計算機アーキテクチャ　66－5
（１９８７．７．１０）

２進木マシンＣｏｒａｌ６８Ｋのシステム構成と性能評価

遠藤俊雄，松尾賢二，白方新洋，槌谷一，高橋義造

徳島大学工学部情報工学科

実際の応用問題に関する並列計算に使用するために、２進木マシンCoral68Kを開発した。Coral68Kは、63台のプロセッサ要素からなり、ホスト計算機に接続されている。プロセッサ要素は、MPUに68000(10MHZ)を転送方式はDMA方式としDMAC 68450(10MHZ)を採用し、ROMは16KB、RAMは512KBからなる。また、ソフトウェア環境整備のためにIPL、デバッガ及びランタイムサブルーチンを作成した。IPLは、ホスト計算機から送られた各種メッセージを処理を行う。
このCoral68K上でナップザック問題、並列ソート、ガウス・ジョルダン法、巡回セールスマン問題を実行し、性能評価した結果、等価的に40MIPS,0.3MFLOPS処理能力が得られた。

Structure and Performance of the Binary-Tree Machine Coral68K

Toshio ENDOU, Kenji MATSUO, Sinyou SIRAKATA, Hajime TSUCHITANI, and Yoshizou TAKAHASHI

Department of Information Science, Tokushima University
2-1 Minami Jyousanjima-cho,Tokushima 770, Japan

In order to practice parallel computing of various real-scale problems, a binary-tree machine Coral68K was developed. The Coral68K consists of 63 processor elements interconnected in a five-level binary tree, and connected to a host computer which is a UNIX workstation. The processor element is composed from a 16 bit MPU 68000(10MHZ), a DMAC 68450(10MHZ), 16 KB ROM, and 512 KB RAM. The software development environment for Coral68K including an IPL, a debugger, and run-time subroutines was also developed. The IPL accepts messages transmitted from the host computer and processes them.

Several parallel computing programs were executed on Coral68K to evaluate the performance of this machine. The results indicate performance of Coral68K as 40 MIPS and 0.3 MFLOPS.

1.はじめに

Coralは2進木構造を持つ並列計算機である。これは数百数千のプロセッサを用いた高次並列処理を目標としている。15台のプロセッサ（MPU 8085）より構成されているCoralプロトタイプ'83[1]が1983年に完成した。これは並列処理プログラムの開発や、並列処理の効率評価を行うのに使用してきた。しかし、MPUの演算処理能力が劣り通信速度が遅くメモリも少ないという問題があり実用的な問題に使用するには力不足であった。この経験を生かし実用的な計算に使用することを目的として、新にMPUに68000を採用し、プロセッサ間通信にはDMA転送方式を用いてメモリも増加させたCoral68K（63台結合）を開発した[2]。またCoralとホスト計算機を結ぶインターフェイス製作し、ソフトウェア開発のためのIPL、デバッガ、ランタイムサブルーチンを作成した。そしてCoral68Kの性能評価をするためにアプリケーションプログラムを作成し効率を測定した。

2.システム構成

開発したCoral68Kのシステム仕様、システム構成図と、筐体前面の図を表2.1,図2.1,図2.2に示す。

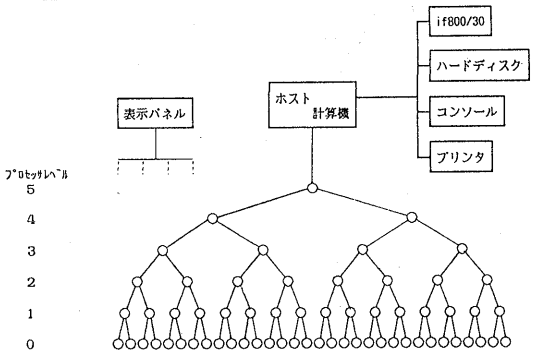

図2.1 システム構成図

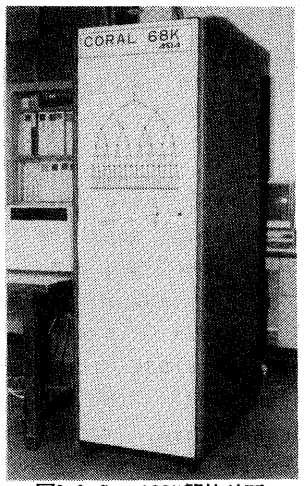

図2.2 Coral68K筐体前面

ホスト計算機にはUNIXベースのワークステーションOA-8040を使用している。ホスト計算機にの8ﾋﾞｯﾄパラレルインターフェイスを利用してCoral68Kと接続した。63台のプロセッサ要素は4台のシャーシに分けて筐体に納められている。筐体正面には、各プロセッサ要素の動作とプロセッサ間の通信方向をLEDで示す表示パネルを付けている。またホスト計算機の端末としてif800/model30が接続されている。

表2.1 システムの仕様

要素	項目	Coral'83	Coral68K
ホスト計算機	モデル CPU メモリ ﾊｰﾄﾞﾃﾞｨｽｸ OS	if800/30 Z80B(4MHz) 128KB － CP/M 80	シャープOA-8040 MC68000(10MHz) 2MB 20MB UNIX SYSTEM III
プロセッサ要素	台数 CPU ROM RAM	15台 i8085(2.5MHz) 6KB 17KB	63台 MC68000(10MHz) 16KB 512KB
転送速度	HOST-PE間	1.2KB/秒	20KB/秒
	PE-PE 間	9KB/秒	2.0MB/秒

