計算機アーキテクチャ　97－1
（１９９２．１２．３）

# 超細粒度VLIW計算機ＫＩＤＯＣＨの特徴とＣコンパイラの性能評価

安倍　正人　　岡部　公起　　根元　義章
東北大学大型計算機センター

本文では、従来の計算機における１つの命令を複数のステップからなるマイクロ命令で表すことにより、スーパーパイプライン計算機と同じように命令のピッチ縮小の効果を持たせた上で、更に条件分岐におけるハザードの影響をなくしたＶＬＩＷ計算機ＫＩＤＯＣＨ IV のアーキテクチャについて述べ、続いて開発したＣコンパイラの性能を条件分岐のあるプログラムについて調べた結果を報告する。

# A SUPER FINE GRAIN VLIW COMPUTER KIDOCH AND THE PERFORMANCE OF THE C COMPILER

Masato Abe　　Kouki Okabe
Yoshiaki Nemoto
Computer Center, Tohoku University

This paper describes the architecture of a VLIW computer KIDOCH IV, in which one instruction of a conventional computer is represented by several steps of more fine grade micro instructions. By this architecture, the effect of hazard due to condition jump is less in KIDOCH compared to a conventional computer such as a super pipeline computer. Also described is the performance of the C compiler for the VLIW computer KIDOCH with a sample programs involving JUMP instructions.

## 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、直接各演算器を動作させるための（デコード済みの）マイクロ命令が入っているため、従来の計算機では３～８と多かったパイプラインの段数がＫＩＤＯＣＨでは１である。そのため、スーパーパイプライン計算機で問題となる条件分岐の際のハザードの影響が高々１マイクロ命令分だけとなり、影響が少ない。

（３）複数の演算器と複数のメモリキャッシュをサポートしている。

という３つの点にある。本文では、この点について条件分岐を含むサンプルプログラムを用いて検証すると共に、開発したＣコンパイラの性能を調べた結果についても述べる。

## 2．ＫＩＤＯＣＨの全体構成

図１にＫＩＤＯＣＨIVの全体構成を示す。ホスト計算機はＵＮＩＸマシンである。このホスト計算機とＫＩＤＯＣＨIVはVMEバスを通じて結合される。

ＫＩＤＯＣＨIVは独立に稼働する１５個の装置（２つのアドレス計算用ＡＬＵ、それぞれ６４ｋＷ／３２ｂｉｔの容量の２つのデータメモリキャッシュ、１つの浮動小数点／整数演算用ＡＬＵ、１つの浮動小数点／整数演算用ＡＬＵ兼ＭＰＹ兼ＤＩＶ、ホスト計算機との間のパイプライン的データ交換に用いられるバンク切り替え方式の６４ｋＷ／３２ｂｉｔ×２バンクの容量のメモリユニット、ホスト計算機との間で制御コードの受け渡しに用いられる４ｋＷ／３２ｂｉｔの容量のデュアルポートメモリ等）からなり、トリプルハイトのプリント基板２枚で構成される。

## 3．５ポートレジスタファイルによるバスネックの解消

ＫＩＤＯＣＨIVは、ＫＩＤＯＣＨ III［２］と異なり、データ用の内部バスは２本である。そのため、ＫＩＤＯＣＨ III以上にバスネックが問題となる。５ポートレジスタファイル（６４Ｗ／３２ｂｉｔ）は図２に示すように入力ポートが３、出力ポートが２であるので、データ用内部バスの数を３本にすることも可能ではあったが、いくつかのサンプルプログラムについて調査した結果、３本の入力ポートの内１本をデータの帰還用に用いたほうが有効な場合が多いため、図２の構成となった。

