計算機アーキテクチャ 73－11
（1988. 10. 31, 11. 1）

SIMP（単一命令流／多重命令パイプライン）方式に基づく『新風』プロセッサの高速化技法および性能予測

入江直彦　久我守弘　村上和彰　富田眞治
（九州大学）

SIMP（Single Instruction stream／Multiple instruction Pipelining：単一命令流／多重命令パイプライン）方式に基づくプロセッサ『新風』／【ʃimpu:】を開発中である．『新風』は4本の5ステージ命令パイプラインを備え，単一命令流中の4個の命令を同時に処理していく．命令パイプライン単体の最大性能は8MIPSであり，4本の命令パイプラインにより理論的には最大32MIPSが可能である．システム最終形態としては，シングル・ユーザに対し対話型スーパーコンピューティング環境を提供する『新風』／【ʃimpu:】DTS-edition（DTS：DeskTop Supercomputer）を計画している．

『新風』プロセッサは高速化のため種々の高速化メカニズムを採用している．特に，局所データフローに基づくout-of-order実行制御，ならびに，命令間の並列実行を妨げる要因となるフロー依存および制御依存への対処の仕方に特長がある．

ソフトウェア・シミュレーションによる性能予測を行なった結果，これらの高速化メカニズムによって，約90%の性能向上が得られた．またプロセッサ単体の性能としては最適化されていないオブジェクトコードに対しても実効的に15MIPSの性能が得られ，シングル・ユーザ向けのスーパーコンピュティング環境を十分提供できるものと予測している．

Speedup Mechanisms and Performance Estimate for SIMP Processor Prototype：【ʃimpu:】

(in Japanese)

Naohiko IRIE, Morihiro KUGA, Kazuaki MURAKAMI, and Shinji TOMITA
Kyushu University
6－1, Kasuga-koen, kasuga-shi, Fukuoka, 816, Japan

SIMP (Single Instruction stream／Multiple instruction Pipelining) is a multiple instruction-pipeline parallel architecture targeted for enhancing the performance of SISD processors. SIMP architecture combines instruction pipelining used by conventional SISD processors, and low-level parallelism exploited by VLIW processors and multiple functional-unit (MFU) processors.

【ʃimpu:】is a prototype processor based on SIMP architecture. It provides 4 instruction pipelines of 8MIPS each.【ʃimpu:】processes a single instruction stream based on out-of-order execution model in local dataflow fashion. It provides several speedup mechanisms to detect and handle flow dependent and control dependent hazards, which prevent out-of-order execution.

The result of software simulation for【ʃimpu:】shows that 90% of the speedup is achieved by these speedup mechanisms. At least 15MIPS of scalar performance can be obtained for non-optimized object code.

1. はじめに

我々は，高速汎用プロセッサ・アーキテクチャとして，SIMP（Single Instruction stream／Multiple instruction Pipelining：単一命令流／多重命令パイプライン）方式を提案している[1)2)]．SIMP方式とは，従来の時間並列処理（命令パイプライン方式）に空間並列処理（低レベル並列処理）を加えることで，命令バンダリな応用への応答時間（responce time）向上を目指したものである．現在，この方式に基づく試作プロセッサ『新風』／【∫impu：】を開発している[3)]．『新風』のシステム最終形態としては，『新風』プロセッサと高速グラフィックス・プロセッサとを組み合わせたデスクトップ・スーパーコンピュータ『新風』／【∫impu：】DTS-editionを計画している[4)]（DTS：DeskTop Supercomputer）．『新風』プロセッサは，SIMP方式に基づく試作第1号機であり，高速化のために数々のハードウェア機構を投入している．

本稿では，まず『新風』DTS-editionの概要を述べたあと，『新風』プロセッサの構成および特長，および『新風』に採用されている高速化メカニズムについて述べる．最後に，高速化メカニズムによる性能向上の予測を行なう．

