解　説

マルチプロセッサのメイン・メモリ*

堀　越　　弥** 百　瀬　次　生**

1. はじめに

近年，計算機の処理能力に対する要求はますます高まっており，従来のように1台の計算機では処理しきれずに，複数台の計算機を用いる場合が増えてきている．一方では，計算機システムにさらに高い信頼性が期待されるようになっていること，LSI 化の進行により相対的にプロセッサ (Processor) の価格が下がっていること，既存のシステムと有機的に結合して運用の効率を上げたいという要求などが，ますますこのような動きを助長しているといえる．

さて，このような計算機の結合したシステムはその種類も多様で，単純に分類することはできないが，

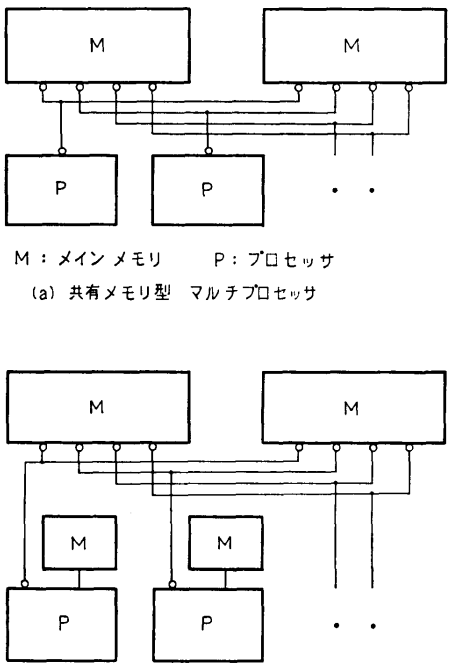
（1） 通信回線によって比較的遠距離にある計算機を接続する計算機網 (Computer Network)，

（2） チャネル結合などにより比較的近距離に存在する計算機を結合する計算機複合体 (Computer Complex)，

（3） メイン・メモリ (Main Memory) を共有するマルチプロセッサ (Multi-Processor) に分けられる[1),2)]．このうち計算機網，計算機複合体の構成では，ハードウェア的にはほぼ従来の計算機を基本とし，これらを結合した制御はソフトウェアの役割になるのが普通である．したがって，ハードウェア方式という視点から計算機の複合したシステムを見る場合には，マルチプロセッサが重要になってくる．以下，マルチプロセッサ方式について，メイン・メモリに関係する部分に焦点を合わせて概観することにする．

2. 共有メモリ型マルチプロセッサ

マルチプロセッサ方式はメイン・メモリという視点からは**図-1** に示すように，共有メモリ型と分散メモリ型に分類することができる．共有メモリ型マルチプロセッサとは，どのプロセッサからも差別なしに参照できるメイン・メモリを中心に構成されるマルチプロセッサである．したがって，どのプロセッサもプログラム実行能力を持つなら，メイン・メモリ上のいかなる処理にも各プロセッサが使え，プロセッサをもっとも有効に活用することができる．これに対して，共有メモリ型マルチプロセッサのハードウェア方式上の問題点は，



M：メイン メモリ　　P：プロセッサ

(a) 共有メモリ型　マルチプロセッサ

(b) 分散メモリ型　マルチ プロセッサ

図-1　マルチプロセッサの構成方法

（a） マルチプロセッサ間の競合を解決する論理に時間が費され，メイン・メモリのアクセス時間が長くなること，

（b） 他のプロセッサがメモリモジュールを使用している間待たされること，

* Main Memory Architecture of Multi-Processor by Hisashi HORIKOSHI and Tsuguo MOMOSE (Hitachi Central Research Laboratory)