3.プロセッサ要素

3.1 構成

プロセッサ要素[3]のブロック図を図3.1に示す。

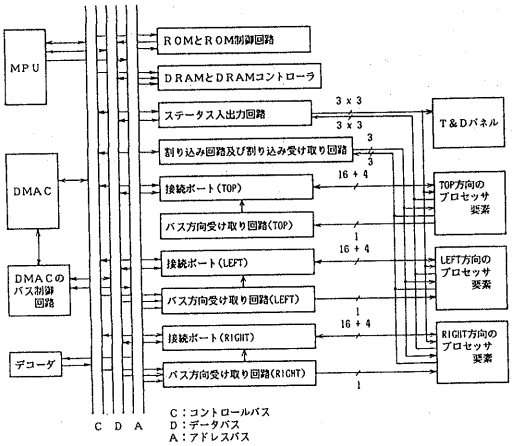

図3.1 プロセッサ要素のブロック図

MPUに68000(PGA 10MHz)を、DMACには68450(PGA 10MHz)を使用している。

ROMはTMM2764を2個使用し16KBのエリアを持ち、例外処理ベクタの必要性より0番地に設定する。RAMはTMM41256を16個使用し512KBのエリアを持ち、DRAMコントローラ（MB1422）で、リード、ライト、リフレッシュの制御を行う。

1台のプロセッサ要素はサイズが32(cm)x26(cm)の4層プリント基板に収納され、LSI以外にTTL 67個、PAL 8個が搭載されている。コネクタ部は96ピンのDINコネクタが2組あり、3方向の信号線にそれぞれ32ピンずつ使用し、GNDはTOPに32ピン、LEFT、RIGHTは共通で32ピン使用する。残りは外部リセット端子、割り込みスイッチ信号端子等に使用している。

3.2 プロセッサ要素間通信回路

3方向の接続ポートは転送のためのハンドシェイク回路であり、TTLとPALで構成されている。DMACの4チャネルの内、3チャネルをTOP,LEFT,RIGHTの3方向の接続ポートに利用している。優先順位はTOP,LEFT,RIGHTの順である。転送は、TOPとLEFTまたはTOPとRIGHTの接続ポート間でお互いのREQ-OUTをアクティブにして転送要求することで開始される。我々はDMA転送の中でも転送速度の速いバーストモードを採用した。これは複数データを転送する間バスを保持し続ける方法である。DMACの内部レジスタは各チャネルごとに17個あるが、転送

アドレス、転送語数、その他の制御レジスタ（計5個）を設定して転送する。バス方向の制御はLEFT,RIGHTのバス方向選択回路より信号を出し、TOPはその信号を受け取る。

表示パネルには、プロセッサ要素の状態を示す赤色LEDが２進木状に、またその間に転送方向を示す黄・橙色の2色LEDが取り付けられている。これらはTOPのステータス出力の最上位1ﾋﾞｯﾄ、ハンドシェイク信号のREADY-INとBUS directionで駆動している。この表示パネルはデバッグ等に利用する。

また、割り込み受け取り回路では、割り込まれたレベルに応じて内部で割り込みベクタを発生させるオートベクタリングを採用している。デバッガのための割り込みは割り込みスイッチを押すことで発生し、3方向の割り込みレベルよりも上位である。

ステータス入出力信号は3方向入出力として各々3ﾋﾞｯﾄ計18本あり、割り込みの情報等に使用する。またTOP方向のステータス出力3ﾋﾞｯﾄをLED3個を使って外部へ出力するようにしたT&Dパネルは、後で述べるテストプログラムのエラーの表示に使用する。割り込み、ステータス出力回路には、PALを使用している。

バス方向選択、割り込み、ステータス、接続ポートの各回路にはアドレスが割り当てられておりそれをデコードすることでセレクトされる。

プロセッサ要素を図3.2に、接続ポート、TOPステータス出力回路、表示パネル、T&Dパネルの関係を図3.3に示す。

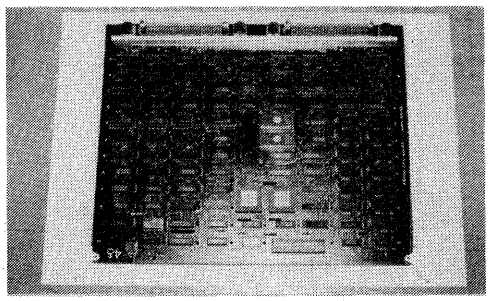

図3.2 プロセッサ要素

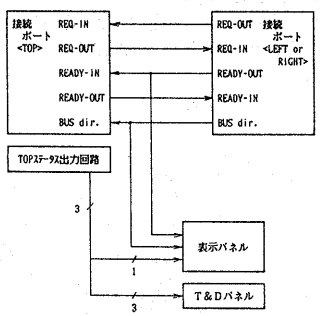

図3.3 接続ポート、表示パネル、T&Dパネルの関係図

４.ホストープロセッサ要素間インターフェース

４.１　インターフェース回路

ホスト計算機には、8ﾋﾞｯﾄパラレル入出力インターフェースがある。しかし、プロセッサ要素は16ﾋﾞｯﾄパラレルである。ホストープロセッサ要素間インターフェース回路とデバイスドライバーで8ﾋﾞｯﾄ-16ﾋﾞｯﾄ変換を行い転送する。

