社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－ARC－170 (10)
2006／11／30

異種命令セット同時実行プロセッサOROCHIの構成

島田貴史† 田端猛一† 北村俊明††

近年，携帯機器などの機能向上が激しく，これに用いられるプロセッサは，高性能化と低消費エネルギーの両立を要求されている．我々は，これに応える1方式としてVLIW方式プロセッサを取り上げ，画像処理などのワークロードを用いて評価を行ってきた．しかし，システム全体を考えると，陽に並列性を含むアプリケーションだけでなく，システム制御など，並列処理ハードウェアを効果的に利用できない処理も存在し，そのために別途汎用プロセッサを搭載するというシステム構成を取ることが多かった．これを改善するため，VLIWプロセッサで，SMT(Simultaneous Multi-Threading)を構成し，しかもそのスレッドの1つは，異なる命令セットであるようなプロセッサについて検討を始めた．本稿では，この背景と，概要について述べる．

Development of A Simultaneous Multiple Instruction-sets Execution Processor - OROCHI -

TAKASHI SHIMADA,† TAKEKAZU TABATA† and TOSHIAKI KITAMURA††

Recently, mobile devices have many functionalities and these applications require high performance and low power consumption to processors used in them. To study the processor architecture for this requirement, we have evaluated VLIW processor with various video processing workloads and get good results. However, when we design the whole system, there are not only parallel executable applications, that can be executed by VLIW processor, but also serial executable processes such as system control. To execute these non-parallel programs efficiently, we need another general purpose processor. So, to overcome this situation, we start studying the new SMT (Simultaneous Multi-Threading) processor which execute different instruction sets, general purpose instruction set for system control and VLIW instruction set for multimedia processing. In this paper, we describe the background and overview of this study.

1. はじめに

近年，携帯機器やゲーム機器などの機能向上は著しいものがあり，家庭用ゲーム機器のプロセッサは，少し前のスーパーコンピュータに匹敵するような性能が達成されている．しかも，安価な家庭用機器であるので，特別な冷却方式など望むべくも無く，使用環境から静寂性を配慮すると自然空冷で対応できる程度の発熱量で押さえることが必要となっている．これが，携帯機器になると，バッテリー駆動の稼働時間に直結し，より厳しい消費電力削減が要求される．

これに対して，我々は，VLIW方式によるプロセッサの，高性能化と低消費電力化の両立に着目し，画像処理プログラムをワークロードとして評価を行ってきた．[1),2)] 結果として，陽に並列性を持っており，コンパイル時に並列性を引き出せて効率の良いプログラムを生成できる場合は，VLIW方式がスーパスカラ方式などに対して，より良い性能と低消費エネルギーを実現できそうだということが分かった．しかし，システム全体を考えると，並列度を持ったアプリケーションだけでなく，システム制御と言った並列処理に向かないプログラムも存在し，これをVLIWプロセッサで実行すると，大変ハードウェア使用効率の悪い処理となってしまう．このため，VLIWプロセッサとは別に汎用プロセッサも採用してシステムを構成することになる．本稿では，この問題点を解消すべく，VLIWプロセッサでありながら，SMT(Simultaneous Multi-Threading)プロセッサのように仮想的なマルチプロセッサの環境を提供し，しかも，そこで実行される命令セットは，VLIW命令と汎用プロセッサ命令という異なった命令セットであるOROCHIプロセッサを提案する．

† 広島市立大学 大学院情報科学研究科
Graduate School of Information Sciences, Hiroshima City University
†† 広島市立大学 情報科学部
Faculty of Information Sciences, Hiroshima City University

2. 背　　景

2.1 VLIW方式のプロセッサをシステムに用いる利点と問題点

我々は，画像処理の中でも基本的な画面小区画の近似性を評価するSAD（Sum of Absolute Difference）ループを用いて，ステレオ画像処理[1]や，移動物体追跡システム[2]などを構築し，VLIWプロセッサと他のプロセッサの方式比較を行ってきた．使用したプロセッサの設計目的が異なるため，厳密には，公平な比較とはならないが，できるだけ評価基準を合わせ，VLIW方式が，性能向上と消費電力の削減の両立と言う点で優れているという感触を得ている．