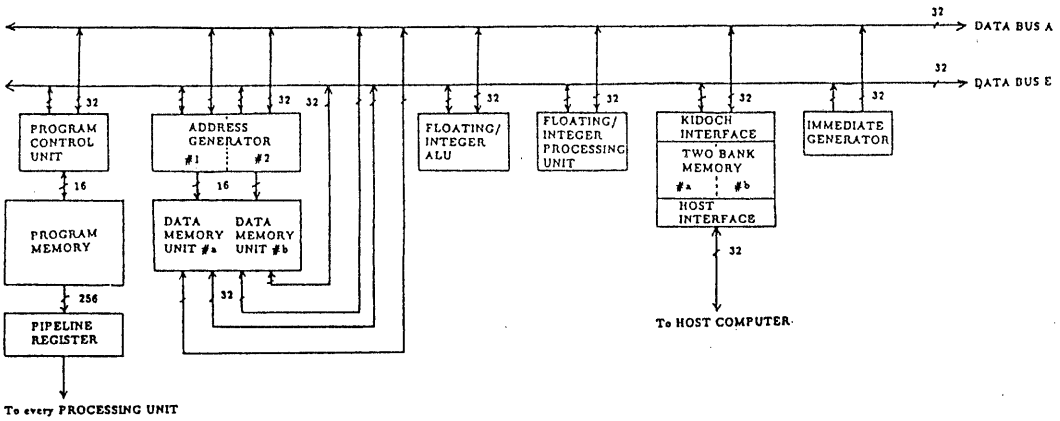


図１　ＫＩＤＯＣＨ IV の全体構成

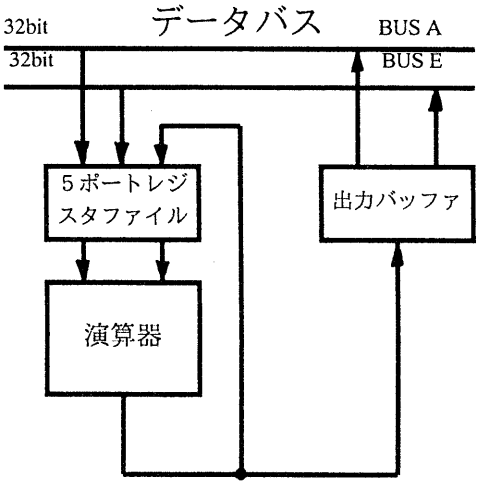


**図２　５ポートレジスタと演算器の構成**

また、レジスタファイルを用いると、コードコンパクションの際の同期の問題が緩和されるという利点と、後で再び同じデータを使うとわかっている場合に、そのデータをレジスタファイルの別の領域に待避しておくことにより、同じデータを再転送しないですむという利点がある。

## ４．　マイクロ命令

従来の計算機の１命令を、ＫＩＤＯＣＨ IV では図３に示すように複数のマイクロ命令で実現している。図中、ロード命令は４ステップ、加算命令は３ステップで実行される。このステップ数が、従来の計算機（パイプラインの段数：３と仮定）における命令の実効サイクルの長さとなるので、ＫＩＤＯＣＨ IV 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１／３～１／

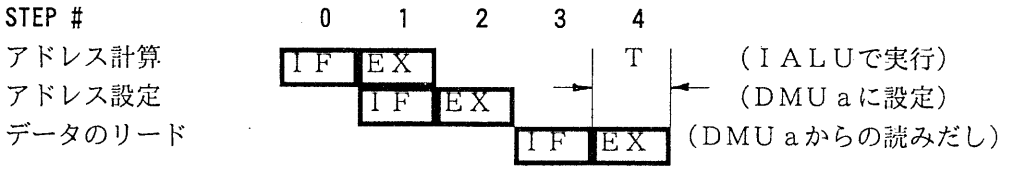


図３ａ　ＫＩＤＯＣＨにおける複合命令としてのＬＯＡＤ命令

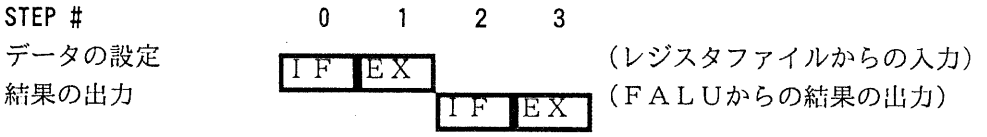


図３ｂ　ＫＩＤＯＣＨにおける複合命令としての加算命令


図３ｃ　従来の計算機におけるＬＯＡＤあるいは加算命令

**図３　従来の計算機とＫＩＤＯＣＨにおける複合マイクロ命令によるＬＯＡＤあるいは加算命令の実行の様子**

ＩＦ　：　命令のフェッチ
ＤＥＣ　：　命令のデコード
ＥＸ　：　命令の実行

４となると期待できる。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做すことが可能で、それが直接各ユニットを動作させるため、命令のデコードサイクルが必要なく、レイテンシは小さい。

