計算機アーキテクチャ　63－21
(1986・11・21)

データ駆動形プロセッサQ-pの実行制御方式

An Execution Control Scheme of Data-Driven Processor Q-p

嶋　憲司† 明智光夫† 小守伸史† 岡本俊弥‡
Kenji SHIMA Mitsuo MEICHI Shinji KOMORI Toshiya OKAMOTO
宮田宗一‡ 戸倉　毅‖ 清水雅久‖ 原　秀次‖
Souichi MIYATA Tsuyoshi TOKURA Masahisa SHIMIZU Shuji HARA
西川博昭‖‖ 浅田勝彦‖‖ 寺田浩詔‖‖
Hiroaki NISHIKAWA Katsuhiko ASADA Hiroaki TERADA

† 三菱電機㈱応用機器研究所 / LSI研究所
Mitsubishi Electric Corporation
‡ シャープ㈱超ＬＳＩ開発研究所
Sharp Corporation
‖ 松下電器産業㈱システム研究所
Matsushita Electric Industrial Co.,Ltd.
‖‖ 三洋電機㈱中央研究所
Sanyo Electric Co.,Ltd.
‖‖大阪大学　工学部
Faculty of Engineering, Osaka University

あらまし　流れ形処理方式と呼ぶ自己タイミング形制御処理機構を特徴としたデータ駆動形プロセッサQ-pの実行制御方式ついて述べる。まず、データ駆動形プロセッサQ-pにおいて実行される言語とその体系について述べた後、履歴依存処理、およびカラー処理を実現する実行制御方式について詳述し、更に、履歴依存処理を実行する記憶機構、および、マルチプロセッサ構成について述べる。

１．　まえがき

この研究の目的とするデータ駆動形プロセッサ（以下Q-x[1]）は、プログラム作成が極めて容易な、高機能素子の実現手段として計画された。研究では、まず、言語の体系を導入し、この体系の最下位の部分をＶＬＳＩによるハードウェアとして実現する言語主導の立場を採ってきた。また、ハードウェア実現に際しては、単一のチップでも独立に機能することをも考慮し、同時に、必ずしも補助的な外付けの接続機構を用いることなく、多数のチップを相互に接続して、極めて高能力のプログラム処理システムをも実現できるよう充分配慮してきた。

ここでは、Q-xシステムの基本的な構想に基づきQ-xシステムの可能性を検証するためのプロトタイプとして試作された、Q-p[2]の実現例において、まず、この研究を進めるにあたって主導的役割を担った言語とその体系について述べ、続いて、履歴依存処理を可能とし、ストリーム形データ処理と再帰処理を実現するために採用されたQ-pの動的データ駆動形実行制御方式について詳述し、併せて履歴依存処理を実行する記憶機構の機能、更にマルチプロセッサ構成についても述べる。

２．言語

２．１　言語とその体系

Q-xの言語体系としては、３つの階層、即ち、最上位にある超高位言語、ハードウェアから独立な中間言語、およびハードウェアに依存する実行形式を想定し、最上位の超高位言語は信号流れ図あるいは状態遷移図などのように、視覚的にも了解性を重視した階層的記述形式の導入が、数値、非数値処理を問わず、必須であると判断し、利用者にはプログラム知識を殆ど要しないレベルの記述手段を提供することを目標としてきた。

そのため、高位の記述形式が一般に有する、並列処理性および高位の仕様記述により示される構造を保存することを考慮して、中間言語にも超高位言語同様、図的な記述形式を採用している。同時に、中間言語の実行方式としてデータ駆動原理を導入し、ハードウェアレベルで実行可能なプログラムの検証性を保証している。現在、中間言語の基礎とした図的データ駆動言語Ｄ^{3}Ｌ[3]の記述能力を拡張したＵＬ１[4],[5]とその改良版であるＵＬ２が完成しており、これらの言語を用いて具体的なソフトウェア開発を進めてきた。

Q-xの言語体系の最上位水準のものとして、差し当り、図的な記述形式が定着している応用分野を対象として超高位言語の検討が進められており、階層的図式言語記述およびデータ駆動原理の採用によって、数値処理以外の記述の可能性や、更には、信号流れ図表現を直接に実行可能なプログラムに変換する手法によって、プログラム知識を持たずともディジタル信号処理プログラムを生成可能であることが明らかになっている。[6]

