≪論　文≫

論 3—7

時分割メモリ共有型マルチマイクロプロセッサシステムの試作およびアナログシミュレーションへの応用†

小迫秀夫*・後藤佳之**・木作　洋***・児島義明*

ABSTRACT This paper describes a compact time-shared common memory multi-microprocessor system, TCMS. It has m (not exceeding 8) processor units and a high speed common memory. It also uses a time-shard common bus system. Each unit consists of two microprocessors and a local memory and is connected throught a interface to a personal computer (used as a host computer). Each unit can access common memory at a speed one eighth (regardless of the value m) of a memory-cycle time of the microprocessor and transfer data to the other units within one memory-cycle.

As an example of the use of TCMS a simulation method is given for analog operations, using a digital processing approach. An analog operational component set up to the analog computer is represented in the form of a byte-serial packet, which contains the description of analog operations, the coeffients and the connecting imformation of the component. A set of such packets is stored into the common memory to be executed by means of a parallel processing scheme.

The test operations using this simulation method were tried to make certain the functions of TCMS.

1. まえがき

複数個のプロセッサユニット（PU）が共有メモリ（CM）を共通バスで結合されたマルチプロセッサシステムについては，すでに数多くの報告[1]がなされているが，その多くは非同期式である．この場合の問題点はバスアクセス競合である．このためにはバスアービタが必要になるとともに，PU からのバス使用要求が出される頻度が高い程 PU の処理効率[2]が低く評価される．一方，時分割同期方式ではこの問題は解消されるが，CM を介するデータ転送に時間がかかる．

本稿で述べる試作機（TCMS[3]と呼ぶ）は時分割バス共有方式を採用しているが，PU 間の CM を介するデータ転送速度を高速化した装置である．この方式は，PU を構成するマイクロプロセッサ（MP）の1動作サイクル時間 T_α より短い時間 t_{CM} でアクセスできる高速メモリを CM として使用し，各 PU と CM との結合を T_α/t_{CM} に時分割して切り換えるものである．

1つの CM モジュールへの PU の結合個数には上限があるが，回路設計の容易性および低コストを特長とした機種として実現され，当初の目標に達したものとしてここに報告する．

1つの応用例として，アナログ演算のシミュレーションを挙げた．ここではアナログ演算の並列性に着目し，接続された演算器と1対1対応のパケット処理方式を提案し，その処理結果について述べている．

2. TCMS の構成

2.1 基本構想

本稿で述べる試作機 TCMS はパーソナルコンピュ

A Trial Time-Shared Common Memory Multi-Microprocessor System and Its Application to Analog Simulations. By *Hideo Kosako* (Fuculty of Englneering, Univ. of Osaka Prefecture), *Yoshiyuki Goto* (Matsushita Electric Industrial Co., LTD), *Hiroshi Kisaku* (Sharp Corporation) and *Yoshiaki Kojima* (Fuculty of Engineering, Univ. of Osaka Prefecture).

* 大阪府立大学工学部，**松下電器産業（株）
***シャープ（株）

† 1983年10月3日受付

ータレベルのポストコンピュータ (HC) を用い，その管理のもとで動作されることを前提とした小規模システムであって，その基本構想は次のようである．

(1) システムは並列演算用ソフトウェア開発のための実験用装置とし，回路設計および取り扱いの容易さに重点をおいたコストパーフォーマンスのよいシステムを目標とする．

(2) 複数個のプロセッサユニットはメモリを時分割同期方式で共有し，このメモリを介するデータ転送の高速化を図る．

(3) 各プロセッサユニットは，ローカルメモリを有し，これを2個のマイクロプロセッサが共有するデュアルプロセッサ形式とし，ユニット内での処理速度を高める．

2.2 ハードウェア構成

TCMS では，各プロセッサユニット (PU) の CM へのアクセスには時分割多重方式を用いるが，この全分割の1周期をマイクロプロセッサ (MP) の1動作サイクル時間 T_α としている．したがって，PU の CM への結合数には上限があり，本機ではこれが8個に制限されている．

記号表

TCMS	：試作されたマルチマイクロプロセッサシステムの総称．HC を含む．
HC	：ホストコンピュータシステム
PU_i	：特定のプロセッサユニット i=0, 1, …7 とし，どの PU_i も同一構成．
PU	：$PU_0, PU_1, \cdots, PU_7$ の任意の1つ．
MP_M	：マスタマイクロプロセッサ (PU 内)．
MP_S	：スレーブマイクロプロセッサ (PU 内)．
MP	：MP_M または MP_S の呼称．
m	：PU の CM への接続個数 ($m \leqq 8$)．
LM	：ローカルメモリ (PU 内)．
CM	：$PU_0, PU_1, \cdots, PU_7$ の共通メモリ．
PA_i	：PU_i の MP_M からのアドレスバス．
PD_i	：PU_i の MP_M からのデータバス．
S_a	：アドレスセレクタ (マルチプレクサ)．
S_d	：データセレクタ (マルチプレクサ)．
A	：CM への単一アドレスバス上のアドレス．
D_w	：CM への書込みデータバス上のデータ．
D_r	：CM からの読出しデータバス上のデータ．
L_{Ri}	：D_r をラッチするラッチレジスタ．
L_{pi}	：L_{Ri} へのラッチパルス．(T_i 分割区間)
R_i	：PU_i の MP_M からのリードメモリ指令．
W_i	：PU_i の MP_M からのライトメモリ指令．
C_{cl}	：S_a, S_d への同期切換えクロック．
T_a	：MP の1動作サイクル時間．
T_i	：T_a を8分割した1つの時分割区間 i．
t_{CM}	：任意の PU による CM へのアクセス時間