## ５．条件分岐の際のマイクロ命令の効果

．図４に示すようなプログラムの場合、ＫＩＤＯＣＨＩＶのアセンブラ出力結果および一般のＲＩＳＣ計算機（ＳＰＡＲＣ　Ｓｔａｔｉｏｎ２）の出力結果は、それぞれ図５（ａ），図５（ｂ）となった。

```
int     sum,a,minus,plus;

main()
{
        sum=minus=plus=0;
        if(a<0) {
                        sum = sum-a;
                        minus++;
        }
        else {
                        sum = sum+a;
                        plus++;
        }
}
```

図４　分岐命令のあるプログラムの例

```
;i=0
_ma;in:
        IMM             XD_3
        ALU1            A0,WENA
        BUSA            IMM
/
;i=1
        IMM             XD_1
        ALU2            A8,WENA
        BUSA            IMM
/
;i=2
        IMM             XD_0
        ALU1            A1,WENA
        ALU2            C8
        ALU2            PASSX
        BUSA            IMM
/
;i=3
        IMM             XD_2
        ALU1            C0
        ALU1            PASSX
        ALU2            A8,WENA
        BUSA            IMM
/
;i=4
        IMM             IC_0
        ALU2            C8
        ALU2            PASSX
        FALU            A6,WENA
        FPU             A2,WENA
        DMU2            AEN
        BUSA            IMM
/
;i=5
        IMM             IC_0
        ALU1            C1
        ALU1            PASSX
        FALU            A7,WENA
        FPU             A3,WENA
        DMU1            AEN
        DMU1            DEN
        BUSA            DMU2
        BUSE            IMM
/
;i=6
        IMM             IC_0
        FALU            C6
        FALU            D7
        FALU            IADD
        FPU             C2
        FPU             D3
        FPU             ISUB
        DMU1            WEN
        DMU2            AEN
        DMU2            DEN
        BUSA            IMM
/
;i=7
        IMM             IC_0
        FPU             C3
        FPU             D2
        FPU             ICOMP
        DMU1            AEN
        DMU1            DEN
        DMU2            WEN
        BUSE            IMM
/
;i=8
        IMM             IC_9
        FALU            A6,WENA
        FALU            B7,WENB
        FPU             A2,WENA
        FPU             B3,WENB
        DMU1            WEN
        BUSA            IMM
/
;i=9
        ALU1            C1
        ALU1            PASSX
        ALU2            C8
        ALU2            PASSX
        COND            FPU_GE
```

```
/
;i=10
        IMM             L$0
        CJP             LATCH_A
        CJP             DELAY
        BUSA            IMM
/
;i=11
        FALU            E6,WENE
        FPU             E3
        DMU1            AEN
        DMU1            DEN
        DMU2            AEN
        DMU2            DEN
        BUSA            FALU
        BUSE            FPU
/
;i=12
        DMU1            WEN
        DMU2            WEN
        COND            TRUE
/
;i=13
        IMM             L$1
        CJP             LATCH_A
        CJP             DELAY
        BUSA            IMM
/
;i=14
/
;i=15
L$0:
        ALU1            C1
        ALU1            PASSX
/
;i=16
        ALU1            C0
        ALU1            PASSX
/
;i=17
        FALU            E7
        DMU1            AEN
        DMU1            DEN
        BUSE            FALU
/
;i=18
        FPU             E2
        DMU1            AEN
        DMU1            DEN
        DMU1            WEN
        BUSE            FPU
/
;i=19
        DMU1            WEN
/
;i=20
L$1:
        IMM             IC_0
        ALU2            A2,WENA
        ALU2            C0
        ALU2            PASSX
        BUSA            IMM
/
;i=21
        COND            TRUE
/
;i=22
        CJP             LATCH_A
        CJP             DELAY
        BUSA            ALU2
/
;i=23
/
```

