共有バスで接続されたマルチプロセッサのキャッシュメモリ構成†

福永　泰†† 坂東忠秋†† 平沢宏太郎††
加藤　猛††† 井手寿之†††

モジュール性のよい共有バスで互いに接続されたマルチプロセッサ方式におけるキャッシュメモリの構成について考察した．検討の中心は，プロセッサ個別にローカルキャッシュメモリを設けた場合，各プロセッサの高性能化と，共有バスの負荷低減とのトレードオフを，とくにオンラインリアルタイムで動作するというきびしい条件で定量評価したことである．その結果，(1) 16 バイトの比較的小さいブロックサイズのキャッシュメモリをプロセッサ側に実装することにより，実効メモリリードサイクル数を 20% に減少でき，かつバスの負荷を 30% に減少できること，(2)オンラインでタスクスイッチが 2 ms に 1 回の割合で発生する環境においても，実効性能が 1 MIPS の計算機では，90% 以上のヒット率が得られること，を明らかにした．

1. はじめに

半導体技術の進展により，LSI 1チップに集積できるゲート数は，3年に2倍程度という高率で向上し，32 ビットのプロセッサを1チップに実装することも可能となってきた[1]．このため，こうしたプロセッサを複数個用いて並列に動作させ，全体のスループットを増大させようとするマルチプロセッサ方式，とくに，共有バス構成によって，各プロセッサを付加したり，取りはずしたりすることが比較的容易なモジュール化を図ったマルチプロセッサ方式が実現されている[2]．

こうしたバス構成を用いたマルチプロセッサのアーキテクチャ研究として考慮しなければならない点は，

(1) VLSI 化によって，メモリ容量やゲート数が増大することはそれほど問題とならないが，接続バスの信号線の増大は，実装を困難にすることから避けなければならないこと，

(2) 演算部の能力は，方式上の工夫や，LSI の高速化により，ますます向上するのに対し，メモリアクセス時のオーバヘッドが問題となり，それによる性能低下をおさえる必要があること，

があげられる．たとえば，LSI 11 や，インテルの 8080 でさえ，メモリバスの負荷は 40〜70% というデータ[3]もあり，最近のように，より高速な 16 ビット，32 ビットプロセッサでは，バスの負荷はさらに大きくなっていると予想される．このため，こうしたプロセッサを複数台接続した場合は，バスネックのため，プロセッサの性能が十分出ないことは明らかである．

こうした問題は，大型計算機についても，性能向上に伴い発生し，高速のメモリの開発や，高速転送が可能な太いバスを設けること等で問題解決が図られてきたものである．しかしながら，こうした解決手段のなかには，物理的規模の小さい小型計算機には採用できないものも多く，本報告では，とくに制御用計算機や，高級パーソナルコンピュータのように実装規模の小さい CPU に対し，メモリアクセスの応答性，スループットをあげる方式について検討した．以下でその内容につき報告する．

2. マルチプロセッサシステム構成と問題点

図1に，バス構成を採用した一般的なマルチプロセッサのシステム構成図を示す．複数のプロセッサは，1本の共通バスを介して，主メモリを共有し，並列に割り当てられた仕事を実行するように制御される．共有バスはプロセッサ相互の連絡用としても使用されるが，第1の目的は，プロセッサとメモリの間のコマンド，データの転送用として使用される．共有バスのメリットとしては，各プロセッサを自由に接続したり，取りはずしたりできること，および接続用のハードウェアの信号線本数が少なく，コンパクトな構成が可能であることがあげられる．

本構成を採用した場合，各プロセッサが，たとえば

† Cache Memory for Bus-Structured Multiprocessor System by YASUSHI FUKUNAGA, TADA'AKI BANDOH, KOTARO HIRASAWA (Hitachi Research Laboratory, Hitachi, Ltd.), TAKESHI KATO and JUSHI IDE (Omika Works, Hitachi, Ltd.).

†† (株)日立製作所日立研究所
††† (株)日立製作所大みか工場

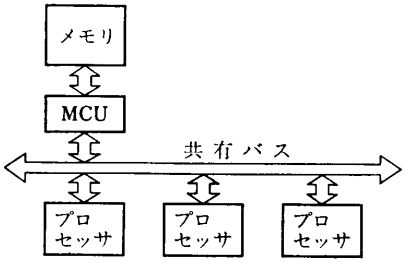

図 1 バス構造を有したマルチプロセッサ
Fig. 1 Multi-processor system interconnected by a common bus.