** (株) 日立製作所中央研究所

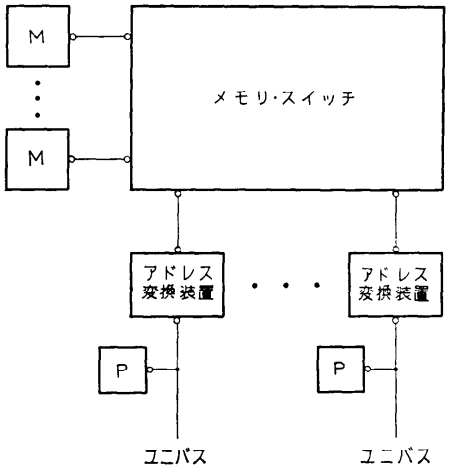
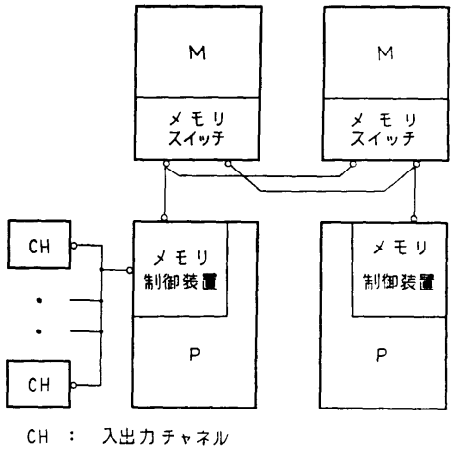


図-2 C. mmp のハードウェア構成



CH ： 入出力チャネル

図-3 汎用マルチプロセッサの構成

（c）（a），（b）の処理を高速で行なうため，かなりのハードウェアが必要になること，

などである．

この方式の例には数多くのものがあるが，次にその1例を示す．

（1） C. mmp (Carnegie-Mellon University の multi-mini-processor)[3),4)].

C. mmp は共有メモリ型マルチプロセッサの典型で，**図-2** に示すように最大16 台のプロセッサがメイン・メモリを共有することができる．メイン・メモリは1モジュール 128 kB（キロバイト）のものを最大16台接続することが可能で，1モジュールはアクセス時間 250 ns，サイクル時間650 ns のものを8ウェイにインタリーブして構成されている．プロセッサには PDP-11 を使用しており，プロセッサとメイン・メモリの間にはアドレス変換装置とメモリスイッチが存在する．アドレス変換装置は仮想アドレスを物理アドレスに変換するもので，具体的にはプロセッサ内で生成された 18 ビットのアドレスを，アドレス変換装置内のテーブルを利用して 21 ビットの物理アドレスに変換する．この処理に 50 ns かかる．この装置はかならずしもマルチプロセッサを実現するために必要となるものではない．むしろ，マルチプロセッサを有効に生かすに十分なメイン・メモリ容量を確保するためのものといえる．さて，アドレス変換装置での処理が完了すると，生成された 21 ビットのアドレスにより，いずれかのメモリモジュールを選択し，メモリスイッチへ要求を送り出す．対応するメモリモジュールが空いていれば受け付けられて，メイン・メモリの参照が開始される．メモリスイッチの処理に片道 100 ns を要するので，アクセス時間は100＋250＋100 ns になる．

C. mmp は接続できるプロセッサ台数は 16 台とかなり多いが，その基本は次に述べる汎用機でのマルチプロセッサの方式と同じものである．

（2） 汎用マルチプロセッサ

共有メモリ方式はすでに多くの商用機で用いられている．大型計算機の場合の典型的構成は**図-3** のようである．プロセッサ内にメモリ制御装置とよばれるものがあり，プロセッサ内の命令実行に派生して発生するメイン・メモリ参照要求と，入出力動作に関係してチャネルから発生する要求を絞り，メイン・メモリへ送出する役割をしている．メイン・メモリにはメモリスイッチが付属していて，他のプロセッサからの要求との競合を解決する．よく見られる変形としては，各チャネルからの要求をチャネル制御装置とよばれる装置で絞ってメモリ制御装置へ接続する場合，あるいはチャネル制御装置を直接メモリスイッチへ接続する場合などがある．

図-3 の構成は一見すると図-2 の構成とかなり差があるように見えるが，図-2 のユニバスの制御はプロセッサ内のバス制御装置が司っているわけで，これは図-3 のメモリ制御装置に等価である．また，図-2 のメモリスイッチをメイン・メモリ対応に横に切って分割すれば，図-3 の構造になる．実際，汎用マルチプロセッサのメモリスイッチでも，物理的位置は1つの場所に集められ，むしろ図-2 のように示した方がよい場合も多い．

