計算機方式論

第14章 キャッシュ

–置換アルゴリズム等–

1

ライン置換アルゴリズム

- キャッシュ制御機構は、読出アクセスのミスヒット時に、アクセス対象を含む主記憶ラインをキャッシュに読み込む。
- フル/セットアソシアティブマッピングでは、キャッシュに空きラインがない場合、空きにするキャッシュラインを選び、その更新ビットが“オン”のときそのキャッシュラインを主記憶に追い出し、更新ビットが“オフ”のときそのキャッシュラインを廃棄する。
- 書込アクセスのミスヒット時にも、ライトアロケート方式では、主記憶ラインをキャッシュに読み込むため、空きラインがない場合、キャッシュから追い出すまたは廃棄するラインを選ぶ。
- どのラインをキャッシュから追い出すまたは廃棄するかはライン置換アルゴリズムで決め、ハードウェアで実現する。

2

色々なライン置換アルゴリズム

①LRU(Least Recently Used)

②FIFO(First In First Out)

③FINUFO(First In Not Used First Out)

④LFU(Least Frequently Used)

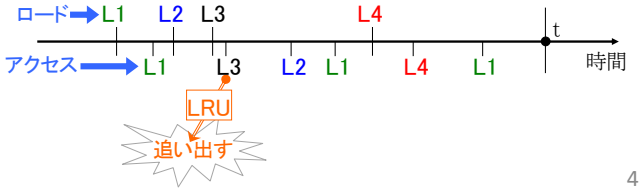
⑤RANDOM

- キャッシュラインの種々の属性(更新ビット、キャッシュへのコピー時刻、アクセス時刻、アクセス回数等)は、タグテーブルに記述する。

3

①LRU(Least Recently Used)

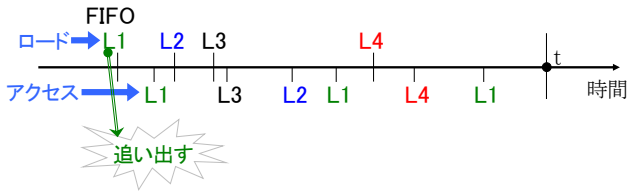
- 最後に参照されてからの経過時間の最も長いキャッシュラインを追い出す。
- 参照局所性を使っているので、ヒット率は良好だが、アクセス時刻を管理する機構が必要になり、判定時間がオーバーヘッドになる。



4

②FIFO(First In First Out)

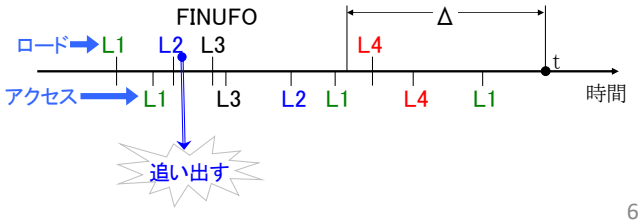
- **一番最初**にキャッシュに**コピー**したラインを追い出す。
- LRUに比べ、参照局所性は劣るが、機構は簡単になる。



5

③FINUFO(First In Not Used First Out)

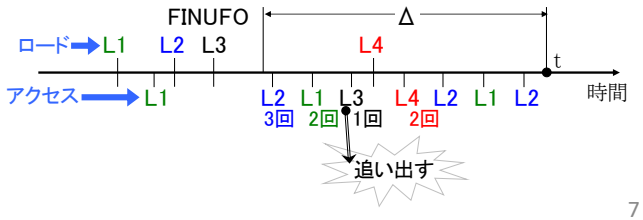
- **一定時間アクセスのない**ラインの内で、**最初に**キャッシュにコピーしたラインを追い出す。
- **WS**(Working Set)法のひとつ。



6

④LFU(Least Frequently Used)

- **一定時間内でアクセス回数が最小**のラインを追い出す。
- **WS**(Working Set)法のひとつ。



7

⑤RANDOM

- **無作為**にラインを決める。機構が簡単。

8

物理キャッシュと論理キャッシュ

- 仮想記憶を使用する計算機
  番地変換機構　論理番地⇒物理番地
- 番地変換機構を
  CPU−?−キャッシュ−?−主記憶
  のどこに?入れるかで、2つの方式がある。