この回路は図4.1に示すバッファと制御回路からなる。ホスト計算機側の入出力ポートコントローラには8255を2個使用している。各々の8255において、ポートAには入力データ線、ポートBには出力データ線、ポートCには制御線がつながっている。制御線にはINT,STRB,C/D,CLR,I/O合計5本ある。INTは入力で、残り4本の制御線は出力である。一方、プロセッサ要素側は、入出力データ線とREQ-OUT,REQ-IN,READY-OUT,READY-IN,BUS direction５本の制御線がつながっている。

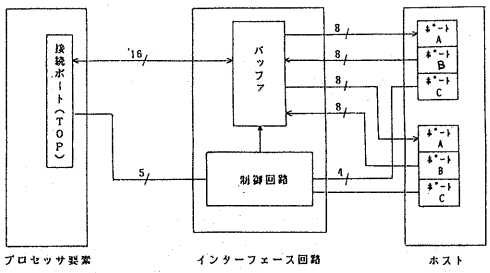

図4.1 インターフェース回路ブロック図

バッファは、ホスト計算機からの8ﾋﾞｯﾄ×2入出力データとプロセッサ要素16ﾋﾞｯﾄ入出力データを蓄える。

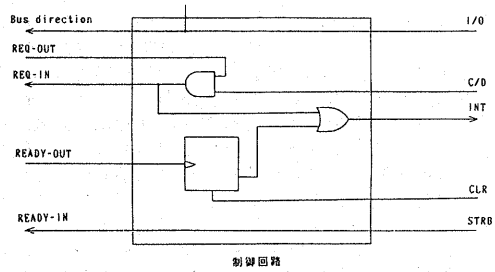

図4.2 制御回路図

制御回路を図4.2に示す。I/O信号は、バッファを制御し、他の信号は転送のために使用される。転送方法を簡単に示す。

1.転送要求が、プロセッサ要素はREQ-OUTからホスト計算機はC/Dから出される。
2.転送要求が、プロセッサ要素はREQ-INからホスト計算機はINTから伝わる。
3.プロセッサ要素ではDMACがバスを確保する。ホスト計算機は割り込み処理ルーチンにとぶ。
4.プロセッサ要素はREADY-IN,READY-OUT、ホスト計算機はINT,STRB,CLRを使ってハンドシェイクを行う。

READY-OUT信号のアクティブ状態は1μ秒と短いため、直接INTより入力しても8255ポートCからの信号が確認できないので、この間にﾌﾘｯﾌﾟﾌﾛｯﾌﾟを入れた。このﾌﾘｯﾌﾟﾌﾛｯﾌﾟを1データ転送毎にクリアするのがCLR信号である。

INTが割り込みとハンドシェイク両方に使われるのは、この信号線1本しかないためである。ハンドシェイクの際、割り込みがかからないような回路にしてある。

４.２　デバイスドライバー

UNIXにおいてはデバイスは、キャラクタ型とブロック型に区別され、それぞれメジャーデバイス,マイナーデバイス番号を持っている。Cora168Kはキャラクタ型

デバイスとして登録している。キャラクタ型デバイスの場合、多くの仕事をドライバー自体がする。

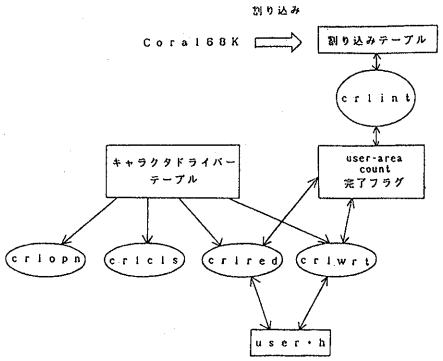

図4.3 入出力関連図

ユーザプログラムにおいて、open、close,read,writeが呼ばれるとメジャーデバイス,マイナーデバイス番号をもとにｷｬﾗｸﾀﾄﾞﾗｲﾊﾞｰﾃｰﾌﾞﾙからドライバー関数crlopn,crlcls,crlred,crlwrtが呼ばれる。（図4.3）

Coral68Kよりデータを受信する場合、初めにopenが呼ばれ、ｷｬﾗｸﾀﾄﾞﾗｲﾊﾞｰﾃｰﾌﾞﾙよりドライバー関数crlopnが呼ばれ、8255の初期設定とCoral68Kからの入力を可能にする。次にreadを呼ぶとcrlredが呼ばれ、user.hよりユーザ入出力アドレスと入出力バイト数をuser_areaとcountに格納する。そしてCoral68Kに転送要求を出して、転送完了フラグが立つまで待ち続ける。Coral68Kは転送要求をだしホスト計算機に割り込みをかける。すると割り込みテーブルよりcrlintが呼び出される。crlintでデータを受信して、転送完了フラグを立てる。crlredは転送完了フラグが立つと入出力アドレスに処理バイト数を加え残りのバイト数を返す。最後にcloseを呼ぶとcrlclsが呼ばれCoral68Kからの入力を受け付けないようにする。

