NAL TR-1154

ISSN 0389-4010
UDC 681.323

航空宇宙技術研究所報告

TECHNICAL REPORT OF NATIONAL AEROSPACE LABORATORY

TR-1154

並列計算機の検証模擬

原田公一・吉田正廣・中村絹代

1992年5月

航空宇宙技術研究所
NATIONAL AEROSPACE LABORATORY

並列計算機の検証模擬*

原田公一*1　吉田正廣*1　中村絹代*1

Simulation of a Parallel Computer*

Koichi HARADA*1, Masahiro YOSIDA*1
and Kinuyo NAKAMURA*1

ABSTRACT

Using the National Aerospace Laboratory (NAL) numerical simulater, a software program simulating a parallel computer configurated with 128 x 256 processing elements was run following the implementation of its associated assembly language .

The software simulates a parallel computer's memory access and register operation at the clock level and generates an operational time diagram along with results, whereas the assembly language describe its execution procedures.

The operation of this computer was evaluated by employing a basic function subroutine, with the instruction set and network between the memory and the processing elements, being subsequently examined, thereby enabling estimation of its maximum efficiency.

In addition, efficiency was further evaluated by comparing the resultant parallel processing time using a two dimensional computational fluid dynamics program to the corresponding sequential processing time obtained with a processing element.

Keywords: Parallel computer, parallel processing, numerical simulator

概　　要

当研究所の数値シミュレータのもとで動作する128×256個の演算器を持つ並列計算機の模擬プログラムと，そのアセンブラを作成した。前者は記憶装置やレジスタの動作をクロックのレベルで模擬しタイムチャートおよび演算結果を出力するものであり，後者は並列処理の手続きを記述するためのものである。

本論文では基本的なサブルーチンを使い並列計算機の評価例について述べ，命令セットと記憶装置・演算器間のネットワークについて検討しその結果を基に最大性能についての推定をおこなう。

最後に，二次元の流れ場解析プログラムを使って一台の演算器による処理時間と並列処理によるそれとを比べ，本並列計算機の性能評価をまとめる。

* 平成3年11月29日受付（received 29 November 1991）
*1数理解析部（Computational Sciences Division）

1. はじめに

近年，物理現象をスーパーコンピュータを用いて解明する数値シミュレーション技術の進展は目覚ましく，以前では計算量が膨大なため解明が困難と考えられていた分野でも実用の段階に移行しつつある。数値シミュレーション技術は計算機の処理性能と密接に関連しており，より複雑な現象を精度よく迅速に解析するためにはさらに処理性能を大幅に増強した新しいスーパーコンピュータが必要である。

特に，演算速度の向上のためには素子の高速化，演算器の改良，演算器の並列化が不可欠であり，高性能な1チップ計算機が出現するにおよんでベンチャービジネスと言われる企業から並列計算機が市販されるようになってきた。今後は回路の集積度が進みパイプラインの1チップ化が可能になればなおのこと性能向上策としての並列化は避けられないものと思われる。

しかし，並列計算機の処理性能は応用プログラムの性質に大きく依存し，演算器の増設と性能向上とは同一義ではない。将来のスーパーコンピュータの一形態と考えられる並列計算機の性能・有用性を検証するためには，並列計算機を実際に開発し解析プログラムを動作させ，その処理時間により性能評価を行う方法が確かであるが，多額の費用・労力・時間を要する。他方，典型的なルーチンを選びその処理時間を机上で算出する簡便な方法がある。しかし，後者の場合はルーチンを正確に並列計算機の命令に変換しているか，またその命令を追跡する過程が正確に計算されているという保証が無い。そこで，以下に述べる方法で並列計算機の性能を推定することを計画した。

(1) 22MHzのクロックで動作し1MFLOPSの演算器および128KBの記憶装置が2cm×2cm×100cm程度の空間に実装できるものと仮定し，それを128×256台並べ（流れ場解析プログラムでは流れ方向に上下各々約100点，境界層方向に約100点を取ることが実用化の目安となる。この数に近い2冪乗として128と256を採用した）全体で32GFLOPSの処理性能を有する並列計算機を想定する（付録1および2参照）。

(2) この並列計算機の動作をクロック単位で模擬し計算結果をも出力するシミュレータおよびプログラム記述のためのアセンブラを開発する。

(3) 応用プログラムを並列処理に適応させ（計算法の変更およびメモリへの割り付け等を含む）並列計算機に移植する。

(4) 移植したプログラムの命令を並列計算機のシミュレータで追跡し記憶装置に対する競合，演算器の稼動状況・処理時間を検証し性能評価を行う。

(5) 最後に並列計算機の性能を左右する因子について考察して性能向上策を提言し，その結果もたらされる性能について推定し，本並列計算機の性能評価についてまとめる。

このたび，最大値の検索や二次元非圧縮非定常ナビエストークス方程式による解析プログラム処理時における並列計算機の模擬を実施したのでその性能について報告する。

2. 並列計算機のイメージ

計算機を命令の解読・制御を行う制御部とデータの加工を行う演算部およびデータの格納を行う記憶部に分割した場合，並列処理の可能な形態としては

(1) 制御部・演算部から成る処理装置を複数台配し記憶部を共有する形態（演算部に局所的な記憶部を有する形態も含む。図1）

(2) 制御部・演算部・記憶部から成る計算機を結合し必要に応じてデータを転送する形態（図2）

(3) 複数の演算部と複数の記憶部とを対応付ける結合回路を有し，原則として1個の制御部により同一処理を行う形態（図3）

がある。

科学技術計算を並列処理で高速化するためには各演算器の負荷を均一にし，同期を取りながら処理することが必要である。

また，(1)では処理装置の間で，(2)では計算機の間で互いに同期を取るための負荷が避けられずその動作も複雑であるため処理装置・計算機を大規

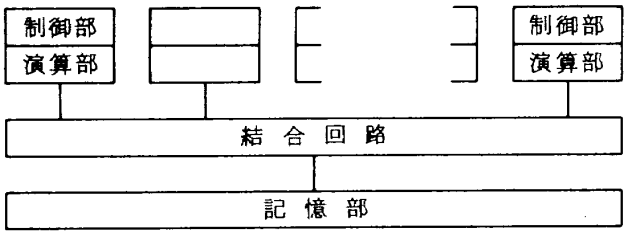


図1　マルチプロセッサ

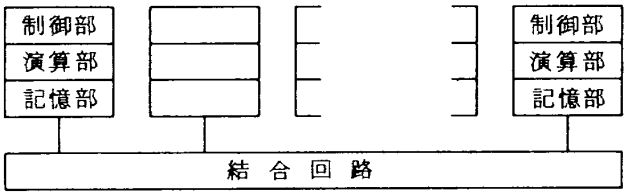


図2　複合計算機

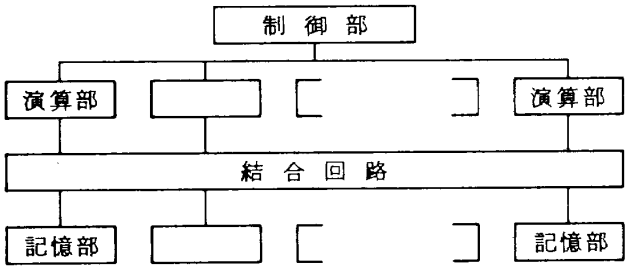


図3　並列計算機

模に増設してもそれに見合う性能向上が明確には判断できないのに対し(3)ではハードウェアで完全な同期を取るため動作も簡明であり増設に見合う性能向上が期待できる。このため並列計算機のイメージとして(3)の方式を基本として構成を決めることにした。

また，航空宇宙技術計算では機体・翼等の周りの流れ場解析が大きな比重を持っており，これらは流れ方向に約200点，境界層方向／翼幅方向に各々約100点に分割することが実用に耐えられる目安となる。

そこでアレイデータの演算では演算器を128×256台，記憶部を同数配置し，並列演算により処理速度の向上を図るべくその構成を検討した。複数の演算器と複数のメモリとを接続するネットワークについては多くの提案が発表されており[1)-12)]，これらを整理検討すると演算器・メモリの台数により接続ケーブル数等の実装上の制約から表1の

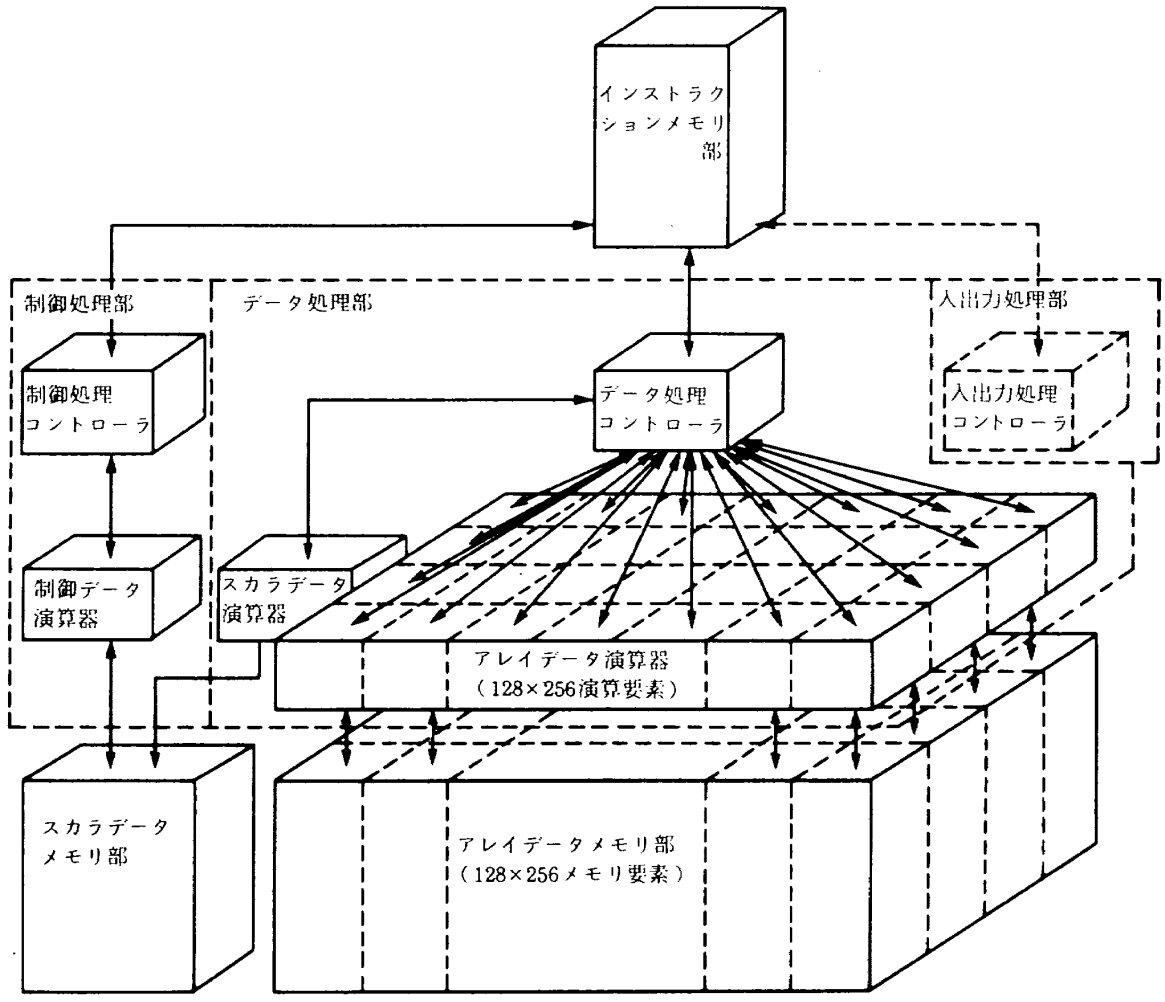


図4　並列計算機の構成

様になり，128×256台のネットワークとして隣接結合を採用した。

また入出力処理およびシステム全体を制御・監視する処理装置をスカラ/アレイデータを演算する処理装置から各々独立して設けることとした。

図4に並列計算機の構成を示す[13]。これはインストラクションメモリ部，制御処理部，データ処理部，入出力処理部，スカラデータメモリ部，アレイデータメモリ部より成る。

(1) インストラクションメモリ部（図5）

これは制御処理部，データ処理部，入出力処理部の命令を格納するインストラクションメモリと

表1 要素数と結合方式の目安

要素数	結合方式
～ 10^2程度	Cross point switch 等の完全結合
～ 10^3程度	Omega/N-cube network 等の多段結合
～ 10^4程度	Shuffle network 等の単段結合
それ以上	隣接結合

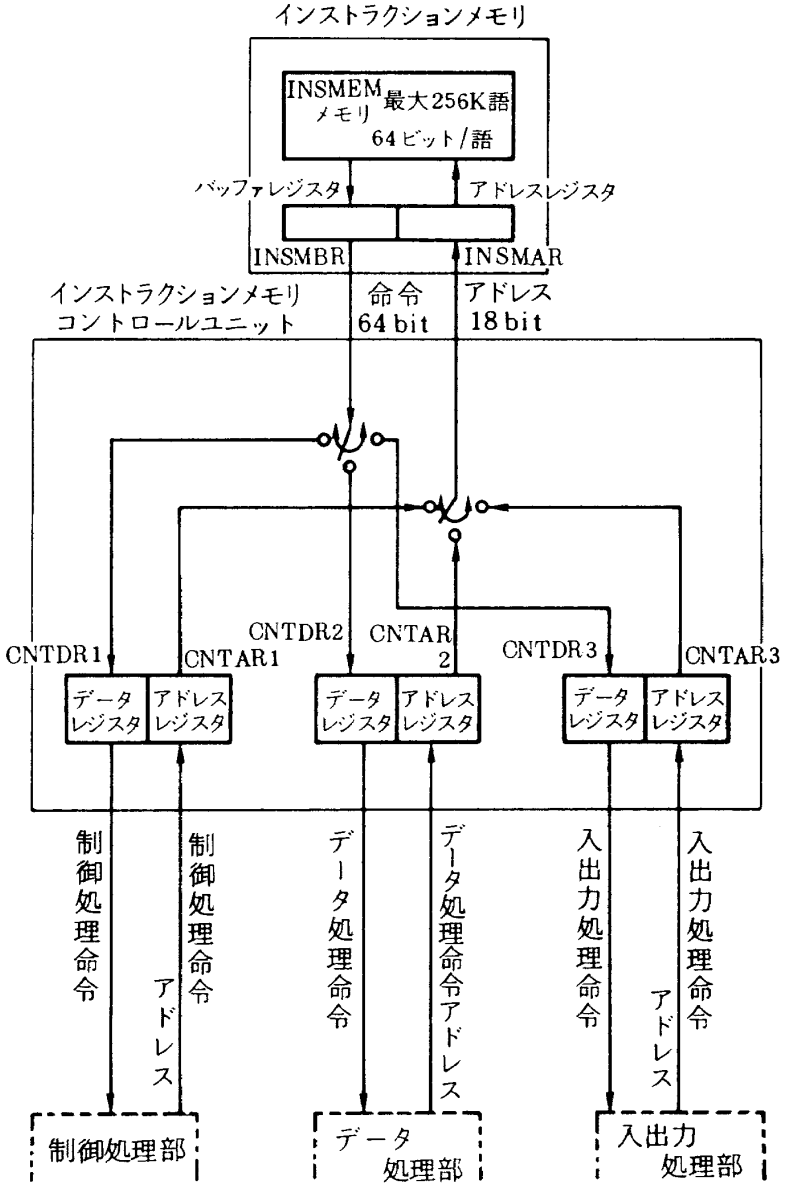


図5 インストラクションメモリ部の構成

メモリ読み出し/書き込みの制御を行うインストラクションメモリコントロールユニットから成る。

(a) インストラクションメモリ

1語64ビット，最大256K語を想定し，アクセスタイム5クロックのメモリ(INSMEM)とアドレスおよびデータを一時保持するレジスタ(INSMARおよびINSMBR)より成る。

(b) インストラクションメモリコントロールユニット

制御処理部，データ処理部および入出力処理部に対応してデータレジスタ(CNTDR1，CNTDR2およびCNTDR3)，アドレスレジスタ(CNTAR1，CNTAR2およびCNTAR3)が用意されておりメモリアクセス競合を

1. 入出力処理部
2. データ処理部
3. 制御処理部

の優先順位に従って制御する。

(2) 制御処理部(図6)

これは制御処理コントローラ，制御データ演算

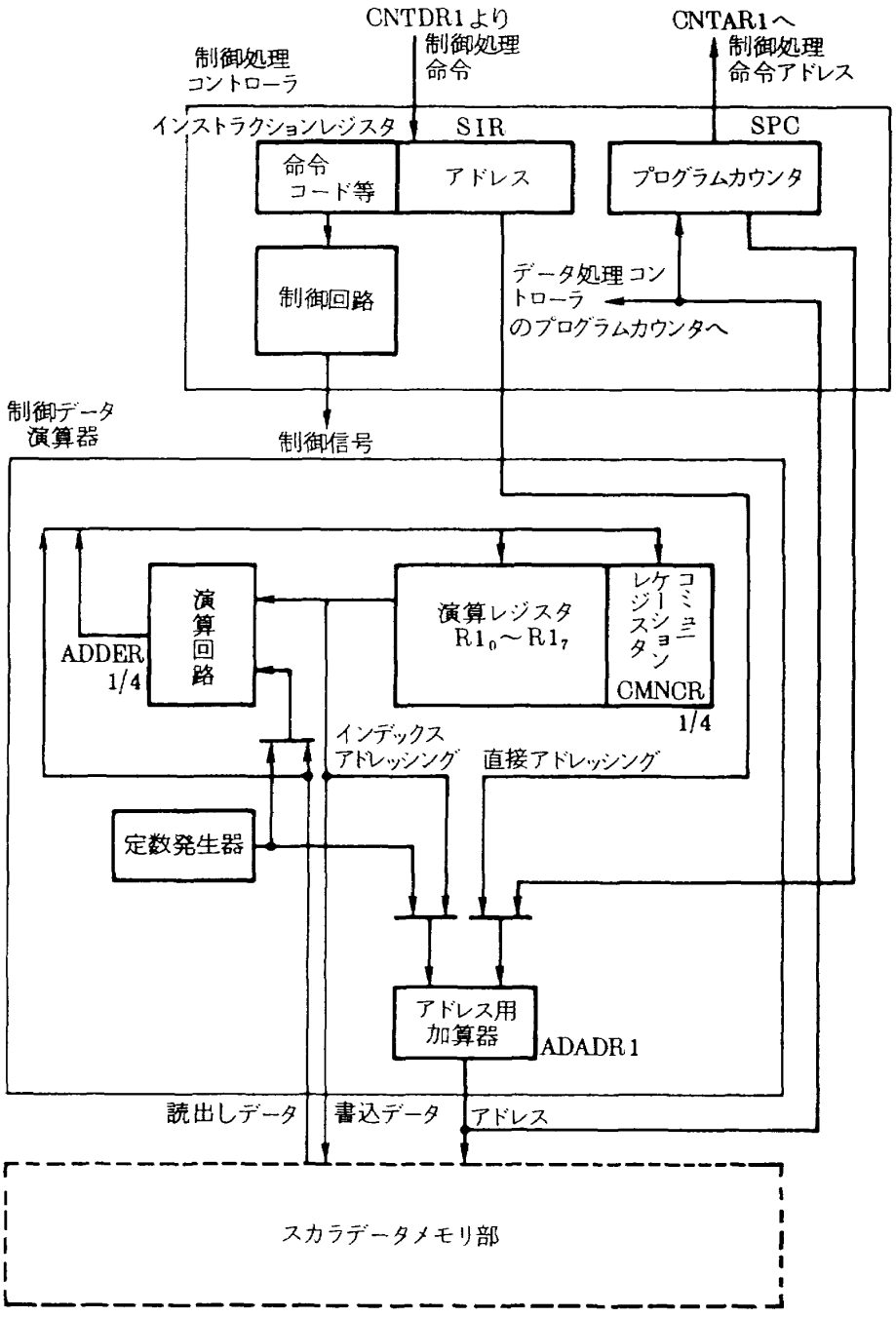


図6　制御処理部の構成

器から成りシステム全体を制御・監視するための処理を行う。また後述するデータ処理部と共にスカラデータメモリ部を共用する。

(a) 制御処理コントローラ

インストラクションメモリ部に命令読み出し要求を出し命令を解読する。

その構成は命令の記憶番地を保持するプログラムカウンタ (SPC)，命令を保持するインストラクションレジスタ (SIR) と命令を解読しその実行を進める信号を発進する制御回路より成る。