2.2.1 プロセッサユニット PU の構成

システム全体の構成を **Fig. 1** に，また，PU 内の構成を **Fig. 2** に示す．PU 内の2つの MP はそれぞれマスタ MP (MP_M) およびスレーブ MP (MP_s) と呼ばれる．両者はほぼ同等の機能であるが，MP_s はとくに演算処理専用であり，MP_M は演算処理以外に，他の PU およびホストコンピュータ (HC) との通信制御などのインタフェースが付加され，主にこれらの間の通信および CM へのアクセスができる．

双方の MP は，**Fig. 3** に示すように，互に逆位相の MP クロックで駆動され，それぞれ $T_\alpha/2$ の間交互に LM にある DRAM をアクセスすることができる．この時分割・メモリ共有方式は MP に6809 E[注)] を使用することにより可能となる．これは，6809 E のメモリアクセス時のリード・ライトデータ (D_r/D_w) がともに1動作サイクル (1クロックサイクル) の後半で入出力できる特徴があるからである．Fig. 2 に示す S_L は2台の MP の DRAM へのバス結合を $T_\alpha/2$ ごとに切り換えるためのスイッチである．R_L は DRAM のリフレッシュ回路であって，Self-refresh 型である．

1つの PU 内のメモリマップを **Fig. 4** に示す．両側は各 MP のアドレス領域を示し，中央は DRAM からみた共通および専用領域を示す．MP_s が独立に指定できるアドレス領域に DM_2 がある．これは DRAM では下の MP_s 領域に対応している．この対応に

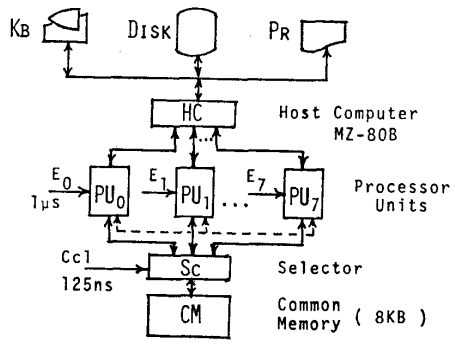

Fig. 1 Blook diagram of TCMS

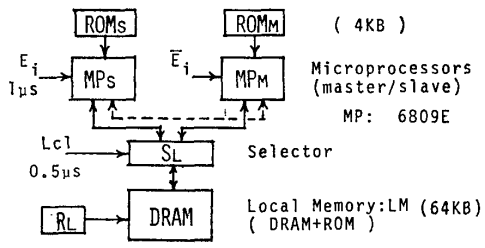

Fig. 2 A PU_i (i=0, 1, …, 7)

はアドレス変換回路が付加される．各 MP にはこの DRAM のほかにそれぞれ専用の ROM が結合されている．

以上のように，2つの MP が全アドレス領域（64 KB）内のどれを指定しても，LM および CM のすべてのメモリ領域が活用できる構成になっている．

2.2.2 PU のメモリ共有方式

1つの PU 内では MP の 1/2 動作サイクルで時分割して DRAM を共有する方式を用いたが，複数個の PU が CM を共有する場合に対しても同じ考え方を適用することができる．ただし，CM に用いるメモリには，高速メモリを用いるものとする．

いま，m 個のプロセッサユニット PU_0, PU_1, …PU_{m-1} が1つの CM を共有する回路として，各 PU のアドレスバス PA_0, PA_1, …PA_{m-1} およびデータバス PD_0, PD_1, …PD_{m-1} がそれぞれ同期的にセレクタスイッチ S_a および S_d（マルチプレクサ等）によって順に切り換えられて CM に結合されるような回路を考える．さらに，CM から読み出されるデータは各 PU ごとに置かれているラッチレジスタ $L_{R0}, L_{R1}\cdots, L_{Rm-1}$ の順にラッチされるものとする．

MP の1動作サイクル時間 T_α 内ですべての PU が CM をアクセスするためには，PU の接続可能個数は

$$m \leqq T_\alpha / t_{CM} \qquad (1)$$

でなければならない．ただし，t_{CM} は

$$t_{CM} = t_d + t_{AC} + t_L \qquad (2)$$

とし，t_d を S_a および S_d のバス切り換え時間，t_{AC} を CM へのアクセス時間および周辺の遅延時間，t_L をラッチ回路へのリードデータのセットアップ時間とする．

本試作機では，CM として最大アクセス時間が55nsなる8 KB CMOS RAM（HM 6147 P—3×16）を使用し，他の全遅延時間が 70ns 以下になるように設計されたセレクタ回路およびラッチ回路をこれに結合している．これは技術的にも実現は容易であり，$m=8$ ならば(1)式の制約を充分満たすことができる．

