MIMD 型汎用複合計算機の一方式について (シミュレーションによる評価)†

宮脇 冨士夫†† 佐藤 邦弘†† 福井 和久海††

計算機の処理速度を向上させることを目的として，MIMD 型汎用複合計算機の1モデルを提案する．本モデルの特徴は，(1) パケット方式のバスを採用し，その調停は分散方式によっている，(2) 共有メモリのスループットを高めるために，インタリーブ方式を採用するとともに，多重先入先出バッファを導入している，(3) プロセッサ間の通信のためにハードウェアによる2つの機構を導入している，ことにある．そして，モデル上で，シミュレーションによって，いくつかの特徴あるプログラムを実行した結果，最高121台のプロセッサが同時に動作して，1台のプロセッサに比較して，処理時間を 1.3～3.2% に短縮できることが明らかになった．また，通信機構についても，ソフトウェアによるのと比較して，顕著な効果があることが明らかになった．これらの結果から本モデルは MIMD 型汎用複合計算機のモデルとして有効であろうと判断した．

1. まえがき

現在，MIMD 型汎用複合計算機として決定的な評価を得ているといったものはなく，実現されたものとしては並列台数が 10 程度のものが精々で[1]，また，それ以上は効果がないという報告さえある[2]．我々は先の論文 3) で，MIMD 型汎用複合計算機の1モデルとして，シングル・バス密結合型複合計算機を提案し，理論的にその有効性を明らかにしたが，本論文では，より具体的にモデルの構造を示すとともに，シミュレーションによって，モデル上でいくつかのプログラムを実行した結果を報告する．

2. モデルの構造とシミュレーションの方法

2.1 モデルの全体的構造

モデルの全体的概念図を図1に示す．n 台のプロセッサ (以後 PU と呼ぶ)，アドレス・データ・バス (以後 AD バスと呼ぶ)，コモン・メモリ，データ・バスおよび通信機構 I，II からなる．各 PU はコモン・メモリ上のプログラムに従って独立に動作する．PU がコモン・メモリにアクセスする場合は図2に示す情報のパケットを AD バスのノードへ送り出す．パケットはノードからノードへ転送されてコモン・メモリに至る．このノード間の転送に要する時間を1マイナサイクルと呼ぶ．コモン・メモリではパケットの OP 部に従って処理が行われる．そして，メモリのアクセス時間の後，OP 部が READ もしくは SWAP であれば，図3に示すパケットがデータ・バスに出て来る．各 PU は，PU 番号によって該当するデータを識別し，データ部をとりこむ．

ある PU が他の PU に処理を依頼する場合は，通信機構 I に向けて処理すべきルーチンの先頭番地を送り出す．通信機構 I はそれを待状態にある PU に伝達し，その PU が動作に入る．こうして，各 PU は独立に動作しながらプログラムに従ってまとまった処理を実行する．なお，通信機構 II は資源の占有状態を PU に知らせるもので，2.6 節で説明する．

2.2 AD バスの構造

図1のノードの部分を拡大したのが図4である．1マイナサイクルごとに，次のルールに従って情報が転送される．

(1) PU の情報はまず W ノードに入る．

(2) 上流からの情報は B_1 に入る．

(3) 上流からの情報が流入すると，バッファ・ノードの情報は順に押し上げられる．

(4) ゲートの優先順序は，第1位から S_m，S_w，そして S_1～S_{m-1} のうち，情報のあるバッファの最上位に対応するゲートの順である．

(5) PU が W に情報を出そうとして，なお W に情報が留っている時は PU の動作をフリーズする．そして，W の情報が流れ去ると，同時に待機中の情報が W に移り，フリーズは解除される．

こうすれば，バッファの深さ m を適切に設計することによって，PU に所望のアクセスを可能にすることができる[3]．

† A MIMD Type General Purpose Multi-proccessor System (Estimated by Simulation) by FUJIO MIYAWAKI, KUNIHIRO SATO and WAKUMI FUKUI (Department of Electrical Engineering, Himeji Institute of Technology).

†† 姫路工業大学電気工学教室

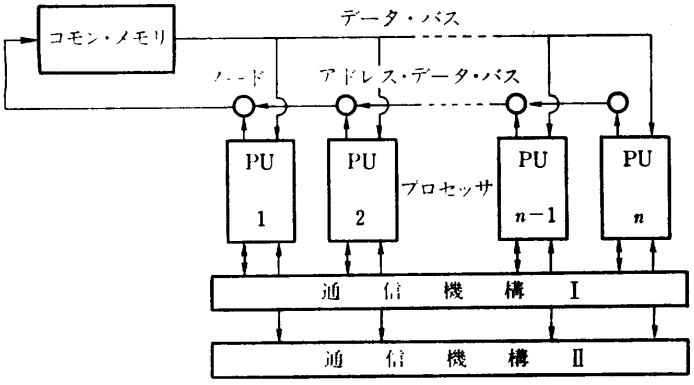
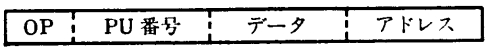


図 1 モデルの構造
Fig. 1 Structure of the model.



OP : メモリ上での処理
READ 読出し
WRITE 書込み
SWAP 読出書込み
PU 番号: 情報を送り出したプロセッサの番号
データ : WRITE, SWAP の書込みデータ
アドレス: アクセスすべきメモリ上の番地

図 2 アドレス・データ・バスの情報パケット
Fig. 2 Information packet on address data bus.

PU 番号: データの送り先プロセッサ番号
データ : メモリから読み出されたデータ

図 3 データ・バスの情報パケット
Fig. 3 Information packet on data bus.

