二階層キャッシュをもつマルチプロセッサシステムにおけるアドレス変換機構の実装方式の考察†

山本 登††

処理装置群と主記憶装置群とを，マトリクス状のスイッチで接続した密結合マルチプロセッサシステムの実用化を阻む最大の要因は，スイッチの実現に膨大な量の回路が必要なことである．この解決のため，スイッチ装置内にキャッシュを設け，メモリ参照の応答時間を短縮することにより，主記憶インタフェースの数を削減し，スイッチに必要な回路の量を減らす方法が考えられる．本論文では，処理装置のほか，スイッチ装置にもキャッシュをもつ密結合マルチプロセッサシステムで，プログラムのアドレス空間を，実記憶のアドレス空間に写像する機構を，どこに設けるべきかを，これらのキャッシュとの関係で検討している．

1. まえがき

2，30台の処理装置群と記憶装置群とを，マトリクス状のスイッチで密に結合したシステムは，記憶装置を占有した場合に匹敵する高速の記憶システムを実現することにより，多数の処理装置による並列処理機能を生かし，電子計算機の応用分野をいっそう拡大する可能性をもっている．しかし，処理装置群と記憶装置群を接続するスイッチに，膨大な量の回路が必要なため[1),2)]，実用化されないまま今日に至っている．1970年代に開発されたC. mmpシステム[3),4)]は，それぞれ16台のミニコンピュータと記憶装置とを，16×16のマトリクス・スイッチで接続した，形式の上では本格的なマトリクス・スイッチ結合のマルチプロセッサシステムであった．しかし，当時の商用ミニコンピュータを用いたため制約が多く，現在の汎用大型電子計算機を上まわる密結合マルチプロセッサシステムの実用化に，寄与する点は少ない．この形式のシステムのもつ可能性を考えれば，実用化の促進に努力する価値は十分にある．しかし，スイッチ部の実現を容易にする手段として，これに必要な回路技術を発展させるアプローチよりも[2),5)]，電子計算機アーキテクチャの面から解決すべきと考え，マトリクス・スイッチ装置内にキャッシュを設け，実効的なメモリ参照速度を大きくすることにより，スイッチ装置の記憶装置インタフェースの数を削減する方法を考えた[6)]．このシステムでは，処理装置にキャッシュが内蔵されるほか，スイッチ装置にもキャッシュが内蔵される．このシステムの論理方式の設計過程で，これらのキャッシュは，論理アドレスと実アドレスのいずれで動作させるべきかを，検討する必要に迫られた．本論文ではその検討結果を報告するものである．

2. 検討対象システム

2.1 アドレス方式による分類

アドレス変換機能を，処理装置とスイッチ装置のどちらに組み込むか，また，アドレス変換を参照アドレス領域がキャッシュに存在するか否かの検査（デレクトリ検索）の前に行うか，同時あるいはそれ以降に行うかにより，次の四つの方式に分類できる．

（1） 実アドレス方式

処理装置がキャッシュを参照する前にアドレス変換をする方式で，このため，処理装置内のキャッシュ（固有キャッシュ）も，スイッチ装置内のキャッシュ（共有キャッシュ）も実アドレスで参照される．

（2） 論理・実アドレス方式1

処理装置では高速アドレス変換バッファ（TLB）によるアドレス変換と，キャッシュのデレクトリ検索が同時に行われる方式で，このため固有キャッシュは論理アドレスで，共有キャッシュは実アドレスで参照される．

（3） 論理・実アドレス方式2

アドレス変換機構はスイッチ装置に設けられ，共有キャッシュのデレクトリ検索に先立ち，アドレス変換が行われる．このため固有キャッシュは論理アドレス，共有キャッシュは実アドレスで参照される．

（4） 論理アドレス方式

共有キャッシュのデレクトリ検索と同時に，TLBによるアドレス変換が行われる方式で，このため固有

† A Consideration of an Implementation of an Address Translation Mechanism in a Multi-processor System with a Cache in a Crossbar Switch by NOBORU YAMAMOTO (Department of Electrical Engineering, Faculty of Engineering, Nihon University).