この理由として，シンプルなパイプラインプロセッサに対しては，演算器などのプロセッサの一部分だけを増設することで得られる性能向上比が，元々の性能／ハードウェア量比を上回るためで，命令レベルの並列処理がうまく機能していると言える．また，同じ命令レベル並列処理を行うスーパスカラプロセッサに対しては，並列性検出を事前にソフトウェアで行うか，実行時にハードウェアで行うかの本質的な違いがあることで，ハードウェア量の差が生じることが挙げられる．また，省電力化設計にあたって，次実行命令のセレクト回路がクリティカルパスになるスーパスカラプロセッサでは，使用しない演算器に対するクロックゲーティングや電源遮断などの手法が適応しにくいが，VLIWプロセッサでは，命令を取り出した時点でその命令を実行するときに不必要となる演算器を確定でき，これらの手法を実現し易いと言う点も挙げられよう．

しかし，この特徴は，スーパスカラプロセッサは実行時でないと検出できない並列性にも対応できるが，VLIWプロセッサの場合は，コンパイル時に検出できなかった並列性（検出できても，並列に実行しても問題ないことが検証できなかった場合を含む）は，利用できないという逆の評価につながる．これは注意すべき点で，我々が行ってきた評価は，あくまでも陽に並列性を持つようなコンパイル時に並列性を抽出しやすいワークロードに対してのものであり，システム構築には必須となる，システム管理を行うOSコードなどについては，全く当てはまらない．もちろん，先ほど挙げたように省電力化設計を厳密に行っておれば，VLIWプロセッサで並列に動作できなくても電力消費は大きくないかもしれないが，これは本末転倒であろう．

また，VLIWプロセッサの場合は，並列度がコンパイル時に確定するため，並列度の予測が困難なスーパスカラプロセッサに比べて性能見積もりを行いやすい．しかし，OSが同一のプロセッサで走行すると，このためにメディア処理が中断し，メディア処理性能は低下する．OSの走行による影響は予測をつけにくく，厳密にVLIWプロセッサでの性能見積もりを行った意味がなくなってしまう．このように，画像処理などのメディア処理のシステム開発に，VLIWプロセッサのみを用いて高性能低消費電力を狙っても，OSを考えるとこの構成で充分とは言いがたい．このため，次に述べる「アプリケーション・プロセッサ＋汎用プロセッサ」の構成を取ることになる．

2.2 アプリケーション・プロセッサ+汎用プロセッサ

この「アプリケーション・プロセッサ＋汎用プロセッサ」の構成は，メディア処理システムの一つの解だと考える．実際，Cellプロセッサでは，SPE(Synergistic Processor Element)という演算に特化したブロックを複数個実装し，VLIWに比べると粒度の大きい並列処理でアプリケーション処理を行い，PPE(Power Processor Element)を設けて，管理処理を行う構成を念頭においていると考えられる．

この構成は，設計量が2プロセッサ分となることや，LSIの集積度が低い場合は複数チップ構成となると言う点で，不利であったが，1チップに複数プロセッサを搭載することが普通になり，また，汎用プロセッサコアもIPとして既存のものを流用できるようになって，問題とならなくなったと言える．別の表現をすると，CMP(Chip Multi-Processor)の構成に，均一構成のホモジニアスCMPと言う選択だけでなく，用途に応じて最適な命令セットを持ったプロセッサを組み合わせて，ヘテロジニアアスなCMPを構成することが可能になったと言える．

この発想はVLIWプロセッサとOS用汎用プロセッサの組み合わせにも適用することができ，VLIWプロセッサのチップに汎用プロセッサコアを搭載したチップは，このままでも，マルチメディア処理システムの一つの解となる．しかし，この構成を良く検討し，もう一工夫して低消費電力化を図れないか考えた結果が，次に述べる2つのプロセッサの融合である．

2.3 異種命令セットのSMT

VLIWプロセッサと汎用プロセッサの1チップ・ヘテロ・マルチコアシステムに対する改良は，SMT(Simultaneous Multi-Threading)と同様の発想による．すなわち，並列度の大きいスーパースカラプロセッサの場合，実行するプログラムの並列度が少なかったり，キャッシュミスなどで長時間ストールする場合など，多くの演算器は使用されない．この非効率さを緩和するために，同時に複数スレッドからの命令実行パイプラインに投入することで，演算器使用率を高めようと言うものである．

VLIWプロセッサの場合，1つの命令で指定できる演算は，命令長を短くするために実装されている同時に実行できる演算器の数より少ないことが多く，また，かなりの並列度を持つプログラムでもすべての命令スロットを使い切ることは少ない．このため，それほどの性能を要求されないのであれば，汎用プロセッサの

