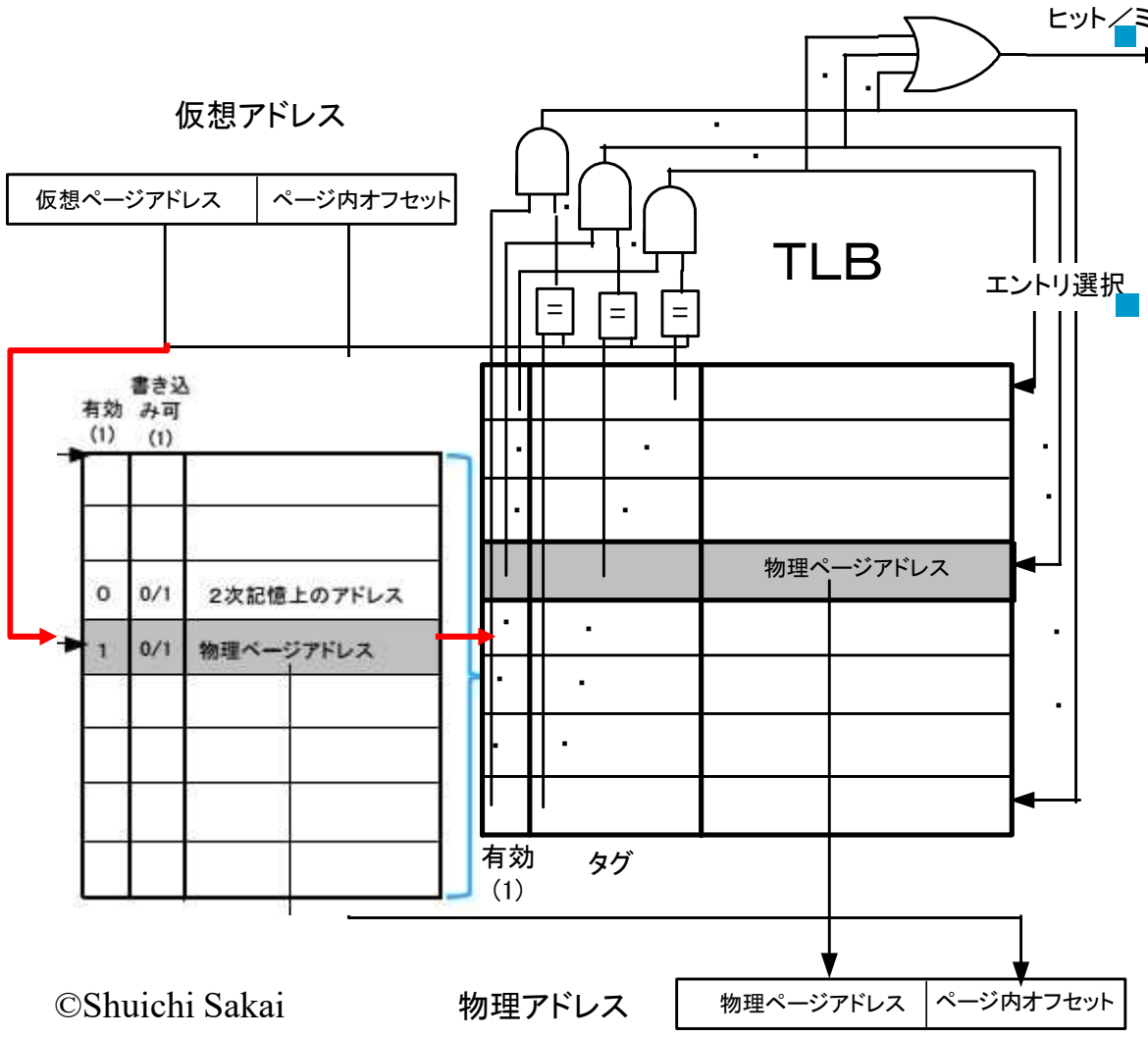
＝ページテーブルのキャッシュ

フルアソシアティブ方式

動作

- (1) メモリアクセスが起こると、仮想アドレスをタグとして、TLBが参照される。
- (2) TLBがヒットすると、該当する物理ページアドレスが取り出され、ページ内オフセットとあわせて物理アドレスが作られる。
- (3)TLBがミスすると、ページテーブルが参照され、TLBの空いているエントリに、現在参照している仮想ページアドレスに対応する物理ページアドレスが入れられる。TLBが空いていない場合は、LRUなどのやりかたでエントリがひとつあけられる。