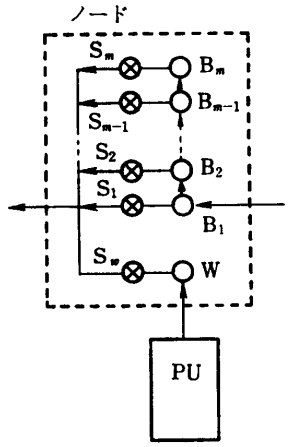
2.3 コモン・メモリの構造

コモン・メモリの構造を**図 5** に示す．AD バスで転送されて来た情報は入口ノードに至る．通常，次のマイナサイクルでバッファにスタックされるが，バッファが一杯の場合は AD バスの流れをフリーズし，バッファに空きができた次のマイナサイクルで解除する[3]．バッファは多重先入先出バッファで[3]，スイッチはスタックされている情報のうち，次の条件を満たすものを先入先出のルールでメモリ・バンクに転送する．

(1) アドレスが非動作中のバンクに当たっている．

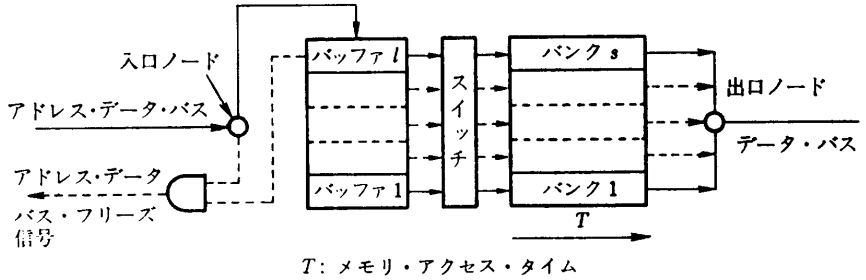
(2) 先に同一 PU 番号の情報がスタックされていない．

各メモリ・バンクはインタリーブ方式である．1 つのメモリ・バンクの構造を**図 6** に示す．バンクが動作に入るとフラグが 1 となり，カウンタが動作を開始する．OP 部が READ もしくは SWAP であれば，カウンタがアクセス・タイムを経過すると，指示アドレスの内容と PU 番号を出口ノードに送り出す．READ の場合はフラグが 0 となり，SWAP の場合は OP 部が WRITE に変わって，データの書き込み動作に入り，アクセス・タイムの後にフラグが 0 となる．



B_1〜B_m: バッファ・ノード
W : ウエイト・ノード
m : バッファの深さ
S_1〜$S_m S_w$: ゲート

図 4 ノードの構造
Fig. 4 Structure of a node.



T: メモリ・アクセス・タイム

図 5 コモン・メモリの構造
Fig. 5 Structure of common memory.

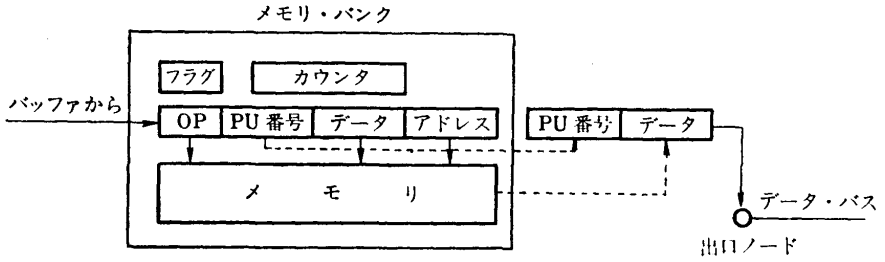
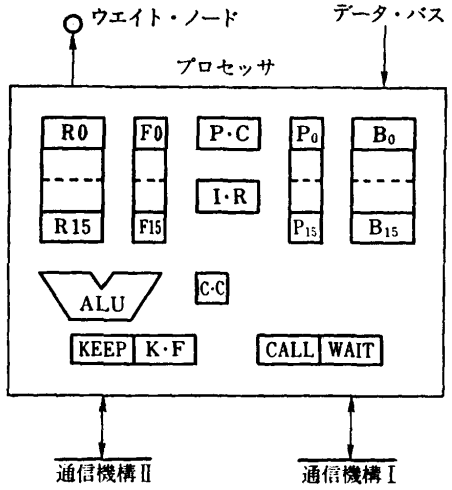


図 6 メモリ・バンクの構造
Fig. 6 Structure of a memory bank.



P·C : プログラム・カウンタ
I·R : 命令レジスタ
ALU : 算術論理演算ユニット
C·C : コンディションコード・レジスタ
$R_{0\sim15}$: 汎用レジスタ
$F_{0\sim15}$: レジスタ有効フラグ—汎用レジスタの内容が有効か否かを示す.
$B_{0\sim15}$: バッファ—データ・バスからデータをとりこむ FIFO バッファ.
$P_{0\sim15}$: ポインタ・レジスター—バッファのデータをどのレジスタに転送するかを指示する FIFO レジスタ.
CALL : コール・レジスター0でなければ, コール要求が出ていることを示す.
WAIT : ウェイト・レジスター—1 であれば待状態にあることを示す.
KEEP : キープ・レジスター通信機構 II との交信情報を保持する.
K·F : キープ・フラグ—KEEP 命令が実行されると −1, FREE 命令が実行されると 1, その他の場合は 0 を示す.

図 7 プロセッサの構造
Fig. 7 Structure of a processor.

2.4 プロセッサの構造

プロセッサは通常のマイクロプロセッサの構造を想定しているが, 本モデル特有の部分もあるので, 主要部分を図 7 に示す. レジスタ有効フラグ ($F_{0\sim15}$) はロード命令等によってデータをレジスタに移す場合に有効なデータが入ったか否かを示す. すなわち, 無効なデータを使って処理が進むことを防ぐためである. バッファ ($B_{0\sim15}$) はデータ・バスと PU の速度を整合させるためのものである. PU の処理時間の単位は区別してマシンサイクルと呼ぶ. 通常, マシンサイクルはマイナサイクルよりも大きい. ポインタ・レジスタ ($P_{0\sim15}$) はデータ・バスから得たデータをどのレジスタに格納するかを記憶しておくレジスタである. コール・レジスタ (CALL) は他の PU に依頼する処理ルーチンの先頭番地を通信機構 I に知らせるためのものである. ウエイト・レジスタ (WAIT) は PU が待状態にあることを通信機構 I に知らせるためのものである. キープ・フラグ (K·F) は資源に対する要求を通信機構 II に知らせるためのもので, −1 ならば占有要求, 1 ならば解放要求, 0 ならば要求なしを示す. キープ・レジスタ (KEEP) は K·F に関連して資源番号を通信機構 II に知らせるためのものである.