Fig. 5 に CM への時分割アクセス共有方式の回路構成を示す．S_a および S_L は 125 ns のクロック C_{cl} で切り換えられるマルチプレクサである．PU_i（i=0 1, …, 7）からの書き込みは，書き込み信号 W_i によって PD_i（8ビット）上のデータが D_w として CM に書き込まれる．書き込まれるアドレスは PA_i（アドレス13ビット，コントロール2ビット）上のアドレス A

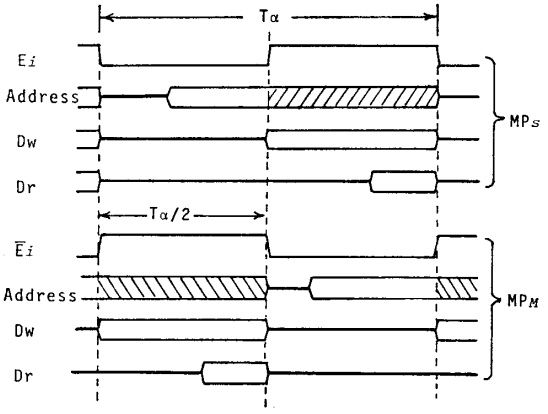

Fig. 3 Timing chart of MP_S/MP_M in a PU

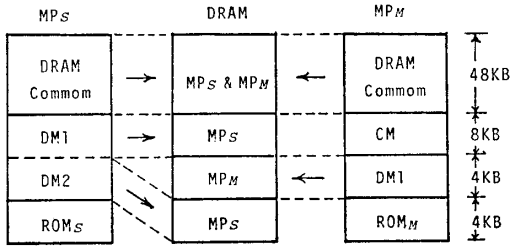

Fig. 4 The memory map of LM

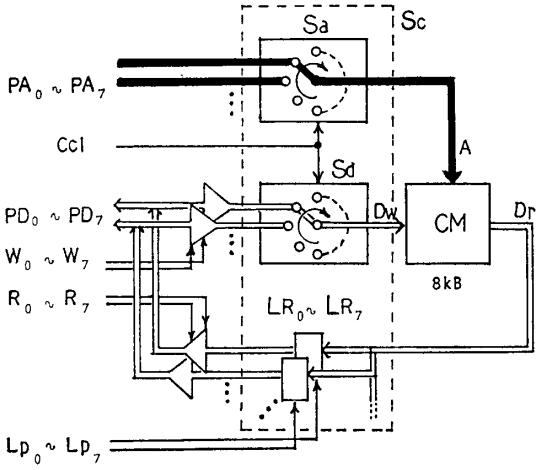

Fig. 5 Selector switches in TCMS

注）6089は6800のアーキテクチャと命令体系がさらに強化された8ビットマイクロプロセッサで，8ビットと16ビットのギャップを埋めるプロセッサとして位置づけられている．6809 E は，6809が内部で基本クロックをつくり出すのに対して，外部からこれを入力するタイプであって，メモリアクセスを事前に知らせるAVMA信号，バスロックのためのBUSY信号が追加されているため，マルチプロセッサ用に構成され易い．

である．PU_i への読み出しデータ D_r は PA_i のアドレスの内容であって，L_{Ri} に一時的にラッチされ，読み出し信号 R_i によって読み取られる．

以上のように CM が $T_\alpha/8$ の時分割によって8個の PU に共有される場合，CM から PU を見て，すべて同じ動作タイミングでアクセスされるのがもっとも望ましい．このためには **Fig. 6** に示されるように，$PU_0, PU_1, \cdots, PU_7$ の順に $T_\alpha/8$ ずつ動作タイミングを時間的にずらせて動作させなければならない．すなわち，どの PU も1動作サイクルを8分割した共通の1区間（前半の区間内とする）が CM アクセスに割当てられた区間であるとする．

Fig. 7 は Fig. 5 で示した主なる個所の観測波形である．これらの波形は PU_7 および PU_1 が CM に対して“同時に”（T_α 以内）リード指令を発し，PU_0 は CM をアクセスしていない状態にある場合である．波形 A は $\overline{CE}$（チップイネーブル），波形 D_r はある1つのビットデータを示す．クロック C_{cl} の立上り（セレクタの切り換え開始時）から D_r が読み出されるまでの時間（t_d+t_{AC}）は80～90ns，残りの 35～45ns の間で D_r がラッチパルス L_p（C_{cl} の負の半サイクル）によって L_R にラッチされる．時分割は…T_7, T_0, T_1, …の順に進行し，CM から出力されたデータ $D_r(T_7)$ $D_r(T_0)$，$D_r(T_1)$ はそれぞれ L（低レベル）Z（高インピーダンス状態），H（高レベル）を示す．この連続した波形 D_r は，ラッチパルス L_{P7} および L_{P1} によって $L_{R7} \leftarrow D_r$ (T_7)，$L_{R1} \leftarrow D_r$ (T_1) にそれぞれ分配されることになる．

2.2.3 PU 間の通信および割込み制御

TCMS には PU 間の通信および割込み制御のために8バイト（64ビット）分のフラグレジスタとその制御回路からなるフラグ回路が設けられている．このレジスタは，8バイト連続したアドレス $F_0, F_1, \cdots, F_7$ として CM 上にプログラマブルに設定できる．ここで，アドレスに属する第 i ビットのフラグを $(F_j)_i$ $(i, j=0,1,\cdots,7)$ と表わし，その機能を説明する．

