計算機アーキテクチャ 119－19
（1996．8．28）

教育研究用スーパスカラ・プロセッサ・シミュレータMikageの概要

土江竜雄† 佐々木敬泰†
弘中哲夫† 児島彰†

Mikageは様々なスーパースカラ・プロセッサの内部動作をレジスタトランスファレベルでプロセッサ・アーキテクチャをシミュレートし観測することが可能なスーパースカラ・プロセッサ・シミュレータである．Mikageを用いることで迅速にプロセッサ・アーキテクチャやオブジェクトコード最適化技法の評価を行うことが可能にある．本稿ではMikageの開発構想，および，シミュレートするアーキテクチャ仕様について述べる．

Superscalar Processor Simulator 'Mikage' for Research and Education

TATSUO TUCHIE †, TAKAHIRO SASAKI †, TETSUO HIRONAKA †
and AKIRA KOJIMA †

Mikage is a software simulator of a superscalar processor with the ability to simulate various processor architectures. Mikage makes it possible to rapidly evaluate various processors and methods for object code optimization. In this paper,we discuss the concept of Mikage and its ability of simulating architectures.

1. はじめに

現在，機械命令レベルの並列処理を行うプロセッサであるスーパスカラ・プロセッサやVLIWプロセッサなどが広く普及し始めている．しかしながら，これらのプロセッサを使用するにあたりプロセッサ・アーキテクチャの性能を十分に引き出せていない場合が多い．これは，機械命令レベル並列処理を行うプロセッサの実効性能がプロセッサ・アーキテクチャだけでなく，コンパイラで採用されるオブジェクト・コード最適化技法により大きく左右されるためである．つまり，スーパスカラ・プロセッサ・アーキテクチャを設計するに当り，適用するオブジェクト・コード最適化技法を十分に考慮する必要がある．また，逆にスーパスカラ・プロセッサを設計するに際にさまざまなハードウェアによる高速化手法を導入することが次々と研究されている [Ando95] [Koseki95] [Kaneoka95] [Hironaka96b] が，それぞれの高速化手法を設けることによるハードウェアの増加と得られる性能を十分に評価し，実際に導入するべきか評価する必要がある．また，最適化技法によってはスーパスカラ・プロセッサのハードウェアで実現することも，コンパイラのオブジェクト・コード最適化技法としても実現できるものがあり，いずれの方法を採用するべきかを選択する必要があるものもある．このようにさまざまな条件を組み合わせて，最適なプロセッサ設計を行う必要がある．このような状況下において最適なプロセッサ設計を行うには，さまざまな組合せでプロセッサ・アーキテクチャ上の選択しと，オブジェクト・コード最適化技法を評価する環境が必要である．本研究ではこのような評価を迅速に行うことが可能なハードウェア／ソフトウェアの評価用ワークベンチの作成を目標としている．

2. Mikageの概要

我々は上記のワークベンチを実現するため，スーパスカラ・プロセッサ・アーキテクチャの様々な選択肢における効用をシミュレーションすることにより，評価を容易に行うことのできるシミュレータを開発中である．本シミュレータはSPARC V8命令セット・アーキテクチャ [SPARC92] に基づいた命令セット・アーキテクチャを採用している．従来の関連研究として我々同様

† 広島市立大学 情報科学部 情報工学科
Department of Computer Engineering, Faculty of Information Science, Hiroshima City University

にSPARCの命令セット・アーキテクチャをシミュレートするシミュレータとして [Yasuda95] らが開発した sparcemu などが存在する．しかし，これらのSPARCシミュレータは，単に命令セットをシミュレートするだけである．一方，我々が開発しているのは，プロセッサのハードウェアをシミュレートするものである．我々が開発中のシミュレータは以下の目標をもつ．

- プロセッサ・アーキテクチャの性能評価
- オブジェクト・コード最適化技法の評価
- キャッシュ・アーキテクチャ評価
- ハードウェア / ソフトトウェア教育

以下の節でそれぞれの目標について詳細に述べる．

2.1 プロセッサ・アーキテクチャの性能評価

アーキテクチャの決定には高速化技法の導入に伴うハードウェア量の増大と，プロセッサの処理速度の向上のトレードオフを十分に吟味しなくてはならない．現在さまざまなスーパスカラ・プロセッサ・アーキテクチャの高速化技法が知られているが，共通の土台においてのトレードオフの評価はまだ十分にされていない．本シミュレータの使用により，その高速化技法の有無または適用法の変化によりどの程度処理速度の向上が図れるか，またどの部分がボトルネックになっているかなどといった動作レベルでの変化や状況の把握がより容易となる．これにより，設計コストの高い詳細設計の段階に入る前に，バランスを考慮した取捨選択が可能になる．

