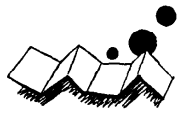
解　説

# マイクロプロセッサによるマルチプロセッサシステムの技術動向†



大　森　健　児††

## 1. 序　論

最近のマイクロプロセッサの多くは，マルチプロセッサ機能を備えている．これは，この十年間にミニコンピュータとマイクロプロセッサの分野で精力的に進められたマルチプロセッサシステムの研究が，広く認識されたことを示す一つの現象である．近い将来にはマイクロプロセッサの多くの分野でマルチプロセッサシステム技術が利用されることと思われる．そこで，ここでは，ミニコンピュータとマイクロプロセッサの分野での今日までのマルチプロセッサシステムの技術動向をまとめることによって，これからの応用分野を考える上での助けとしたい．

ミニコンピュータとマイクロプロセッサでの分野での最初のマルチプロセッサシステムは 1972 年に出現した．その一つは C. mmp[1]であり，他の一つは PRIME[2]である．C. mmp は，マルチプロセッサシステムについてできるだけ多くの知識を得るということを目的に開発されたシステムである．PRIME は信頼性の高いシステムを得ることを目的にしたもので，後のフォールトトレラントシステムへ影響を与えた．1973 年には，ARPA ネットワークでのメッセージ交換システムとして，Pluribus[3]が発表され，1974 年には，8 ビットマイクロプロセッサ 8008 を用いた MICS[4] が発表された．

1975 年以降になるとマルチプロセッサシステムの研究は，最盛期を迎え，ACE[5]，MINERVA[6]，EPOS[7]，Cm*[8]，BT I8000[9] 等が発表された．この時期までは，マルチプロセッサシステムに対する研究的要素が強く，マルチプロセッサシステムのアーキテクチャが研究の対象になる時期であった．

1979 年ごろからは，今までの計算機では，処理時間がかかりすぎるため，あるいは，処理が複雑すぎるため，応用分野とならなかった分野を，情報処理の対象領域とするための一つの手段として，マルチプロセッサシステムが利用されるようになってきた．この時期には，図形処理，シミュレーション，データフローマシンのためのマルチプロセッサシステムが開発された．

マルチプロセッサシステムの利用分野が広がるに従い，マイクロプロセッサでその機能を用意するようになってきた．その一つの例は，プロセスの同期のためのテストアンドセット命令を備えたことである．もう一つの例は，プロセッサ間の通信機能を用意したことである．たとえば，Intel 432[10]では，個別通信と一斉通信の両方の機能を用意している．その他の例は，競合するアクセスを調停するバスアービタがでてきたことである．

## 2. 代表的なシステムとそのアーキテクチャ

今までに発表されている主なマルチプロセッサシステムを応用分野によって分類し，その分野での代表的なシステムとそのアーキテクチャについて述べる．

### 2.1 初期システム

初期のシステムには，C. mmp, PRIME, Pluribus, MICS 等がある．初期のシステムの中で，密結合の結合形態[11]が全て現われた．C. mmp はクロスポイントスイッチ結合を，PRIME はマルチポートメモリ結合を，Pluribus, MICS はバス結合の形態をとった．これらのシステムとこれ以前に開発された汎用計算機でのマルチプロセッサシステムとの違いは，プロセッサのアドレス能力に差があることでこれらのシステムではアドレス能力が小さい．そのため，初期のシステムにおいては，64 k バイト程度のアドレス能力しか持たないプロセッサが，それを越えるメモリ容量を有する共有メモリにアクセスできるようにするために，C. mmp では Dmap が，PRIME では Map が，Pluribus ではバスカプラが，MICS ではアドレス変換器が開発された．

† Technical trends of multimicroprocessor systems by Kenji OHMORI (C & C Systems Labs, Nippon Electric Co., Ltd.).
†† 日本電気(株) C & C システム研究所