(b) 制御データ演算器

演算器は固定小数点乗除加減算回路 (ADDER1/4)，固定小数点演算レジスタ8個(R10～R17：このうちR11～R17は指標レジスタを兼ねる。付録3参照)，データ処理部との間でデータを転送するためのコミュニケーションレジスタ (CMNCR 1/4)，アドレス用加算器(ADADR1) および定数発生器より成る。

(3) データ処理部(図7)

ここで想定した並列計算機のデータ処理部はその動作が簡明で同期にともなう負荷が大きくならないSIMD方式とし，データ処理コントローラとスカラデータ演算器およびアレイデータ演算器より成る。

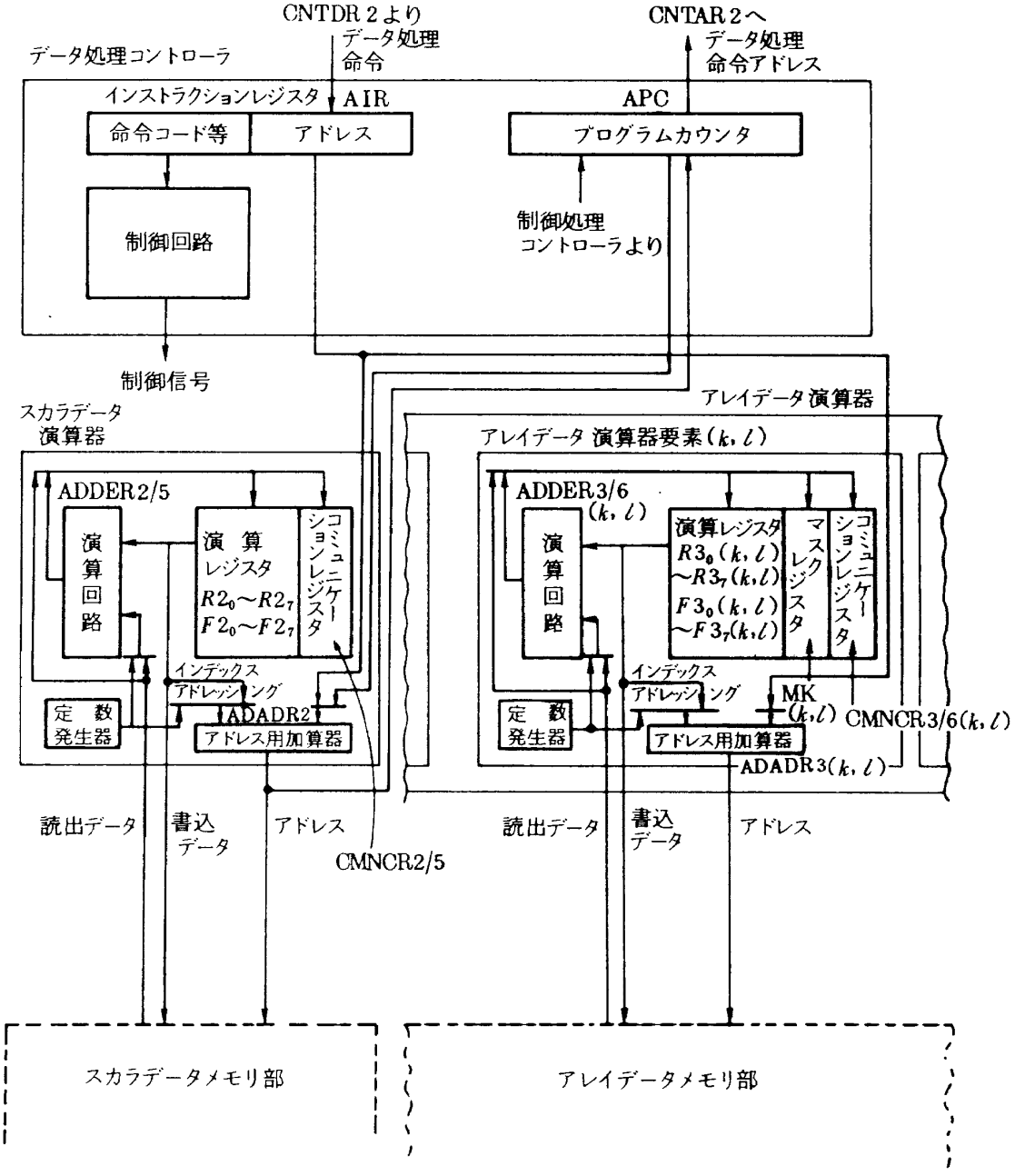


図7 データ処理部の構成

(a)　データ処理コントローラ

これはインストラクションメモリより送られて来る命令を解読し，スカラデータ演算器 / アレイデータ演算器で処理すべきかを判定し処理を進める。その構成は命令の記憶番地を保持するカウンタ (APC)，命令を保持するインストラクションレジスタ (AIR) および命令を読み出し解読し実行するための信号を発進する制御回路より成る。

(b)　スカラデータ演算器

この演算器は固定 / 浮動小数点乗除加減算回路 (ADDER 2/5)，固定小数点演算レジスタ 8 個 (R 20～R27 ; このうちR21～R27は指標レジスタを兼ねる。付録 3 参照)，浮動小数点演算レジスタ 8 個 (F20～F27)，アレイデータ演算器および制御データ演算器との間でデータを転送するためのコミュニケーションレジスタ (CMNCR 2/5)，アドレス用加算器 (ADADR2) および定数発生器より成る。

(c)　アレイデータ演算器

これは128×256台の演算器要素から構成されており，各要素は固定 / 浮動小数点乗除加減算回路 (ADDER 3/6 (k, l))，固定小数点演算レジスタ 8 個 (R30(k, l)～R37 (k, l) ; このうちR31 (k, l) ～R37 (k, l) は指標レジスタを兼ねる)，浮動小数点演算レジスタ 8 個 (F30(k, l)～F37(k, l)，マスクレジスタ (MK(k, l))，スカラデータ演算器との間でデータを転送するためのコミュニケーションレジスタ (CMNCR 3/6 (k, l))，アドレス用加算器 (ADADR 3 (k, l)) および定数発生器より成る。

なお，各演算器のコミュニーケーションレジスタは CMNCR 1/4 と CMNCR 2/5 とは 1 対 1 で，CMNCR 2/5 と CMNCR 3/6 (0, 0)～CMNCR 3/6 (127, 255) とはマスクレジスタ制御下のスキャナーを通して 1 対128×256で接続され互いにデータの送受ができる (図 8)。

(4)　入出力処理部

入出力処理はプログラムに大きく依存するため，多くのプログラムを試して性能を決めるべきとの考えからこの部分についての詳細は省略する。

(5)　スカラデータメモリ部 (図 9)

これは制御処理部およびデータ処理部に共有され，スカラデータを格納するスカラデータメモリと両処理部とスカラデータメモリとの間にあり，メモリへの読み出し / 書き込みの制御を行うスカラデータメモリコントロールユニットから成る。

(a)　スカラデータメモリ

1 語64ビット最大256K語を想定しアクセスタイム 5 クロックのメモリ (SDMEM) とアドレスおよびデータを一時保持するレジスタ (SDMMAR および SDMMBR) より成る。

(b)　スカラデータメモリコントロールユニット

制御処理部とデータ処理部に対応してデータレジスタ (DMBDR 1/4 および DMBAR 2/5)，アドレスレジスタ (DMBAR1 および DMBAR2) が用意されており両者からのメモリアクセス競合を優先順位

1.　データ処理部
2.　制御処理部

に従って制御する。

(6)　アレイデータメモリ部 (図10)

これはアレイデータを格納するアレイデータメモリと演算器およびメモリとを接続するアレイデータメモリコントロールユニットより成る。

(a)　アレイデータメモリ

これは128×256個のメモリ要素から構成されており，各要素は 1 語64ビット最大16K語を想定した記憶装置 (ADMEM (k, l))，およびデータ / アドレスを一時保持するレジスタ (ADMMBR (k, l)) および (ADMMAR (k, l)) より成る。メモリアクセスタイムは 5 クロックである。

(b)　アレイデータメモリコントロールユニット

これはアレイデータ演算器要素とアレイデータメモリ要素とを対応つける結合網で128×258対のデータおよびアドレスを保持するレジスタ (DMBDR 3/6 (k, l) および DMBAR 3 (k, l)) を行列各々隣接結合による上下左右のシフトレジスタで終端がリング結合しているネットワークを成す(図11)。

メモリ要素からデータを読み出すときには各演算器要素 (k, l) よりアドレスを DMBAR 3 (k, l) に移し以後データ処理コントローラの制御により 1 クロック毎に行または列方向に 1 個だけ移動し，所定の位置 (k', l') に到達した後メモリ要素の