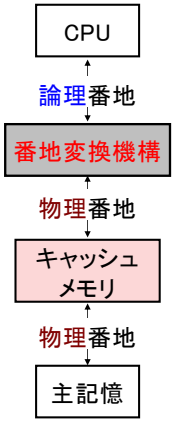
①物理キャッシュ

②論理キャッシュ

9

物理キャッシュ

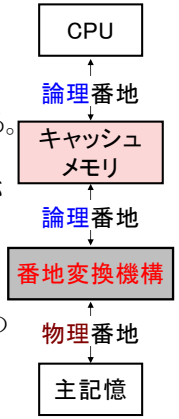
- 番地変換機構が
  CPUとキャッシュ間に入る。
- キャッシュは物理番地で指定。
- キャッシュにアクセスするとき、
  番地変換が必要となり、
  キャッシュアクセス時間が長く
  なる。
- 実アドレスキャッシュともいう。



10

論理キャッシュ

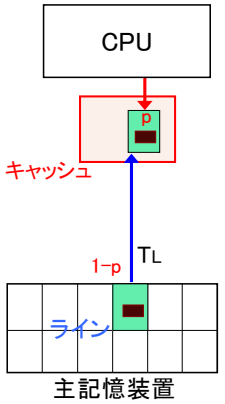
- 番地変換機構が
  キャッシュと主記憶間に入る。
- キャッシュは論理番地で指定。
- キャッシュのアクセスが速やかに行える。
- ラインのマッピング機構に番地変換機構が必要となり、コヒーレンシの保持が複雑な処理になる。
- 仮想記憶空間を切り替える度に、
  キャッシュの内容を無効にするため
  フラッシュを行うので、キャッシュ容量の
  大きいときはオーバーヘッドになる。
- 仮想アドレスキャッシュともいう。



11

2階層キャッシュへ

- キャッシュの実効アクセス時間Te
  $T_e = T_c + (1-p)T_L$
- キャッシュ容量を増やす！
  ⇒キャッシュアクセス時間Tcも増大
  ⇒Teが減るとは限らない！
  TLは主記憶なので大きい
  ⇒高速の二次キャッシュを設ける



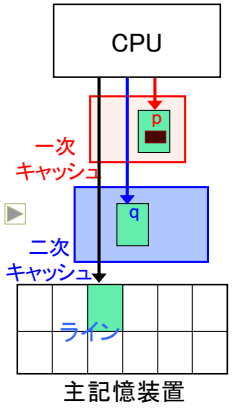
12

2階層キャッシュ

- キャッシュメモリ(チップ内)と主記憶との間に高速の二次キャッシュを設ける

⇒一次キャッシュでミスヒットしても、二次キャッシュでヒットすれば、Teを縮めることができる!

- 二次キャッシュの容量は、数Mバイト(intel core i7ではL3に相当)

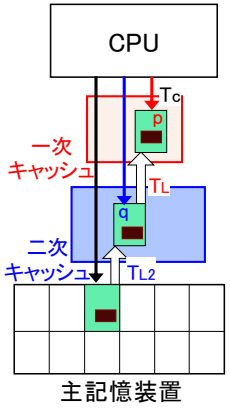


13

2階層キャッシュの実効アクセス時間

- 2階層キャッシュのアクセス手順

一次キャッシュ
ヒット
一次キャッシュラインをアクセス
ミスヒット
二次キャッシュ
ヒット
一次キャッシュにラインをコピー、
一次キャッシュラインをアクセス。
ミスヒット
主記憶ラインを、二次キャッシュに、その二次キャッシュラインを
一次キャッシュに、コピー。
一次キャッシュラインをアクセス。



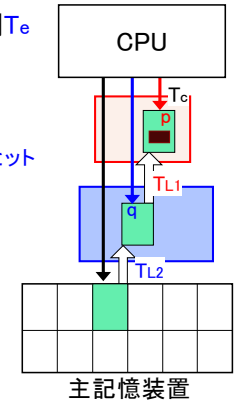
14

2階層キャッシュの実効アクセス時間

- 2階層キャッシュの実効アクセス時間Te

$T_e = pT_c$ 一次ヒット
$+(1-p)$ 一次ミスヒット
$\{q(T_c+T_{L1})$ 二次ヒット
$+(1-q)(T_c+T_{L1}+T_{L2})\}$ 二次ミスヒット
$= T_c+(1-p)T_{L1}+(1-p)(1-q)T_{L2}$

