PIM/mのメンテナンス系アーキテクチャとCSP

6L−4

高橋勝己[1], 武田保孝[1], 村沢靖[1], 小森隆三[1], 杉野栄二[2]

1: 三菱電機(株) 2: 北陸先端科学技術大学院大学

1 はじめに

並列推論マシン『PIM/m (Parallel Inference Machine / model M)』は、第五世代コンピュータプロジェクトの一環として開発を行なったものであり、先に開発した並列推論マシンのプロトタイプである『マルチPSI』[1]の後継機として位置付けられている。従って、PIM/mの構成が2次元格子状であることや、各要素プロセッサがCISCタイプであることなどの基本設計はマルチPSIを継承したものとなっている。しかし、PIM/mでは、要素プロセッサの性能向上やプロセッサ数の増大に対応するために、パイプライン・アーキテクチャの採用やRefuge Stackの導入といったいくつかの改良を行なっている[2]。

『CSP (Console System Processor)』は、このPIM/mの立ち上げや、異常発生時の処理、ファームウェア/ソフトウェアのデバッグ支援など、PIM/mの開発や運用を行なうためのシステムである[3, 4]。このCSPは、マルチPSIのCSPをPIM/mに合わせて改良したものである。

本稿では、PIM/mのCSPにおける要素プロセッサ内部の資源アクセス方法など、PIM/mの各要素プロセッサのメンテナンスを行なう機能について報告する。

2 PIM/mのハードウェア構成

PIM/mは、推論の高速処理のために専用に開発された推論型の要素プロセッサ(PE)を2次元格子状の高速ネットワークで接続した疎結合型並列マシンである。このマシンは1筐体32PE (8 × 4)を単位として最大8筐体256PE(16 × 16)まで、任意の大きさの長方形で、構成することができる。

図1は、PIM/mの運用時のシステム構成を示したものである。PIM/mの入出力機能は、SCSIによって接続されたPSI-IIによって提供され、FEP(Front-End Processor)と呼ばれる。PIM/mでは、8PEごとにSCSIが装備され、FEPだけでなく磁気ディスクなど様々な周辺機器を接続することができる。

PIM/mの立ち上げ、異常処理やPEの内部資源へのアクセスといった、PIM/mのメンテナンス機能は、メンテナンスバスで接続されたCSPによって提供される。通常はFEPとして用いられるPSI-IIの1台をCSPとして兼用する。

CSPと各PEとを接続するメンテナンスバスは、CSPがマスタ、PEがスレーブといった、シングルマスタ/マルチスレーブの低速バスであり、8bitのデータアドレス共用線と、幾つかの信号線からなっている。

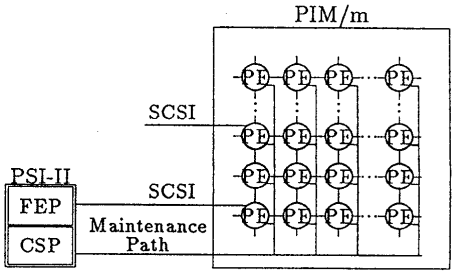

図1: PIM/mの構成

PIM/mのメンテナンス機能はこのバスに対して、CSPがコマンドを送ることによって、実現されている。

3 CSPの主機能

CSPは、各PEに用意されたPEの起動や停止、エラー情報の保持や受渡しを行なうレジスタを用いて、個々のPEを制御するためのものであり、以下のような機能を持つ。

1. PIM/mの立ち上げ
   電源ON/OFFを含むシステムの立ち上げと初期化
2. 異常発生時の処理
   異常時のメッセージ表示や、PE停止の検出・報告と処置
3. プロセッサ構成の変更
   PEが故障した場合に、そのPEを除いたシステムの再構成を可能にする
4. PE内部資源(レジスタなど)の表示/変更
5. デバッグ機能
   PIM/mでの実行レベルは上位から、KL1、KL1-B命令、マイクロ命令の3つがある。KL1は、ユーザーのプログラムレベルであり、KL1-B命令とは、それをコンパイラにかけた結果、出力される命令列である。マイクロ命令はKL1-Bの各命令を1つずつ解釈実行するものである。CSPは、このそれぞれの実行レベルに対して、次のようなデバッグ機能を持つ。

   (a) マイクロ命令レベル
   ステップ実行/連続実行や指定マイクロアドレスによるブレイク、トレースなど

   (b) KL1-B命令レベル
   ステップ実行/連続実行や設定条件によるブレ

Maintenance Architecture and CSP on PIM/m
Katsumi Takahashi[1], Yasutaka Takeda[1], Yasushi Murasawa[1], Ryuzo Komori[1], Eiji Sugino[2]
1:Mitsubishi Electric Corporation. 2:Japan Advanced Institute of Science and Technology, Hokuriku.