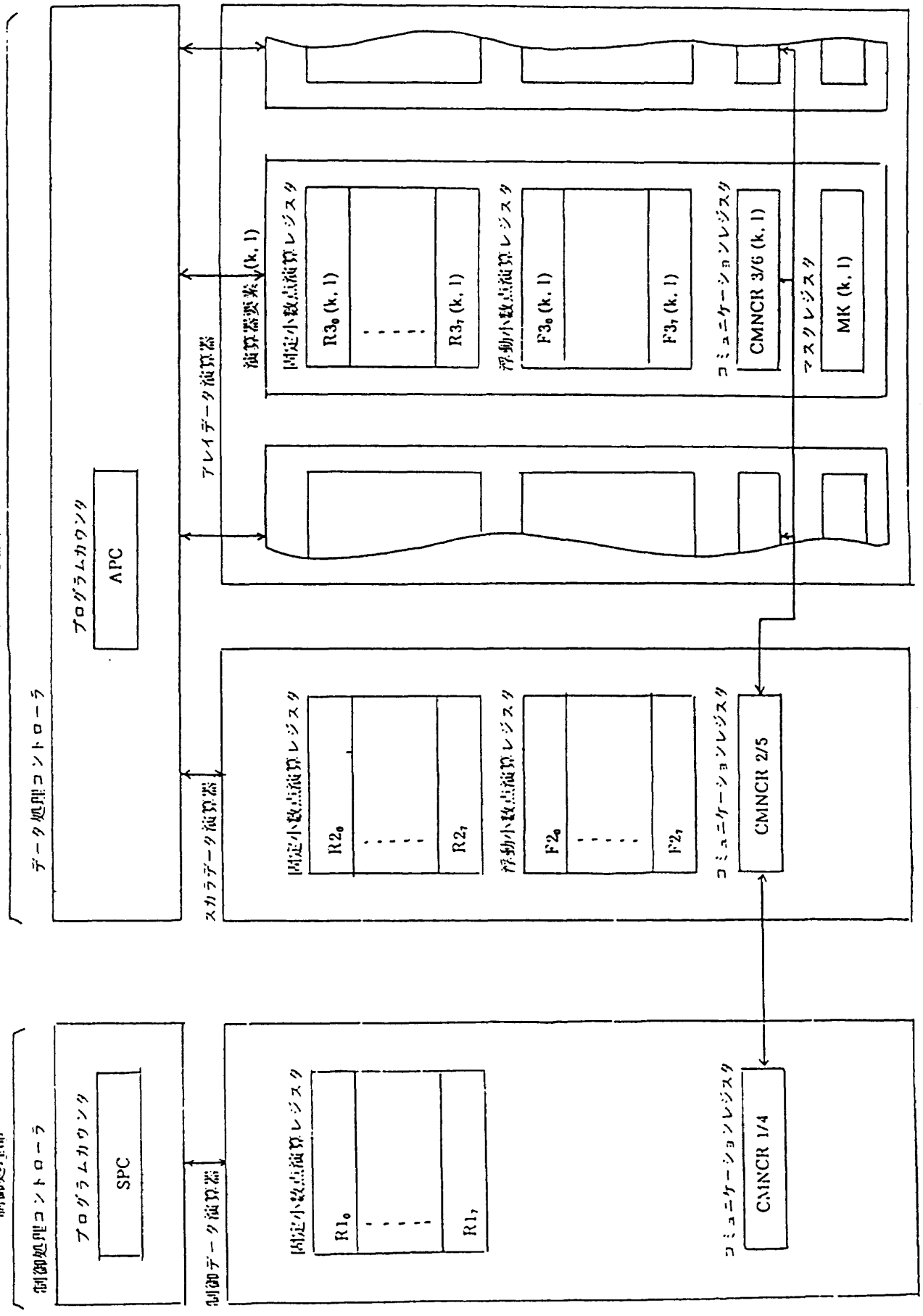


図8 制御処理部/データ処理部のレジスタ

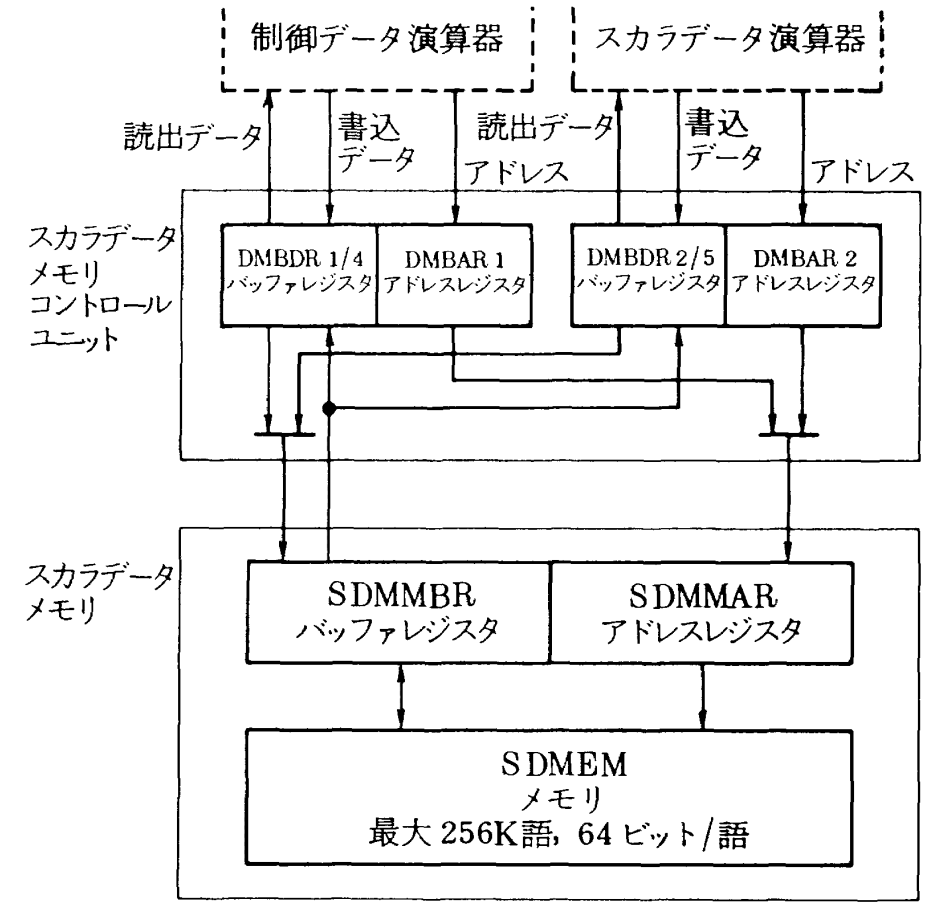


図9　スカラデータメモリ部の構成

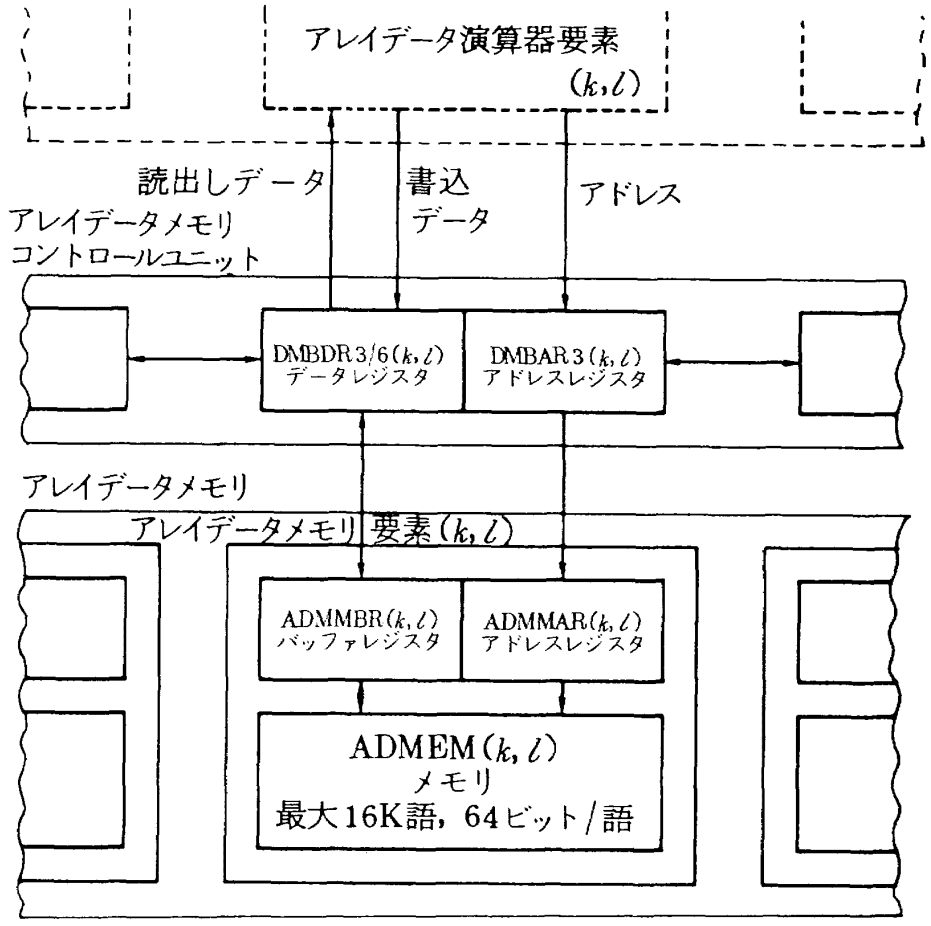


図10　アレイデータメモリ部の構成

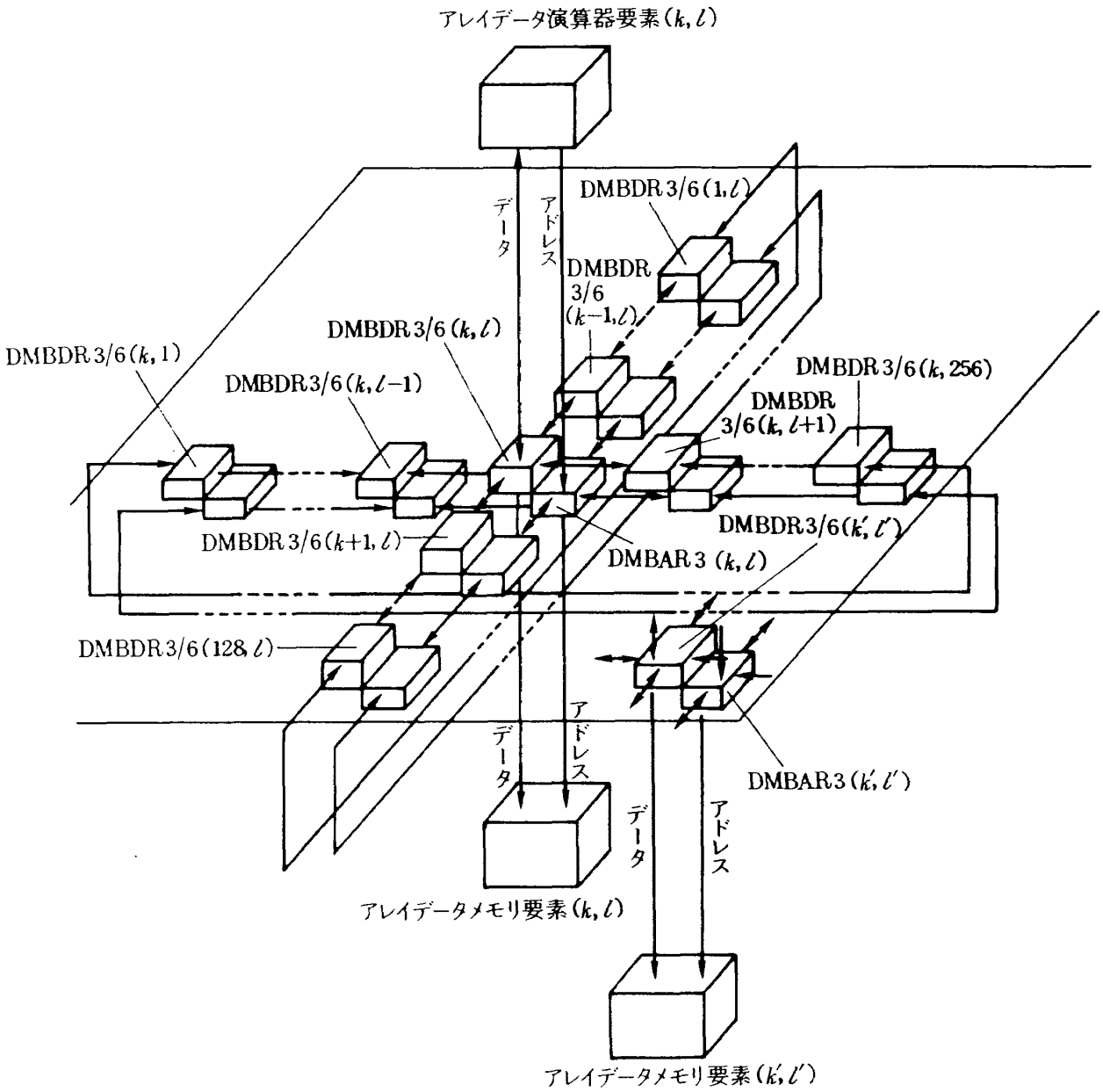


図11 アレイデータメモリコントロールユニットの構成

ADMMAR (k', l') に移され読み出しがおこなわれる。5クロック後に読み出されたデータはADMMBR (k', l') に入りそれをDMBDR3/6 (k, l) まで行および列を移動させ，データを演算器要素 (k', l) に移す。データを書き込む時にはデータおよびアドレスを演算器要素 (k, l) よりDMBDR3/6 (k, l)，DMBAR3 (k, l) に移し以後所定の位置 (k', l') まで行列を1個毎移動した後にメモリ要素のADMMBR (k', l')，ADMMAR (k', l') に移し書き込む。

3. 命令系とその処理

並列計算機の命令はインストラクションメモリに格納され各処理部より読み出され解読・実行される。

全命令は64ビットで構成され表2に制御処理部，表3にデータ処理部の命令を，また略号を以下に示す。

ここで

i と j は 0～7

k は 0～127

l は 0～255

を表す。

R/Ri/Rj ；制御データ演算器およびスカラデータ演算器の固定小数点演算レジスタ

F/Fi/Fj ；制御データ演算器およびスカラデー

タ演算器の浮動小数点演算レジスタ

R (k, l) ；アレイデータ演算要素 (k, l) の固定小数点演算レジスタ

F (k, l) ；アレイデータ演算要素 (k, l) の浮動小数点演算レジスタ

X ；アドレス

$\tilde{X}$ ；コントロール命令ではインストラクションメモリの実行アドレス，スカラ演算命令ではスカラデータメモリの実行アドレス

$\tilde{X}$ (k, l) ；アレイデータメモリ要素 (k, l) の実行アドレス

〔　〕 ；レジスタおよび記憶装置の内容

T ；指標修飾

0の時；

$\tilde{X}=X$ および $\tilde{X}(k, l)=X$

1～7の時；

$\tilde{X}=X+$〔R〕および

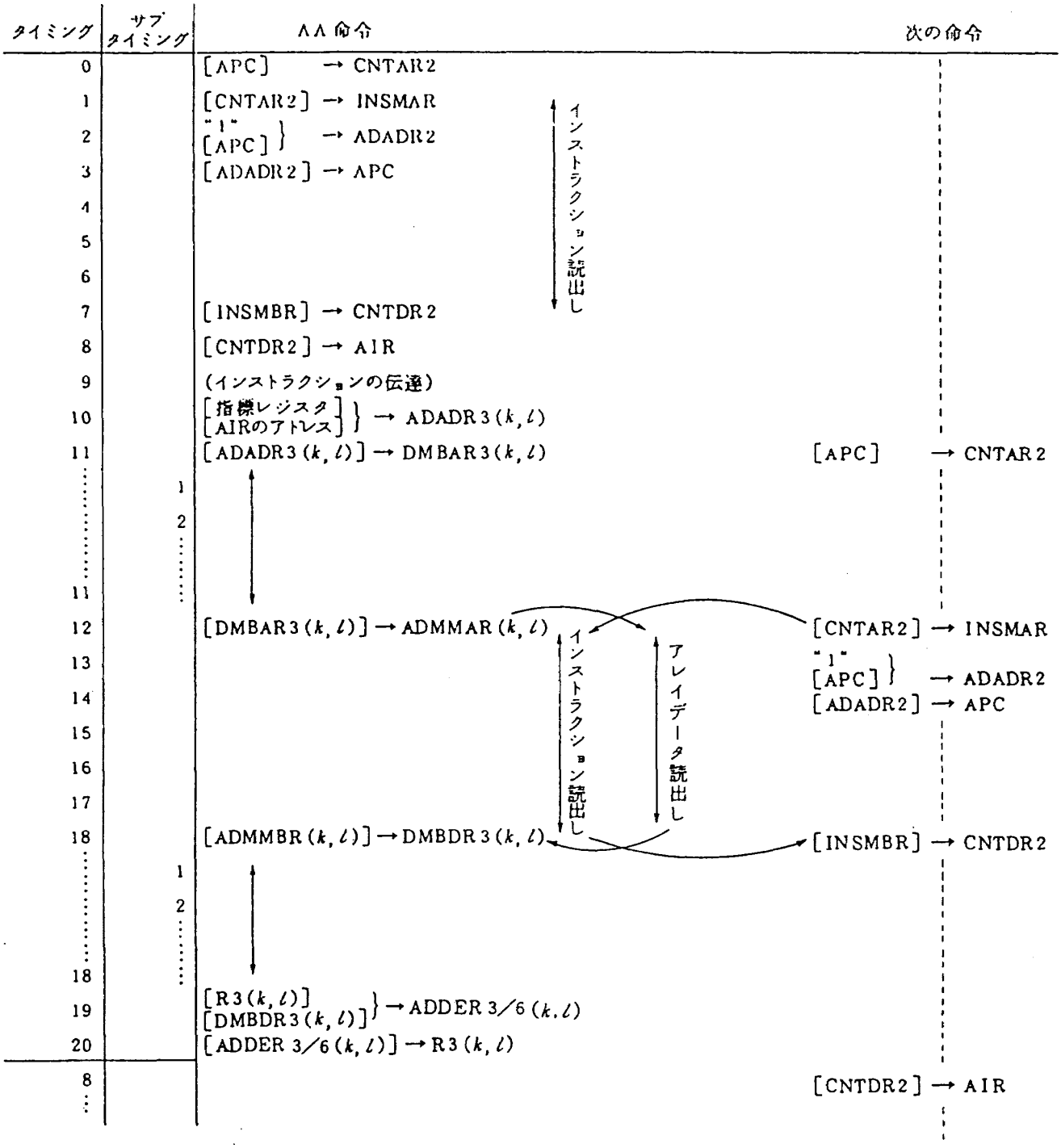


図12　Add Array (AA) 命令のタイムチャート

$\tilde{X}(k, l) = X + [R(k, l)]$

C ；演算結果の条件判定
4の時；零
2の時；負

EC ；命令実行の条件判定
0の時；マスク状態に関係なく実行
1の時；マスクが“ON”の要素のみ実行
2の時；マスクが“OFF”の要素のみ実行
3の時；マスク状態に関係なく実行

MO ；条件(C)を満たす時のマスク操作
0の時；マスク操作なし
1の時；マスクを“ON”にする

表2 制御処理部の命令

	命令	命令記号	命令
コントロール命令	Jump	J	1100 0000 / T / X
	Jump Minus	JM	1100 0001 / R / T / X
	Jump Zero	JZ	1100 0010 / R / T / X
	Synchronize Jump	SJ	1100 0101 / T / X
	Start Arithmetic Processor	SAP	1100 0110 / T / X
	Halt and Preceed	HP	1100 0111
メカラ係命令	Add	A	0100 0000 / R / T / X
	Subtract	S	0100 0001 / R / T / X
	Multiply	M	0100 0010 / R / T / X
	Divide	D	0100 0011 / R / T / X
	Load	L	0100 0100 / R / T / X
	Transfer	T	0100 0101 / R / T / X
	Add Register	AR	0100 0110 / R_1 / R_2 / C
	Subtract Register	SR	0100 0111 / R_1 / R_2 / C
	Multiply Register	MR	0100 1000 / R_1 / R_2 / C
	Divide Register	DR	0100 1001 / R_1 / R_2 / C
	Move Register	MV	0100 1010 / R_1 / R_2 / C
	Load Negative	LN	0100 1110 / R_1 / R_2 / C
	Compare	CMP	0100 1111 / R_1 / R_2 / C
	Increase	IC	0100 0000 / R_1 / C
特殊命令	Read Arithmetic Communication Register	RAC	1000 0101
	Load Supervisor Communication Register	LSC	1000 0110 / R
	Store to ・ ・ ・	SSC	1000 1010 / R

2の時；マスクが“OFF”にする
3の時；マスク操作なし

CB　；アレイデータメモリコントロールユニットの結合状態（0；隣接結合）

LS　；データの移動行数

CS　；データの移動列数

SPC　；制御処理部のプログラムカウンタ

APC　；データ処理部のプログラムカウンタ

CMNCR1/4　；制御データ演算器のコミュニケーションレジスタ

CMNCR2/5　；スカラデータ演算器のコミュニケーションレジスタ

CMNCR3/6(k, l)；アレイデータ演算器のコミュニケーションレジスタ

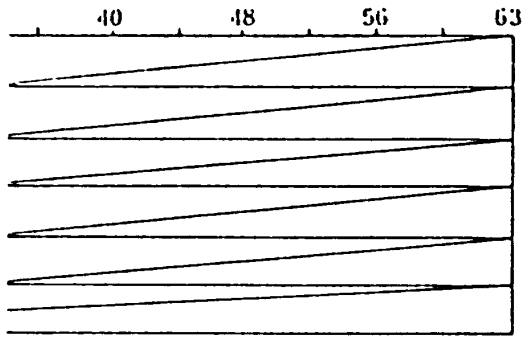
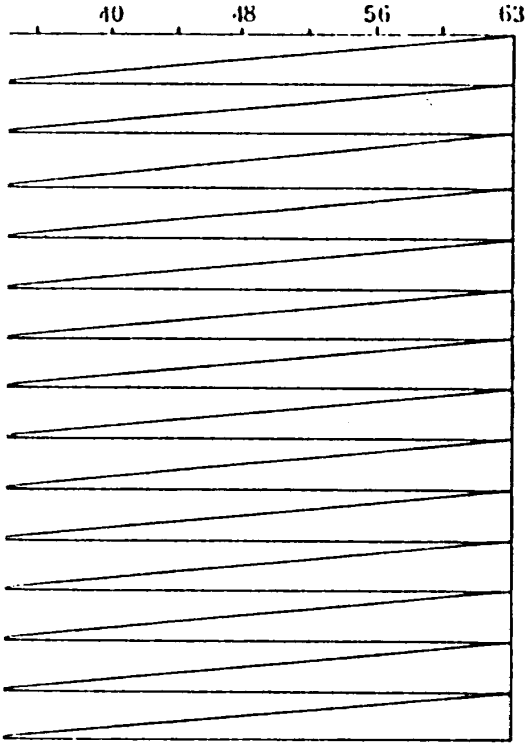
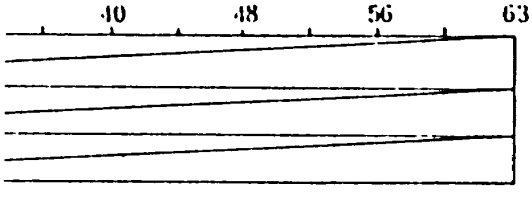
型　式	内　容
	$\widetilde{X}$ → SPC [R] < の時 $\widetilde{X}$ → SPC [R] = 0 〃　〃　〃 データ処理部が動作中の時 $\widetilde{X}$ → SPC $\widetilde{X}$ → APC とし，データ処理部を起動する。 制御処理部が停止する。
	[R] + [$\widetilde{X}$] → R [R] - [$\widetilde{X}$] → R [R] × [$\widetilde{X}$] → R [R] / [$\widetilde{X}$] → R [$\widetilde{X}$] → R [R] → $\widetilde{X}$ [R_i] + [R_j] → R_i，C を満す時次の命令をスキップ [R_i] - [R_j] → R_i，　〃　〃 [R_i] × [R_j] → R_i，　〃　〃 [R_i] / [R_j] → R_i，　〃　〃 [R_j] → R_i，　〃　〃 −[R_j] → R_i，　〃　〃 [R_i] − [R_j] が C を満たす時次の命令をスキップ [R_i] + 1 → R_i，C を満たす時次の命令をスキップ
	[CMNCR2/5] → CMNCR1/4 [CMNCR1/4] → R [R] → CMNCR1/4

表3 データ処理部の命令（その1）

	命令	命令記号	命令
			0 8 16 24 32
コントロール命令	Jump	J	1100 0000 / T X
	Jump Minus	JM	1100 0001 R T X
	Jump Zero	JZ	1100 0010 R T X
	Floating Jump Minus	FJM	1100 0011 F T X
	Floating Jump Zero	FJZ	1100 0100 F T X
	Halt and Preceed	HP	1100 0111
	Mask Initiate	MI	1100 1000
			0 8 16 24 32
スカラー命令	Add	A	0100 0000 R T X
	Subtruct	S	0100 0001 R T X
	Multiply	M	0100 0010 R T X
	Divide	D	0100 0011 R T X
	Load	L	0100 0100 R T X
	Transfer	T	0100 0101 R T X
	Add Register	AR	0100 0110 R_i R_j C
	Subtract Register	SR	0100 0111 R_i R_j C
	Multiply Register	MR	0100 1000 R_i R_j C
	Divide Register	DR	0100 1001 R_i R_j C
	Move Register	MV	0100 1010 R_i R_j C
	Load Negative	LN	0100 1110 R_i R_j C
	Compare	CMP	0100 1111 R_i R_j C
	Increase	IC	0101 0000 R_i C
	Floating Add	FA	0110 0000 F T X
	〃 Subtract	FS	0110 0001 F T X
	〃 Multiply	FM	0110 0010 F T X
	〃 Divide	FD	0110 0011 F T X
	〃 Load	FL	0110 0100 F T X
	〃 Transfer	FT	0110 0101 F T X
	〃 Add Register	FAR	0110 0110 F_i F_j C
	〃 Subtract Register	FSR	0110 0111 F_i F_j C
	〃 Multiply Register	FMR	0110 1000 F_i F_j C
	〃 Divide Register	FDR	0110 1001 F_i F_j C
	〃 Move Register	FMV	0110 1010 F_i F_j C
	〃 Load Negative	FLN	0110 1110 F_i F_j C
	〃 Compare	FCMP	0110 1111 F_i F_j C