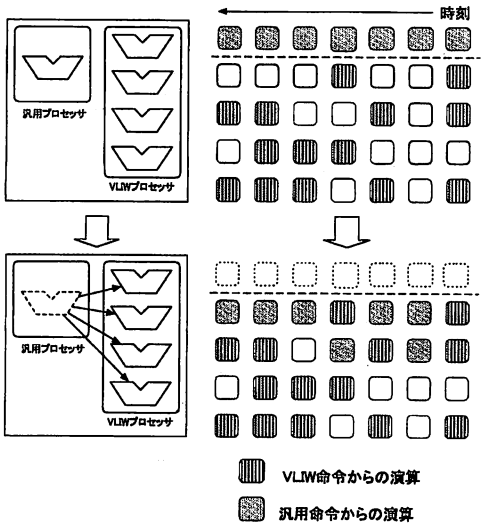

図 1 VLIW プロセッサでの SMT 概念図

命令は，VLIW プロセッサの使用されない演算器で実行できないか，と考えた．このイメージを図 1 に示す．

しかし，通常の SMT と大きく異なるところは，命令を供給するスレッドの命令セットが異なると言うことである．一般的に，異なる命令セットを同一のハードウェアで実行することはできない．しかし，古くは，マイクロ命令による機械語の実装の例や，最近では，Intel PentiumPro のように，ハードウェアが内部命令に変換して実行するような例もあり，各種の VM(Virtual Machine) 技術も参考にして，異種命令セットの実行を検討した．実際に実存する命令セットで，実行するハードウェアを共通化するのに問題となる機能を洗い出したところ，以下のような点が挙げられた．

- 主記憶のエンディアンの違い．ただし最近の命令セットは big/little の両方をサポートすることが多い
- 特権命令の定義は異なることが多い
- MMU 関連の仕様すなわち仮想記憶のアドレス変換方式
- データ形式．
  固定小数点データ　演算幅の違いがあるが，符号拡張などで対応可能．
  浮動小数点データ　ほとんどの場合は浮動小数点数は IEEE 形式に統一されている
  その他のデータ　十進データ形式をサポートするものがある
- コンディションコードの条件が異なる場合がある
- 命令の定義が複数の演算を含むことがある．例えば，複数レジスタのロード／ストアやロード／ストアとアクセスアドレスの更新など

これらを踏まえて考えると，対象としている「VLIW プロセッサと汎用プロセッサ」の組み合わせで，システム制御は汎用プロセッサで行って VLIW プロセッサでは行わないと言う前提であるから，VLIW プロセッサの命令セットとしては，MMU 制御などの命令を持たず，特権命令系の違いに悩むことはない．

また，命令の定義が複雑なものを含む場合の対応であるが，複雑な命令の定義を分析すると，他の命令セットで持っているような基本的な命令の組み合わせで実現できることが分かる．例えば，先ほど例示した，複数レジスタのロード／ストアは，単純なロード／ストアとアクセスアドレスを更新して行く機能で実現できるし，push/pop などのスタック操作を指向した命令も，単純なロード／ストアとアドレス更新の演算命令に分解できる．

その他の差異は，両機能を実装する／包含できる機能を実装するなどの対応を行っても，大きな問題となるものはなく，当初の目論みは，十分実現可能と考えられる．

2.4 VLIW の SMT

「VLIW プロセッサと汎用プロセッサ」の組み合わせは，もう一つ従来の SMT プロセッサと異なるところがある．それは，従来の SMT プロセッサが，命令実行部いわゆるバックエンドを，スーパスカラ構成を取っているのに対して，この組み合わせでは，必然的にバックエンドは VLIW 構成となることである．スーパスカラ構成であると，命令の依存関係が解決されるまで格納されるリザベーションステーションが存在し，各スレッドから取り出された命令は，これに単純に投入することで，実行される．異なるスレッドからの命令は，コンテキストが異なるので，依存関係が発生することはなく，並列に実行可能である．

しかし，バックエンドが VLIW 構成のときは，取り出された VLIW 命令は，次々とバックエンドに投入することができ，リザベーションステーションのような待ち合わせキューは設けないのが普通である．しかし，本提案方式では，VLIW 命令の空きスロットに，汎用プロセッサの命令を投入していかなければならないので，キューが必要となる．ただし，このキューは，図 2 に示すように，命令の先取りのための命令バッファと言う位置付けで良く，このために，ハードウェア量が大幅に増加すると言うことはない．

