計算機アーキテクチャ　96－2
（1992. 10. 22）

マイクロプロセッサＭ３２／１００の評価
－ｘｂｅｎｃｈにおける内蔵キャッシュの効果－

上田　達也　　近藤　弘郁　　吉田　豊彦　　三宅　俊光

三菱電機株式会社　ＬＳＩ研究所

〒664　兵庫県伊丹市瑞原４丁目１番地

あらまし　　３２ビットマイクロプロセッサM32/100の内蔵命令キャッシュの効果を評価した。命令キャッシュのサイズは２５６バイトである。評価プログラムとして、xbench、stanford、Dhrystoneベンチマークを用いた。xbenchを実行する時、M32/100が実行するビットマップディスプレイ制御プログラム（Ｘサーバ）のサイズは５００Ｋバイトである。M32/100内蔵キャッシュは小容量であるにもかかわらず、xbenchにおいて２０％の性能向上をもたらす。M32/100内蔵キャッシュは、サイズの小さなプログラムのみならず、ビットマップディスプレイ制御プログラム等の実アプリケーションにおいても性能向上をもたらす。

和文キーワード　　マイクロプロセッサ、キャッシュ、トロン、xbench、Ｘウィンドウ

Evaluation of the microprocessor M32/100
- Boosted performance by on-chip cache in "xbench" program -

Tatsuya Ueda　Hiroyuki Kond　Toyohiko Yoshida　Toshimitsu Miyake

LSI Research and Development Laboratory,Mitsubishi Electric Corporation

4-1 Mizuhara, Itami, Hyogo 664, Japan

Abstract

We evaluated the effect of the on-chip cache of the 32-bit microprocessor M32/100. The on-chip cache is a 256-byte direct map instruction cache. The benchmark programs we used are xbench, Stanford, and Dhrystone benchmarks. The M32/100 executes a 500-Kbyte long X-server program to control a bit-map display when it executes the xbench program. Its on-chip cache is very small but can boost the performance of the xbench program about 20 percent. The on-chip cache is useful not only for small programs but also for real applications such as the X-window program.

英文 key words　　microprocessor　cache　TRON　xbench　X-window

１．はじめに

近年、微細化技術の進歩によりマイクロプロセッサの大規模化、高速化は飛躍的に進んでいる。しかし、メモリに関しては、大容量化は進んでいるものの、アクセスタイムはあまり向上しておらず、マイクロプロセッサのメモリアクセス時間に追従できていない。このため、メモリアクセスがマイクロプロセッサの性能向上を妨げる大きな原因となっている。

メモリアクセスによる性能低下を軽減するため、最近の高性能マイクロプロセッサの多くが内蔵キャッシュを備えている。また、そのサイズも増加する傾向にあり、１６Ｋバイト以上を内蔵するマイクロプロセッサも現れている。

ＴＲＯＮ仕様に基づく３２ビットマイクロプロセッサM32/100[1][2][3]も他の高性能マイクロプロセッサと同様に、内蔵キャッシュを備えている。M32/100の内蔵キャッシュは命令キャッシュであり、サイズは２５６バイトと小容量である。

本報告では、M32/100の内蔵キャッシュがビットマップ・ディスプレイ制御プログラムでは有効であることを、Ｘウィンドウ評価用ベンチマークであるxbenchを用いて評価した結果について述べる。

２．Ｍ３２／１００の特長

M32/100は、機器組込み制御やメモリ管理を必要としない小型ワークステーション、ビットマップディスプレイ制御などをターゲットとしたマイクロプロセッサである。M32/100は高性能化ため、５段のパイプライン制御、２５６バイトの命令キャッシュ、先行ジャンプ処理、ビットマップ処理命令の機能強化など、各種高速化なための機能が備えられている。一方、価格を抑えるため、チップサイズは小さく、ＡＳＳＰ展開におけるＣＰＵコアとしても利用されている。