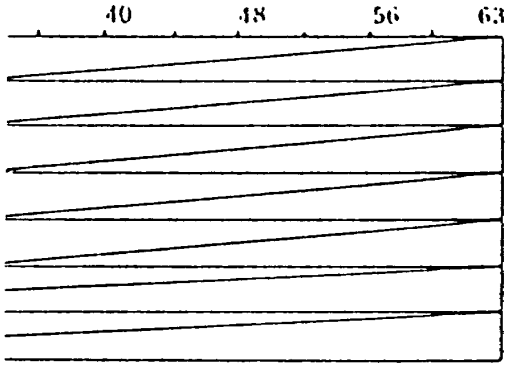
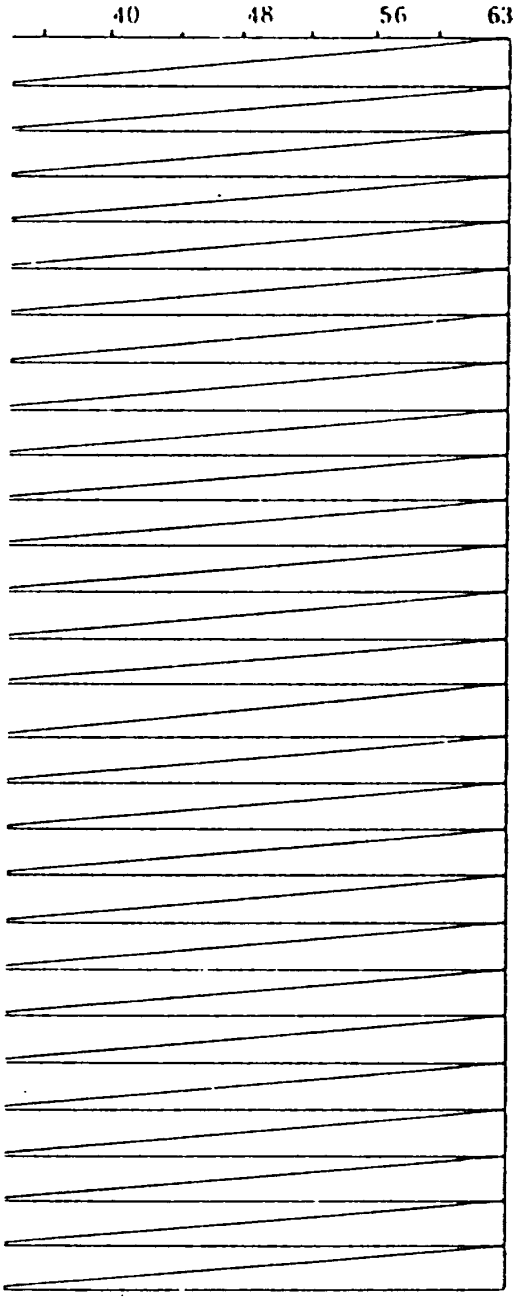
形　式	内　容
	$\widetilde{X} \rightarrow$ APC $[R]<$の時 $\widetilde{X} \rightarrow$ APC $[R]=0$ 〃　〃　〃 $[F]<0$ 〃　〃　〃 $[F]=0$ 〃　〃　〃 データ処理部が停止する。 $0 \rightarrow \{MK(k,\ell);\ k=1\sim128,\ \ell=1\sim256\}$
	$[R]+[\widetilde{X}] \rightarrow R$ $[R]-[\widetilde{X}] \rightarrow R$ $[R]\times[\widetilde{X}] \rightarrow R$ $[R]/[\widetilde{X}] \rightarrow R$ $[\widetilde{X}] \rightarrow R$ $[R] \rightarrow \widetilde{X}$ $[R_i]+[R_j] \rightarrow R_i$ ．Cを満す時次の命令をスキップ $[R_i]-[R_j] \rightarrow R_i$ ．　〃　〃 $[R_i]\times[R_j] \rightarrow R_i$ ．　〃　〃 $[R_i]/[R_j] \rightarrow R_i$ ．　〃　〃 $[R_j] \rightarrow R_i$ ．　〃　〃 $-[R_i] \rightarrow R_i$ ．　〃　〃 $[R_i]-[R_j]$がCを満たす時次の命令をスキップ $[R_i]+1 \rightarrow R_i$．Cを満たす時次の命令をスキップ $[F]+[\widetilde{X}] \rightarrow F$ $[F]-[\widetilde{X}] \rightarrow F$ $[F]\times[\widetilde{X}] \rightarrow F$ $[F]/[\widetilde{X}] \rightarrow F$ $[\widetilde{X}] \rightarrow F$ $[F] \rightarrow \widetilde{X}$ $[F_i]+[F_j] \rightarrow F_i$．Cを満たす時次の命令をスキップ $[F_i]-[F_j] \rightarrow F_i$．　〃　〃 $[F_i]\times[F_j] \rightarrow F_i$．　〃　〃 $[F_i]/[F_j] \rightarrow F_i$．　〃　〃 $[F_j] \rightarrow F_i$ ．　〃　〃 $-[F_j] \rightarrow F_i$ ．　〃　〃 $[F_i]-[F_j]$がCを満たす時次の命令をスキップ

表3　データ処理部の命令（その2）

	命令	命令記号	命令 0 8 16 24 32
	Add Array	AA	0000 0000 R T EC MO C CB
	Subtruct Array	SA	0000 0001 R T EC MO C CB
	Multiply Array	MA	0000 0010 R T EC MO C CB
	Divide Array	DA	0000 0011 R T EC MO C CB
	Load Array	LA	0000 0100 R T EC MO C CB
	Transfer Array	TA	0000 0101 R T EC MO C CB
	Add Register Array	ARA	0000 0110 R_i R_j EC MO C
ア	Subtruct Register Array	SRA	0000 0111 R_i R_j EC MO C
	Multiply Register Array	MRA	0000 1000 R_i R_j EC MO C
レ	Divide Register Array	DRA	0000 1001 R_i R_j EC MO C
	Move Register Array	MVA	0000 1010 R_i R_j EC MO C
イ	Load Negative Array	LNA	0000 1110 R_i R_j EC MO C
	Compare Array	CMPA	0000 1111 R_i R_j EC MO C
ｲﾝｸ	Increase Array	ICA	0001 0000 R_i EC MO C
	Floating Add Array	FAA	0010 0000 F T EC MO C CB
ﾌﾟ	• Subtract Array	FSA	0010 0001 F T EC MO C CB
	• Multiply Array	FMA	0010 0010 F T EC MO C CB
ﾌﾞ	• Divide Array	FDA	0010 0011 F T EC MO C CB
	• Load Array	FLA	0010 0100 F T EC MO C CB
ﾌﾝ	• Transfer Array	FTA	0010 0101 F T EC MO C CB
	• Add Register Array	FARA	0010 0110 F_i F_j EC MO C
	• Subtract Register Array	FSRA	0010 0111 F_i F_j EC MO C
	• Multiply Register Array	FMRA	0010 1000 F_i F_j EC MO C
	• Divide Register Array	FDRA	0010 1001 F_i F_j EC MO C
	• Move Register Array	FMVA	0010 1010 F_i F_j EC MO C
	• Load Negative Array	FLNA	0010 1110 F_i F_j EC MO C
	• Compare Array	FCMPA	0010 0000 F_i F_j EC MO C
	Move Arithmetic Communication Register	MAC	1000 0000
ﾑｰﾌﾞ	Move Communication Register	MCR	1000 0001
ｽﾄｱ	Store to Communication Register	SCR	1000 0010 R
	Floating Store to Communication Register	FSCR	1000 0011 F
ﾛｰﾄﾞ	Read Supervisor Communication Register	RSC	1000 0100
	Load Communication Register	LCR	1000 1000 R
	Floating Load Communication Register	FLCR	1000 1001 F

型　式	内　容
40 48 56 63 LS CS X	[R(k,ℓ)]＋[$\widetilde{X}$(k,ℓ)]→R(k,ℓ)，Cを満たす時マスク操作
LS CS X	[R(k,ℓ)]－[$\widetilde{X}$(k,ℓ)]→R(k,ℓ)，〃
LS CS X	[R(k,ℓ)]×[$\widetilde{X}$(k,ℓ)]→R(k,ℓ)，〃
LS CS X	[R(k,ℓ)]／[$\widetilde{X}$(k,ℓ)]→R(k,ℓ)，〃
LS CS X	[$\widetilde{X}$(k,ℓ)]→R(k,ℓ)，〃
LS CS X	[R(k,ℓ)]→$\widetilde{X}$(k,ℓ)，〃
	[R_i(k,ℓ)]＋[R_j(k,ℓ)]→R_i(k,ℓ)，Cを満たす時マスク操作
	[R_i(k,ℓ)]－[R_j(k,ℓ)]→R_i(k,ℓ)，〃
	[R_i(k,ℓ)]×[R_j(k,ℓ)]→R_i(k,ℓ)，〃
	[R_i(k,ℓ)]／[R_j(k,ℓ)]→R_i(k,ℓ)，〃
	[R_j(k,ℓ)]→R_i(k,ℓ)，〃
	－[R_j(k,ℓ)]→R_i(k,ℓ)，〃
	[R_i(k,ℓ)]－[R_j(k,ℓ)]がCを満たす時マスク操作
	[R_i(k,ℓ)]＋1→R_i(k,ℓ)，C　〃　〃
LS CS X	[F(k,ℓ)]＋[$\widetilde{X}$(k,ℓ)]→F(k,ℓ)，Cを満たす時マスク操作
LS CS X	[F(k,ℓ)]－[$\widetilde{X}$(k,ℓ)]→F(k,ℓ)，〃
LS CS X	[F(k,ℓ)]×[$\widetilde{X}$(k,ℓ)]→F(k,ℓ)，〃
LS CS X	[F(k,ℓ)]／[$\widetilde{X}$(k,ℓ)]→F(k,ℓ)，〃
LS CS X	[$\widetilde{X}$(k,ℓ)]→F(k,ℓ)，〃
LS CS X	[F(k,ℓ)]→$\widetilde{X}$(k,ℓ)，〃
	[F_i(k,ℓ)]＋[F_j(k,ℓ)]→F_i(k,ℓ)，Cを満たす時マスク操作
	[F_i(k,ℓ)]－[F_j(k,ℓ)]→F_i(k,ℓ)，〃
	[F_i(k,ℓ)]×[F_j(k,ℓ)]→F_i(k,ℓ)，〃
	[F_i(k,ℓ)]／[F_j(k,ℓ)]→F_i(k,ℓ)，〃
	[F_j(k,ℓ)]→F_i(k,ℓ)，〃
	－[F_j(k,ℓ)]→F_i(k,ℓ)，〃
	[F_i(k,ℓ)]－[F_j(k,ℓ)]がCを満たす時マスク操作
40 48 56 63	[CMNCR2/5]→CMNCR3/6(k,ℓ)（ただしMK(k,ℓ)が'OFF'）
	MK(k,ℓ)が'OFF'でk,ℓが最小の[CMNCR3/6(k,ℓ)]→CMNCR2/5
	[R]→CMNCR2/5かつ[R(k,ℓ)]→CMNCR3/6(k,ℓ)
	[F]→CMNCR2/5かつ[F(k,ℓ)]→CMNCR3/6(k,ℓ)
	[CMNCR1/4]→CMNCR2/5
	[CMNCR2/5]→Rかつ[CMNCR3/6(k,ℓ)]→R(k,ℓ)
	[CMNCR2/5]→Fかつ[CMNCR3/6(k,ℓ)]→F(k,ℓ)

MK (k, l) ；マスクレジスタ

全ての命令はその実行手順が決められており，一例として図12にアレイデータの加算命令(AA)の実行手順を示す。ここで各タイミングの概略は

タイミング0～7；命令を読み出す動作
タイミング8　；命令を解読する動作
タイミング9　；命令を解読し各演算器に制御信号を伝達する動作
タイミング10　；オペランド番地の計算
タイミング11　；オペランド番地の設定と結合網の動作によるメモリ要素の選択
タイミング12～17；オペランドの読み出し
タイミング18　；オペランドの逆転送
タイミング19～20；配列演算の実行

を示す。またタイミング11～18で次の命令を先取りする様子も示されている。

さらに，メモリの競合が起きタイミングが進められない時にはメモリが解除されるまで待ちになる。

4. 並列計算機のアセンブラ

計算機は機械語の命令しか理解できないため応用プログラムから演算結果を得るためには処理手順をその機械語に変換する必要がある。しかし直接機械語に変換することは多大の労力が必要であり誤りも起き易い。また，高級言語で記述し直接機械語に翻訳できれば理想ではあるが並列計算機用のコンパイラを開発しなければならず並列計算機の有用性を検証する手続きとしてはあまりにも大げさである。

そこで前記二通りの手段の中間であるアセンブ

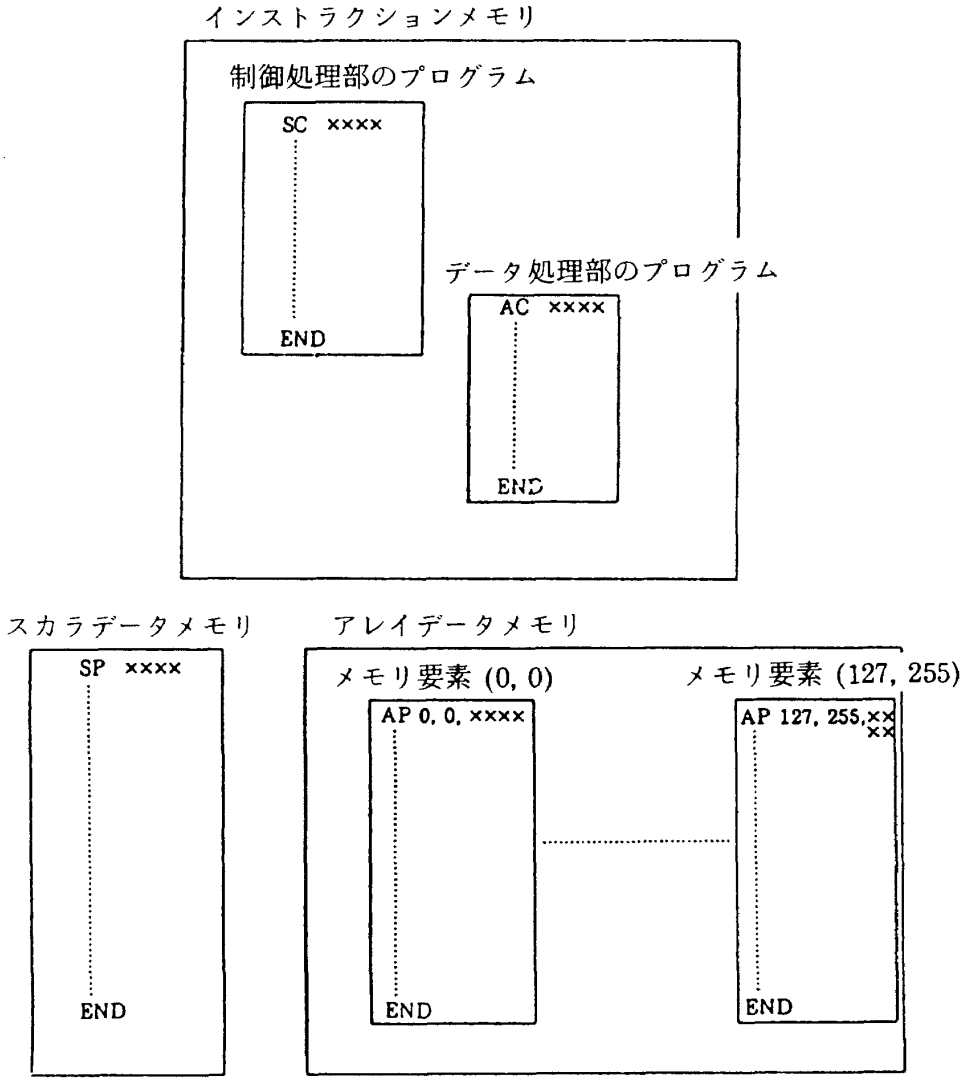


図13　並列計算機のプログラム構造

ラを開発することにした[14]。

4.1 アセンブラの機能

アセンブラの原始プログラムは文から成り文は

ラベル　命令　オペランド

の形式をし，ラベルおよび命令は記号で表され，オペランドは命令によりレジスタを示すレジスタオペランド，アドレスを示すアドレスオペランド等に分かれるが共に記号および数値による式で表される。

以下にアセンブラの機能を示す。

(1) プログラム構造の定義

並列計算機のプログラムは図13に示す構造であることを前提とし表4に示す命令で構造を指定する。

(2) 記号の直接定義

記号に数値を直接対応付ける。数値としては整数および実数であり，その表記法は定義済み記号を用いた式である（表5）。

(3) アドレスの自動割り当ておよび機械命令語の生成

表2，表3に示す命令に記号を対応させ1命令ごとに機械命令に変換し，この命令と同じ文に付けられた記号（ラベル）にアドレスの値を割り当てるアセンブラの中核を成す機能である。

(4) 作業域の確保と定数の生成

スカラデータメモリおよびアレイデータメモリ要素内に作業域および定数を生成する（表6）。

定数は1語64ビットが割り当てられ，整数では下32ビットに，実数では上32ビットに生成する。

表4　プログラム構造を定義する命令

表　記	意　味
SC　アドレス	制御処理部の命令列を定義する。アドレスはインストラクションメモリ内における命令列の先頭アドレスを示す。
AC　アドレス	データ処理部の命令列を定義する。アドレスはインストラクションメモリ内における命令列の先頭アドレスを示す。
SP　アドレス	スカラデータメモリの作業域および定数を定義する。アドレスはスカラデータメモリ内における作業域および定数の先頭アドレスを示す。
AP　行，列，アドレス	アレイデータメモリ要素の作業域および定数を定義する。行および列は128×256個のアレイデータメモリ要素を指定し，アドレスは各要素内における作業域および定数の先頭アドレスを示す。
END	プログラムの終了を示す。