2.2 オブジェクト・コード最適化技法の評価

スーパスカラ度を上げることによってプロセッサの最大性能を上げたとしても，命令コードがスーパスカラ度に見合う並列度を持っていなければそのプロセッサによる実効性能の向上は望めない．スーパスカラ・プロセッサの能力を引き出すものとして，コンパイラのオブジェクト・コード最適化による，命令コードの並列度の抽出は欠かせないものである．しかしながら，プロセッサ・アーキテクチャとオブジェクト・コード最適化技法は，それぞれ独立なものとして実効性能に寄与するのではなく両者の組合せとして実効性能に寄与する．以上の理由から，機械命令レベル並列処理を行うプロセッサにおいてプロセッサ・アーキテクチャとオブジェクト・コード最適化技法の関係を研究することが，より高性能なスーパスカラプロセッサを実現する上で重要である．本シミュレータでは，パイプラインの使用状況が理解しやすくなっているほか，プロセッサ・アーキテクチャが採用する高速化技法や，スーパスカラ度の変更が可能であるのでさまざまな状況下でのプロセッサ・アーキテクチャとオブジェクト・コード最適化技法の関係を評価できる．

2.3 キャッシュ・アーキテクチャ評価

スーパスカラ・プロセッサでは，スーパスカラ度に見合う十分なデータ供給バンド幅を確保する必要がある．そのためにはロード／ストア・ユニットの多重化が重要であるが，これを実現するには同一サイクルに多重度分に対応するデータ供給バンド幅でデータを供給するデータ・キャッシュが必要である．このような高いデータ供給バンド幅を実現するマルチポートのデータ・キャッシュの実現方式を研究するには非常に多くのキャッシュ構成方式をスーパスカラプロセッサの挙動を考慮しながら検討する必要がある [Sasaki96].

2.4 ハードウェア / ソフトウェア教育

現在の最新のマイクロ・プロセッサが当たり前のように採用しているパイプライン・アーキテクチャ，スーパスカラ・アーキテクチャ，アウト・オブ・オーダ命令発行，レジスタ・リネーミングといった高速化技術に関する教育が大学で十分に行われていないのが現状である．このような状況に対して [Iwaihara93], [Inoue95] や，[Matsumoto93] らがハードウェア設計製作を行う学生実験という形で対処することを提案および実践している．しかしながら，高度なアーキテクチャ教育に関してはこのようなアプローチをとると回路規模が巨大になること，また，意味のある学習を所定時間内に行うことが難しい．このような問題に対し，我々はこの教育を観測性が非常よいソフトウェア・シミュレータを用いてハードウェア / ソフトウェア教育を行うことを提案している [Hironaka96a].

3. ステージ構成

ここで，本シミュレータが想定している命令パイプライン・アーキテクチャについて説明する．

3.1 命令パイプライン構成

命令パイプラインは4ステージから成り，命令ブロックのフェッチとプリデコードを行うIFステージ，データ依存解析，オペランドフェッチ，ジャンプ命令の実行を行うDステージ，命令の実行を行うEステージ，レジスタファイルへの書き込みを行うWステージで構成されている．ただし，分岐命令は早期分岐解消のため，実行をDステージで行っている．図1に命令の種類ごとのパイプライン処理過程を示す．