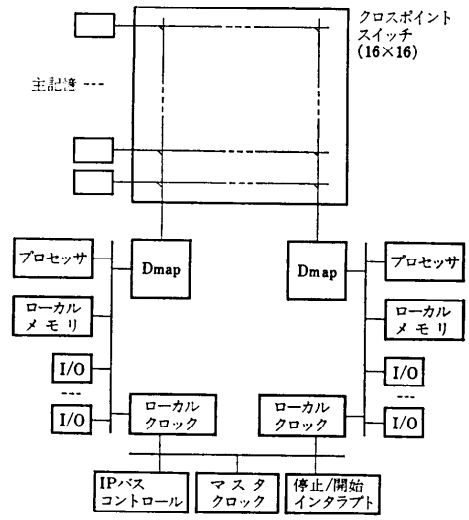



**図-1** C. mmp の構成

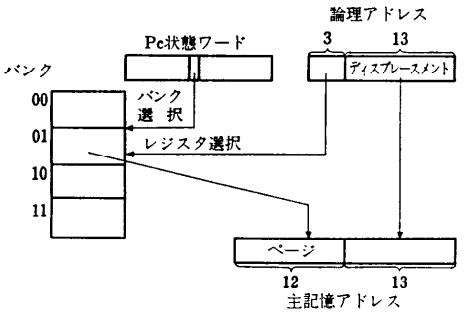



**図-2** C. mmp でのアドレス変換

C. mmp は，最大16台のプロセッサと，2.5 Mバイトの主記憶を **図-1** に示すようにクロスポイントスイッチにより結合したものである．プロセッサの発生する論理アドレスは，**図-2** に示す方法により主記憶のアドレスに変換される．すなわち，$P_c$ 状態ワードにより Dmap 内のバンクを選択し，そのバンク内のレジスタを論理アドレスの上位3ビットにより選択し，そのレジスタより 12 ビットのページを読み出す．そして，12ビットのページと論理アドレスの下位13ビットのディスプレースメントを結合して 25 ビットの主記憶のアドレスを作る．

C. mmp のもう一つの特徴は，オブジェクト指向のオペレーティングシステム Hydra[12] が開発されたことである．このオブジェクト指向の概念は Cm*, Intel 432 へと引きつがれてきている．オブジェクトとは，タイプと許される操作とが与えられた抽象化されたデータであり，モジュール単位でのソフトウェア設計を容易にするものである．

初期システムの中で，実用システムとして利用されたのは Pluribus である．Pluribus は，ARPA ネットワークでのメッセージ交換システムとして作られた．このシステムの特徴は，割り込まれることのない，短時間に処理を完了するストリップと呼ばれるタスクにより，ソフトウェアが構成されていることである．Pseudo-Interrupt Device (PID) と呼ばれるハードウェアキューより，実行可能なストリップの名前を，プロセッサが取ってきて，それを実行する．この実行を完了すると，プロセッサは次のストリップの名前を取りにゆく．なお，プロセッサあるいは I/O デバイスは，実行可能となったストリップの名前を PID に登録する．

### 2.2 フォールトトレラントを指向したシステム

マルチプロセッサシステムに期待されている技術の一つに，高信頼性がある．高信頼性システムを狙ったシステムとして，PRIME, HXDP[3], H-80[14], Tandem 16[15], MICS-MS[16]をあげることができる．このうち，PRIME と MICS-MS は，タイムシェアリングシステムを，H-80 はオンライン制御を，Tandem 16 はトランザクション処理を狙っている．

H-80 はマルチポートメモリを用いて，Tandem 16 はマルチポートのディスク及び I/O 制御部を用いて，システムの二重化により信頼性の向上を図っている．HXDP, MICS-MS は制御の分散により高信頼のシステムを狙っている．

マルチプロセッサシステムでの高信頼性の特徴は，故障の程度に応じて段階的にシステムを縮退できることにある．たとえば，MICS-MS はフォールトトレラントなタイムシェアリングシステムを狙ったシステムで，**2.5** 節で述べる MICS-Ⅱ と同様な構成になっている．各処理部には TSS 用の OS が実装され，TSS ユーザの面倒をみる．ユーザがシステムにログインすると，そのユーザがログインした端末をその時点で管理していた OS のもとにユーザが利用するタスクの集合からなるオリジナルグループが，また，他の OS のもとに故障時にオリジナルグループとなりえるダミーグループが用意される．今，**図-3** に示すように，OS1 を動かしていたプロセッサに故障が起こったとすると，OS1 にオリジナルグループを有していたユーザに対しては，ダミーグループがオリジナルグループとなりその処理を継続する．さらに，次の故障にそなえて新たなダミーグループが他の OS のもとに作られ