以下に、M32/100の特長について述べる。

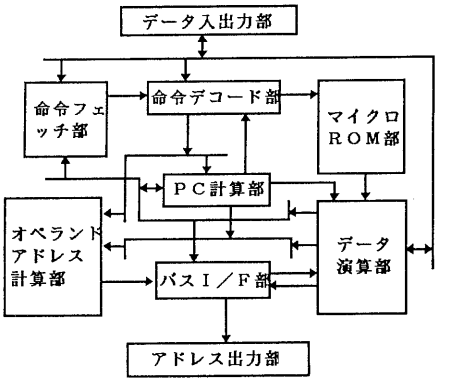

図1　Ｍ３２／１００の機能ブロック

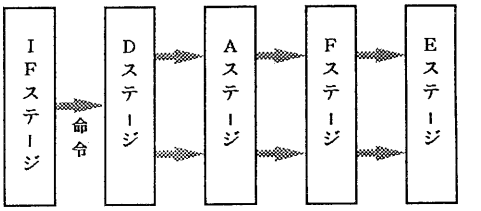

図2　Ｍ３２／１００のパイプライン処理

２．１　Ｍ３２／１００の内部構造

図１にM32/100の機能ブロックを示す。命令フェッチ部は、１６バイトの命令プリフェッチキューと２５６バイトの命令キャッシュを持つ。命令デコード部は、命令フェッチ部から取り込んだ命令を２バイト単位で処理する。ＰＣ計算部は、デコードされた命令のＰＣ値を計算する。オペランド・アドレス計算部は、３入力加算器を備え、オペランド・アドレスの計算を行う。マイクロＲＯＭ部は、１１１ビット×１．４Ｋワードのマイクロ ＲＯＭを持ち、マイクロシーケンサに従ってデータ演算部に制御信号を出力する。データ演算部は、レジスタファイル、ＡＬＵ、バレルシフタ、プライオリティ・エンコーダなどから構成され、各種演算を行う。外部バスＩ／Ｆ部は、各ブロックからの外部バスアクセス要求の調停および外部バスアクセスの制御を行う。

M32/100は５段のパイプライン処理を行っている[4]。図２にM32/100のパイプライン構成を示す。パイプラインは、命令フェッチを行うＩＦステージ、命令デコードの前半の処理を行う

Dステージ、オペランド・アドレスの計算および命令デコードの後半の処理を行うAステージ、オペランドのプリフェッチとマイクロROMのアクセスを行うFステージ、命令の実行を行うEステージで構成されている。各パイプラインステージは、1つの命令を最小2クロックで処理する。

2．2　内蔵命令キャッシュの構成

M32/100は、チップサイズを抑え、かつ高性能なプロセッサを目標として設計された。そこで、内蔵キャッシュとして256バイトの命令キャッシュを採用した。内蔵キャッシュを命令用に限定したのは、オペランド・データに比べ命令コードのほうがローカリティがあり、小容量でも効果があると判断したからである。M32/100において、命令キャッシュがチップ全体に占める面積の割合は3%程度である。

図3にM32/100内蔵命令キャッシュの構成を示す。命令キャッシュの1エントリは24ビットのアドレスタグ・フィールド、32ビットの命令コード・フィールド、2ビットのリング番号フィールド、1ビットの有効ビット・フィールドよりなり、64エントリのダイレクト・マップ方式である。

命令キャッシュの参照は以下の手順で行う。命令アドレスのインデックス・フィールド（6ビット）で命令キャッシュの1エントリが選択され、選択されたエントリの各フィールドの値が出力される。出力された有効ビット・フィールドの値が有効であることを示しており、命令アドレスのタグ・フィールド（24ビット）およびPSW中のリング値が命令キャッシュからのアドレスタグ値およびリング値と一致する場合、キャッシュ・ヒットとなる。命令キャッシュがヒットした場合、命令コードが命令プリフェッチ・キューへ送られる。命令キャッシュのアクセスは1クロックで行われる。命令キャッシュがヒットし続ければ、命令コードが1クロックに32ビットずつパイプライン中へ送り込まれることになり、命令フェッチがパイプラインのボトルネックになることはない。

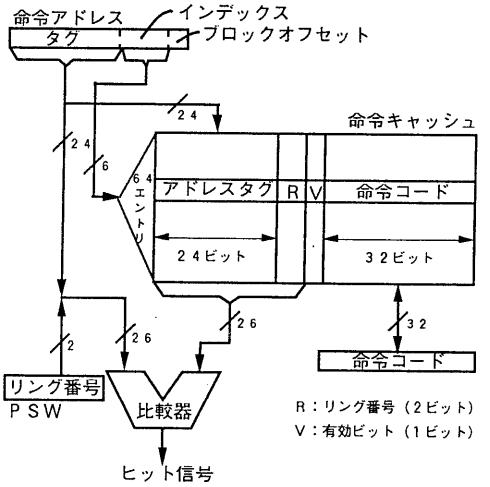

図3　M32／100内蔵キャッシュの構成

命令キャッシュの参照がミスした場合、メインメモリへ命令コードのアクセスが行われる。メインメモリより読み込まれた命令コードは、命令アドレスのインデックス・フィールドが示すエントリに登録される。

