計算機アーキテクチャ 117－9
（1996．3．5）

ロード・アドレス予測方法の検討

勝野 昭 木村康則

ILP(Instruction Level Parallelism)の向上を目的とし，投機的なメモリ・システムのアクセスと後続命令の実行を行なうため，ロード命令のアドレスの予測方法について検討する．本稿では，主に，状態ビットを用いたアドレス変位Δの更新方法とΔの最適なビット長についてシミュレータによって評価する．4ウェイ,512エントリ,7ビットのΔ,1ビットの状態ビット構成のロード・アドレス・キャッシュについて測定した結果，予測成功率は平均70%以上であった．

An Address Prediction Scheme for Load Instructions

AKIRA KATSUNO and YASUNORI KIMURA

This paper describes an address prediction scheme for load instructions in order to improve instruction level parallelism, which allows a processor to do both memory access and instruction execution speculatively. We evaluate a method of updating delta by using a state bit and bit length of delta. Experimental results are given to demonstrate that prediction accuracy of a load address cache with 4-way, 512-entry, 7-bit delta and 1-bit state bit is over 70%.

1. はじめに

今日の高速マイクロプロセサは，命令レベル並列度ILP(Instruction Level Parallelism)を向上させるため，スーパスカラ，スーパパイプライン，(V)LIWなどの技術を用いて作られている．この中で特に既存のバイナリ・プログラムを実行できるスーパスカラ方式が優位に立っている．

一般に，スーパスカラとともに用いられるILP向上の技術に，out-of-order実行と投機的実行がある．out-of-order実行は，プログラムの順序どおりに命令列を実行するのではなく，命令間の依存性や演算器などのリソースのコンフリクトをハードウェアで検出しそれらが解消でき次第その命令を実行する，すなわち，プログラム順序とはout-of-orderに命令を実行する方法である．

投機的実行として，分岐命令に関する方法が用いられている．これは，分岐命令に関して，分岐するか/しないかという予測を行ない，その予測に基づいて後続の命令列を実行する方法である．そして，プロセサは，分岐予測が外れたときにその分岐命令直前の状態までリカバリすることができる機能を備えている．

今日の最高速プロセサは，このようなout-of-order実行と投機的実行の機能を備えるスーパスカラ方式によって構成されている．しかしながら，4命令発行幅のスーパスカラ・プロセサ[1)~5)]のIPC(Instructions Per Cycle)は，高々1.5～2.0であることがわかってきた．そのため，よりIPCを向上させる技術が求められている．

ところで，一般にプロセサは，メモリ・システムからデータを持ってきて，そのデータに対して何らかの処理を行ない，得られた結果のデータをメモリ・システムにストアするという順序でプログラムを実行する．従って，高速化のためにはロード命令の処理が鍵である．例えば，データ・キャッシュ・レイテンシが2サイクルであるスーパスカラ・プロセサについて，仮に，レイテンシがゼロになったとすると約15%のIPC向上が得られる．つまり，ロード命令のアドレスを予測し，投機的にメモリ・アクセスを実行することによって，IPCを10%以上向上できる可能性がある．さらに，ロード命令のアドレス計算やその計算に用いるレジスタの依存関係を考慮する必要なくアドレスを予測することができれば，実際にはそれ以上のIPC向上が期待される．

以上のように，IPC向上を目的とし，投機的なメモリ・システム・アクセスと後続の命令の投機的実行を行なうために，ロード・アドレス予測方法について検討する．次章で，その予測方法について述べた後，作成したシミュレータとそれを用いた測定結果とその考察について記述する．

† （株）富士通研究所 プロセサ研究部
Processor Laboratory, Fujitsu Laboratories LTD.

2. ロード・アドレス予測方法

2.1 概　　要

ロード命令のメモリ・アドレスの予測方法について，Dr. R.J.Eickemeyer らの研究[6] が知られている．彼らは，IBM ESA/390^{TM} アーキテクチャに対する評価を行なっている．我々は，RISC プロセサの一つである SPARC^{TM} アーキテクチャ[7] について評価を行なう．

まず，ロード・アドレス予測を用いた場合，命令の依存性にどのように影響するかを説明する．次のような命令列を考えてみる．

```
(1)sethi  idd, r20
(2)or     r20,2de,r20
(3)ld     r20,r10,r08
(4)ld     r26,r10,r23
(5)sub    r23,r08,r23
(6)....
```

