U.D.C. 681.32.066:681.327.2-181.2.067.2

OS VOS3/ASの高性能・大容量化方式

Development of Performance Improvement Using Enlarged Virtual Storage for Operating System VOS3/AS

吉澤康文*　*Yasufumi Yoshizawa*
新井利明**　*Toshiaki Arai*
原田　晃***　*Akira Harada*
旭　寛治***　*Hiroharu Asahi*

大形計算機は，企業の情報システムの中で中枢機能として組み込まれ利用されている。そこで，1990年代に課せられた大形計算機の高性能化と大容量化について述べる。

高性能化を達成するために三つのアプローチがとられている。第一は従来磁気ディスクに格納されていたプログラムやデータをメモリ上に格納し，「入出力を大幅に削減」することである。このために，データ格納領域を「G(ギガ)からT(テラ)へ」拡大する大容量化が必須(す)である。データ空間，スーパー空間の開発目標はここにある。第二は，拡張された記憶領域を利用する機能の開発であり，データアクセスを「m(ミリ)からμ(マイクロ)へ」とする機能である。データ イン バーチャル，VSAM/HAF(Virtual Storage Access Method/High Performance Access Facility)，PREST(Parallel Reference and Synchronous Transfer Facility)，XPL(Extended Program Loading)，ES(Extended Storage)ファイル，ESページングなどによって実現される。第三は，プロセッサパワーの強化を図る高多重プロセッサの効率よいサポートである。

また，大形計算機はますますデータの集中と共用が図られる。このため，データのセキュリティが重要課題である。さらに，プログラムとデータのアクセス権の確立というケーパビリティを達成し，計算機システムの信頼性を高める必要がある。データ空間は上記の高性能化以外にもこのような目的をもって開発された。

1　緒　　言

1980年代の後半では第三次オンライン バンキング システムに代表されるOLTP(On Line Transaction Processing)の開発が盛んとなり，大形計算機が大量に適用されてきた。これらのシステムには，汎(はん)用のOS，汎用のDBMS(データベース マネジメント システム)，そして汎用のオンライン コントロール プログラムが利用されるようになった。また，アプリケーションプログラムの汎用化が進み，ソフトウェアの流通を可能とする開発がなされてきた。このような一連の汎用化はソフトウェアの生産性を高める効果ばかりでなく，今後のOLTPの拡張を効率的なものにすると期待される。

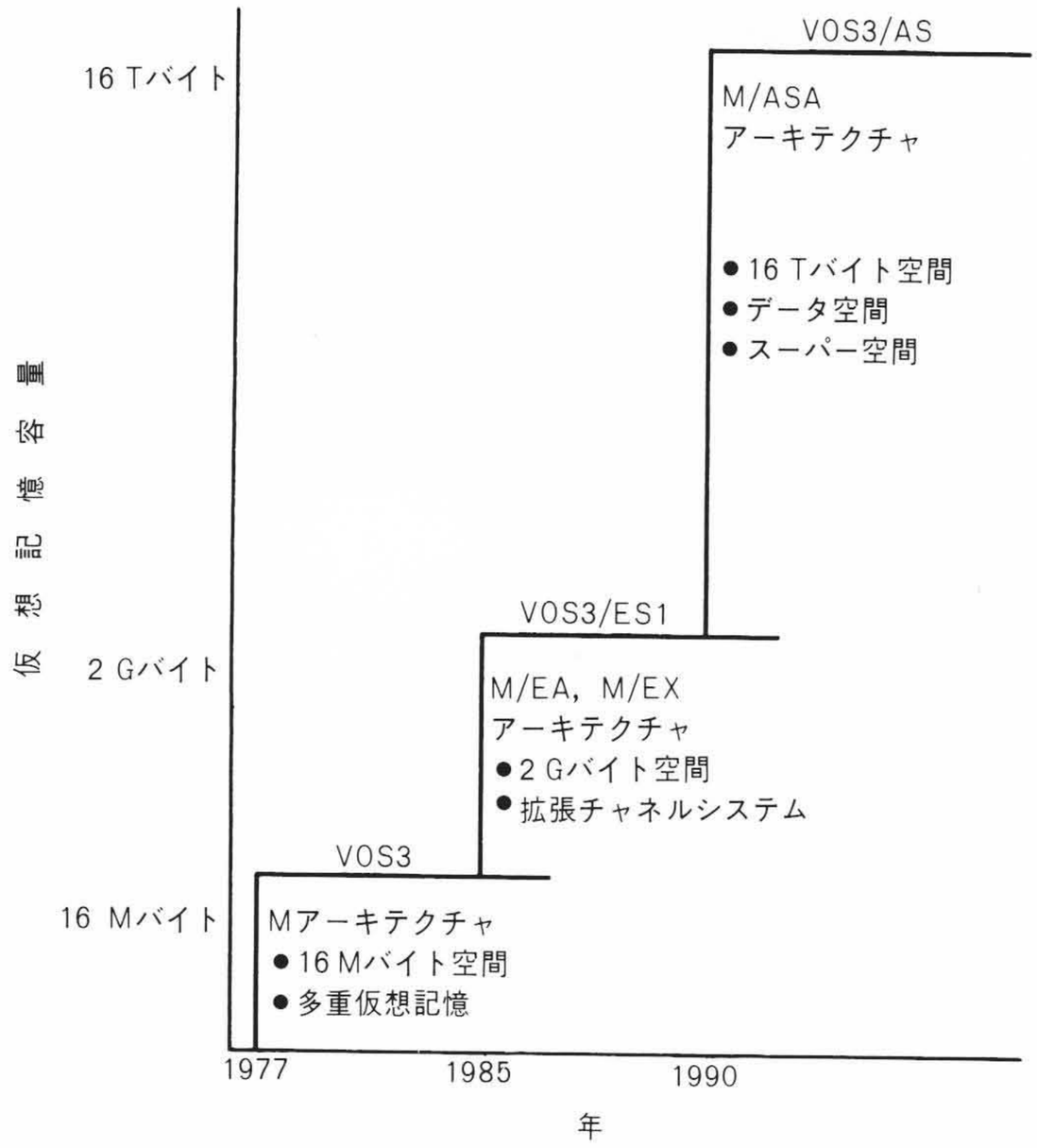
現在，大形計算機システムの役割は，OLTPと大規模計算のサーバであることが明確となってきた。上記の汎用化は数々の利点を持つが，計算機システムの性能という面ではある程度の犠牲を伴う。そこで，1990年代の大形計算機システムに強く求められる高性能化と大容量化について，以下，新しく開発されたVOS3/AS(Virtual Storage Operating System3/Advanced System Product)の機能について述べる。

2　領域の拡張と階層化

2.1　アドレス領域を拡大するアーキテクチャ

アドレス領域の拡大は，計算機アーキテクチャの一つの進歩である。Mシリーズも，初期の24ビットによるアドレス表示を31ビットへ拡大してきた。この拡張によって，ユーザーの使用できるVS(Virtual Storage：仮想記憶)領域は16 Mバイト(Mバイト：10^6)から2 Gバイト(Gバイト：10^9)へと飛躍的に拡大した。VSの拡張について**図1**に示す。

* 日立製作所 システム開発研究所 工学博士　** 日立製作所 システム開発研究所　*** 日立製作所 ソフトウェア開発本部



注：略語説明 VOS3/AS (Virtual Storage Operating System3/Advanced System Product)
M/ASA (M-series/Advanced System Architecture)
VOS3/ES1 (VOS3/Extended System Product 1)
M/EA(M-series/Extended Addressing)
M/EX (M-series/Extended)

図1 HITAC Mシリーズでの仮想記憶の拡大 プログラムデータの大容量化に伴い飛躍的な仮想記憶の拡大を図り，大量データ処理を可能にしている。

大形計算機の性能は，4年で約2倍のスピードで向上しているが，使用可能なMS(Main Storage：主記憶装置)の容量は年率30%以上となっている。このため，プロセッサの性能以上にMSの大容量化が進んでいる。このような半導体メモリ技術の進歩を利用し，計算機システムの性能を向上させるためには，大容量記憶を用いてプロセッサパワーを節約する機構が有用になってくる[1]。このための一つの有力な方式は，DASD(Direct Access Storage Device：磁気ディスクのような直接アクセス記憶装置)上のファイルやデータベースをVS上に配置し，命令で直接レコードにアクセスすることである。

このような方式のねらいは，入出力操作が省略できること，OSのオーバヘッドを削減するところにある。さらに，レコードに対するアクセス時間がミリ秒のオーダからマイクロ秒のオーダとなり，対話形利用の応答性の向上に寄与することになる。

プロセッサの性能向上とメモリ拡大のギャップを埋め，計算機システムとしてのトータルな性能向上を図るためには，ファイルやデータベースをVSに配置する方式が有効である。つまり，31ビット化は1980年代後半のOLTPや大規模計算のニーズにこたえてきたが，1990年代に要求される大形システムの性能を満たすためには，31ビットによるアドレッシングでは不十分である。そこで，より飛躍的に大きなVSを実現する必要がある。M/ASAでは，16 Tバイト(Tバイト：10^{12})のVS領域を利用可能とした。この領域はデータを専用に格納することができ，データ空間と呼んでいる。

