5ZB−04

大規模データパスアーキテクチャのコード最適化に関する研究

岩崎 慎介† 服部 直也†† 飯塚 大介††† 坂井 修一†† 田中 英彦††
†東京大学工学部 ††東京大学情報理工学系研究科 †††東京大学工学系研究科

1 始めに

1.1 研究の背景

半導体技術の進歩によりトランジスタの集積度は年々増え続けている。この傾向は今後も続くと予想され、次世代マイクロプロセッサでは現在の数倍のトランジスタが利用可能となる。しかし現在マイクロプロセッサの主流であるスーパースカラは、その構造的複雑性より資源の投入による効果的な性能向上は期待できない。これを受けて、新しいアーキテクチャとして、大規模データパスアーキテクチャ(VLDP:Very Large Data Path)アーキテクチャが提案されている [1]。

1.2 研究の目的

VLDPでは従来にない処理単位として命令ブロックという複数命令の集まりを導入している [2]。命令ブロックはコンパイラにより生成される。この生成手法を工夫することでVLDPの性能向上が期待できる。よって本研究ではコンパイラによる命令ブロックの最適な生成手法を比較、検討することを目的としている。

2 大規模データパスアーキテクチャ

2.1 大規模データパスアーキテクチャの概要

VLDPは将来利用可能であると考えられる豊富な資源を利用し、大規模な投機的実行を行うことにより性能向上を目指す、従来のアーキテクチャの延長上にはない新しいアーキテクチャである。平均実行IPC(Instruction Per Clock cycle) 8の性能を目標として設計が行われている。

VLDPでは命令ブロック(IB:Instruction Block)と呼ばれる最大32の命令からなる命令列を処理単位として導入している。分散した複数の実行ユニット(EU:Execution Unit)を持ち、各々が一つずつIBを処理する。

各EUごとにレジスタを持つ分散レジスタ構成を取っており、同一EU内へのデータアクセスは高速であるが、異なるEUへのデータアクセスは時間を要する。

Code Optimization for Very Large Data Path Architecture
Shinsuke IWASAKI†, Naoya HATTORI††, Daisuke IIZUKA†††, Shuichi SAKAI††, Hidehiko TANAKA††
{iwasaki,hato,iizuka,sakai,tanaka}@mtl.t.u-tokyo.ac.jp
†School of Engineering, The University of Tokyo
††Graduate School of Information Science and Technology, The University of Tokyo
†††Graduate School of Engineering, The University of Tokyo

2.2 命令ブロック

IBの構成を図1に示す。IBは4つのField(最大命令数8のBB)から構成され、1つのFieldは8つの命令スロットから構成される。各Field最後の4つのポイントをBP(Break Point)と呼ぶ。分岐命令はBPにのみ配置でき、分岐の飛び先はIBの先頭のみ指定できる。BP以外の場所からの分岐すること、IBの途中に分岐してくることは許されない。この制限により命令を埋めることのできないスロットにはNOP命令が挿入される。

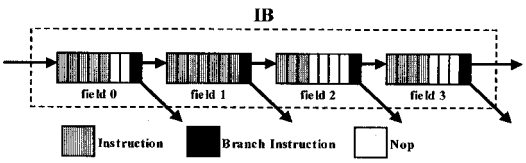

図1: IBの構成

3 命令ブロックの生成

3.1 基本生成法

まず命令数が8を越えるBBは8命令ごとに区切って、Fieldを生成する。この生成されたFieldを4つ結合し、空いた命令スロットにNOP命令を詰めて32命令とすることでIBが生成される。ただしIBの途中へ分岐してくることは許されないので、Fieldが4つ未満でも次のFieldへの分岐があればそこでIBを区切る。また関数呼び出し、RET命令などコードが途切れる命令の直後でも同様にIBを区切る。これにより2.2節で述べたような規則を満たすIBが生成される。

3.2 最適化の方針

異なるIB間のデータ依存があると2.1節で述べた通りEU間のデータ通信が起こり、多くの時間を要してしまう。よってIB間データ依存をできるだけ少なくする必要がある。

しかし2.2節で述べた通り、分岐の飛び先やコードの途切れる命令によりIBが区切られるため、IB内命令数が少なくなり、IB間のデータ依存が多くなってしまう。これによりEU間データ通信が増加してしまい、好ましくない。よってこのようにIBが細かく区切られてしまう原因に対処し、できるだけIB内命令数を大きくして、EU間通信を減らす最適化の実装について検討した。