p:一次キャッシュのヒット率
q:二次キャッシュのヒット率
T_{L1}:二次から一次へのコピー時間
T_{L2}:主記憶から二次へのコピー時間



15

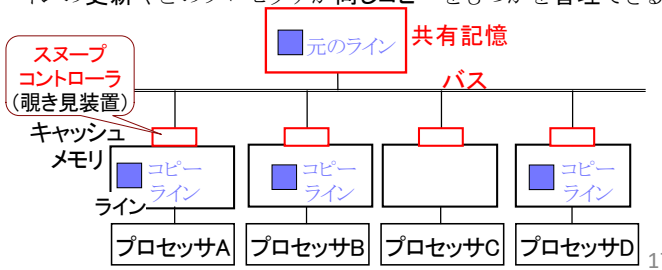
共有記憶型並列計算機

- 共有記憶型並列計算機では、複数個のプロセッサによる共有記憶へのアクセスが集中!
  ···· プロセッサ個々に共有記憶を分散させたり、
  プロセッサ個々にキャッシュメモリを設けて回避
- 共有バス方式
  共有バス上に複数のプロセッサを配置。
  スヌープキャッシュ法等がある。
- ディレクトリ方式
  プロセッサが一般のネットワーク上に配置。
  共有記憶は、各プロセッサ近くに分散配置(分散共有)され、
  各プロセッサに付加されたディレクトリが、各ライン情報をもつ。

16

スヌープキャッシュ

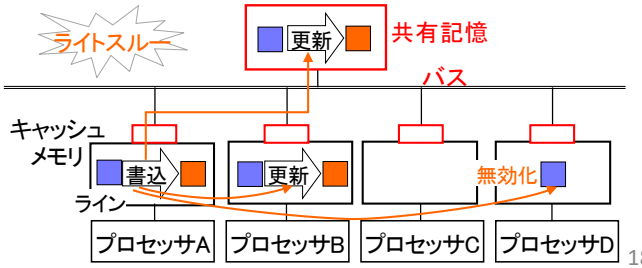
- バス接続の共有記憶に適した管理方式
- 共有記憶ラインを各プロセッサのキャッシュにコピーしアクセス
- 共有記憶へのアクセスやその番地等はバスを通して、すべてのプロセッサが同時に知ることができる(スヌープコントローラ)。
- 各プロセッサがバスを監視していれば、自分のもつキャッシュラインの更新やどのプロセッサが同じコピーをもつかを管理できる。



17

書き込み-ライトスルー

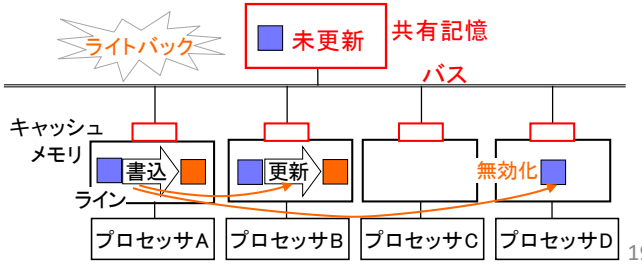
①プロセッサAのラインに書き込み
②共有記憶のラインを更新
②'他のプロセッサのキャッシュ中のラインは、
更新するか、無効化する。



18

書き込み-ライトバック①

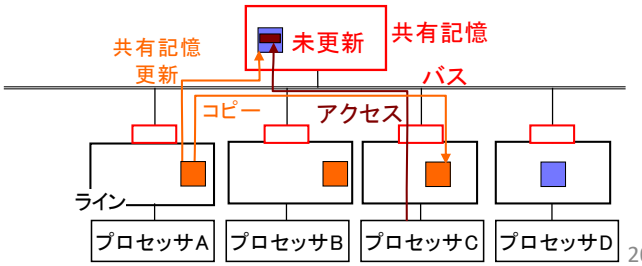
①プロセッサAのラインに書き込み
②他のプロセッサのキャッシュ中のラインは、
更新するか、無効化する。



19

書き込み-ライトバック②共有記憶更新

①プロセッサCがキャッシュにないラインを共有記憶にアクセス。その共有記憶ラインが未更新のとき、
②最新のラインをもつプロセッサAのキャッシュからコピー
③プロセッサAのキャッシュラインで共有記憶ラインを更新



20

Intel Core i9-12900KS Q1'22 specifications

- Core i9, 16core, 3.4GHz, 128GB

  L1キャッシュ
  命令キャッシュ16x32KB, 8way set associative
  データキャッシュ16x48KB, 8way set associative

  L2キャッシュ 10x1.25MB, 8way set associative

  L3キャッシュ 30MB, 16way set associative

  ラインの大きさ 64B

21