テクニカルノート

Zキャッシュ：オンチップマルチプロセッサ用キャッシュ

寺澤卓也†

オンチップマルチプロセッサでは，チップ内のキャッシュ間データ転送とチップ外の主記憶とのデータ転送の速度差が大きいため，主記憶との間の転送を極力避けることが望ましい．本論文ではキャッシュラインのチップ外への追い出し量を減少させる手法として，追い出しの対象となるラインを一時的に保存しておくキャッシュ機構（Zキャッシュ）を提案する．命令レベルシミュレーションによる簡単な評価の結果，アプリケーションによっては実行性能を最大20％程度改善できることが明らかになった．

Z-cache: A Cache Mechanism for On-chip Multiprocessors

TAKUYA TERASAWA†

In future on-chip multiprocessors, the access frequency of the off-chip main memory must be minimized considering the large gap of latency compared with quick on-chip snoop cache. In order to reduce the number of replace, an extra small cache (Z-cache) which holds the replacing line is added. Instruction level simulation results demonstrate that the Z-cache improves the performance up to 20% in some applications.

1. はじめに

デバイス技術と実装技術の発達にともない，近い将来には主記憶を除いた共有バス結合型マルチプロセッサ全体を1チップ上に搭載することが可能になるといわれている．この場合，チップ内のバスの動作速度は従来のバックプレーン上のバスに比べてはるかに高速になり，チップ外の主記憶のアクセス時間とのギャップが大きくなることが予想される．

そこで，このような1チップ上のマルチプロセッサ（ここではオンチップマルチプロセッサと呼ぶ）のスヌープキャッシュでは，極力チップ外への転送を抑える必要がある．また，チップ上のキャッシュでは，チップの実装密度が大きくなったとしても，従来のボード上に構成するキャッシュに比べてその容量は制限されることが予想される．このため，チップ内のキャッシュをできる限り効率的に利用する必要がある．以上を目的としたキャッシュプロトコルはいくつか提案されているが[1),2)]，プロトコルの改善だけでは性能の向上はある程度の範囲に制限される．

オンチップマルチプロセッサでは，他のプロセッサのキャッシュといえども同一チップ上に存在するため，チップ外の主記憶に比べればはるかに短時間にアクセスすることができる．また，チップ内のバス上に小規模な高速共有キャッシュを設けることも容易である．本論文では，このようなオンチップマルチプロセッサの性質を利用し，スヌープキャッシュの性能をさらに向上させる機構Zキャッシュを提案する．

† 東京工科大学 情報通信工学科
Department of Information Networks, Tokyo Engineering University

2. スヌープキャッシュプロトコル

本研究の基本となるスヌープキャッシュプロトコル[2)]は無効化型のプロトコルであり，主記憶と一致するラインに対してもOwnershipを持たせることで主記憶との転送を最小化している．各ラインは有効/無効（V/I），Ownershipを持つ/持たない（O/N），主記憶と一致する/異なる（C/D），他にコピーが存在しない/存在する可能性がある（E/S）の4 bitのタグにより**図1**に示す6状態（I (Invalidate)，CEO (Clean Exclusive Owned)，DEO (Dirty Exclusive Owned)，CSO (Clean Shared Owned)，DSO (Dirty Shared Owned)，S (Shared)）を持つ．あるキャッシュでミスが生じた場合に，同一ラインのコピーが他のプロセッサのキャッシュに存在すれば，その内容が主記憶と一致するかどうかにかかわらず，必ずキャッシュから転送が行われる．これにより主記憶との転送をできる限り避ける．

また，ラインがリプレイスされる場合も，書き戻しを可能な限り避けることによりさらに主記憶との転送を減らす．すなわち，Cleanなラインはもちろん Dirtyなラインでも Ownershipを持たない場合（S状態）については，単に書き潰され，書き戻しは発生しない．

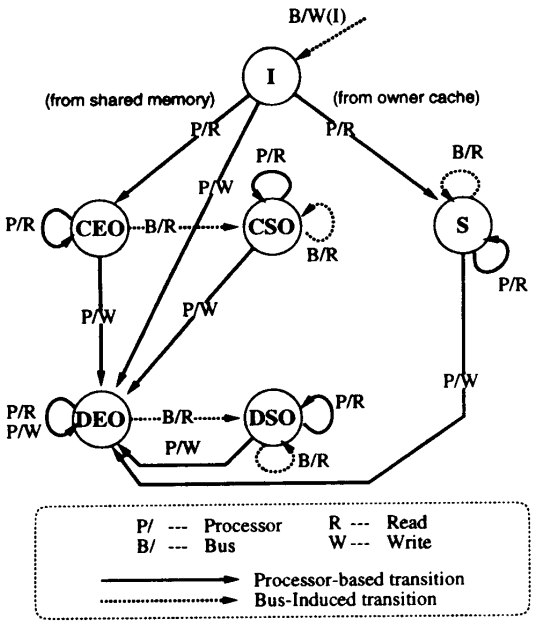

図1 オンチップマルチプロセッサ用キャッシュプロトコル
Fig. 1 A cache protocol for on-chip multiprocessor.