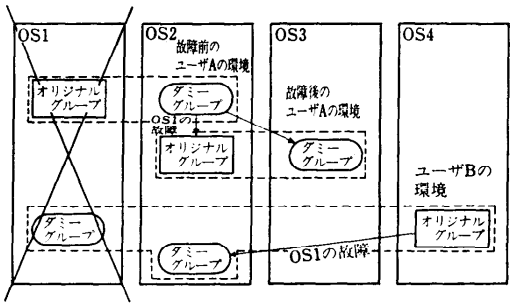



図-3　故障時の縮退

る．また，OS1 にダミーグループを有していたユーザに対しては他の OS のもとにダミーグループが作られる．このようにして，MICS-MS での TSS ユーザは，故障が起こったとしても処理を継続することができる．しかし，この縮退時には処理速度は遅くなる．

### 2.3 タイムシェアリングシステムを指向したシステム

マルチプロセッサシステムを採用する動機の一つに，拡張性がある．マルチプロセッサシステムにおいては，要求性能が増大したとき，従来のように高位の機種へ移行するのではなく，プロセッサ，メモリ等の増設によって対処することができる．そのため，マルチプロセッサシステムの応用分野の一つとして，システムが使われるに従い利用者の増大をともなうタイムシェアリングシステムが選ばれた．

タイムシェアリングシステムを狙ったものとして，PRIME, BTI 8000, MICS-MS, EPOS がある．PRIME がマルチポートメモリの結合形態をとっている以外は，他の 3 システムともバス結合の形態をとっている．これは，バス結合のシステムでは基板の抜き差しにより，簡単にシステムの拡張，縮少が行えるためである．BTI 8000は，サイクルタイムが 66.7 ns の高速バスを用いたシステムである．その構成は図-4 の通りである．また，構成によりスループットがどのように変動するかを示したのが 表-1 である．この表より，プロセッサの台数にほぼ比例して性能が向上することが判る．

### 2.4 高級言語マシンを指向したシステム

マルチプロセッサシステムでは，プロセッサの台数に応じて処理時間の短縮を図ることができる．その方法には，一つの仕事を分割し，それを各プロセッサに割り当てることによって，一つの仕事にかかる処理時間を短くすることを狙った負荷分散と，システムの中

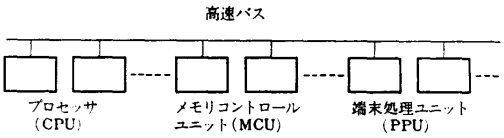



図-4　BTI 8000 の構成

表-1　BTI 8000 のスループット

| CPU | MCU | PPU | スループット (等価な CPU 数) |
|---|---|---|---|
| 1 | 1 | 1 | 1.0 |
| 2 | 2 | 2 | 1.9 |
| 3 | 3 | 2 | 2.7 |
| 4 | 4 | 2 | 3.5 |
| 5 | 5 | 2 | 4.3 |
| 6 | 6 | 2 | 4.9 |
| 7 | 7 | 2 | 5.5 |

で行う必要のある仕事を機能ごとに分け，それをプロセッサに分配することによって，システム全体の処理時間を短くすることを狙った機能分散とがある．

コンカレント PASCAL, PASCAL あるいは APL をインタプリタにより実行することを狙ったマルチプロセッサによる言語マシンに，この二つの形態をみることができる．ACE に実現されたコンカレント PASCAL マシン[17]は，コンカレント PASCAL で書かれた仕事を構成するプロセスを，プロセッサに分配した負荷分散型のマルチプロセッサシステムである．

EPOS は，PASCAL プロセッサ，APL プロセッサ，ファイルプロセッサ等を，マイクロプログラミングを利用して，各々のコンピュータモジュールの上に実現することを狙った機能分散型のマルチプロセッサシステムである．EPOS のアーキテクチャの特徴は，複数のバスを用意していることである．バスの使用は次のようにして行われる．あるコンピュータモジュールからあるコンピュータモジュールへデータを転送する場合には，コンピュータモジュールは全てのバスに対して要求を出す．もし，この要求が複数のバスによって受けつけられた場合には，最初に承認を与えたバスを選び，他のバスに対する要求を取り消す．