2．2　図的データ駆動形言語ＵＬ２の図的表記法

データ駆動形プロセッサQ-p[7]とそのエミュレーションシステム上でのソフトウェア開発支援のために、図的データ駆動形言語ＵＬ２を開発した。すでに開発しているＵＬ１同様、ＵＬ２はソフトウェア開発者が用いる言語として、具体的な応用に対するプログラミングを容易に行なえることを目標として設計したが、ＵＬ１の評価結果に基づき仕様拡充を行った。また、ＵＬ２の処理系は、グラフィック・エディタ、コンパイラ、マッパーにより構成され、ＵＬ２をQ-p上でプロクラムの並列性を最大限に活用して効率良く実行させるために処理系の機能拡充も併せて進めた。

ＵＬ２は、ノードとアークによるフローグラフの形でプログラムを記述する図的言語であって、プログラムは図１のように関数ブロックごとにひとつのフローグラフで記述され、全体として関数ブロックの階層的あるいは再帰的な図的表現により記述される。

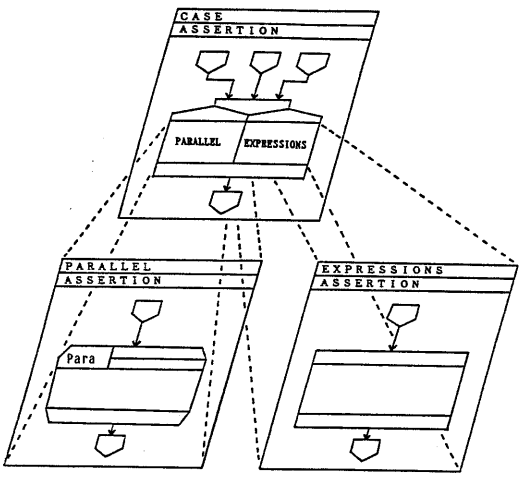


図１．ＵＬ２プログラムの階層的構造

ＵＬ１を用いたプログラミングにおける評価結果は主に以下の通りであった。

１）図的であり、了解性に優れる。
２）数式記述の採用により処理内容を簡明に記述可能である。
３）アークとアーク上を流れるデータの対応が不明確である。
４）コントロールアークの意味づけが不明確である。
５）命令レベルの図的記述が必要である。
６）履歴処理に対応する機能は構成上の制限から最適化を要する。

そこで、ＵＬ２はＵＬ１に、以下の仕様拡充を行った。

１）アークの意味の明確化。
　アークはデータと一対一に対応させる。数値、配列のポインタを問わず全データはアークにより従属関係を明示する。
２）同期の明示。
　コントロールアークをなくし、処理間の順序関係は利用者がアークを用いて明記する。また、同期のためのデータの待合わせを明示する表記を導入する。
３）命令レベルの図的記述の導入。
　フローグラフで命令レベルの表記を可能とする。
４）単一割当規則の局所化。
　プログラム記述の容易さ、および了解性を改善するため、数式記述における変数名の有効範囲を一つのブロックとし、単一割当規則を局所化する。

このような仕様拡充を図ったもので、記述性、操作性に大きい改善があるのが特徴である。

ＵＬ２の記述要素を図２に示す。ノードはデータに対する処理を表し、以下のものがある。

１）データの入力端を表すノード
　ソース（リンクソース、ポインタソース、トリガーソース）
２）データの出力端を表すノード
　シンク（リンクシンク、ポインタシンク、トリガーシンク）
３）データに対する処理を表すノード