Dirty Owned なライン（DSO もしくは DEO 状態）についても他のキャッシュに同一ラインが存在する場合は，そのキャッシュに Ownership を移転することで書き戻しを避ける．結果として共有メモリに対して書き戻しを行うのは DEO 状態のラインが追い出される場合のみである．Ownership の制御は専用の信号線を用いて行う．

3. Zキャッシュ

3.1 Zキャッシュ-タイプ1

上記のキャッシュはスヌープキャッシュのプロトコルの工夫により実現できる範囲ではほとんど限界までチップ外との転送を減らしている．しかし，チップ内のキャッシュ容量の制限により，容量ミスによるラインのリプレイスが生じるため性能の向上が制限される．

バックプレーンやオンボードのバスで接続されたマルチプロセッサとは異なり，オンチップマルチプロセッサでは，他のプロセッサのキャッシュも同一チップ内にあり，主記憶に比べればはるかに小さいコストでアクセスすることが可能である．そこで，他のプロセッサのキャッシュの無効状態（I）の領域を，チップ外に追い出すキャッシュラインのバッファとして利用する方式，Z（Zombie）キャッシュ-タイプ1を提案する．

図2 にこの方式を示す．ミスにより，あるラインをチップ外へ追い出す必要が生じた場合，当該キャッシュは他のキャッシュの空きを探し，いずれかのプロセッサのキャッシュに無効領域が存在すれば，追い出すラインは，主記憶ではなく無効領域に対して転送される．無効領域の候補が複数のプロセッサに存在する場合，ラウンドロビン等の方法で利用する順番を決めておく．図1のプロトコルでは，ミスした場合に他のキャッシュに当該ラインが存在すれば，ラインの状態にかかわらずキャッシュ間転送が行われる．したがって，無効領域に転送されたラインが追い出したプロセッサにより再びアクセスされた場合，そのラインはキャッシュ間転送で得ることができる．

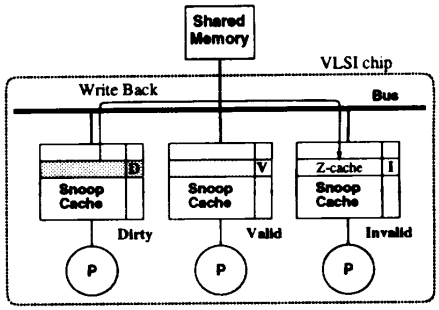

図2 Zキャッシュ-タイプ1
Fig. 2 Z-cache type 1.

3.2 Zキャッシュ-タイプ2

Zキャッシュ-タイプ1は，他のプロセッサの無効領域を利用することにより，キャッシュの利用効率を上げることができる．しかし，キャッシュの容量が全体的に不足する場合は，無効領域は接続しているプロセッサにより速やかにラインの読み込みに使われるため，利用することのできる無効領域は少なく，効果も制限される．そこで，チップ外に追い出すキャッシュラインを一時的に保存しておくために，チップ内のバス上に小容量の共有キャッシュを設ける．バックプレーンやオンボードの共有バスとは異なり，オンチップマルチプロセッサでは，共有キャッシュはチップ内に設けることができるので，主記憶と比べてはるかに高速なアクセスが可能である．この方式をZキャッシュ-タイプ2と呼ぶ．

図3 にZキャッシュ-タイプ2を示す．あるプロセッサのキャッシュラインを追い出す必要が生じたとき，ラインはまずZキャッシュに格納される．このときZキャッシュに空きがなければ，Zキャッシュ中のラインがFIFO等のアルゴリズムに従って選択され追い出される．各プロセッサは自分のキャッシュがミスした場合，まずZキャッシュをアクセスし，ヒットすればそこからラインを貰い，Zキャッシュの該当領域を無効化する．

Zキャッシュ-タイプ2の考え方は，ユニプロセッサのダイレクトマップキャッシュの性能を向上させるため

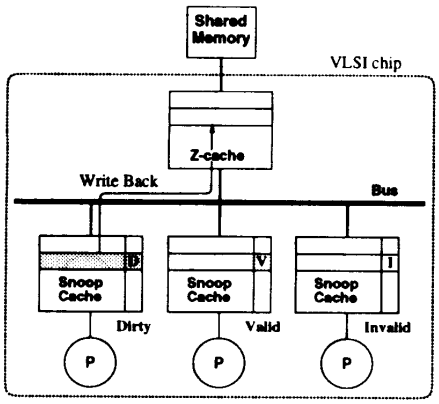

図3 Zキャッシュ-タイプ2
Fig. 3 Z-cache type 2.