2．3　先行ジャンプ処理と動的分岐予測

パイプラインのEステージでジャンプ命令が実行されると、パイプライン中の後続の命令は全てキャンセルされ、ジャンプ先の命令が新たにフェッチされる。ジャンプ命令はパイプライン処理の効率を下げる大きな原因となっている。そこで、M32/100では先行ジャンプ処理機構を導入した[5]。先行ジャンプ処理機構は、Dステージにおいてジャンプ先アドレスを計算し、後続の命令列をジャンプ先の命令列に切り換える機構である。M32/100において先行ジャンプ処理が可能な命令は、ジャンプ先アドレスがPC相対アドレッシング・モードのジャンプ命令に限られる。

条件分岐命令であるBcc命令が、Dステージで先行ジャンプ処理されるかどうかを判断するため、M32/100では動的分岐予測機構を採用した。M32/100の動的分岐予測機構は、1ビット×256エントリのRAMで構成された分岐

予測テーブルを備えている。分岐予測テーブルは、分岐命令アドレスの最下位１ビットを除く下位８ビットでアクセスされ、１ビットの分岐予測を出力する。分岐予測テーブルの各エントリは、対応するアドレスの条件分岐命令の過去１回の分岐履歴を記憶している。動的分岐予測機構は、分岐予測テーブルの分岐履歴に基づいてＢｃｃ命令の分岐予測をする。予測が外れた場合は、Ｅステージで正しいシーケンスに戻す処理を行う。

２．４　ビットマップ処理命令

M32/100では、ビットマップディスプレイ制御の応用に向くビットマップ処理命令の機能を強化した。ビットマップ処理命令は、任意長のビット列操作を１命令で行う。M32/100では、ビット列の処理方向を指定するオプションとして、順方向と逆方向の両方を備えている。

また、マイクロプログラム制御により、ビット列のシフトや演算、連結などの内部処理とビット列のメモリアクセスを並列に実行し、高速な処理を実現している[6]。

３．　命令キャッシュ評価用プログラムおよびその評価環境

M32/100の応用の１つとして、ビットマップディスプレイ処理システムがある。xbenchは、ビットマップディスプレイ処理システムの１つであるＸウィンドウを評価するためのベンチマーク・プログラムである。本報告では、命令キャッシュの効果を評価するのにxbenchを用いた。

また、比較のために、stanfordベンチマークやDhrystoneベンチマークなどのプロセッサの性能評価用プログラムも用いた。

３．１　Ｘウィンドウ

Ｘウィンドウはマサチューセッツ工科大学で開発されたビットマップ・ディスプレイ用のウィンドウ・システムである[7]。図４に示すように、Ｘウィンドウはクライアント・サーバ型のウィンドウ・システムである。クライアント・プロセスは、アプリケーション・プログラムの実行を行う。サーバ・プロセス（Ｘサーバ）はディスプレイへの描画制御、キーボード、マウスからの入力制御を行う。Ｘサーバの命令コードサイズは５００Ｋバイト程度である[8]。クライアント・プロセスとサーバ・プロセスはＸプロトコルと呼ばれる通信プロトコルに従って通信を行い、処理を進める。Ｘプロトコルは、通信に用いるネットワークの種類までは規定していない。Ｘプロトコルには次の４種類がある。

・リクエスト：クライアントからサーバへ送られる描画要求メッセージ。
・リプライ：クライアントの問い合わせに対し、サーバが返す応答メッセージ。
・イベント：サーバ側で発生したイベント（キーボードやマウス入力）をクライアントに伝えるメッセージ。
・エラー：クライアントの要求に対して、サーバ側でエラーが発見されたことをクライアントに伝えるメッセージ。