### 2.5 仮想計算機を指向したシステム

マルチプロセッサシステムにおいては，それぞれのプロセッサに同一のあるいは異なるオペレーティングシステム (OS) を実行させることによって，一つのシステムの中で複数の OS を動かすことができる．これは，従来の大型計算機での仮想計算機を，形態を変えて実現したものである．仮想計算機の利点は次の通りである．1) 世代の異なる OS や，性質の異なる OS

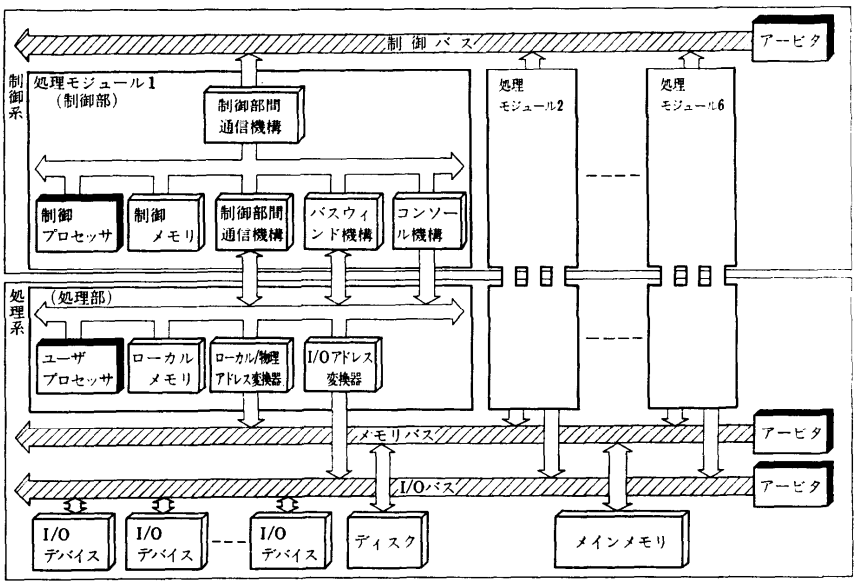



図-5 MICS-II のブロック図

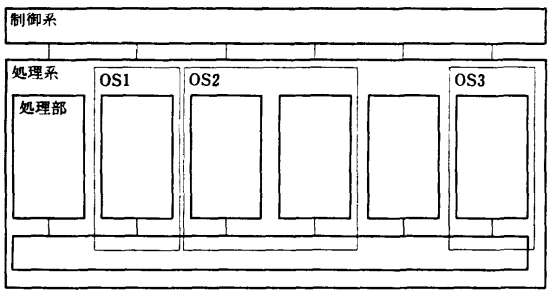



図-6 OS の実装

を同時に走行させることができる．2) OS を動かしたままで，保守用のプログラムや評価用のプログラムを動かすことができる．3) 従来の OS を動かしたままで，次の OS を開発することができる．

マルチプロセッサシステムの上に仮想計算機を実現したシステムに MICS-Ⅱ[18]がある．MICS-Ⅱ の構成は 図-5 に示すようになっている．その特徴は，マルチプロセッサシステム構成の処理系と，ネットワーク構成の制御系とで構成されていることである．処理系には，マルチプロセッサシステム用の OS を含むいくつかの OS が 図-6 のように実装される．OS に対するメモリ，プロセッサ等の資源の割り当ては，制御系によって行われる．資源の割り当てのきっかけを起こすのは，OS が資源の割り当てられていない領域に対してアクセスしたときに起こるアクセスフォールトである．アクセスフォールトが起こったとき，割り当てられていない資源が，あるいは今後使用される見込みの少ない資源が，アクセスフォールトを起こしている OS に割り当てられる．これら一連の処理は，制御系内の制御部間通信によって行われる．また，OS に対するプロセッサの割り当ても，固定的ではなく，プロセッサの稼動率により動的に変化する．

### 2.6 特定目的を指向したシステム

マイクロプロセッサとメモリの価格が急激に下がったために，応用に合わせて独自のシステムを作ることが可能になってきた．最近発表されたマルチプロセッサシステムには，図形処理，シミュレーションあるいは非数値処理等のための専用マシンや，DDMI[19]，X-tree[20]，TOPSTAR[21]，CORAL[22] 等のデータフローマシンがある．データフローマシンの多くは，共有メモリを持たず，ネットワーク結合の形態をとる．一方，専用マシンは，共有メモリを利用しての処理の高速化を狙ったものが多い．