2.5 SMT の QoS

最後に留意した点は，SMT プロセッサの性能保証である．特に対策を施していないと，SMT プロセッサで実行される各スレッドは，何らかの優先順位は付けられているが，どのくらいの性能で走行できるかは，同時に走行しているスレッドの並列度に依存する．これでは，システムを設計する際，対象としているアプリケーションが所望の性能を達成できるか検証するこ

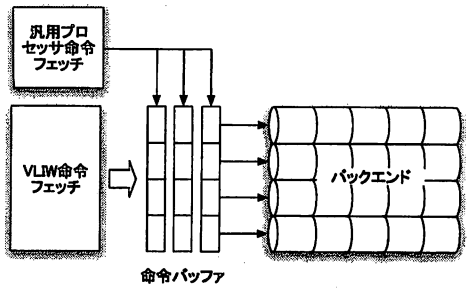

図 2　VLIW － SMT プロセッサのブロック図

とが困難である．このため，SMT プロセッサの各スレッドに対して一定の処理性能を保証するようなハードウェア機能が検討されており，[3)~5)]，また，市販の SMT プロセッサなどでも取り入れられようとしている．本研究の目標はメディア処理であるから，例えば，画像の処理速度を秒３０フレーム確保するとか，外部からの割り込みレスポンスを数ミリ秒以下に抑えるとか，当然要求されるものである．したがって，各スレッドに対して最低限の性能保証要求を設定させ，一定期間内の性能が要求を満足できないようなときは，バックエンドへの命令投入を調節して，性能保証することを考えている．

3. OROCHI の構成

本章では，背景で述べた研究目標に対して，具体的にどのような仕様のプロセッサを開発しようとしているのかを述べる．

3.1　命令セット仕様

命令セット仕様は，今までの研究に使用してきて設計のノウハウが蓄積されているという点を評価して，VLIW 命令には，FR-V 550 の命令セットからユーザーアプリケーションで使わないような機能を削除したサブセットを採用することとした．また，システム制御用の汎用プロセッサ命令は，ARM を用いることにした．システム制御のための OS をこの上で構築することを考えると，Linux などでサポートされている命令セットであることが望ましく，この点でも目的にかなったものである．ARM は RISC であると開発元は主張しているが，１命令で四則演算とシフト演算を逐次的にできたり，複数レジスタのロード／ストア命令があったりと，さすがにメモリ間演算はないが，ユニークな命令を持っている．FR-V の命令と比べると，お互いの基本的な命令と操作にはそれほどの隔たりはない．また，ARM の複雑な命令も，基本的な命令の組み合わせで実現できるので，問題とはならないと判断した．

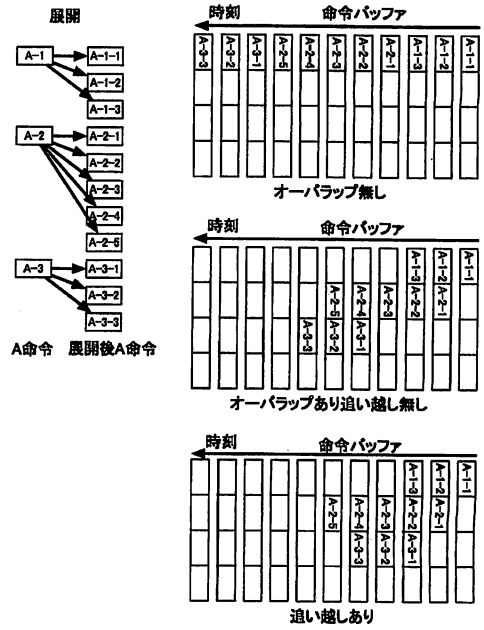

図 3　ARM 命令の詰め込み方

3.2　実行モデル

概略の OROCHI プロセッサの構成は，図 2 に示す通りであるが，細部についてどのように動作させるかについては，いろいろ検討を加えないといけない点がある．

まず，ARM 命令（以下 A 命令）の詰め込み方である．基本的に FR-V サブセット命令（以下 F 命令）の空きスロットに，A 命令もしくは A 命令を展開して生成された A 命令列を詰め込んでいくが，A 命令と F 命令間には依存関係は生じないので，演算の種類制限のみ考慮すれば空きスロットに詰め込んでいくことができる．しかし，A 命令間では，依存関係を生じる可能性があるため，図 3 に示すように，A 命令は全く並列に実行しないように埋め込んでしまうか，それともアグレッシブに 1 つの F 命令に複数の空きスロットがあるときは，依存関係を調べた上で複数の A 命令を並列に実行すべく詰め込むか，あるいは，レジスタリネーミングをおこなうなど，積極的に依存関係を解析解消してアウトオブオーダー実行まで目指すのか評価を行っていく必要がある．