$(F_j)_i$ にビットデータ 1/0 を書込むために，PU_i の MP_M からのライト（STA）/リード（LDA）命令を用いるが，そのアドレス指定には $F_0 \sim F_7$ とは異る CM 内の特定アドレス A_{ji} を用いる．書込まれた $(F_j)_i$ の値はアドレス F_j を使ってリードする．このとき，F_i に属する他のビット内容とともに（1バイト分）読出される．これらのフラグ回路には次のような制御機能が付加されている．$(F_j)_i=1$ のとき，割込み信号 $IRQ_i=1$ が以下の PU に送られる．$i \neq j$ ならば PU_j のみに，$i=j$ ならば，PU_j を除くすべての PU に送られる．$(F_j)_i=0$ のとき，$IRQ_i=0$ となり，割込みが解消される．

上の機能を用いた例として，PU_i の MP_M のレジスタのデータ A_{ci} を割込みを使って PU_j（割込みソース i は未知とする）の A_{cj} へ転送する方法を次に示す．

- PU_i からの命令：

  STA A_{ji}；$(A_{ji}) \leftarrow A_{ci}$，$(F_j)_i \leftarrow 1$ および PU_j $\leftarrow IRQ_i=1$

- PU_j（割込みを受けた）からの命令：

  LDA F_j；$A_{cj} \leftarrow (F_j)$

中略（割込みソース i を識別，$k \leftarrow i$ としたあと次へ）

  LDA A_{jk}；$A_{cj} \leftarrow (A_{ji})$，$(F_j)_i \leftarrow 0$ および $IRQ_i=0$

ただし，（××）は××番地の内容とする．

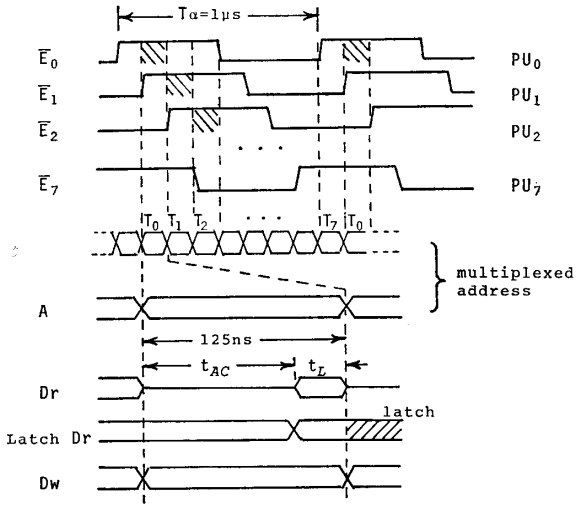

Fig. 6 Access timing of each PU to CM

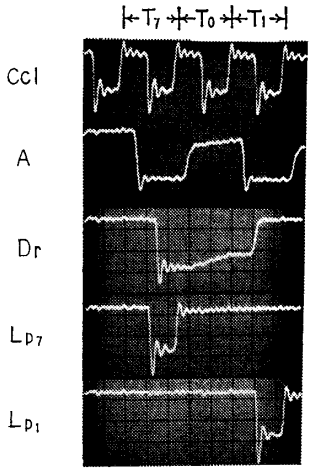

Fig. 7 Observation waveforms of the circuit shown in Fig. 5

なお，PU 内の 2 MP 間の通信および割込みには 2 ビットの上と同機能のフラグ回路を用いている．

2.2.4 PU および HC インタフェース

TCMS の周辺機器は PU に直接接続されないで，すべて HC の周辺機器として結合される．本稿では，複数台の PU 並列演算処理能力を調べることを初期目標においているため，入出力の並列処理について試みていない．しかし，各 PU ともインタフェース（HC との間のデータ交換および通信制御のみに使用）を有することから，PU と直接接続できる入出力装置を共有資源とするような応用面の開発が今後の 1 つの課題である．

2.3 基本ソフトウエア

2.3.1 モニタプログラムの概要

TCMS に組み込まれたモニタプログラムは，マルチプロセッサ側の各 PU（MP_S および MP_M）で処理を受けもつプログラムと HC 側で受けもつプログラムに分けられる．前者は各 PU の ROM におかれていて，PU 側コマンド（HC が各 PU に与えるコマンド）の解釈および実行が主である．後者は HC 側コマンドの PU 側コマンドへの翻訳，加えてそれらの解釈および実行，マルチマイクロプロセッサシステムのターミナルとしての動作を行い，フロッピーディスクに格納される．

2.3.2 PU 側モニタプログラム

PU 側メモリおよび HC 側メモリ間のデータの転送，PU 内レジスタの続み出しおよび書き換え，ユーザプログラムの起動（全 PU 同時起動を含む）などのコマンドの解釈および実行がこのモニタプログラムの主なる機能であるが，そのほか，メモリチェック，MP 間インタフェースのチェックを含む．

2.3.3 HC 側モニタプログラム

HC 側の主なるコマンドには次のものがある．

キー入力データの主メモリへの格納および図面表示，スクリーンエディット，HC 側レジスタ類の表示および値の変更，ファイルのフロッピーディスクへのセーブおよび FDOS の起動などである．