それでは，このような汎用マルチプロセッサのメイン・メモリ接続方式は，どういう点に注意して決めていくかというと，概略次のようになろう．

（a） チャネルをプロセッサを介して接続するか否かが最初の分岐点で，これが決まると，

（b） どの程度マルチポート (multi-port) 方式を採用し，どの程度デージチェイン (daisy-chain) 方式を採用するかが第2の問題となる．これは，性能，信頼性，種々の構成の組み易さ，拡張のし易さが判断規準になる．

（c） 最後はきめの細い性能上の検討が必要になる．この時特に大事なのは，チャネルからのメイン・メモリ参照時間に対する配慮である．現在の入出力装置は規定時間以内にデータが到着しないと，いわゆるオーバラン (overrun) となり，誤動作になるものが多い。したがって，多数の要求がメイン・メモリでぶつかるマルチプロセッサ構成では，つねに緊急度の高い入出力装置の要求が早く処理されるように，方式の決定に注意を配る必要がある．

3. 分散メモリ型マルチプロセッサ

メイン・メモリを共有する方式をマルチプロセッサと分類するなら，その上分散メモリ型というのは奇妙な話である．しかし，この点はあまり難しく考えないで，図-1 の(b)のように基本的にはメイン・メモリを共有するが，個々のシステムの設計段階での考慮から，プロセッサ固有のメモリを分散して所有する方式を，分散メモリ型マルチプロセッサと称することにする．

一般的にいって，マルチプロセッサ方式を採用した以上，ソフトウェアの視点からの分散型の利点はあまりないといえる．何故ならこの方式ではすべてのメイン・メモリ領域を個々のプロセッサから一様には参照できないわけで，そのような制限を満たすようにプログラムを構成し，またデータの配置や移動を行なわなければならない．これは性能を低下する大きな要因になる．しかし，もしこの問題が解決できれば次のような効果を期待することができる．

（a） ハードウェアにとって多数のプロセッサからの要求を高速で処理することはきわめて難しい問題であるが，分散メモリ化によりこれが緩和される．

（b） 障害時にメイン・メモリの内容がどの程度信用できるかは，その後システムを回復する際の重要な点になるが，分散メモリ化によりその分析が容易になることが期待できる．

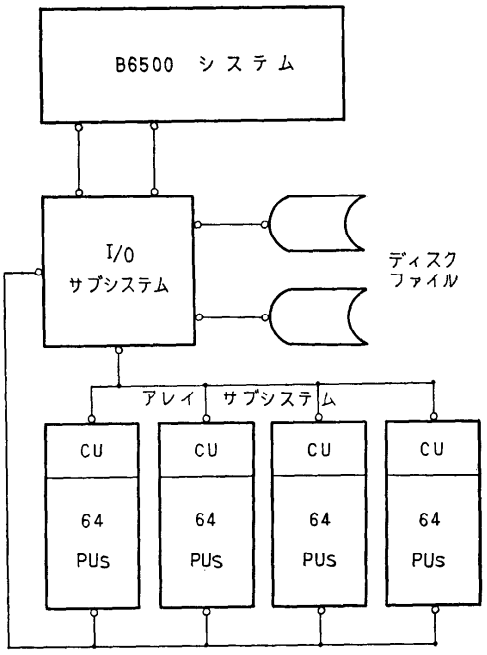


図-4 ILLIAC IV システムの構成

（c） 間接的なことであるが，ソフトウェアを分散メモリ型に改変したことにより，入出力装置，プログラムといったリソースでの競合も低減し，個々のプロセッサがより能力を発揮することができよう．しかし，逆にリソースの融通がしにくくなり，より多くの入出力装置，メイン・メモリなどが必要になる可能性もある．

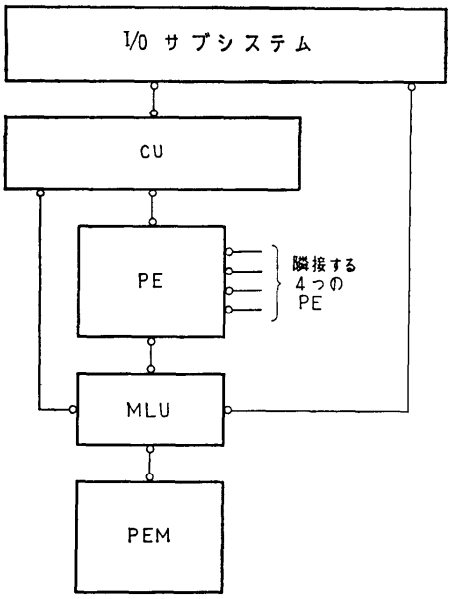
次に，2つのシステムを例として，分散メモリ型マルチプロセッサとメイン・メモリの関係について考察してみることにする．

