CPUとハードウェアアクセラレータの実行切替えの高速化

Speeding up Execution Switching between CPU and Hardware Accelerator

伊藤直也[1] 石浦菜岐佐[1] 冨山宏之[2] 神原弘之[3]
Naoya Ito Nagisa Ishiura Hiroyuki Tomiyama Hiroyuki Kanbara

関西学院大学 理工学部[1] School of Science and Technology, Kwansei Gakuin University
立命館大学 理工学部[2] College of Science and Engineering, Ritsumeikan University
京都高度技術研究所[3] Advanced Scientific Technology & Management Research Institute of KYOTO

1 はじめに

ソフトウェアの一部分をハードウェア化することにより, 実行速度や消費電力を改善できる可能性があるが, その際に制御やデータをCPUとハードウェア間でいかに効率的に受け渡すかが課題となる. 戸田らは, CPU密結合型アクセラレータ [1] を提案しているが, 制御とデータの授受にまだ実行サイクルを短縮する余地があった. 本稿では, 1) プログラムカウンタ (PC) の出力ではなく入力の監視, 2) フォワーディングユニット (FWU) を利用したレジスタファイル (RF) への書き込み, 3) アクセラレータによる複数区間連続実行の際の待機の削減により, CPUとアクセラレータの実行をさらに高速に切り替える手法を提案する.

2 CPU密結合型アクセラレータ

戸田 [1] の手法は, 機械語プログラム中の指定区間をハードウェア (アクセラレータ) 化する. アクセラレータはPCを監視し, PCが指定区間の先頭番地に達すると自らが起動する. 実行中はPCの値を固定してCPUにNOPを供給し, 処理を終えると次の番地をPCに書き込むことによりCPUに制御を戻す. アクセラレータはRFおよびメインメモリに直接アクセスしてデータの読み書きを行う他, FWUからもデータを取得する.

3 PCへの入力信号線監視によるアクセラレータ起動

[1] では図1の信号線aを監視していたが, 本手法では信号線bを監視する. これにより, アクセラレータは起動を1サイクル早めることができる. ただし, 5段パイプラインのMIPSの場合, アクセラレータの最初の状態が起動直前の命令のIDステージと重なる (図2) ため, その命令がレジスタを読んでいる場合, アクセラレータはそれと同じレジスタしか読むことができない.

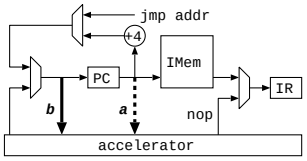

図1 監視場所の変更

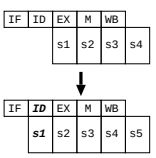

図2 起動の高速化

4 アクセラレータからフォワーディングユニットへの書き込み

[1] では, アクセラレータはRFの読み出しだけにFWUを用いていたが, 本手法ではRFの書き込みにも用いる. 5段パイプラインのMIPSの場合, アクセラレータはRFに書き込むと同時に, 同じデータをWBステージからFWUへ送られてくるデータに上書きする. CPUはアクセラレータの計算結果をFWUから受け取ることにより, [1] よりも1サイクル早く起動できる.

5 アクセラレータの連続実行

従来手法では, ハードウェア化される2つの区間を連続して実行する場合でも, 最初のアクセラレータの処理終了時に一旦PCを監視する状態 (図3の状態0) に戻っていた. 本手法では最初のアクセラレータ (開始番地A1) の処理終了時, PCの値が次の開始番地 (A2) なら, 対応する先頭状態に直接遷移する (図3). これにより, 連続するアクセラレータの実行を1サイクル短縮できる.

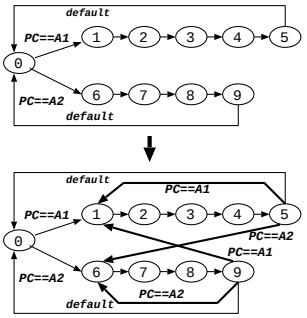

図3 連続するアクセラレータ

```
0: addi $1,$0,10 MIPS
1: addi $2,$0,20
2: add  $3,$1,$2
3: R1<=$1,R2<=$2 ACC1
4: R3<=$3
5: R4<=R1+R2
6: R5<=R3+R4
7: $4<=R5
8: R6<=$3,R7<=$4 ACC2
9: R8<=R6+R7
10: $5<=R8
11: add  $6,$4,$5 MIPS
```

図4 プログラム

6 実験

RTLシミュレーションにより, 図4のプログラムでMIPSと接続したアクセラレータの実行サイクル数を計測した. 図4において, 3–7行目, 8–10行目の演算がそれぞれACC1, ACC2で実行される. 従来手法が実行に17サイクル要したのに対し, 本手法では14サイクルで実行でき, 3サイクル削減することができた.

7 むすび

本稿では, CPUとアクセラレータの処理切替えを高速化する手法を提案した. アクセラレータの自動合成を行う処理系の開発が今後の課題として挙げられる.

謝辞 本研究に関して有益な御助言を頂いた元立命館大学の中谷嵩之氏に感謝致します.

参考文献

[1] 戸田, 石浦: “CPUと密に結合したコプロセッサによるハードウェア/ソフトウェア協調設計,” 情処研報, 2010-EMB-15-16 (Jan. 2010).