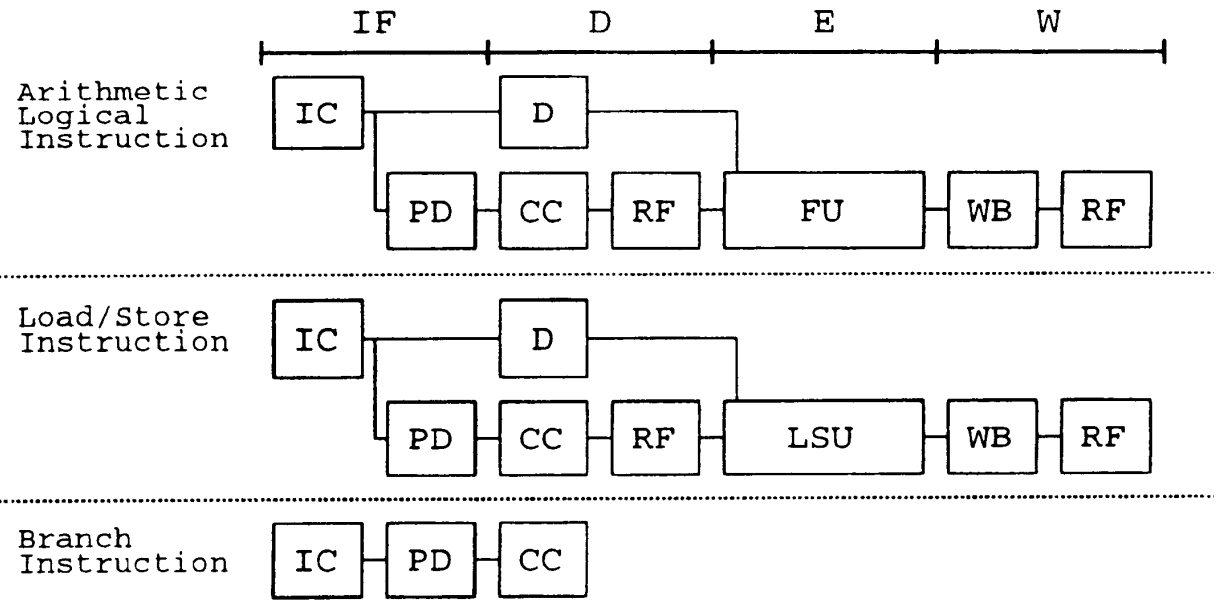

図1 Instruction Pipelines

3.1.1 IF ステージ

(a) 命令ブロックのフェッチ

プログラムカウンタ (PC:Program Counter) の指すアドレスから連続するパイプライン数分の命令を同時に命令キャッシュ (IC:Instruction Cache) から読み出す.

(b) 分岐予測

命令フェッチと同時に，過去の分岐の履歴とTA(Target Address) を読み出す．履歴情報から分岐予測アルゴリズムに従って予測を行う．また，分岐命令は逐次実行を行うため，同一命令ブロック内に分岐命令が複数存在する場合，2番目以降の分岐命令はブロックされる.

(c) nPC の更新

次サイクルでフェッチすべき命令の先頭アドレスをnPC に書き込む．nPC の候補として以下のものから選択する.

(1) PC+(今回のフェッチ命令数): 通常の連続フェッチにおける次のフェッチ先アドレス

(2) TA(Target Address): 分岐予測の結果がTAKEN の場合の予測分岐先アドレス.

(3) 再フェッチアドレス: 分岐予測が外れた場合の正しい分岐先アドレス.

これらのうち (3) がもっとも優先度が高く，(2) → (1) の順に低くなる.

(d) NOP 置換

分岐予測の結果がTAKEN の場合，当該命令と遅延命令 (Delay Instruction) を残してその後続命令をNOP に置換する.

(e) プリデコード

フェッチしてきた命令ブロック内の命令全てに対してプリデコードを行ない，命令の種類，使用する機能ユニット，ソース／ディスティネーションレジスタ番号など，依存解析 (D ステージ) に必要な情報を得る.

3.1.2 D ステージ

(a) データ依存関係の検出

プリデコードの結果より，命令ブロック内の命令間および先行命令に対してフロー依存，出力依存の検出を行う．依存は，各命令のソースおよびディスティネーションレジスタ番号を総当たりで比較することにより検出する．なお，本アーキテクチャはレジスタへの書き込みをパイプラインの最終ステージのみおこなっているので，逆依存は構造上起こり得

ない.

(1) 命令ブロック内の命令同士について

フロー依存の検出は，命令ブロック内の全ての命令に対して，その命令のソースレジスタ番号(最大2個)と先行命令のディスティネーションレジスタ番号(最大パイプライン数－1個)について行い，出力依存の検出は，その命令のディスティネーションレジスタ番号(1個)と先行命令のディスティネーションレジスタ番号(最大パイプライン数－1個)について行う.