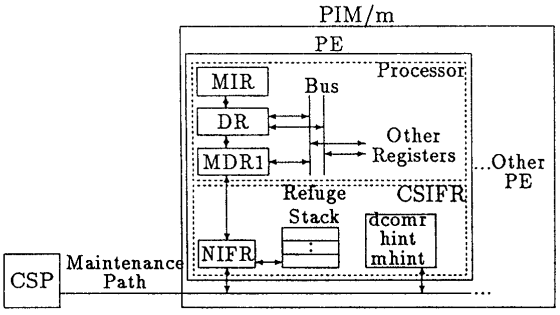

図 2: PE 内部のメンテナンス系の構成

イク。マイクロプログラムとの連係により実現

(c) KL1 命令レベル
スパイやトレース、インスペクタなど。同様に、マイクロプログラムとの連係により実現

4 CSP による PE 内部資源へのアクセスとその高速化

4.1 1 つの PE に対する内部資源へのアクセス

図 2 に PE 内部のメンテナンス系の構成を示す。ここで、CSP が直接アクセスできるのは、CSIFR (Console System InterFace Register) と呼ばれる、PE の起動や停止、エラー情報の保持をおこなうレジスタ群と 3 つのレジスタ (MIR, DR, MDR1) のみである。これ以外のレジスタへのアクセスは、この 3 つのレジスタを経由して行なわれる。

CSP が DR を経由してレジスタへの書き込みを行なう場合を例に、レジスタへの書き込み手順について述べる。この場合 CSP は、MIR に DR の値を目的のレジスタに書き込むマイクロ命令をセットし、DR に書き込み値をセットした後、1 ステップの実行を行なうことで、そのレジスタへの値の書き込みを実現している。この時、CSP がセットするマイクロ命令は、NIFR や MDR1, DR といったレジスタの内容を破壊しながら、MIR へ送られる。このため、PIM/m ではその破壊される内容を退避させておくためのスタック、『Refuge Stack』が用意されている。

この Refuge Stack の効果は、構成 PE 台数の多い場合に発揮されるが、これに関しては次節で述べる。

CSP からある PE のレジスタの書き込みを行なう手順を以下に示す。

1. PE の指定
2. 破壊されるレジスタ値の退避
3. レジスタに値を書き込むマイクロ命令のセット
4. 書き込む値のセット
5. マイクロ命令の 1 ステップ実行
6. 破壊されたレジスタ値の復元

CSP から行なう PE 内部資源への書き込みは、この手順の 2 ～ 6 を繰り返すことで実現している。CSP から行なう PE 内部資源への読み出しも、レジスタ値を読み出すマイクロ命令をセットすることで、同様に行なうことができる。

4.2 複数 PE に対する内部資源へのアクセス

PIM/m の立ち上げ時には、パラメータの初期値設定など、複数の PE に対して同じ操作を行なう処理がほとんどである。このため、CSP は複数の PE に同時にアクセスするブロードキャスト機能を持っている。

PIM/m のプロトタイプであるマルチ PSI の CSP にも同様の機能は存在したが、マルチ PSI では Refuge Stack が用意されていなかったため、PE の内部資源へのアクセスの際に破壊されるレジスタの内容は、CSP が PE ごとに読み出し CSP 上に保存しなければならなかった。このため、マルチ PSI での複数 PE の内部資源への書き込みは、PE 数に比例する時間を必要とした。

一方、PIM/m では、各 PE が Refuge Stack を持っているため、破壊されるレジスタの退避 / 復元といった操作が、各 PE ごとに独立かつ同時に実行できるようになった。従って複数 PE の内部資源への書き込みは、PE 数に関係なく一定時間で終了することができる。

このように Refuge Stack を用いることにより、プロセッサ数の増加による立ち上げ時間の増大を防いでいる。

5 おわりに

本稿では、PIM/ みおけるメンテナンス系アーキテクチャについて、特に CSP による PE 内部資源へのアクセスの高速化に焦点をあてて報告を行なった。この高速化では、各 PE に Refuge Stack を持たせることで、PE 数に比例しない、一定時間による PE 内部資源への書き込みが行なえるようになった。

また、PIM/m ではこれと合わせて、マルチ PSI の CSP が行なっていた各種情報の生成を各 PE のマイクロプログラムによる生成に切替えることで、立ち上げ時間を大幅に短縮することを可能にした。PIM/m では、この 2 つの改良によって、マルチ PSI (64PE 版) で約 10 分かかっていた立ち上げ時間が、256PE の場合でも 3 分程度で終了できるようになっている。

参考文献

[1] Y.Takeda,H,Nakashima,K.Masuda,T.Chikayama and K.Taki: "A Load Balancing Mechanism for Large Scale Multiprocessor Systems and Its Implementation." Proc.Intl.Conf.on Fifth Generation Computer Systems 1988(1988)

[2] 中島、武田、中島: 『PIM/m 要素プロセッサのアーキテクチャ』, JSPP'90.

[3] 中島、田辺、瀧: 『マルチ PSI におけるデバッグ機能・メンテナンス機能』, 第 29 回情報処理学会全国大会論文集 1B-9、1984

[4] 杉野、古市、稲村、瀧: 『マルチ PSI におけるデバッグ機能・メンテナンス機能』, JSPP'89.