メモリを読む場合，下記手順が必要である．

（1） プロセッサからのバス占有要求
（2） プロセッサからメモリ制御装置 (MCU) へのコマンド，アドレス情報の転送
（3） メモリ制御装置からメモリへのアクセス
（4） メモリ制御装置からのバス占有要求
（5） リードデータのプロセッサへの転送

以上の手順で，(2)から(4)の間は，他のプロセッサへ共有バスを解放できるため，バスのスループットを増大させることができる．

上記のような複数ステップが必要なメモリリード手順では，プロセッサ内部の性能が向上しているため，メモリアクセス待ちの間，プロセッサの動作が停止してしまう．たとえば，オーソゴナルな命令体系を有する計算機のベンチマークプログラム平均の1命令当りのメモリリード回数，ライト回数はそれぞれ 2.04, 0.23 回であるというデータが得られている．一方，10 万ゲート弱で実現されたハードウェアで，上記命令体系を実行しようとすると，平均 2〜3 マイクロステップ数で1命令を実行できる．このため，2.04 回主メモリをアクセスするのに要するマシンサイクル数によるオーバヘッドは，たとえば，上記リード手順をおのおの1マシンサイクルで実行するとしても 10.20 マシンサイクルとなり，2〜3 マイクロステップの5倍弱の間プロセッサが停止することにもなりかねない．

こうした現象を防止するため，キャッシュメモリやローカルメモリを付加する方式が考えられている．ローカルメモリの場合は，あらかじめ，ローカルメモリに置いておくデータを，性能の面を考慮しながらソフトウェアによって決定しておく必要がある．そのため，オペレーティングシステム用のようにあらかじめ，処理手順に対する考察が十分な場合は，よく使用されるデータをローカルメモリ上に置くことが可能であるが，ユーザプログラムのデータ記憶部としては不適当と考えられる．そこで，ここではとくに共有バスの転送オーバヘッドを少なくし，かつ，バス要求時の衝突を少なくするため，プロセッサ側に置いたローカルキャッシュメモリ方式について検討した．

以下，3章ではローカルキャッシュメモリの構成について，とくに検討課題を明らかにするため説明し，4章では，シミュレーションによる検討結果について説明する．

3. ローカルキャッシュ構成

ローカルキャッシュメモリを有したプロセッサのブロック図を図2に示す．プロセッサが要求したデータがキャッシュメモリ上にコピーされていない場合は，主メモリから共有バスを介して，プロセッサが要求したデータを含む一つのブロックがキャッシュメモリへ転送される．一方，上記データがキャッシュメモリ上にある場合は，そのデータを使用することができるため，共有バスを使用する必要はなくなる．他プロセッサが主メモリデータを書き込む場合，そのブロックがキャッシュメモリ上へコピーされていると，主メモリとキャッシュメモリ内のデータの不一致が発生する．これを避けるには，主メモリへのアクセスの情報は，必ず共有バス上を通過するため，キャッシュメモリは共有バスを常時監視しておき，該当ブロックがキャッシュメモリ上にコピーされていれば，そのブロックの内容を無効にする制御を行う必要がある．この制御方式は，すべてのメモリアクセスが1本の共有バス上を必ず通過するという特質から，キャッシュメモリのタグフィールドをプロセッサとバスで時分割して使用することで可能となる．

さらに，プロセッサからのメモリライトについては，以下の二つの方式が考えられた．

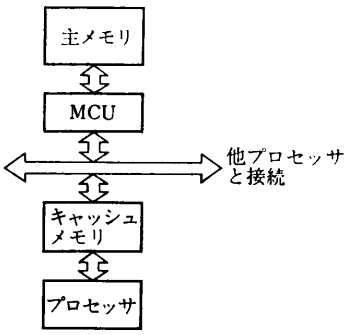

図 2 ローカルキャッシュメモリを有したプロセッサ構成
Fig. 2 Processor with local cache memory.