また，バックエンドを VLIW 構成にしているので，詰め込まれた F 命令と A 命令は，必ず同時に実行され，どちらかの命令でストールが発生すると，待たなくても良い命令まで待ってしまうことになる．特にキャッシュミスのペナルティは大きいので，詰め込むときに演算の種類の組み合わせに制限を加えるなどの対策が効果を持つかどうかも，興味深いところである．

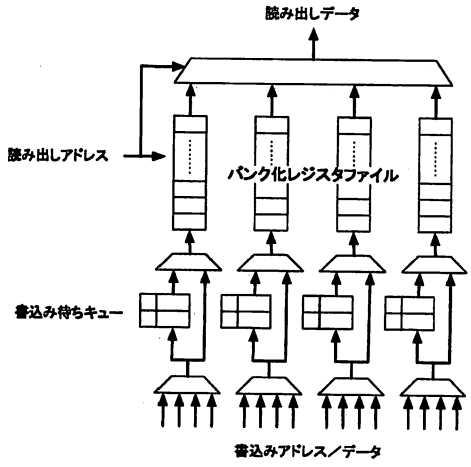

図 4　レジスタファイルの構成図

4. 開発に向けての準備

OROCHI プロセッサは，2つのスレッドの組み合わせで，いろいろな実行状態が考えられるため，シミュレーションなどで断片的な評価を行うことは，困難と考えた．そのため，実チップを試作して検証することを予定している．この試作には，既存のゲートアレイの LSI 製造サービスを利用し，回路設計の不良などによるリスクを最小に抑えようとしている．しかし，このため，多ポートのレジスタファイル用 RAM を使用することができず，1Read/1Write の RAM，あるいは 2Read/2Write の RAM を用いて構成する必要があり，まずその評価を行った．

4.1 レジスタファイルの構成

本評価で想定したレジスタファイルの構成を図 4 に示す．VLIW の構成として 8 並列（固定小数点系：4 と浮動小数点／メディア演算系：4）を考えているので，同時に必要となるレジスタファイルのポート数は，固定小数点レジスタの読み出しで，$2\times4+2(forstoredata)=10$ となる．また，浮動小数点レジスタは，$2\times4=8$ となるが，倍長データも扱うため，レジスタの奇数／偶数ペアでの読み出しとなる．読み出しの方は，レジスタファイルの RAM を多重化してコピーを作ることで，ハードウェア量としては問題となるが，一度に読み出すことが可能なため，考えないこととした．書込みは，結果がそれぞれ 4 個ずつ生成されるので，4 つのレジスタに同時に書込まなくてはならない．これは読み出しのようにハードウェア量をつぎ込めば解決できるものではなく，書込むためのサイクル数が増加してしまう．そこで，レジスタファイルをバンク化し，異なるバンクに対する書き込みであれば同時に書込めることを利用することにした．しかし，この場合も，書込み要求のバンクがぶつかる可能性があり，これに備えて書込み待ちキューを各バンクに備えることにした．しかし，このキューの段数があまり多くては，レジスタファイルそのものをラッチで作ってしまった方がハードウェア量を節約できるかもしれず，また，あまり少ないと満杯になってパイプラインを止めることになって性能に影響を及ぼすため，どのくらいの段数まで蓄積される可能性があるかを検討した．

表 1　バンク構成レジスタファイルの評価

レジスタの構成	ワークロード	最大キュー長
GR4	INT4	1
GR4	INT8	1
GR4	OPT	2
FR1-4	OPT-hand	8+
FR2-4	OPT-hand	2
FR1-8	OPT-hand	2

4.2 評　　価

評価のワークロードには，4 種類を用意した．

INT4　固定総数点数の 2 次元配列の内積を計算するプログラムのベクトルの積和ループを 4 重にループアンローリングしたもの．コンパイラは FR-V 550 用 gcc で，コンパイルオプションは，-O2 と-mcpu=fr550 を指定．

