計算機アーキテクチャ 89−21
（1991. 7. 19）

実時間用並列処理計算機 CODA-r のアーキテクチャ

西田健次*、戸田賢二、坂井修一、 平木敬†、島田俊夫
電子技術総合研究所
〒305 茨城県つくば市梅園 1-1-4

CODA-r はセンサフュージョンシステムの中核となることを目指した実時間用並列処理計算機である。CODA-r は同型のプロセッシングエレメントをパケット通信ネットワークで結合した形態をとり、高い処理効率とスケーラビリティを両立することを目指している。CODA-r のプロセッシングエレメントは、32bit の優先度フィールドを持つことでデッドライン時刻を直接優先度として扱うことが可能である。これにより優先度の動的管理が不要となり効率的な実時間処理を行なうことができる。

The Architecture of a Real-Time Paralell Processor CODA-r

Kenji NISHIDA, Kenji TODA, Shuichi SAKAI,
Kei HIRAKI, Toshio SHIMADA
Electrotechnical Laboratory
1-1-4 Umezono Tsukuba-shi, IBARAKI 305, JAPAN

This paper presents CODA-r, a parallel processor architecture for real-time applications. CODA-r consists of multiple identical processors connected via a packet communication network. A processor element for CODA-r integrates computation and communication to a processor architecture, while prioritizing every essential operation. The packet communication network consists of prioritized multistage routers, which can relieve priority inversion. CODA-r provides a 32-bit priority field to handle deadlines as a priority to reduce the overheads for priority management. Thereby, CODA-r provides effective hardware support for building real-time systems.

*Email: nishida@etl.go.jp
†東京大学理学部情報科学科併任

1 はじめに

将来の高度情報処理システムにおいては、外界の認識及び判断を含んだ高度な自律性が要求されることが予想される。このような要求に対して、膨大な数の異種センサからの情報を統合し、従来にない新たなセンシング機能を付与し、認識 / 判断等の高度な処理に対する支援を行なおうとするものとしてセンサフュージョンシステムが提案されている [2]。

センサフュージョンシステムを支える計算機アーキテクチャは、外界の情報を得るため多数のセンサからの情報を処理するための高い I/O 能力と実時間性、そして認識 / 判断などを高速に行なうための高い計算能力の両立が必要となる。従来より、高い I/O 能力と計算能力を両立させるためには並列処理が有望であると考えられてきたが、並列処理に特有のプロセッサ間通信やプロセススケジューリングのオーバヘッドにより処理時間の予測可能性が損なわれるため、実時間処理への適用は困難であると考えられてきた。

本稿ではプロセッサのアーキテクチャレベルで実時間性を保証する支援を行なうことで、並列プロセッサによる実時間処理を行なうアーキテクチャを提案する。

並列プロセッサとしてみた場合、CODA-r は高速な同期機構を有し通信と計算能力をプロセッサに統合したものである。実時間性を保証するための機能としては、充分に広い (32bit) 優先度フィールドを持つことでスケジューリング時刻を直接優先度として扱うこと、プロセッサ内部に優先度キューを持つこと、通信ネットワーク上で優先度にしたがったルーティングが行なわれる [4] ことなどがあげられる。

本稿の構成は、まず実時間用並列計算機の満たすべき要件について述べ、次に CODA-r での優先度処理について述べる。そして CODA-r のハードウェア構成についての提案を行なう。

2 センサフュージョンシステム用並列計算機の要件

センサフュージョンシステム用並列計算機の満たすべき条件は、(1) 実時間処理に対するハードウェアによる支援、(2) 膨大な数のセンサ / アクチュエータの制御と高速な計算処理を両立するスケーラビリティ、の 2 点であると考えられる。