３．２　ｘｂｅｎｃｈ

Ｘウィンドウの図形、文字描画機能およびウィンドウ操作機能などを評価するためのベンチマーク・プログラムとして、xbenchが広く用いられている[9]。xbenchは、ＸウィンドウにおけるＸサーバの描画能力を測定するものである。xbenchは、Ｘウィンドウのクライアント・プロセスとして動作し、Ｘサーバに対して様々な描画のリクエストを出力し、Ｘサーバがこのリクエストを処理する速度を計測する。xbenchにはディスプレイとしてモノクロを使用するものと、カラーを使用するものの２種類がある。xbenchでは、６種類のテスト項目ごとの性能値（項目名に_stoneを付けてline_stone、fill_stoneとよぶ）と、さらに、各項目に重みを付けて平均をとったXstone値が得られる。Xstone値はSun3/50の性能を１万とした時の相対的性能である。

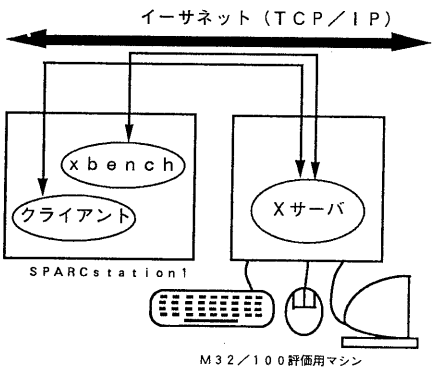

図4　ｘｂｅｎｃｈ測定用ハードウェア構成

以下に、xbenchに含まれる各種テスト項目およびXstone値の算出法を示す。

・line：縦、横、斜めの直線描画
・fill：矩形領域の塗りつぶし
・blt：矩形領域のコピー
・text：文字列処理
・arc：円弧および円弧の塗りつぶし
・complex：ウィンドウの作成、消去

Xstone値 =10000/(A+B+C+D+E+F)
A=2050/line_stone
B=1900/fill_stone
C= 400/arc_stone
D=1850/blt_stone
E=3000/text_stone
F= 800/compelx_stone

３．３　ｘｂｅｎｃｈ測定用システム

図４に、今回xbenchの測定に用いたシステムの構成を示す。SPARCstation1上でxbenchを走らせ、M32/100評価用マシン[8]上でＸサーバを走らせる。両プロセスはイーサネットを介して通信する。

図５にM32/100評価用マシンのハードウェア構成を示す。M32/100評価用マシンはビットマップディスプレイを備えた、デスクトップ型ワークステーション相当の機能を備えている。１６ＭバイトのDRAMを登載しており、Ｘサーバの命令コードおよびデータにとって十分なサイズである。DRAMは０～４ウエイト、ビデオRAMは１～６ウエイトの設定が可能である。

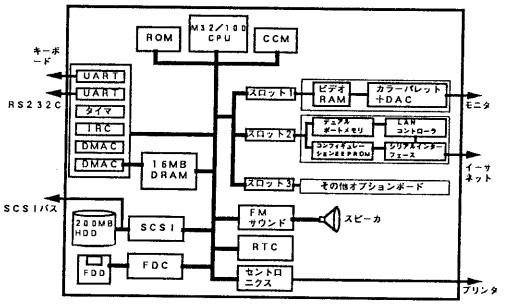

図5　Ｍ３２／１００評価用マシンのハードウェア構成

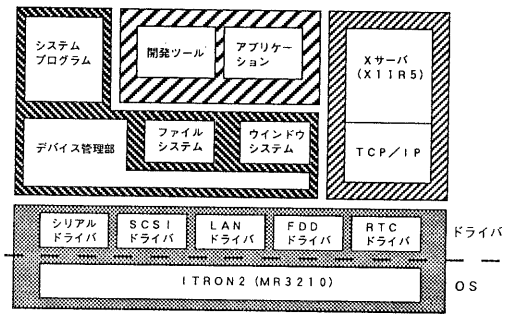

図6　Ｍ３２／１００評価用マシンのソフトウェア構成

図６にM32/100評価用マシンのソフトウェア構成を示す。ＩＴＲＯＮ２仕様のＯＳ（MR3210）をベースとし、その上でＸサーバおよびTCP/IPが動作するように構成されている。Ｘサーバは、Ｘ１１Ｒ５のサンプル・サーバである。

