

## 推論プロセッサ UNIREDIIIの多重コンテキスト処理

1 L - 9

島田 健太郎, 小池 汎平, 田中 英彦  
 東京大学 工学部

### 1 初めに

我々は現在並列推論マシン PIE64[1] の製作を進めている。PIE64では、対象言語として Committed Choice 型言語 FLENG、及びその上位言語として並列オブジェクト指向言語 FLENG++ で書かれたプログラムを実行する。PIE64は64台の推論ユニット (Inference Unit: IU) を2系統の自動負荷分散機能を持つ回線交換ネットワークで結合した構成を採る。各IUにはネットワークと接続し FLENG 向きの高度な通信機能を提供する NIP(Network Interface Processor)[2]、IU内での FLENG の実行処理を行なう推論プロセッサ UNIREDI、及び全体の管理を行なう汎用プロセッサ SPARCがある。我々はこの推論プロセッサ UNIREDI について、先に第2版のアーキテクチャ (UNIREDIII) の設計を行ない [3]、また FLENG を効率良く実行するための命令セットの設計も行なった [4]。現在具体的な回路設計を進めているが、ここではそのハードウェア構成の概要、及びアーキテクチャ上の大きな特徴である多重コンテキスト処理機構の実現について述べる。

### 2 推論プロセッサ UNIREDIII

推論プロセッサ UNIREDIII は並列マシンの要素プロセッサとして、Committed Choice 型言語 FLENG を実行するために設計された専用プロセッサである。並列推論マシン PIE64 の推論ユニット (IU) において、ネットワーク・インターフェース・プロセッサ (NIP)、管理プロセッサ (SPARC) と協調して動作する。主な特徴としては以下のようなものが挙げられる。

- 記号処理向きの機能としてタグ・アーキテクチャを採用、FLENG の処理を効率化している。
- 命令バス、メモリ読み出しバス、メモリ書き込みバスの三つの独立したメモリバスを持ち、バンド幅の広い並列メモリ・アクセスを行なう。
- 多重コンテキスト処理によって動的なパイプライン充足率の向上を行なう。
- ヒープ・メモリを管理するためのレジスタ・セットを持ち、ガベージ・コレクションを支援する機能を備えている。
- コプロセッサ・コマンドバスを持ち、NIP、SPARC との間でのコマンド / リプライの通信プロトコルをサポートしている。
- FLENG の処理を効率化する強力な命令セットを持つ。

UNIREDIII の命令は、すべて1ワード (32ビット) で構成されており、パイプラインの1スロットで実行される。

#### Multi-Context Processing Mechanism of the Inference Processor UNIREDI

Kentaro SHIMADA, Hanpei KOIKE, and Hidehiko TANAKA  
 University of Tokyo

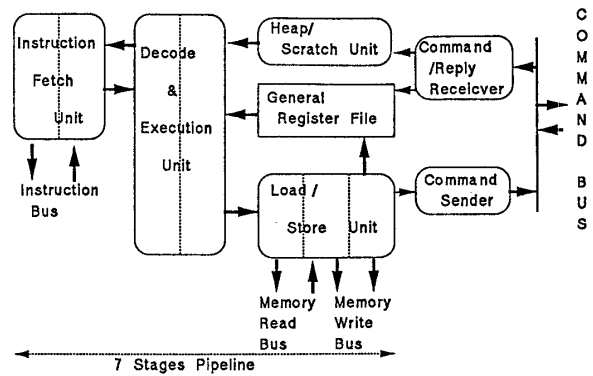


図1: UNIREDIIIの内部ブロック構成

### 3 ハードウェア構成の概要

UNIREDIII は効率化のために内部でパイプライン構成を採っている。パイプラインの段数は7段である。これは、PIE64ではIU内メモリをNIP、SPARCとバス接続で共有することからメモリ・アクセスがアービトレーション・フェーズとデータ転送フェーズの2段になっていることと、メモリに対する Read-Modify-Write 型の処理をパイプラインの1スロットで実現するために、メモリの読み出しと書き込みが別々のステージで行なわれていることによる。即ち1段目と2段目において命令フェッチが行なわれ、3段目で命令デコードとレジスタ読みだし、4段目が実行、5、6段目でメモリ読み込み、6、7段目でメモリ書き込みが行なわれている。

UNIREDIIIの内部ブロックを図1に示す。ブロック構成上の特徴は、NIP、SPARCと通信するコプロセッサ・コマンドの送受信ブロック、ヒープ管理を行なうヒープ・レジスタ/制御ブロックなどを持っていることである。また、次節で述べる多重コンテキスト処理のために、IFU (Instruction Fetch Unit) にはコンテキスト数分のプログラム・カウンタ、各コンテキストの状態レジスタなどがある。命令単位のスケジューリングを行なう機能も IFU 内に設けられている。

### 4 多重コンテキスト処理

#### 4.1 多重コンテキスト処理の目的

UNIREDIII において多重コンテキスト処理は、次の目的のために導入されている。

1. IU 間の並列動作時の同期処理におけるコンテキスト・スイッチングのコストを低減する。
2. パイプライン化された命令の実行において、パイプライン・ハザードの回避を行なう。

並列動作時の同期処理については、例えば PIE64 では他 IU 上のメモリは NIP にコプロセッサ・コマンドを発行すること

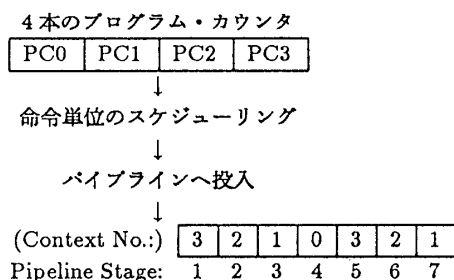


図 2: 多重コンテキスト処理の流れ

によりアクセスされるが、それには UNIREDI のパイプライン・サイクルに比べて長い時間がかかる。このような時に有効な手段としては別に実行可能なコンテキストを用意しておく、そちらへ実行を切り替えることが考えられるが、そのコストが大きいと効果が少ない。そこであらかじめ実行可能な複数のコンテキストで各パイプライン・ステージを満たしておくことにより、一つのコンテキストが同期待ちになって停止しても次のクロックには小さなコストで別のコンテキストの処理を続けられるようにしたものである。