上記から，大形システムでは記憶容量を「ギガからテラへ」，そしてデータベースへのアクセスは「ミリからマイクロへ」という進歩を成し遂げようとしているのである。

2.2 仮想記憶を支える記憶階層

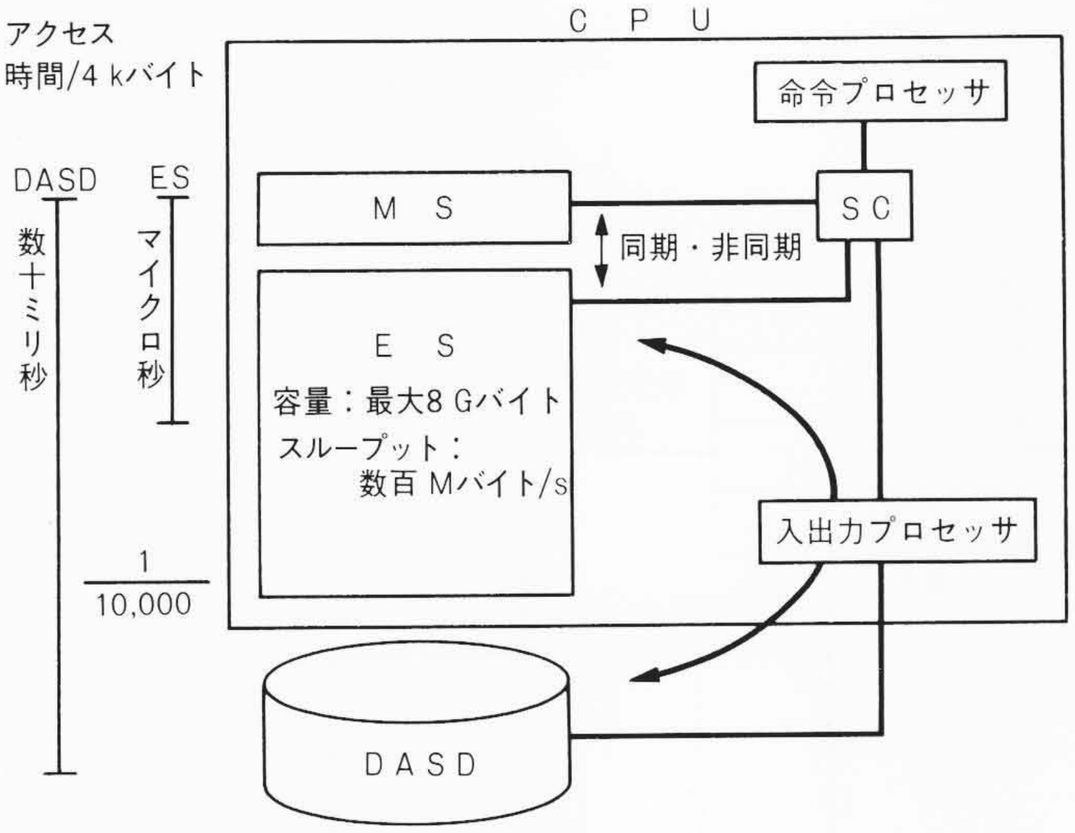
同時に大量のトランザクションを効率よく処理するためには，大きな記憶領域が必要となる。トランザクション管理情報，トランザクション処理に必要な基本プログラムや処理プログラム，またデータベースの一部またはそのすべてなどを格納する記憶領域である。また，大規模計算でもデータ領域の拡大に対する要求は強い。つまり，計算機システムのサービス機能を拡充し，より高度な利用を図るに従ってますます記憶領域の増大化という傾向となる。これらの情報は拡張アーキテクチャのVSに配置されることになる。

このような背景から，大容量のVSを支えるにはMSの拡大だけではコストパフォーマンスの面から限界がある。そこで，VSを支える記憶階層に新たなES(Extended Storage：拡張記憶装置)を開発することが望まれ，VSの性能を低下させない機能の開発を行う必要があった。

従来，VSを支える記憶階層はMSとDASDの2階層であり，DASDのアクセスは数十ミリ秒(10^{-3})を要し，性能劣化要因であった。また，OSの処理に多くのオーバヘッドを必要としていた。新たに開発されたESはMSとDASDとの間に存在し，命令プロセッサによってMSとES間の転送を数マイクロ秒(10^{-6})で実行することができる(**図2**)。また，MSとES間の大量のデータ転送やESとDASDへの転送は，チャネルを通して命令プロセッサとは非同期に実行させる機能も備えている。

2.3 データ空間機構

データ空間機構はジョブが使用できるVS容量を飛躍的に拡大し，大容量データの操作性向上とデータ機密保護強化による信頼性向上を図る機能である。従来のVOS3/ES1(VOS3/Extended System Product 1)では，各ジョブが使用できる仮想領域(アドレス空間)は最大2Gバイトであり，その中にプログラム，データ，さらにOSなどの制御テーブルが混在していた。VOS3/ASでは，上記のアドレス空間のほかに，2Gバイトの領域であるデータ空間を最大8,192個生成し，データを格納することを可能とした(**図3**)。データ空間はジョブ間で共用でき，おのおの異なるアクセス権限を設定することも可能である。そのため，OSの制御情報のように，すべてのアドレス空間から共用され，かつ高い機密性が要求される情報を，



注：略語説明 ES (Extended Storage), MS (Main Storage)
DASD (Direct Access Storage Device)
SC (System Controller)

図2 磁気ディスクとESのアクセス性能比較　　磁気ディスクへのアクセスは数十ミリ秒を要していたが，ESでは数マイクロ秒のオーダとなり大幅な時間短縮が図られ，対話処理の性能向上が期待できる。

データ空間に配置することでアクセスの容易性と気密保護強化を両立できる。

データ空間のアクセス方法を**図4**に示す。データ空間のアクセスにはアクセスリストおよびAR(アクセスレジスタ)を使用する。アクセスリストの各エントリにはアクセス可能な空間識別子を登録する。ARはアクセス リスト エントリへのインデックス番号である。従来，命令のベースレジスタ指定部はアドレス計算用のベースレジスタ番号を示していたが，M/ASA(M-series/Advanced System Architecture)ではARも合わせて指定する。すなわち，ARはベースレジスタの拡張部となっている。この機構により，命令で指定したARが示すアクセス リスト エントリを経由して示される空間がアクセスされる。

次にアクセス権限機能について説明する。アクセス リスト エントリには，その空間とプログラム間のアクセス可能性を示す権限指標を設定することができる。この機構では，共用可能なデータ空間であっても，権限指標の設定によってはアクセスを防ぐことができる。つまり，アクセスリストに登録されたデータ空間だけが参照可能であるため，データ空間上のデータは機密性が守られる。

データ空間は従来のアドレス空間とまったく同様のVSとしての扱いをOSから受けるため，ESを用いたESページング(後述)が利用できる。以上の特徴を利用して，データ空間には以下の情報を格納することが有効である。