４．　命令キャッシュの評価

xbenchによる命令キャッシュの性能評価を行った。xbenchは第３章で示したように、Ｘウィンドウの性能を評価するためのベンチマーク・プログラムであるが、プログラム・サイズと実行時間が十分大きく、一般のアプリケーションプログラムの１つとも考えられる。

xbenchによる評価結果と比較するため、従来よりプロセッサの性能評価に用いられているstanfordベンチマーク[10]およびDhrystoneベンチマーク[11]による命令キャッシュの性能評価

を行った。stanfordベンチマークは、１０個のプログラムで構成されているが、今回はこのうち、整数演算関連の８個のプログラムを用いた（Perm,Bubble,Queens,Intmm,Quick,Tree,Towers,Puzzle）。以下、Dhrystoneベンチマークおよびstanfordベンチマークの８プログラムを合わせて、小ベンチマークとよぶ。小ベンチマークは、プログラム・サイズが１Ｋバイト前後と小さく、実行時間も１秒以下である。

４．１　キャッシュの評価方法

キャッシュの効果を評価するのに、一般にヒット率がよく用いられている[12]。ヒット率は、キャッシュ・アクセス時にターゲットがキャッシュ中に存在する割合である。

最近の３２ビットマイクロプロセッサでは、パイプライン制御が通常行われている。パイプライン制御のプロセッサでは、命令キャッシュのヒット率向上がプロセッサの性能向上につながらない場合がある。実行に多くのクロック数を要する命令が実行されている場合に、後続の命令がキャッシュ・ヒットしてパイプライン中に取り込まれても、実行ステージの前段ステージで待ち状態となるだけである。また、ジャンプ命令に引き続く命令がキャッシュ・ヒットしてパイプライン中に取り込まれても、先行するジャンプ命令によってキャンセルされてしまう。このような場合は、命令キャッシュがヒットしても性能向上には寄与しない。命令がパイプライン中を順調に流れている状態や分岐直後でパイプラインが空の状態の時に、命令キャッシュがヒットしてはじめて性能向上に寄与する。

パイプライン制御のプロセッサでは、実行結果トレースの解析によるヒット率の測定のみでは、実システムにおいて、どれほどの性能向上が得られるかわからない。本研究報告では、キャッシュの性能評価の方法として、キャッシュによって得られる実システムの性能向上率を用いた。

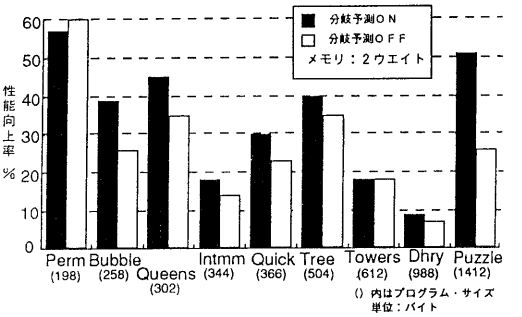

図7　命令キャッシュによる性能向上率（小ベンチマーク）

```
Label1:
        mov @(r4),r0
        mul @(r5),r0
        add #1,r6
        add #4,r4
        add #164,r5
        add r0,@(r7)
        cmp #40,r6
        ble Label1
```

（ａ）Intmm中のループ

```
        bra Label3
Label1:
        cmp #0,@(r5)
        ble Label2
        cmp #0,@(r4)
        ble Label2
        mov #0,r0
        bra Label4
Label2:
        add #4,r4
        add #4,r5
        add #1,r7
Label3:
        cmp r6,r7
        ble Label1
        mov #1,r0
Label4:
```

（ｂ）Puzzle中のループ

図8　IntmmとPuzzle中のループ

４．２　評価結果

＜小ベンチマーク＞

図７に、小ベンチマークを用いた命令キャッシュによる性能向上率を示す。主メモリは２ウエイトである。

各ベンチマークにおいて、分岐予測機構をＯＮした場合とＯＦＦした場合における命令キャッシュによる性能向上率を示してある。ほとんどのベンチマークにおいて、分岐予測機構をＯＮした場合のほうが性能向上率は高い。

プログラムにより命令キャッシュの効果は大きく異なる。効果が大きいものは６０％程度あり、小さいものは１０％程度である。図において、各プログラム名の下にそのプログラムの命令コードサイズが示してある。Permのように、プログラム全体が命令キャッシュに納まってしまうものは、当然ながら命令キャッシュの効果は大きい。プログラムサイズが大きくなると命令キャッシュの効果は小さくなる傾向にある。しかし、Intmmは命令コードサイズの割に命令キャッシュの効果が小さい。また、Puzzleは命令コードサイズの割に命令キャッシュの効果が