†† 日本大学工学部電気工学科

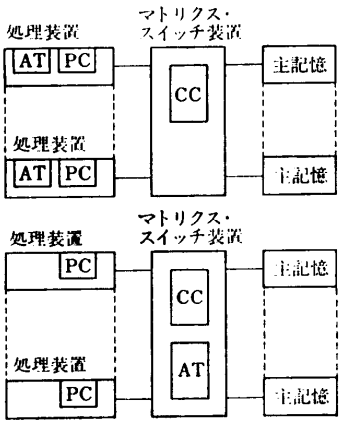


図 1 アドレス変換機構の実装方式
Fig. 1 The implementation of address translation mechanism.
PC: 固有キャッシュ, CC: 共有キャッシュ,
AT: アドレス変換機構

キャッシュ，共有キャッシュとも，論理アドレスで参照される．

2.2 検討対象システムの機能

本節では図1に示す検討対象システムの機能のうち，以下の検討で必要となるものについて述べる．

仕様 1 固有キャッシュのストア方式

各処理装置が書き変えるメモリの領域は，その処理装置固有の情報の場合もあるし，複数の処理装置で共有する場合もある．共有情報の場合，共有する他の処理装置が，書き変えられた内容を早期に利用できることが望ましい．このため，固有キャッシュにストアすると同時に，共有キャッシュにもストアする方式を採用する（ストアスルー方式）．

仕様 2 共有キャッシュのストア方式

スイッチ装置に内蔵されるキャッシュは，多くの処理装置から共用されるため，ストア動作も高速に行う必要がある．そこで，ストア動作は共有キャッシュのみになされる方式を採用する（ストアイン方式）．このため次の各場合，共有キャッシュの該当ブロックの内容は，主記憶に書き戻されねばならない．

（1） 新しい論理ブロックを共有キャッシュに収容するため，これまで共有キャッシュに収容されていたブロックと置換するが，この候補となったブロックが主記憶から転送後，その内容が書き変えられていれば，主記憶に書き戻さねばならない．

（2） 新しいページを実記憶に収容するため，ある論理ページが実記憶から置換されるとき，共有キャッシュに存在しているその論理ページに属するブロックのうち，主記憶から転送後内容が書き変わったものは，主記憶に書き戻されねばならない（そのまま残留させると，（1）に述べたブロック置換の場合，実記憶がないため書き戻しができなくなる）．

仕様 3 仮想記憶の消滅

仮想記憶に存在しない情報は，その存在を認められない．したがって，ジョブの終了により仮想記憶が消滅する場合，その空間に属するキャッシュのブロックは無効とされねばならない．

仕様 4 実ページからの消滅

実ページから抹消された論理ページに属するブロックは，仮にキャッシュ中にその論理ページに属するブロックが残存していても，アドレス変換の過程でページ不在となるため，実アドレスによる参照はできない．しかし，新たにその実ページに読みこまれた論理ページのブロックと区別がつかなくなるため，キャッシュが実アドレスで参照される場合，実ページから消滅する時点で，キャッシュに存在する該当論理ページに属するすべてのブロックは無効とされねばならない．

一方キャッシュが論理アドレスで参照される場合，書き変えのない命令コードはキャッシュに残留することができるが，演算数としては残留を許されず，仕様2の(2)項と仕様6の制約を受ける．

仕様 5 共有キャッシュの書き変え管理情報

任意の処理装置が共有キャッシュを書き変えたとき，その写しをもつ処理装置群に書き変えられたアドレス領域を示し，その内容が変わったことを知らせねばならない（バッファ合せと呼ぶ）．この機能を果たすため，共有キャッシュの各ブロックごとに，どの処理装置がその写しを保有しているかを示す情報（複写表示子）をもたせる．この方式では，共有キャッシュから消滅するブロックを固有キャッシュに残留することを許すと，共有情報の書き変えの管理ができなくなるため，共有キャッシュに存在しないブロックが固有キャッシュに存在することは許さない．ただし，読出ししかなされない命令コードには，この規則は適用されない．

仕様 6 演算数無効表示子

キャッシュに収容される各ブロックは，アドレス空間をキャッシュの一ブロックの大きさで機械的に区切ったものである．したがって，命令コードまたは演算数のみを含むブロックと，命令コードと演算数の両方を含むブロックができる．現在の電子計算機では，これらを識別する手段をもたないのが一般的であるが，