(1) 大容量で頻繁に参照されるデータ

(2) 各ジョブから参照され一般のユーザープログラムからの機密保護を強化したい制御情報

(3) 大容量でかつ複数のジョブ間で共用されるデータ

2.4 スーパー空間

ESを備えた計算機システムでは，従来のアドレス空間(2Gバイト)よりも大きなデータ領域を高速にアクセス可能となっ

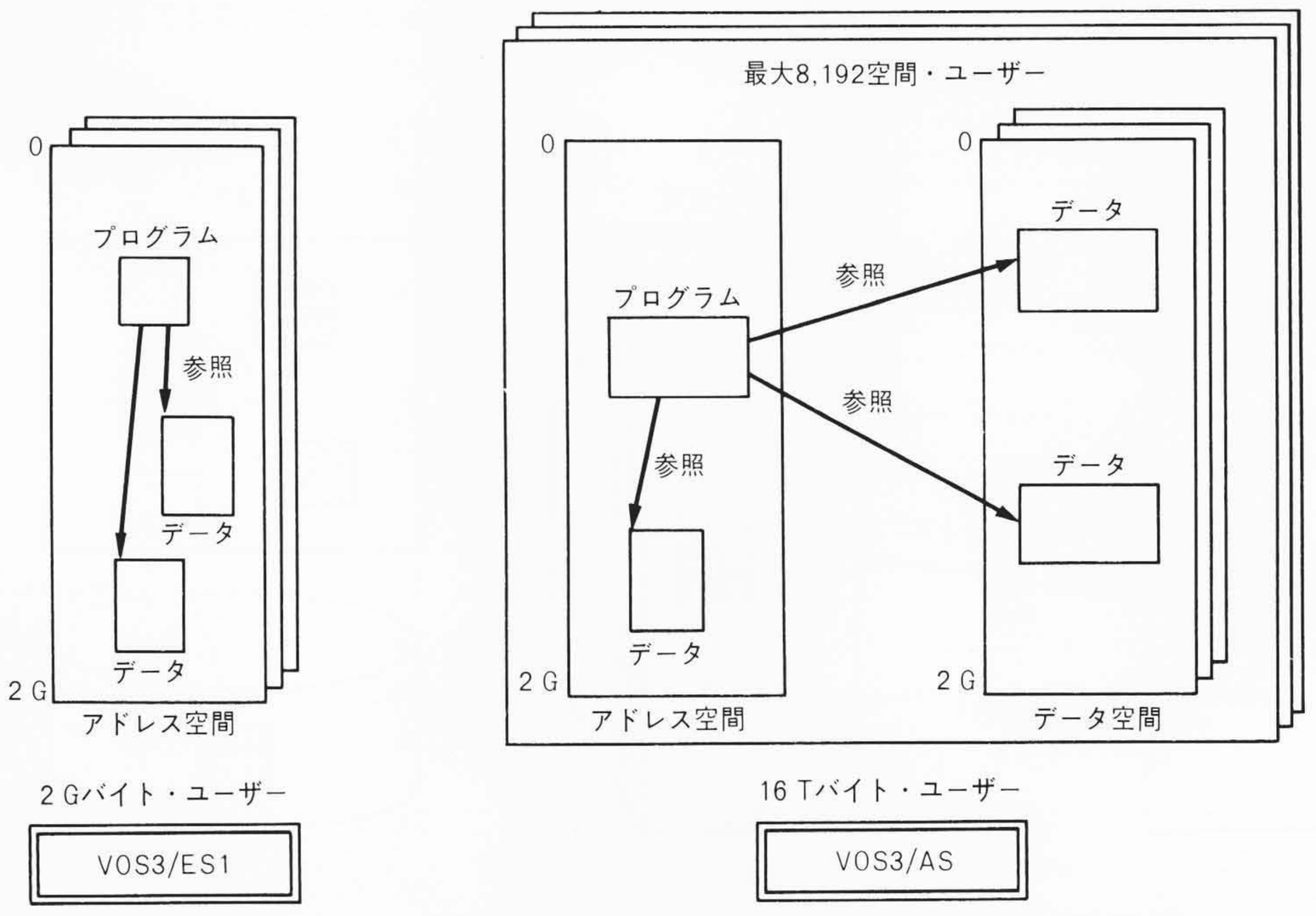


図3 仮想領域量の比較　　新アーキテクチャではアドレス空間の8,192倍のデータ領域が利用可能となり，データベースの仮想記憶への常駐による高性能化が期待できる。

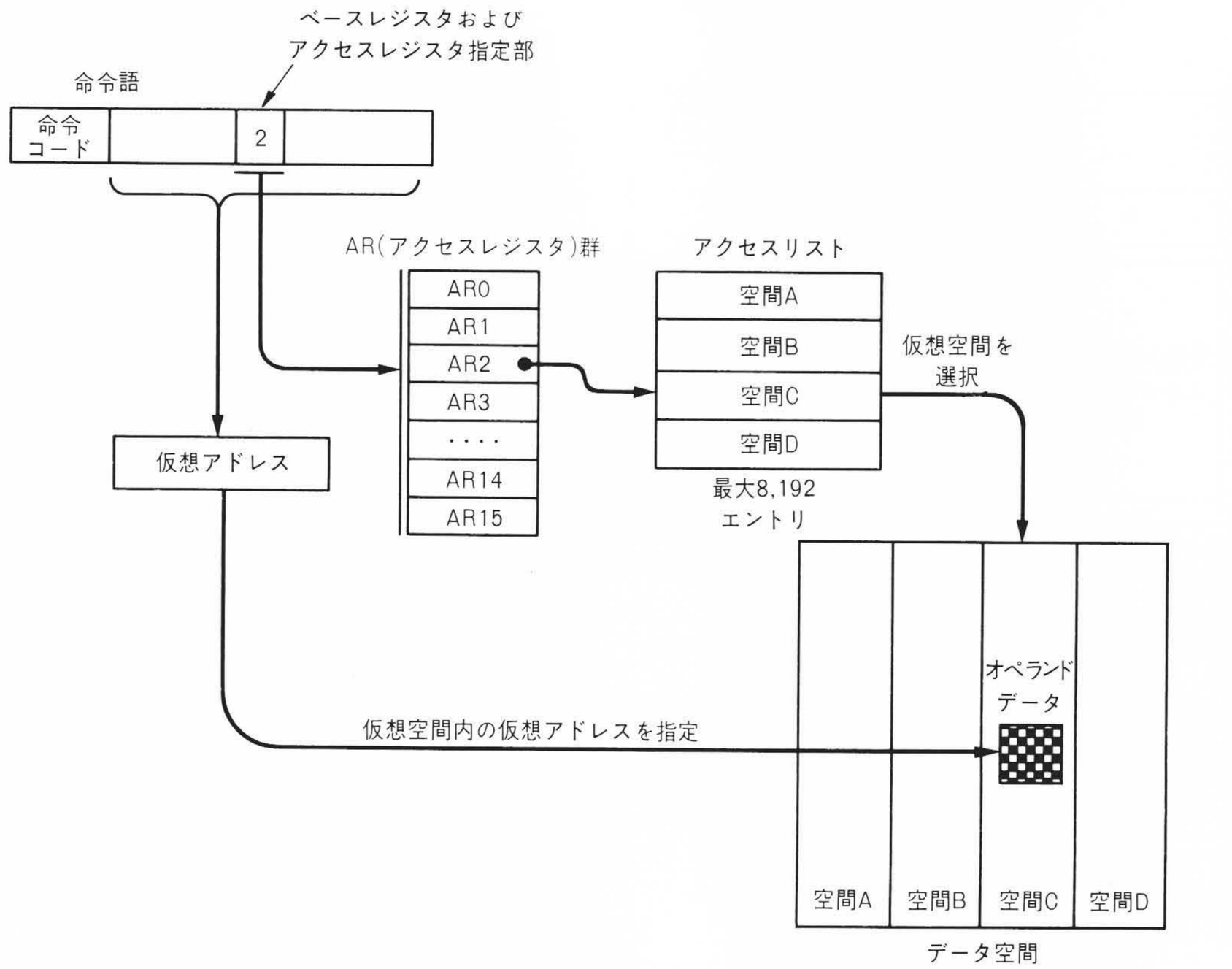


図4　データ空間アクセス方法　　ベースレジスタとAR(アクセスレジスタ)がペアとなり，ARの内容がアクセスリストのインデックスを表し，データ空間へのアクセスとなる。

た。データ空間はそのようなニーズを満たすために開発されたVS領域であるが，新たなARの操作を必要とする。つまり，まったく新しいプログラミング技術を必要とすることになる。そこでデータ空間と同様に，16 Tバイトまでの大容量記憶を従来のプログラミング方法でも利用でき，また，高性能化のためにESを積極的に活用する機能が望まれる。スーパー空間はこのような背景のもとに開発された新しい機能である。

スーパー空間の構成を**図5**に示す。スーパー空間の実体は4 kバイトブロック単位に分割されたデータであり，ES中に格納されている。データ空間はアクセスレジスタを操作して直

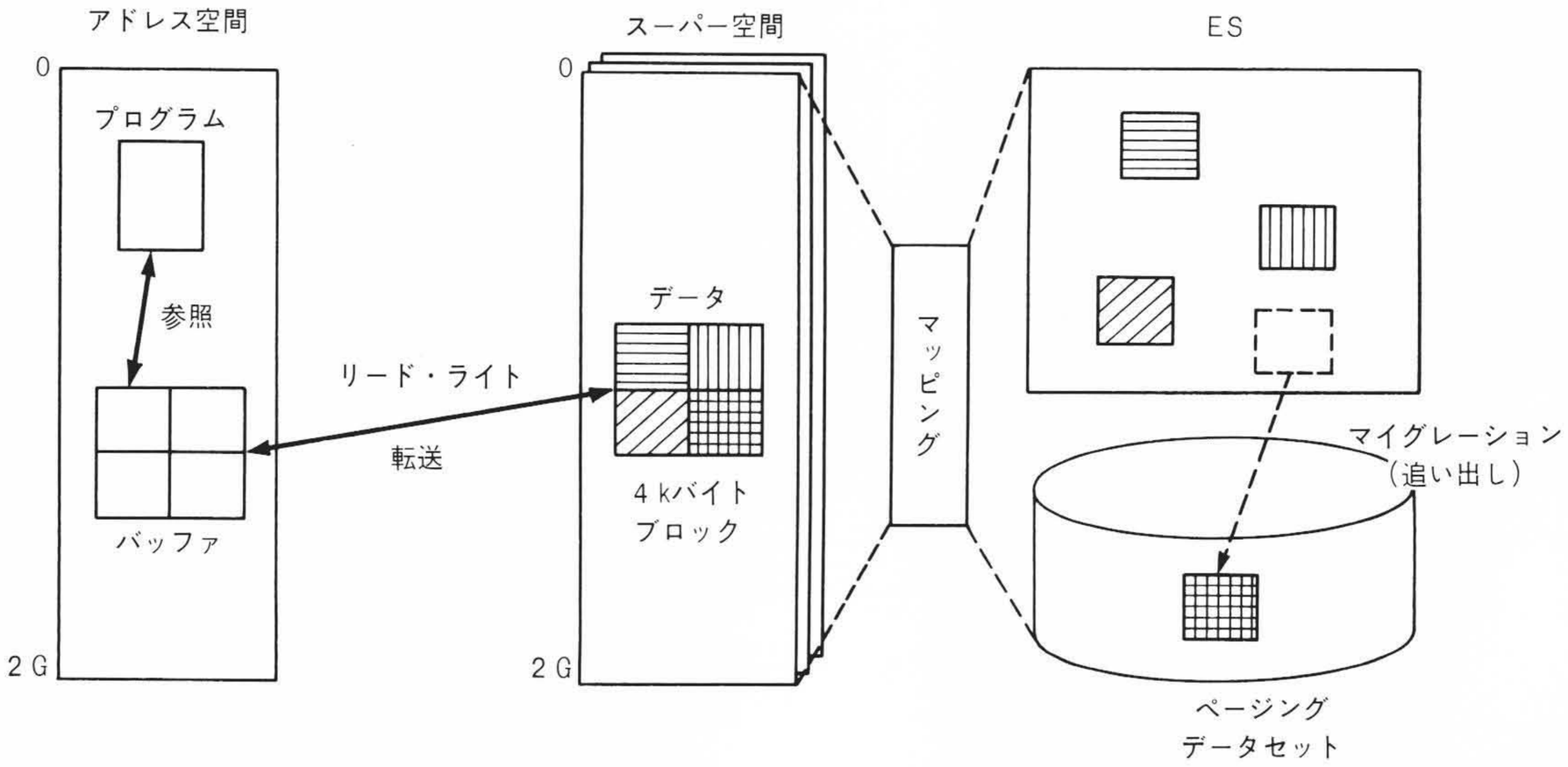


図5　スーパー空間の構成　　スーパー空間の実体は，拡張記憶あるいは磁気ディスク上に格納されるが，プログラムからはアドレス空間上のバッファを通してアクセスされる。

接データへアクセスするが，スーパー空間では，アドレス空間またはデータ空間にバッファ領域を確保し，バッファ領域を介してアクセスする。操作終了後は，バッファ領域のデータをスーパー空間に書き出す。