TLB



TLB

Translation Lookaside Buffer

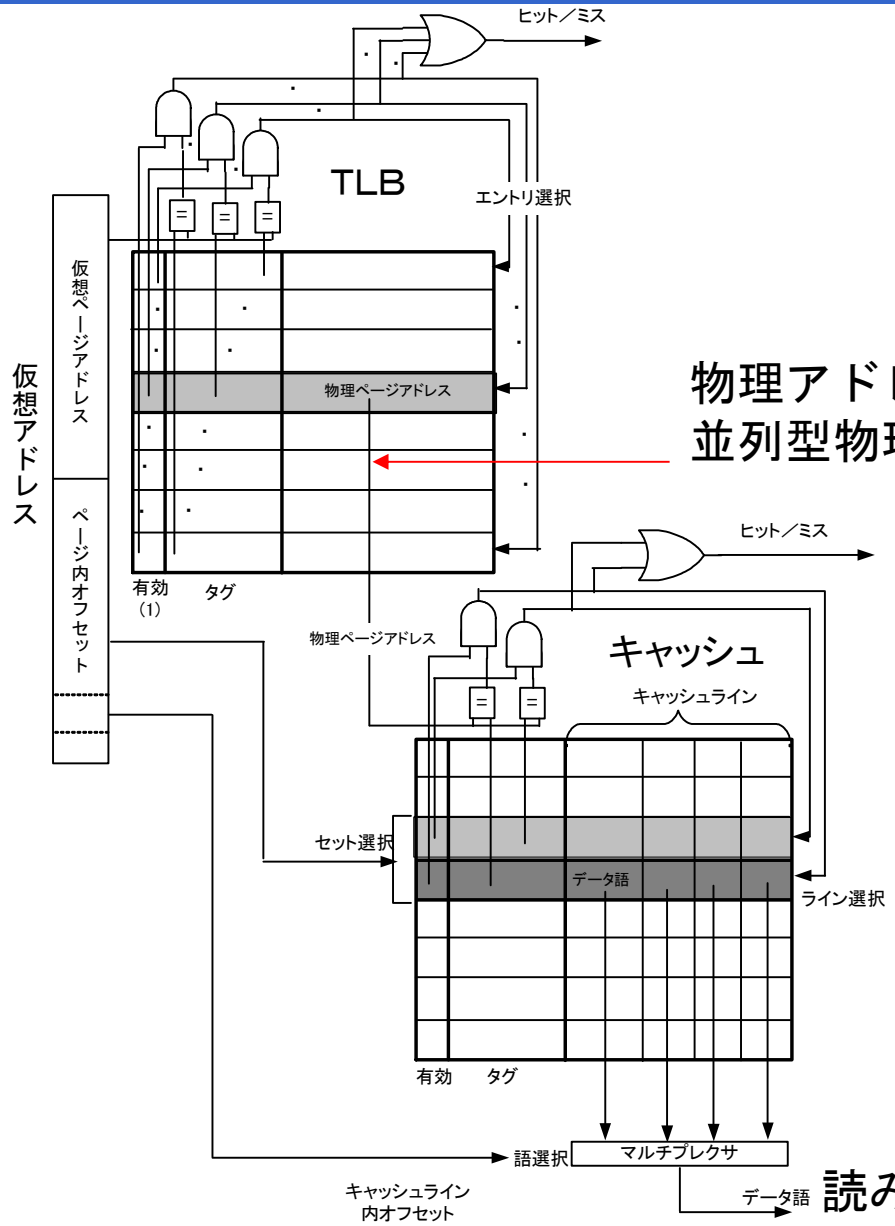
＝ページテーブルのキャッシュ

フルアソシアティブ方式

動作

- (1) メモリアクセスが起こると、仮想アドレスをタグとして、TLBが参照される。
- (2) TLBがヒットすると、該当する物理ページアドレスが取り出され、ページ内オフセットとあわせて物理アドレスが作られる。
- (3)TLBがミスすると、ページテーブルが参照され、TLBの空いているエントリに、現在参照している仮想ページアドレスに対応する物理ページアドレスが入れられる。TLBが空いていない場合は、LRUなどのやりかたでエントリがひとつあけられる。

メモリアクセス機構



物理アドレスで照合しているので
並列型物理アドレスキャッシュ

2way Set associative

読み出し

透過性と互換性

- 透過性 （ transparency）
  - 機械語プログラムが実装の詳細に影響されない性質
  - キャッシュ、仮想記憶、スーパスカラなどの導入で保持される
    - ハードウェア量の増大・速度低下を招いても実現する！
- 互換性 (compatibility)
  - 異なるコンピュータ間で同じ機械語プログラムが同じ動作をする性質
  - コンピュータの世代間や上位・下位機種の間で保持されるのが望ましい
    - システム発展の障害にもなりえる