命令コードと演算数とは，前述のように扱われ方に大きな違いがあるので区別したい．そこで両キャッシュのデレクトリに，命令コードとしては使用可能であるが，セットされていれば演算数としては参照不可能で，キャッシュミスの扱いとなる演算数無効表示子を設ける．

3. アドレス方式の評価

3.1 メモリ参照速度の検討

図2はそれぞれの方式のメモリの参照の仕方を，読出し動作を例にして示したものである．図で，キャッシュのデレクトリ検索時間（T_{ds}），データ読出し時間（T_{cr}），およびアドレス変換時間（T_{at}）は，処理装置の場合がスイッチ装置の場合より，2倍速く行われるものとして示してある．

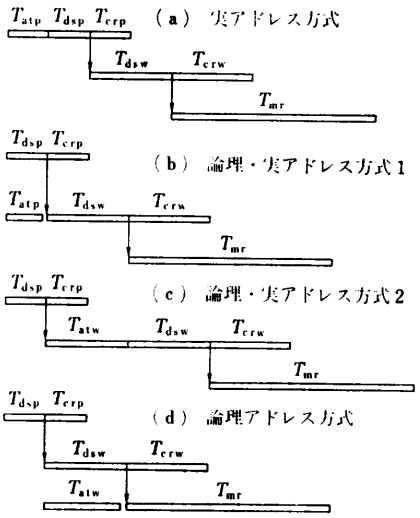
（1） 図2によれば，キャッシュのアドレスとして何を用いるかにより，読出し情報を得るまでに要する時間に差が生じている．すなわち，論理アドレスで参照する場合は事前にアドレス変換をする必要はなく，すぐにキャッシュのデレクトリ検索を開始でき，しかも図示のように，必要ならアドレス変換をデレクトリ検索と並行して行うことができるため，高速アドレス変換バッファ（TLB）による変換に成功すれば，アドレス変換の所要時間が表面に現れない．

（2） 論理・実アドレス方式2ではスイッチ装置内でアドレス変換をするが，アドレス変換時間が処理装置の場合の2倍となっているため，所要時間が長くなっている．このアドレス変換時間を処理装置の場合と同じにすれば，主記憶から読み出す場合も実アドレス方式と同じになり，しかも固有キャッシュは論理アドレスで参照されるため，ヒットすれば実アドレス方式より早く読み出すことができる．

（3） 論理・実アドレス方式1と論理アドレス方式は，アドレス変換機構を実装する装置は異なるものの，アドレス変換の所要時間が表面に現れない点は同じなので，図2では同じ結果となっている．

3.2 キャッシュの記憶領域の有効利用

記憶装置を階層的に構成したシステムの上位記憶装置（処理装置に近い）と下位記憶装置（処理装置から遠い）の情報の関係は，下位の記憶装置の情報の一部をすぐ上位の記憶装置が保有するという関係が，つねに成立するように考えられる．たしかに必要な情報は直接の下位記憶装置から受けねばならないが，いったん受けてしまえば，直接の下位記憶装置からは消滅しても，上位の記憶装置に残留を許すよう制御することは可能で，しかも残留できれば，設置されている限られた記憶領域を，より広く使用することとなるため，有利な方式といえる．この観点から，検討対象の四つのシステムが，どのような性質をもつかを以下で考察する．



凡 例

（1） 記号の説明

T_{at} アドレス変換時間
T_{ds} デレクトリ検索時間
T_{cr} キャッシュ読出し時間
T_{mr} 主記憶読出し時間

} サフィックスの末尾のpとwは，それぞれ処理装置またはスイッチ装置で行われる操作を示す．

（2） 矢印はキャッシュミスの場合行われる操作を示す．

図2 キャッシュ参照アドレスのちがいによる読出し速度のちがい
Fig. 2 Memory access time.