2. 『新風』DTS-editionの概要

『新風』DTS-editionのハードウェア構成を図1に示す．

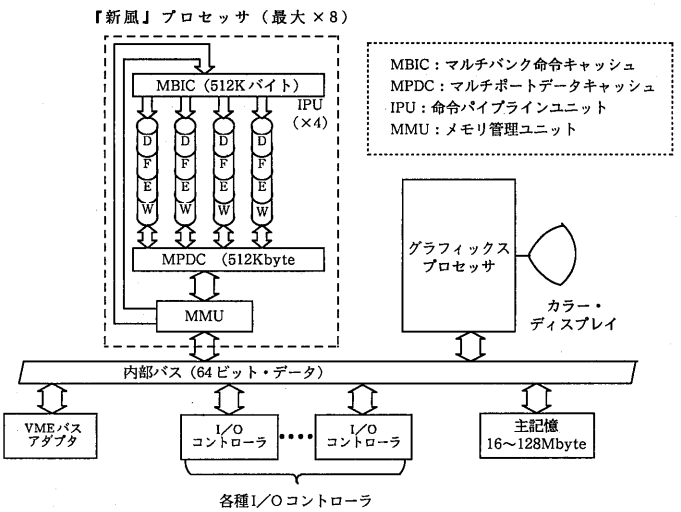


図1. 『新風』DTS-editionの構成

『新風』DTS-editionは，シングル・ユーザ向けのデスクトップ・スーパーコンピュータであり，対話型スーパーコンピューティング環境を提供することを目標にしている．その実現には，以下の性能を満たす必要があると一般に言われている．

① 整数演算：10MIPS以上（ピーク性能）

② 浮動小数点演算：5MFLOPS以上
（倍精度のLinpack）

③ 3次元描画：5万ポリゴン／秒以上
（100画素のポリゴン）

『新風』DTS-editionにおいて数値計算処理を担当するのが，メイン・プロセッサの『新風』プロセッサである．これはその諸元が示すように（表1参照），上記の整数および浮動小数点演算に対する性能要求①②に充分応えている．また，3次元ポリゴン描画は，将来開発予定のグラフィックス・プロセッサが担当し，上記の性能要求③を満足するようにする．

3. 『新風』プロセッサの構成

『新風』プロセッサは図2に示すように，以下の6種類の主要ユニットから成る．

（1） 命令ブロック供給ユニット

IBSU（Instruction Block Supplying Unit）．マルチバンク命令キャッシュ（MBIC）から連続する4個の命令をプリフェッチして，4本の命令パイプライン・ユニット（IPUs）にそれぞれ投入する．これら4命令を命令ブロック（instruction block）と呼び，以後命令ブロック単位で管理する．5ステージの命令パイプライン処理過程における，最初の命令フェッチ（I）ステージに相当する．分岐予測機構により，プリフェッチすべき命令ブロックの先頭アドレスを決定する．

（2） 命令パイプライン・ユニット

IPUs（Instruction Pipeline Units）．『新風』プロセッサでは，同一構成のIPUを4本備える．5ステージの命令パイプライン処理過程のうち，以下の残り4ステージを担当する；

①命令解読（D）ステージ：命令ブロック中の個々の命令のop-codeフィールドを解読する．

②依存解析（F）ステージ：解読結果に従って，依存解析機能付きレジスタファイル（DHRF）に対してレジスタアクセス要求を出す．DHRFか

表1.『新風』プロセッサ諸元

プロセッサ・アーキテクチャ	SIMP（単一命令流／多重命令パイプライン方式）
命令セット・アーキテクチャ	BISC（均衡命令セット・コンピュータ）
命令長	32bit固定長
レジスタ数	汎用レジスタ　：64本（32bit）
	浮動小数点レジスタ：64本（32bit）
プロセッサ構成法	32bitビルディング・ブロックLSIを中心に高速TTL−IC（SSI／MSI）を使用（将来的には，パイプライン・スライスLSI化を計画）
マシン・サイクル	60nsec（16.7MHz）
命令パイプライン本数	4本
命令パイプライン構成	5ステージ（2サイクル・パイプライン）
演算器構成	整数演算：AMD　Am29332（加減除算） Am29C323（乗算）
	浮動小数点演算：Weitek　WTL2265（加減算） WTL2264（乗除算）
演算性能（ピーク性能）	整数：　8MIPS（パイプライン単体） 32MIPS（プロセッサ全体）
	単精度浮動小数点（ベクトル加減乗算）： 16MFLOPS（パイプライン単体） 64MFLOPS（プロセッサ全体）
	倍精度浮動小数点（ベクトル加減乗算）： 8MFLOPS（パイプライン単体） 32MFLOPS（プロセッサ全体）
キャッシュ容量	命令キャッシュ　：512Kバイト データ・キャッシュ：512Kバイト
キャッシュ方式	・ダイレクト・マッピング方式 ・仮想アドレス・キャッシュ （命令およびデータ・キャッシュとも）