2.5 通信機構 I の構造

通信機構 I は PU 間の通信を行うもので, 各 PU のコール・レジスタの情報を受けてウエイト・レジスタに渡す役割を果たす. 本モデルでは, 各 PU からコモン・メモリにいたる距離が異なるため, コモン・メモリに近い若い番号の PU ほど性能がよい. したがって若い番号の PU を優先的に稼動させる. すなわち, コールを出している最も若い番号の PU から待状態にある最も若い番号の PU に情報を受け渡す.

通信機構 I は次のように動作する. PU のコール・レジスタは通常 0 であるが, CALL 命令(3.1 節参照)を実行すると, コール・レジスタに CALL 命令に付随した開始番地が格納される. 通信機構 I はコール・レジスタが 0 でないことを検出すると, その情報を受け取ってウエイト・レジスタが −1 のレジスタに格納

し，それを受けた PU が開始番地から処理に入る．

2.6 通信機構Ⅱの構造

PU が他の PU とコモン・メモリを通して通信する場合，必要な情報の書き込みが完了するまで他の PU にアクセスされては困る場合がある．このような場合，通信機構Ⅱが他の PU に特定資源の占有を知らせる働きをする．通信機構Ⅱは次のように動作する．

PU が占有命令を実行すると，命令に付随した資源番号がキープ・レジスタに格納され，K·F レジスタに −1 が格納される．通信機構Ⅱがそれを検出すると，その資源が空いていれば K·F レジスタに 0 をかえす．もし，他の PU がすでにその資源を占有していれば解放されるまで待たねばならない．解放命令を実行すると，K·F レジスタに 1 が格納される．それを検出した通信機構Ⅱはキープ・レジスタから資源番号を知り，その資源を解放するとともに K·F レジスタに 0 をかえす．通信機構ⅡもⅠと同じく若い番号の PU を優先する．資源の占有機能は複合計算機にとって不可欠である．

2.7 シミュレーションの方法

シミュレーションの方法は 1 マイナサイクルごとに各レジスタの変化を追うレジスタ・レベルのシミュレーションが主であるが，PU の動作については各機械語に所要のマシンサイクル数を決めて，その時間が経過すると論理的な値および状態をもどす機能レベルのシミュレーションを行った．

表 1 時間定数
Table 1 Time constant.

項　目	マイナサイクル	実時間 (ns)
AD バスのクロック	1	63
メモリ・アクセス・タイム	4	250
通信機構 Ⅰ，Ⅱ	2	100*
プロセッサのマシンサイクル	5	300

* プロセッサ 200 台を接続した場合

各部の時間定数は設計から試算して表 1 のように設定した．1 マイナサイクルすなわち AD バスのクロックは図 5 のスイッチ部の処理時間によって抑えられる．通信機構Ⅰの時間とは CALL 要求をしている一番若い番号の PU から WAIT 状態にある一番若い番号の PU へ情報を転送する時間である．また，通信機構Ⅱの時間とは資源の占有状態を PU に知らせる時間である．

3. 評価に用いるプログラム

モデルを評価するために用いる主な 2 つのプログラムについて述べる．

3.1 機械語

機械語はすべて 32 ビット幅とし，表 2 に示すも

表 2 シミュレーションに使用した命令セット
Table 2 Machine instruction set used in simulation.

命　令	記　号	機械語	マシン・サイクル*1	機　能
ロード	L	02, R1, R2, D	5	(R1)←((R2)+D)*2,*3
ロード (R-R)	LR	06, R1, R2	4	(R1)←(R2)
ロード・アドレス	LA	08, R1, R2, D	5	(R1)←(R2)+D，R2=0 の時 (R1)←D
記　憶	ST	05, R1, R2, D	6	((R2)+D)←(R1)
加　算 (R-R)	AR	07, R1, R2	4	(R1)←(R1)+(R2)
加　算	A	0D, R1, R2, D	6	(R1)←(R1)+((R2)+D)，R2=0 の時 (R1)←(R1)+D
減　算 (R-R)	SR	11, R1, R2	4	(R1)←(R1)−(R2)
減　算	S	03, R1, R2, D	6	(R1)←(R1)−((R2)+D)，R2=0 の時 (R1)←(R1)−D
条件分岐	CB	04 S, D	T 6 F 5	T：C·C(コンディション・コード)=S の時 (PC)←D F：C·C≠S の時 (PC)←(PC)+1
無条件分岐	UB	12, D	4	(PC)←D
呼出し	CALL	09, D	5	コール・レジスタにDを格納する．
待　ち	WAIT	01	5	待ち状態に入る．
アイドル	IDLE	10, D	3+D	Dマシン・サイクルの間アイドル状態になる．
占　有	KEEP	0A, D	5	通信機構Ⅱに資源Dの占有を要求する．
解　放	FREE	0B	4	通信機構Ⅱに資源を解放する．
スワップ	SW	13, R1, R2, D	6	(R1)↔((R2)+D) データの交換
停　止	STOP	0C		シミュレーションを終る．

*1 フェッチサイクルを含む，*2 (X)：X番地もしくはレジスタXの内容，*3 矢印：データの転送を示す．

表 3 プログラム (1)
Table 3 Program (1).

番地	機械語	アセンブリ記号		処理説明
1	000000C8	EC	200	終り確認カウンタ
2	000000C8	SC	200	始り確認カウンタ
3	000000C8	N	200	ベクトルの要素数
4	000001D6	SA	470	結果格納先頭番地
5	00000032	SA1	50	被演算数1の先頭番地
6	00000104	SA2	260	被演算数2の先頭番地
7	08700000	START	LA, 7, 0, 0	(R7)←0
8	02570002		L, 5, 7, 2	(R5)←(SC)
9	03500001		S, 5, 0, 1	(R5)←(R5)−1
A	0420000D		CB, 2, JMP	もし (R5)=0 ならば JMP へとぶ.
B	05570002		ST, 5, 7, 2	(SC)←(R5)
C	09000007		CALL 7	他のプロセッサに実行指令を出す. 開始番地は7.
D	02170004	JMP	L, 1, 7, 4	(R1)←(SA)
E	02270005		L, 2, 7, 5	(R2)←(SA1)
F	02370006		L, 3, 7, 6	(R3)←(SA2)
10	02470003		L, 4, 7, 3	(R4)←(N)
11	11450000		SR, 4, 5	(R4)←(R4)−(R5)
12	03400001		S, 4, 0, 1	(R4)←(R4)−1
13	07140000		AR, 1, 4	(R1)←(R1)+(R4)
14	07240000		AR, 2, 4	(R2)←(R2)+(R4)
15	07340000		AR, 3, 4	(R3)←(R3)+(R4)
16	02820000		L, 8, 2, 0	(R8)←((R2))
17	0D830000		A, 8, 3, 0	(R8)←(R8)+((R3))
18	05810000		ST, 8, 1, 0	((R1))←(R8)
19	1000 T_i		IDLE T_i	T_i マシンサイクルの間アイドル状態になる.
1A	0A000001		KEEP 1	メモリ1番地を占有する.
1B	02570001		L, 5, 7, 1	(R5)←(EC)
1C	03500001		S, 5, 0, 1	(R5)←(R5)−1
1D	04200021		CB, 2, END	もし (R5)=0 ならば END へとぶ.
1E	05570001		ST, 5, 7, 1	(EC)←(R5)
1F	0B000000		FREE	メモリ1番地の占有を解く.
20	01000000		WAIT	待状態に入る.
21	0C000000	END	STOP	シミュレーションを終る.