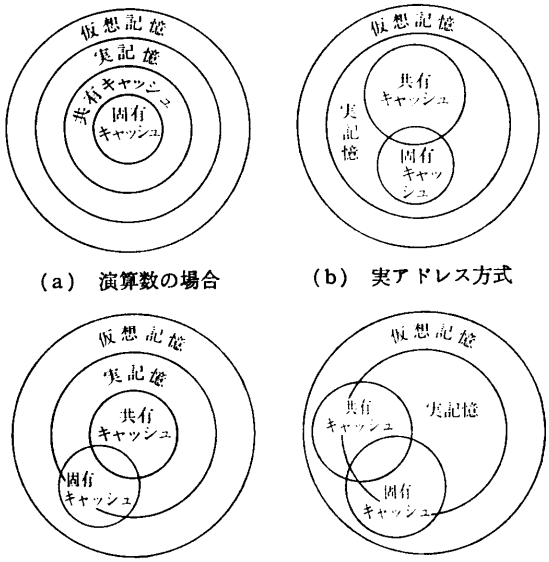
3.2.1 演算数の場合

仕様3により仮想記憶に存在しないものは，いずれのキャッシュにも存在できない．また仕様4により，実ページに存在しない領域は共有キャッシュに存在できない．また仕様5により共有キャッシュに存在しないブロックは固有キャッシュにも存在が認められないため，演算数については図3（a）に示すように，上位記憶装置は直接の下位記憶装置の完全な部分集合でなければならず，四つの方式とも差異はない．

3.2.2 命令コードの場合

（1） 実アドレス方式

共有キャッシュも固有キャッシュも実アドレスで参照されるので，実記憶に存在しない論理ページに属するブロックはキャッシュに存在できない．一方，属する論理ページが実記憶に存在する限り，共有キャッシュから消滅するブロックでも固有キャッシュに残留できるので，図3（b）のような包含関係となる．



(a) 演算数の場合　(b) 実アドレス方式

(c) 論理・実アドレス方式 1/2　(d) 論理アドレス方式

図 3 キャッシュの記憶域の有効利用
Fig. 3 Effective utilization of each cache.

(2) 論理実アドレス方式 1/2

固有キャッシュは論理アドレスで動作するため，実記憶に存在しない論理ページに属するブロックでも，固有キャッシュに残留することが許される．また共有キャッシュに存在しないブロックも固有キャッシュに残留が許される．一方，共有キャッシュは実アドレスで参照されるため，実記憶に存在しない論理ページに属するブロックは，残留を認められない．このため図3(c)に示す包含関係となる．

(3) 論理アドレス方式

この場合，共有キャッシュも固有キャッシュも論理アドレスで動作するため，実ページから消滅する論理ページに含まれるブロックでも，共有キャッシュと固有キャッシュに残留できる．また，共有キャッシュから消滅しても固有キャッシュには残留できるため，図3(d)に示す包含関係となる．

共有キャッシュや固有キャッシュなど，容量の限られたメモリに，より多くの情報を収容できることは望ましいことである．この観点からすれば，論理アドレス方式が最も優れており，ついで論理・実アドレス方式 1/2 がよく，実アドレス方式は最も劣っていることになる．

3.3 主記憶書き戻し操作の容易性

共有キャッシュはストアイン方式なので，新しいブロックの収容のため，共有キャッシュの任意のブロックを抹消するとき，仕様2により，実記憶に書き戻しが必要な場合がある．以下ではこの操作の難易を考察する．

(1) 実アドレス方式と論理・実アドレス方式 1/2 の場合，共有キャッシュは実アドレスで参照されるため，キャッシュのデレクトリ中のアドレスを上位アドレスとし，参照中のアドレスから列アドレス部を取り出し下位アドレスとし，これらの二つから書き戻しアドレスを作り，書き戻しを行えばよい．

