4E-2 C'を用いたJava Chipの設計

菅原 康夫、柳沢 秀明、森 秀樹、上原 稔
東洋大学工学部情報工学科

1. はじめに

Java の普及によりシリコン化 Java(Java Chip) が必要とされているが、Java Chip の設計にはソフトウェアも同時に開発する必要がある。

我々は多種多様な高性能 CPU を短期間で開発することを目的とした設計自動化ツール C'を開発している。C'はハードウェアだけでなくソフトウェアも同時に開発できることを目的とした開発ツールである。

本研究では、Java Chip の設計を通じて C'の有効性や開発環境の作成を含めた改良の方向性を検討していくことを目的としている。

2. C' (C-like Design Automation Shell)

C'[1]は、CPU の開発を目的とした設計自動化システムである。C'ではビット列とニモニックを対応させアセンブラ仕様をC言語に似た文法で記述することで、アセンブラ、逆アセンブラ、シミュレータの自動生成を可能にし、アプリケーションプログラムと同時に CPU を開発することを目指している。

C'ではハードウェアの動作確認をアセンブリ言語で書いたプログラムによって行うことができる

3. picoJava

picoJava[2][3]のコアには命令キャッシュ、整数演算、浮動小数点演算、命令キャッシュ、データキャッシュ、スタックマネージャ、バスインターフェース、といった基本ユニットがある。このうち命令キャッシュとデータキャッシュは0～16Kバイトで自由に変更することができ、浮動小数点演算ユニットも省くことができる。また、Java 仮想マシン(JVM)[4]で使われている命令に加え picoJava 特有の命令（拡張2バイト命令）がある。拡張2バイト命令はユーザーが利用できるものではなく、OS のみが利用できる命令である。

picoJava には JVM 仕様をサポートするハードウェアスタックが実装されており、データ操作はスタックキャッシュを介して行われる。また、コアには Java の実行速度を改善する命令セットがハードウェア内に実装されている。ハードウェアに実装されていない命令でシステムパフォーマンスに重要な影響を与えるものはマイクロコードやステートマシンとして実行され、それ以外は割り込みを実現する命令である。

4. C'による picoJava の設計

メモリスタック、レジスタ等は以下のように定義する。

```
reg frame 32        ;# frame
reg vars 32         ;# vars
reg optop 32        ;#optop(top of operation stack
                          , s[optop+1])
reg   oplim 32            ;# oplim
reg   const_pool 32       # const_pool
reg   psr 32                        ;# psr
reg   pc 32         ;# program counter in m
reg   af 16         ;# access flag
reg   mp 32         ;# method vector
ram m 8 64K ;#main memory(or method memory)
ram s 32 1024                 ;# stack cache
input   traptype   8        ;# input traptype 8
```

Design of Java Chip by using C'

Yasuo Sugawara

Dept. of Information and Computer sciences,Toyo Univ.

直接ハードウェアで実行される命令の例を以下に示す。演算はスタックメモリを用いて行う。

```
instruct iadd {01100000} {
    s[optop+2] = s[optop+2] + s[optop+1];
    optop = optop + 1;
}

instruct isub {01100100} {
    s[optop+2] = s[optop+2] - s[optop+1];
    optop = optop - 1;
}
```

割り込みで実現される命令の例を以下に示す

```
instruct invokevirtual
 {10110110 dddddddeeeeeeee} {

    soft_trap(0xb6);
}
instruct invokespecial
 {10110111 dddddddd eeeeeeee} {

    soft_trap(0xb7);
}
```

ここで割り込み(soft_trap)は以下のように行われる。

1. 要求にしたがってレジスタをスタック上にPushする。
2. 割り込みを禁止にする
3. 特権モードに入る
4. スタック上に保存されたPCの値の場所を指示する為にフレームをアップデートする。
5. 識別したトラップの値(又はsoft_trap引数)をTRAPBASEレジスタの8ビットTT領域に書き込む。
6. トラップハンドラアドレスをトラップベクタアドレスでメモリの場所を読むことで決定し、それはTRAPBASEレジスタの値になる。
7. PCをトラップハンドラアドレスへ初期化し、そのアドレスから実行を続ける

5. 今後の課題

現在のC'版picoJavaではスタックをメモリで実現しており効率が悪いので、今後はキャッシュで実現を行うにする。また、モニターおよび基本ソフト（OSなど）が必要である。今回生成したソフト環境を用いてこれら基本ソフトの実装を行う。また、picoJavaをFPGAまたはASICで論理合成する。

6. まとめ

本研究ではC'を用いてpicoJava相当のCPUを設計した。picoJavaではソフトで割り込み処理を行わなければならないが、C'ではソフトウェアも自動生成するので開発が容易になった。現在の問題点を解決し開発環境を整えればさらに有効なツールになると思われる。

参考文献

[1]柳沢　秀明、森　秀樹、上原　稔「Design Automation Shell」、機能集積情報システム研究会発表論文集　II、電子情報通信学会　、1999

[2] サン・マイクロシステムズ株式会社、「picoJava™ Iマイクロプロセッサのコアアーキテクチャ」
http://www.sun.co.jp/products/wp/picojava_arch/

[3]Sun Microsystems
「picoJava-II Programmer's Reference Manual」
http://www.sun.com/

[4] Jon Meyer,Tory Downing 共著　鷲見　豊　訳/編著「Javaバーチャルマシン」、発行所　オライリー・ジャパン、発売元　オーム社、1997