(1) キャッシュメモリにデータを書き込む際，同時に主メモリ上にも書き込むように制御するライトスルーといわれる方式．本方式を採用すればつねにメモリの内容とキャッシュメモリの内容を一致させることができるが，つねに主メモリに書き込みにいくため，ライトサイクルが長くなる．

(2) キャッシュメモリにのみライトデータを書き込んでおいて，書き込んだキャッシュメモリを置換する必要性が生じたときのみ，主メモリにデータを書き込むライトスワップといわれる方式．キャッシュアクセスタイムと，主メモリアクセスタイムの比が 1：4 以上の場合，本方式の方が性能がよいといわれている．

(2)の方式のほうが，一般的に性能がすぐれているが，他のプロセッサからのメモリライトが行われた場合，データを一致させる制御が複雑であるため，ここでは(1)の方式を採用することで検討を進めた．

以上のような構成を採用したキャッシュメモリに対しては，キャッシュ容量，ブロックサイズ，セット数等のパラメータに対し，文献4)～7)で検討されているため，ここではとくに下記2点について定量的に評価した．

(1) プロセッサのメモリアクセス空間が変換されるとキャッシュのヒット率が低下するため，アドレス空間が頻繁に変更されるとキャッシュメモリの効果が出ないおそれがある．このため，タスク切換えが頻発するオンライン処理においても，プロセッサからの実効メモリアクセス時間が低減可能であるかどうか．

(2) キャッシュメモリをプロセッサ側に設けることで，各プロセッサからの共有バスの負荷を低減できるかどうか．

以上，2点につき，シミュレーションによって性能を調査，評価した．

4. シミュレーション結果

シミュレーションの入力データとしては，ハードウェアモニタによって収集した数万ステップに及ぶアドレストレース結果を用いた．ハードウェアモニタを使用したことにより，各プログラムの実行過程を正確にトレースできるため，実際の動作に近い状況でシミュレーションが可能となる．

アドレスデータとしては，リードアドレスのみを用い，キャッシュメモリのセット数，ブロックサイズ，容量をパラメータとしてキャッシュメモリのヒット率を求めることにした．ライトアドレスを使用しない理由は，ライトスルー方式の場合，ライトのアクセスは，キャッシュメモリのヒット率には影響しないからである．またシミュレーションには，入出力装置によるメモリアクセスのキャッシュヒット率への影響は考慮されていない．これは，(1)入出力装置によるメモリアクセスの頻度は，プロセッサからのメモリアクセスの頻度と比較して少ないこと，(2)入出力動作に使用するメモリ空間とプロセッサが使用するメモリ空間とは基本的に異なるため，入出力装置で使用するメモリ空間がキャッシュメモリ上にコピーされている場合が少ないこと，の二つの理由による．

シミュレーションにおいては，3章で述べた二つの目的に対応できるように，キャッシュメモリ上に何も有効なデータがない状態から，プロセッサのメモリアクセス回数が進むにつれてキャッシュヒット率がどのように変化するかを調べた．パラメータとしては，キャッシュメモリの容量，セット数，ブロックサイズを使用したが，前2者については，文献4)～7)に示されているため，本論文ではとくにブロックサイズをパラメータとした場合の結果を示す．セット数，キャッシュメモリ容量は2，および 8kB に固定してある．

図3は，キャッシュメモリに有効なデータがないと

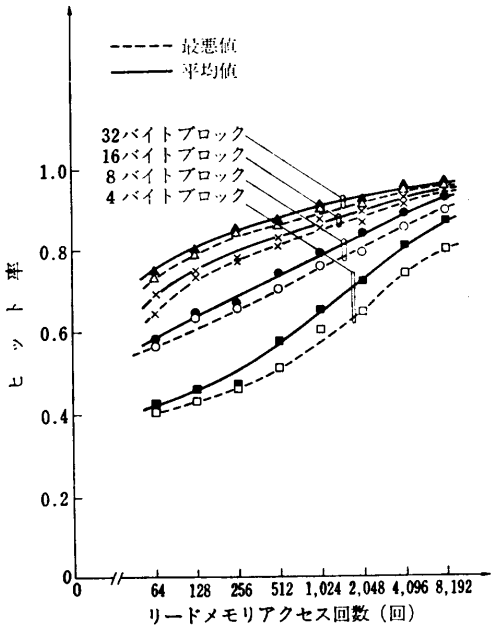

図3 ヒット率の立上り形態
Fig. 3 Hit ratio transition.