この命令列の真のレジスタ依存関係は，

```
(1) -> (2) -> (3) -> (5)
               (4) -> (5)
```

である．これに対して，ロード・アドレス予測が成功した場合，依存関係は，

```
(1) -> (2)
(3) -> (5)
(4) -> (5)
```

のようになり，依存関係の深さは，1/2 になる．従って，ILP が増加し，高速処理が期待できる．

2.2 予測アルゴリズム

基本的な各ロード命令に対する予測メモリ・アドレスは，

予測アドレス ＝ 前回のアドレス ＋ Δ，
Δ ＝ 前回のアドレス － 前前回のアドレス

と表される．

ハードウェアとしては，LAC(Load Address Cache) と呼ぶ一種のキャッシュを用いる (図1参照)．LAC は，プログラム・カウンタをタグ情報として，ロード命令についてメモリ・アドレス部 (前回のアドレスを保持) とアドレス変位Δ (前回のアドレスと前前回のアドレスの差) からなる．

予測アドレスを Ap，LAC からリードしたアドレスと差をそれぞれ Ar とΔ r，LAC へのライトを Aw とΔ w，正しいアドレスを Ac とすると，予測アドレス Ap と LAC の更新方法は，

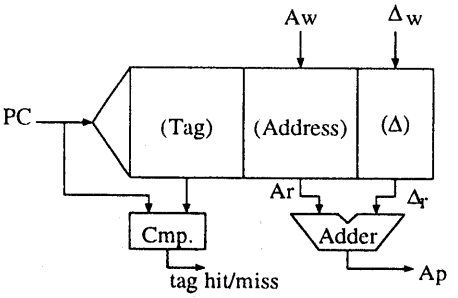

図1 ロード・アドレス・キャッシュの構成
Fig. 1 Block diagram of a load address cache

```
Ap = Ar + Δr;
if (アドレス予測が成功)
  then Aw = Ap, Δw = Ap - Ar = Δr;
else  /* 予測が失敗 */
  then Aw = Ac, Δw = Ac - (Ap - Δr);
```

となる．

例えば，あるロード命令が，

```
PC[5:2]='1010': load{rs1 + rs2} -> rd;
```

であり、最初に実行されたときのアドレス (rs1 + rs2) が '10010000' であり，2回目に実行されたときのアドレスが '10010010' であるとすると，LAC のアドレス '1010' には '10010010' と '10' がライトされる．そして3回目にこのロード命令がフェッチされたとき，予測アドレスは，'10010010'+'10'='10010100' となる．

2.3 オプション

我々は，この予測方法に対して次の2つのオプションを提案する．

(1) 予測アドレスが連続して2回間違ったときのみ，デルタを更新する．
(2) LAC に格納するアドレスを物理アドレスとする．

前者は，LAC のヒット率を向上させることが目的である．LAC の各エントリは図2に示すように1ビットの状態ビットを持つ．状態ビットが '1' であるとき，予測が失敗すると状態ビットは '0' になる．さらにこの状態で予測が失敗すると，Δの値を更新する．一方，予測が成功すると状態ビットは '1' になる．この方法は，後述のシミュレーションで検証する．

後者は，LAC に MMU のアドレス変換機能を持たせる方法である．これによって，PC 値から直接ロード命令の物理アドレスを予測することができる．我々は，このような LAC と小容量でかつ高速な1次データ・

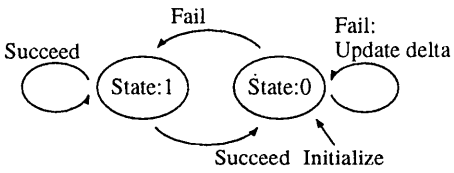

図2 状態ビットのステート・マシン
Fig. 2 State machine of the state bit

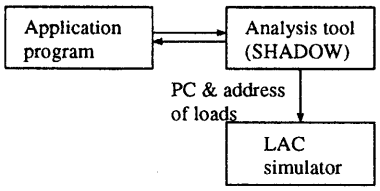

図3 評価シミュレータ (LACsim)
Fig. 3 Simulator for a load address cache

キャッシュを持つスーパスカラ・プロセサを目標としている．マルチ・プロセサに備えて，1次データ・キャッシュは，物理アドレス・タグを持つ．この構成では，フェッチ・ステージでLACアクセスと予測アドレス計算を行ない，次のデコード・ステージでその予測アドレスを用いて1次データ・キャッシュをアクセスすることによって，実行ステージでロード命令の予測データが後続の命令列にとって使用可能となる．これは，アドレス予測が成功した場合，ロード命令のレイテンシがゼロ・サイクルであることを意味する．本方法の詳細については，別の機会で紹介する予定である．