らは，全IPUs中の先行命令に対するデータフロー・グラフおよびレジスタデータが供給される．　データフロー・グラフはSSL（ソース供給リスト）の形で表現される．SSLには，ソースオペランドに書き込む可能性のある（制御依存により未確定）先行命令のIDが登録されている．

③実行（E）ステージ：整数ALU，整数乗算器，浮動小数点ALU，浮動小数点乗除算器，データキャッシュアクセス要求ユニット（DCAR）の5つの機能ユニットから構成される．各々の機能ユニットはさらに演算パイプライン化されている．演算命令の場合はFステージで生成されたデータフロー・グラフに従い，発火可能条件が整った時点で演算の実行を遂行する（局所データフロー実行制御）．LOAD／STORE命令の場合は，アドレス計算を行ってマルチポート・データキャッシュ（MPDC）に対してメモリアクセス要求を出す．分岐命令の場合，分岐先アドレス計算を行うと同時に，分岐のtaken／untakenを決定する．

④格納（W）ステージ：命令ブロックの全命令の実行が終了したら，DHRFに対し結果の格納を要求する．

(3)　依存解析機能付きレジスタファイル

DHRF（Dependency Handling Register File）．汎用レジスタ64本および浮動小数点レジスタ64本を備えた，12ポート（{左ソース＋右ソース＋シンク}×4）のレジスタファイル，レジスタ予約テーブルおよび制御依存テーブルから構成される．全IPUに共有される．命令パイプライン処理過程のうち，FおよびWステージに関与する．Fステージに関しては，レジスタ予約テーブルおよび制御依存テーブルを参照することで命令間の制御依存関係およびデータ依存関係を解析し，データフロー・グラフを作成する．Wステージに関しては，結果の格納と同時に，これをソースオペランドとして要求しているFステージの命令が存在すればそれに渡す（オペランド・フォワーディング）．

(4)　マルチバンク命令キャッシュ

MBIC（Multiple Banked Instruction Cache）．4バンク（4バイト／バンク）構成で，64バイト（＝16命令）／ラインである．ライン当り，4個の分岐予測アドレスを記録する．

(5)　マルチポート・データキャッシュ

MPDC（Multiple Ported Data Cache）．4ポート構成かつ16バンク（1バイト／バンク）構成で，64バイト／ラインである．ストア・イン方式を採用する．内部にはLOAD／STOREバッファを設け，out-of-orderなアクセス要求に対応可能である．

(6)　メモリ管理ユニット

MMU（Memory Management Unit）．仮想アドレスから物理アドレスへのアドレス変換を行う．マルチ『新風』プロセッサ構成に備えてスヌーピング機構を有する．

4. 高速化メカニズム

『新風』プロセッサの処理において，パイプラインの入口と出口に関してはin-order実行となる．パイプラインの出口つまりWステージについてのin-order実行は，データの出力依存への対処するため，およびpreciseな割込みを保証するためである．しかしながら『新風』プロセッサ内部の実行モデルは，Eステージにおける局所データフロー（local／restricted dataflow）実行制御に立脚しており，オブジェクト・プログラム上の命令出現順序とは異なるout-of-order実行となる．したがって，処理の高速化は効率の良い局所データフロー実行に係ってくることになるが，これを