またパイプライン・ハザードについては、一つのコンテキストだけではジャンプ命令の実行によるパイプライン・ブレイク、あるいはメモリからロードしたデータを直後の命令で参照した場合に問題となる。UNIREDI ではこのような場合にもコンテキスト間の並列性を利用してパイプラインを充足することにより、等価的にハザードの全くないパイプラインを実現している。

#### 4.2 多重コンテキスト処理の機構

多重コンテキスト処理において、コンテキスト毎にプログラム・カウンタ、状態レジスタ、汎用レジスタ、フラグ・レジスタ等が別々に取られる。UNIREDI では、ハードウェア規模から同時に処理できるコンテキスト数を 4 としている。

図 2 に UNIREDI における多重コンテキスト処理の流れを示す。各クロック毎に状態レジスタによって実行可能なコンテキストが選ばれ(スケジューリングは後述)、そのプログラム・カウンタの値で命令をフェッチするようにパイプラインに投入される。各パイプライン・ステージでコンテキストは可能な限り別のものが投入されるので、各コンテキストは定常的には図のように順番に実行される(図は四つのコンテキストがすべて実行可能な状態)。また、パイプライン中で汎用レジスタ等はレジスタ番号とコンテキスト番号の組で指定されることになる。

以上のようにして、UNIREDI は言わばパイプライン共有型の MIMD プロセッサとして動作する。これは、FLENG が並列言語であることを積極的に利用して、単一のプロセッサの上でもその並列性を複数のコンテキストという形で抽出したものと考えることもできる。

#### 4.3 多重コンテキストのスケジューリング

多重コンテキスト処理においては、各コンテキストのパイプライン投入のスケジューリングをどのように行なうかは大きな問題である。特に UNIREDI ではハードウェア・コスト

の小さい方法で実現することも重要である。ここで、スケジューリングに対する要求は次の二点である。

- 実行可能なコンテキストがある限り、パイプラインを空けない。
- 同一のコンテキストはなるべく離して投入する。

第一点目は、パイプラインをできるだけ遊ばせないと言うことを意味している。第二点目は、パイプライン・ハザードを減少させるための条件である。

このスケジューリングの問題は、一般的には過去に実行したコンテキストの履歴に基づいて次のコンテキストを決定する最適化問題である。これには同時に実行可能な最大のコンテキスト数分の履歴を用いれば充分であるが、それでも状態数は少ない。例えばコンテキスト数が最大 4 のとき過去の履歴による状態数は 65 である。そこで UNIREDI では状態数を減らすために、コンテキスト番号の昇順に次に実行可能なものをスケジューリングすると言うことを行なう。このようにすると、今実行したコンテキストによって過去の履歴が一意に定まるので、状態数がコンテキスト数+1(実行可能なものがない状態)となり、コンテキスト数が 4 であれば 5 状態で済む。またコンテキストの実行を固定した順序に制限したことによって不利となるのは、実行可能なコンテキスト数が変化した時である。例えば、新しく実行可能なコンテキストが生じた時、先の第二点目の要求条件によればそれをすぐに実行した方が良いが、UNIREDI におけるスケジューリングでは割り当てられたコンテキスト番号に対応する順番が回ってくるまで他のコンテキストが実行される。しかしそれ以外、実行可能なコンテキスト数が変わらない定常状態では、このように制限したことによる差異は生じない。

#### 5 終りに

推論プロセッサ UNIREDI のハードウェア構成の概要、及び多重コンテキスト処理の実現について述べた。UNIREDI は CMOS ゲートアレイ(富士通製 SOG タイプ)として実現され、総端子数 256 ピンの PGA パッケージ(信号数 212 本)に実装される予定である。今後の課題としては、チップ完成後の効果的なデバック / メンテナンス法の検討、プロセッサ・アーキテクチャの評価、及び PIE64 における 64 台の並列動作の評価が挙げられる。

なお、本研究は文部省特別推進研究 No.62065002 による。

#### 参考文献

- [1] Koike, H. and Tanaka, H.: "Multi-Context Processing and Data Balancing Mechanism of the Parallel Inference Machine PIE64" Proc. of Fifth Generation Computer Systems, Tokyo, Japan, November 1988.
- [2] 清水, 小池, 島田, 田中: "並列推論マシン PIE64 のネットワーク・インターフェース・プロセッサ" 並列処理シンポジウム JSPP '89, 情報処理学会, 1989 年 2 月
- [3] 島田, 下山, 清水, 小池, 田中: "推論プロセッサ UNIREDI のアーキテクチャ" 情報処理学会計算機アーキテクチャ研究会 77-2, 1989 年 7 月
- [4] 島田, 下山, 清水, 小池, 田中: "推論プロセッサ UNIREDI の命令セット" 情報処理学会計算機アーキテクチャ研究会 79-5, 1989 年 11 月