のを想定した. 通常の機械語のほかに CALL, WAIT, KEEP, FREE がつけ加えられている. また, それぞれの使い方は表 3 のプログラムの中で明らかになる. IDLE と STOP はシミュレーションのための便宜的なものである.

3.2 並列条件の良いプログラム (1)

プログラム(1)はベクトルとベクトルの加算を行うもので, 要素ごとに独立に処理を行う. 表3がそのプログラムである. 7番地から処理を始め, 自己の担当する要素の番号を得て, それから1を減じ, C番地の CALL 命令で, 他の PU に7番地から処理を始めるように, 通信機構Iに依頼する. こうして次々と PU が並列処理に入っていく. D〜18 番地の命令によって自己の担当する要素の加算を行う. 19 番地の IDLE 命令を実行すると PU は T_i マシンサイクルの間アイドル状態に入って何の動作もしない. これは, 演算子の処理時間が異なる場合を評価するためである. 1A〜1F 番地は最後の処理か否かを 1 番地の情報によって調べているところである. この場合, 1番地を占有しておかないと誤った情報が書き込まれる可能性がある. もし最後でなければ1番地の情報から1減じて占有を解き, 待状態に入る.

このプログラムでは T_i が長くなるほどメモリにアクセスする間隔が長くなり, アクセスの衝突が緩和されるので並列条件が良いプログラムである.

表 4 プログラム (2)
Table 4 Program (2).

番地	機械語	アセンブリ記号		処理説明
1	000000C8	EC	200	終り確認カウンタ
2	000000C8	SC	200	始り確認カウンタ
3	000000C8	N	200	ベクトルの要素数
4	00000030	SA	48	結果格納開始番地
5	08700000	START	LA, 7, 0, 0	(R7)←0
6	02570002		L, 5, 7, 2	(R5)←(SC)
7	03500001		S, 5, 0, 1	(R5)←(R5)−1
8	0420000B		CB, 2, JMP	もし (R5)=0 ならば JMP へとぶ.
9	05570002		ST, 5, 7, 2	(SC)←(R5)
A	09000005		CALL 5	他のプロセッサに実行指令を出す. 開始番地は5.
B	02170004	JMP	L, 1, 7, 4	(R1)←(SA)
C	02470003		L, 4, 7, 3	(R4)←(N)
D	11450000		SR, 4, 5	(R4)←(R4)−(R5)
E	03400001		S, 4, 0, 1	(R4)←(R4)−1
F	07140000		AR, 1, 4	(R1)←(R1)+(R4)
10	0820 T_r		LA, 2, 0, T_r	(R2)←T_r 繰り返し数をセットする.
11	02310000	LOOP	L, 3, 1, 0	(R3)←((R1))
12	0D300001		A, 3, 0, 1	(R3)←(R3)+1
13	05310000		ST, 3, 1, 0	((R1))←(R3)
14	03200001		S, 2, 0, 1	(R2)←(R2)−1
15	04200017		CB, 2, OUT	もし (R2)=0 ならば OUT へとぶ.
16	12000011		UB LOOP	LOOP へとぶ.
17	0A000001	OUT	KEEP 1	メモリ1番地を占有する.
18	02570001		L, 5, 7, 1	(R5)←(EC)
19	03500001		S, 5, 0, 1	(R5)←(R5)−1
1A	0420001E		CB, 2, END	もし (R5)=0 ならば END にとぶ.
1B	05570001		ST, 5, 7, 1	(EC)→(R5)
1C	0B000000		FREE	メモリ1番地の占有を解く.
1D	01000000		WAIT	待状態に入る.
1E	0C000000	END	STOP	シミュレーションを終る.

3.3 並列条件の悪いプログラム (2)

プログラム(2)はベクトルの各要素に何回か1を加えるプログラムである (表 4). ただし, 表4の 11〜16 番地をみればわかるように, 1を加える都度, メモリとの間でロードと記憶を行うようにプログラムされている. このプログラムでは1を加える回数 T_r が大きくなるに従って1つの要素を処理する時間が長くなり, 多くのPUが並列処理に入るが, メモリへのアクセスも多くなり並列条件の悪いプログラムである.

4. シミュレーションによる評価 (1)

4.1 プログラム (1) による評価

演算子の処理時間 T_i をパラメータとしてプログラム(1)を実行した結果を図 8 の実線に示す. 演算子の処理時間は AMD 社の演算ユニット Am 9511 の仕様 (表 5) を参考にした. 1台の PU による処理時間は演算子の処理時間に従って直線的に増加する. 処理時間の短縮は 121 台の PU が同時に稼動し, 1台の PU による処理時間に比較して 1.3% にまでなっている. このプログラムは演算子の処理時間に関係なく, 機械語のステップ数は一定であるから, 処理時間が長くなるほど, 単位時間当りのメモリ・アクセス確率が小さくなり, バスの競合は小さくなる. したがって, バスの飽和は起きない. シミュレーションの結果からも AD バスおよび各 PU のフリーズは起きていないことを確認した.