これらのコマンドの記述には簡単な記号言語形式を用いる．PU 側の送るコマンドはさらに簡単なコマンドであり，HC 側コマンドは PU 側コマンドの集合として翻訳される．勿論，PU_i の指定およびその中の MP の指定もコマンド記述の中で与えることができる．

ユーザプログラムは現在のところアセンブリ言語（6809適用）を使用し，HC ディスクシステムのクロスアセンブラによって翻訳される．

3. アナログシミュレーションへの応用

現在のマルチマイクロプロセッサシステムの特徴を大別すると，ある限られた演算に重点をおいて設計・試作された専用機およびマイクロプロセッサの結合方式等ハードウェアにその特徴をもたせた汎用機であろう．いずれにおいてもプログラムの実行に際して，そのシステムに適切なタスクの分担やオーバヘッドの抑制などに充分な配慮がなされなければならない．

本稿で述べる応用は，これらの点を考慮し，汎用機としての 1 つの試みとしてアナログ演算のシミュレーションをとり挙げた．

3.1 アナログプログラムの処理形式

一般にいわゆるアナログプログラムとは，アナログ演算器の種類とそれらの間の接続および接続係数，初期値および演算モード形式で示されると考える．ここでは，以下で述べるようなパケットで表現されたアナログプログラムを用いる．

3.1.1 アナログプログラムのパケット表現

あるアナログプログラムにおける演算器 i ($i \in I$, $I=\{1,2,\cdots,M\}$) と 1 対 1 対応をなすパケット P_i として，**Fig. 8** に示すような可変長のバイトシリアルのデータ集合を与え，次の 5 つのフィールドに分ける．

フィールド1. f：演算器 i の種類を示すコード．

フィールド2. C_i：演算器 i への入力接続数．

フィールド3. a_{ik}, a_{il}, …：演算器 $k, l, \cdots$ から i への入力接続係数（C_i 個）．

フィールド4. $A_k, A_l, \cdots$：演算器 $k, l, \cdots$ から i への入力値 $y_k(n), y_l(n), \cdots$（後述）の格納アドレス（C_i 個）．

フィールド5. A'_i：演算器 i の出力値 $y_i(n+1)$（後述）の格納アドレス．

このような集合 P_i の集合族 $\{P_i \mid i \in I\}$ をここではアナログプログラムと定義する．プログラムとして P_i の並び順序は任意に定めうる．なぜならば，どのパケット P_i もフィールド 2～4 において他のパケッ

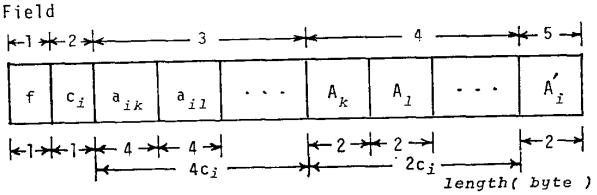

Fig. 8 The configuration of an operation-packet

トとの接続情報が与えられるからである．以後，この並びを $P_1, P_2, \cdots, P_M$ の順にする．

3.1.2　1つの PU におけるパケット処理

プログラム $\{P_i \mid i \in I\}$ が任意の1つの PU によって実行されうるために，プログラムおよびその関連データ (**Fig. 9**) は CM 上に格納される．ここで用いる記号アドレス（相対アドレス）には次のデータが格納される．

記号アドレス：格納されるデータ〔長さ：バイト〕

AP_i　：P_i〔$6C_i+4$〕の先頭データ

TA_i　：データとしての AP_i〔2〕

TC　：データとしての TA_i〔2〕

A_i, A'_i　：$y_i(n), y_i(n+1)$〔4〕,〔4〕

P_i なる一連のデータ集合は AP_i を先頭アドレスとして CM 上の任意の場所（特定アドレス領域は除く）に置かれうる．TA_i はテーブルアドレスとして定義され，$TA_{i+1}=TA_i+2$ である．TC はこのテーブルのアドレスカウンタとしてメモリ上で定義されるアドレスで，TC の内容 (TC) によって間接に AP_i が指定されうる．$y_i(n+1)$（浮動小数点表示）は P_i について n 回目のきざみ演算が繰返されたときの演算結果であり，P_i の出力値といい，A'_i におかれる．$y_i(n)$ はそのときの演算に用いられ，P_i の入力値といい，A_i におかれる．$y_i(1)$は P_i の初期値として A_i におかれる．

つぎに，1つの PU が CM から P_i をとり込んだあとでこれを解読し，演算を実行するパケット処理プログラムは PU 内の LM 上に格納されなければならない．

1つの PU が1つのパケット P_i を処理するステップは次の4つに分けられる．

ステップ1　パケット P_i のフェッチ (MP_M)：

1.1 (TC) の更新：$(TC)=TA_i$ を読む．$i>M$ なら $(TC)\leftarrow(TC)+2$ として1.2へ，$i=M$ なら $(TC)=TA_1$ とし，後続の全ステップ終了まで他の PU からのパケットフェッチを禁止（ステップ4で解除）する．