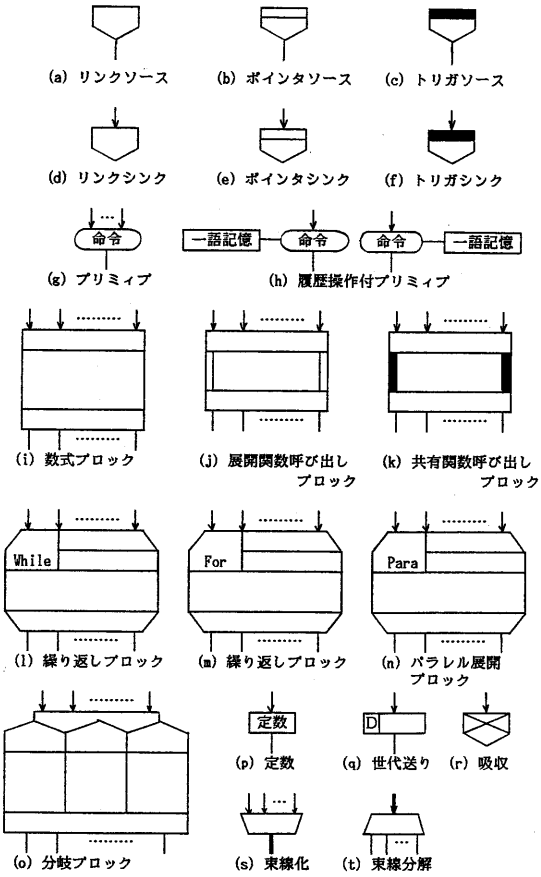
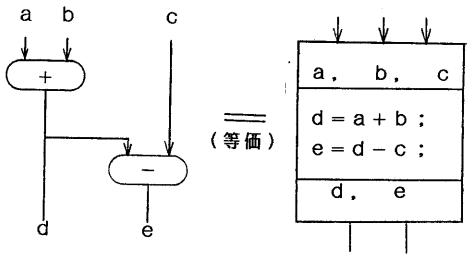


図２．ＵＬ２の記述要素

プリミティブ、数式ブロック、関数呼出しブロック（展開関数呼出し・共有関数呼出し）、　　構造ブロック（繰返し構造（while,for）・パラレル展開構造・分岐構造）、定数、世代送り、吸収、束線化、束線分解

アークは処理（ノード）間のデータの依存関係を表す。アークとデータ（値）は一対一に対応する。全てのデータはそれぞれ異なるアークにより、その依存関係を明示して記述する。略記法として複数アークを１本のアークにまとめて束アークとして表記できるが束アークは一般のアークとは区別する。束アークを用いるときには、束線化ノードあるいは数式インタフェースによって対応するアークあるいはデータ群を明示せねばならない。

処理の記述形式には以下の２通りがあり、同一の処理記述例を図３に示す。



（ａ）フローグラフ記述　　（ｂ）数式記述

図３．処理記述の例

１）フローグラフ記述

プリミティブノードを用いてデータに対する演算を表し、データフローグラフの形で処理を記述する。

２）数式記述

数式インタフェースによりアーク（データ）に名前（データ名）を付け、その名前を用いて数式により処理を記述する。

数式で記述する場合には、単一割当規則に従って行い、データ名とデータ（値）との一対一対応により、副作用のない数式記述が保証される。単一割当規則を局所化し、データに付せられた名前は、数式記述されたそのブロック内でのみ有効とされる。

このように、フローグラフ記述では処理の構造、特に並列性が明快に表現され、また、数式記述により処理内容は簡明に記述できる。それぞれの特質を使い分けることにより、了解性の高いプログラム記述が可能となる。

ＵＬ２では履歴依存処理を扱うために以下の記憶が用いられる。

・一語記憶：１回書き込み１回読み出しの記憶
・定数記憶：読み出し（参照）専用一語記憶
・アレイ　：世代管理された配列構造記憶
・メモリ　：世代の制限のない配列構造記憶

一語記憶と定数記憶は名前によってアクセスされる記憶であり、関数内また別関数からでもアクセスできる。アレイ、メモリの配列構造記憶はポインタに対して参照、更新の操作を行う。ポインタがデータとして扱われアークを用いて陽に記述される。これらの記憶を用いてストリーム形処理環境（多世代実行環境）下での副作用のない履歴依存処理を実現している。

３．　実行制御方式

３．１　動的データ駆動形実行制御方式

Q-pの動的データ駆動形実行制御方式は、以下のように特徴づけられる。[8]