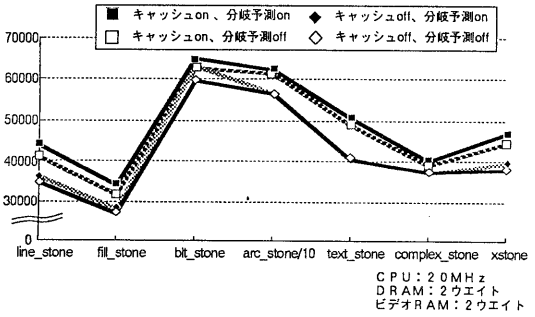

図9　xbench（モノクロ版）によるM32／100の性能評価

大きい。Dhrystoneは命令コードサイズの割に命令キャッシュの効果が小さい。その理由としては、Dhrystoneは、プロシージャコールが頻繁に起こり、各プロシージャは短い処理であるため、プログラムのローカリティがほとんどないためだと考えられる。

IntmmおよびPuzzleは、プログラムをどちらのプログラムもかなり小さなループ部分にほとんどの時間を費やしている。図８に、IntmmおよびPuzzleのループ部分を示す。Intmmのループ中にはmul命令が存在する。M32/100ではmul命令を実行するのに４２クロック必要であり、mul命令以降の命令がキャッシュ・ヒットして素早くパイプライン中に取り込まれても、パイプライン中で先行するmul命令の実行終了を待ち続けるので、命令キャッシュの効果はあまり得られない。

一方、Puzzleのループは、処理時間が２クロックの基本命令のみで構成されており、命令がパイプライン中を滞りなく流れており、命令キャッシュの効果は大きい。

＜ｘｂｅｎｃｈ＞

図９に、xbench（モノクロ版）によるM32/100の性能評価を示す。M32/100評価用マシンにおいて、DRAMは２ウエイト、ビデオRAMは２ウエイトに設定した。動作周波数は２０MHzで測定した。命令キャッシュおよび分岐予測機構をＯＮ、ＯＦＦした4通りについて測定した。グラフの都合上、arc_stoneのみ１０分の１の値にしてある。

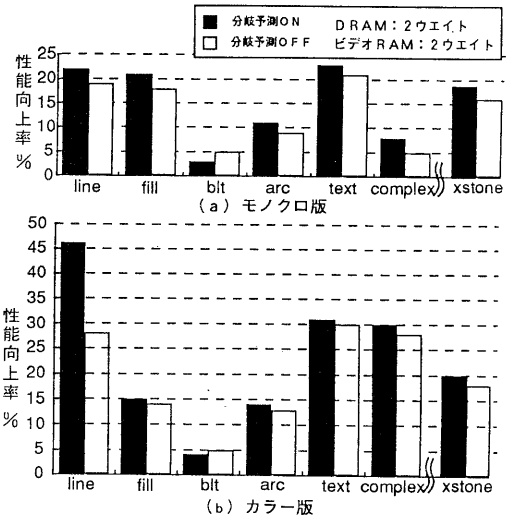

図１０　命令キャッシュによる性能向上率（ｘｂｅｎｃｈ）

図１０に、xbenchを用いた命令キャッシュによる性能向上率を示す。(a)はモノクロ版であり、(b)はカラー版である。

ほとんどの項目で、分岐予測機構をＯＮした場合のほうが性能向上率は高い。line（カラー版）の分岐予測機構ＯＮの場合が、特に性能向上率が高い。

テスト項目によりキャッシュの効果が大きいものと、あまり効果がないものとがあるが、効果の大きいテスト項目では４５％程度の性能向上があり、平均でも２０％以上の性能向上がある。全般的にみて、カラー版のほうがモノクロ版よりもキャッシュの効果が若干大きい。これは、カラー版の描画用プリミティブ・ルーチンが扱うデータ量がモノクロ版の８倍あり、描画処理のループ回数が増え、プログラムのローカリティが増すためだと考えられる。

bltにおいてキャッシュの効果が小さいのは、bltではビットマップ命令が使われており、ビットマップ命令のオペランド・アクセスが処理時間の多くを占めているためだと考えられる。ビットマップ命令は、単純なmov命令をループで繰り返すように構成されていたビット列の転送を、１命令で実行できるようにしたものである。ビットマップ命令を用いることによって命