表1 シミュレーション条件
Table 1 Simulation conditions.

プロセッサ数	1, 2, 4, 6, 8
スヌープキャッシュサイズ	8 KB (1 プロセッサあたり)
ラインサイズ	16 byte
way 数	4

に提案されたVictim Cache[3]と似ているが，Zキャッシュ-タイプ2はオンチップマルチプロセッサの特殊性を利用して，共有キャッシュとして用いている点が異なる．また，Zキャッシュはメモリ階層的には一般の共有二次キャッシュと同じ位置に存在するが，主記憶に書き戻されるラインの一時的なバッファであるため，ミス時にZキャッシュに存在しないラインを主記憶からロードする際にはキャッシュとして機能しない．このため，Zキャッシュは小容量でよい．

4. 評 価

Zキャッシュ-タイプ1およびタイプ2の効果を調べるために，命令レベル並列計算機シミュレータMILL[4]を用いて評価を行った．シミュレータ上で実行するアプリケーションとしては並列アプリケーション集SPLASH[5]中のMP3D，Choleskyおよび，並列論理シミュレータLogique[6]の3種類を選択した．評価は，(1) アプリケーション実行時間の短縮，(2) 共有主記憶への書き戻し量の変化について，Zキャッシュが存在しない場合に対して比較した．**表1** にシミュレーション条件を示す．

4.1 Zキャッシュ-タイプ1

表2 にZキャッシュがない場合の実行時間を1としたときのZキャッシュ-タイプ1を用いた場合の相対実行時間を，**表3** にプロセッサ数を2から8まで変化させたときの，Zキャッシュがない場合に対する書き戻し量の変化（減少率）を示す．MP3Dでは書き戻し量の減少が顕著で．性能が最大16%改善されるが，他のアプリケーションではほとんど効果は見られない．

表2 Zキャッシュ-タイプ1の相対実行時間
Table 2 Relative execution time of Z-cache type 1.

プロセッサ数	2	4	6	8
MP3D	1.00	0.99	0.93	0.86
Cholesky	1.00	0.99	1.03	0.92
Logique	0.99	1.02	0.99	1.00

表3 主記憶への書き戻し量の変化（タイプ1）
Table 3 Amount of write-backs (type 1).

	MP3D	Cholesky	Logique
書き戻し量の減少率（%）	0.4～32	0.1～2.5	−1.2～2.5

表4 受け入れ率（%）
Table 4 Acceptance ratio (%).

プロセッサ数	2	4	6	8
MP3D	5.29	22.3	37.8	54.6
Cholesky	2.6	9.4	14.2	18.3
Logique	0.9	2.4	5.1	6.9

表5 Zキャッシュ-タイプ2の相対実行時間
Table 5 Relative execution time of Z-cache type 2.

プロセッサ数	2	4	6	8
MP3D	0.99	0.98	0.96	0.94
Cholesky	0.99	0.94	0.99	0.97
Logique	0.87	0.88	0.85	0.83

表4 はラインを追い出す際に他のキャッシュ中の無効なラインを探し，受入先が見つかった割合（受け入れ率と呼ぶ）を示している．いずれのアプリケーションでもプロセッサ数が増えるに従って潜在的な受入先が増えるため，受け入れ率は増加する傾向がある．MP3Dでは共有ラインが多く頻繁に無効化も生じるため，受け入れ率が高い．これに対し，Logique，Choleskyでは容量ミスが頻繁に起こるため無効化されているラインが少なく，受け入れ率が低い．これがZキャッシュ-タイプ1がMP3D以外で効果を発揮しない原因であると考えられる．

4.2 Zキャッシュ-タイプ2

表5 にアプリケーションごとにZキャッシュがない場合の実行時間を1としたときのZキャッシュ-タイプ2を用いた場合の相対実行時間を示す．Zキャッシュは4wayセットアソシアティブでサイズは4KBである．それぞれ，性能比ではMP3D，Choleskyで1～6%，Logiqueで13～20%の向上が見られる．

また，**表6** はプロセッサ数を2から8まで変化させたときの，Zキャッシュがない場合に対する書き戻し量の変化（減少率）を示している．特にキャッシュの容量ミスの大きいLogiqueでは主記憶への書き戻し量がZキャッシュのないときに比べて大きく減少して

表6 主記憶への書き戻し量の変化（タイプ2）
Table 6 Amount of write-backs (type 2).