(2) 先行命令について

フロー依存は各命令のソースレジスタ番号(最大2個)，出力依存はディスティネーションレジスタ番号(1個)についてスコアボードのレジスタ書き込み予約フラグが立っているか調べる．ソースレジスタ番号に書き込み予約があればフロー依存，ディスティネーションレジスタ番号に書き込み予約があれば出力依存である.

(b) データ依存によるハザードの回避

データ依存関係の検出により依存関係にある命令に対して，ハザードの回避を行う.

(b') 分岐命令の実行

分岐条件コードより分岐結果(TAKEN/NOT-TAKEN)を判定する.

(i) 分岐結果より分岐予測が正しければ，分岐結果をキャッシュにフィードバックして分岐命令を終了する.

(ii) 分岐予測が外れた場合，当該命令以降のパイプラインをフラッシュし，正しい分岐先アドレスをnPCに渡す．その後，分岐結果をキャッシュにフィードバックして分岐命令を終了する.

(c) 機能ユニットの競合検出

(a)によりデータ依存関係にない命令に関して，機能ユニットの使用による競合の検出を行う．もし競合がある場合には先行命令が優先され後続命令はブロックされる.

(d) オペランドのフェッチ

現段階で命令発行可能な命令に対してソースレジスタの読み出しを行う．レジスタファイルの読み出しポートは十分にあるものとしているので，ここで競合は発生しない.

(e) 命令の発行

オペランドの揃った命令は，次のクロックにEステージに進む．ただしその命令の発行先機能ユニットにてインターロックが生じている場合はブロックされる．発行が可能であれば，スコアボードに書き込み予約を行う.

(f) インターロック制御

命令ブロック内に発行をブロックされた命令がある場合，当該命令はDステージにてインターロックされる.

3.1.3 Eステージ

(a) 命令実行

Dステージより発行された命令は，各機能ユニットにおいて独立して実行される．各機能ユニットの実行時間はプロセッサの設定による.

(b) Wステージバッファへの登録

実行が終了した命令は次のクロックにWB(Wステージバッファ)に登録される．ただし，WBに空きスペースがない場合，その命令はブロックされる．その際の優先権は登録を要求する機能ユニット内でローテーションを行うことで機能ユニット間の偏りをなくしている.

(c) インターロック制御

実行ステージにおいてブロックされている命令が存在する場合，その機能ユニットはインターロックされる.

3.1.4 Wステージ

(a) 命令の読み出し

WBよりポート数分の命令の読み出しを行う.

(b) レジスタファイルへの書き込み

読み出した命令により，レジスタファイルへの書き込みを行う．スコアボードの書き込み予約を取り消す.

4. Mikageの提供するGUI環境

Mikageは X Window System ベースのプログラムで，プロセッサの設定，シミュレーションの実行を通してすべてマウスで操作可能となっている．また，設定および実行はすべてウィンドウ上で行う．ウィンドウは，プロセッサの制御を行なう CONTROL ウィンドウ，設定を行う OPTION ウィンドウ，レジスタ観測用の REGISTER ウィンドウ，整数ユニットおよび命