3. シミュレータ

ロード・アドレス・キャッシュの性能を評価するためにシミュレータ (以下，LACsim) を作成した (図3)．LACsimは，命令レベル解析ツールSHADOW[8]の解析モジュールである．LACsimは，SHADOWから各ロード命令のプログラム・カウンタと実効メモリアドレスの履歴を受けとる．そして，LACsimは，予測アドレスの生成とその予測アドレスの予測成功率を算出する．

4. 評価プログラム

ベンチマーク・プログラムとして，SPECint92より，gcc (入力ファイル: 1stmt.i), xlisp (li-input.lsp), espresso (ti.in), eqntott (int_pri_3.eqn), compress (in), sc (loada1), SPECfp92より，fpppp (natoms), doduc (doducin), wave5, tomcatv, nasa7, hydro2d (hydro2d.in) の合計12個を選んだ．

それぞれのプログラムの総実行命令数とロード命令数とその比率を表1に示す．総命令数の約20%がロード命令であることがわかる．fppppだけは，総実行命令数に対して，ロード命令の割合が41.9%にもなる．

表1 ベンチマーク・プログラム特性
Table 1 Characterization of benchmark programs

プログラム	総命令数 (a)	ロード命令数 (b)	(b)/(a)
gcc	78M	14M	18.4%
xlisp	1023M	241M	23.6%
espresso	742M	158M	21.3%
eqntott	1611M	279M	17.3%
compress	94M	17M	17.9%
sc	889M	177M	19.9%
fpppp	1547M	648M	41.9%
doduc	1349M	307M	22.8%
wave5	4153M	941M	22.7%
tomcatv	1300M	410M	31.5%
nasa7	8229M	2041M	24.8%
hydro2d	7296M	1608M	22.0%

5. 測定結果

LACsimにより，gcc, eqntott, fpppp, nasa7について測定した結果を図4～図7に示す．各図のX軸は，LACのウェイ数を2のn乗で表し，それぞれ，ダイレクト・マッピング，2ウェイ，4ウェイ，8ウェイのセット・アソシアティブ構成であることを表す．タグ・ミス時のリプレースメントのアルゴリズムは，全てLRUである．Y軸は，タグ・ヒット率あるいはアドレス予測成功率を表す．各図の破線はLACのタグ・ヒット率を，実線はアドレス予測成功率を示す．そして，各6本の線は、ウェイあたりのエントリ数を変えた場合のヒット率を示し，下からそれぞれ，128, 256, 512, 1K, 2K, 4K-entry/wayである．測定条件は，

- デルタΔのビット長はPCと同じ32ビット．
- 1ビットの状態ビットによるΔの更新（オプション(1)）

である．

タグ・ヒット率が高いということは，プログラム中にロード命令の局所性があることを示す．従って，もし，タグ・ヒット率が低ければ，当然，それにつれて，アドレス・ヒット率も低下する．例えば，fppppは，ロード命令の割合が高く局所性もないためにタグ・ヒット率が極端に低い．しかし，タグ・ヒットに対する予測成功率は，96%である．eqntottとnasa7については，タグ・ヒット率も予測成功率も非常に高い，すなわち，本予測方法が非常に有益であることがわかる．

表2に，各ベンチマークに対する4ウェイ,512エントリのLACのタグ・ヒット率と予測成功率を示す．タグ・ヒット率は，12個のベンチマークの平均で89.5%であり，fpppp:12.1%, doduc:76.2%, gcc:90.5%とsc:97.1%を除くと99%以上であることがわかる．また，全てのロード命令に対する予測成功率は，平均

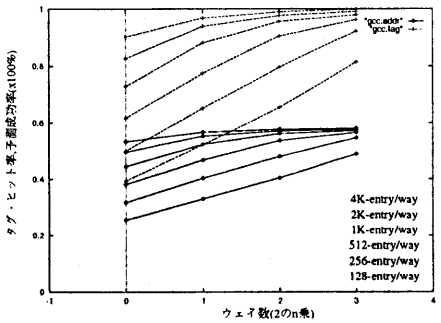

図4 測定結果(gcc)
Fig. 4 Simulated result(gcc)