表5　記号を定義する命令

表　記	意　味
記号　EQ　数値	記号の値を設定する。

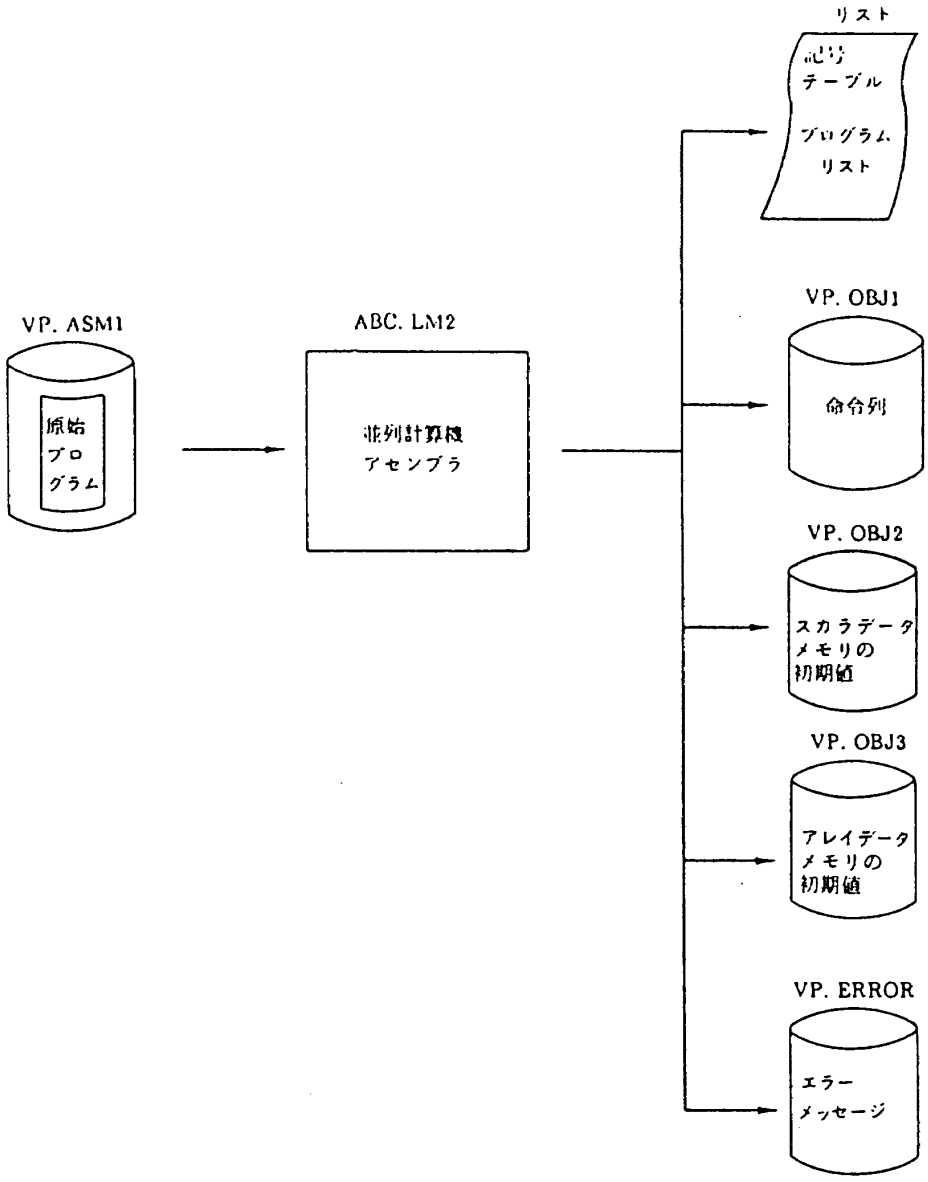
表6　作業域を確保し定数を生成する命令

表　記	意　味
記号　BS　語数	作業域を指定した語数だけ確保する。
記号　DC　数値	定数を生成する。

4.2 特徴

このアセンブラは並列計算機の構成に適合するため以下の特徴を有する。

(1) 制御処理部およびデータ処理部の命令列を一時にアセンブルする。通常の計算機は命令解読部が一個であるため複数の命令列を扱うことは無い。これに対して制御処理部とデータ処理部は独立した命令解読部を各々有し，単独で処理を行い得るため両処理部の命令列を生成する必要がある。ただし，制御処理部がデータ処理部を起動する際に制御処理部からデータ処理部に対して先頭命令アドレスが渡される。このため両者の命令列を一



アセンブラのジョブ制御文例

```
//VPASM  EXEC PGM=ASO,PARM='###.VP.OBJ3',REGION=1024K
//STEPLIB DD DSN=###.ABC.LM2,DISP=SHR
//FT01F001 DD DSN=###.VP.ASM1,DISP=SHR
//FT07F001 DD SYSOUT=*,DCB=(RECFM=VBA,LRECL=137,BLKSIZE=141)
//FT08F001 DD SYSOUT=*,DCB=(RECFM=VBA,LRECL=137,BLKSIZE=141)
//FT09F001 DD DSN=###.VP.OBJ1,DISP=SHR
//FT10F001 DD DSN=###.VP.OBJ2,DISP=SHR
//FT11F001 DD DYNAM
//FT12F001 DD DSN=###.VP.ERROR,DISP=SHR
//
```

図14 アセンブラの入出力ファイル

時にアセンブルする必要がある。

(2) 並列計算機はインストラクションメモリ，スカラデータメモリ，アレイデータメモリを有するためコントロール命令のアドレス部はインストラクションメモリのアドレス，スカラ演算命令のアドレス部はスカラデータメモリのアドレス，アレイ演算命令のアドレス部はアレイデータメモリのアドレスを指す。

(3) 機械語の命令とデータに変換されたプログラムは命令列とスカラデータの初期値およびアレイデータの初期値として分割されて出力される。特にアレイデータはアレイデータメモリ要素ごとに区分される。

4.3 アセンブラの実行

このアセンブラはFACOM-M780で動作する様に作られていて起動する時のジョブ制御文と入出力ファイルの関係を図14に示す。

記号命令で書かれた原始プログラムはファイル“VP. ASM1”に格納されている。アセンブラはこれを並列計算機の機械語に変換し，命令列をファイル“VP. OBJ1”に，スカラデータ初期値をファイル“VP. OBJ2”に，アレイデータ初期値を各アレイデータメモリ要素ごとにファイル“VP.OBJ3”に区分データセットとして出力する。

これらのファイルは並列計算機の動作を忠実に模擬するシミュレータの入力となる。

またリストは次の2種類よりなる。

(1) 記号テーブル（図15）

原始プログラムの中で使用した記号の値および値が確定した時の文の位置を示す。例では左側の記号が中央の文番号で確定し(DEFN)，その値(VALUE)を16進数で右側に示す。

(2) プログラム・リスト（図16）

これは原始プログラムとそれに対応する機械語を示すリストで，左よりアドレス欄(LOC)，機械語欄(OBJECT CODE)，文番号欄(STMT)，原始プログラム欄(SOURCE STATEMENT)で構成されている。機械語は全て64ビットでありアドレス欄はバイトアドレスで機械語欄と共に16進数で示される。

さらに，この図のプログラムは

(1) EQ 命令による記号定義
(2) 制御処理部のプログラム
（SC 命令～END 命令まで）
(3) データ処理部のプログラム
（AC 命令～END 命令まで）
(4) スカラデータメモリの初期値
（SP 命令～END 命令まで）
(5) アレイデータメモリ要素の初期値
（AP 命令～END 命令まで）

のサブプログラムに分かれておりアレイデータメモリ要素についても要素(0, 0)と要素(127, 255)についてだけ示してある。また並列計算機には3種類のメモリと3種類の演算器があるため，各サブプログラムごとに上記の意味が異なりその詳細を表7に示す。

SYMBOL	DEFN	VALUE
CONST1	158	00000000
CONST2	159	00000001
CONST3	160	00000002
C000	3	00000000
C001	4	00000001
C002	5	00000002
C004	6	00000004
C008	7	00000008
C016	8	00000010
C032	9	00000020
C064	10	00000040
C128	11	00000080
MINS	12	00000002

図15　記号テーブル

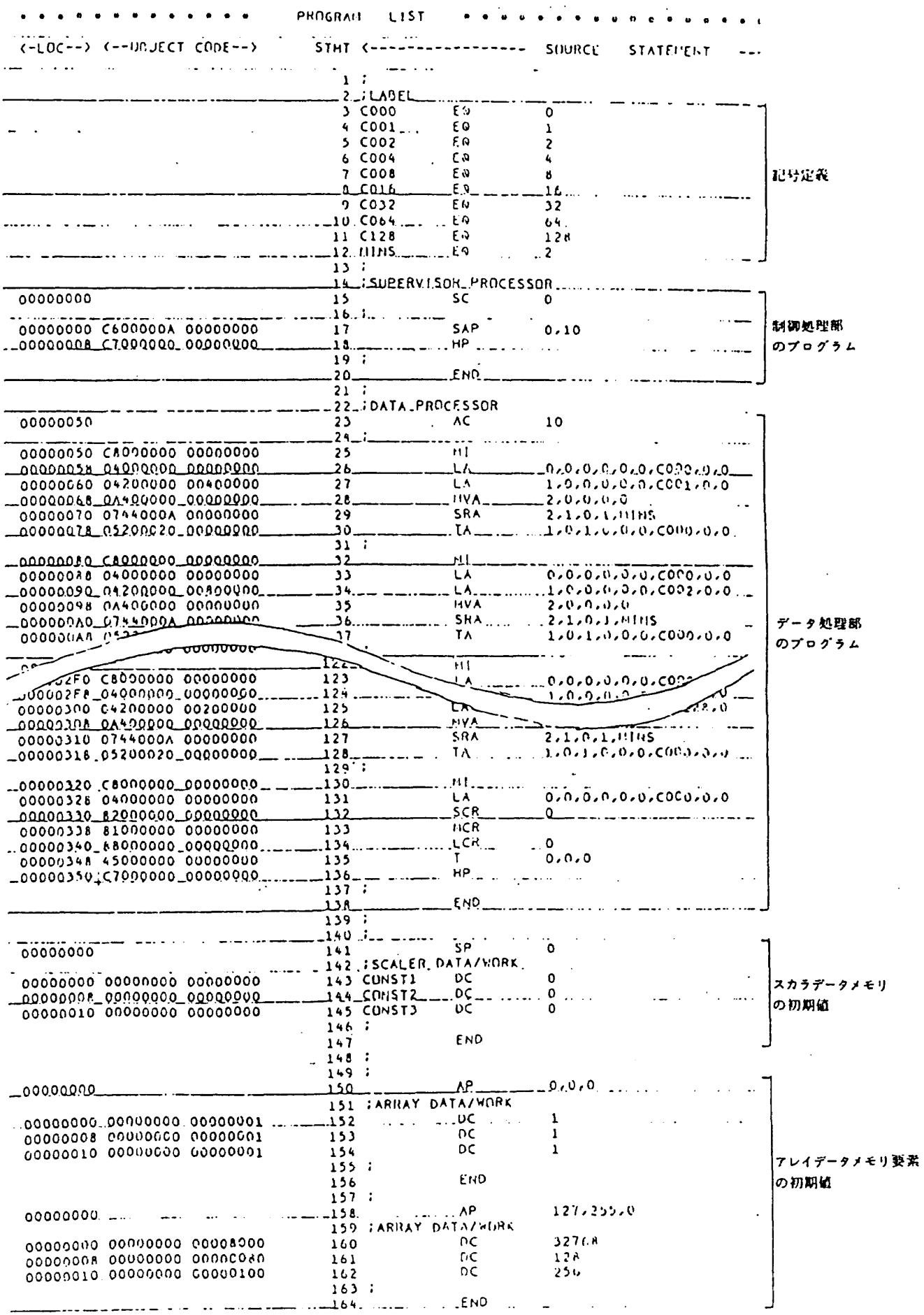


図16 プログラムリスト

表7　プログラム欄での意味

サブプログラム	アドレス欄	機械語欄	文番号欄	原始プログラム欄
記号定義			原始プログラムの文番号を示す。	ラベルの記号をオペランドの値とする。
制御処理部のプログラム	命令の入るインストラクションメモリのバイトアドレス（16進数）を示す。	1命令を8バイトで構成した機械語を16進で示す。	同　上	○スカラ演算命令のレジスタオペランドは制御データ演算器のレジスタを，アドレスオペランドはスカラデータメモリのアドレスを示す。 ○コントロール命令のレジスタオペランドは制御データ演算器のレジスタを，アドレスオペランドはインストラクションメモリのアドレスを示す。 ○会話命令のレジスタオペランドは制御データ演算器のレジスタを示す。
データ処理部のプログラム	命令の入るインストラクションメモリのバイトアドレス（16進数）を示す。	1命令を8バイトで構成した機械語を16進で示す。	同　上	○スカラ演算命令のレジスタオペランドはスカラデータ演算器のレジスタを，アドレスオペランドはスカラデータメモリのアドレスを示す。 ○アレイ演算命令のレジスタオペランドは全アレイデータ演算器要素のレジスタを，アドレスオペランドは全アレイデータメモリ要素のアドレスを示す。 ○コントロール命令のレジスタオペランドはスカラ演算器のレジスタを，アドレスオペランドはインストラクションメモリのアドレスを示す。 ○会話命令のレジスタオペランドはスカラ演算器および全アレイデータ演算器要素のレジスタを示す。
スカラデータメモリの初期値	スカラデータメモリのバイトアドレス（16進数）を示す。	1データを8バイトで構成した機械語を16進数で示す。	同　上	スカラデータメモリの初期値（DC命令）および作業域（BS命令）を示す。
アレイデータメモリ要素の初期値	アレイデータメモリ要素のバイトアドレス（16進数）を示す。	同　上	同　上	アレイデータメモリ要素の初期値（DC命令）および作業域（BS命令）を示す。 AP命令ごとに区分データセットとして生成される。

5. シミュレータの動作

並列計算機の各部動作・効率を検討するため，このシミュレータは命令を処理する手順をタイミングごとに忠実に追跡するように作成されており演算結果をも得ることができる。

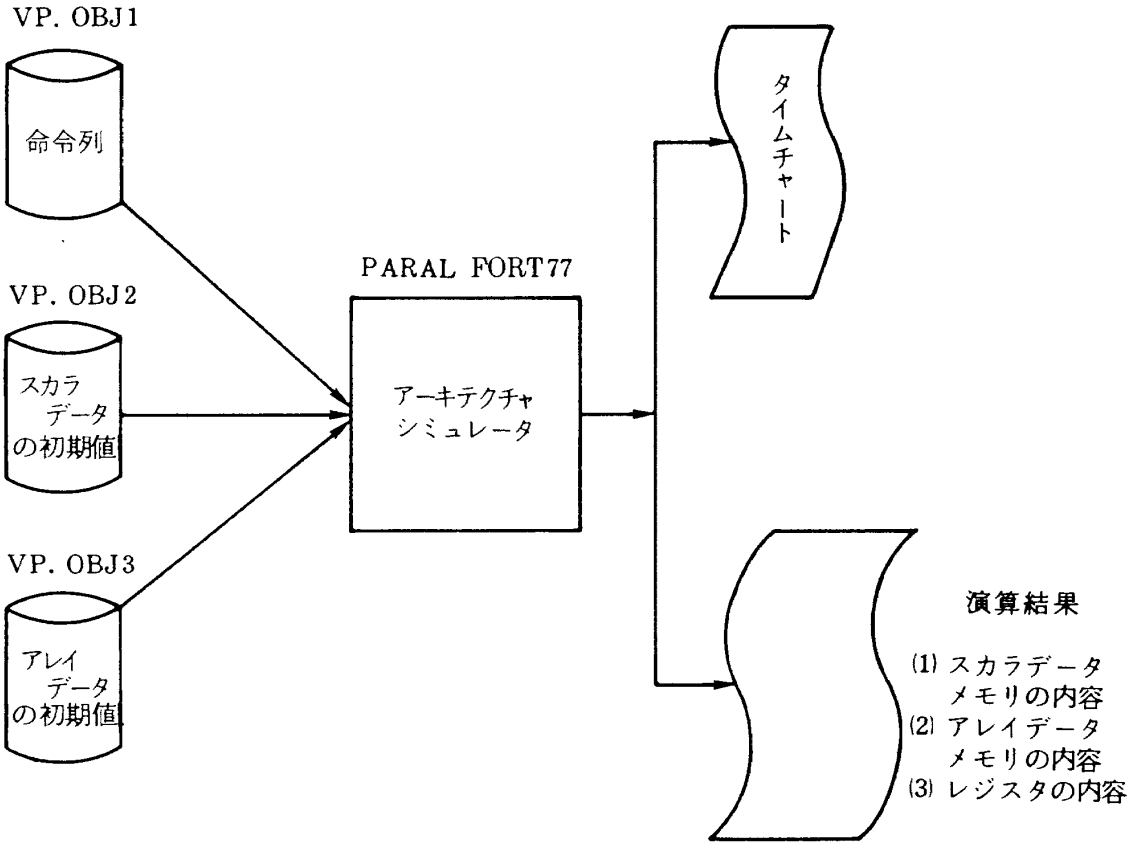
図17にその入出力ファイルの関係を，図18にシミュレータの動作概略を示す。

シミュレータが起動されると命令列，スカラ/アレイデータの初期値が読み込まれた後，制御処理部が起動される。以後，1クロックずつ時刻を進ませ，それに応じて制御処理部の命令が解読・実行され，演算器・メモリ等の間でデータが移動・加工される状況が模擬される。

制御処理部がデータ処理部起動命令(SAP)を解読するとアドレス部から求めたデータ処理部命令の先頭アドレスをデータ処理部に送り起動する。以後，両処理部は独立して処理を進め，各処理部の命令が解読・実行される。そして各自の命令列に停止命令(HP)を解読するとその部を停止し，両処理部が停止した時点で模擬が終了しタイムチャートと演算結果が出力される。

6. 命令追跡による検証模擬

最大値を求めるプログラムと二次元の円筒まわりの流れを解析するプログラムを用いてシミュレータでプログラム命令の追跡を行った。



アーキテクチャシミュレータのジョブ制御文例

```
// EXEC FORTC,PARAM='NOPRINT',SF='XXX.PARAL.FORT77'
// EXEC LIED
// EXEC GO
// EXPAND USDK,RNO=10,FILE='XXX.VP.OBJ1'
// EXPAND USDK,RNO=11,FILE='XXX.VP.OBJ2'
// EXPAND USDK,RNO=12,FILE='XXX.VP.OBJ3(AP010100)'
//
```