INT8　固定総数点数の 2 次元配列の内積を計算するプログラムのベクトルの積和ループを 8 重にループアンローリングしたもの．コンパイラは FR-V 550 用 gcc で，コンパイルオプションは，-O2 と-mcpu=fr550 を指定．

OPT　オプティカルフローを計算する画像処理プログラムのコアループ．コンパイラは，Softune で，コンパイルオプションは，-O4 を指定．

OPT-hand　オプティカルフローを計算する画像処理プログラムのコアループをメディア命令を用いてハンドコーディングしたもの．コンパイラは，Softune で，コンパイルオプションは，-O4 を指定．

評価対象のレジスタの構成は，固定小数点レジスタについては，

GR4　1Read/1Write の RAM を用いて 4 バンク構成としたもの

浮動小数点レジスタに対しては，

FR1-4　1Read/1Write の RAM を用いて 4 バンク構成としたもの

FR2-4　2Read/2Write の RAM を用いて 4 バンク構成としたもの

FR1-8　1Read/1Write の RAM を用いて 8 バンク構成としたもの

を対象とした．評価結果を，表 1 に示す．

固定小数点レジスタについては，今回の評価に用いたワークロードでは，書込み待ちキューに貯まるデー

タの最大個数は２であり，しかも，キュー内のデータは，一回のループが終了する時点で，常に無くなっていた．これから，固定小数点レジスタのバンク構成は，1Read/1Write の RAM の４バンク構成で，書込み待ちキューは２エントリ持てば充分と考えられる．

浮動小数点レジスタについては，固定小数点レジスタと同様の 1Read/1Write の４バンク構成では，キュー内に貯まるデータの最大個数は，８と多くなり，しかも，１回のループが終了する時点で，キュー内のデータは無くなっていないため，ループの回数が増えるにつれて，キューに貯まるデータ数も増えていく．これは，浮動小数点レジスタを用いるメディア演算命令の１命令で書込まれるレジスタが，複数レジスタである場合が多く，バンクの衝突が頻発するためである．このような状況から，浮動小数点レジスタのバンク構成は，1Read/1Write の RAM を用いる場合は，８バンク構成にして書込み待ちキューは２エントリ用意するか，2Read/2Write の RAM を用いる場合は，４バンク構成で書込み待ちキューのエントリは２つ用意する必要がある．

5. おわりに

VLIW プロセッサを用いてメディア処理システムなどを構築する場合問題となるシステム制御の効率的な実行環境について，追加の汎用プロセッサを用意するのではなく，VLIW プロセッサのハードウェア資源を有効利用して実現しようとする試みについて，その背景と概要を述べた．また，開発初期段階の準備として，レジスタファイッルの構成についても検討を加えた．今後，全体的な設計を行い，チップ試作を行う予定である．

謝辞 本研究は，半導体理工学研究センターとの共同研究による．また，本研究は，東京大学大規模集積システム設計教育研究センターを通し，シノプシス株式会社の協力で行われたものである．

参考文献

1) 清水雄歩，津邑公暁，中島康彦，五島正裕，森眞一郎，北村俊明，富田眞治: 距離画像生成処理におけるメディアプロセッサの評価，情報処理学会論文誌：コンピューティングシステム，Vol. 44,No.SIG 11(ACS 3),pp.257–267 (2003)
2) 島田貴史，酒井智也，北村俊明: 組み込みプロセッサを用いた動体検出システムの構築と評価，情報処理学会研究会報告 (SWoPP2005), Vol.2005-ARC-164, pp.31–36 (2005)
3) Francisco J. Cazorla, Alex Ramirez, Mateo Valero, Peter M.W. Knijnenburg, Rizos Sakellariou, Enrique Fernandez : QoS for High-Performance SMT Processors in Embedded Systems, IEEE Micro, Vol.24, No.4, pp.24–31(2004)
4) Joseph Sharkey, Dmitry Ponomarev : Balancing ILP and TLP in SMT Architectures through Out-of-Order Instruction Dispatch, International Conference on Parallel Processing (ICPP'06), pp.329–336 (2006)
5) Francisco J. Cazorla, Peter M.W. Knijnenburg, Rizos Sakellariou, Enrique Fernandez, Alex Ramirez, Mateo Valero : Predictable Performance in SMT Processors: Synergy between the OS and SMTs, IEEE Transactions on Computers, Vol.55, No.7, pp.785–799 (2006)