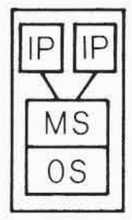
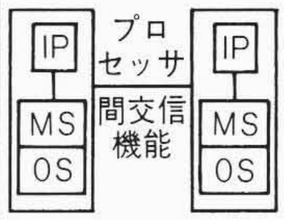
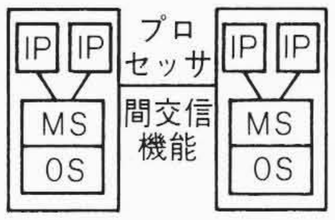
スーパー空間には二つのタイプがある。一つのタイプはESが不足した場合にあふれた部分を自動的にDASDのページングデータセットに書き出される。もう一つは，スーパー空間をES上に固定的にしたもので，DASDへのマイグレーションを行わないタイプである。

上記のように，スーパー空間はMSへの負荷が比較的小さく，かつ高速のアクセスが可能である。このことから，以下の性格を持つデータの格納に有効である。

(1) ブロック単位に集中的にアクセスされる大容量データ

(2) 従来DASDに格納していたが高速アクセスを必要とするデータ

上記の例として，ソートやRDB(Relational Data Base)などの作業領域，科学技術計算の巨大アレーなどの使用が期待できる。

多重プロセッサ構成方式	密結合方式	疎結合方式	密結合・疎結合混在方式
構成			
処理能力	△	○	◎
信頼性	△	○	○
価格比	○	△	△
操作性	○	△	△

注：略語説明 IP(命令プロセッサ)，MS(主記憶装置)

図6 密結合方式と疎結合方式の比較 密結合方式，疎結合方式，両者の混在方式をVOS3/ASでは実現し，かつAOMPLUS(Auto Operation Monitor PLUS)では運用の容易性を図り，それぞれの特徴を生かしたシステム構成が可能である。

3 多重プロセッシングの進展

3.1 密結合多重プロセッシング

複数の命令プロセッサがMSを共用し，一つのOSの下で同時に複数のプロセスが実行される処理形態が密結合多重プロセッシングである。多くのサービス機能を実現するためにソフトウェアや管理情報が密に実装され，また，データベースを共用するようなトランザクション処理の高性能化には有力なシステム構成法である。また，大規模計算では並列計算を可能とすることにより，高速な問題解決が可能となる。このような背景から，近年，大形計算機の分野では複数の計算機を同時に動作させて，高性能化を達成する傾向にある。

密結合多重プロセッサ方式を，疎結合多重プロセッサ方式と比較した結果を**図6**に示す。同図から，処理能力および信頼性の項目で密結合方式はやや劣るが，性能価格比，操作性の点で有利である。VOS3/ASでは，密結合多重プロセッサ方式の処理能力を高めるために幾つかの改良を図り，所期の目的を達成した(**4.4**で述べることとする)。

3.2 疎結合多重プロセッシング

大容量のデータベースを複数の計算機システムで共用したり，あるいは計算機能力が密結合多重プロセッシングだけでは不足するようなときは，疎結合多重プロセッシングが有効となる。疎結合多重プロセッシングの特徴は，**図6**に示すとおり処理能力の強化，信頼性の向上に有効である。しかし，密結合多重プロセッサ方式に比べると価格性能比はやや低くなる。また，計算機の運用の面では，複数のOSが動作することからOSオペレーションが増大するため，オペレータ要員の負担が重くなる。このような従来の欠点を補うために，VO3/ASではAOMPLUS(Auto Operation Monitor PLUS)による操作性の向上を図った。

疎結合方式と密結合方式を組み合わせた混合方式は，大規模なデータ処理に有効な方式である。このような大規模ローカル複合システムでは，従来，オペレーションなどの運用性に問題があった。そこで，VOS3/ASでは運用性を向上するためにAOMPLUSを新たに開発し，オペレータの負荷の低減を可能としている。AOMPLUSは，大規模ローカル複合システムに接続されたすべての計算機システムの統合運転を行い，また，コンソールメッセージに対する自動応答も可能とし，省力化，無人運転などを可能としている[9]。

4 OS機能の拡充

4.1 拡張記憶の利用方式

ESは増大するVSを支える新たな記憶階層として有効利用されるだけでなく，高速なアクセスを活用した応用が期待できる。つまり，OSの固有の処理に占有して使用するのではなく，広くユーザーに開放されたものでなければならない。このためには，ESを複数の論理的な分割単位として利用できること，つまりファイルの考え方を取り入れたES管理方式の導入が必要である[4](**図7**)。

(1) 仮想記憶のページング利用

VOS3/ASではESを論理的に分割し，ESファイルとして管理する方式としている。そこでOSがVSのバッキングストレージとして使用するES領域は，システム立上げ時に一つのESファイルとして確保する。この領域は，MSが不足したときのページングやスワッピング用に従来のDASDよりも優先して使用

される。これをESページングと呼んでいる。**図8**にESページングの構成を示す。これにより，従来方式では数十ミリ秒であったページング時間を数マイクロ秒に短縮し，さらにOSオーバヘッドも$\frac{1}{5}$に削減した。バッチ処理で利用される一時的ファイルとしてのVIO(Virtual I/O)もESを利用した制御方式が適用される[3]。

データ空間やスーパー空間の利用などから，VS領域の大幅な拡大が予想される。したがって，ESをVSのバッキングストレージとして積極的に利用する方式を採用し，データ空間やスーパー空間を利用したアプリケーションの性能を保障することとした。

(2) ESファイルの利用

大規模なバッチ処理では複数のSAM(Sequential Access Method：順編成)ファイルをジョブステップ間の引き継ぎ情報として利用することが多い。このような業務処理では，I/O時間が全処理時間に占める割合が高く，従来から入出力ネックとして問題となっていた。

新たに開発されたESファイルを利用すると，SAM系ファイルをES上に作成することが可能となる。PL/I，COBOL，FORTRANなどで開発された定型バッチ処理の入出力ネックの問題を解決すると期待される。

(3) マイグレーション

MSとページング用ESファイルの容量の和が，要求されるVSの全容量よりも大きい場合は，ESページングが行われる。しかし，VSの全要求量が増大してくるとDASD上のページングデータセットを利用することになる。

OSは，計算機システムの性能を安定させ，スループット応答時間を保つために，負荷の急激な増大を常に予測した制御を行う必要がある。VS方式ではMSの要求を予測し，実ページの負荷に応じた制御方式が重要であった[11]。

ESを利用したVS方式では，MSだけの負荷適応制御方式をさらに発展させ，ESを含む最適化でなければならない。そこで，VSの要求量が大きくなりメモリ負荷が増大すると，ESの中から使用頻度の低いフレームをDASDに転送する。この操作をマイグレーションと呼んでいる(**図5**)。この操作により，MSからESへのページをESスロットの空きを待つことなく実行できる。

マイグレーションの目的はESを有効に利用することである。ES上のフレームの中で，近い将来参照されそうもない不活性なフレームを選択し，DASD上に追い出すことで参照頻度の高いフレームをESに配置する。マイグレーションはVSの要求量が高くなるに従って起動されるようになり適応形制御方式となっている。