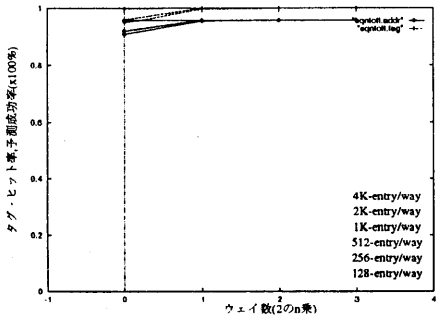

図5 測定結果(eqntott)
Fig. 5 Simulated result(eqntott)

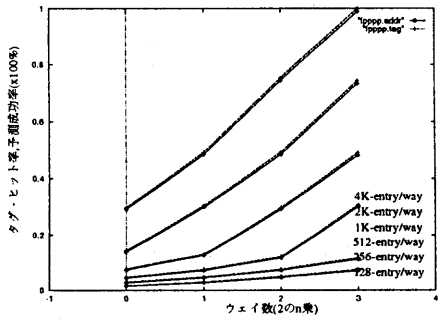

図6 測定結果(fpppp)
Fig. 6 Simulated result(fpppp)

74.7%である．特にタグ・ヒット率の低いfppppを除くと，80%以上の予測成功率が得られる．

また，表2の予測成功率(a)は，オプション(1)の状態ビットによるΔの更新機能を備えたLACに関するものであり，一方(b)はその機能を備えないLACに関するものである．そして，表の最後のコラムは，状態ビットによる予測成功率の向上を表している．1ビットの状態ビットにより，平均2.1%の予測成功率の向上が得られることがわかる．これは，少ないハード量にもかかわらず，十分な効果が得られることを意味する．

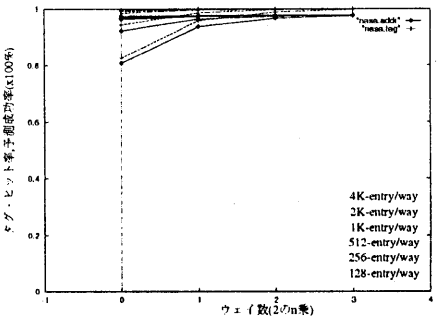

図7 測定結果(nasa7)
Fig. 7 Simulated result(nasa7)

表2 予測成功率(4-way,512-entry LAC)
Table 2 Prediction accuracy with 4-way 512-entry LAC

プログラム	タグ・ヒット率	予測成功率 (a)	(b)	(a) - (b)
gcc	90.5%	53.6%	51.8%	+1.8%
xlisp	99.9%	57.1%	53.2%	+3.9%
espresso	99.0%	68.7%	62.5%	+6.2%
eqntott	99.9%	95.9%	94.7%	+1.2%
compress	99.9%	72.4%	69.2%	+3.2%
sc	97.1%	86.8%	84.2%	+2.6%
fpppp	12.1%	11.6%	11.5%	+0.1%
doduc	76.2%	63.9%	62.8%	+1.1%
wave5	99.6%	91.2%	88.3%	+2.9%
tomcatv	99.9%	99.6%	99.2%	+0.4%
nasa7	99.9%	97.6%	96.4%	+1.2%
hydro2d	99.9%	97.8%	96.8%	+1.0%
平均 w/fp	89.5%	74.7%	72.6%	+2.1%
平均 wo/fp	96.5%	80.4%	78.1%	+2.3%

(平均 w/fp:fppppを含む，平均 wo/fp:fppppを含まない)

6. 考　察

デルタΔ(前回のアドレスと前前回のアドレスの差)の効果について考察する．前述のように，予測アドレスは，前回のアドレスにΔを加えることによって算出される．前章の測定では，Δのビット長はPCと同じ32ビットとしていた．これは，PCが32ビットなので，実質的にΔの値は無制限であったことを意味する．LACのインプリメントの面から最適なΔのビット長について，予測が成功したときのΔの値から調べる．

gcc，eqntott，fppppおよびnasa7について，予測が成功したときのΔのヒストグラムを図8～図11に示す．測定に用いたLACの構成は，4ウェイ,512エントリである，すなわち，表2の予測成功率(a)の構成と同じである．X軸はΔの値を，Y軸は各Δに対する出現回数を表す．gccとeqntottについては，Δ:ゼロを中心に対称であることがわかる．nasa7に関しては，Δが広い範囲で特定の値をもつことがわかる．いずれのプログラムでも，Δは，ゼロまたは絶対値が非常に小さい範囲に集中しているといえる．それでは，Δのビット長を