１）　世代番号により、データ入力順序の識別を行って、画像データなどのストリーム形データのパイプライン処理が可能である。

２）　カラー番号により、関数呼出し環境の識別をおこなって、再帰・共有関数の実行が可能である。

３）　世代番号により参照・更新の制御を行う一語記憶機能を導入したことにより、副作用のない履歴依存処理が可能である。

４）　入力制御部を導入したことにより、システム内に滞留するデータの量を自律的に制御することが可能である。

これらの特徴を有する、動的データ駆動形実行制御方式について詳述する。

Q-pに採用された動的データ駆動方式を図４に示す。この基本部は５つの階層から成り、まず、階層０は入力処理部であり、入力制御部（ＩＮ）によって、許可された世代の入力を行うなど必要な場合には有限資源下の処理パケットの溢れを制御することが行われている。

階層１は履歴更新部（Ｓｐ）であり、履歴依存処理における記憶の更新が行われる。

階層２はプログラム読みだし部であり、プログラム記憶部（Ｓｄ）から命令の取出しが行われる。

階層３は発火処理・履歴処理部であり、実行可能な命令を検知しデータ対を生成する発火制御部（ＦＣ）および履歴依存処理の履歴参照部（Ｓｇ）と出力部（ＯＵＴ）から成る。

階層４はデータ処理部であり、演算処理部（ＦＰ）、制御命令処理部（ＣＩＰ）、カラー処理部（ＣＰ）および配列データ記憶部（Ｓａ）から成る。

入力処理部で、入力制御（ＩＮ）によって、初期設定あるいは先行の世代によって入力の許可されている世代の入力処理が行われる。先行

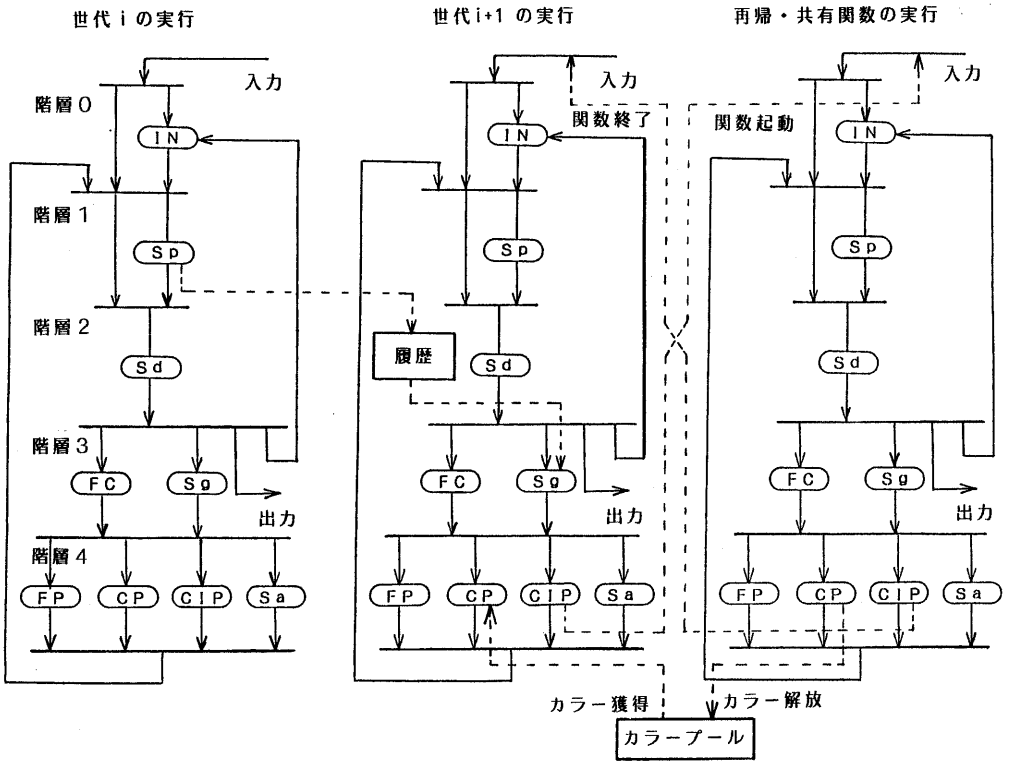


図４．Q-pの動的データ駆動形実行制御方式

する世代の処理が所定の段階まで完了した時点で、次世代に対する入力許可パケットが入力制御（ＩＮ）に送られて入力処理が行われるもので、このような入力制御の不必要な処理の場合には、この階層を経由しないで入力することも可能である。