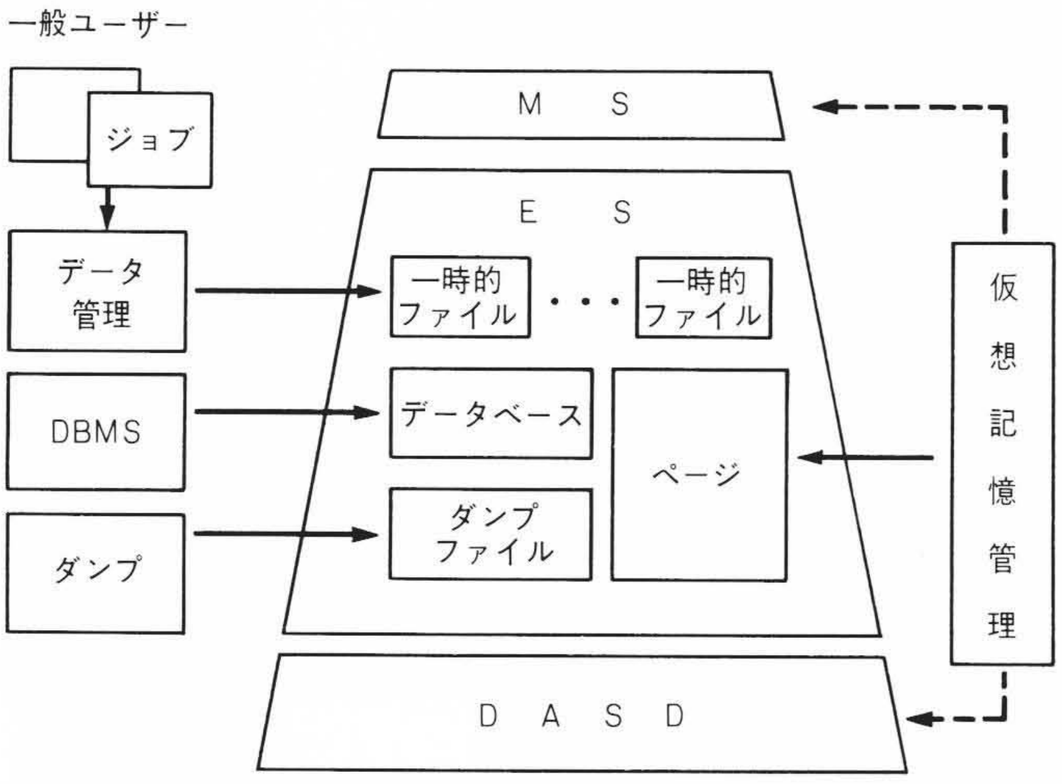


図7 ES(拡張記憶装置)の利用形態　高速アクセスを実現するESを，OSだけでなく広範囲に利用してもらえるようESファイルを実現した。

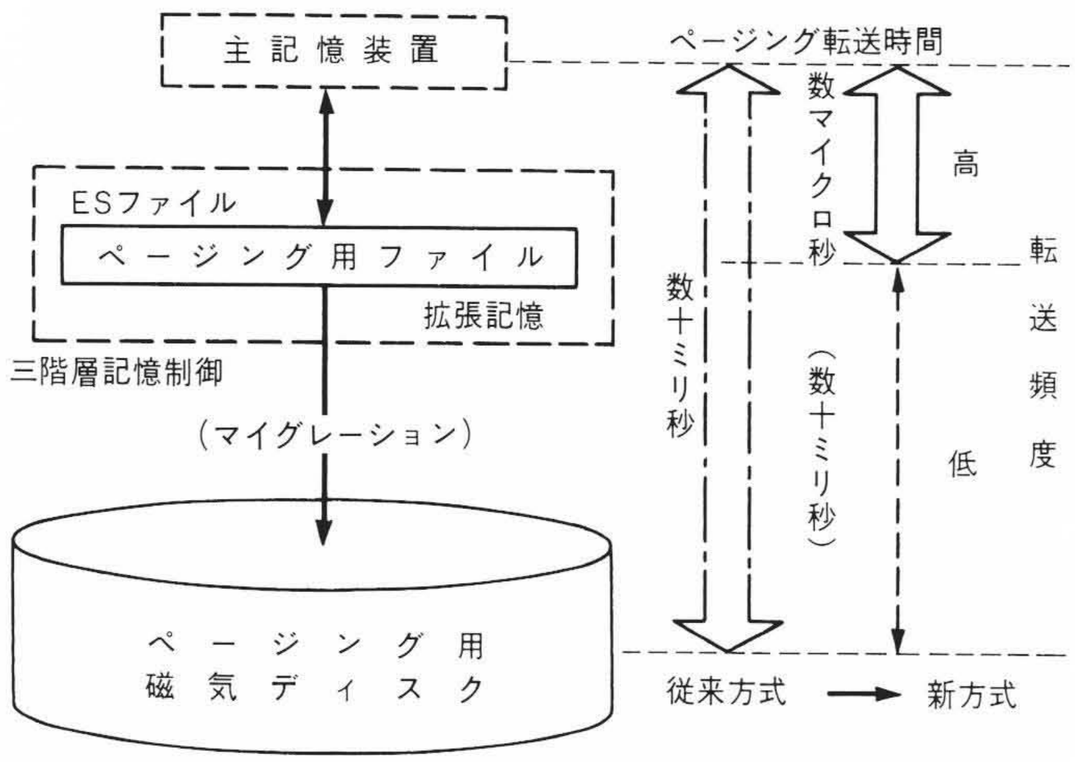


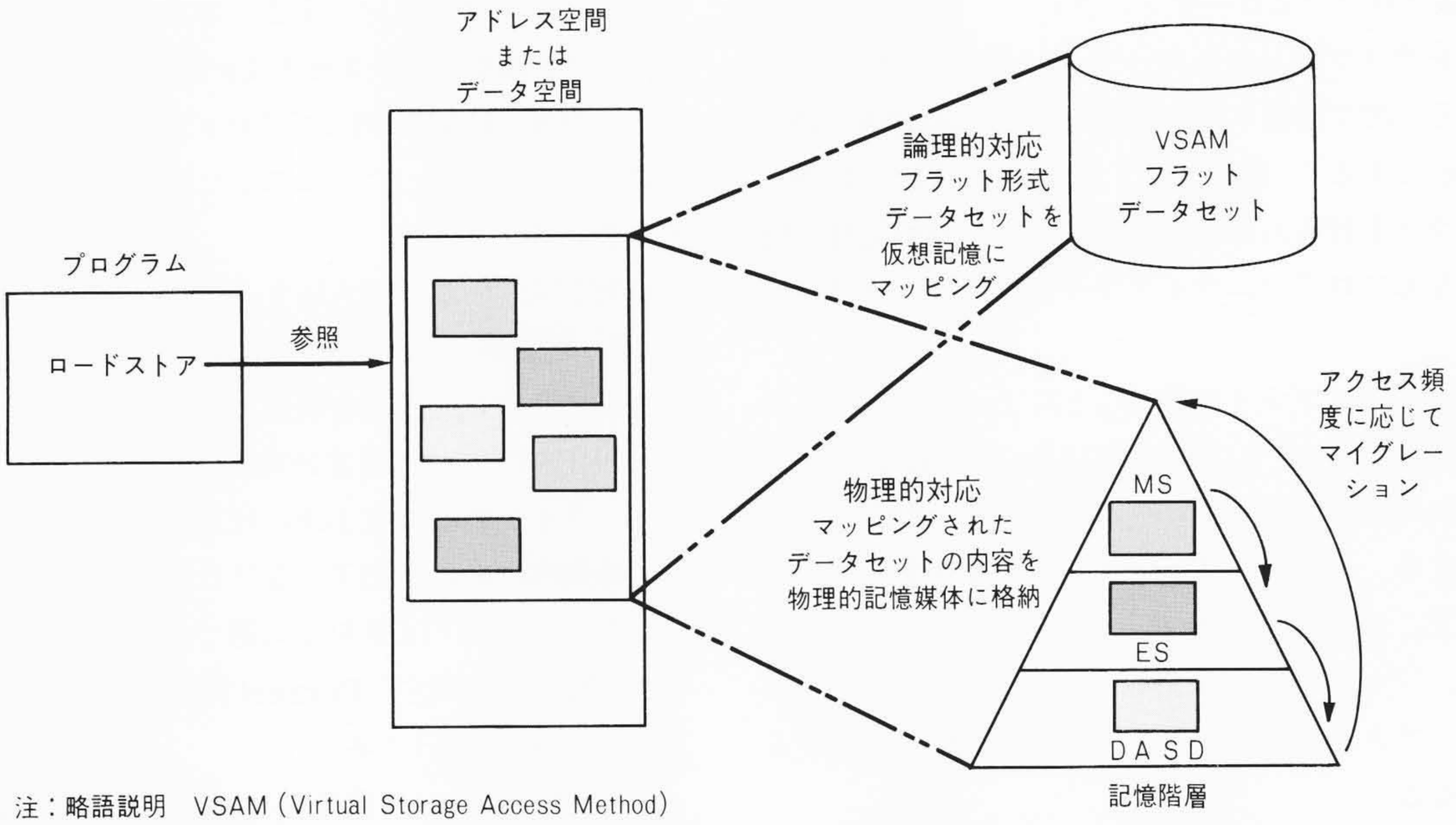
図8 ESをページングに使用した三階層記憶制御　ESページングを使用することにより，ページング時間を数十ミリ秒から数マイクロ秒に短縮できる。

4.2 データ イン バーチャル機構

DIV(Data In Virtual：データ イン バーチャル)機構は入出力に伴うオーバヘッドの解消と，プログラム生産性向上を目的とした機能である。DIVはファイルの内容をVSにマッピングし，ファイルアクセスをVSへのアクセスとする。本機構により，DIVではファイルアクセス速度をmsオーダからμsオーダに短縮し，OSオーバヘッドの削減が可能となる。また，プログラミングの際に入出力の操作手順が不要であり，プログラムの生産性が向上する。

DIVの対象となるデータセットは，VSAM(Virtual-storage Access Method)データセットの特殊な形式であるFDS(Flat Data Set)である。FDSは新しい形式のVSAMデータセットであり，CI(Control Interval)中に制御情報を含まずCI全体にユーザーデータを格納する形式である。

DIVの構成を**図9**に示す。FDSをVSにマッピングすることにより，プログラムから直接領域をアクセスしてFDSの内容



注：略語説明　VSAM (Virtual Storage Access Method)

図9　DIV(データ イン バーチャル機構　　VSAMフラットデータセットを仮想記憶に一次元的に展開し，プログラムから直接アクセス可能としているので，従来の入出力のイメージがなく使い勝手が向上する。

を参照することができる(論理的対応)。マッピングされた領域(ウインドウ領域と呼んでいる。)の実体は，MS，ESあるいはDASD上のページングデータセットに格納されている(物理的対応)。頻度高くウインドウ領域をアクセスすれば，内容はMSに存在し高速なアクセスとなる。

スーパー空間やFDSを高級言語から利用可能とするため，ウインドウサービス機能をあわせて開発した。ウインドウサービス機能は，スーパー空間を連結した最大16 Tバイトの作業領域や，最大4 GバイトのFDSをウインドウを通してアクセスする機能である(**図10**)。配列などの変数領域をウインドウ領域とすることにより，変数を参照するだけでFDSをアクセスしたり，大容量の作業領域(スーパー空間)をアクセスすることが可能となる。