図17 シミュレータの入出力

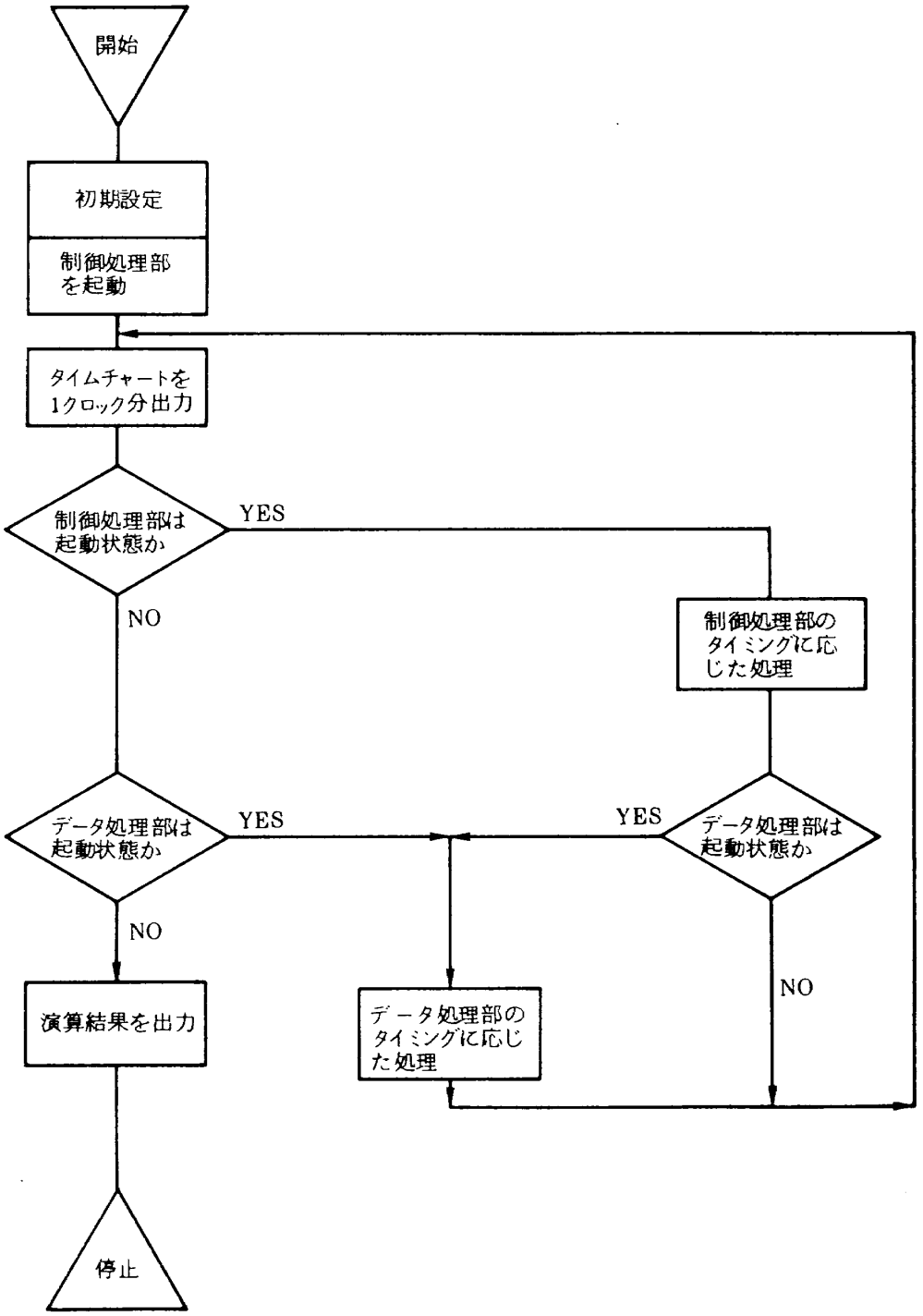


図18　シミュレータの動作概略

6.1　アレイデータメモリ要素内データの最大値を求めるプログラムの場合[16)]

アレイデータを

〔Ai, j, x ; i=1～128, j=1～256, x=1～128〕

としiおよびjに各アレイデータメモリ要素を対応させ同一メモリ要素にx個のデータを格納させるとその手順は次のようになる。

① 制御処理部よりデータ処理部を起動する。各アレイデータメモリ要素内の最大値を求める。これを〔Bi, j ; i=1～128, j=1～256〕とする。

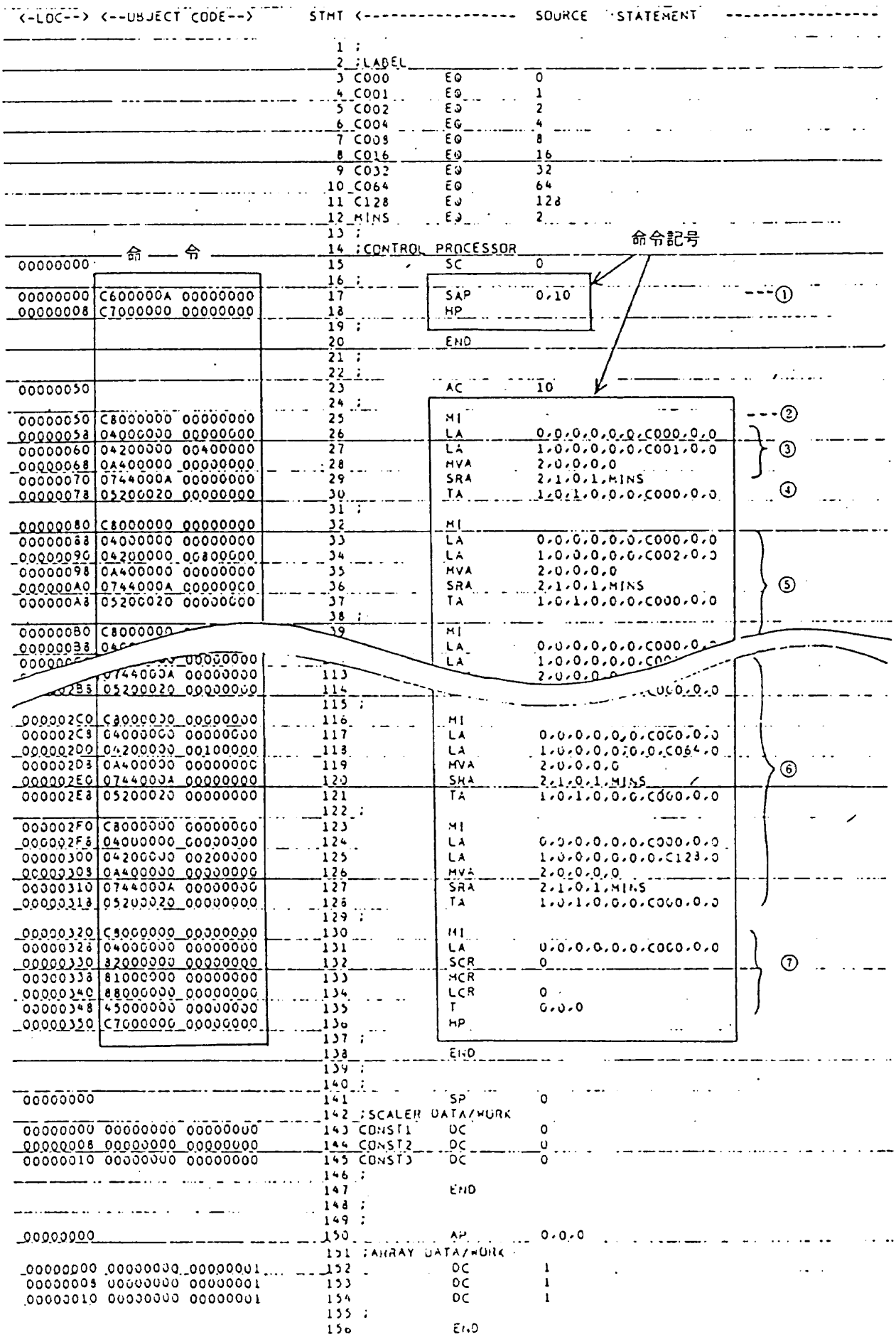
② マスクを払う。

③ k=0としてBi, jとBi+2^k, jとを比較し，前者が小さければマスクを設定する。

④ マスクが設定された要素だけを大きいBi+2^k, jで置き換える。

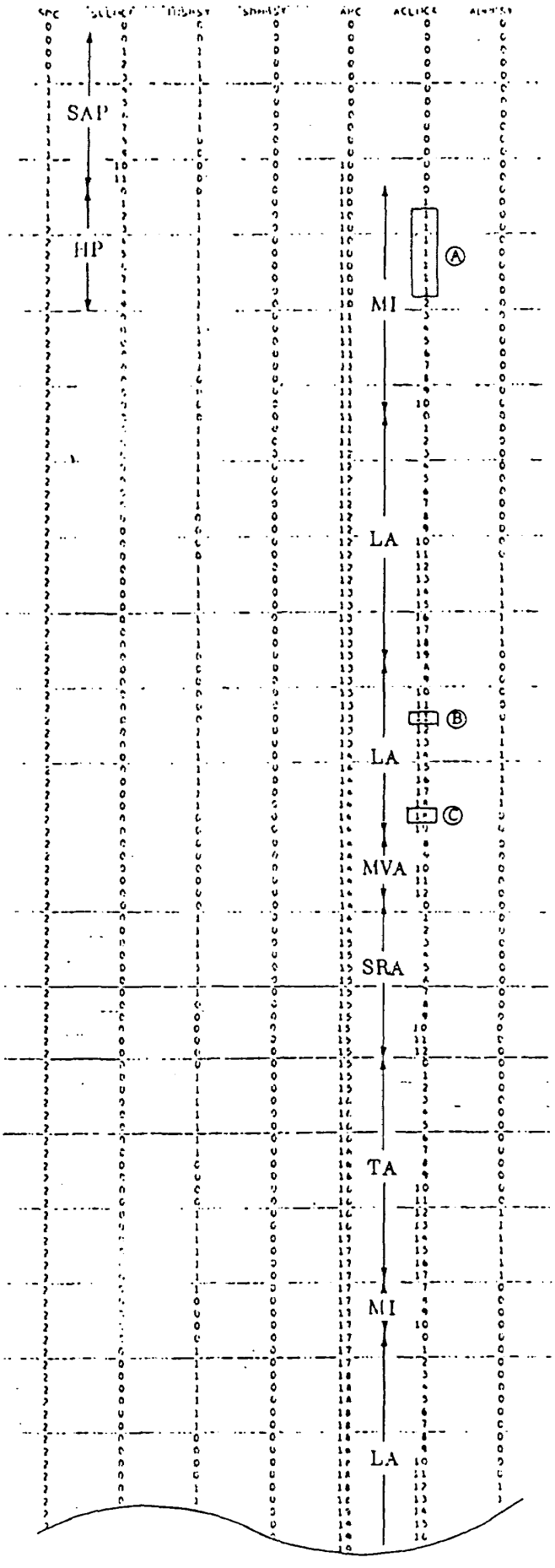
⑤ k=1～6について②～④を繰り返すと

B1, j～B128, j




```
<-LOC--> <--OBJECT CODE-->      STMT <---------------- SOURCE STATEMENT ----------------
                                   1 ;
                                   2 ;LABEL
                                   3 C000      EQ      0
                                   4 C001      EQ      1
                                   5 C002      EQ      2
                                   6 C004      EQ      4
                                   7 C008      EQ      8
                                   8 C016      EQ      16
                                   9 C032      EQ      32
                                  10 C064      EQ      64
                                  11 C128      EQ      128
                                  12 MINS      EQ      2
                                  13 ;
                                  14 ;CONTROL PROCESSOR
00000000                          15           SC      0
                                  16 ;
00000000 C600000A 00000000        17           SAP     0,10
00000008 C7000000 00000000        18           HP
                                  19 ;
                                  20           END
                                  21 ;
                                  22 ;
00000050                          23           AC      10
                                  24 ;
00000050 C8000000 00000000        25           MI
00000058 04000000 00000000        26           LA      0,0,0,0,0,0,C000,0,0
00000060 04200000 00400000        27           LA      1,0,0,0,0,0,C001,0,0
00000068 0A400000 00000000        28           MVA     2,0,0,0,0
00000070 0744000A 00000000        29           SRA     2,1,0,1,MINS
00000078 05200020 00000000        30           TA      1,0,1,0,0,0,C000,0,0
                                  31 ;
00000080 C8000000 00000000        32           MI
00000088 04000000 00000000        33           LA      0,0,0,0,0,0,C000,0,0
00000090 04200000 00800000        34           LA      1,0,0,0,0,0,C002,0,0
00000098 0A400000 00000000        35           MVA     2,0,0,0,0
000000A0 0744000A 00000000        36           SRA     2,1,0,1,MINS
000000A8 05200020 00000000        37           TA      1,0,1,0,0,0,C000,0,0
                                  38 ;
000000B0 C8000000                 39           MI
00000038 04                                    LA      0,0,0,0,0,0,C000,0,0
000000                                         LA      1,0,0,0,0,0,C00
                  0744000A 00000000  113                2,0,0,0,0
                  05200020 00000000  114                                C000,0,0
                                     115 ;
000002C0 C8000000 00000000  116           MI
000002C8 04000000 00000000  117           LA      0,0,0,0,0,0,C000,0,0
000002D0 04200000 00100000  118           LA      1,0,0,0,0,0,C064,0,0
000002D8 0A400000 0000000C  119           MVA     2,0,0,0,0
000002E0 0744000A 00000000  120           SRA     2,1,0,1,MINS
000002E8 05200020 00000000  121           TA      1,0,1,0,0,0,C000,0,0
                            122 ;
000002F0 C8000000 00000000  123           MI
000002F8 04000000 00000000  124           LA      0,0,0,0,0,0,C000,0,0
00000300 04200000 00200000  125           LA      1,0,0,0,0,0,C128,0,0
00000308 0A400000 00000000  126           MVA     2,0,0,0,0
00000310 0744000A 00000000  127           SRA     2,1,0,1,MINS
00000318 05200020 00000000  128           TA      1,0,1,0,0,0,C000,0,0
                            129 ;
00000320 C9000000 00000000  130           MI
00000328 04000000 00000000  131           LA      0,0,0,0,0,0,C000,0,0
00000330 82000000 00000000  132           SCR     0
00000338 81000000 00000000  133           MCR
00000340 88000000 00000000  134           LCR     0
00000348 45000000 00000000  135           T       0,0,0
00000350 C7000000 00000000  136           HP
                            137 ;
                            138           END
                            139 ;
                            140 ;
00000000                    141           SP      0
                            142 ;SCALER DATA/WORK
00000000 00000000 00000000  143 CONST1    DC      0
00000008 00000000 00000000  144 CONST2    DC      0
00000010 00000000 00000000  145 CONST3    DC      0
                            146 ;
                            147           END
                            148 ;
                            149 ;
00000000                    150           AP      0,0,0
                            151 ;ARRAY DATA/WORK
00000000 00000000 00000001  152           DC      1
00000008 00000000 00000001  153           DC      1
00000010 00000000 00000001  154           DC      1
                            155 ;
                            156           END
```

図19 最大値を求めるプログラムリスト



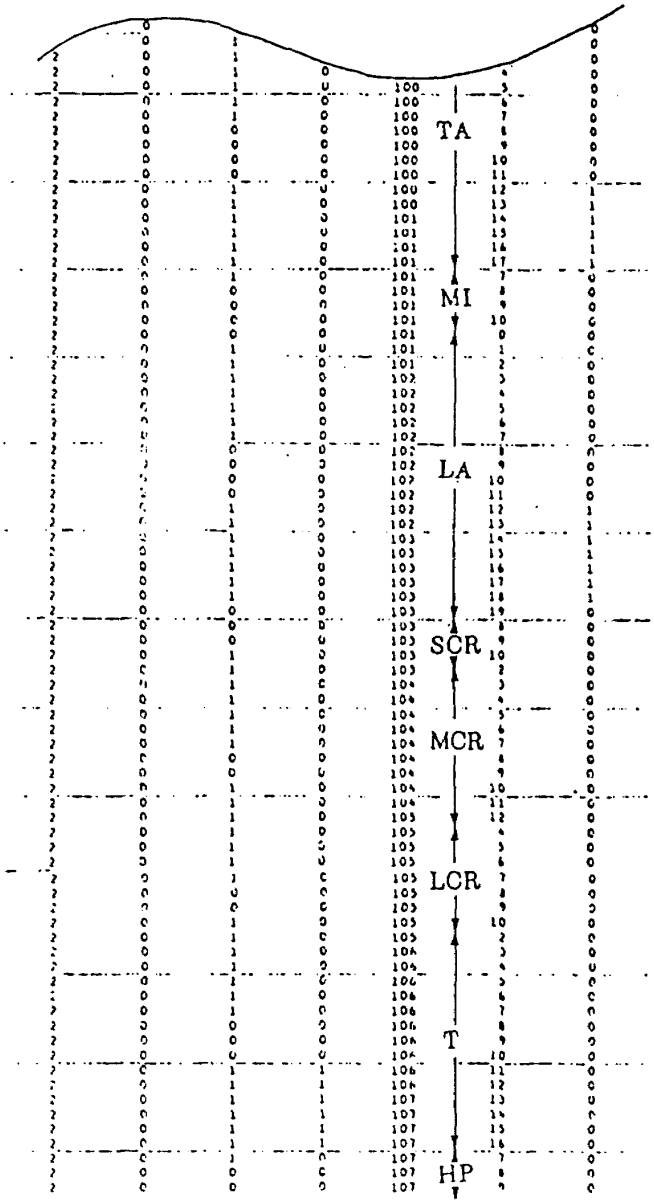


図20　最大値を求めるタイムチャート

の最大値が列 j の全要素に求まる。

⑥ 同様の操作を列間で行うと全要素に全体の最大値が求まる。

⑦ この最大値をスカラデータ演算器に移しスカラデータメモリに格納する。

各アレイデータメモリ要素内の最大値を求める部分については後述するので，各要素内の最大値 (Bi, j) から全体の最大値を求めるプログラム例を①～⑦に対応して図19に，そのタイムチャートを図20に示す。

また，全体をシミュレータで模擬した時のクロック数をスカラデータ演算器による場合とアレイデータ演算器による場合とで比較し，演算器数で割って処理効率とし図21に白丸で示す。ここで処理効率1はスカラ演算器に比べ128×256倍の処理性能であることを示す。

この図を観察すると，(1)データ数が増加するにつれ効率は向上するがそれも0.6程度が限度であり，(2)データ数の小さい所での効率が極端に低いことがわかる。

(1)の原因は各アレイデータメモリ要素内の最大値を求める手順に依存している。処理手順をスカラ演算命令とアレイ演算命令で比較すると図22の様になり，処理効率が約0.6になる理由は明白である。アレイ演算命令として新たに命令を合成（図23のCMPA：MVA, MVA：ICA命令）してやると回路の動作からCMPA, MVA命令に各2クロックを追加するだけで済むため図21の三角印の線まで効率向上が期待できる。なお図23に示すようにアレイデータメモリ要素の容量に制限が無い場合には最大効率は0.89に成り得る。

(2)の原因は②～⑦の処理にあり，ネットワークの改善以外に効率向上策は無い。命令系の見直しに加えてデータがネットワークを移動している時間を全て取り去った時の効率を推定すると図21の×印の様になる。たしかにそれなりの改善は見られるがデータ数が多くなるにつれ，その効果は薄れて行く。