基本的な命令実行はトークンがＳｄ→ＦＣ→ＦＰ→（Ｓｄ）のループを周回することによって行われる。プログラム記憶部（Ｓｄ）で命令の取出しが行われて、パケットの物理的行先を表すセレクションコード、処理内容を表すオペレーションコード、次の命令の格納場所を表す次行先ノード番号などの情報が読出されてパケットに付加される。発火処理（ＦＣ）では二つの被演算データが揃って実行可能となった命令を検出してデータ対を生成する。このデータ対が演算処理部（ＦＰ）において処理実行される。

このような循環式パイプラインによって処理実行が行われることと同様、ハードウェア実現にも統一的に「流れ形処理」という自己タイミング形非同期パイプライン方式を導入している。

３．２　履歴依存処理

履歴依存処理の実行は１語記憶機能（Ｓｐ，Ｓｇ）および配列構造記憶部（Ｓａ）において行われ、演算処理と履歴依存処理とが一回の周回で終了するようＳｐ、Ｓｇが配置されている。

Ｑ-ｐでは、参照数・更新数による世代番号操作機能を持つ１語記憶機能の概念を導入している。例えば世代ｉの実行によって更新された履歴を世代i+1 の実行中に参照する履歴依存処理を、Ｓｐ、Ｓｇにおいてこの履歴に対するget/put 操作によって実現する。

配列に対する履歴依存処理は、個々の配列に対して世代管理することはせず、配列構造記憶部（Ｓａ）では配列構造を配列への識別子で管理し、配列への識別子は１語記憶機能で世代管理している。このような世代管理の手法を用いることにより履歴依存処理により生ずる副作用を配列構造に関しても回避することができる。

時系列データ処理において用いられる次世代へのデータの引渡しは、世代番号を１だけ増加させることにより行うことも可能で、そのための命令が導入されている。

３．３　再帰・共有関数処理実行方式

再帰・共有関数呼出しは、カラーの獲得・解放の命令と世代／カラーと処理終了後の行先情報を退避・復帰させる命令を導入し、トークン流の中にＬＩＦＯ(Last In-Fast Out)機能を実現している。

カラーを用いた再帰・共有関数処理は、まず世代ｇのトークンがカラーｒを獲得すると同時に、世代ｇをカラーｒと対にして退避したのち関数を起動する。関数処理終了後、結果トークンはカラーを解放するとともにカラーｒによって世代ｇのトークンとして復帰する。関数実行は図４に示すように、カラーによって呼出し環境が識別され、多重化されている。

カラーの獲得と解放はカラー処理部ＣＰにおいて、実効的なスタックを生成する処理は制御命令処理部ＣＩＰにおいて行われる。

このデータ駆動形実行制御方式はデータ駆動形プロセッサQ-p上において実現され、その妥当性について確認されている。[9]

４．　流れ形処理機構と記憶機能

４．１　流れ形処理機構

Q-pのハードウェア実現[10]の基本方針は以下に示す通りである。

１）ＶＬＳＩ機能素子として動作、即ち、１チップでも独立に動作し、高処理、高機能を示す。

２）マルチプロセッサ構成要素として動作する。

３）ハードウェアの動作にも関数処理形概念を適用し、パイプライン形処理を採用する。

Q-pは履歴依存処理を可能としており、通常のアドレス指定形メモリにパイプライン処理を用いて履歴処理を行なっている。また、Q-pの機能要素間の転送にもパイプライン転送概念を導入[11]したのをはじめ各処理機構[12]におけるハードウェア設計[13]の基本概念として非同期パイプライン処理である「流れ形処理」を統一的に採用している。