3 次元色彩図形処理のために開発された PSYCO[23] は機能分散と並列処理とを組み合わせたシステムである．Intel 8086 のコントロールプロセッサがシステムの制御を行う一方で，16 台の TMS 9900 が一つの命令で異なったデータを並列に処理する．

離散系のシミュレーションのために開発された

KDSS-1[24]は 64 台までのプロセッサを接続可能なシステムで，分割した作業を各プロセッサに割り当てる負荷分散のシステムである．KDSS-1 の特徴は，同時読出し可能な共有メモリにある．共有メモリは，プロセッサごとに用意されたパケットメモリよりなる．プロセッサが自身のパケットメモリに書き込みを行うと，その時，他の全てのパケットメモリにも同一のデータが書き込まれる．各プロセッサは，そのデータを自身のパケットメモリより読むことができるので，データの同時読み出しが可能である．

## 3. 制御の方法

マルチプロセッサシステムにおいては，システムを作る上で，ソフトウェアをどのようにするかがもう一つの問題であるが，ここでは分散プロセスの方法と代表的なオペレーティングシステムについて述べる．

### 3.1 分散プロセス

負荷分散を行うシステムでは，仕事は分割され，各プロセッサに分配される．各プロセッサでは，分割された仕事をプロセスが処理してゆくが，このとき，異なるプロセッサ上にあるプロセス間でのデータの受け渡しが必要である．

Brinch Hansen[25] の提案する実時間処理のための分散プロセスは次のようになっている．1) 実時間プログラムは，同時に動作するプロセスからなりたっていて，しかも，プロセスは自身の変数にしかアクセスしない．2) プロセスは，他のプロセスの中に定義されている共通のプロシージアを呼ぶことができる．このプロシージアは，プロセス間通信のために使用されるもので，プロセスは条件が満たされるまでこのプロシージアの中で待ちつづける．3) プロセス間の同期は監視領域 (guarded region) と呼ばれる非決定ステートメントにより行われる．

監視領域は次のようになっている．

```
when B1 : S1|B2 : S2|……end
cycle B1 : S1|B2 : S2|……end
```

when ステートメントにおいては，B1, B2,……のうち少なくとも一つが条件を満たすまで，when ステートメントの中で待たされる．一つでも条件を満たすものがあれば，その中の一つ，たとえば $Bi$ を選択し，そのステートメント $Si$ を実行して，when ステートメントを抜けでる．cycle ステートメントは，ステートメントを実行した後，cycle ステートメントを再度繰り返す．その他は when ステートメントと同じである．

プロセス間で send と receive により文字列を転送する場合を考える．このとき，文字列を一時的に記憶するバッファが必要であるが，このバッファの操作を行う process buffer は次のようになる．

```
process buffer; s : seq [n] char
procedure send (c : char)
  when not s.full : s.put (c) end
procedure receive (c : char)
  when not s. empty : s.get (c) end
s : =[ ]
```

このプロセスは，初期動作により s を空にする．文字を受信したいプロセスは，call buffer. receive ($x$) ステートメントを実行する．これにより，buffer プロセスが一つ起こされるが，現在 s が空であるため when ステートメントの中で文字が送られてくるのを待つ．文字を送信したいプロセスは，call buffer. send ($y$) ステートメントを実行する．これにより，もう一つの buffer プロセスが起こされる．これは s が充でないため s に $y$ を書き込み，このプロセスは消滅する．s が空でなくなったため，最初に起こされた buffer プロセスは，s より文字 $y$ を得て，call ステートメントを実行したプロセスにこの文字を渡し，自身は消滅する．

### 3.2 オペレーティングシステム

マルチプロセッサシステムのオペレーティングシステムとしては，C. mmp の上に作られた Hydra, Cm* の上に作られた Star OS[26]，Medusa[27]が知られている．このうち，Hydra と Star OS は，オブジェクトを指向し，ケーパビリティをベースとしたオペレーティングシステムとして，また，Medusa は分散されたオペレーティングシステムとして知られてい