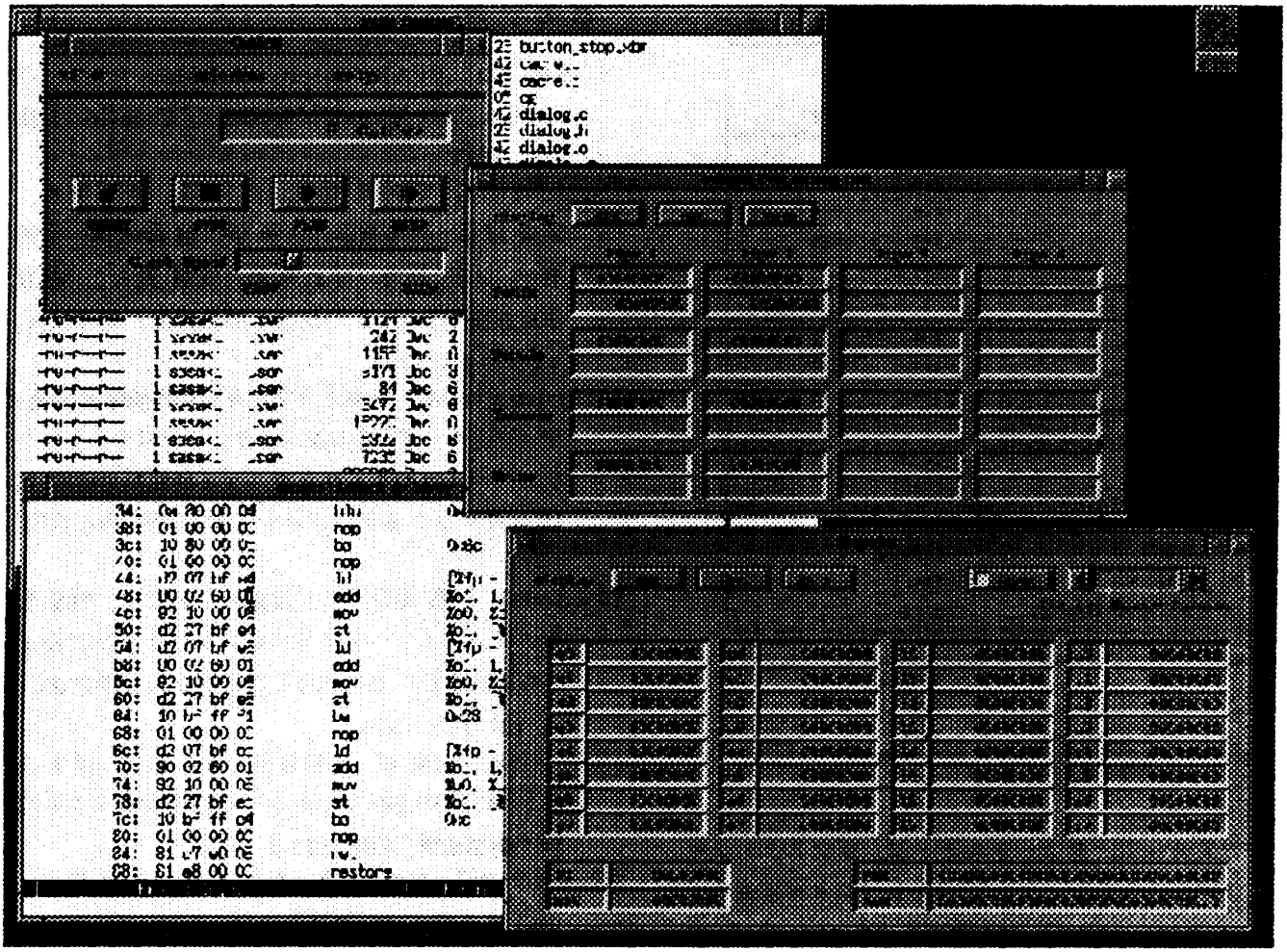

図2 環境風景

令フェッチ，デコード，レジスタ・メモリ書き込み観測用の IPU(Instruction Processing Unit) ウィンドウ，乗算等の終了までに数クロックを要するユニット観測用の FU(Function Unit) ウィンドウで構成されている．図 2に環境例を示す．

- **CONTROL ウィンドウ**
  このウィンドウでは仮想プロセッサの制御を行う．実行，停止，リセットのほか，他のウインドウを開くことも，このウィンドウで行う．
- **OPTION ウィンドウ**
  このウィンドウでは，想定するプロセッサの設定を行う．変更可能なプロセッサの設定項目を表1に示す．
  その他，以下に述べる機能の有無の選択が可能である．
  - インターロック制御
    - アウト・オブ・オーダ実行
    - レジスタリネーミング
  - 命令キャッシュ
    - 分岐予測
    - 分岐予測アルゴリズム
  - データキャッシュ
    - ノンブロッキングキャッシュ

  ただし，ここで述べた設定は GUI がサポートしている範囲であり，シミュレータ自身に明示的な制約はない．
- **REGISTER ウィンドウ**
  このウィンドウでは現在プロセッサからアクセス可能なレジスタがすべて観測できるようになっている．整数レジスタに対し，16 進数，10 進数，アスキー表示の選択が可能．また，レジスタウィンドウを変更することで他のレジスタを観測することも可能である．
- **IPU(Instruction Processing Unit) ウィンドウ**
  このウィンドウでは整数ユニットおよび命令フェッ