流れ形処理の特徴は以下の通りである。

１）処理機構とデータ転送機構のハードウェア実現手法上の統一性がある（流れ形処理概念の一貫性）。

２）処理機構とデータ転送機構にエラスティック性を付与している（全データ処理・転送機構が伝送機能とバッファ機能を併せ持つこと）。

これらの特徴によってQ-pでは、集中システム制御機構の排除、タイミングクロックの除去、

バス構造の除去が可能となっている。

4．2　記憶機構の機能

Q-pにおける記憶機能には、3種類の機能メモリ、即ち、

1）プログラムを蓄積するプログラム記憶Ｓｄ

2）配列データを処理する配列構造記憶　Ｓａ

3）定数記憶と履歴依存処理に係る記憶　Ｓｃ

があり、方式上のＳｐ、ＳｇはＳｃで実現している。これらの機能メモリは、演算処理機構や転送機構の能力と完全に均衡する能力を持ち、システム全体としてメモリアクセスによるボトルネックを完全に回避する流れ形処理機構により実現している。流れ形処理の条件を満たし、どの記憶機構にも柔軟に適用できる記憶制御方式はアドレス指定形のメモリをＣ素子[11]によってアクセス制御するとともに、流れ形処理によりパケットデータを処理する方式を導入している。[14]　機能メモリ制御機能の動作は次のようにまとめられる。

1）Ｓｄ：入力によって参照・更新が決められ、実行状態で次位命令参照の際に、結果データを複製して使用することが判明すると参照データに従ってアドレスを自動的に更新し、再度参照を行う。

2）Ｓａ：入力によって参照・更新が決められ、ポインタアドレスおよび参照・更新データのいずれもが、操作可能である。

3）Ｓｃ：カラー情報に従って、履歴の参照・更新状態を読みだし、一語記憶へのアクセスが可能なことが判明すると、命令情報に従って一語記憶を操作する。また、異なる世代間の管理が必要な場合は、同時にメモリ内の対応する世代番号を更新する。

これらいずれの機能メモリも、ロードデータのダウンロードとメモリダンプデータのアップロードは任意の状態で可能である。

4．3　プログラム記憶部および配列構造記憶

転送機構のパケットスルーレートが設計値1に対してアクセス時間0．5以下のＳＲＡＭを用いる場合には多重バンクへのメモリ分割が不要であり、転送系と同等のスルーレートを得ることができる。Ｓａにおけるリニアセルアレイへのアクセスは、世代管理を行う場合、履歴依存処理部ポインタアドレスを得たのち、配列への参照・更新を行う。世代管理を行わない場合には、履歴依存処理部の定数記憶からポインタアドレスを得ることができる。パケットデータの流れに沿ったＳｄおよびＳａの構成をそれぞれ図5、図6に示す。図中、ADL BUFFERは非同期遅延線路[11]を表している。

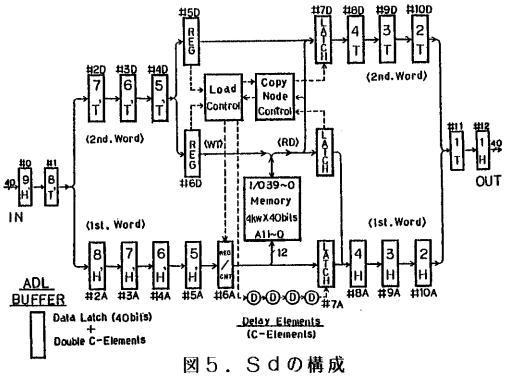


図5．Ｓｄの構成

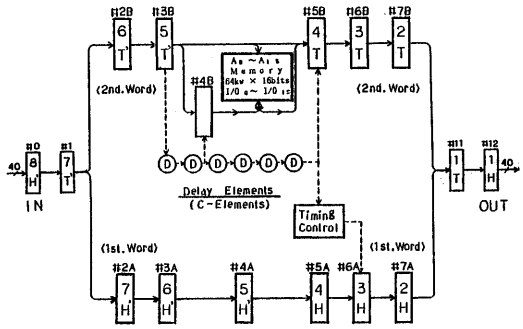


図6．Ｓａの構成

4．4　定数記憶および履歴依存処理部

Ｓｃでは、読みだし専用の定数記憶と、参照・更新回数を各々1回に限定した一語記憶（セルメモリ）が、異なるアドレス空間に割りつけられるが、同一方式でアクセス制御される。履歴依存処理では、同一世代および連続する二世代にまたがるデータの更新・参照を、ハードウェアで直接実行し、ＵＬ2プログラムにおいて記述される参照・更新の全順序性を保証している。即ち、指定された一語記憶に、参照したい世代のデータが存在しなければ、着目する世代のデータが更新されるまで、また、着目する世代に続く次世代のデータとして一語記憶を更新したいとき、着目する世代のデータが更新されるまで、いずれも内蔵のループバックバッファを巡回する。これらは、履歴依存処理の一語記憶に関する命令の実行規則に基づく全順序性を、ハードウェア水準で保証する制御方式となって