(2) 論理アドレス方式では，共有キャッシュは論理アドレスで参照されるため，実記憶に書き出す場合，論理アドレスから実アドレスを求める必要がある．

3.4 バッファ合せの容易性

任意の処理装置が共有キャッシュの任意のブロックを書き変えるとき，そのブロックの写しをもつ処理装置群に，そのブロックの内容が書き変わったことを連絡する．この連絡の要否の検査は，共有キャッシュのデレクトリ検索と並行して行われる．この検査の結果，バッファ合せが必要な処理装置群には，その操作に必要な情報を送られるが，このなかには書き変えられたブロックのアドレス情報が必ず含まれねばならない．この場合，共有キャッシュと固有キャッシュの参照アドレスの種類が同じなら，共有キャッシュを書き変えたアドレスを，そのままバッファ合せの必要な処理装置群に送ればよい．また論理・実アドレス方式2では両キャッシュの参照アドレスの種類は異なるが，スイッチ装置には変換前の論理アドレスがあるので，それをそのまま送ればよい．一方，論理・実アドレス方式1の場合，スイッチ装置には実アドレスしかないので，バッファ合せを行う処理装置側で，実アドレスから論理アドレスに変換しなければならない．しかし通常のアドレス変換機構は，実アドレスから論理アドレスへの逆変換が便利なようには設計されていないため，バッファ合せ機能を実現する場合，論理・実アドレス方式1は大きな欠点をもつことになる．

3.5 アドレス変換機構

3.5.1 処理装置内でアドレス変換する場合

実アドレス方式と論理・実アドレス方式1は，1台の処理装置でシステムを構成することを基本として設計されたアドレス変換手法を，そのまま密結合マルチプロセッサシステムに適用しようとするものである．このため密結合マルチプロセッサシステムの場合，以下に述べる不具合が顕在化する．これらの不具合は，接続する処理装置の数が少ないうちは性能に大きな影

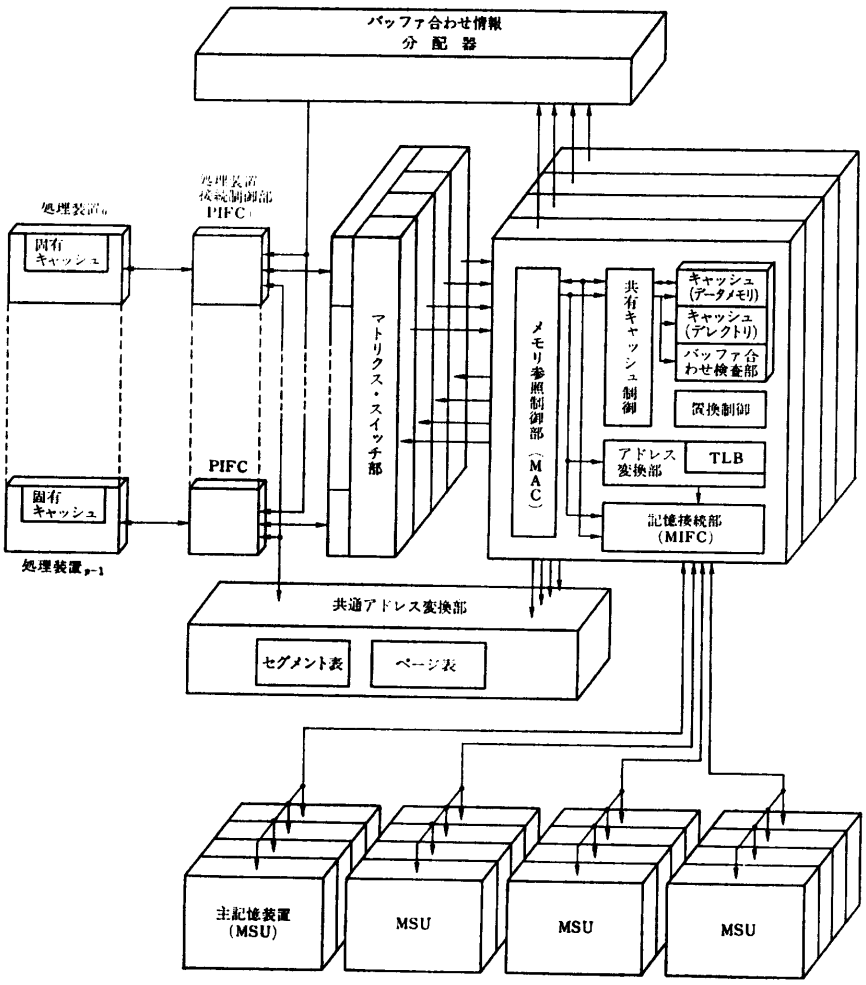


図 4 スイッチ装置にキャッシュとアドレス変換機能を実装した例
Fig. 4 A system in which cache and address translation mechanisms are implemented.