送信の場合も同様に行うが、writeを呼ぶとcrlwrtが呼ばれ転送要求の他にI/O信号線よりライト信号を出す。

crlintでは、次の様にしてリード・ライト動作を行う。

```
read()                          write()
{                               {
  do{                             do{
    check();                        STRB=OFF;
    CLR=ON;                         PB0=user_area++;
    STRB=OFF;                       PB1=user_area++;
    user_area++=PA0;                CLR=OFF;
    user_area++=PA1;                STRB=ON;
    CLR=OFF;                        check();
    STRB=ON;                        CLR=ON;
  }while(count-=2);               }while(count-=2);
}                               }
```

checkルーチンは、INTを見て、OFF状態がある時間続けばエラーが生じたとして強制的にcrlintよりリターンする。read()では、データが送られたことを確認し、write()ではデータを受け取ったことを確認する。これに対して、STRBは、read()ではデータを受け取ったことを伝えwrite()ではデータを送ったことを伝える。

crlintでは、端末，内蔵フロッピーディスクからの割り込みを避けるために割り込みマスクをかけている。このために、他の入出力デバイスのインターラプトルーチンは呼ばれず、Coral68Kとの転送のためだけに機能する。

5.構造

5.1 シャーシ

シャーシ（44cm(w)×35cm(h)×275cm(d)）には16スロットあり、16台のプロセッサ要素が格納でき、このシャーシ4個を使って63台のプロセッサ要素を格納する。

各々のプロセッサ要素の配線にはﾂｲｽﾄﾗｯﾋﾟﾝｸﾞﾜｲﾔｰを使用した。各シャーシでは、15台のプロセッサ要素を木の中間順(inorder)に並べて配線する。これは、TOP,LEFT,RIGHT方向のコネクタが図5.1に示す配置のため中間順は昇順と違ってLEFT,RIGHT方向のプロセッサ要素が自分の左右にあるため、LEFT,RIGHTコネクタ付近の重なりがなく配線し易いためである。また、ﾚﾍﾞﾙ0とﾚﾍﾞﾙ1を配線し、次にﾚﾍﾞﾙ1とﾚﾍﾞﾙ2、最後にﾚﾍﾞﾙ2とﾚﾍﾞﾙ3を配線すれば簡単に行える。

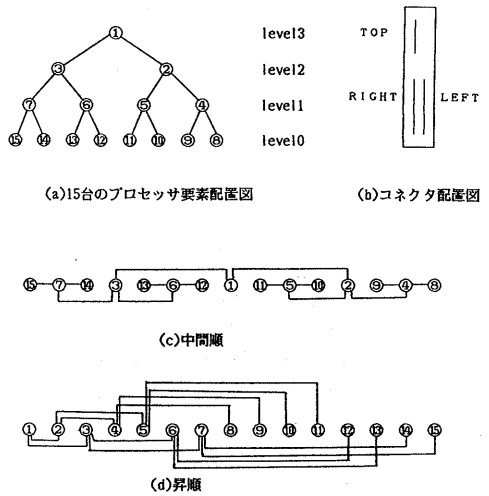

図5.1 コネクタ配線図

全体は図5.2に示す様に配線する。シャーシ4には15台のプロセッサ要素しか格納しないので、余ったスロットに、ホストープロセッサ要素間インターフェースを格納する。

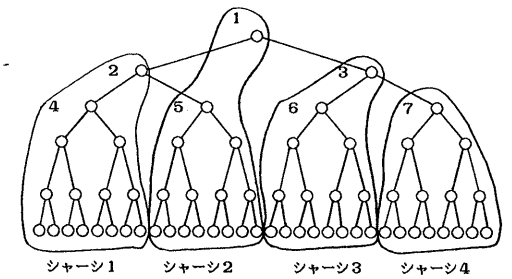

図5.2 全体配線図

5.2 筐体

筐体は、58cm(w)×180cm(h)×85cm(d)の大きさである。この筐体正面扉には表示パネル,電源スイッチ,割

り込みスイッチがある。筐体内には、63台のプロセッサ要素,ファン,電源,ノイズフィルター,表示パネル制御基板,T＆Dパネルが格納されている。

電源,ノイズフィルターは筐体内の扉に取り付け、また表示パネル制御基板は正面扉の裏に取り付けた。T&Dパネル（図5.3）は背面のシャーシ間に取り付けてある。このT&Dパネルの後方のシャーシの間にはファンユニットが入っている。

5.3 消費電力

直流電源は、容量5V×100Aのものを2個使用している。実測によれば全体の消費電流は174Aであり、消費電力は870Wであった。

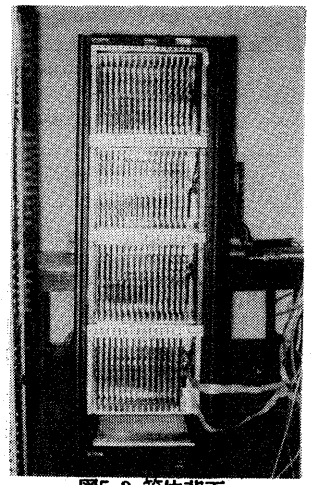

図5.3 筐体背面

6.ソフトウェア開発環境

6.1 IPLとデバッガ

Coral68Kのプログラミング言語はC言語であり、IPL、デバッガ、ランタイムサブルーチンはほとんどC言語で記述している（一部アセンブリ言語で記述している）。Coral68KではIPLとデバッガの処理をホスト計算機からのメッセージを受信することで開始されるようにした。図6.1に状態遷移図を示す。