1.2 $(TA_i)=AP_i$ を読む．

1.3 P_i をその先頭アドレス AP_i から読む．

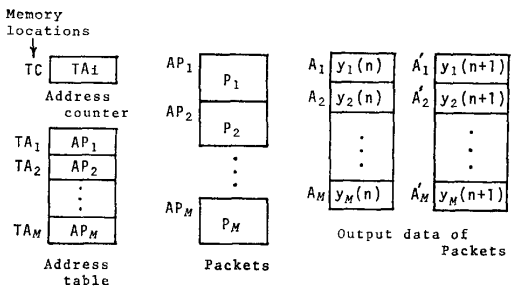

Fig. 9　An analog program $\{P_i\}_i \in I\}$ and related data to be stored in CM

ステップ2　P_i の P_i^* への変換 (MP_M)：

P_i のフィールド1のコード f を f_{SUB} とし，フィールド4の $A_k, \cdots$ をデータ $y_k(n)\cdots$ に置換えたものをオブジェクトパケット P_i^* として LM の共有領域におく。なお，f_{SUB} は LM 上にある演算用サブルーチンの演算開始アドレスである．

ステップ3　P_i^* の解読・実行 (MP_S/MP_M)：

P_i^* を LM からとり込んだ MP は，命令実行を f_{SUB} に移し，P_i のフィールド2～4に対応する P_i^* のデータを用いて演算器 i に関する演算（DDA 方式とする）を実行する．この演算は1つのきざみ幅について行われ，結果は $y_i(n+1)$ として MP_M に渡される．以上の処理では，MP_S または MP_M のみによる単独か，両 MP による並行運転が可能である．

ステップ4　出力データ $y_i(n+1)$の格納 (MP_M)：

$y_i(n+1)$は CM の A'_i に格納され，ステップ1に戻る．ここで $i=M$ のとき，他の PU によるパケットフェッチの禁止（ステップ 1.1）が解除され，きざみ回数 n の更新とともに，すべての i についてデータの入れ替え $(A_i)\leftarrow(A'_i)$（実際上はアドレスの読み替え $A_i\leftarrow A'_i$ で充分である）を実行する．

以上の処理に先立って，パケット数，初期値，きざみ幅およびきざみの全繰返し回数の設定が必要である．

つぎに，このパケット処理方式の主な特徴を挙げる．

(1)　ステップのパケットフェッチ方法は2レベルマイクロプログラミング方式と類似である．その理由は，パケットのフェッチはその先頭アドレスで指定するが，どのアドレスを指定するかはテーブルアドレスカウンタ TC の内容で間接に指定しているからである．

(2)　ステップ3で，MP_S のみが演算を分担する場合，MP_M はステップ1および2を並行して実行できる．これはオブジェクトパケットを Queue 構造（LM 上に対応）に先取りしたパイプライン方式を意味する．

3.1.3　マルチプロセッサを用いたパケット処理

どの $PU_i(i \in I)$ がどのパケットを処理するかについての一般的な割当て方法には，(1) 固定割当て方法および (2) ダイナミック割当て方法が考えられる．(1)は各 PU にあらかじめ定められたパケットの処理を割当てるものである．(2) は3.1.2で述べた方法であって，

“空”となったどのPUも（*TC*）で間接に指定されるパケットの処理が自動的に割当てられ，(1) に比してPUの稼動率が高い．ただし，(2) では，どのPUもCM上に存在する全種類のパケットが処理されうるように，LM上にパケット処理プログラムが準備されていなければならない．

つぎに，ハードウェア上ではどのPUもCMへのバスアクセス競合を起こすことなく，独立してステップ1～4の処理過程を実行しうる．しかし，ソフトウェア上で，パケットフェッチの時点に限って次の事態が発生する．

「複数個のPUがともにステップ1.1であるとき，すなわち，TCの内容が1つのPUによって更新されないうちに他のPUがTCの内容を読みとるとき，これらのPUは同じパケットを処理することになる．

この状態をここではマルチプロセッサシステムにおけるパケットのオーバラップ処理と呼ぶ．これを回避するために，2.2.3で述べたフラグ機能を用いた次のような対策を講ずる．

〔オーバラップ処理への対策〕

1つの PU_i がステップ1.1を実行する前後に，**Fig. 10** に示すような過程を追加する．

① 常時は割込み禁止とし，PU_i でパケットのアクセス要求が発生した時点で PU_i を割込み許可（EI）とする．

② 他の $PU_j(i \neq j)$ のすべてに割込み信号を送出し，PU_i が“TCを更新中”であることを知らせる（$(F_i)_i \leftarrow 1$）．

③ TCの更新が終わると，これを解消する（$(F_i)_i \leftarrow 0$）．

④ PU_i が割込みを許可してから割込み禁止（DI）が実行に移されるまでの間（②の期間で8 μs）に，他の PU_i からの割込みが入るとき，Fig. 10の右に示す割込み処理プログラムを実行する．これがSTA A_{ii} 実行開始前ならば $(F_i)_i=0$ である．開始後ならば $(F_i)_i=1$ であり，同時割込みとなる．この状態をパケットのアクセス競合と呼ぶ．

⑤ 開始前のとき，すべての j について $(F_j)_j=0$ となるまで待つ．

⑥ 開始後のとき，あらかじめ定められたPUの優先順（PU_0，PU_1，…PU_7 の順）に従って，自己と他の割込みPUとの間の順位を調べ高位のPUが割込みを解除するまで待つ．