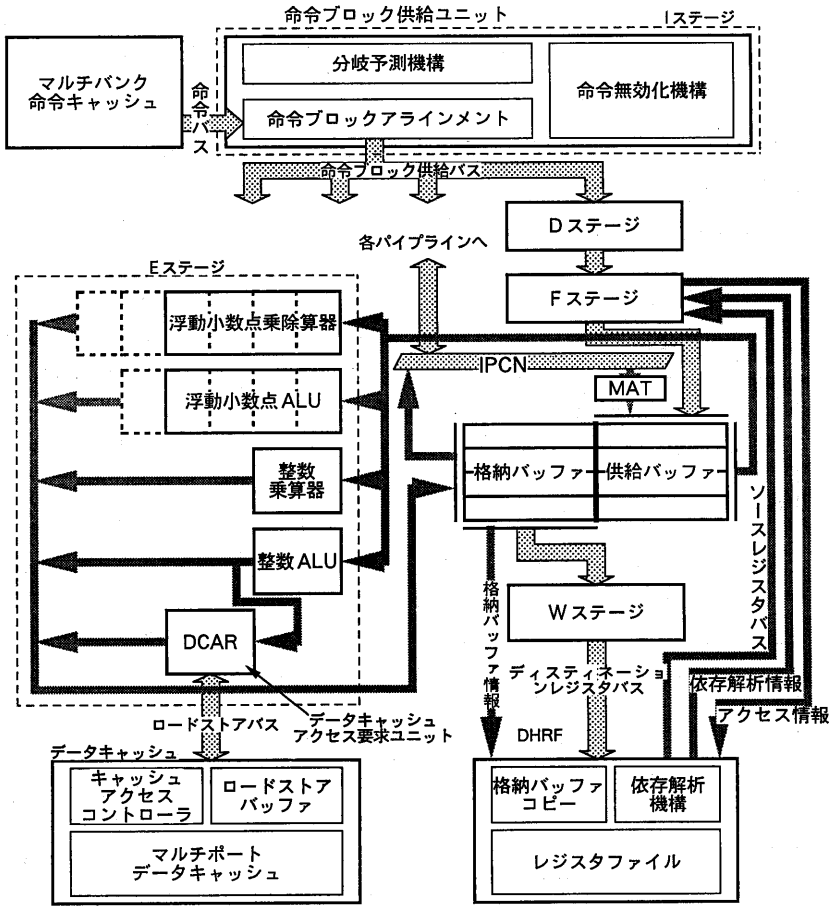


図2. 『新風』パイプライン構成

妨げる大きな障害として,

① フロー依存関係 (flow dependence)

② 条件分岐命令に起因する制御依存関係

などがある.『新風』プロセッサでは,これらに以下のように対処している[5].

4. 1 フロー依存関係

これには,以下の2機構が協調して対処する.

(1) 供給バッファおよび格納バッファ

供給バッファおよび格納バッファはFステージおよびWステージから見ると,連動して動作するFIFOキューとしての役割をもつ.しかしEステージからみた場合,オペランドの待ち合わせ場所 (reservation station) および実行結果の格納場所 (reorder buffer) という役割を担う.(図3参照)

Fステージから供給バッファへの投入は,命令ブロック単位で行ない,投入アルゴリズムはFIFOに従う.Eステージは供給バッファ中の命令の中で,発火条件の揃ったものを取り出して命令の実行を行なう.ここでの発火条件とは,ソースオペランドが揃っていて,かつ使用したい演算器が空いていることをいう.このときの取り出しはFIFOに従わない.これによりパイプライン中でのout-of-order実行を実現している.実行した結果は,その命令の入っていた供給バッファに対応する格納バッファへ一時保存する.Wステージによる格納バッファからの取り出しは命令ブロック単位で行い,これはFIFOに従う.またこのとき格納バッファから取り出される命令ブロック中の命令は全て実行が終了しており,しかも,その結果はsafe (後述) であることが保証されていなければならない.このようにして結果の格納についてin-orderを保証している.

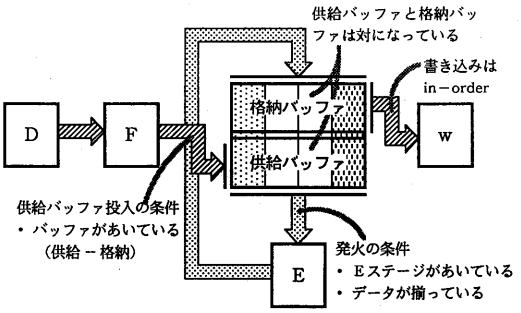


図3. 供給－格納バッファの概念図

(2) パイプライン・チェイニング機構

局所データフロー実行を円滑に進めるには，Eステージでの演算結果を，それを待っている命令に速やかに分配する必要がある．このために，単一命令パイプラインにおけるデータ・バイパス機構を多重命令パイプライン用に拡張した．すなわち，各々のパイプラインの格納バッファから全てのパイプラインの供給バッファへ直接演算結果を送出できるように，これらを命令パイプライン・チェイニング網（IPCN）で相互結合している．