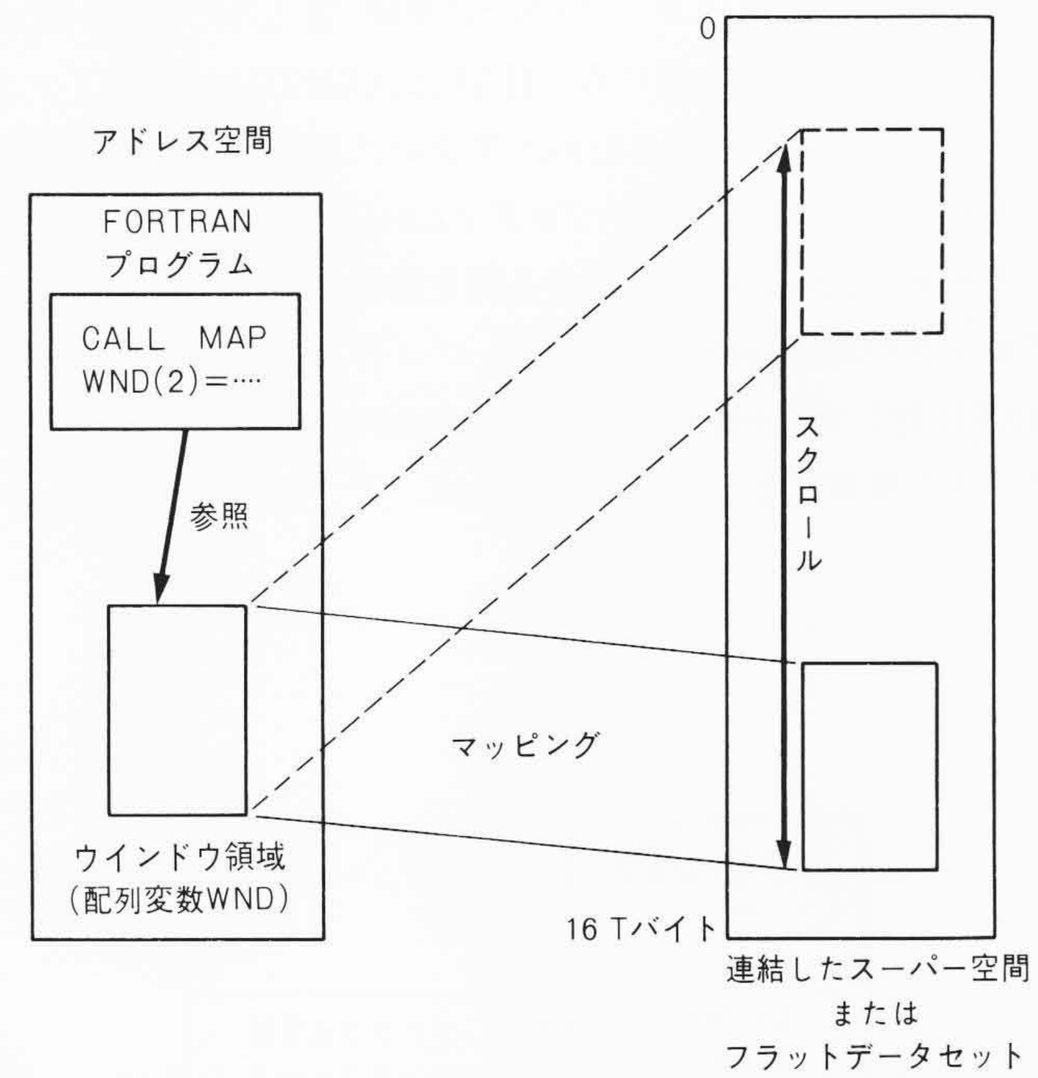


図10　ウインドウサービス機能　　スーパー空間やVSAMフラットデータセットを，アドレス空間上のウインドウを介してアクセスする各種の機能を提供している。

4.3　入出力仮想化技術

(1)　背景と目的

計算機システムは，今後ますます思考過程の道具として利用されるようになっていく。このため，従来はバッチ処理で行っていた問題解決を対話形で行うのが普通になる。つまり，対話処理の応答性能を大きな計算機負荷に対しても保障する必要が生まれている。

応答性能の向上にはプロセッサの能力を向上させること以外に，入出力スピードの向上も大切な要因である。先に述べたように，プロセッサの性能向上に比べて利用できるメモリの容量は，より大きな伸びを示している。また，VS容量もG(ギガ)からT(テラ)へと拡大された状況にある。一方，ファイルを格納する代表的な入出力装置の代表であるDASDは，格納可能な容量は飛躍的に大きくなってはいるが，アクセス時間の短縮はプロセッサやメモリほどの率で達成されていない。

これらの状況から，利用可能な大きなメモリを活用し，入力装置に対する物理的な操作を削減する方式が有効と考えられる。入出力仮想化のねらいはまさにここにあり，入出力操作の省略は機械的な操作時間を電気的な操作時間に置き換え，またOSオーバヘッドの削減を図る有効な手段となる。

(2) XPL(拡張プログラムローディング)

プログラムをライブラリからローディングする際に生じる出力を削減する目的で開発された機能がXPL(Extended Program Loading)である[1]。**図11**に示すように，XPLではVS上に仮想ライブラリを持ち，計算機システムの初期設定時に仮想化の対象となるプログラムライブラリを仮想ライブラリ内に格納する。

このような仮想ライブラリを持つことによって，プログラムローディングの要求が生じたとき(LOAD，LINKなど)，仮想ライブラリから要求されたプログラムを探し，指定された領域にコピーする。その後，プログラム内のリロケーション情報を書き換え，実行可能な状態として処理を完了する。XPLによって，プログラムローディングに要する入出力を完全に省略でき，アクセス時間をミリ秒からナノ秒に短縮することが可能となる。

(3) HAF(汎用ファイル仮想化機能)

大容量のVS領域を入出力仮想化に使用することを目的として開発された機能がHAF(High performance Access Facility)である[7),8)]。HAFは，大容量のVS領域にファイルを論理的に取り扱い，ブロック単位でVS上に格納(書込み)・検索(読出し)する機能を提供している。HAFは汎用的なユーザーインタフェースを用意しているため，アクセス法だけでなく，サブシステムやアプリケーションプログラムからも使用できる(**図12**)。

アクセス法がHAFを利用する例を**図13**に示す。アクセスは，ブロックの読込み要求がユーザープログラムから発生すると，HAFに対して問い合わせを行う。HAFはVS領域上に該当ブロックを検索する。もし存在すれば，アクセス法の指定した領域にブロックをコピーする。存在しない場合は，初めてのアクセスとみなしてアクセス法がDASDからブロックを読み込む。その後，HAFに対してブロックの保存要求を行いVS上に格納する。つまり，アクセスしたブロックだけをVS上に保存することにする。

VOS3/ASでは，VSAMデータセットにHAFが適用されている[6)]。したがって，カタログやOLTPでのVSAM利用のデータベースに対して有効な機能と考えられる。

(4) PREST(ジョブ間並列同期転送機能)

オンライン処理の完了後に行われる夜間ジョブでは，限られた時間内に処理を完了しなければならない。ますます増大化する夜間ジョブは深刻な問題である。この問題を解決する一つの機能がPREST(Parallel Reference and Synchronized Transfer Facility)である。

PRESTは，ジョブ間で引き継ぐ一時的なSAMファイルを対象としている。PRESTの処理方式を**図14**に示した。出力ジョブが中間ファイルを書き込み，入力ジョブがそのファイルを読み込むことによって処理を完了するような処理を想定している。この場合，従来は出力ジョブの完了を待ってから入力ジョブが開始するという方式である。したがって，出力ジョブが中間ファイルの出力ネックで入力ジョブを実行することができなかった。

そこで，PRESTでは，出力ジョブの実行と入力ジョブ実行を同時に行わせることを可能とした。つまりPRESTは，VS上に大きなバッファを確保し出力ジョブの書込みファイルをVS上に行い，入力ジョブのファイルに対する読込みをVS上から行う。このような入出力操作の仮想化(シミュレーション)