3.3 profile の利用

あるPCから始まるIBは、図2のように最大8通り考えられる。このうち静的分岐予測を用いてもっとも実行確率の高いものだけを選びその1つだけを生成する。これによってよく実行されるIB内の命令数が大きくなり、EU間通信が削減される。例えば図2で太い矢印の方が実行確率が高い場合、IBは色の付いた部分から構成される。

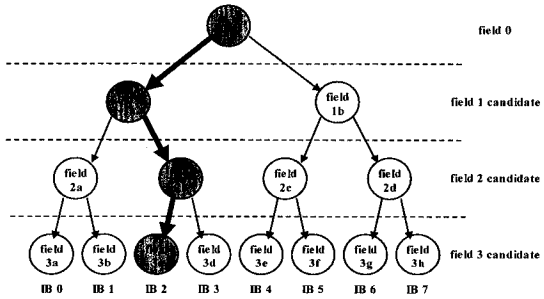

図 2: profile の利用

3.4 合流の除去

図3a)のような4つのFieldからIBを生成する場合を考える。IBの途中へ分岐してくることは許されないので、IBの入り口はその先頭のみである。よって図では入り口が複数あるFiled Dの手前でIBを区切らなければならない。しかし図3b)のようにField Dを複製することによって、IBが途切れるのを防ぎ、EU間データ通信を削減することができる。

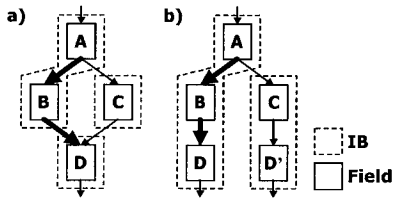

図 3: 合流の除去

3.5 ループの展開

図4a)のような1つのFieldが繰り返し実行されるループは図4b）のようにループを展開する。これによってIB内命令数を増やし、EU間データ通信を削減することができる。

4 評価

最適化Cコンパイラnewcc[3]に3.1節で述べたIBの基本生成法、3.3節、3.4節、3.5節で述べた最適化を実装し、最適化によってEU間データ通信をどの程度削減できるか評価を行った。それぞれの最適化個々の効果と3つ全てを合わせた場合の効果について測定した。ベンチマークとしてSPECint95を用いた。

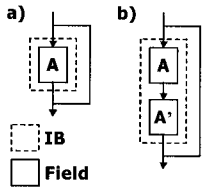

図 4: ループの展開

結果を図5に示す。全ての最適化を合わせた場合において平均5%の削減に成功した。

なお評価においてユニット数、EU間ネットワークバンド幅は無限大とした。またキャッシュは全てあたるものとし、分岐予測は100%ヒットとした。

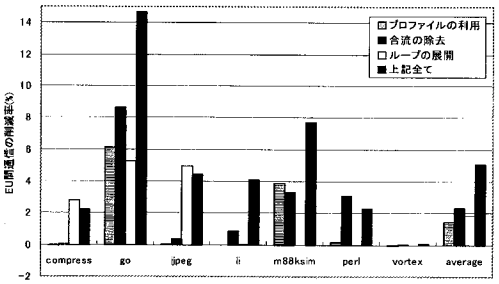

図 5: 最適化によるEU間通信の削減

5 おわりに

本稿では大規模データパスアーキテクチャにおける命令ブロックの最適な生成手法について検討した。今後はIキャッシュを考慮した場合の最適化について研究を行う。

参考文献

[1] 辻秀典、安島雄一郎、坂井修一、田中英彦
大規模データパス・アーキテクチャの提案
情報処理学会研究報告 2000-ARC-139、pp.49-60、2000.

[2] 塚本泰通、安島雄一郎、辻秀典、坂井修一、田中英彦
大規模データパス・アーキテクチャにおける命令ブロックの検討
情報処理学会研究報告 2000-ARC-139、pp.61-66、2000.

[3] 飯塚大介、小沢年弘、坂井修一、田中英彦
Cコンパイラにおけるループ最適化の検討
情報処理学会研究報告 1999-HPC-77、pp.65-70、1999.