響を与えなくとも，処理装置の数が 20 とか 30 に増大すると，性能に及ぼす影響を無視できなくなる．

（1） アドレス変換表を仮想空間にもつこと．

（a） セグメント表やページ表のため仮想空間と実記憶が消費され，利用者の利用可能領域が減少する．またセグメント表やページ表の一部がキャッシュに収容されるため，キャッシュの利用可能領域も減少する．

（b） TLB によるアドレス変換に失敗すると，セグメント表，ページ表によるアドレス変換を行うため，メモリ参照要求の競合がその分増加する．

（2） 実ページの割当て変更時，他の処理装置の TLB も更新が必要となり，性能低下の要因となる．

（3） 入出力動作時，各入出力チャネルはアドレス変換機構を備えねばならない．

3.5.2 論理・実アドレス方式 2，論理アドレス方式

これらの方式では，スイッチ装置内でアドレス変換操作が行われるが，以下ではスイッチ装置にアドレス変換機構を設ける方式の実現可能性を示すため，筆者が開発中の図 4 に示す密結合マルチプロセッサシステムの例を説明する．

（1） p 台の処理装置は処理装置インタフェース（P接続路）によりスイッチ装置に接続される．スイッチ装置内の各部とは処理装置接続制御部 (PIFC) を介して接続される．

（2） それぞれインタリーブ可能な m 群の記憶装置は，主記憶インタフェース（M接続路）によりスイッチ装置に接続される．スイッチ装置内の各部とは，記憶装置接続制御部 (MIFC) を介して接続される．

（3） 処理装置群と記憶装置群とは，$p \times m$ のマト

リクス・スイッチにより接続される．このスイッチを通して，メモリ参照に関連した情報は伝送される．

（4） スイッチ装置の内部には，各M接続路ごとにキャッシュやセットアソシアテブ方式の高速アドレス変換バッファ (TLB)，MIFC 部などが設けられ，これらをメモリ参照制御部 (MAC) が制御する．

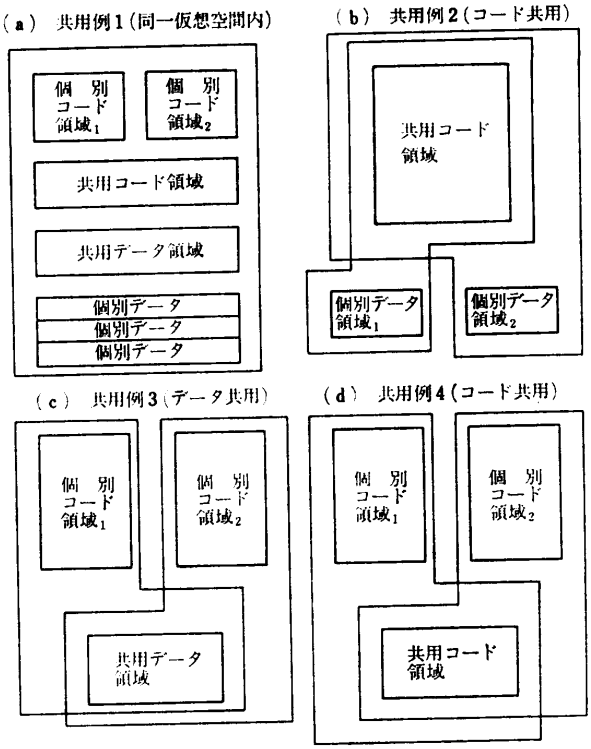
（5） m 群の MAC から共用される形で，1組の共通アドレス変換部 (CATU) があり，ここにセグメント表やページ表を収容するための専用のメモリ（アドレス変換表メモリと呼ぶ）が設けられる．TLB によるアドレス変換に失敗した場合，それぞれの MAC は，共通アドレス変換部にアドレス変換を依頼する．通常，TLB によるアドレス変換の成功率が高いため，m の値が4とか8のシステムでは，m 群のM接続路に対し，1組の共通アドレス変換部で十分と考える．

（6） セグメント表やページ表の作成を，任意の処理装置から行えるように，各 PIFC 部は CATU 部と単一のバスで接続されている．このバスを通して，セグメント表やページ表の作成や削除の指示がなされる．なお，これらの表の保存や復元のため，各処理装置からアドレス変換表メモリを読書きすることが可能となっている．