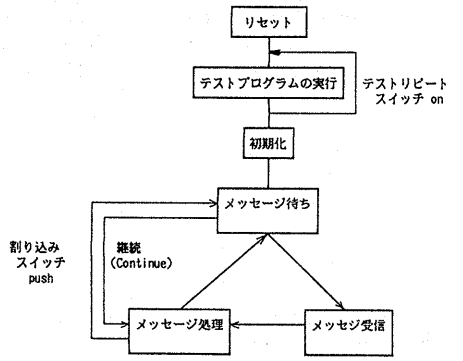

図6.1 IPLとデバッガの状態遷移図

ハードウェアリセット（筐体正面のリセットスイッチを押す）によりCoralがリセットされる。次ぎに割り込み、ステータス、通信のチェックを行うテストプログラムが実行される。エラーはT&Dパネルを使用して表示する。またテストリピートスイッチ（筐体内の半扉に付けている）をonにすることでテストプログラムの繰り返しも可能である。このテストプログラムは、不完全な2進木であっても接続されているプロセッサ要素のみでテストを行える。次いで以下に示す6個のシステムパラメータを初期化する。

- プロセッサ番号
- プロセッサレベル
- LEFTのプロセッサ要素数
- RIGHTのプロセッサ要素数
- 自分自信を含む部分木のプロセッサ要素数
- 現在のシステムのプロセッサ要素のメンバー

初期化が終るとプロセッサ要素はメッセージが受信可能となる。この時表示パネルのプロセッサ要素を示すLEDは点灯している。この状態においてホスト計算機からメッセージが送られると、現在動作しているすべてのプロセッサ要素に放送される。この時下位プロセッサ要素からメッセージ受信の確認信号が返ってこなければ、時間監視で上位のプロセッサ要素がエラーと判断しTOPへエラーコードを返すようにしている。メッセージ処理では、メッセージの中に含まれるコマンドを実行する。実行中は表示パネルのプロセッサ要素を示すLEDは消灯している。この処理が終了すれば再びメッセージ待ちになる。

割り込みスイッチ（筐体正面にある）はユーザプログラムのデバッグ等に使用する。（この割り込み優先度はプロセッサ要素間の割り込みよりも上位である。）このスイッチを押すことによってMPUのレジスタが格納されメッセージ待ちへ移行する。この時のメッセージ待ち、受信、処理は通常と同じであるが、メッセージのコマンドとして継続（Continue）が受信されるとユーザプログラムの実行が継続される。

メッセージは20ﾊﾞｲﾄの固定長であり次の様なコマンドとパラメータで構成される。

ｺﾏﾝﾄﾞ	対象ﾌﾟﾛｾｯｻ要素	ｱﾄﾞﾚｽ	ｻｲｽﾞ
(4B)	(8B)	(4B)	(4B)

◎ コマンド

コマンドは8種類あり内1つを選択する。

①初期化(Initialization)

先に述べた6個のパラメータを初期化する。

②ロード(Load)

存在するすべてのプロセッサ要素にプログラムまたはデータをロードする（指定されたアドレスに直接書き込む）。

③実行(Go)

指定されたプロセッサ要素を指定されたアドレスより実行させる。割り込みスイッチが押された後でも同様である。

④ダンプメモリ(Dump memory)

指定されたプロセッサ要素のメモリの一部をダンプする。

⑤ライトメモリ(Write memory)

指定されたプロセッサ要素のメモリの一部にプログラムまたはデータを書き込む（メッセージデータバッファを用いて書き込む）。

⑥ダンプレジスタ(Dump register)

指定されたプロセッサ要素のMPUレジスタをダンプする。

⑦テスト(Test)
　割り込み、ステータス、通信のチェックを行う（リセット後に行うテストプログラムと同じである）。
⑧継続(Continue)
　MPUレジスタ、ステータス、DMACレジスタをリストアしてユーザプログラムを継続する。

◎　対象プロセッサ要素
8ﾊﾞｲﾄのパラメータで63台のプロセッサ要素をﾋﾞｯﾄ対応で示し、実行、ダンプメモリ、ライトメモリ、ダンプレジスタのいずれかのコマンドを指定した場合必要となる。