Eステージから格納バッファに送られてきた実行結果がsafeならば，この結果は供給元命令識別子と共にIPCNを経由して全パイプラインへ放送される．ここで実行結果がsafeというのは先行命令による制御依存が全て解決されている状態をいう．各パイプラインでは，供給バッファ中のオペランド待ち状態にある命令が放送されてきた供給元命令識別子と自命令のSSL（ソース供給リスト）を比較し，マッチングがとれればそのデータをオペランドとして受け取る．

4. 2 条件分岐命令に起因する制御依存関係

『新風』プロセッサでは，

① 動的命令ブロック流（dynamic instruction-block stream）

② 命令ブロック内の静的命令流（static instruction stream within an instruction block）

といった2種類の異なる流れ（stream）を処理対象とする．これらはそれぞれ，条件分岐命令の出現により流れが乱される可能性がある．加えて，条件分岐命令の存在は，先のout-of-order実行の効果の度合を低下させてしまう．これに対しては，以下の対策を施している．

(1) 命令ブロック単位の分岐予測

『新風』では，実行時に分岐予測テーブルの作成／更新を行なう動的分岐予測方式を採用する．ただし予測は命令ブロック（＝4命令）単位に行なう．分岐予測テーブルは，命令キャッシュ・ライン（＝16命令）対応に，4個の分岐予測アドレスを保持する．分岐予測テーブルの登録／更新は，過去1～2回の履歴をとることにより行なわれる．現在効率の良いアルゴリズムを検討中である．

(2) 条件付き先行実行（conditional advanced execution）

一般にデータフロー実行を行なう場合，制御依存関係にある命令は制御依存関係が解決した後でないと発火し得ない．しかし『新風』では，制御依存まで考慮したデータフローグラフの作成および実行のバックトラックのサポートを行なうことにより，制御依存関係にある命令のout-of-orderな発火を可能にしている．

つまり，制御依存関係にある命令でも，ソース・オペランドが揃って発火可能になれば，とりあえず実行を行なうようにする．しかし，その演算結果は制御依存が解決されるまでunsafeの状態である．unsafeな演算結果は後続命令のソースオペランドとはなり得ない．したがって先行する条件分岐命令の実行が終了し，結果がsafeになってからIPCN上に出力される．これにより，先行実行の度合は1レベルに制限されるが，複雑なバックトラックを避けることができる．

(3) 先行条件決定（advanced conditioning）

通常の機械命令セットでは，条件分岐命令においてコンディション・コードと分岐条件との間のマスク演算を行ってtaken／untakenを決定し，takenの場合分岐するのが一般的である．『新風』プロセッサでは，このtaken／untaken決定操作（conditioning）とtakenの際の分岐操作とを分離させる手法を採る[6]．これにより，条件分岐命令に先行して，taken／untakenが判明可能となるので，前述の条件付き先行実行の効果がより高まる．

5. 疑似『新風』プロセッサの性能予測

『新風』プロセッサの開発と並行して，ソフトウェア・シミュレータによりSIMP方式の性能予測を行なっている[7]．シミュレータでは任意の命令パイプライン段数／本数のマシン・モデルにおいて種々の高速化メカニズムを取り入れた上での性能の測定が可能である．本章では，このシミュレータを使用して行なった疑似『新風』の性能予測について述べる．

5. 1 マシンモデル

今回行なったシミュレーションにおけるマシンモデルは以下の通りである.

(1) 命令パイプライン構成：5段パイプライン

- Iステージ：命令フェッチ
- Dステージ：デコード
- Fステージ：フロー解析, レジスタアクセス
- Eステージ：演算実行, メモリアクセス
- Wステージ：結果格納（レジスタ）

各ステージは2サイクルで動作する2サイクルパイプラインとなっている. また上記の構成は, Eステージの細部を除いて『新風』と同じである.