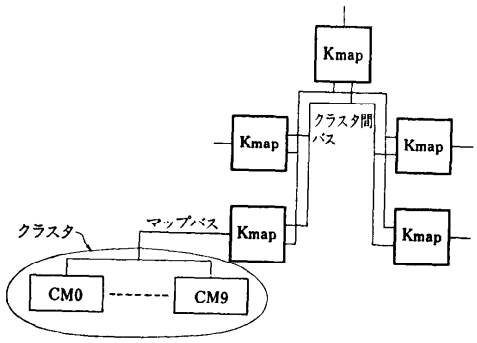



図-7 Cm* の構成

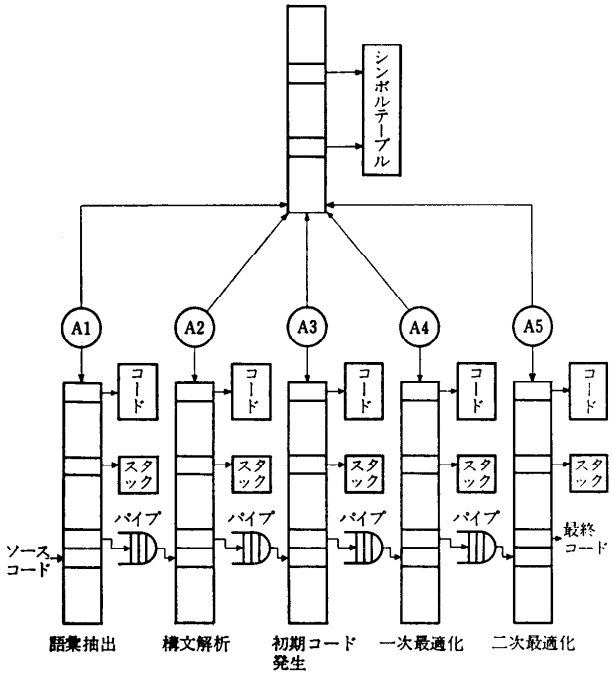



図-8 コンカレントコンパイラ

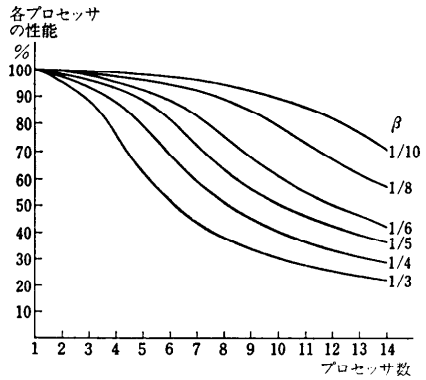



図-9 共通バスに起因する性能低下

る．ここでは，Medusa について簡単に説明する．

Medusa が動くマルチプロセッサシステムは Cm* であるが，これは 図-7 のようなハードウェア構成となっている．Cm* の特徴は，ローカル，クラスタ内，クラスタ外のメモリへのアクセスが，3.5 μs, 9.3 μs, 24 μs というように，より遠いメモリへは，より多くのアクセス時間を必要とすることである．Medusa ではこの特性に合わせ，あるコンピュータモジュール (CM) にはメモリ管理を，また，他の CM にはファイル管理をというように，CM の中でほとんどの処理が閉じるようにオペレーティングシステムが設計されている．すなわち，オペレーティングシステムは，機能的に一つのまとまった作業の単位——これはユーティリティと呼ばれる——に分割される．さらに，このユーティリティは各プロセッサへの分配の単位となるアクティビティに分割される．

図-8 は，コンパイラであるユーティリティを語彙抽出，構文解析等のアクティビティに分割したときの構造を示したものである．シンボルテーブルは，ラベル，変数名等からなる表で，全てのアクティビティが利用する共通のデータである．アクティビティは同一クラスタ内の異なった CM に実装される．語彙抽出のアクティビティから構文解析のアクティビティへの呼び出しは，パイプにより行われる．このパイプを通して，送信側のアクティビティから必要なデータが，受信側のアクティビティへ送られる．受信側のアクティビティは，これにより起動され，所定の処理を行う．処理が終ったとき，必要があればリターンパイプにより結果を渡す．