（7） セグメントやページの新規登録や削除の指示を受けると，CATU はセグメント表やページ表の各エントリの追加，あるいは削除を行うとともに，MAC を介して，m 組のアドレス変換部に TLB の更新を指示する．アドレス変換部はこれを受けると，処理中のメモリ参照動作の終了を待って，TLB の更新を行う．この方式では，アドレス変換表メモリは，論理アドレス空間からも，実アドレス空間からも独立しているため，3.5.1 項(1)に述べた不具合は解消されている．また，TLB が各処理装置に存在する場合，3.5.1 項(2)に述べたように，実ページの割当て変更に伴う TLB の更新の指示を，各処理装置ごとに行う必要があった．しかし，この方式ではこの操作はスイッチ装置内で完結するので，この操作に伴う性能の低下の点でも，システムの制御の複雑さの点でも改善されている．

3.6 プログラムの共用

命令コードやデータの共用は単一処理装置のシステムでも行われるが，密結合マルチプロセッサシステムでは，共用されるのが常態といってよい．**図5**に共用の例を示すが，(a)は一つの仮想記憶空間のなかでの共用の例で，複数のデータに対して命令コードが共用



細線で囲まれた部分はそれぞれ別々の仮想記憶領域を示す．

図5 プログラム領域を共用する例

Fig. 5 Various examples of common usage of the program area.

される．(b)～(d)は別々に仮想記憶にローデングされるプログラム間で共用情報がある場合で，(b)は命令コード，(c)はデータ，(d)は命令コード領域の一部を共用する場合である．

（1） (a)の場合のように一つのローデング単位の中での命令コードの共用は，命令コードを再入可能な形式に設計し，個別データとの対応に間違いがなければ，共有命令コードを異なる処理装置で実行しても何ら問題は生じない．

（2） 一方(b)～(d)のように，異なる仮想空間をもつプログラム間で領域を共用する場合，キャッシュを論理アドレスで参照するシステムでは，その方法を検討する必要がある．

従来，プログラム領域を共用する場合には，**図6**に示すように，仮想記憶空間には共用を意識せずにローデングしておき，仮想記憶空間から実記憶のアドレスに変換する過程で，実記憶の同一領域を参照するように，アドレス変換表の内容を設定していた．

一方，論理アドレスでキャッシュが参照される場合，命令コードやデータなどのプログラム領域を共用

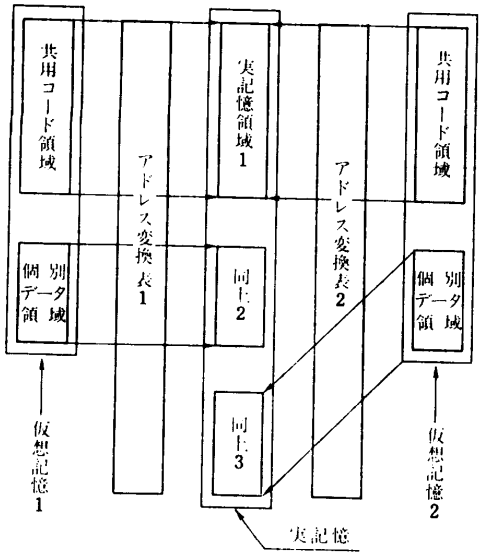


図 6 プログラム領域の共用（従来方式）
Fig. 6 An example of common usage of the program area by address translation (current system).

するには，仮想空間アドレスが一致していなければならない．すなわち，図5で(b)の場合は命令コード領域が，(c)の場合はデータ領域が，(d)の場合は共用コード領域が，それぞれ仮想記憶空間にローデングされる時点で，同一領域を参照するようにその値を設定されねばならない．なお(b)の場合，演算数の参照時，命令コードは同一でありながら，それぞれ別領域の演算数を参照することのできる機構（たとえばベースレジスタ）も必要となる．

3.7 評価（まとめ）

マトリクス・スイッチ装置にもキャッシュをもつシステムでは，アドレス変換機構をどの装置に実装し，それぞれのキャッシュは，論理アドレスと実アドレスのいずれで参照されるのが望ましいかという問題について，これまでの検討結果を総合し評価する．