（1） ILLIAC-IV・システム[5),6)]

厳密な意味での分散メモリ型マルチプロセッサはほとんど存在しないが，ILLIAC-IV はその貴重な例の1つである．ILLIAC-IV システムは図-4 に示すように，

- B 6500・システム
- ディスク・ファイル
- I/O・サブシステム
- アレイ・サブシステム

からなる．アレイ・サブシステムは4つの Quadrant と呼ばれるものからなり，1つの Quadrant は1つの CU (Control Unit) と 64 の PU (Processing Unit)



CU : Control Unit
EL : Processing Element
MLU : Memony Logic Unit
PEM : Pnocessing Element Memory

図-5 PU の構成

からなっている．PU は**図-5** に示すように PE (Processing Element) と MLU (memory Logic Unit) と PEM (Processing Element Memory) の3つの部分からなる．

次に，ILLIAC-IV・システムがどのように動作するかというと，アレイ・サブシステムは B 6500・システムに接続される特殊周辺機器であると考えるとわかりやすい．すなわち，B 6500・システムは I/O・サブシステムを介してアレイ・サブシステムを制御し，ディスク・ファイルを介してアレイ・サブシステム用のプログラムやデータを交換することになる．B 6500・システムはアレイ・サブシステムで処理すべき仕事を，まずアレイ・サブシステムで実行可能な形に変換して，自分のメイン・メモリ内に用意する．これが完了すると I/O・サブシステムに通常の入出力装置を起動するかのごとく指令を発行し，I/O・サブシステムの制御のもとに，B 6500・システムのメイン・メモリ内のプログラムとデータをディスク・ファイルへ移し，引き続いて各 PU 内の PEM へ配置する．この間，CU と PE は一部情報の通路にはなるが，基本

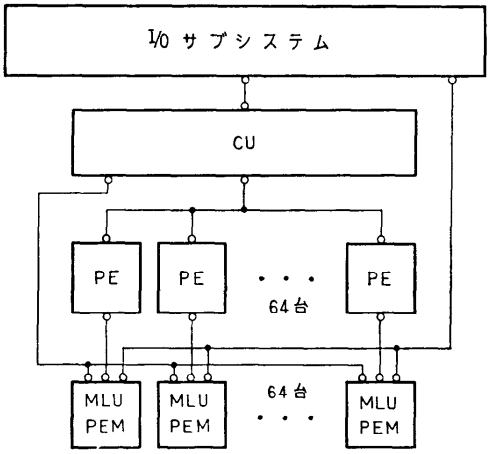


図-6 ILLIAC IV のメイン・メモリの共有方式

的にはこの処理には関与しないと考えることができる．さて，このように PEM 上のプログラムとデータの配置が終了した段階で，B 6500・システムはふたたび I/O・サブシステムに指令を発行し，CU 内の状態を初期設定することを指示する．I/O・サブシステムはこの情報を B 6500・システムのメイン・メモリから読み出し，CU 内の各レジスタへ書き込む．この書き込みによって CU は動作を開始することになる．すなわち，この時 CU 内の ICR (Instruction Counter) がプログラムを開始すべきアドレスに設定され，CU はこのアドレスから実行すべき最初の命令が読み出され，以下逐次実行される．

いったん CU が命令処理を開始すると，CU の実行する命令は PEM 全体から読み出される．したがって，PU から見ると全 PEM が一様に見えるわけで，**図-6** のような構成と考えられる．全メイン・メモリを参照できるのは CU と I/O・サブシステムだけなので，図-1(b)に示した分散メモリ型そのものではないが，きわめて近い構成といえる．アレイ・サブシステムでは命令語はすべて CU で解釈され，1本の流れしかない．それに対し演算処理は各 PU で並列に行なわれるので SIMD (Single Instruction Multiple Data) ということがある．この記法に従えば，通常のマルチプロセッサは MIMD になる．