令キャッシュの効果は小さくなるが、プログラム全体の処理速度は向上する。

xbenchと小ベンチマークによる命令キャッシュの性能向上率とを比較してみると、line（カラー版）は４５％の性能向上率で、Permに匹敵するほど向上していることがわかる。また、text（カラー版）も３０％と小ベンチマークにおける性能向上率の平均値以上を示している。

４．３　結論

xbenchはプログラムのローカリティが高く、小容量の命令キャッシュでも十分効果がある。xbenchにおいては、メモリが２ウエイトの実システムで平均して２０％程度の性能向上率がある。また、全般的に、分岐予測を行った場合のほうが、命令キャッシュの効果がある。このように、一般のアプリケーション・プログラムにおいても、ローカリティの高いプログラムでは小容量の命令キャッシュでも十分効果があことが確認できた。また、その効果は、stanfordベンチマークなどの、命令キャッシュの効果を評価するにはプログラム・サイズが小さすぎると言われているプログラムと同程度のものもある。

命令キャッシュの効果は、プログラム・サイズよりもプログラムの性質に大きく依存する。命令キャッシュの性能を評価するのに、サイズの小さなプログラムでも各種用いてテストすることは、有効であると思われる。

５．まとめ

M32/100の内蔵キャッシュがビットマップ・ディスプレイ制御プログラムでは有効であることを、Ｘウィンドウ評価用ベンチマークであるxbenchを用いて評価した。xbenchを実行する際、M32/100は５００Ｋバイトのビットマップ・ディスプレイ制御プログラムを実行する。内蔵キャッシュは２５６バイトと小容量であるにもかかわらず、xbenchではメモリが２ウエイトの実システムで２０％の性能向上が得られた。M32/100の命令キャッシュが実システムにおいて有効であることが確認できた。

謝辞

本研究の機会を与えて頂いた三菱電機ＬＳＩ研究所ＬＳＩ設計技術第三部富沢部長に感謝します。本研究に関して多大なる援助を頂いたＬＳＩ設計技術第三部の方々に感謝します。

参考文献

[1]K.Sakamura, "Architecture of the TRON VLSI CPU", IEEE MICRO, 1987.4, pp.17-31

[2]O.Tomisawa,T.Yoshida,Y.Saito,M.Matsuo, T.Shimizu and T.Enomoto, "Design Consideration of the GMICRO/100", TRON Project 1987, Springer-Verlag, p.249-258

[3]T.Shimizu, T.Yoshida, Y.Saito, M.Matsuoand T.Enomoto, "A 32-bit Microprocessor based on the TRON Architecture:Design of GMICRO/100",Proceeding of IEEE Compcon Spring, 1988.3, pp30-33

[4]吉田、斉藤、松尾、清水　「ＴＲＯＮ仕様マイクロプロセッサＧMICRO／１００のパイプライン処理構造」電子情報通信学会集積回路研究会　CPSY87-52, 1988.3.25, p.25-30

[5]松尾、上田、吉田、斉藤　「ＴＲＯＮ仕様３２ビットマイクロプロセッサＭ３２／１００の高速化手法とその性能評価」　電子情報通信学会集積回路研究会　ICD89-161,1989.11.22,p.65-72

[6]岩田ほか、「ＴＲＯＮ仕様３２ビットマイクロプセッサＧMICRO／１００（１）マイクロプログラムによる高機能命令の実現とその評価」　情報処理学会第３８回全国大会論文集、1989.3, p.1518

[7]酒匂　「Ｘ－ｗｉｎｄｏｗの概要」ｂｉｔ 1983.3, p.20-30

[8]平野、斉藤　「X端末へのＭ３２／１００の応用」第12回トロン技術研究会　1992.3.30

[9]今泉、権藤　「ｘｂｅｎｃｈとｘｌｌｐｅｒｆ」UNIX MAGAGINE 1990.10 p.36-56

[10]Walter J.Price,"A Benchmark Tutorial",IEEE MICRO, 1989.10, p.28-43

[11]R.P.Weicker,"Dhrystone:A Syntheti System Programming Benchmark", Communications of the ACM, Vol.27 No.10, 1984, p.1013-1030

[12]J.Hennessy,D.Patterson,"Computer Architecture A Quantitative Approach", 1990, p.403-425