(2) 命令パイプライン本数

1〜6本

(3) バッファリング

『新風』における供給バッファおよび格納バッファと同一のものを, Fステージ－Eステージ間とEステージ－Wステージ間にそれぞれ設けている（エントリ数：4).

(4) 命令セット

MC68020の命令セットをLOAD／STOREに変換したもの.

(5) 高速化メカニズム

① データフロー実行の高速化（図4参照）

（命令実行順序制御アルゴリズム）

・命令ブロック間in-order実行…命令ブロック間のオーバーラップ実行を許さない. したがって概念的には命令ブロック単位で処理を行なう「太い」パイプラインとして振舞う.

・命令ブロック間オーバーラップ実行…命令ブロック間でのオーバーラップ実行を行なう. ただしパイプライン中では一般のパイプラインと同様に命令間の追越しはない.

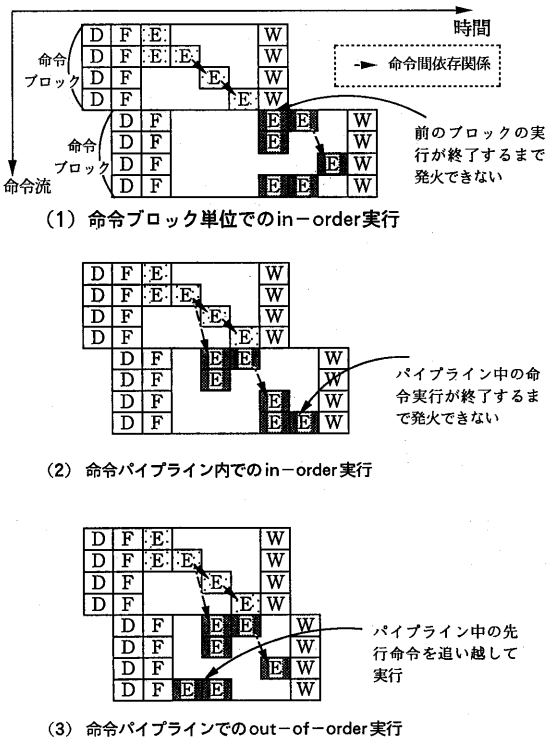
・命令パイプライン内out-of-order実行…『新風』の採る方式である. 実行ステージ前段のバッファがオペランド待ち合わせ機構として働き, パイプライン中でのout-of-order実行を行なう.

② フロー依存への対処

・チェイニング機構…データフローグラフに従い, オペランド待ち合わせ場所において高速に先行命令の実行結果を得るための機構

③ 制御依存への対処

・命令ブロック単位での分岐予測…過去の履歴に従ってヒストリバッファを設けその情報により次の命令アドレスを決定する. 今回のシミュレーションでは過去1回の履歴を保存している. またヒストリバッファのオーバーフローは考慮していない.



(1) 命令ブロック単位でのin－order実行

(2) 命令パイプライン内でのin－order実行

(3) 命令パイプラインでのout－of－order実行

図4. 命令実行順序制御アルゴリズム

(6) テストプログラム

テストプログラムは, クイックソートを行なうソートプログラムを使用した.

5. 2 評価結果

図5〜9に評価結果を示す. 評価の基準としてissue rate（ステージ当りの命令実行数）を用いた. すなわちグラフの縦軸がissue rateに当る. またグラフの横軸はパイプラインの本数を表す.

図5はチェイニングによる性能向上を示している. これから分かるようにチェイニング自体では数％しか性能向上に寄与しないが, 命令実行順序制御アルゴリズムと組み合わせることによって（命令ブロック間オーバーラップ実行, 命令パイプライン内out-of-order実行）最大30％程度の性能向上がみられる.

分岐予測による性能向上は図6に示すとおりである. 分岐予測での性能向上率はパイプラインの本数や命令実行順序制御アルゴリズムとほとんど関係なく, 20％前後の性能

向上が見られる．今回のシミュレーションでは過去1回の履歴しか採っておらず，予測アルゴリズムも非常に単純なものとなっている．『新風』で採用される分岐予測については別途検討中であり，更に性能向上が期待できる．