4.2 プログラム (2) による評価

プログラム(2)の T_r をパラメータとして実行した結果を図 9 に示す. AD バス1本とラベルを付けた線である. 処理時間は T_r が 50 までは一様に短縮され 5.25% に至るが, それ以上は少し短縮の度が悪くなる傾向にある. そこで, 各 PU の担当した要素数

表 5 演算子の処理時間*1
Table 5 Processing time of operators.

演算子*2	マシン・サイクル
加 算	54～ 368
減 算	70～ 370
乗 算	146～ 168
除 算	154～ 184
平方根	782～ 870
指 数	3794～ 4878
正 弦	3796～ 4808
余 弦	3840～ 4878
自然対数	4298～ 6956
常用対数	4474～ 7132
正 接	4894～ 5886
逆正接	4992～ 6536
逆正弦	6230～ 7938
逆余弦	6304～ 8284
累 乗	8290～12032

*1 AMD 社の演算ユニット Am 9511 の説明書を参考にした.
*2 32 ビット実数型データ

を調べた結果が図 10 である. T_r が 50 を過ぎるところで, 若い番号の PU が担当する要素の数が少なくなっているのが目につく. これは下流の PU のアクセスが上流の PU のアクセスによって抑えられているためである.

また, メモリ・バンクの並列動作数およびメモリ・バッファの深さの頻度分布を調べた結果が図 11 である. T_r が 50 以上になると状態の変化がみられず, メモリのスループットが飽和して, バッファが一杯になっていることが予想される. 破線はメモリ・アクセス要求の確率, 最大同時動作 PU 台数, バッファの深さ, バンク数, アクセス・タイムから先の論文 3) に従って計算した理論値である. バンクの並列動作数の分布はよく合致しているが, バッファの深さについては全く合っていない. しかし, T_r が 50 以上では変化がみられないところは理論値と同じである.

5. シミュレーションによる評価 (2)

前述の結果を基準にして他のパラメータを変化させた場合の効果について述べる.

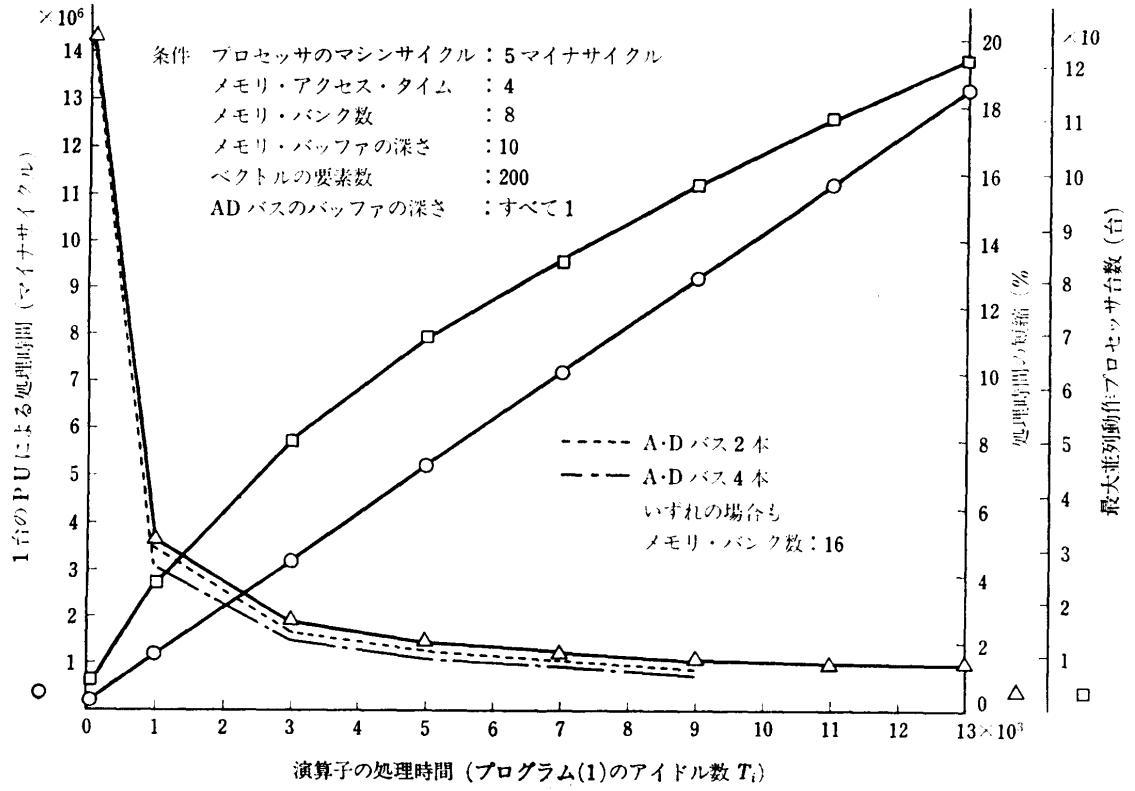


図 8 プログラム (1) による性能評価
Fig. 8 Performance estimation on program (1).

