6L−2

マイクロ命令を用いたVLIW型計算機と新しいソフトウェアパイプライン技術の有効性の検討

斎藤正孝*、安倍正人**、塚本龍男*、根元義章**
(*東北学院大学教養学部、**東北大学大型計算機センター)

1 .まえがき

我々は、256bit/Wという長大な命令長を持つプログラムメモリで15のユニットを1ステップで同時に動作させることによって高速化を図るVLIW型計算機KIDOCHⅣ[1]を開発中である。この計算機の特徴は、

(1) 従来の計算機の1命令をより細かい複数ステップのマイクロ命令で表しているため、マイクロ命令のピッチをスーパーパイプライン計算機のピッチと同程度に設計可能である。
(2) プログラムメモリ中には、直接各演算器に動作させるための(デコード済の)マイクロ命令が入っているため、従来の命令にあったデコードサイクルを必要とせず、トータル的に1命令の短縮につなげている。また、従来において、パイプラインの段数が3から8であったものが、KIDOCHⅣにおいては1である。つまり、条件分岐の際のハザードの影響が、1マイクロ命令分だけとなり、従来に比べ、影響が少ない。
(3) 複数の演算器と複数のキャッシュメモリをサポートしている。従来のVLIW計算機では、1ステップで複数の命令を処理することが可能であっても、データポート数の不足により、実際に処理しているデータの数が限られてしまうというデメリットがあった。KIDOCHⅣにおいては、キャッシュメモリを複数採用することで、データポート数の増加につながり、実際同時に扱えるデータ数を増やすことを可能としている。

以上の3点があげられる。

本文ではこれらの特徴をもつことにより、モジュロソフトウェアパイプライン[2]と呼ばれる最適化技術の効果をより一層上げられることをいくつかのベンチマークプログラムにより検証する。

2 マイクロ命令

従来の計算機の1命令を、KIDOCHⅣでは図1に示すように複数のマイクロ命令で実現している。このステップ数が従来の計算機(パイプライン度3と仮定)における命令の実行サイクルの長さとなるので、KIDOCHⅣのマイクロ命令のピッチは、従来の計算機の実効サイクルのピッチの1/3程度になると期待できる。また、KIDOCHⅣにおいては、プログラムメモリの内容そのものにはデコードされたコードが入っていると見なすことが可能で、それが各ユニットを直接動作させるため、命令のデコードサイクルが必要なく、レイテンシを小さくできるというメリットもある。

3 モジュロソフトウェアパイプライン技術[2]

図2に示すようなループを含むプログラムの場合、KIDOCHⅣでは図3(a)に示すように、1つのループをN'(N'=CK　C,K:整数)になるように条件分岐の位置を調節し、演算器が使われているかどうかを表す使用表を作る。次に図3(b)のようにKだけずらし、はみ出した部分を後ろに図3(c)のように重ね合わせる。こうして得られた使用表をもとの使用表図3(a)と重ね合わせた部分のうち、演算器が実際に使われている部分が重なるかどうかを調べる。もし重なったならば、重なった部分のマイクロコード以降を1ステップずらし、演算器の使用が重ならないようにする。

次に、図3(d),図3(e)のように2Kずらし、はみ出した2Kの部分を図3(f)のように重ね合わせ、さらに、図3(g)のようにさらにはみだしているKの部分を重ね合わせる。こうして得られた使用表のうち、演算器が実際に重ならないかどうかを調べる。

これをC回繰り返すと図3(h)に示すように最終コードのループの先頭は(C-1)Kずれた位置となり、1データ当たりのステップ数はKとなる。

4.ベンチマークプログラムによる性能評価

リバモアループベンチマークプログラムにより、性能評価を行った結果を表1に示す。SPARC(最適化オプション)により生成されたアセンブラ出力のステップ数を基にして、種々の並列条件の際、ステップ数がどのように減少していくのかを調査した。①~⑧は下記の各並列条件を示す。

①SPARC(-S　-Oオプション　最適化)
②32ビット演算
③整数、浮動少数点系演算の並列化
④整数、浮動少数点系演算、アドレス計算の並列化
⑤整数、浮動少数点系演算、アドレス計算、JUMP(分岐)命令の並列化
⑥整数、浮動少数点系演算、アドレス計算、JUMP(分岐)命令、ロード/ストア系命令の並列化
⑦KIDOCHⅣ(最適化なし)
⑧KIDOCHⅣ(最適化あり)

SPARCにおいて-S　-Oオプションを用いて生成されたアセンブラ出力とKIDOCHⅣのアセンブラ出力をステップ数、ならびに、クロック長という観点より、仮定した並列条件において比較した。表1においては、ループ3について、6種の並列条件について比較し、表2においては、仮定した中で一番並列度の高い⑥とKIDOCHⅣについてステップ数(a)、ならびに、クロック長(b)を9種のベンチマークプログラムについて比較した。但し、KIDOCHⅣのクロック長は各種アーキテクチャ計算機の1/3と仮定した。

提案したアーキテクチャとモジュロソフトウェアパイプライン技術という最適化技術により、KIDOCHⅣの最適化後において一番並列度の高い⑥のアーキテクチャと比べてみても、ステップ数で約2倍、実効サイクルで約4倍から最高約20倍の高速化が図れることが分かった。