## 4. 性 能

ここでは，バス結合のマルチプロセッサシステムの性能特性を，MICS-Ⅱ[28]の場合を例にとり説明する．

### 4.1 共通バスに起因する性能

バス結合のマルチプロセッサシステムにおいては，複数のプロセッサが同時にバスにアクセスすることがある．この場合には，一つのプロセッサのみがバスにアクセスすることを許され，他のプロセッサは待たされる．そのため，バス結合のシステムにおいては，プロセッサの台数が増大するに従い，プロセッサの処理速度は遅くなる．これを示したのが 図-9 である．なお，図で $\beta$ は，プロセッサが共通バスにアクセスする確率である．共有メモリへのアクセス比率が高い場合には $\beta$ は高く，ローカルメモリへのアクセス比率が高い場合には $\beta$ は低くなる．

### 4.2 バスアービタに起因する性能特性

共通バスで衝突したときに，どのプロセッサにアクセスを許すかは，優先度の与えかたにより決まる．この優先度制御のため，個々のプロセッサの処理速度の低下は，図-9 のようにはならない．MICS-Ⅱ においては，図-10 の構成のアービタトリにより優先度制御を行った．このため，①のプロセッサは，⑤あるいは

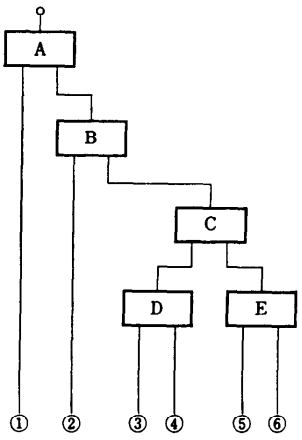



図-10　バス制御のためのアービタトリ

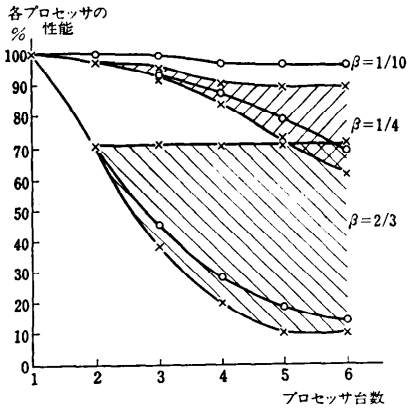



図-11　バスアービタに起因する性能低下

⑥のプロセッサより8倍アクセスしやすくなっている．①のプロセッサと⑥のプロセッサの処理速度の低下を示したのが図-11である．図で上側の×印は①のプロセッサの性能低下を，下側の×印は⑥のプロセッサの性能低下を，○印は平均の性能低下を示す．図より，バスへのアクセス頻度が高くなると，優先度の差が大きく現われ処理速度に大きな開きがでてくることが判る．

## 5. まとめ

ここ十年間のミニコンピュータとマイクロプロセッサの分野でのマルチプロセッサシステムの動向を述べた．密結合のシステムで始まったこの分野は，近年のネットワーク結合のシステムまで大きな広がりを見せている．この傾向は今後も大きく変わることはなく，用途に合わせたシステムが多数でてくることと思われる．また，もう一つの期待は，一つのチップの中にいくつかのプロセッサが組み込まれたプロセッサアレイの出現である．

## 参考文献

1) Wulf, W. A. and Bell, C. G.: C. mmp: A multi-miniprocessor, 1972 AFIPS Fall Jt. Computer Conf., pp. 765-777 (Dec. 1972).
2) Baskin, H. L., Borgerson, B. R. and Roberts, R.: PRIME-A modular architecture for terminal-oriented systems, Proc. AFIPS Spring Jt. Computer Conf., pp. 431-437 (1972).
3) Ornstein, S., Crowther, W., Kraley, M., Bressler, R., Michel, A. and Heart, P.: Pluribus: A reliable multiprocessor, Proc. AFIPS Nat. Computer Conf., pp. 551-559 (1975).
4) Ohmori, K., Koike, N., Nezu, K. and Suzuki, S.: MICS-a multi-microprocessor, Proc. IFIP Cong., pp. 98-102 (1974).
5) Iizuka, H., Ishii, O., Fujii, K., Ohmote, R. and Furuya, T.: ACE-A new modular computer architecture, Proc. US-Japan Computer Conf, pp. 36-41 (1975).
6) Widdoes, L. C. Jr.: The MINERVA multimicroprocessor. 3rd Annual Symp. on Computer Architecture, pp. 34-39 (1976).
7) Maekawa, M., Yamazaki, I., Maeda, A., Miyata, M., Kamiya, S. and Kasai, H.: Experimental polyprocessor system (EPOS)—Architecture, Proc. Int. Symp. Computer Architecture, pp. 188-195 (1979).
8) Swan, R. J., Fuller, S. H. and Siewiorek, D. P.: Cm*-A modular multi-microprocessor, Proc. AFIPS Nat. Computer Conf., pp. 637-644 (1977).
9) Lewis, G. R. and Henry, J. S.: The BTI 8000-homogeneous, general-purpose multiprocessing, Proc. AFIPS Nat. Computer Conf., pp. 513-528 (1979).
10) iAPX General Data Processor Architecture Reference Manual, Intel Corp. (Jan. 1981).
11) 高橋義造：並列処理のためのプロセッサ結合方式，情報処理 23 巻，3号，pp. 201-209 (Mar. 1982).
12) Wulf, W. A., Cohe. E., Corwin, W., Jones, A., Levin, R., Rierson, C. and Pollack, F.: HYDRA: The kernel of a multiprocessor operating system, Comm. ACM, Vol. 17, No. 6, pp. 337-345 (June. 1974).
13) Jensen, E. D.: The Honeywell Experimental distributed-An Overview, Computer pp. 28-38 (Jan. 1978).