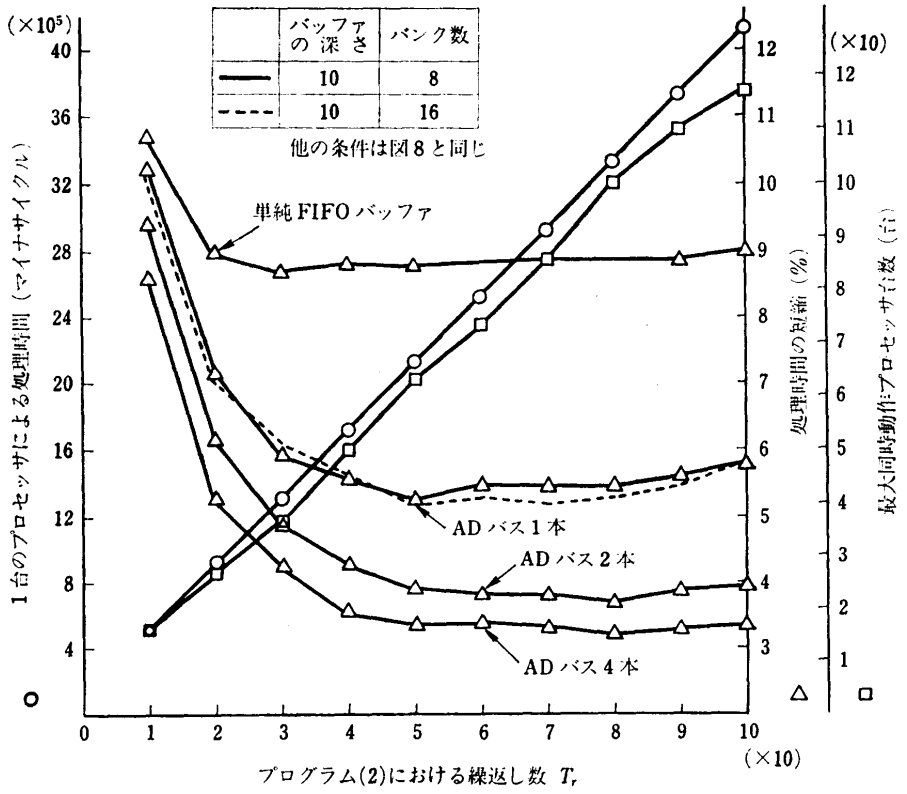


図 9 プログラム（2）による性能評価
Fig. 9 Performance estimation on program (2).

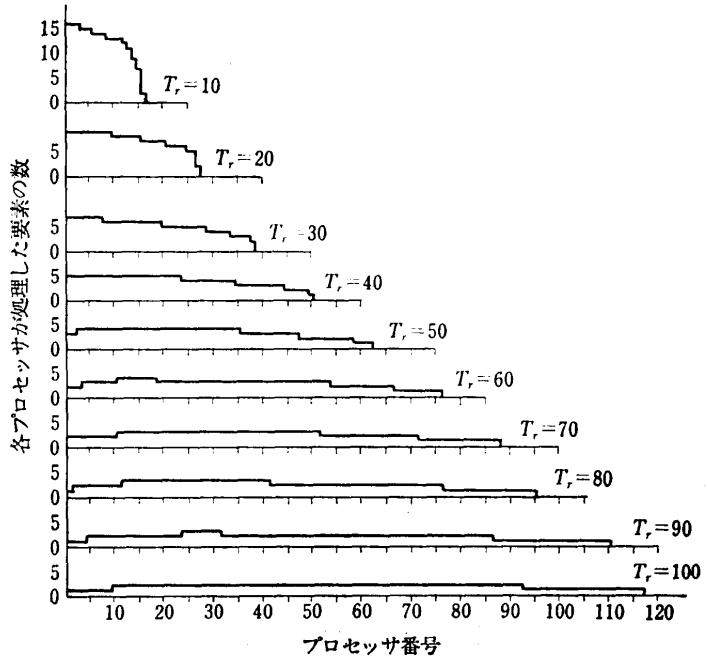
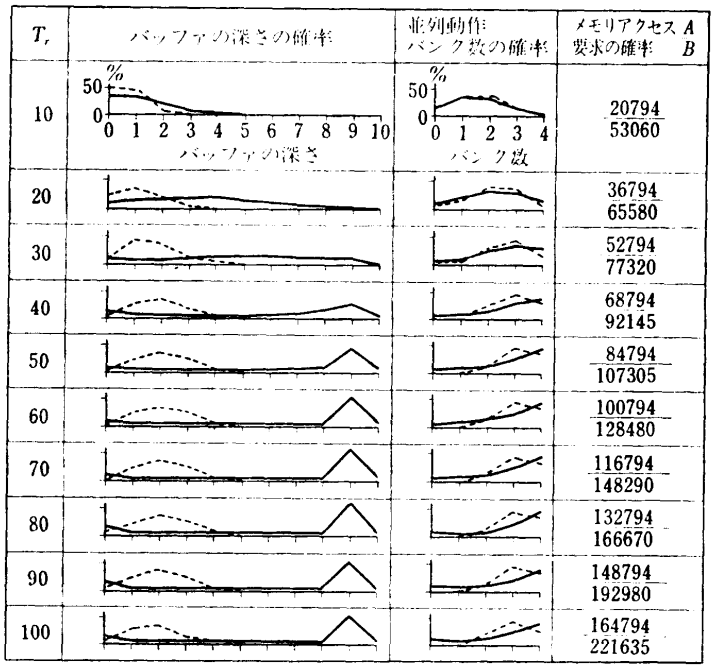


図 10 各プロセッサの処理したベクトル要素の数（プログラム（2））
Fig. 10 Number of elements processed by each processor.



A：メモリ・アクセスの回数，B：AD バスの動作した時間（マイナサイクル），実線：シミュレーションによる値，破線：理論値

図 11 メモリ・バンクの並列動作数とバッファの深さの確率分布
Fig. 11 Probability of the number of busy memory banks and the depth of buffer.

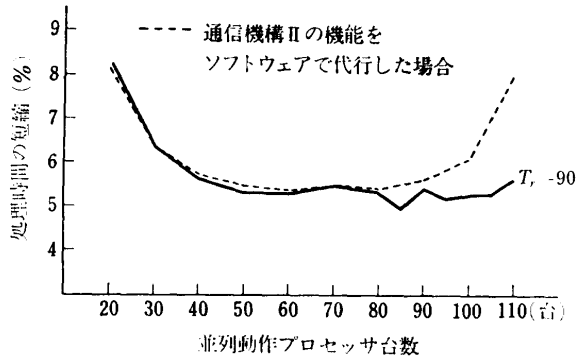


図 12 並列動作プロセッサ台数の効果
Fig. 12 Effect of the number of processors.