従来，並列計算機についてはネットワークをことさら重要視した議論がなされてきたが，むしろマスク操作に関する命令系の整合性がより重要であることを示唆し，メモリ要素内のデータの多い三次元問題を扱う場合には考慮しておくべき事項であろう。

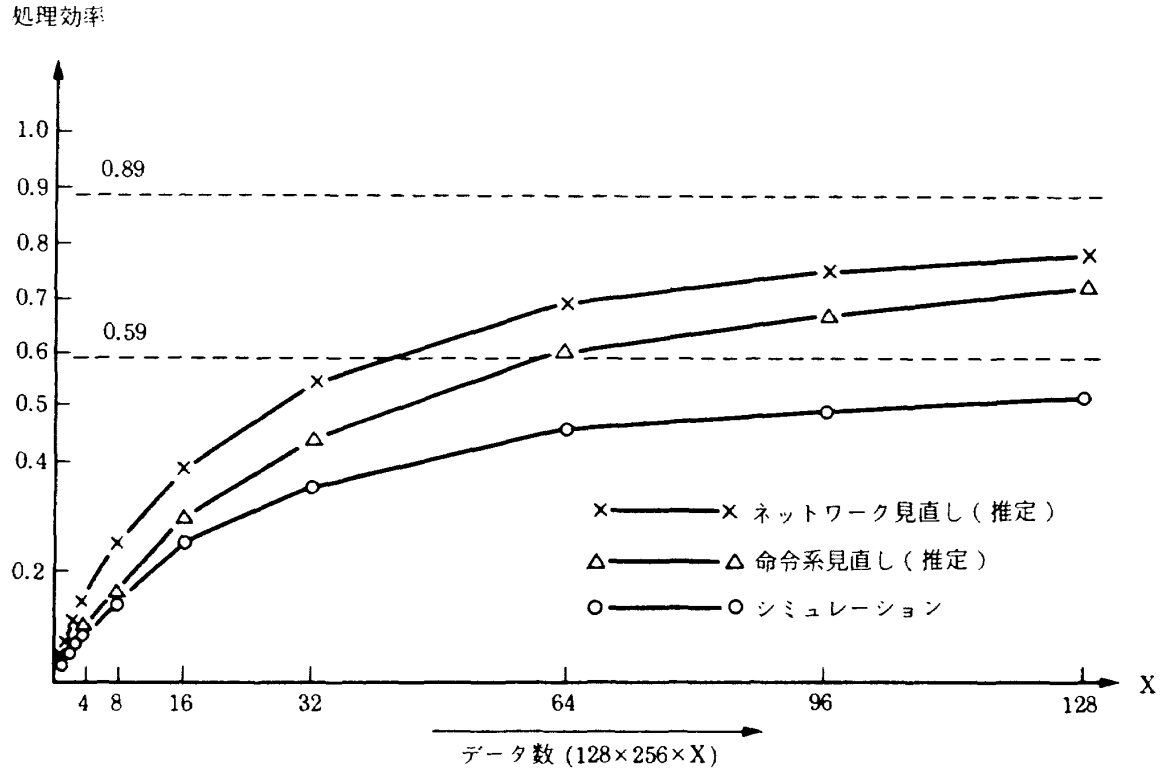


図21 処理効率の推定

スカラ演算命令の場合

```
AA  L    1,7,0
    CMP  1,0,6
    J    0,AB  ──→  AB  MV  0,1,0
AC  IC   7,0   ←──      MV  2,7,0
    IC   6,ZERO ←──     J   0,AC
    J    0,AA
```

58 ～ 104 Clock

アレイ演算命令の場合

```
AA  MI                                マスクリセット
    LA     3,7,0,0,0,0,0,0,0,1         次の値を取り出す
    CMPA   0,3,0,1,MINS                現最大値より大なら
                                         マスクセット
    MVA    2,7,1,0,0                   最大値の指標を更新
    MVA    0,3,1,0,0                   最大値を更新
    ICA    7,0,0,0                     指標を更新
    IC     0,ZERO                      カウント
    J      0,AA                        ジャンプ
```

98 Clock

58／98 ＝ 0．59

R0:現最大値，　R2:現最大値の指標，　R6:カウンタ，　R7:指標

図22　処理手順の比較

```
AA  MI
    LA     3,7,0,0,0,0,0,0,0,1  *
    CMPA   0,3,0,1,MINS

    MVA    2,7,1,0,0            *
    MVA    0,3,1,0,0            *
    ICA    7,0,0,0
    IC     0,ZERO
    J      0,AA
```

98 Clock

```
AA  LA     3,7,0,MI,0,0,0,0,0,1
    CMPA   0,3,0,1,MINS: MVA   2,7,1,0,0

    MVA    0,3,1,0,0    : ICA   7,0,0,0

    IC     0,ZERO
    J      0,AA
```

65 Clock

58／65 ＝ 0．89

図23　命令系の見直し

6.2　二次元円筒まわりの流れ場解析プログラムの場合

時刻 t=0 で一様流が円柱まわりに発生し，以後各時刻ごとに流れ場の渦度・流れ関数の計算，表面渦度の補正，表面圧力と抗力・揚力の計算を繰り返して二次元非定常流を追跡する解析プログラムである。図24に FACOM-M780での計算例を示す。

これを並列計算機に適合させるため，円柱まわりを256等分し半径方向 $(1\sim\exp(\Pi))$ を等比間隔になるように128分割し，各交点を計算のための格子点とした。また，時間の刻み幅を0.0025とした。

なお，このプログラムは文献15をもとに対称の条件をはずしたものである[15),17)]。

(1)　並列計算機への移植

解析プログラムを並列計算機に適合させる際に遭遇した事項について記す。また図25に移植した流れ関数の計算部を示す。

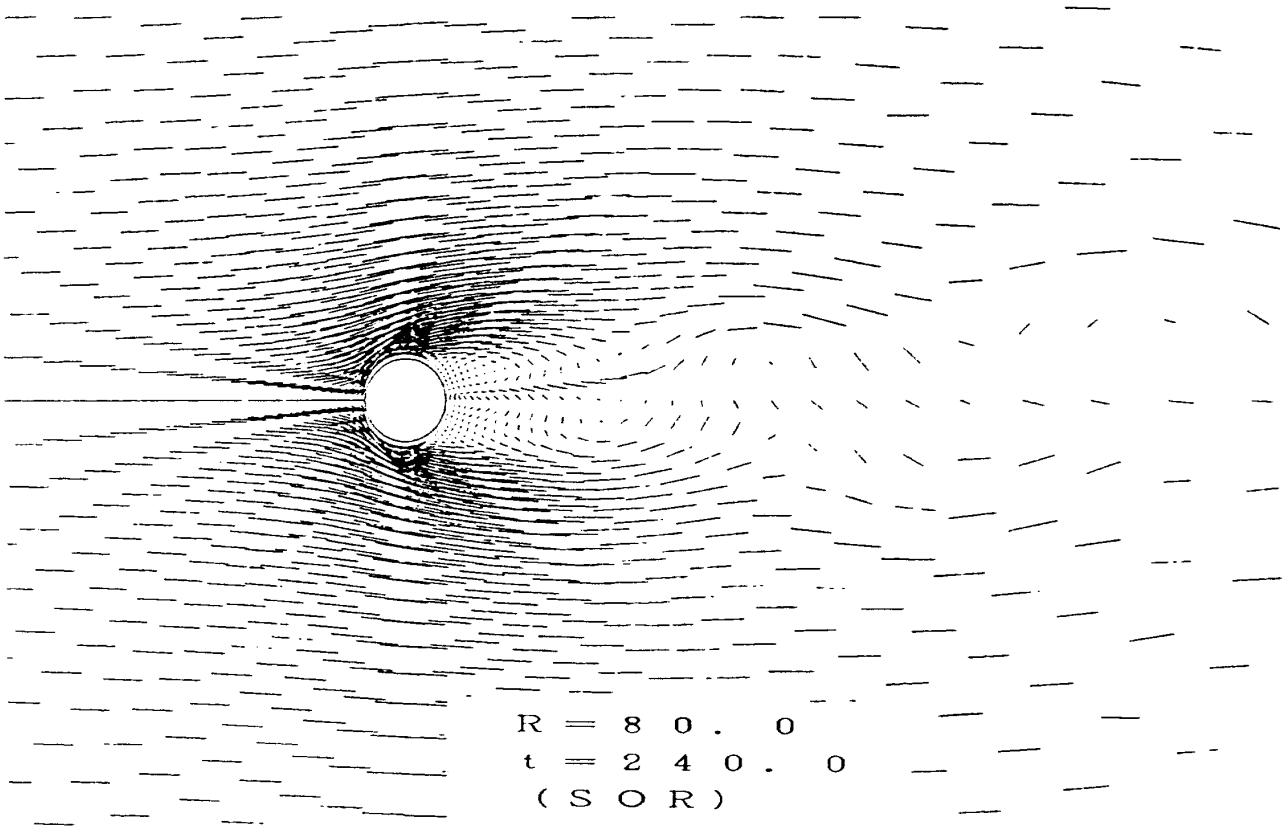


図24 計算例

① 格子と演算器の対応

格子と演算器・記憶装置の対応付けは並列処理に適合させるための基本的条件であり，単に格子の分解能ばかりで無く反復法や反復数に与える影響が大きい。

2次元の格子を演算器・記憶装置に1対1で対応させると反復法では“旧値のみ”から新値を求めることになりヤコビ法を用いることになる。この時，加速パラメータを用いることができない。また一方向に隣接する2格子，二方向に隣接する4格子を対応させると演算器を遊ばせることなくゼブラ法，偶奇法を用いることができる。図26に128×256格子点において逐次処理用のSOR法，並列処理用のヤコビ法・ゼブラ法を用い収束条件を10^{-5}以下とした時の反復回数を示す。各々の加速パラメータは1.95, 1.00, 1.32である。

ヤコビ法・ゼブラ法ではSOR法に比べ初期の反復数が極端に多い。しかし以後，急激・単調に減少し，ついにはSOR法よりも少なくなる。時刻をどこまで進めれば流れが落ち着くか定かでないが，計算法別に反復回数が全部で多くなるかどうかは速断できないようである。

② 並列処理とマスク操作

流れ関数の反復計算時における各格子点での変数の修正量が収束誤差以内であるかを判定する場合，逐次処理では各点の修正量を条件と比較し，次の格子点の変数を検査することになるのに対し並列処理では各格子点の収束判定が同時にできる。ただし全格子点が収束条件を満たし反復計算を打ち切るかどうかを判定するためには各演算器の判定信号を一カ所で監視・判定する必要がある。

図27にその回路の概念を示す。ここでは各演算器にマスクMK (k, l) を設定しておき収束条件を満たさない時には払う。マスクが払われた演算器のCMNCR 3/6 (k, l) の内容がCMNCR2/5に送られ，その内容が“空”であれば収束と判定できる。

さらに，表面圧力と抗力・揚力の計算には円柱に沿って積分する必要があり逐次処理ではDOループで累積すれば済むのに対し並列処理ではシフトとマスク操作による演算拒諾の制御が必要である。図28にその様子を示す。図中“*”はマスクによる演算制御を示す。

```
                                679 ;
                                680 ;C**************************************
                                681 ;C
                                682 ;          DO 1700 L = 1 , 9999
                                683 ;C
                                684 ;          DO 1800 I=2,127
                                685 ;          DELTA( I,1 ) = ( PSI( I+1,1 ) + PSI( I-1,1 )
                                686 ;     1    + PSI( I,2 ) + PSI( I,256 ) )/4
                                687 ;     1    + EE( I ) * ZETA( I,1 ) - PSI( I,1 )
                                688 ;          DELTA( I,257 ) = DELTA( I,1 )
                                689 ;          LPSI( I,1 ) = PSI( I,1 ) + CONST * DELTA( I,1 )
                                690 ;          LPSI( I,257 ) = LPSI( I,1 )
                                691 ;C
                                692 ;          DO 1800 J= 2 , 256
                                693 ;          DELTA( I,J ) = ( PSI( I+1,J ) + PSI( I-1,J )
                                694 ;     1    + PSI( I,J+1 ) + PSI( I,J-1 ) ) / 4
                                695 ;     1    + EE( I ) * ZETA( I,J ) - PSI( I,J )
                                696 ;          LPSI( I,J ) = PSI( I,J ) + CONST * DELTA( I,J )
                                697 ;1800     CONTINUE
                                698 ;C
                                699 ;          DO 1730 I= 2 , 127
                                700 ;          DO 1730 J= 2 , 256
                                701 ;          PSI( I,J ) = LPSI( I,J )
                                702 ;1730     CONTINUE
                                703 ;C********C*********C
                                704 ;
00000C98 44000000 00000000      705           L         0,0,K00000
00000CA0 45000003 00000000      706           T         0,0,L
                                707 ;
00000CA8 44000003 00000000      708 A1800     L         0,0,L
00000CB0 50000000 00000000      709           IC        0,EMPT
00000CB8 45000003 00000000      710           T         0,0,L
00000CC0 44200001 00000000      711           L         1,0,K01001
00000CC8 47040002 00000000      712           SR        0,1,MINS
00000CD0 C0000189 00000000      713           J         0,A1750
                                714 ;
00000CD8 C8000000 00000000      715           MI
00000CE0 04200000 0000001D      716           LA        1,0,ECNO,MONO,EMPT,0,C000,C000,1
00000CE8 0120000C 00000005      717           SA        1,0,ECNO,MOST,ZERO,0,C000,C000,K00001
00000CF0 0120000C 00000007      718           SA        1,0,ECNO,MOST,ZERO,0,C000,C000,K00127
00000CF8 24000000 00400029      719           FLA       0,0,ECNO,MONO,EMPT,0,C001,C000,PSI
00000D00 20000000 3FC00029      720           FAA       0,0,ECNO,MONO,EMPT,0,M001,C000,PSI
00000D08 20000000 00004029      721           FAA       0,0,ECNO,MONO,EMPT,0,C000,C001,PSI
00000D10 20000000 003FC029      722           FAA       0,0,ECNO,MONO,EMPT,0,C000,M001,PSI
00000D18 23000000 0000000D      723           FDA       0,0,ECNO,MONO,EMPT,0,C000,C000,KF0004
00000D20 21000000 00000029      724           FSA       0,0,ECNO,MONO,EMPT,0,C000,C000,PSI
00000D28 24200000 00000026      725           FLA       1,0,ECNO,MONO,EMPT,0,C000,C000,EE
00000D30 22200000 0000002B      726           FMA       1,0,ECNO,MONO,EMPT,0,C000,C000,AZETA
00000D38 26040000 00000000      727           FARA      0,1,ECNO,MONO,EMPT
00000D40 25000040 0000002C      728           FTA       0,0,ECRS,MONO,EMPT,0,C000,C000,DELTA
00000D48 22000000 00000001      729           FMA       0,0,ECNO,MONO,EMPT,0,C000,C000,CONST
00000D50 20000000 00000029      730           FAA       0,0,ECNO,MONO,EMPT,0,C000,C000,PSI
00000D58 25000040 00000029      731           FTA       0,0,ECRS,MONO,EMPT,0,C000,C000,PSI
                                732 ;
                                733 ;C**************************************
                                734 ;C
                                735 ;          DO 1770 I = 1 , 127
                                736 ;          DO 1770 J = 1 , 256
                                737 ;          X1 = CONST * ABS( DELTA( I , J ) )
                                738 ;          IF( X1 .GE. HOLD ) GO TO 1700
                                739 ;1770     CONTINUE
                                740 ;          GO TO 1750
                                741 ;1700     CONTINUE
                                742 ;1750     COUNT( K ) = L
                                743 ;C********C*********C
                                744 ;
00000D60 C8000000 00000000      745           MI
00000D68 2400000A 0000002C      746 A1770     FLA       0,0,ECNO,MOST,MINS,0,C000,C000,DELTA
00000D70 2E000020 00000000      747           FLNA      0,0,ECST,MONO,EMPT                    ;ABS
00000D78 22000000 00000001      748           FMA       0,0,ECNO,MONO,EMPT,0,C000,C000,CONST
                                749 ;
00000D80 2724000C 00000000      750           FSRA      1,1,ECNO,MOST,ZERO                    ;ALL
00000D88 24200000 00000004      751           FLA       1,0,ECNO,MONO,EMPT,0,C000,C000,HOLD
00000D90 27200012 00000000      752           FSRA      1,0,ECNO,MORS,MINS
                                753 ;
00000D98 67240000 00000000      754           FSR       1,1,EMPT          ;F2(1)=0
00000DA0 83200000 00000000      755           FSCR      1                 ;CNMCR2/5=F2(1),CMNCR3/6(K,
00000DA8 81000000 00000000      756           NCR
00000DB0 89200000 00000000      757           FLCR      1                 ;F2(1)=CMNCR2/5 AND MASK OF
00000DB8 6A240004 00000000      758           FHV       1,1,ZERO          ;SKIP IF ALL MASK IS SET
00000DC0 C0000195 00000000      759           J         0,A1800
                                760 ;
00000DC8 44000003 00000000      761 A1750     L         0,0,L
00000DD0 45000004 00000000      762           T         0,0,COUNT
                                763 ;
                                764 ;C**************************************
```