2.1 実時間処理へのハードウェアによる支援

実行時間の予測可能性は実時間処理においてもっとも重要な問題である [3]。逐次処理でのプロセス (或はタスク) スケジューリングにおいては、プロセス毎に優先度を与え優先度にしたがってプロセスを実行することにより優先度の高いプロセスの実行時間の上限を規程できると考えられる。一方、並列処理においては一つの命令の実行時間でさえも、プロセッサ間の同期や通信によって影響されるため、予測可能でないと考えられる。そのため、実時間処理を並列に行なうためには、(1) 命令実行、(2) プロセッサ間通信、(3) プロセッサ間の同期、等の基本的操作の段階から実行時間の予測可能性を考慮しなければならない。そこで、一命令毎に優先度の比較を行ない、優先度の高い命令の実行時間の上限を規定する。

プロセッサ間通信ネットワークもパケット毎に優先度制御を行ない、通信に必要な時間を予測可能にする。また、ネットワーク上での優先度逆転 (Priority Inversion) 解消のための回路を実装する。

2.2 スケーラビリティ

一般の並列処理システムでは、スケーラビリティはプロセッサ台数に比例して実行速度が向上できることと考えることができる。しかし、実時間処理においては単に実行速度の向上だけでなく、実行時間の予測可能性も保持されなくてはならない。すなわち、最小デッドライン間隔とプロセッサに許される最大ロードファクターの改善が必須である。我々は、実時間用並列計算機におけるスケーラビリティを次のように考える。

拡張性: 同型のプロセッサを接続し台数を増やすことによって、システム規模を容易に拡張できる。

応答性: プロセッサ台数を増やすことにより、最小デッドライン間隔を短縮することができる。

ロードファクター: プロセッサ台数を増やすことにより、処理可能なプロセス (タスク) 数を増やすことができる。

CODA-r は上記 3 点の要求を満たすように設計され、センサフュージョンシステムの中核となる計算機システムを目指す。

3 CODA-rにおける優先度処理

多くの実時間システムでは、処理の実時間性を保証する手段として優先度が導入されている。このようなシステムでは、優先度はそれぞれのプロセスの持つデッドラインに対応した緊急性を表していなくてはならない。しかし、優先度はデッドラインを直接表現できるほどの広さを通常は有していないため、優先度とデッドラインを静的に対応させただけでは実時間性を保証することは困難である。そこで、多くのシステムでは、優先度を動的に管理し各プロセスのデッドラインに応じて適宜変化させることにより、実時間性を保証しようとしている。

優先度の動的な管理では優先度の変更はシステム全体で一貫性を保ったまま行なわれなくてはならない。即ち、並列プロセッサ上で優先度を変化させる際には、各プロセッサ上の優先度キューを並べ換えるだけでなく、全システムで同期をとり同時に優先度を変化させなければならない。全システムでの同期はプロセッサ台数が増加した場合には大きなオーバヘッドとなる。したがって、並列プロセッサを実時間処理に適用する際に優先度の動的管理を導入することは、前節で述べたスケーラビリティを損なうものであると考えることができる。

CODA-rでは、優先度フィールドを大きくとることでデッドラインを直接優先度として取り扱えるようにする。プロセスのデッドラインは生成時に動的に割り当てられるが、一旦割り当てられた後はプロセスの実行が終了するまで変化しない。また、プロセス間のデッドラインの順序も変化することはない。したがって、CODA-rではプロセス生成時に優先度キューの適当な位置に挿入するようにすれば、その後の優先度キューの並べ換えは必要ない。また、優先度管理のための全プロセッサの同期も必要ないため、プロセッサ台数が増加しても優先度管理のオーバヘッドは増加しない。そのため、CODA-rは実時間処理に対して高いスケーラビリティを有していると考えられる。

4 CODA-rのハードウェア構成

CODA-rは、128台のプロセッシングエレメント(PE)とセンサがパケット通信ネットワークを介して結合された並列処理計算機である(図1)。各PEはローカルメモリとセンサ/アクチュエータ用のデータ入出力ポートを有する。CODA-rは物理的な共有メモリは持たないが、論理的な共有変数は各PEのローカルメモリ上に保持される。本節ではCODA-rのPEのハードウェア構成について述べる。