◎　アドレス
コマンドの処理アドレスを示し、ロード、実行、ダンプメモリ、ライトメモリ、ダンプレジスタのいずれかのコマンドを指定した場合必要となる。

◎　サイズ
ダンプメモリまたはライトメモリのコマンドを指定した場合のデータサイズを示す。

メモリマップを図6.2に示す。

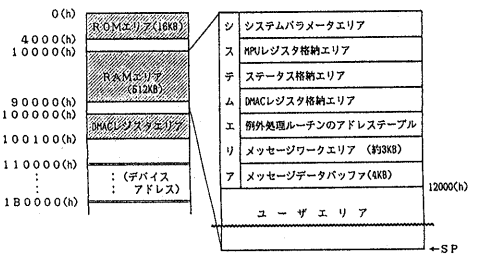

図6.2 メモリマップ

システムパラメータエリアは初期化で述べたパラメータのエリアである。MPUレジスタは、割り込みスイッチを押した後MPU格納エリアに格納される。ステータス、DMACレジスタは、プログラム実行中で更新されるごとに格納される。例外処理ルーチンのアドレステーブルは、割り込み等の処理ルーチンアドレスを設定しておくテーブルである（ユーザが設定できる）。メッセージワークエリアは、メッセージを格納するエリアであり、メッセージデータバッファは、メッセージ処理（ダンプメモリ、ライトメモリ）でデータ転送用バッファとして使用する。

またホスト計算機上でホストプログラムを一時停止したり、継続したりすることが難しいので、デバッグする場合ホストのプログラムを完全に停止させ新たにホスト上でデバッガを起動させてからCoralのプログラムをデバッグすることになる（そのためホストとCoral間の通信が頻繁なプログラムのデバッグは難しくなってくる）。

6.2　プログラム作成法

ホストの実行ﾌｧｲﾙ、Coralの実行ﾌｧｲﾙは次のようにして作成する。

（例）ホストのｿｰｽﾌｧｲﾙ：host.c　実行ﾌｧｲﾙ：host
```
cc host.c -o host -lhr -lm -lc
```
（例）Coral のｿｰｽﾌｧｲﾙ：crl.c　実行ﾌｧｲﾙ　：crl
```
cc -c crl.c
ld -f 12000 -o crl crl.o -lcr -lm -lc
```
hr cr は、ホスト、Coralのランタイムサブルーチンを示している。これらはunixのライブラリlibhr.a libcr.aとして/libの下にリンクしている。またホストの実行ﾌｧｲﾙはccによって実行開始アドレスが0x200000に指定される。Coralではユーザエリアが0x12000からなのでld（リンクエディタ）で先頭アドレスを0x12000にしている。

6.3　ランタイムサブルーチン

Coralでプログラムをインプリメントするためのサブルーチンとしてホスト計算機とCoralのランタイムサブルーチンを用意した。それらを表6.1,表 6.2に示す。

表6.1 ホスト計算機のランタイムサブルーチン

ルーチン名	機能
crlopen() crlclose()	Coralをオープンまたはクローズする
crlputh(adr) crlgeth(adr)	2Bytesデータを送信また受信する
crlputi(adr) crlgeti(adr)	4Bytesデータを送信また受信する
crlputbk(adr,size) crlgetbk(adr,size)	複数Bytesのデータを送信または受信する
load(filename,adr)	Coralにプログラをロードする
go(adr)	Coralのプログラムを実行する

表6.2 Coral68Kのランタイムサブルーチン

ルーチン名	機能
intrpt(dir)	dirで示された方向に割り込みをかける
insts(dir)	dirで示される方向のステータスを読む
outsts(dir,signal,cnt)	dirで示される方向にステータスを出力する
puth(adr,dir) geth(adr,dir)	2Bytesのデータを送信または受信する
puti(adr,dir) geti(adr,dir)	4Bytesのデータを送信または受信する
putbk(adr,size,dir) getbk(adr,size,dir)	複数Bytesのデータを送信または受信する

7.性能評価

7.1　プロセッサ要素の演算処理能力

演算処理能力を、汎用大型計算機FACOM M360、Coral'83と比較する（表7.1）。これは固定小数点（M360は32ﾋﾞｯﾄ,Coral68K,Coral'83は16ﾋﾞｯﾄ）、浮動小数点（Coral68K,M360は64ﾋﾞｯﾄ,Coral'83は32ﾋﾞｯﾄ）で四則演算を繰り返した実行時間を基にMIPS,MFLOPSを求めた。

表7.1 演算処理能力

	MIPS	MFLOPS
Coral68K(1PE)	0.8	0.05
Coral68K(63PE)	40	0.3
FACOM M360	2.67	0.64
Coral'83	0.7	0.01

Coral68Kは、63組のプロセッサ要素が80%の利用率があったとして計算で求めた。

M360と比べ、MIPSは15倍、MFLOPSは半分である。これは、数値演算プロセッサがないためである。

7.2　転送速度

プロセッサ要素間、ホストープロセッサ要素間の転送速度を測定した。

初めに、プロセッサ要素間の転送速度と転送語数の

関係を図7.1に示す。

転送速度は、最高2MByte/secである。しかし転送語数の大きさによって大きく変化する。これは転送時間に占めるセットアップ時間の割合が少なくなるためである。

２進木構造であるため、1番のプロセッサ要素から63番のプロセッサ要素までファイルをloadするには10回の転送で済む。256KByteのファイルをloadした場合1.3秒で行える。