14) Kamiuchi, Y. and Nakanishi, H.: H-80 a load-sharing, $N$:1 backup multisystem, COMP CON Spring, pp. 261-264 (1978).
15) Katzman, J. A.: A Fault-tolerant Computing System, IEEE Int. Conf. of System Sciences (Jan. 1978).
16) 大森，小池，近藤：マルチプロセッサシステムでのフォールトトレランスな TSS 実現に関する考察，信学研資 EC 81-8 (1981).
17) 古谷立美：マルチプロセッサシステムにおける Concurrent Pascal マシン，情処学論，21 巻，2 号，pp. 125-132 (Mar. 1980).
18) 大森，小池，山崎，大宮：計算機複合体 MICS-Ⅱの設計思想と構成，情処学論，20 巻，3 号，pp. 235-242 (May 1979).
19) Davis A. L.: The Architecture and System Method of DDM1, A Recursively Structured Data Driven Machine, Proc. Fifth Int. Symp. Computer Architecture, pp. 1-7 (April 1978).
20) Despain, A. M. and Patterson, B.: X-tree: A tree-structured multiprocessor computer architecture, Proc. 5th Ann. Symp. on Computer Architectures, pp. 144-151 (1978).
21) Suzuki, T., Kurihara, K. and Motooka, T.: Procedure Level Data Flow Processing on Multiprocessors, Journ. Information Processing, Vol. 5, No. 1 (1982).
22) Takahashi, Y., Wakabayashi, N. and Nobutomo, Y.: A binary Tree Multiprocessor: CORAL, Journ. Information Processing, Vol. 3, No. 4, pp. 230-237 (Feb. 1981).
23) Kubo, M., Taguchi, Y., Agusa, K. and Ohno, Y.: Multi-microprocessor system for three-dimentional color graphics, Proc. IFIP Cong., pp. 145-150 (1980).
24) Nakagawa, T., Kumata, I., Hasegawa, T., Matsumoto, T., Abe, K., Kobayashi, N. and Aiso, H.: A multiprocessor approach to discrete system simulation, COMPCON spring 80, pp. 350-355 (Feb. 1980).
25) Brinch Hansen, P.: Distributed processes: A concurrent programming concept, Comm. ACM Vol. 21, No. 11, pp. 934-941 (Nov. 1978).
26) Jones, A. K., Chansler, R. J. Jr., Durham, I., Schwans, K. and Vegdahi, S. R.: Star OS, a multiprocessor operating system for the support of task forces, Proc. 7th Symp. Operating Systems Princeples, pp. 117-127 (Dec. 1979).
27) Ousterhout, J. K., Scelza, D. A., and Sindhu, P. S.: Medusa: An Experiment in Distributed Operating System Structure, Comm. ACM Vol. 23, No. 2, pp. 92-105 (Feb. 1980).
28) 大森，小池，山崎，大宮：計算機複合体 MICS-Ⅱのシステム評価，情処学論，20 巻，2 号，pp. 130-137 (Mar. 1979).

(昭和 57 年 4 月 12 日受付)