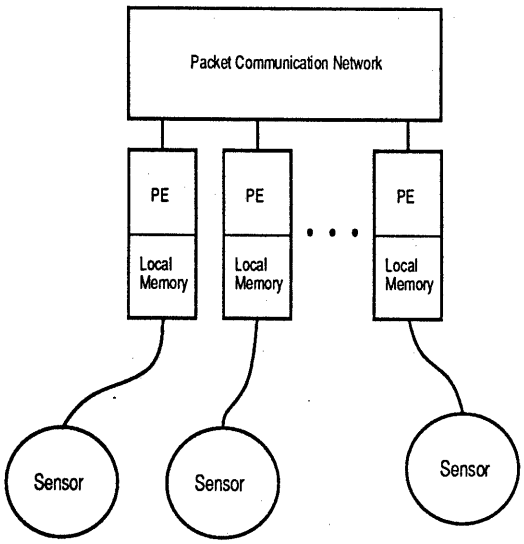

図 1: CODA-rの全体構成

4.1 CODA-rプロセッシングエレメントの概要

並列プロセッサにおいて、実行時間の予測可能性を損なうもっとも大きな要因は、通信と同期である。したがって、パケット処理と同期処理が命令実行を妨げないような設計を行なわなくてはならない。CODA-rは、命令実行パイプライン(IP)とパケット処理パイプライン(PP)の2本のパイプラインを有する(図2)。PPは命令実行に影響することなくパケットの入出力を行ない、入力パケットのデータは5ポートレジスタ(2書き込みポート、3読みだしポート)ないしメモリに直接書き込まれる。レジスタファイルは2セット用意されており、IPが「表」のレジスタセットを使用してプログラムを実行している間に、「裏」のレジスタセットに次に実行される(プロセス優先度により推定される)プロセスをあらかじめロードしておくことによりコンテクスト切替えの高速化を図っている。

同期操作のオーバヘッドは、同期の伴うコンテクスト切替えの他に同期操作そのものの効率も考慮されなくてはならない。CODA-rではレジスタ読み出しの際に同期判定を行なうことにより、余分なメモリアクセスを行なうことなく、また、パイプライ

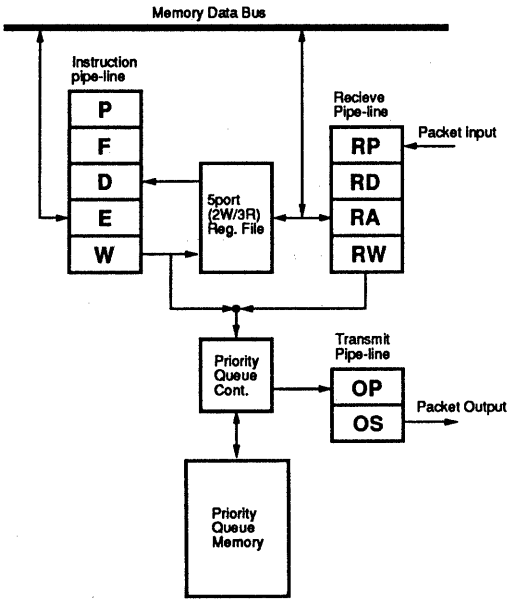

図 2: CODA-r PE の構成

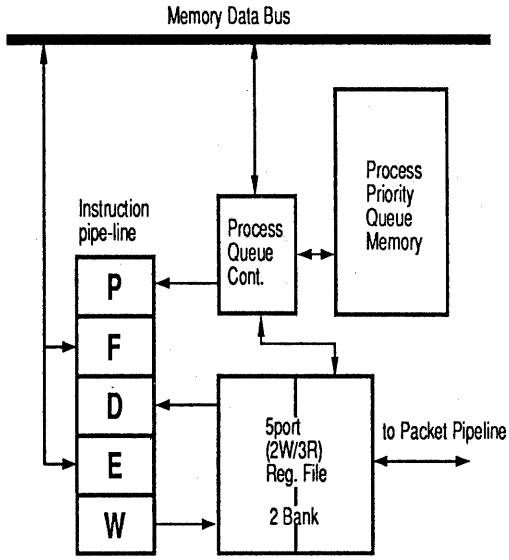

図 3: 命令パイプラインの構成

ンブレークを発生することなく同期操作を行なうことが可能である [5]。

4.2 パイプライン構成

IP(図 3) は、優先度判定段 (P)、命令フェッチ段 (F)、命令デコード段 (D)、実行段 (E)、レジスタ書き込み段 (W) の 5 段のパイプライン構成をとる。P 段では実行中のプロセスの優先度とプロセス優先度キュー中の最高優先度を比較し、プロセス優先度キューの優先度の方が実行中のものより高い場合にはプロセスを切り換える (プリエンプション)。