流れ関数の計算

収束判定

図25　移植プログラム例

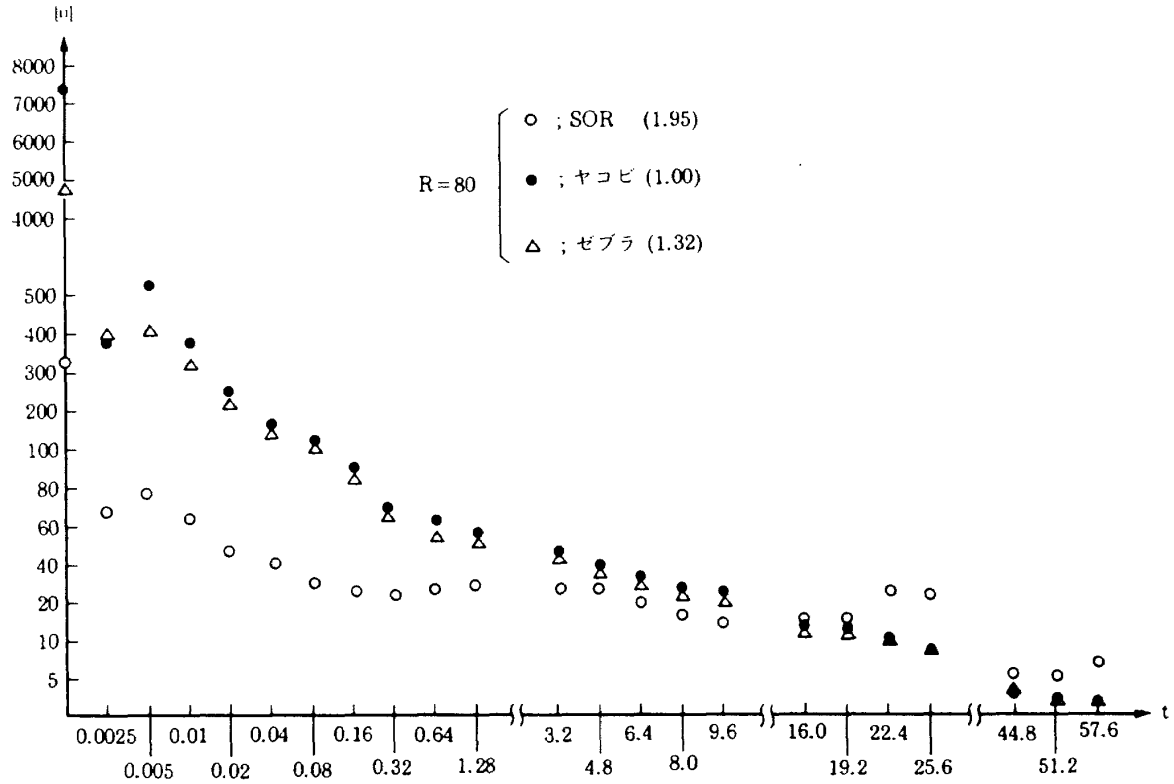


図26 反復回数の変化

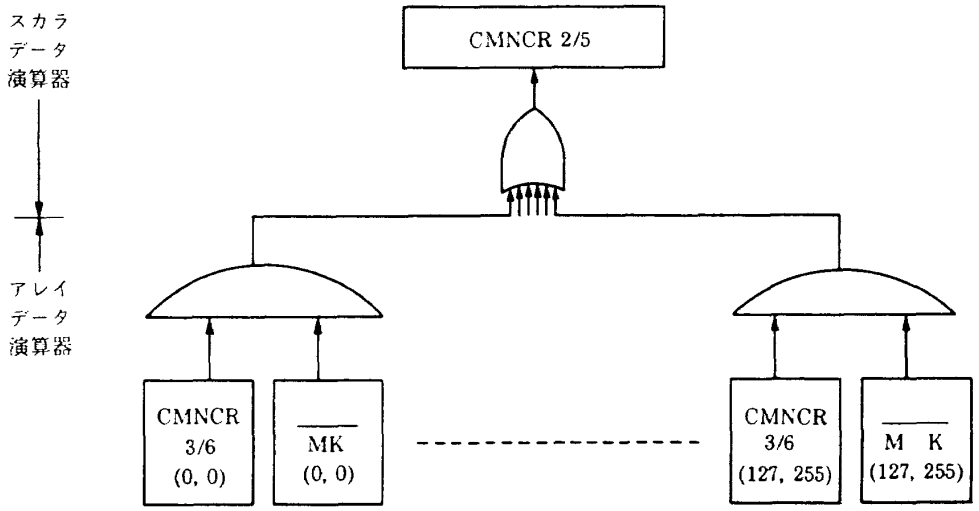


図27 収束の判定回路

③ 境界条件とマスク操作

この解析プログラムでは，渦度・流れ関数の計算時に上下左右の格子点の値を用いるため半径方向の先端（円柱表面）と終端（ここでは$\exp(\Pi)$）ではその値を更新しない。また渦度・流れ関数の計算後，円柱表面の渦度の補正を行う。このため，内側の格子点と境界のそれとでは計算手続きが異なり処理を拒諾するマスク操作が必要である。

(2) 考察

一台の演算器による逐次処理と128×256台の演算器による並列処理との相対的処理効率を見極めることを目標とし，解析プログラムの流れを渦度・流れ関数の計算，表面渦度の補正，表面圧力・抗力・揚力の計算に分割し各々について調べた（表8）。

以下に考察を加え，性能向上策について述べる。

$$p(\theta)_{r=1} = \frac{4}{R}\int_0^{\theta}\left(\frac{\partial\zeta}{\partial\xi}\right)_{\xi=0} d\theta + C$$

$$\begin{cases} C_{Dp} = -\dfrac{1}{2}\displaystyle\int_0^{2\pi} p(\theta)\cos\theta\, d\theta \\ C_{Df} = -\dfrac{2}{R}\displaystyle\int_0^{2\pi} \zeta_{\xi=0}\sin\theta\, d\theta \end{cases}$$

$$\begin{cases} C_{Lp} = -\dfrac{1}{2}\displaystyle\int_0^{2\pi} p(\theta)\sin\theta\, d\theta \\ C_{Lf} = \dfrac{2}{R}\displaystyle\int_0^{2\pi} \zeta_{\xi=0}\cos\theta\, d\theta \end{cases}$$

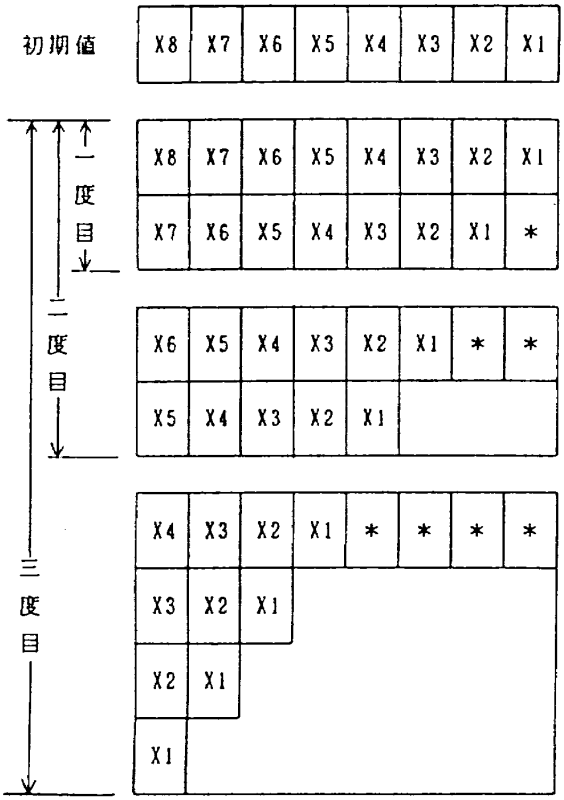


図28　表面圧力・抗力・揚力の計算

表8　処理時間（クロック）

処理内容	逐次処理	並列処理	比
渦度の計算	24,463,783	788	31,045
流れ関数の計算	11,152,098/回	543/回	20,537
表面渦度の補正	77,165	327	235
表面圧力の計算	70,229	1,475	47
抗力・揚力の計算	84,273	3,719	22

① 渦度の計算では演算器の台数（128×256台）に見合う性能が期待できる。

② 流れ関数の計算では期待の約6割しか性能が発揮されない。その主な原因は，この内容に収束判定の手続が含まれているためである。逐次処理では収束を満たさない格子点を発見すればただちに反復計算にもどることができ，全点を調べる必要がないのに対し並列処理ではアレイデータ演算器からスカラデータ演算器に集められた信号を基に収束判定する必要がありこれが処理時間の約4割を占める。したがって，より一層の性能向上を達成するためには全格子点での収束判定回路および命令系の最適化を図り，かつプログラム内での流れ関数本来の計算比重を増すことが考えられる。図26に示したように収束のための反復回数は始めは多く以後減少して行くことから初期には100回単位で収束判定を行い，以後前回の反復回数を参考に収束判定手続の頻度を可変にすることにより実効上の性能向上を図ることができると思われる。

③ 表面渦度の補正では円柱表面の割り当てられた格子点に対応する演算器のみ（256台）が動作しておりほぼ予想通りである。しかし，大部分の演算器が停止しているため計算法の改善が望まれる。

④ 表面圧力と抗力・揚力の計算は他の部分に比べ極端に効率が低い。その原因は円柱表面上の格子だけで演算が行われ，かつ円柱表面に沿って積分（累積）計算のデータ転送が行われ各々処理時間の約3割5分，5割5分を占めるためである。したがって，結合網を改良しても約70倍，50倍にしかならないことは注意を要する（付録4参照）。

性能向上策としてはこの結果が渦度・流れ関数

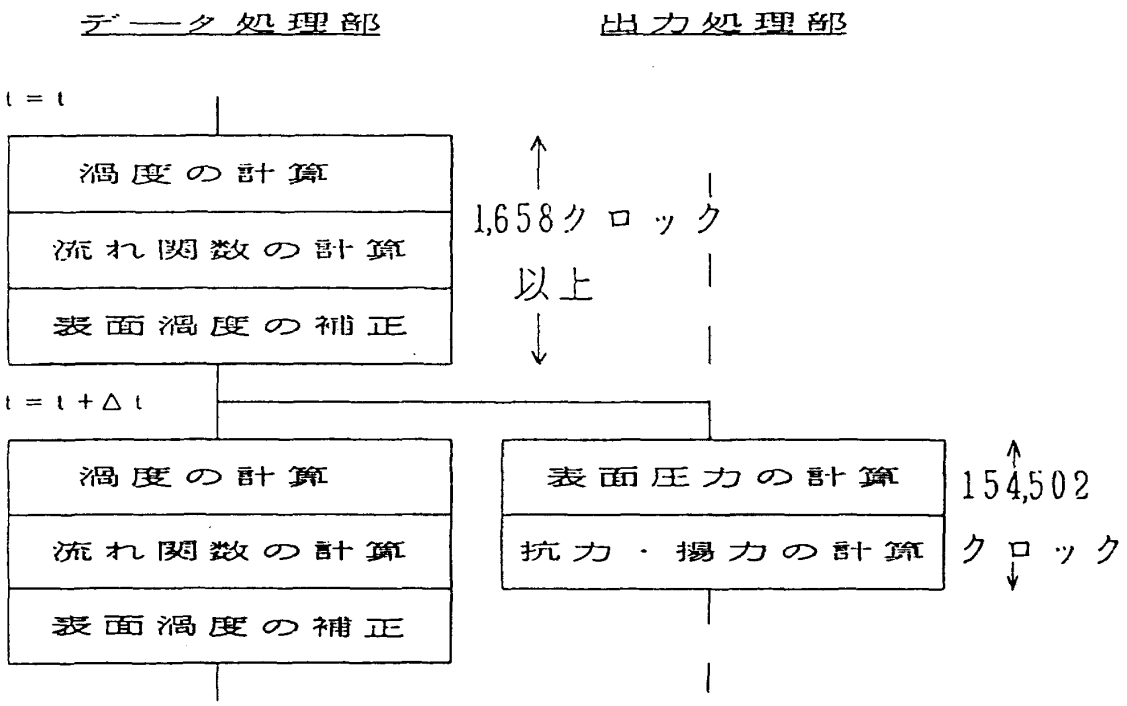


図29 負荷分散の考察

の計算に使用されないことから他の処理部（例えば入出力処理部）に移動でき，またすべきと思われる。ただし，渦度・流れ関数の計算と表面渦度の補正に要する時間以内でこれらを処理しようとすればその処理部はスカラデータ演算器の約100倍の性能を有するものでなければならない(図29)。

⑤ 渦度・流れ関数の計算および表面渦度の補正だけを観ると隣接結合の並列計算機による2次元非圧縮非定常ナビエストークス方程式による円柱まわりの流れ場解析に対しては約21,000倍の性能が達成できるものと思われる。

なお，並列計算機の構成，演算器，結合網，命令系および計算法等の見直しにより，より一層の性能向上を図ることが今後の課題である。

7. おわりに

シミュレータによる命令追跡はかなりの処理時間を必要とするため，プログラムを分割したルーチンごとに逐次処理による演算時間と並列処理による演算時間を得て，それらを比較することにより，並列計算機の性能を定量的に示すことが可能となった。この結果は性能向上策を検討する時の基礎データとして極めて明快である。並列計算機の計算結果をも出力できることは，プログラムの変換が正確に行われていることを確かめる上で特に必要である。これらの成果として，改善後の性能を推定することができるようになった。

他方，入出力処理についてはさらに多くのプログラムの検討が必要であり今後の課題である。

なお，この並列計算機の構想は1980年につくられ，1987年には1チップ計算機で1 MFLOPSが実現するとの予想のうえにモデル化されている。その後の計算機技術の進歩は予想をはるかに越え現在では数10 MFLOPSのものも可能になっている。

したがって並列計算機の構成も見直す時期に来ているが，一台の計算機による処理時間と複数台の計算機によるそれとの比は計算機構成が変わらない限り大幅に変化するものではない。ここではクロック数による比較で計算機の性能を計ったのもそのためである。

また，並列計算機のシミュレータは約5000フォートランステートメントから成る。今後は三次元問題等の各種応用プログラムを並列計算機に移植し命令追跡による検証模擬を実施することを計画しており，計算機の構成／命令系の検討・評価および改良を行う予定である。

謝　　辞

本研究を行うにあたり富士通(株)を始めとする多くの方々から有益な御意見・御協力をいただいた。また，流れ場解析プログラムの作成にあたり当所の井上応用解析研究室長より助言を頂いたことを記し感謝の意を表する。

参 考 文 献

1) Howard Jay Siegel : "A Madel of SIMD Machines and a Camparision of Various Interconnection Network", IEEE Trans. on Camp., Vol.C-28, No.12, December 1979.
2) Harold S. Stone : "Parallel Processing with the Perfect Shuffle", IEEE Trans. on Comp., Vol.C-20, No.2, February 1971.
3) Roger C. Swanson : "Interconnections for Parallel Memories to Unscramble p-Ordered Vectors", IEEE Trans. on Comp., Vol. C-23, No.11, November 1974.
4) Raphael A. Finkel and Marvin H. Soloman : "Processor Interconnection Strategies", IEEE Trans. on Comp., Vol.C-29, No.5, May 1980, p.360.
5) Kenneth E. Batcher : "The Flip Network in STARAN", Proc. of the 1976. International Conference on Parallel Processing.
6) Marshall C. Pease : "The Indirect Binany n-Cube Microprocessor Array", IEEE Trans. on Comp., Vol.C-26, No.5, May 1977.
7) Sullivan H.T.R.Bashkow and K. Klappholz : "A large scale homogeneous fully distributed parallel machine", Fourth Annual Symposium on Computer Architecture, March 1977, p.105.
8) Howard Jay Siegel, Robert J. McMillen and Philip T. Mueller JR : "A Sarvay of interconnection methods for reconfigurable parallel processing systems", National Computer Conference, 1979, p.529.
9) Duncan H. Lawrie : "Access and Alignment of Data in an Array Processor", IEEE Trans. on Comp., Vol.C-24, No.12, December 1975, p.1145.
10) G. Jack Lipovski, Anand Tripathi : "A Reconfigurable Varistructure Array Processor", Proc. of the 1977 International Conference on Parallel Processing, p.165.
11) Tse-Yan Feng : "Data Manipulating Functions in Parallel Processors and their Implementations", IEEE Trams. on Comp., Vol.C-23, No.3, March 1974, p.309.
12) 星野；"PAXコンピュータ"，オーム社，昭和60年3月15日.
13) 原田："並列計算機のアーキテクチャシミュレータ"，航空宇宙技術研究所資料，TM-583, 1988年3月.
14) 原田："並列計算機のアセンブラ"，航空宇宙技術研究所資料，TM-586, 1988年6月.
15) Kawaguti, M. and Jain, P. : "Numerical Study of a Viscous Fluid Flow Pasta Circular Cylinder", J. Phys. Soc. Japan, 21 (1966) 2055-62.
16) 原田："並列計算機のシミュレーション"，第6回航空機計算空気力学シンポジウム論文集，pp.49～54，航空宇宙技術研究所，SP-9, 1988年12月.
17) 原田，吉田，中村："2次元CFDによる並列計算機のシミュレーション"，第7回航空機計算空気力学シンポジウム論文集，pp.13～17，航空宇宙技術研究所，SP-10, 1989年11月.

付 録 1

演算器および記憶装置の寸法について

計算機はクロックに同期して動作するので，信号が回路を伝播することによる遅れを許容する限界としてクロックの半周期を目安とした。信号を干渉させないよう同軸ケーブルで接続すると波長短縮が起こり，ポリエチレン等の充填した材料の比誘電率の平方根に逆比例して信号伝播速度は低下する。ポリエチレンの比誘電率は2.25～2.30であり，自由空間の伝播速度（光の速度）に比べて66～67%となり1nsに伝播する距離は約20cmである。クロックの周波数を22MHzとすると半周期は約22nsでありその間に信号は約4.4m進むことができ，これが信号の伝わる最大の長さとなる。

各演算器要素を128×128台並べて二段に重ねクロックの信号源を中央に配置し稜線に平行してその信号が伝播する場合，最も離れている場所への経路の長さは

128cm＋128cm＋100cm

となり上記の制限以下となる。

付 録 2

記憶装置の容量について

三次元格子256×256×256点を想定すると各要素には2×256点が配置される。筆者の有する三次元流れ場解析プログラムでは各点に14変数を対応させているのでそのまま対応させると

2×256点×14変数/点×8バイト/変数

が必要となり余裕をもたせてその倍にし，それに最も近い2の冪乗として128Kバイトとした。

付 録 3

固定小数点演算レジスタの数について

固定小数点演算レジスタの7個は指標レジスタを兼ねることとした。例として，代入文

X(i)＝Y(j)＋Z(k)

のためには記憶装置のX，Y，Zの番地より(i－1)，(j－1)，(k－1)離れた番地を操作することになる。

この(i－1)，(j－1)，(k－1)を保持するのが指標レジスタである。代入文の右辺の変数が多くなれば，指標レジスタも多く必要となる。

また，固定小数点演算レジスタはループのカウンタとしても使用される。このため8個とした。

付 録 4

数値の説明

表8に示すように表面圧力，抗力・揚力の計算に並列処理で1,475クロック，3,719クロック必要である。このうち隣接結合でデータ転送に要する時間は

(1) 表面圧力の計算で

(1＋2＋4＋8＋16＋32＋64＋128)×2（往復）クロック

(2) 抗力・揚力の計算で

(1＋2＋4＋8＋16＋32＋64＋128)×2（往復）×4（式）クロック

であり，ネットワークを変えて転送時間を無くしても並列処理の時間は

(1) 表面圧力の計算に

(1,475－510) クロック＝965クロック

(2) 抗力・揚力の計算に

(3,719－2,040) クロック＝1,679クロック

必要である。これらを同表の逐次処理の時間と比較する。

航空宇宙技術研究所報告1154号

平成4年5月発行

発行所 航空宇宙技術研究所
東京都調布市深大寺東町7丁目44番地1
電話三鷹(0422)47-5911(大代表)〒182

印刷所 株式会社 三興印刷
東京都新宿区西早稲田2-1-18

Printed in Japan