いる。さらに、定数および一語記憶の参照データと処理データの左右オペランド位置の入替えも、Ｓｃにおいて実行できる。

パケットデータの流れに沿ったＳｃの構成を図７に示す。Ｓｃのハードウェアの動作は以下のようである。まず、

１）パケット第１語によって、命令デコードと世代番号ラッチを行う。次に、

２）パケット第２語の一語記憶のＩＤに従って、世代の一致判定を行う。単一ＩＤ当り８世代の参照・更新データを処理可能で、８世代同時に比較判定を行なえる。最後に、

３）判定結果および命令に従って、データの参照・更新（および、世代間にまたがる命令については、世代の更新）を行うか、又は参照・更新の全順序性を保証するため入力パケットを保存して、ループバックバッファに戻す。

Q-pでは、このような記憶機能により、副作用のない履歴依存処理を実現している。

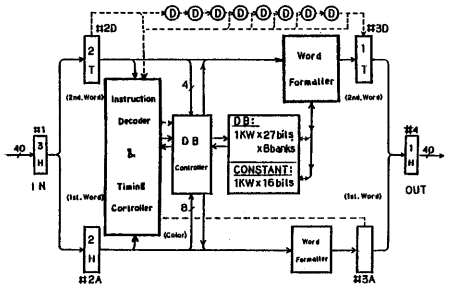


図７．Ｓｃの構成

５．　Q-pのマルチプロセッサ構成

データ駆動方式の大きな特徴の一つは、並列実行可能な処理を複数のプロセッサに分散配置することにより、問題に内在する並列性を生かした処理の効率化が図れることである。マルチプロセッサのハードウェアを構成する上でプロセッサ間のデータ転送効率の低下を抑制することが極めて重要である。

Q-pでは選択コードを用いてプロセッサ内通信と同様に、プロセッサ間のパケット通信が行なえ、マルチプロセッサ構成への拡張が容易となっている。プロセッサ間通信の手段としては、共通バスによらず、図８のようにシャッフルネットワーク構成によった。これは、Q-pを４台用いたマルチプロセッサ構成（２入力－２出力）であり、入力制御部（ＩＮ）をも含めて示している。各ＰＥ(Processing Element)は図９のように分岐部と合流部を用いて２入力－２出力として構成した。

図において、各Q-pは、入力については２入力の合流の後、データパケットの行先ＰＥ番号により、取入れるかバイパスさせるかが決められる。出力については、Q-p本体出力とバイパスパケットとの合流後、これら行先ＰＥ番号の偶奇によって出力線路を選択する。

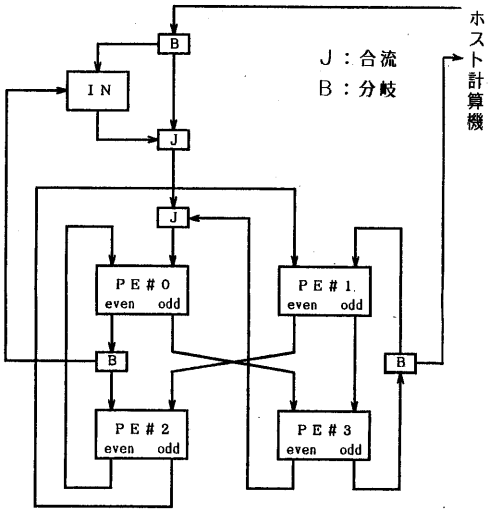
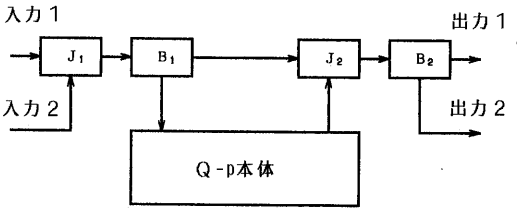