5.1 プロセッサ台数の効果

プログラム(2)の T_r を 90 として，PU 台数を変化させた結果を図 12 の実線に示す．この例では PU が 60 台までは処理時間の短縮がみられるが，それ以上は並列動作 PU 台数が増加しても効果はない．

5.2 AD バスのバッファの効果

前節と同じく T_r を 90 として，AD バスのバッファの深さを変えた結果を図 13 に示す．バッファの深さは論文 3) に基づくものである．バッファの深さがすべて1の(a)から(b)～(d)とバッファの深さを調整することによって負荷が若い番号の PU に移っている．しかし，処理時間の短縮に及ぼす効果は 2% と小さい．

5.3 バンク数の効果

バンク数を 16 とした結果を図9の破線に示す．一部では逆の効果がみられる．このパラメータの組合せでは，バンク数が8でバンクの並列動作はその極に達している．

5.4 多重先入先出バッファの効果

本モデルの特徴の1つである多重先入先出バッファの効果をみるために，単純先入先出バッファにした場合の結果を図9に示した．処理時間の短縮からみて 1.6 倍程度の差があらわれている．

5.5 AD バスを増すことによる効果

バスを2本，4本と増した場合の結果を図8，9に示した．プログラム(1)については15%程度の改善であるが，プログラム(2)については2本で 32%，4本で 41% 程度の改善がみられる．すなわち，処理時間の短縮は 3.2% にいたっている．これは，各 PU からコモン・メモリに至る平均距離が短くなることと，並列動作バンク数が2倍，4倍となることによる効果である．しかし，その傾向からみて，AD バスを

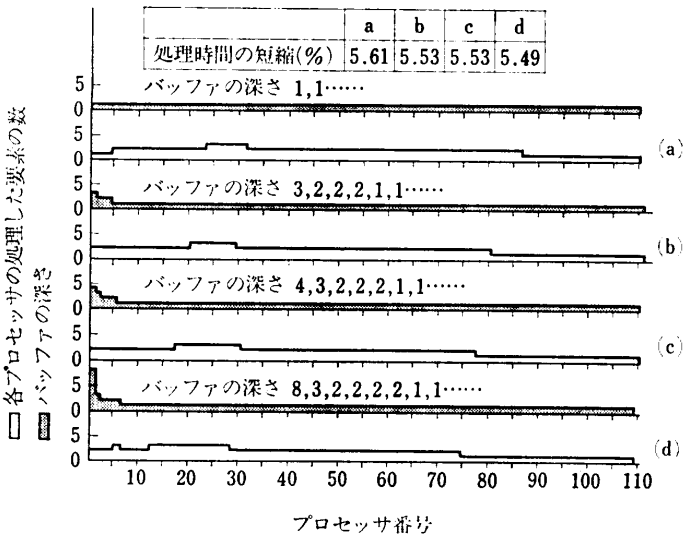


図 13 アドレス・データ・バスのバッファの効果
Fig. 13 Effect of address-data bus buffer.

これ以上増すことによる効果はあまり期待できないであろう.

6. 通信機構Ⅰの必要性とその効果

2.5 節に述べたように，通信機構Ⅰは PU が他の PU に処理を依頼するためのハードウェアであるが，同じ機能をソフトウェアで行うことも可能である．すなわち，コモン・メモリ上に特別の番地を定めておいて，各 PU がそれをみることによって情報を伝達する方法である．**表 6** に示すプログラム(3)がその方法である．プログラム(1)と異なるところは 9～C番地と 23 番地の部分である．待ち状態の PU は 9～C番地の部分によって，絶えずカウンタの番地をみており，このことが有効な処理をしている PU のメモリ・アクセスを妨げ，極端な場合はデッド・ロックが予測される.

プログラム(1)，(3)において，アイドル・タイム T_i を 50 として，PU 台数を変化させた結果を**図 14** に示す．プログラム(1)では PU 台数が6までは処理時間が一様に減少し，それ以上多く接続しても無効であるが，プログラム(3)では PU が増すに従って処理時間がどんどん増加し，1台の処理時間よりも長くなる．このことから，通信機構Ⅰの必要性と効果は明らかである.

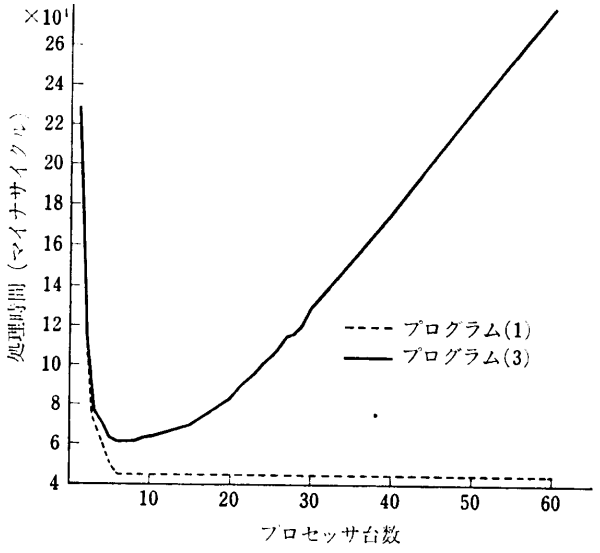


図 14 通信機構Ⅰの効果
Fig. 14 Effect of communication system Ⅰ.

7. 通信機構Ⅱの効果

表 7 に示すプログラム(4)は占有命令で通信機構Ⅱを用いる代りに，同様のことをソフトウェア的に行ったものである．T_r を 50 とし，PU 台数を変化させた結果が図 12 の破線である．PU 台数が 80 以上のところで違いが大きく出ている．しかし，通信機構Ⅰほど大きく違いが出ないのは，前者では待状態にある PU が多くなるほど，無駄なアクセスが多くなるのに対して，後者では PU が目的を達するに従ってアクセスが減少するからである.

8. ま と め

本論文で提案した MIMD 型汎用複合計算機のシミュレーションの結果，主な点は次のとおりである.

(1) 並列度が高くかつメモリ・アクセスの競合が少ない条件の良いプログラムを実行した場合，121 台の PU が同時に働いて，1台に比べて処理時間が 1.3% になった.

(2) 並列度が高くかつメモリ・アクセスの競合が多くなる条件の悪いプログラムを実行した場合，63 台の PU が同時に働いて，処理時間が 5.25% になった.

(3) バスの数を4本に増した結果，条件の良いプログラムでは大きな効果は見られなかったが，条件の悪いプログラムでは処理時間の短縮が 3.2% にまで改善できた.

(4) 多重先入先出バッファは通常の先入先出バッ

表 6 プログラム (3)
Table 6 Program (3).