③′ ここではパケットが P_M であることが識別された場合，割込み解除を全ステップ処理後に出すことを意味する．

3.2 TCMSのテスト例

TCMSの並列処理性能を調べるために用いたシミュレーションの仕様を以下に示す．

(1) パケットの種類：加算係数器，加算積分器，係数器および掛算器

(2) パケットの個数：最大255個

(3) 演算方式および数値形式：浮動小数点演算方式，数値4バイト長（仮数部3バイト（絶対値表示）＋仮数部の符号＋指数部（2つの補数表示）7ビット）

(4) 積分公式：オイラー法またはアダムス法

(5) PU使用個数：1～4個（最大8個接続可能）

TCMSの並列演算処理能のテスト例のうちの1つのプログラム例を **Fig. 11** に示す．演算パケットはすべて同一の積分器とし，P_3～P_M の並列接続部（ダミー）を可変とし，マルチプロセッサシステムの処理時間を調べた．結果を**表1**に示す．

表1で示す t_1 は各PUを MP_M 単独で用いた場合の処理時間，t_2 は MP_S を演算専用（ステップ3）に，MP_M をこれ以外のステップ処理用に分担して MP_M のパケット先取りを試みた場合の処理時間である．なお，試作完了のPUは4個であるため，使用個数 m は

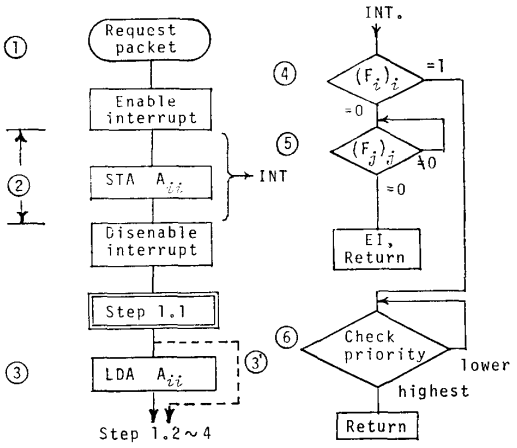

Fig. 10 A method against the overrap processing of packets by multi-processor

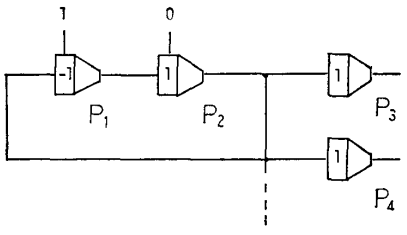

Fig. 11 A program for a test of parallel-operations, P_1, P_2, P_M, $M=2\sim64$

この範囲で可変（PU の起動選択による）とした．

このテスト結果より以下のことが判かる．

(1) $M<m$ のとき，m の増設による処理時間の短縮効果はない．〔理由〕1つのきざみ演算について一通りのパケット処理が実行されない限り，次のきざみ演算にすすめない．このため，$M(<m)$ 個の PU で M 個のパケットが処理されている間は残りの PU は“空”状態である．これはパケットの並列処理に貢献しない．

(2) $M\geqq m$ のとき，$M=km$ (k：整数) ならば，

$$t(M_1, m_1)\approx\frac{M_1}{M_2}\cdot\frac{m_2}{m_1}t(M_2, m_2) \qquad (3)$$

なる傾向にある．ただし，パケット M 個が PU m 個で処理される時間を $t(M, m)$ とかき，表1の値とする．例えば，$t=t_2$ について，$t_2(64, 4)=29.6$ および $t_2(16, 2)=14.8$ の間には(3)式が成立する．

(3)式は次のことを意味する．

「一定個数の PU でパケットを処理する時間は大体パケットの倍数となるが，一定個数のパケットを処理する時間は PU 使用個数の逆数倍である」

表 1　TCMS のテスト結果

パケット数 M	使用PU数 m	1きざみ当りの演算時間 MP_M t_1 msec	1きざみ当りの演算時間 MP_SおよびMP_M t_2 msec
2	1	4.2	4.0
	2	2.2	2.4
	3	2.2	2.4
	4	2.2	2.4
4	1	8.0	7.6
	2	4.2	4.2
	3	4.2	4.0
	4	2.6	2.6
8	1	15.6	14.6
	2	8.2	7.8
	3	6.2	6.0
	4	4.6	4.4
16	1	31.2	28.6
	2	15.8	14.8
	3	12.2	11.2
	4	8.6	8.0
32	1	62.0	56.6
	2	31.4	29.0
	3	22.0	20.2
	4	16.6	15.2
64	1	123.8	112.8
	2	62.4	57.4
	3	44.4	36.6
	4	32.4	29.6

きざみ幅　$\Delta t=2^{-6}$，繰返し回数　5000

$M\neq km$ ならば（表1では $m=3$），M が m に近い場合，(3)式の関係は成立しない．その理由は，m 個のパケットで1つのきざみ演算が繰返されたあとで端数個（M/m の剰余）のパケットが処理されるとき，空 PU ができるからである．しかし，$M\gg m$ の場合はこれらの空 PU の待ち時間は全処理時間に比して小さくなり，(3)式の関係に近づく．