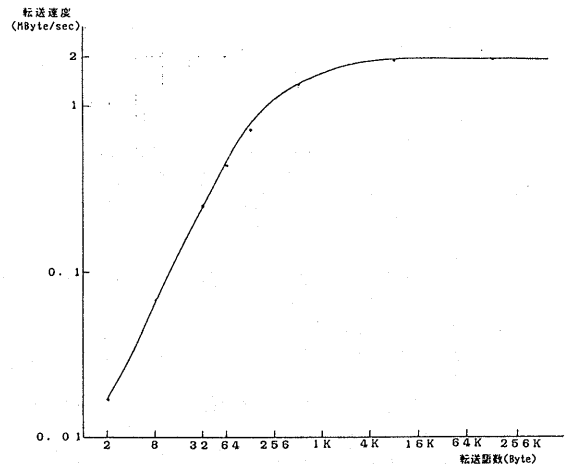

図7.1 プロセッサ要素間転送速度

次に、ホストープロセッサ要素間の転送速度と転送語数の関係を図7.2に示す。

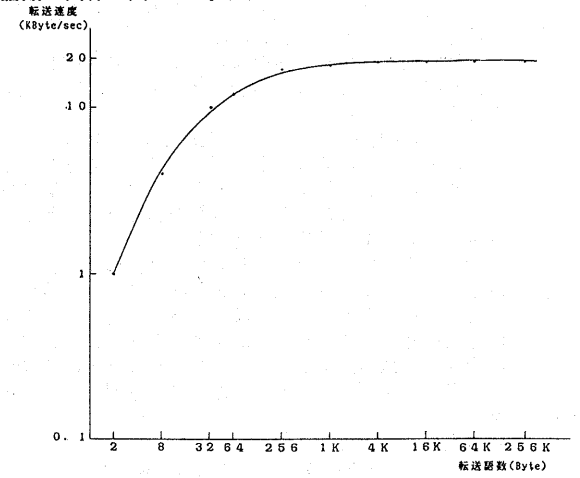

図7.2 ホストープロセッサ要素間転送速度

転送速度は、最高20KByte/secである。これも転送語数によって大きく変化している。これは、転送時間に占めるインターラプトルーチンを呼ぶまでの時間の割合が少なくなるためである。

ホストープロセッサ要素間の転送速度は、プロセッサ要素間の1/100である。ホスト、Coral68K間で転送の頻繁な問題ではボトルネックとなる。

７．３　並列処理の性能

いくつかの並列処理プログラムを作成しCoral68Kで実行して並列処理の効率を測定した。

(1)ナップザック問題

m個ある品物の内の幾つかをナップザックで運ぶ場合、ナップザックの収容できる総重量以下で総利益が最高になる品物の選び方の問題である。これは、各プロセッサ要素において各自解を得て、左右のプロセッサ要素から送られた解と各自求めた解を比較し、送られた解が最適であれば送られた解を、自分の解が最適であれば自分の解を上位のプロセッサ要素に送る。22個の品物で総重量が700の場合の実行結果を表7.2に示す。

表7.2 ナップザック問題実行結果

PE数	1	3	7	15	31	63
整数　(秒)	400	166	68	32	15	7
実数　(秒)	1893	791	322	151	74	36

このアルゴリズムで求めた場合、利用率がプロセッサ要素の数にかかわらず80%以上あるために図7.3に示す様に速度向上率がリニアーに近い。これは、負荷分散が良く、転送回数が少ないためである。総重量,総利益の演算を整数(32ﾋﾞｯﾄ)と実数(64ﾋﾞｯﾄ)で比較すると4倍違う。数値演算プロセッサが搭載されていれば、実数演算も良い結果が得られたと考えられる。

M360において、Pascalを用いて解を得た。整数で271秒、実数では293秒であった。Coral68Kは、M360と比較して整数(32ﾋﾞｯﾄ)では40倍近く、実数(64ﾋﾞｯﾄ)では8倍になっている。これは、Pascalを用いたために実行時間がかかっていると考えられる。

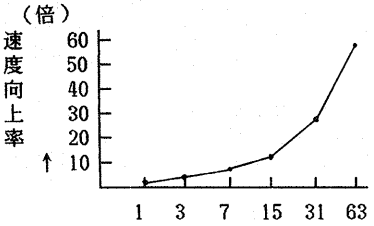

図7.3 ナップザック問題の速度向上率

(2)　ヒープソート

ソートするデータを各プロセッサ要素に等分配し、各プロセッサ要素で独立にヒープソートをした後、左右のプロセッサ要素から送られてデータとマージして上のプロセッサ要素に送る。この様にして得た実行結果を表7.3に示す。データ数は156Kﾜｰﾄﾞ(16ﾋﾞｯﾄ)である。

表7.3 ヒープソートの実行結果

PE数	1	3	7	15	31	63
実行時間　(秒)	177	68	38	27	23	22
速度向上率　(倍)	1	2.6	4.7	6.6	7.7	8.1
利用率　(%)	100	88	67	44	25	13

ヒープソートは完全に並列に行われるが、マージの並列度が上のレベルのプロセッサ要素になるほど減少するために効率はそれほど良くない。

Coral68KとCoral'83との比較を行った。（表7.4）

表7.4 Coral68KとCoral'83の比較

PE数	1	3	7	15
データ数 (ﾜｰﾄﾞ)	5120	14976	34994	65535
Coral68K (秒)	4	5	8	11
Coral'83 (秒)	81	93	103	107

15台Coral68Kは、Coral'83に対して10倍以上の性能が得られた。

M360との比較では、64Kﾜｰﾄﾞの場合Coral68K(16ﾋﾞｯﾄ)では8秒でM360(32ﾋﾞｯﾄ)では5秒であり、0.6倍である。これは、並列度があまり上がらなかったためであろう。