きから，プロセッサによるメモリアクセスが進むにつれて，どのようにヒット率が向上するかを，シミュレーションデータの最悪時と，平均値について示したものである．ヒット率は，累積ヒット率として示している．ブロックサイズが小さいときは，プログラムによって大きなばらつきがあるが，ブロックサイズが大きくなると，ばらつきが小さくなっていることがわかる．これは，同一のアドレスを何度もアクセスする可能性は，各プログラムの構成によって大きく依存しているが，近傍のアドレスをアクセスする可能性は，コンパイラ等の構成によって，あまり各プログラムに依存しない性質であるためと予想される．

図4は，図3のヒット率を平均実効メモリリードサイクル数に変換したもので，図3の中の平均ヒット率をもとに次式で平均実効メモリリードサイクル t_{ma} を求めた．

$$t_{ma}=h_r(n_{mr})+(1-h_r(n_{mr}))\cdot(t_a-1+s_b/4) \quad (1)$$

ここに $h_r(n_{mr})$ は，メモリリード回数 n_{mr} 時までの平均ヒット率，t_a は4バイトブロックサイズ時の主メモリリードサイクル数，s_b はブロックサイズ（バイト）である．キャッシュメモリのアクセスタイムは1マシンサイクルとし，共有バスは1マシンサイクルに4バイトのデータが転送できるものと仮定した．このため，s_b バイトを転送するには $s_b/4$ マシンサイクル

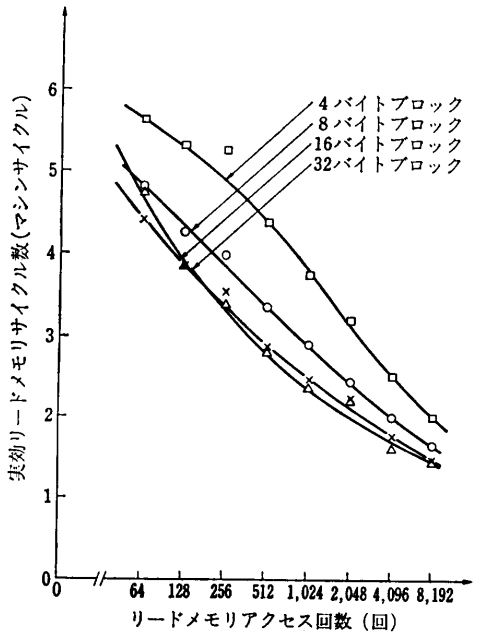

図4 リードメモリサイクル数の立下り形態
Fig. 4 Read memory cycle number transition.

数がバス上のデータ転送に必要である．これより，キャッシュミス時の主メモリアクセスタイムは $(t_a-1+s_b/4)$ サイクル必要であり，平均メモリリードサイクル数は(1)式で求められる．t_a として9マシンサイクルを代入したものが図4のグラフである．

共有バス幅を4バイトとしたのは，CPUが32ビット（4バイト）バスであるためである．

図4より，メモリリード回数にかかわりなく，16バイトブロックサイズまではブロックサイズが大きくなればなるほど，実効メモリリードサイクル数は向上している．しかしながら16バイトブロックサイズと32バイトブロックサイズでは，実効メモリリードサイクル数は，ほとんど相違しないことが明らかである．これは，ヒット率はブロックサイズが大きくなればなるほど向上するが，共有バスのバス幅に物理的な制限があるため，あまりブロックサイズを大きくしても，実質的な性能に影響する実効メモリリードサイクル数は小さくならないことを示している．とくに今後，ハードウェアの規模が小さくなると，バスの本数を多くできなくなるため，適切なブロックサイズの選択が必要であることを図4は示している．

一方，ブロックサイズの相違にかかわらず，実効メモリリードサイクル数は，メモリリードの回数が増加するにつれて，図に示すような割合で減少する．タスクスイッチが2ms以上に1回の割合で起きるリアルタイム制御系を仮定すると，実効性能1MIPSのCPUでは約4,000回のメモリアクセスでメモリ空間が変更され，キャッシュの内容が無効となってしまう．このとき，実効メモリサイクル数としては，16バイトブロックサイズで1.7マシンサイクルまで減少させることができる．1.7マシンサイクルに近い主メモリを設計することは困難であり，リアルタイム制御の環境のなかでも，キャッシュメモリの効果は十分得られることが明らかである．

一方，**図5**は，共有バスの負荷に関するグラフで，横軸は，図4と同様，リードメモリアクセス回数を示し，縦軸は，1リードアクセスによる平均データ転送の負荷を示す．負荷 l_{rd} は，ヒット率 $h_r(n_{mr})$ より，次式で得ることができる．

$$l_{rd}=(1-h_r(n_{mr}))\cdot(s_b+4) \quad (2)$$

ここで，ブロックサイズ (s_b) にさらに4バイトを加算したのは，アドレス転送にさらに4バイト必要としたためである．キャッシュメモリがない場合の1回のリードアクセスの転送量は，アドレス4バイト，デー