（1） **表1**にこれまでの検討結果をまとめた．この表から明らかなように，両キャッシュとも実アドレスで参照する方式はいずれの面でも劣っており，接続処理装置数が多くなるとともに，それらの影響が大きく現れるため，接続処理装置数の小さなシステムを除き，採用できない．

（2） 両方のキャッシュとも論理アドレスで参照される方式は，すべての面で優れており，この形式の密結合マルチプロセッサシステムに最も適した方式といえる．ただしこの方式を採用する場合，別々の仮想記憶空間に読み込まれるプログラム間で，記憶領域を共用するには，共用される領域の論理アドレスを一致させる必要がある．

（3） アドレス変換機構をスイッチ装置内にもち，処理装置内のキャッシュは論理アドレスで，スイッチ装置内のキャッシュは実アドレスで参照する方式は，論理アドレス方式に比べれば劣っているものの，多くの優れた特性をもつので，次善の方式として考えられる．この場合，性能の低下を避けるため，スイッチ装置でのアドレス変換動作を高速で行う必要がある．

表 1 各キャッシュの参照アドレスの違いによる性能への影響（まとめ）
Table 1 Evaluation of performance difference due to reference address type of two hierarchically positioned caches.

方式 / 評価項目	実アドレス方式	論理・実アドレス方式1	論理・実アドレス方式2	論理アドレス方式
固有キャッシュ	実アドレス	論理アドレス	論理アドレス	論理アドレス
共有キャッシュ	実アドレス	実アドレス	実アドレス	論理アドレス
アドレス変換機能実装装置	処理装置	処理装置	スイッチ装置	スイッチ装置
メモリ参照速度	普通	速い	やや遅い	最も速い
キャッシュ領域の有効利用	劣る	良い	良い	最も良い
主記憶への書き戻し	固有の問題無	固有の問題無	固有の問題無	その都度アドレス変換が必要
バッファ合せの容易性	固有の問題無	アドレスの逆変換が必要	固有の問題無	固有の問題無
アドレス変換の容易性	マルチプロセッサでは問題有	同左	優れている	優れている
領域の共用の容易性	実アドレスで共用	同左	同左	論理アドレスで共用

4. むすび

処理装置群と主記憶装置群とを接続するスイッチ装置のなかにキャッシュを設けたマルチプロセッサシステムで，処理装置とスイッチ装置内のそれぞれのキャッシュは，論理アドレスと実アドレスのいずれで参照されるべきか，また，論理アドレスから実アドレスに変換する機構を，どこに設けるべきかについて，本論文では種々の観点から評価した．その結果，メモリ参照速度，キャッシュの記憶領域の有効利用，アドレス変換機構の性能，バッファ合せ制御などのいずれの観点からも，両方のキャッシュを論理アドレスで参照する方式が優れていることを示した．この場合，アドレス変換機構はスイッチ装置に設けられることになるが，本論文では，これまで単一処理装置のシステムで用いられてきた，セグメント表やページ表などのアドレス変換表と，高速アドレス変換バッファを用いたアドレス変換機構を，スイッチ装置に無理なく導入する方式を示した．

参考文献

1) Enslow, P. H., Jr. (村岡訳)：マルチプロセッサと並列処理，p. 354，近代科学社，東京 (1976).
2) Quatember, B.: Modular Crossbar Switch for Large-scale Multiprocessor Systems—Structure and Implementation, Proc. of the AFIPS, Vol. 50, pp. 125-135 (1981).
3) Wulf, W. A. and Bell, C. G.: C. mmp—A Multi-Mini-Processor, Proc. of the AFIPS, Vol. 41, Part 2, pp. 765-777 (1972).
4) 杉本正勝：C. mmp の評価，情報処理，Vol. 20, No. 4, pp. 301-306 (1979).
5) Mudge, T. N. and Makrucki, B. A.: Probabilistic Analysis of a Crossbar Switch, The 9 th Annual Symposium on Computer Architecture, pp. 311-319 (1982).
6) 山本 登：マルチマイクロプロセッサシステム・MμPS-1 のバッファメモリ方式とその効果，情学全大講演論文集，pp. 113-114 (1981).

(昭和59年6月1日受付)
(昭和59年10月18日採録)