図８．４台のQ-pによるマルチプロセッサ構成



J_1，J_2：対等合流
B_1：入力パケットの行先ＰＥ＃が一致のとき分岐
B_2：出力パケットの行先＃が奇数のとき分岐

図９．ＰＥの内部構成

この構成をとれることにより、システムのどの断面で切っても充分な転送能力が確保され、特定の転送路に負荷が集中することがないのが特徴である。更に、多数のQ-pを結合してマルチプロセッサ構成を採ることが可能で、そのときQ-pのＡＬＵ稼働率は理論予測値の 80-90% を示すことが確認されている。[7),(9)]

6. むすび

この報告では、Q-xの言語の体系、およびその試作機Q-p上のソフトウェア開発のための図的データ駆動形言語ＵＬ２の表記法について述べ、続いて、履歴依存処理と再帰・共有関数処理を可能とする動的データ駆動形実行制御方式について詳述し、更に、履歴依存処理を実行する記憶機構の機能およびマルチプロセッサ構成について述べてきた。

ＵＬ２はD^3Ｌの図的な高度並列処理構造表現に加えて、従来形の数式表現形式および配列構造記憶処理を補完し、さらにそれによる副作用を回避したプログラミング言語で、その処理系を用いて具体的な問題記述を行ってきた。また、ここで述べた実行制御方式の妥当性は実現した試作機Q-pにおいて検証し、所期の機能が得られることを確認した。さらに、記憶機構の機能について、履歴依存処理を実現するための具体的手法を明示し、併せてマルチプロセッサ構成についても実際に採られた方式を提示し、いずれもQ-p上でその機能を確認した。

ここで述べた並列処理を可能とするデータ駆動形プロセッサの実行制御方式を既存の半導体技術によりマイクロプロセッサとして実現することが可能となっており、今後はチップ化に向けての検討をも併せて進めて行く。

最後に、この研究をご指導ご支援いただいた関係各位に感謝する。なお、この研究は通商産業省重要技術研究開発費補助金（60工技総第1537号）の援助を受けて行った研究の一部である。

［参考文献］

(1) 寺田他：“データ駆動形プロセッサQ-xの設計思想”，情報処理学会第32回全国大会講演論文集pp.207-208(1986-3).
(2) 寺田他：“データ駆動形プロセッサQ-p”，信学会データフローワークショップ予稿集pp.75-78(1986-5).
(3) 西川他：“履歴依存性を許すデータ駆動図式”，信学論 D,J66-D,10,pp.1169-1176(昭58-10).
(4) 明智他：“図的データ駆動形言語ＵＬ１の仕様”，情報処理学会第32回全国大会講演論文集pp.209-210(1986-3).
(5) 西川他：“図的データ駆動言語ＵＬ１の言語処理系”，情報処理学会第32回全国大会講演論文集pp.211-212(1986-3).
(6) 寺田他：“D^3Ｌによる図的言語処理体系とその一実現法について”，信学会データフローワークショップ予稿集pp.119-126(1986-5).
(7) 岡本他：“データ駆動形プロセッサQ-pのエミュレーションシステムと性能評価”，情報処理学会第32回全国大会講演論文集pp.223-224(1986-3).
(8) 戸倉他：“データ駆動形プロセッサQ-pのアーキテクチャ”，情報処理学会第32回全国大会講演論文集pp.213-214(1986-3).
(9) 戸倉他：“データ駆動形プロセッサQ-pの性能評価”，信学会コンピュータシステム研資(1986-11).
(10)浅野他：“データ駆動形プロセッサQ-pの構成”，情報処理学会第32回全国大会講演論文集pp.215-216(1986-3).
(11)小守他：“データ駆動形プロセッサQ-pの転送系”，情報処理学会第32回全国大会講演論文集pp.217-218(1986-3).
(12)三浦他：“データ駆動形プロセッサQ-pにおける処理機構”，情報処理学会第32回全国大会講演論文集pp.219-220(1986-3).
(13)三浦他：“データ駆動形プロセッサQ-pのハードウェア”，信学会コンピュータシステム研資(1986-11).
(14)宮田他：“データ駆動形プロセッサQ-pにおける記憶機能”，情報処理学会第32回全国大会講演論文集pp.221-222(1986-3).