(3) ガウス・ジョルダン法

ｎ元連立一次方程式においてｎ×（ｎ＋１）マトリクスを各プロセッサ要素に等分し、ピボットを行う行を全てのプロセッサ要素に放送し解を得る。126元の結果を表7.5に示す。

表7.5 126元の実行結果

PE数	1	3	7	63
実行時間 (秒)	320	154	80	16
速度向上率 (倍)	1	2	4	20
利用率 (%)	100	69	57	32

利用率が32%しかない。これは、126元のため各プロセッサ要素には 2行のデータしかなく放送が頻繁に行われ、本来の演算が少ないためであると考えられる。元数が大きくなれば放送するデータサイズは大きくなるが各プロセッサ要素内の演算が増え利用率が高くなると考えられる。

Coral'83(32ﾋﾞｯﾄ)において120元で解いた結果722秒かかった。これとCoral68K(64ﾋﾞｯﾄ)で126元で解いた結果と比較すると45倍ある。同じ元数であれば45倍以上と考えられる。

630元についてCoral68KとM360で実測した結果、Coral68K(64ﾋﾞｯﾄ)では1237秒、M360(64ﾋﾞｯﾄ)では382秒であった。M360/Coral68Kは0.31で表7.1のMFLOPS比に近づいている。

(4) 巡回セールスマン問題

巡回セールスマン問題は、最小の距離のハミルトン閉路を求める問題である。バックトラック法で解く場合、既に見つかっているハミルトン閉路の距離より長ければ、バックトラックして他の閉路を探索する。

Coral68Kでは、最初の 3都市の異なる回り方を各プロセッサ要素に割当て、各々で異なるハミルトン閉路を探索する。10都市を探索した結果を表7,6に示す。

表7.6 巡回セールスマン問題の実行結果

PE数	1	3	7	15	31	63
実行時間 (秒)	2555	859	403	184	99	53
速度向上率 (倍)	1	2.9	6.3	13.9	25.8	48.2
利用率 (%)	100	99	90	93	83	77

この問題も並列度が高いので速度向上率が48倍にもなっている。この際の利用率を図7.4に示す。

この問題は、都市数により63台使用しても1台の3倍くらいの速度向上率であったり、63台全て使うより15台使って解いた方が速いことがある。これは、各プロセッサ要素が各自見つけた最小距離をもとにしてバックトラックを行っていくので、1台の場合に比べ、無駄な探索を行うためである。

M360(32ﾋﾞｯﾄ)ではPascalで解いてみた結果480秒かかった。9倍の性能が得られた。

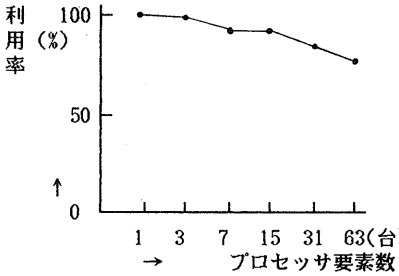

図7.4 巡回セールスマン問題の利用率

８.結論

以上の様に、処理能力、転送速度、メモリ容量で改良されたCoral68Kは、Coral'83に比べ約40倍前後の性能向上が得られた。

また今回開発したシステムでは、次の問題点、改良点が挙げられる。

① DMACを採用したが、少数転送語数(2,4ﾊﾞｲﾄ)で、は最大転送速度に比べて非常に遅い（セットアップ時間との割合に依存している）。FIFOメモリを使用すればよいのではないかと考える。

② ﾚﾍﾞﾙ0のプロセッサ要素32台のLEFT、RIGHT接続ポート(64ポート)には現在はなにも接続されておらず無駄になっている。これは多チャンネルの並列入出力回路への利用などがが考えられる。

③ 経済的な問題から数値演算プロセッサ(MC68881等)を付加しなかったが、やはり浮動小数点演算では不利である。

④ T&Dパネルや表示パネルの制御のための外付け回路をプロセッサ要素の基板上に設けておくべきであった。

⑤ 現在ホスト計算機のハードディスクの容量は20MBであるが、それに対しCoral68K側が約32MBということでホストのメモリ不足が問題である。

尚、このプロジェクトは4年がかりで行われたものであり、この間ブレッドボードの設計、製作、実験などで苦労された藤本和生、桑原昭、吉谷文徳、山本英彦の諸氏に感謝する。またアジアエレクトロニクス㈱作田副部長、黒田氏にはプリント基板や筐体の設計製作等に多大な協力と助言を頂いた。深く感謝する次第である。

参考文献

1）Yoshizo Takahashi,Yoshitaka Yamane et al.: "Efficiency of Parallel Computation on the Binary-Tree Machine CORAL'83",JIP,vol.8,No4, p288～299,March,1986

2）藤本和生，桑原昭，高橋義造：68000による2進木構造並列処理システムCoralプロトタイプの設計，情報処理学会第29回全国大会5B-2(1984)

3）松尾賢二，遠藤俊雄，吉谷文徳，高橋義造：2進木構造並列処理システムCoral68Kのプロセッサ要素の開発，情報処理学会第33回全国大会3C-3(1986)