命令実行順序制御アルゴリズムについては，先にも述べたようにチェイニングとの組合せにより命令ブロック間オーバーラップ実行を行なうことにより性能が約15%～25%向上する（パイプライン数が3～6本の場合）．しかし分岐予測がないと，out-of-order実行できる命令数が少ないため，命令パイプライン内out-of-order実行を行なってもさほど効果が上がらない（図7参照）．しかしこれに分岐予測を付加すると，命令ブロック間オーバーラップ実行することで約20%～40%（パイプライン数：3～6），命令パイプライン内out-of-order実行することで約40%～55%性能が向上する（図8参照）．

したがって，全ての高速化メカニズムを取入れた場合，90%程度の性能向上が期待できる（図9参照）．

本シミュレーションでは命令セットが『新風』のものと異なるため，正確な性能予測とは言い難いが，上記に示したような高速化メカニズムを『新風』が採った場合，issue rateが約1.8（パイプライン本数：4本）であるので，15MIPS程度の性能は少なくとも達成し得ると予想している．

ただし，今回のシミュレーションでは『新風』が採ってい

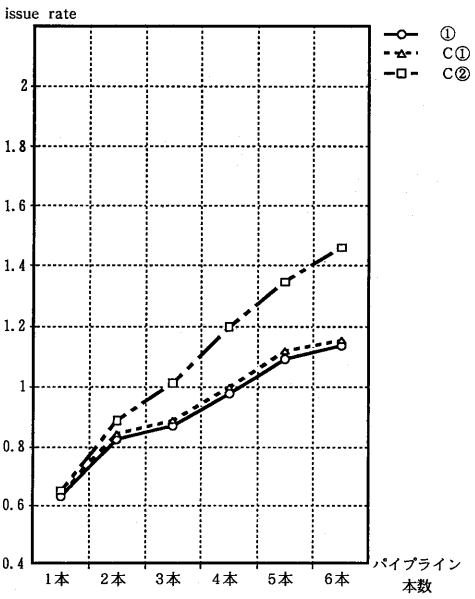


図5. チェイニングによる効果（C）

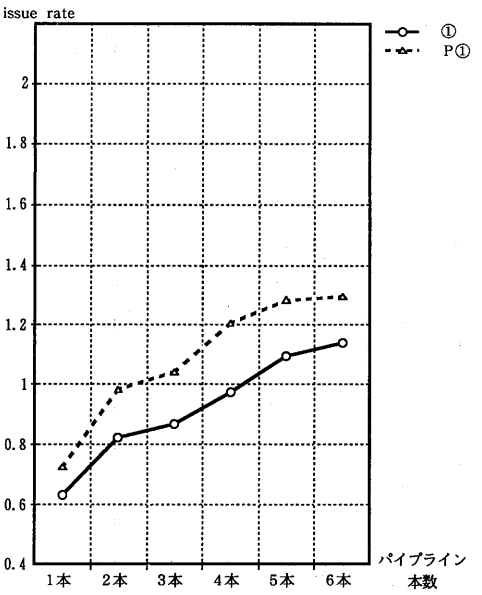


図6. 分岐予測による効果（P）

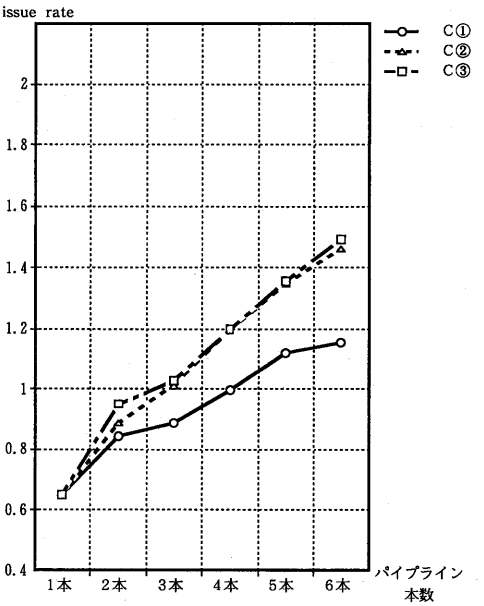


図7. 命令制御アルゴリズムによる効果（with C）

凡例：
①：命令ブロック間in-order実行
②：命令ブロック間out-of-order実行
③：命令パイプライン内out-of-order実行
C：チェイニング機構あり