レジスタは一語毎に presence bit を持ち、レジスタ上のデータが存在する (定義されている) かどうかを示す。presence bit は D 段でのレジスタ読み出しの際に同時に検査され、presence bit が *off* の場合にはトラップが発生し続く 2 サイクルはコンテクスト切替えに使われる [5]。コンテクスト切替えに使用される 2 サイクルは、プロセス優先度キューと PC 群の間で実行アドレスの転送を行なうために最低限必要な時間である、同期操作そのものはパイプラインにバブルを発生することなく実行される。W 段は E 段の演算結果をレジスタかパケット出力キューに書き込む。パケット出力キューは IP からはレジスタとして扱われ、書き込みによって PP を起動しパケット出力を行なう。

PP(図 4) は 6 段のパイプライン構成をとる。パケット受信段 (RP)、パケットデコード段 (RD)、パケット処理段 (RA)、パケット書き込み段 (RW) の 4 段でパケットの受信を処理し、出力優先度判定段 (OP) と出力段 (OS) の 2 段でパケット出力を処理する。入力パケットは RP 段を起動し、RD 段でのデコード結果にしたがって RA 段でレジスタないしメモリにアクセスする。RW 段は遠隔アクセス要求パケットに対して返答を返す場合のみ用いられ、パケット出力キューに RA 段の結果を書き込み OP 段を起動する。

4.3 優先度キュー

CODA-r は、プロセスキューとパケット出力キューの二つの優先度キューを持つ。CODA-r の優先度キューは、上位優先度のエンキューと最上位優先度のデキューを一定時間で実行できるように設計されている。優先度キュー (図 5) は、N 語 (図 5では 8 語) の双方向シフトレジスタ、N 語の比較器、N 語 2 バンクのレジスタファイル、オーバーフローメモリからなる。本節では優先度キューの動作を、パケット出力キューを例にとって詳述する。