このように分散したメイン・メモリを参照するために，命令語のアドレッシングは**図-7**(次頁参照)のように，5つの部分に分けてなされる．すなわち，全体で 0～24 の 25 ビットからなるが，0～4 は使用されず．

(a) 5～15 の 11 ビットが PEM 内の 2048 語を

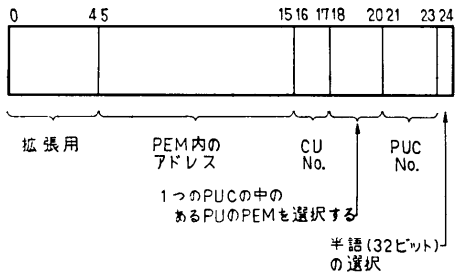


図-7 ICR (Instruction Counter) の形式

アドレッシングするのに用いられる.

(b) 16～17 は CU の指定, すなわち, どの Quadrant かの選択に用いられる. 実際の ILLIAC-IV は 1つの Quadrant しか作られていないので, この部分は意味がない.

(c) 18～20 は 8 台の PU からなる PUC (Processing Unit Cabinet) の中の 1 台の PU の PEM の選択に用いられ,

(d) 21～23 は PUC の選択に用いられる.

(e) 24 ビットは, PEM の 1 語が 64 ビット語からなるため, 32 ビットの命令語が左右のどちらの半語に存在するかを示すためのものである.

さて, このようなアドレッシング方式で読み出された命令語は各マイクロ命令に分解されて, 各 PU に伝えられる. このマイクロ命令で用いるメモリアドレスは基本的には 11 ビットで, 各 PU 内の PEM しか参照することはできない.

以上のように ILLIAC-IV のアレイ・サブシステムは, 一般の計算機と一見かなり違った構造に見えるが, もう一度図-5 と図-6 にもどって, メイン・メモリの構造という面から見ると, **図-8** のように書き直すことができる. すなわち, PEM は

- CU を介しての I/O サブシステムの要求と,
- CU 内の命令語, 演算数参照の要求と,
- PE からの要求,

の 3 つの要求を受け付けて, 対応する装置とデータを交換するわけで, 図-3 に示した汎用マルチプロセッサの構造と大差ないことがわかる.

(2) 汎用マルチプロセッサ

汎用マルチプロセッサの場合の分散メモリ方式は, 一部のメインメモリは個々のプロセッサで持ち, 一部のメイン・メモリは共有するという形で構成されることが多い. この場合, 共有する情報はプログラムでもデータでも可能であるが, いずれにしても新たな技術

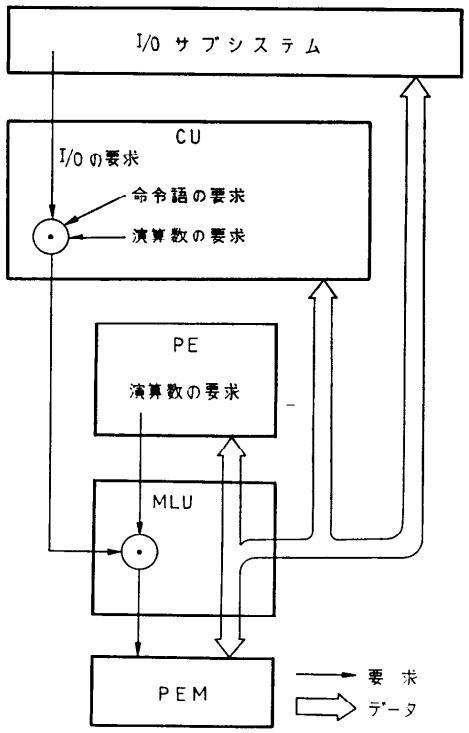


図-8 ILLIAC IV のアレイ・サブシステムのメイン・メモリの構成概念図

的問題はあまりなく, 共有メモリ型マルチプロセッサまたはユニプロセッサの変形と考えることができる.