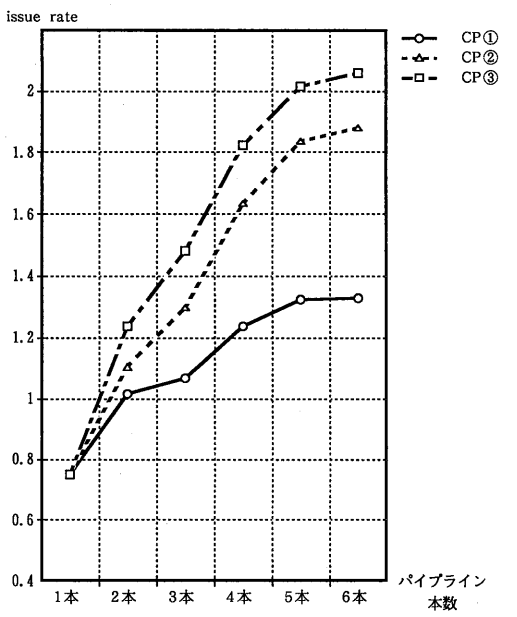


図8. 命令制御アルゴリズムによる効果（with P+C）

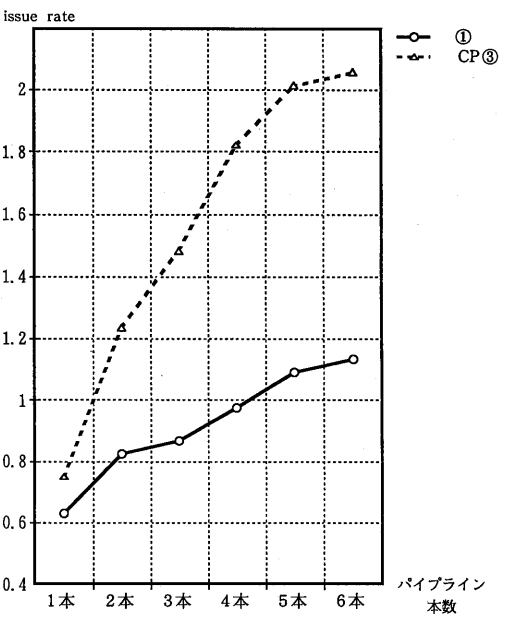


図9. 高速化メカニズム全てによる効果

る並列演算パイプライン処理については考慮していない．これによるout-of-order実行への効果はかなりあると予想される．また，ソフトウェア（コンパイラ）による最適化についても考慮していないため，これらを取り入れることで更に性能向上が期待できる．

6. おわりに

以上，『新風』DTS-editionの概要，そのメイン・プロセッサである『新風』プロセッサの構成および，その性能予測について述べた．『新風』プロセッサ第1号機は，昭和64年3月の完成を目標に現在開発中である．

『新風』でのout-of-order実行に伴って生じる不正確な割込み（imprecise interrupt）問題への対策は，アーキテクチャ・レベルで講じている．また，『新風』プロセッサでは，演算時間のバラツキを小さく抑えるBISC（均衡命令セット・コンピュータ）を採用し，加えてベクトル処理機構を装備している．これらの詳細については，別の機会に譲りたい．

参考文献

1）村上ほか：SIMP（単一命令流／多重命令パイプライン）方式の構想，情処研報，88－CA－69－4（1988年1月）

2）村上ほか：SIMP（単一命令流／多重命令パイプライン）アーキテクチャについて，情処36全大論文集，3C－1（1988年3月）

3）五島ほか：SIMPアーキテクチャに基づくハードウェア・システム構成，情処36全大論文集，3C－2（1988年3月）

4）村上ほか：『新風』DTS-edition：SIMP（単一命令流／多重命令パイプライン）方式に基づくデスクトップ・スーパーコンピュータ，情処37全大論文集，4N－1（1988年9月）

5）久我ほか：『新風』／【ʃimpu：】プロセッサの高速化メカニズム，情処37全大論文集，4N－2（1988年9月）

6）W.G.Rosocha and E.S.Lee：Performance Enhancement of SISD Processors, Proc.6th ASCA（Apr.1979）

7）入江ほか：SIMP（単一命令流／多重命令パイプライン）方式のシミュレーションによる評価，情処37全大論文集，4N－3（1988年9月）