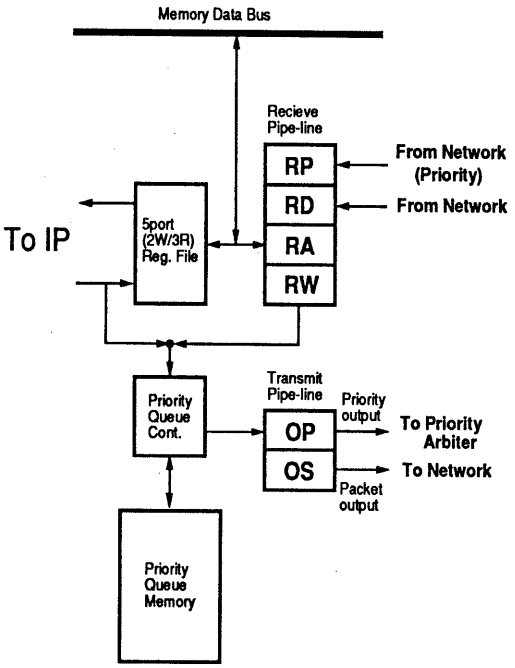

図 4: パケット処理パイプラインの構成

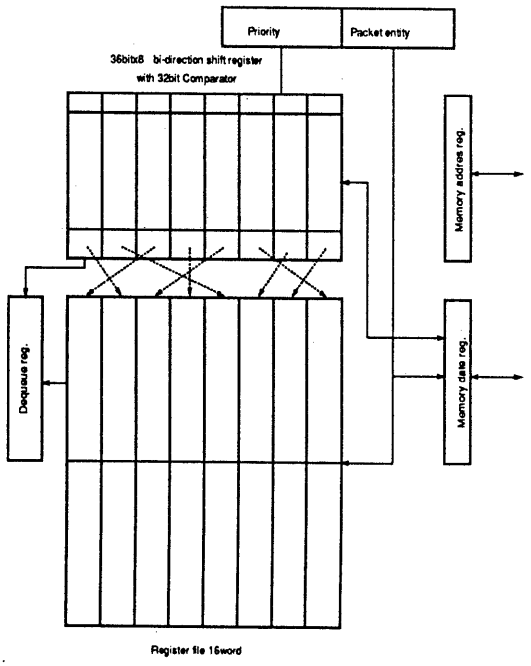

図 5: 優先度キューの構成

パケット本体はレジスタファイルに保持される。優先度はパケット本体の保持されているレジスタアドレスと共に双方向シフトレジスタ上に保持される。双方向レジスタ上の優先度は、その値にしたがって並べられ、図5中の左端には最上位優先度、右端には最低位優先度が保持される。

エンキュー時の動作を以下に述べる。

1. 入力パケットの優先度はシフトレジスタ上の8つの優先度と比較される。

2. 入力パケットの優先度がシフトレジスタ上の最低位優先度よりも高い場合には、入力パケットの優先度よりも低いものが右へシフトされシフトレジスタ上に空きを作る。

3. 入力パケットの優先度が2で作られた空きに書き込まれる。

4. 3と同時に最低位優先度はシフトレジスタから追い出され、レジスタファイルからパケット本体が読み出される。

5. 3,4と同時にパケット本体をレジスタファイルに書き込む。

6. 4で読み出したパケット本体をオーバフローメモリに書き込む。

シフトレジスタ上の最低位優先度よりも低い優先度のパケットが到着した場合は、入力パケットは直接オーバフローメモリに書き込まれる。この場合は、$O(\log N)$(N はオーバーフローメモリ上に保持されたパケットの数) の時間が必要となる。

パケット送出時の動作を以下に述べる。

1. 双方向シフトレジスタ上の最上位優先度にしたがって、パケット本体をレジスタから読み出す。

2. 双方向レジスタを左(上位側)へシフトし、最低位部にオーバーフローメモリからパケットを読み出す。

3. オーバフローメモリから読み出したパケットをレジスタに書き込む。

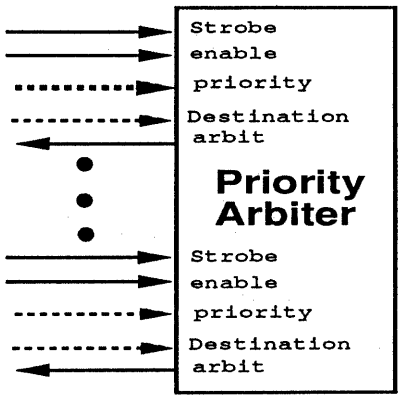

図 6: ビットスライス化優先度判定回路

オーバーフローメモリからの読み出しは、(エンキュー操作に伴う) 書き込み時に優先度順に並べてあれば、メモリ中の最上位を読み出すだけであるため、一定時間で可能である。

4.4 ネットワーク

CODA-r は、優先度制御を行なうと同時に優先度逆転を解消するための機構を備えた多段ルータネットワークを使用する。優先度先送り方式 (PFS)[4] による優先度逆転の解消方法は、もともとは 2×2 ルータのために考案されたものであるが、ネットワーク遅延を改善するため 16×16 ルータに PFS を実装しネットワークを構成する。

16×16 ルータで 32bit の優先度を直接調停するためには、優先度入力だけで 512 本の信号ピンが必要となり、現在の実装技術では実装が困難である。そこで、優先度判定回路をビットスライス化してルータに組み込む (図 6)。strobe はパケット入力要求があることを示し、複数の入力が同時に同一 destination アドレスを要求した場合に優先度を比較する。そして、そのうちの最上位優先度を持つ入力に対して arbit 出力を返すことにより入力を許可する。最上位優先度の入力が複数あった場合には、その全てに arbit 出力を返し他のチップでの判定結果、或は、ラウンドロビンによって最終的な判定を行なう。