これに対して, **図-9** (次頁参照) に示すような最近のキャッシュを有するマルチプロセッサは, ある意味で分散メモリ型マルチプロセッサの 1 例と考えることができる. この方式はメイン・メモリから毎回プログラムやデータを読み出すのでは時間がかかるので, 各プロセッサ内にメイン・メモリの写しを分散して持ち, アクセス時間を短縮しようというものである. この場合, ILLIAC-IV などとは違って, 分散メモリの存在をプログラマには意識させずに一様に見せるための工夫がなされている. 通常はデータの変更はキャシュ, メイン・メモリとも同時に行なっておき, あるプロセッサがあるアドレスの内容を変更すると, そのアドレスを他のプロセッサに送出し, 他のプロセッサのキャシュ内のデータを無効にし, 再度メイン・メモリから読み込む. しかし, この方式はプロセッサ台数が増すとアドレス変更のトラフィックがきわめて大きくなるので現実的でなくなる. この点はハードウェア方式のみで分散メモリ型を実現するのは難しいことを示

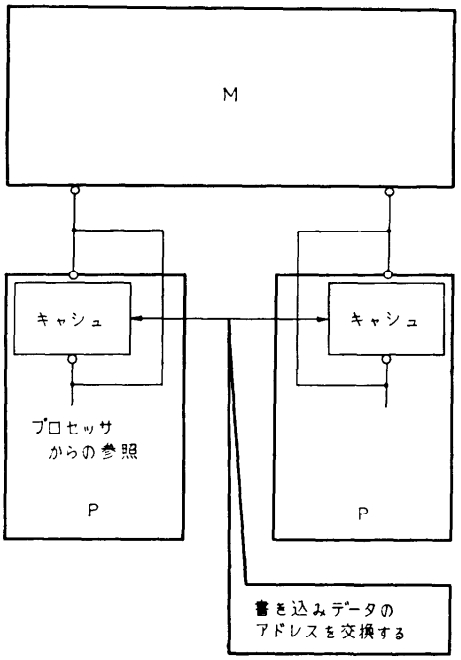


図-9 汎用マルチプロセッサのキャッシュ方式は分散メモリ型の例である.

す1例といえよう.

4. マルチプロセッサの性能

マルチプロセッサ構成がそれほど一般的でなかった時には,マルチプロセッサにすることのおもな目的は信頼性の向上に置かれていた.しかし,近年この方式が広く採用されるにつれて,性能の問題もしだいに重要性を増している.マルチプロセッサの性能の比較はいろいろな要因が関係するためにかなり複雑になる.たとえば,一般に1台に比べ,2台のマルチプロセッサでは性能は2倍以下と考えられているが,各々がもっていたメイン・メモリを合わせて,2倍の容量のメイン・メモリを2台のプロセッサに接続すると,メイン・メモリが有効に生かされて2倍以上の性能になることも少なくない.したがって,次の3つの段階に分けてマルチプロセッサの性能を考える必要がある.

(1) メイン・メモリの競合による性能低下[3),7)]

これはハードウェア的問題が中心で,各プロセッサがメイン・メモリを参照しようとしてぶつかり,参照時間が延びることにより,実効的に各プロセッサの命令実行速度が低下することによっている.原因は単純であるが,これを減らすことはもっとも難しく,マルチプロセッサでの性能低下の大きな部分を占める.この点についてはメイン・メモリのインタリーブ効果との関係で古くから研究されているが,現実のマルチプロセッサでのデータをもとにした研究は今後に残されている.この部分はメインメモリに使われる素子の性能が直接効くところなので,素子の高速化への強い要求がここから出るといってよい.

(2) リソースの競合による性能低下[8)]

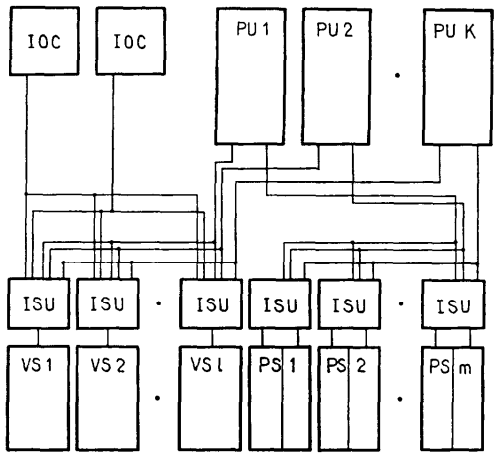
ソフトウェアの視点でのリソースの競合による性能低下で,たとえば,シリアルリユーザブルプログラムの実行のために生じる待ち,入出力装置での競合などその種類も多い.これらの競合はユニプロセッサでも存在するが,マルチプロセッサではこれらのリソースを要求する度合がはるかに大きくなるので,きめの細かい設計が必要になる.この分野の研究はこれからという段階である.