(3) m に比して M が大きいとき，全般に $t_1>t_2$ である．これは MP_S がステップ3を実行している間の MP_M によるパケット先取り効果を示している．しかし，$(t_1-t_2)/t_2$ は0.1以内であり，処理時間の殆どが MP_S の演算時間で占められていることがわかる．なお，両方の MP にステップ3で異るパケットの演算を実行させたテスト（この場合，先取り効果の推定は困難）によると，個々の PU の処理速度は約2倍に向上している．

3.3　シミュレーションの例

本機を用いたシミュレーションの例として，次の Van der Pol 方程式を解くことを試みた．

$$\frac{d^2x}{dt^2}-\varepsilon(1-x^2)\frac{dx}{dt}+x=0 \qquad (4)$$

ただし，dx/dt および x の初期値をそれぞれ $\dot{x}(0)$ および $x(0)$ とする．これを解くためのプログラムを **Fig. 12** に示す．

P_1 および P_2 はそれぞれ3入力および1入力の加算積分器，P_3 は3入力掛算器を示すパケットである．

CM 上におかれるプログラムは Fig. 8 および Fig. 12より次のように表わされる．

AP_1：$I, 3, a_{11}, a_{12}, a_{13}, A_1, A_2, A_3, A'_1$

AP_2：$I, 1, a_{21}, A_1, A'_2$

AP_3：$M, 3, a_{31}, a_{32}, a_{32}, A_1, A_2, A_2, A'_3$

ただし，I および M はそれぞれ積分器および掛算器のコード，$a_{11}=-a_{13}=\varepsilon$，$a_{21}=a_{31}=a_{32}=1$ および $a_{12}=$

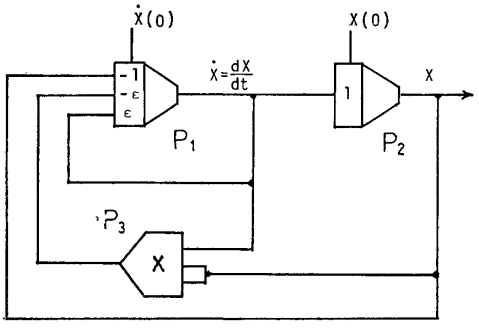

Fig. 12　A progan to solve Van der Pol equation

-1 である．初期値 $(y_i(1), i=1,2,3)$ は $(A_1)=\dot{x}(0)$，$(A_2)=x(0)$ および $(A_3)=0$ とおかれる．

Fig. 13 は $\varepsilon=1$，$\dot{x}(0)=0$ および $x(0)=2^{-4}$ としたときの解 x を示す．ここで用いたPUの処理形式は3.2で示したデュアル形式である．実際の演算時間は，きざみ幅を $\Delta t=2^{-8}$ として，$t=20$ において12.3 sec であった．なお，この解波形は実時間表示ではなく，きざみ演算の繰返しの20回に1回の割合でPUの出力データをHCのメモリに一度転送し，のちにこれをプリントアウトしたものをプロットして描いたものである．

4. あとがき

小規模なマルチマイクロプロセッサシステムの試作機について，そのハードウェアの全貌およびパケット演算方式を用いたアナログシミュレーションについて述べた．プロセッサユニット PU が CM に結合できる数は8個であるが，この接続は単にPUからのバスを CM セレクタの時分割空チャネルに接続するだけで充分であり，ハードウェア上の動作変更は不要である．また，これ以上の拡張については，本機を基本ユニットとした複数個の結合によって実現可能である．現在ROMに搭載されているモニタプログラムは数百バイト程度であるため容量的には充分な余裕があり，今後，FORTHの核などの高級言語の一部，浮動小数点演算用パッケージの搭載も考えられる．応用面では，アナログシミュレーション用高級言語の開発がある．また，各 PU が 2MP から構成されている特徴を活用する分野として，データフローマシン・エミュレータなどが考えられる．

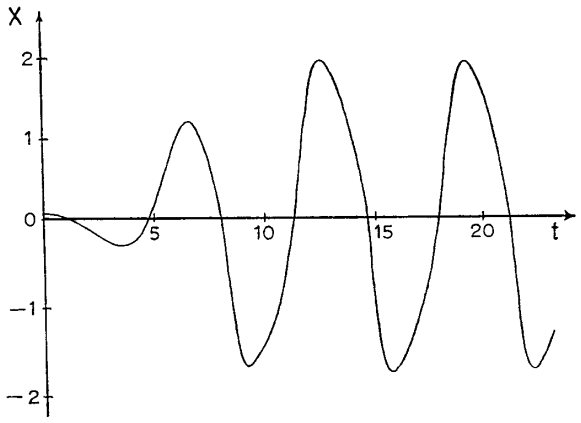

Fig. 13 A solution of Van der Pol equation by TCMS

参考文献

1) 高橋：並列処理のためのプロセッサ結合方式，情報処理，**23**—3，201/209 (1982)
2) 深海，中村：マルチマイクロプロセッサのバス制御方式に関する一考察，昭58．電子通信学会情報，システム全大論文集 [2]，536 (1983)
3) 小迫，後藤，木作，児島：時分割メモリ共有型マルチマイクロプロセッサシステムの試作，日本シミュレーション学会3回シミレーション・テクノロジー・コンファレンス論文集，**1A**—1/4 (1983)