番地	機械語	アセンブリ記号		処理説明
1	000000C8	EC	200	終り確認カウンタ
2	000000C8	SC	200	始り確認カウンタ
3	000000C8	N	200	ベクトルの要素数
4	000001D6	SA	470	結果格納先頭番地
5	00000032	SA1	50	被演算数1の先頭番地
6	00000104	SA2	260	被演算数2の先頭番地
7	08700000	START	LA, 7, 0, 0	(R7)←0
8	0890FFFF		LA, 9, 0, FFFF	(R9)←'FFFF'
9	13970002	LOOP	SW, 9, 7, 2	(R9)↔(SC)
A	06890000		LR, 8, 9	(R8)←(R9)
B	0380FFFF		S, 8, 0, FFFF	(R8)←(R8)−'FFFF'
C	04200009		CB, 2, LOOP	もし (R8)=0 ならば LOOP へとぶ.
D	06590000		LR, 5, 9	(R5)←(R9)
E	03900001		S, 9, 0, 1	(R9)←(R9)−1
F	04200011		CB, 2, JMP	もし (R9)=0 ならば JMP へとぶ.
10	13970002		SW, 9, 7, 2	(R9)↔(SC)
11	02170004	JMP	L, 1, 7, 4	(R1)←(SA)
12	02270005		L, 2, 7, 5	(R2)←(SA1)
13	02370006		L, 3, 7, 6	(R3)←(SA2)
14	02470003		L, 4, 7, 3	(R4)←(N)
15	11450000		SR, 4, 5	(R4)←(R4)−(R5)
16	07140000		AR, 1, 4	(R1)←(R1)+(R4)
17	07240000		AR, 2, 4	(R2)←(R2)+(R4)
18	07340000		AR, 3, 4	(R3)←(R3)+(R4)
19	02820000		L, 8, 2, 0	(R8)←((R2))
1A	0D830000		A, 8, 3, 0	(R3)←(R8)+((R3))
1B	05810000		ST, 8, 1, 0	((R1))←(R8)
1C	1000 T_i		IDLE T_i	T_i マシンサイクルの間アイドル状態になる.
1D	0A000001		KEEP 1	メモリ1番地を占有する.
1E	02570001		L, 5, 7, 1	(R5)←(EC)
1F	03500001		S, 5, 0, 1	(R5)←(R5)−1
20	04200024		CB, 2, END	もし (R5)=0 ならば END へとぶ.
21	05570001		ST, 5, 7, 1	(EC)←(R5)
22	0B000000		FREE	メモリ1番地の占有を解く.
23	12000009		UB LOOP	LOOP へとぶ.
24	0C000000	END	STOP	シミュレーションを終了する.

表 7 プログラム (4) (1番地〜16 番地はプログラム (2) と同じ)
Table 7 Program (4).

番地	機械語	アセンブリ記号		処理説明
17	0890FFFF	OUT	LA, 9, 0, FFFF	(R9)←'FFFF'
18	13970001	LOOP	SW, 9, 7, 1	(R9)↔(EC)
19	06890000		LR, 8, 9	(R8)←(R9)
1A	0380FFFF		S, 8, 0, FFFF	(R8)←(R8)−'FFFF'
1B	04200018		CB, 2, LOOP	もし (R8)=0 ならば LOOP へとぶ.
1C	03900001		S, 9, 0, 1	(R9)←(R9)−1
1D	04200020		CB, 2, END	もし (R9)=0 ならば END へとぶ.
1E	13970001		SW, 9, 7, 1	(R9)↔(EC)
1F	01000000		WAIT	待状態に入る.
20	0C000000	END	STOP	シミュレーションを終る.

ファに比較して，処理時間の短縮に 1.6 倍の効果を示した．

（5） AD バスのバッファは局所的な調停効果は明らかになったが，処理時間の短縮には効果が見られなかった．

（6） 通信機構Ⅰ，Ⅱについて，その必要性を示すとともに，同じ機能をソフトウェア的に行うのと比較して，その効果が顕著に出ることが明らかになった．

以上のことから，本モデルは 100〜200 台の PU による MIMD 型複合計算機として構造が簡単で効果のあるモデルと判断した．

今後の課題としては，ローカル・メモリを設けた場合の評価を行い，プロト・タイプを実現して実際的な性能評価を行いたい．

本研究を進めるに当たって，土師教弘，福井昭也，平野貴紀の 3 君が手伝ってくれた．ここに改めて謝意を表する．

参 考 文 献

1) Fuller, S. H., Ousterhout, J. K., Raskin, L., Rubinfeld, P. I., Sindhu, P. J. and Swan, R. J.: Multi-Microprocessors: An Overview and Working Example, *Proc. of the IEEE*, Vol. 66, pp. 216-228 (1978).
2) 村岡洋一：並列処理概論（2），情報処理，Vol. 16, No. 2, pp. 137-144 (1975).
3) 宮脇冨士夫，富田眞治，萩原 宏：MIMD 型汎用複合計算機の一方式について（理論的評価），情報処理学会論文誌，Vol. 28, No. 8, pp. 829-838 (1987).

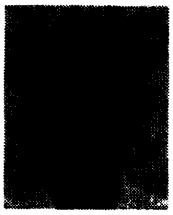
（昭和 60 年 12 月 11 日受付）
（昭和 62 年 5 月 13 日採録）

宮脇冨士夫（正会員）

昭和 10 年生．昭和 37 年姫路工業大学電気工学科卒業．同研究生を経て，昭和 41 年姫路工業大学電気工学教室助手，昭和 53 年同講師，昭和 56 年同助教授，現在に至る．高級言語処理機械，複合計算機，アプリケーション・システムの研究に従事．京都大学工学博士．訳書「ICES 概説」，「ICES システム内部論理構造の解説」（共訳，北尾書籍）．電子情報通信学会，電気学会，日本シミュレーション学会各会員．

佐藤 邦弘

昭和 27 年生．昭和 51 年熊本大学工学部電気工学科卒業．昭和 56 年京都大学大学院博士課程単位取得退学．昭和 58 年 10 月姫路工業大学電気工学科助手．現在に至る．オープン系核融合閉込めの理論的研究に従事．日本物理学会，プラズマ・核融合学会各会員．



福井和久海

大正 13 年生．昭和 22 年東京帝国大学第 2 工学部電気工学科卒業．工学博士．現在姫路工業大学教授．システムシミュレーションに関する研究に従事．照明学会会員．