8bit の優先度判定回路チップを 4 個使用し arbit 出力を下位のチップの enable 入力に接続することにより 32bit の優先度判定回路を構成することができる (図 7)。

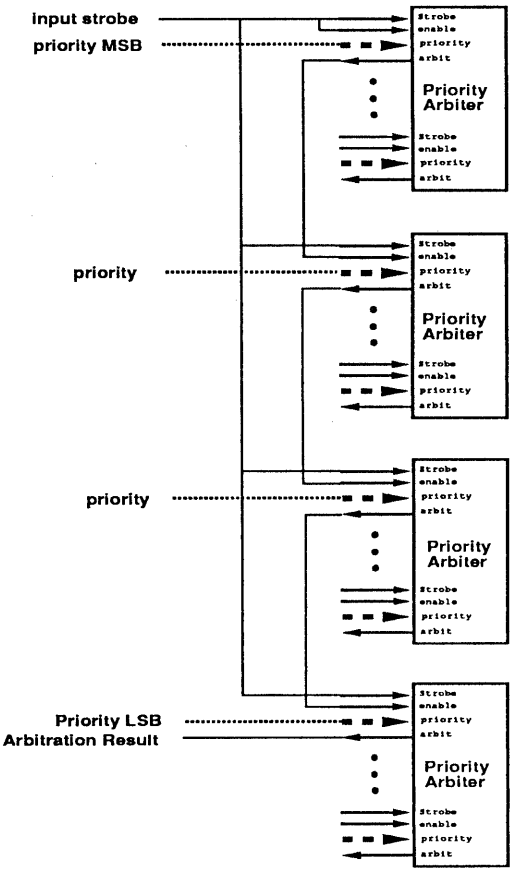

図 7: 優先度判定回路の結線 (1 ポート入力分)

5 今後の検討課題

CODA-r は 32bit の優先度フィールドを持つことで優先度の動的な管理を不要とし、また、優先度制御されたネットワークを持つことにより通信時間の予測可能性を向上させた。しかし、実時間性を保ちながらロードファクターを向上させるためには、負荷バランスをとることの重要性には変わりがない。プロセスマイグレーションは、不均衡化した負荷を再び均衡化するために有効な手段であるが、マイグレーションのオーバヘッドにより実行時間の予測可能性が損なわれることも考えられる。従って、最適な負荷分散をあらかじめ得ることが実時間処理には重要であると考えられる。そこで、ルータチップに自動負荷分散機構 [1] を実装し、低オーバヘッドで負荷の均一化を図ることを検討している。

6 現状

CODA-rのプロセッシングエレメントは10万ゲートのゲートアレイ上に実装するための設計が開始されている。また、ネットワークについては、ルータチップの設計と共に結合トポロジーなどのシミュレーション評価も行なわれている。

謝辞

本研究を遂行するにあたり御指導、御討論いただいた弓場敏嗣情報アーキテクチャ部長ならびに計算機方式研究室の同僚諸氏に感謝致します。

参考文献

[1] K. Hiraki, S. Sekiguchi, and T. Shimada. Load Scheduling Schemes Using Inter-PE Network. *Systems and Computers in Japan*, 18(1):55–65, 1987.

[2] M. Ishikawa. Sensor Fusion System -Mechanism for Integration of Sensory Information-. *Journal of the Robotics Society of Japan*, 6(3):46–53, June 1989. (in Japanese).

[3] J.A. Stankovic and K. Ramamritham, editors. *Hard Real-Time Sytems.* IEEE Computer Society Press, 1988. IEEE Catalog Number EH0276-6, ISBN 0-8186-0819-6.

[4] K. Toda, K. Nishida, Y. Uchibori, S. Sakai, and T. Shimada. CODA: A Multiprocessor Architecture for Sensor Fusion. In *Proceedings of The Fifth International Symposium on Intelligent Control*, pages 261–266, September 1990.

[5] K. Toda, K. Nishida, Y. Uchibori, and T. Sakai, S. Shimada. Parallel Multi-Context Architecture with High-Speed Synchronization Mechanism. In *Proceedings of the Fifth International Parallel Processing Symposium*, pages 336–343, April 1991.