(3) システム性能[9),10)]

この段階ではシステム全体での性能が問題になる.

(1)(2)要因はもちろん含まれるが,これ以外にジョブやタスクの制御方法,メイン・メモリや入出力装置の使い方など多くの問題が含まれてくる.これらの問題のマクロな机上解析はいろいろなされているが,実システムでの解析が今後必要である.

さて,メイン・メモリともっとも関係するのは(1)の項であるが,ここではIBMのMitchellが最近報告した内容について説明してみよう.



IOC : I/O Controller
PU : Processor Unit
VS : Variable Store
PS : Program Store

図-10

解析したマルチプロセッサシステムは**図-10**（前頁参照）の構成からなる．すなわち，k 台の PU (Processor Unit) と l 台の VS (Variable Store) と m 台の PS (Program Store), それにいくつかの IOC (I/O Controller) からなる．これから述べる結果には IOC の影響は考えられていない．このシステムの特徴はメイン・メモリに相当する部分が読み出し専用の PS と通常の VS に分かれている点である．PS は，128 kB からなり，8 バイト語を 8 kW 含む 64 kB の 2 つのモジュールがインタリーブされている．メイン・メモリのサイクルタイムは 500 ns で，2 つのモジュールが 2 つのポートから独立に動くので最大 32 MB/s の処理ができる．これに対して VS は 128 kB が 8 バイト語を 16 kW を含む 1 つのモジュールから構成されていてインタリーブされていない．したがって 1 VS 当りの処理能力は 16 MB/s になる．

一方プロセッサに相当する PU はかなり高度な先行制限を行ない，2 つのメイン・メモリの参照を同時に行なうことができる．したがって，命令実行能力は単純にはいえないが，代表的に次の 4 つの命令ミクスを用いて種々の評価を行なっている．

命令ミクス	ユニプロセッサの処理能力 (MIPS)	メインメモリ参照比率
1	0.50	0.34
2	1.76	1.24
3	2.41	1.98
4	4.72	3.65

MIPS は Million Instruction Per Second の略で大きいほど処理能力が高い．メイン・メモリ参照比率は，

$$\frac{\text{実行命令数}}{\text{8バイト語の参照回数}}$$

で，大きいほど 1 回の参照でより多くの命令が実行されることになる．処理能力と参照比率は密接な関係があることがわかる．

Mitchell は図-10 に示したシステム構成で命令ミクス 1, 2, 3, 4 を基本として処理を実行した時，どのくらいの性能がでるかを求める解析的なシミュレータを作成した．これを実際のシステムでの測定結果と比較したところ，**図-11** のようにきわめてよく一致することをまず確めた．この図は PS が 1 台の場合，プロセッサ台数を変えた時，システム全体の MIPS 値がどう変化するかを示したものである．すべての要求が 1 つの PS に集中するので処理能力が急速に飽和することに示している．