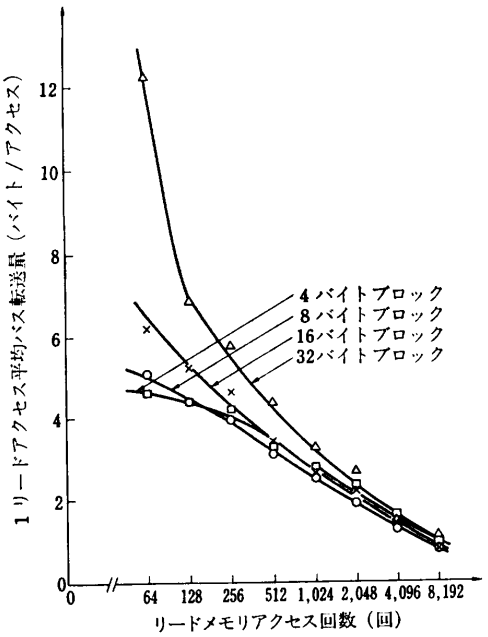

図 5 共用バス負荷の減少
Fig. 5 Degradation of common bus load.

タ4バイトの計8バイトであり，これより，以下のことが明らかになった．

（1） ブロックサイズが比較的少ないキャッシュメモリを導入することで，キャッシュメモリ-主メモリ間のデータ転送量を，キャッシュメモリ-プロセッサ間のデータ転送量より少なくすることが可能である．

（2） リード回数が128回を越えると，8バイトブロックサイズのバスの負荷が最小となり，16バイトのブロックサイズでも，それほど変わらない．しかしながら，32バイトのブロックサイズになると，約20〜30%バス負荷が大きくなる．

図4，図5より，主メモリとキャッシュメモリ間を4バイト幅のバスで接続した32ビットプロセッサではキャッシュメモリのブロックサイズを16バイトとする方式が，各プロセッサの性能を上げる上でも，プロセッサ間のデータアクセスの衝突を避ける上からも適切であり，ブロックサイズを大きくすると制御を複雑とするばかりか，性能も低下してしまうことが明らかとなった．

一方，キャッシュメモリを設けない場合と比較すると，メモリリード回数4,000回で平均的にタスクスイッチが発生した場合，リードデータによるバス負荷を20%以下に減少させることが可能で，またメモリアクセスタイムを1.7マシンサイクルまで高速化することが可能である．

以上，述べてきたように，32ビットプロセッサで，共有バス幅を4バイトとした場合，比較的少ないブロックサイズのローカルキャッシュメモリを有することが，バス構成のマルチプロセッサにおいて重要である．

5. 結　言

モジュール性のよいバス結合のマルチプロセッサ方式について，とくに性能，構成の単純化の点を中心に考察した．その結果，スループットが小さい共有バス方式を採用しても，各プロセッサ個別にローカルキャッシュメモリを設け，かつキャッシュメモリのブロックサイズを16バイトと小さくすれば，実効メモリリードサイクル数を約20%に減少でき，共有バスの負荷を30%に減少できることを明らかにした．また実効性能が1 MIPSの計算機を，2 msに1回の割合でタスクスイッチが発生するリアルタイム制御環境に使用しても90%以上のヒット率が得られることを明らかにした．

謝辞 本研究の機会を与えていただいた日立製作所日立研究所の高砂常義氏，大みか工場の桑原洋氏に深謝します．

参 考 文 献

1) Beyers, J. W. et al.: A 32b VLSI CPU Chip, ISSCC 81 Proc., pp. 104-105 (1981).
2) Lewis, G. R. et al.: The BTI 8000—Homogeneous General-Purpose Multiprocessing, AFIPS Conf. Proc., Vol. 48, pp. 513-528 (1977).
3) Toong, H. D.: Multi-Microprocessor System, Center for Information System Research, MIT.
4) Gibson, D. H.: Considerations in Block-Oriented System Design, SJCC, pp. 75-80 (1967).
5) Kaplan, K. R. et al.: Cache-Based Computer System, IEEE Computer Society Repository Paper, R-72-215.
6) Strecher, W. D.: Cache Memories for PDP-11 Family Computers, 5th Annual Symposium on Computer Architecture, pp. 155-158 (1976).
7) Bell, J. et al.: An Investigation of Alternative Cache Organization, *IEEE Trans. Comput.*, Vol. C-23, No. 4, pp. 346-351 (1974).

(昭和58年6月14日受付)
(昭和58年9月13日採録)