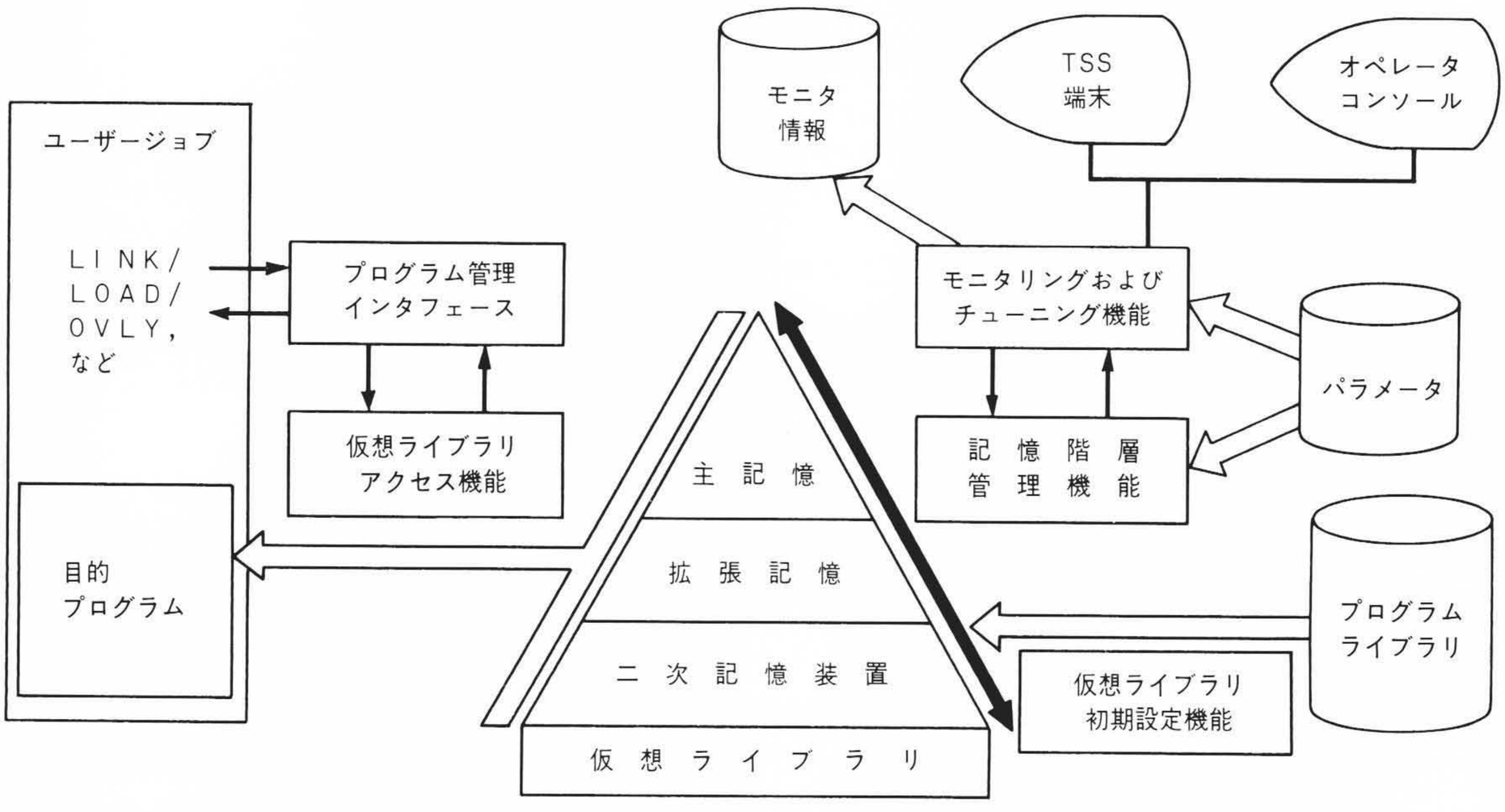
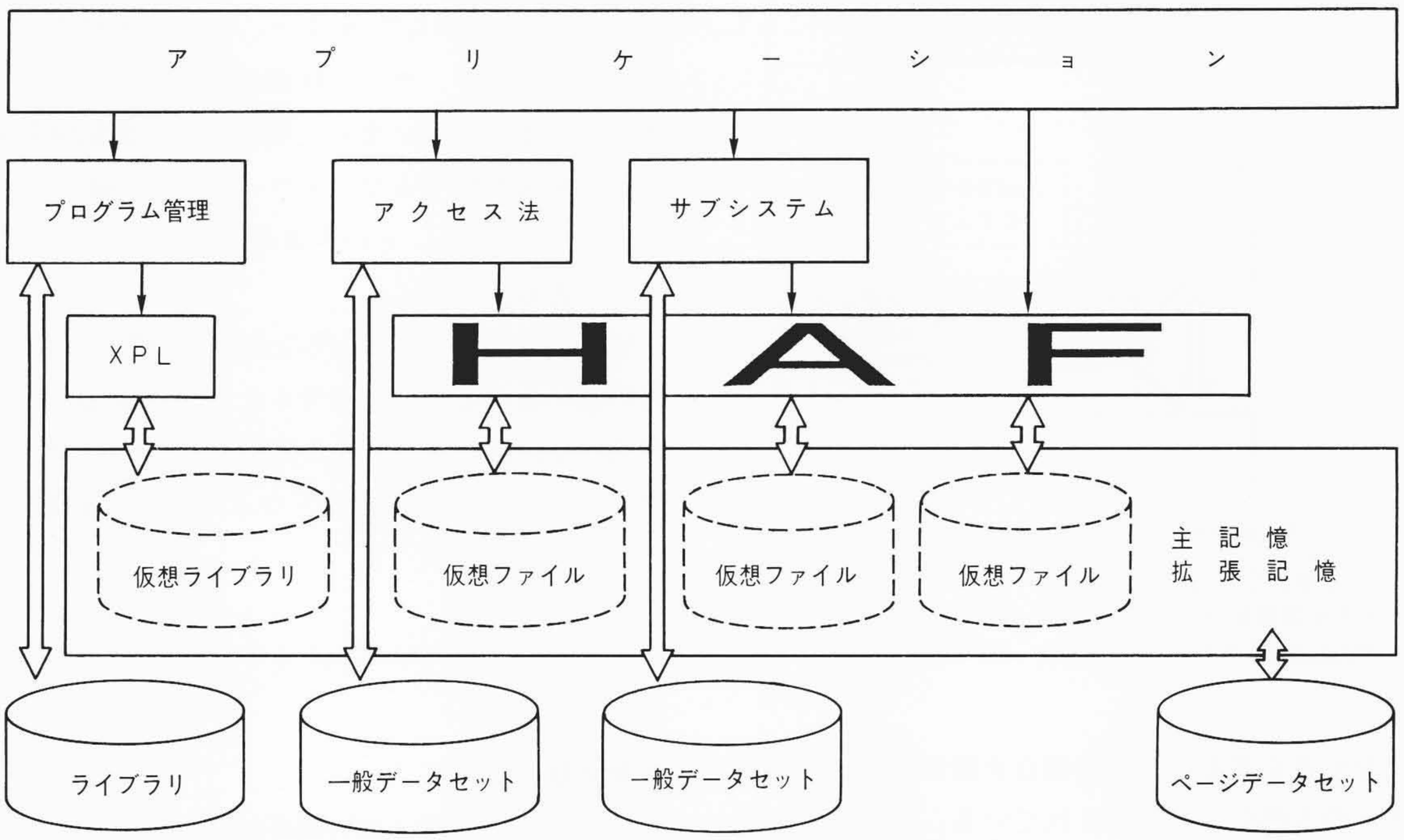


図11 XPL(拡張プログラムローディング)の概要 磁気ディスク上のプログラムライブラリを仮想記憶上にあらかじめローディングしておき，ローディング要求時にメモリ上のコピーとする〔m(ミリ)からμ(マイクロ)へを実現〕。

を行うことで，出力ジョブと入力ジョブの中間ファイルに対するアクセス時間を大幅に短縮するとともに，OSの入出力オーバヘッドを削減することができる。

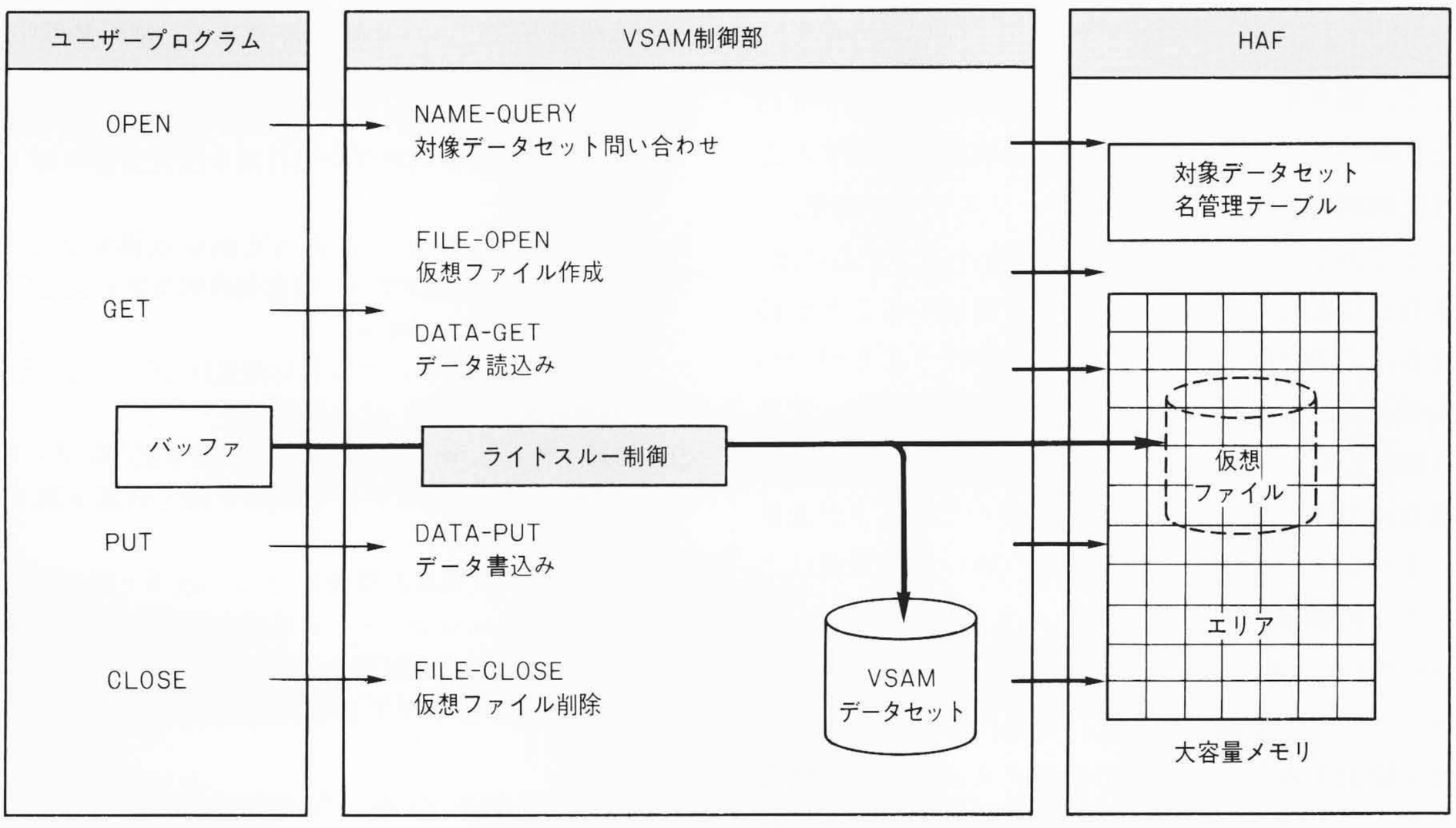
4.4　多重プロセッシング高性能化

多重プロセッサでは，プロセッサの多重度に比例した性能向上が理想である。しかし，ハードウェアおよびソフトウェアを共有していることから競合が発生し，性能の低下を招く。