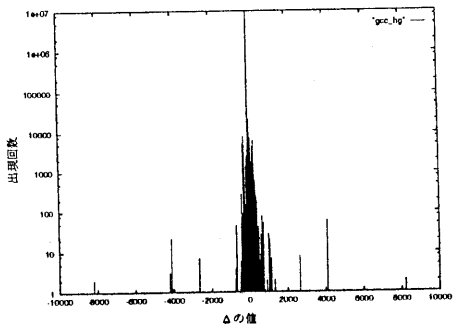

図8 Δのヒストグラム (gcc)
Fig. 8 Histgram of delta (gcc)

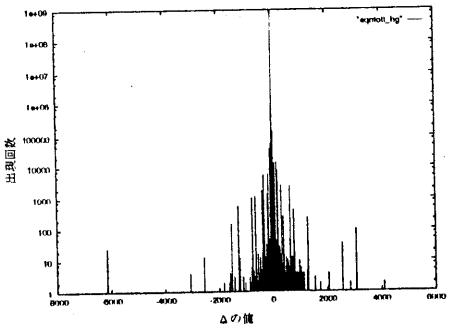

図9 Δのヒストグラム (eqntott)
Fig. 9 Histgram of delta (eqntott)

変化させたとき，予測成功率がどのように変化するかについて調べる．

図12と**図13**は，Δのビット長に関して，32ビットΔに対する予測成功率のレシオを表し，それぞれ，SPECint92とSPECfp92の各6つのプログラムについての測定結果である．X軸は，Δの長さを表している．Δ='1'は，Δの値が常にゼロである，すなわち，"予測アドレス＝前回のアドレス"である場合である．例えば，Δ='6'は，Δが"-32 ⟨ Δ ⟨ +32"の範囲であることを表す．この場合，もし，Δが'-32'以下あるいは'+32'以上であれば，LACのΔ部には，ゼロをライトする．Y軸は，Δが32ビットのときの予測成功率を100としたときの各Δに対する予測成功率を表す．

これらの図より，例えば，scは，Δが高々4ビットで100%になる，つまり，Δのビット長は4ビットで十分であることがわかる．一方，compressでは，16ビットの場合でも95%であり100%に達しない．また，nasa7は，図11から推測されるとおり，ビット長が小さいところでは成功率が低く，ビット長が大きくなるにつれて成功率も高くなり，15ビットになって，100%に達していることがわかる．nasa7を除いては，Δのビット長が7ビットであれば，十分な成功率が得られると推測することができる．

4ウェイ,512エントリ,Δの長さ:7ビットのLACについて，予測成功率を測定した結果を**表3**に示す．表2の結果に対して，6.3%の予測成功率の低下であることがわかる．

以上より，Δのビット長が高々7ビット，すなわち絶対値が64バイト未満で，70%以上の予測成功率が得られることがわかった．

7. ま と め

高速プロセサのIPC向上を目的とし，投機的なメモリ・システム・アクセスと後続の命令の投機的実行を行なうために，ロード・アドレス予測方法の検討及び評価を行なった．

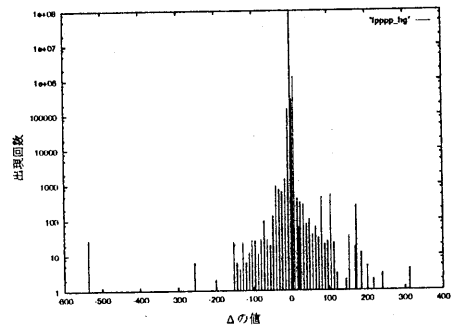

図10 Δのヒストグラム (fpppp)
Fig. 10 Histgram of delta (fpppp)

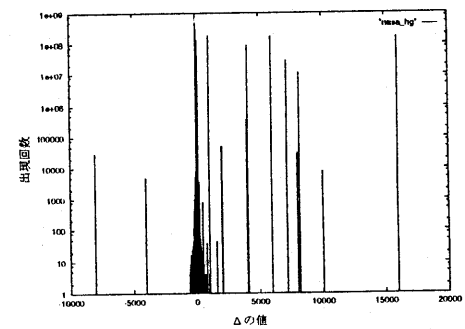

図11 ヒストグラム (nasa7)
Fig. 11 histgram of delta (nasa7)