これに対して，PU の台数を増やすと同時に PS の

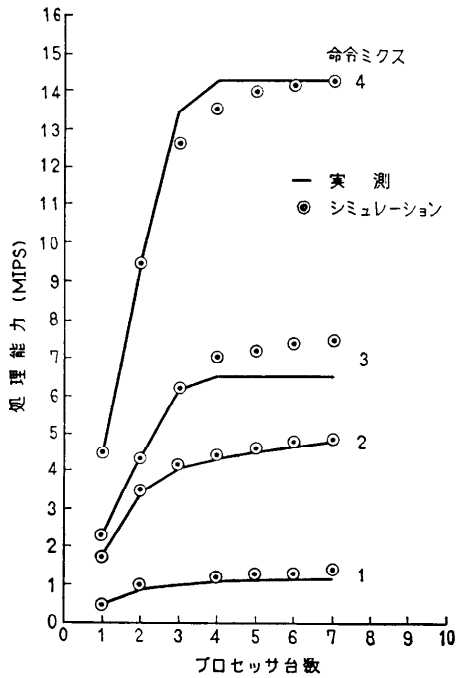


図-11　PS が一台のときの処理能力

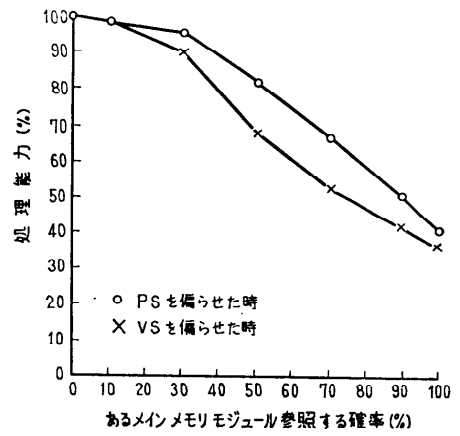


図-12　メイン・メモリモジュール参照の偏りとシステムの処理能力

台数も 1 対 1 に増やした場合にはそれほどの性能低下は起きないことも明らかになっている．しかし，これはプロセッサから発生する参照アドレスが完全にランダムに分布している場合で，あるメイン・メモリモジュールに集中すると図-11 の結果に近くなる．**図-12** は参照アドレスを偏らせたとき，どのようにシステム性能が変化するかを求めたものである．7 台の PU, 9 台の PS, 10 台の VS の構成の場合で，処理能力

は1台1台が独立に動作した時との比較で示してある．命令ミクス3の場合である．システムの性能が参照アドレスの分布に大きく依存することがわかる．

以上のようにマルチプロセッサでのメイン・メモリの競合はシステム性能にきわめて大きな影響があり，今後の中心的課題といえる．

4. むすび

以上，マルチプロセッサのメイン・メモリの方式をいろいろと述べてきたが，要はメイン・メモリが複数の要求元にサービスしなければならないということである．このとき，まず問題になってくるのは要求元からの必要サービス量をメイン・メモリが供給できるかということである．マルチプロセッサになるほど必要サービス量は大きくなるので，より高速のメモリ素子が必要になり，これで不十分ならインタリーブ方式がとられることになる．今後さらにマルチプロセッサ化が進行するとすると，その必要サービス量をメモリ素子の高速化によってのみ補うことは難しいと考えられる．したがってメイン・メモリを取り巻く方式に種々の工夫をこらしてこの要求を満たすことになろう．

参考文献

1) 元岡：コンピュータ・コンプレクスの展望，情報処理，Vol. 15, No. 7, pp. 525～533 (1974).
2) 飯塚ほか：新しい電子計算機システムの調査と評価，電総研調査報告，第179号 (1974).
3) W. A. Wulf and C. G. Bell : C. mmp A multi-mini-processor, Proc. of '72 FJCC. Vol. 41, pp. 765～777 (1972).
4) 飯塚，古谷：C. mmp マルチ-ミニ・プロセッサ，情報処理，Vol. 15, No. 7, pp. 550～556 (1974).
5) W. J. Bouknight et al. : The Illiac VI system, proc. IEEE, Vol. 60, No. 4, pp. 369～388 (1972).
6) ILLIAC IV Systems Characteristics and Programming Manual, Burroghs Corporation (1969).
7) C. E. Skinner and J. R. Asher : Effects of storage contention on system performance, IBM Sys. J. Vol. 8, No. 4, pp. 319～333(1969).
8) W. F. King III, S. E. Smith & I. Wladawsky : Effects of serial programs in multiprocessing systems, IBM J. Res. & Dev. Vol. 18, No. 4, pp. 303～309 (1974).
9) J. L. Baer : A Survey of some theoretical aspects of multiprocessing, Computing Surveys, Vol. 5, No. 1, pp. 31～80 (1973).
10) J. Mitchell et al. : Multiprocessor performance analysis, Proc. of '74 NCC Vol. 43, pp. 399～403 (1974).

(昭和49年12月20日受付)