表 1 プロセッサ設定項目

変更項目	個数	サイクル数
パイプライン本数	1-4	
レジスタウインドウ数	2-32	
ロードストアユニット	1-4	1-8
整数ユニット		
ALU	1-4	1
乗算器	1-4	1-8
除算器	1-4	1-8
シフタ	1-4	1
浮動小数点ユニット		
ALU	1-4	1-8
乗算器	1-4	1-8
除算器	1-4	1-8
W ステージ		
ポート数	1-28	
バッファ	0-32	
命令キャッシュ		
セット数	1-8192	
ウェイ数	1-8	
ラインサイズ (words)	1-64	
データキャッシュ		
セット数	1-8192	
ウェイ数	1-8	
ラインサイズ (words)	1-64	
ミスヒット時の遅延クロック		1-8

チステージ，デコードステージ，レジスタ・メモリ書き込みステージの IP/IR レジスタが観測できる．デコードステージ以降はニーモニック表示が可能である．なお，フェッチステージでのニーモニック表示も，プログラム起動時のオプションとして用意している．

- **FU(Function Units) ウィンドウ**
  このウィンドウでは各機能ユニット内の命令が処理されていく様子を表示する．

5. おわりに

以上，現在開発中であるスーパースカラ・プロセッサ・シミュレータ 'Mikage' の概要について述べた．Mikage は，様々なプロセッサ・アーキテクチャをシミュレートできる環境を提供している．そしてそれは教育や研究，評価に十分利用可能である．今後の課題として，割り込みなどの機能のサポート，GUI 環境の改善等があげられる．

参考文献

[SPARC92] SPARC International Inc.：“The SPARC Architecture Manual, Version 8,” Prentice Hall, Inc., 1992.

[Iwaihara93] 岩井原 瑞穂，山家 陽，中川 智水，國貞 勝弘，斉藤靖彦，永浦 渉，池 兼次郎，中村 秀一，山田 哲也，村上 和彰，安浦 寛人：“教育用計算機 QP-DLX の開発と開発環境,” 信学技法，VLD93-86, 1993.

[Matsumoto93] 松本：“プロセッサ作成学生実験 - チップ，ボードからコンパイラ，アプリケーションまで,” マイクロエレクトロニクス研究開発機構 第 12 回ワークショッププロシーディング，pp.1-12, 1993.

[Ando95] 安藤 秀樹，中西 知嘉子，原 哲也，中屋 雅夫:“プレディケート付き状態バッファリングによる投機的実行,” 並列処理シンポジウム JSPP'95，pp.107–114, 1995.

[Koseki95] 古関 聰，小松秀昭，鈴木 英俊，深澤 良彰：“拡張 VLIW プロセッサ GIFT の命令供給機構,” 情処研報，ARC-113-6, 1996.

[Kaneoka95] 金岡 弘記，高木 浩光，有田 隆也，川口 喜三男：“命令再構成型 VLIW プロセッサ V++ における 2 つの再構成機能の評価,” 情処研報，ARC-113-8, 1996.

[Inoue95] 井上 広士，飯田 全広，大内 正英，久我 守弘，末吉 敏則：“32 ビット RISC マイクロプロセッサ DLX− FPGA の設計教育フィジビリティ・スタディ,” 情処研報，ARC-115-18,DA-78-18,1995.

[Yasuda95] 安田 絹子，久曽神 宏，荻野 達也：“オペレーティングシステム開発環境の設計と実装,” 信学技法，CPSY96-38,1995.

[Sasaki96] 佐々木 敬泰，土江 竜雄，弘中 哲夫，児島 彰：“スーパスカラ・プロセッサ用データ・キャッシュの実現方式の検討,” 情処研報，ARC(SWoPP96), 1996.

[Hironaka96a] 弘中 哲夫，土江 竜雄，佐々木 敬泰：“コンパイラ教育用スーパスカラ・プロセッサ・シミュレータ,” 情報科学のための教育用マイクロプロセッサの開発，平成 7 年度科学研究費補助金 (試験研究 B(1)) 研究成果報告書，1996.

[Hironaka96b] Hironaka, T.：“Speculative Execution by Compiler Supported Branch Prediction Hardware,” 情処研報，ARC-118-4, 1996.