	MP3D	Cholesky	Logique
書き戻し量の減少率（%）	7~15	8~10	39~61

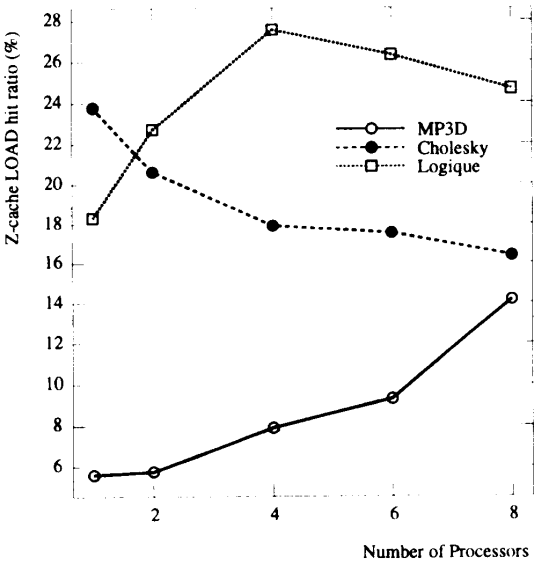

図4 Zキャッシュからのロードのヒット率
Fig. 4 Load hit ratio from Z-cache.

いる．

図4は主記憶からのキャッシュラインの読み込み操作がZキャッシュにヒットした割合（ロード時のヒット率）を示している．ロード時のヒット率は3アプリケーションで異なった傾向を示し，興味深い．

MP3Dではデータの共有率が比較的高いため，プロセッサ数が増加するのに伴い，追い出しの対象となるラインがスタック領域のラインである場合が多くなる．しかも，Zキャッシュへの書き戻しが少ないため，Zキャッシュ中のラインは生存時間が長い．したがって，プロセッサ数の増加に伴い，ライン読み出しのZキャッシュでのヒット率が上昇するものと思われる．Choleskyでは扱うデータが大きく，キャッシュの容量が絶対的に不足するため，プロセッサ数が増加するに従って扱うデータの範囲も広がり，Zキャッシュの容量不足が生じてヒット率が低下する．Logiqueではプロセッサ数に応じて問題を分割するため，プロセッサ数が少ないうちは各プロセッサが担当する問題量が大きいためZキャッシュは有効に働く．しかし，プロセッサ数が増えるとZキャッシュにも容量ミスが生じるようになりヒット率が低下する．

5. まとめ

簡単な評価の結果，Zキャッシュ-タイプ1はMP3Dのような共有率の高い，無効化が頻繁に起こるようなアプリケーションで有効であり，Zキャッシュ-タイプ2は容量ミスが支配的な場合に有効であることが分かった．さらに詳細な評価および両者の組合せは今後の課題である．

参考文献

1) 高橋真史，高野裕之，鈴木清吾，田胡治之：オンチップマルチプロセッサのアーキテクチャ検討，電子情報通信学会技術研究報告，CPSY95-3, pp.17-24 (1995).
2) Terasawa, T. and Amano, H.: A Cache Coherency Protocol for Multiprocessor Chip, *Proc. 7th IEEE Intl. Conf. on Wafer-Scale Integration*, pp.238-247 (1995).
3) Jouppi, N.P.: Improving Direct-Mapped Cache Performance by the Addition of a Small Fully-Associative Cache and Prefetch Buffers, *Proc. 17th Intl. Symp. on Computer Architecture*, pp.364-373 (1990).
4) Terasawa, T. and Amano, H.: Performance Evaluation of the Mixed-protocol Caches with Instruction Level Multiprocessor Simulator, *Proc. IASTED Intl. Conf. on Modeling and Simulation*, pp.1-5 (1994).
5) Singh, J.P., Weber, W. and Gupta, A.: SPLASH: Stanford Parallel Applications for Shared-Memory, Tech. Report, Computer System Laboratory, Stanford University (1992).
6) 工藤知宏，木村哲郎，天野英晴，寺沢卓也：問い合わせに基づく並列論理シミュレーションアルゴリズム，電子情報通信学会論文誌，Vol.J75-D-I, No.4, pp.221-231 (1992).

(平成7年11月7日受付)
(平成8年2月7日採録)

寺澤 卓也（正会員）

1967年生．1989年慶應義塾大学大学理工学部電気工学科卒業．1991年同大学大学院理工学研究科計算機科学専攻修士課程修了．1994年同博士課程単位取得退学．工学博士．現在，東京工科大学情報通信工学科講師．並列計算機アーキテクチャ，シミュレーション等の研究に従事．IEEE会員．