図５（ａ）　図４のプログラムに対するＫＩＤＯＣＨコンパイラのアセンブラ出力結果

```
_main:
        sethi           %hi(_plus),%o0
        st              %g0,[%o0+%lo(_plus)]
        sethi           %hi(_minus),%o1
        st              %g0,[%o1+%lo(_minus)]
        sethi           %hi(_sum),%o2
        st              %g0,[%o2+%lo(_sum)]
        sethi           %hi(_a),%o3
        ld              [%o3+%lo(_a)],%o3
        tst             %o3
        bge,a           LY1
        sethi           %hi(_a),%g2
        sethi           %hi(_a),%o4
        ld              [%o4+%lo(_a)],%o4
        sethi           %hi(_sum),%o0           ! [internal]
        ld              [%o0+%lo(_sum)],%o5
        sub             %o5,%o4,%o5
        st              %o5,[%o0+%lo(_sum)]
        sethi           %hi(_minus),%o0         ! [internal]
        ld              [%o0+%lo(_minus)],%g1
```

```
        inc     %g1
        b       LE16
        st      %g1,[%o0+%lo(_minus)]
LY1:                                                    ! [internal]
        ld      [%g2+%lo(_a)],%g2
        sethi   %hi(_sum),%o0                   ! [internal]
        ld      [%o0+%lo(_sum)],%g3
        add     %g3,%g2,%g3
        st      %g3,[%o0+%lo(_sum)]
        sethi   %hi(_plus),%o0                  ! [internal]
        ld      [%o0+%lo(_plus)],%g4
        inc     %g4
        st      %g4,[%o0+%lo(_plus)]
LE16:
        retl
        nop                                             ! [internal]
```

**図５（ｂ）　図４のプログラムに対するＳＰＡＲＣコンパイラのアセンブラ出力結果**

また、図４のプログラムを実行したときに、要するクロック数を計算した結果を表１に示す。ここで、比較の対象としては一般のＲＩＳＣ計算機（クロック周期：３Ｔ）、スーパーパイプライン計算機（パイプライン段数：８、クロック周期：Ｔ）、スーパースカラ計算機（同時発行命令数：３、整数演算＆浮動小数点演算＆分岐命令、クロック周期：３Ｔ）及びＫＩＤＯＣＨ ＩＶ（クロック周期：Ｔ）の４種類である。

表１から分かるように、ＫＩＤＯＣＨ ＩＶ は所要時間が一番少ないことが分かる。

## ６．むすび

本文では、従来の計算機における１つの命令を複数のステップからなるマイクロ命令で表すことにより、スーパーパイプライン計算機と同じように命令のピッチ縮小の効果を持たせた上で、更に条件分岐におけるハザードの影響をなくしたＶＬＩＷ計算機ＫＩＤＯＣＨ ＩＶ のアーキテクチャについて述べ、続いて開発したＣコンパイラの性能を条件分岐のあるプログラムについて調べ、その有効性を確認した。

## 文献


［１］安倍　他：ＶＬＩＷ型計算機ＫＩＤＯＣＨのメモリ管理機構、電子情報通信学会技術研究報告ＣＰＳＹ８９－４０（１９８９）

［２］安倍　他：音響ディジタル信号処理を主目的とする高速演算装置μＫＩＤＯＣＨ，情報処理学会論文誌２８、１２、ｐｐ．１３０６－１３１７（１９８７）


| | 一般のＲＩＳＣ計算機 | スーパーパイプライン計算機 | スーパースカラ計算機 | ＫＩＤＯＣＨ ＩＶ |
|---|---|---|---|---|
| ａ＜０のとき | ６６Ｔ | ２８Ｔ | ６３Ｔ | ２１Ｔ |
| ａ＞＝０のとき | ７２Ｔ | ３０Ｔ | ６９Ｔ | １９Ｔ |

**表１　図４のプログラムを実行したときに、各種計算機で必要な総クロック数**