この方法の評価のために作成したシミュレータを用いて，SPECベンチマークを対象として，測定を行なった．

シミュレーションにより，4ウェイ,512エントリ,アドレス変位Δの長さ:7ビット,状態ビット:1ビット構成のロード・アドレス・キャッシュ(LAC)の予測成功率は，平均70%以上であることがわかった．

以上より，本ロード・アドレス予測方法によれば，高精度にロード命令のメモリ・アドレスを予測できること

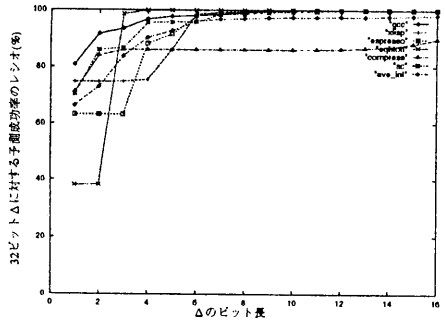

図12 Δのビット長の効果 (SPECint92)
Fig. 12 Effect for bit-length of delta (SPECint92)

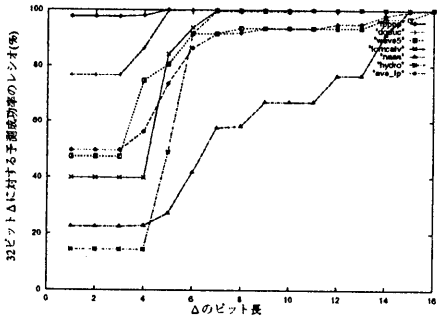

図13 Δのビット長の効果 (SPECfp92)
Fig. 13 Effect for bit-length of delta (SPECfp92)

表3 予測成功率 (4-way,512-entry,delta:7-bit LAC)
Table 3 Prediction accuracy with 4-way 512-entry delta:7-bit LAC

プログラム	タグ・ヒット率 (a)	予測成功率 (b)	(b)/(a)
gcc	90.5%	54.6%	60.3%
xlisp	99.9%	58.2%	58.2%
espresso	99.0%	68.2%	68.9%
eqntott	99.9%	96.0%	96.0%
compress	99.9%	63.3%	63.3%
sc	97.1%	86.0%	88.5%
fpppp	12.1%	11.6%	96.0%
doduc	76.2%	65.0%	85.3%
wave5	99.6%	84.3%	84.7%
tomcatv	99.9%	99.5%	99.5%
nasa7	99.9%	56.3%	56.3%
hydro2d	99.9%	97.4%	97.5%
平均 w/fp	89.5%	70.0%	78.2%
平均 wo/fp	96.5%	75.3%	78.0%

(平均 w/fp:fpppp を含む, 平均 wo/fp:fpppp を含まない)

がわかった.

今後,

- より高精度な予測方法.
- 本方法をスーパスカラ・プロセサに搭載したときのIPC の向上度.
- 予測が失敗したときのリカバリ機構.
- 物理アドレスを格納する LAC.

などについて検討を進めていく予定である.

謝辞 本研究を遂行するにあたり御指導御支援して下さった, 林プロジェクト部長, 津田プロジェクト部長代理に感謝します. また, 討論して頂いたプロセサ研究部の皆様に感謝します.

参 考 文 献

1) Bannon, P., et al., "Internal Architecture of Alpha 21164 Microprocessor," *COMPCON'95 Proceedings*, pp.79–87, March 1995.
2) Hunt, D., "Advanced Performance Features of the 64-bit PA-8000," *COMPCON'95 Proceedings*, pp.123–128, March 1995.
3) Patkar, N., et al., "Microarchitecture of HaL's CPU," *COMPCON'95 Proceedings*, pp.259–266, March 1995.
4) Levitan, D., et al., "The PowerPC 620 Microprocessor: A High Performance Superscalar RISC Microprocessor," *COMPCON'95 Proceedings*, pp.285–291, March 1995.
5) Greenley, D., et al., "UltraSPARC: The Next Generation Superscalar 64-bit SPARC," *COMPCON'95 Proceedings*, pp.442–451, March 1995.
6) Eickemeyer, R. J., et al., "A load-instruction unit for pipelined processors," *IBM J. RES. DEVELOP.*, pp.547–564, Vol. 37 No. 4 July 1993.
7) SPARC International, Inc., "The SPARC Architecture Manual Version 8," 1992.
8) Hsu, P. Y., "Introduction to SHADOW," July 1989.