5.結論

KIDOCHⅣにより高速処理が可能であることが、この性能比較により確認できた。

Performance Evaluation of a VLIW computer using micro instructions and a new software-pipeline technique
M.Saito*,M.abe**,T.Tsukamoto*,Y.Nemoto**
*Tohoku Gakuin University,**Tohoku University

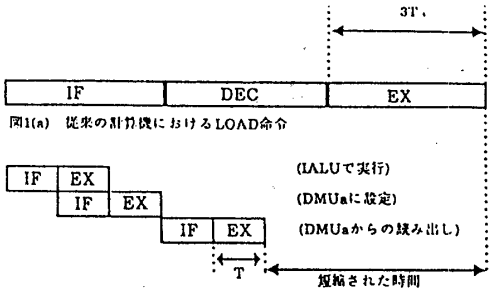


図1(a) 従来の計算機におけるLOAD命令

図1(b) KIDOCHにおける複合命令としてのLOAD命令

(図1 マイクロ命令と従来の命令の比較)

```
/****** BENCH3 ******/
float  z[M],x[M],q;
main()
{
      int i,k;
       for(k=0 ; k<128 ; k++){
                q += z[k] * x[k];
        }
}
```

図2 性能評価のためのサンプルプログラム

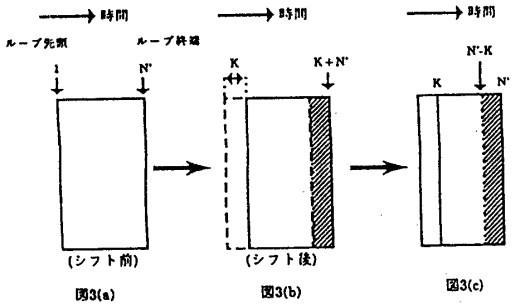


図3(a) 図3(b) 図3(c)

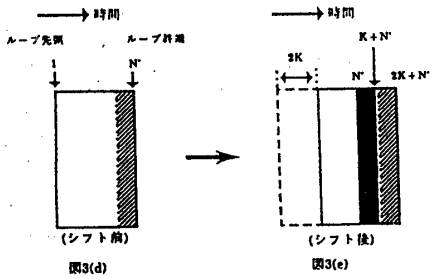


図3(d) 図3(e)

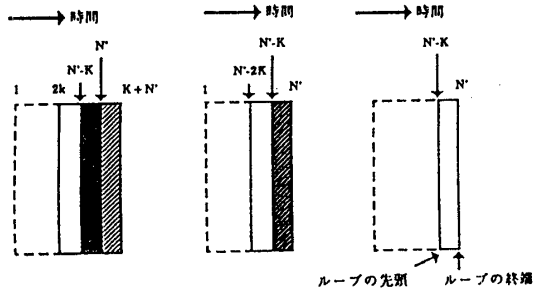
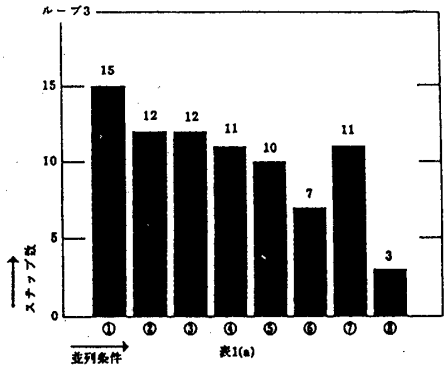


図3(f) 図3(g) 図3(h) 直時コード



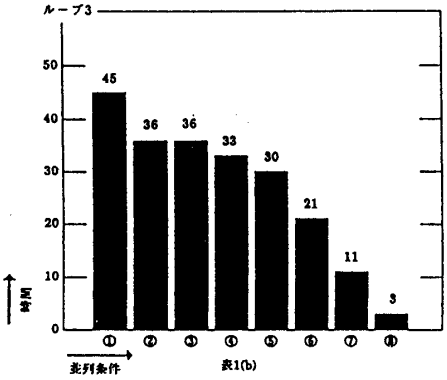


表1 リバモアループベンチマークプログラム ループ3に要する各方式の演算器の性能

	⑥	KIDOCH (最適化なし)	KIDOCH (最適化あり)
ループ1	10 (30T)	21 (21T)	5 (5T)
ループ2	11 (33T)	21 (21T)	8 (8T)
ループ3	7 (21T)	11 (11T)	3 (3T)
ループ11	6 (18T)	11 (11T)	4 (4T)
ループ12	6 (18T)	16 (16T)	4 (4T)
ループ14	15 (45T)	20 (20T)	6 (6T)
ループ19	6 (18T)	15 (15T)	3 (3T)
ループ21	19 (57T)	15 (15T)	3 (3T)
ループ24 (T,F)	8、6 (24T,18T)	20,19 (20T,19T)	15,13 (15T,13T)

T:条件が真の時 F:条件が偽の時 単位(STEP)

表2 種々のリバモアループベンチマークプログラムにおいて、並列条件⑥とKIDOCH Vの性能評価

参考文献

[1] 安倍他:5ポートレジスタファイルを用いたVLIW型計算機KIDOCH(1991)

[2] Touzeau,R.F.,"A Fortran Compiler for the FPS-164 Scientific Computer," *Proc. ACM SIGPLAN'84 Symp. Compiler Construction*, pp.48-57,June 1984