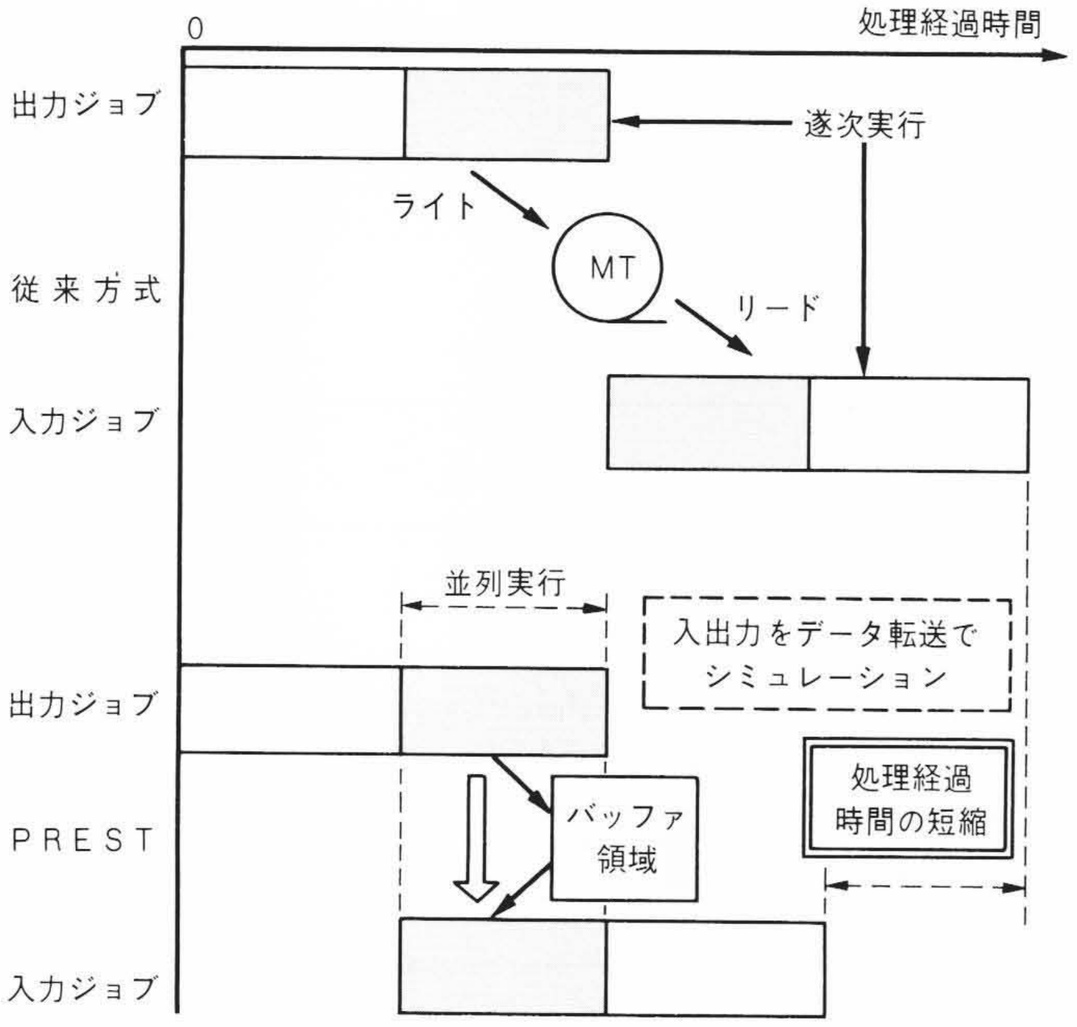
注：略語説明　XPL (Extended Program Loading)，HAF (High Performance Access Facility)

図12　HAFの位置づけ　　HAFは汎用的なインタフェースを用意しているため，アクセス法，サブシステムおよびアプリケーションからも直接利用可能である。



注：　HAFインタフェースマクロ，　VSAMインタフェースマクロ，　データの流れ

図13　HAFのVSAMへの適用　　仮想記憶領域に大きなHAF領域を確保し，アクセスしたブロックをHAF領域内に格納することでディスクアクセスを不要とした〔m(ミリ)からμ(マイクロ)へを実現〕。



注：略語説明 PREST(Parallel Reference and Synchronized Transmission)

図14 PRESTによる並列ジョブ実行の効果 バッチ処理などの中間ファイルへの入出力を仮想記憶上でシミュレーションし，入出力を不要とする〔m(ミリ)からμ(マイクロ)へを実現〕。

そこで，競合を小さくするくふうと競合を制御・管理する方式の改善があらゆる部分に必要とされている。競合状態の管理方式では，使用する資源(情報)に対してロック(lock)をかけ，使用後は解放する。もし，すでにロックがかけられている場合，処理は待たされる。OSが扱うような計算機システム全体にかかわる資源へのアクセスでは，ロック中の源の解除がなされるまでループして待つ。これをスピンループと呼び，このような状況が多発すると実効的な計算機の性能が低下する。

したがって，高多重プロセッサでの高性能化を達成するためには，まず第一に，スピンループとなる状況を回避すること，第二は大規模化に伴う負の効果(ラージスケール効果)を解消することである。スピンループの確率を小さくするには，ロックを保有して走行する処理ステップを削減することが必須(す)である。このために，資源の細分化を行うことが一つの解決方法である。しかし，細分化を進めすぎるとロックを確保したり開放したりするオーバヘッドが逆に増大することになり，競合は小さくなっても全体のOSオーバヘッドが大きくなってしまうことになる。したがって，ロックの分割はこのトレードオフの関係から決定される。

VOS3/ASでは，スピンループの削減を目的としてOSの管理する複数のロックの分割を行った。また，ラージスケール効果の弊害を除去するために実行可能なタスクのリスト構造を改善することや，プロセッサ交信の最適化によってスケジューラの高速化を実現した。

5 結 言

進展する多重プロセッサと大容量記憶を実現するデータ空間，スーパー空間，そして，それらを利用した入出力の削減によるOSの高性能化方式について述べた。これらは，大形計算機の分野での高多重プロセッサ時代に対処する技術であり，また「G(ギガ)からT(テラ)へ」という記憶空間の拡大をもたらしたアーキテクチャの技術的進歩でもある。さらに，半導体記憶技術の進歩によって実現した拡張記憶装置の利用は，入出力を仮想化するソフトウェア技術と融合し，「m(ミリ)からμ(マイクロ)へ」という高速なファイルアクセスを可能としたのである。

VOS3/ASは，1990年代の大形計算機システムを「社会の情報中枢」とするために開発されたOSである[5]。本OSが人間の創造的な活動を支援するより良いマンマシンコグニション(Man Machine Cognition)機能となるために，今後とも努力を続けるつもりである。

参考文献

1) 新井，外：拡張入出力制御方式XVIO(フェーズ1)の開発，情報処理学会OS研究会，84-OS-25-3(1984-12)
2) 新井，外：大形計算機における大容量記憶を利用した高性能化方式，情報処理学会第41回全国大会講演論文集(4)，3D-7(1990-9)
3) 細内，外：拡張記憶を利用したページング方式の性能評価，情報処理学会第41回全国大会講演論文集(4)，4D-2(1990-9)
4) 片田，外：拡張記憶管理機能と仮想記憶管理における拡張記憶利用方式－VOS3/AS基本機能－，情報処理学会第41回全国大会講演論文集(4)，4D-1(1990-9)
5) 久芳，外：大規模計算機システムに求められる機能用件とその実現方式，情報処理学会第41回全国大会講演論文集(4)，3D-6(1990-9)
6) 長須賀，外：ジョブ間並列同期転送機能の開発と評価－VOS3/AS：PREST－，情報処理学会第41回全国大会講演論文集(4)，4D-3(1990-9)
7) 櫻庭，外：汎用入出力仮想化機能HAFの開発，情報処理学会OS研究会，90-OS-47-3(1990-6)
8) 櫻庭，外：入出力仮想化機能と制御方式の開発－VOS3/AS：HAF－，情報処理学会第41回全国大会講演論文集(4)，3D-8(1990-9)
9) 山川，外：大規模計算機システムの運用支援機能の開発，情報処理学会第41回全国大会講演論文集(4)，4D-4(1990-9)
10) 山本，外：入出力仮想化機能HAFのアクセス法への適用とその評価，情報処理学会第41回全国大会講演論文集(4)，3D-9(1990-9)
11) Y.Yoshizawa, et al.: Adaptive Storage Control for Page Frame Supply in Large Scale Computer Systems, ACM/SIGMETRICS at Santa Fe(May, 1988)