VLDP3 アーキテクチャの構想 (1) ～プロセッサ構成～

入江 英嗣[†] 山口 健輔[†] 谷地田 瞬[†‡] 田中 裕治[†]
服部 直也[†] 飯塚 大介[†] 坂井 修一[†] 田中 英彦[†]

1. はじめに

コンピューティング能力の向上に関する社会の要請は留まる所を知らず、処理の中核となるマイクロプロセッサには、依然として高い性能向上率が求められている。マイクロプロセッサは、デバイス技術とアーキテクチャ技術の両輪によって、高い性能向上率を得て来た。デバイスの微細化は今後 10 年以上堅調な見通しであり [1]、数億ものゲートから成る大規模プロセッサも現実のものとなりつつある。一方、アーキテクチャ技術は、省電力化、SMT 化等多様な技術が研究されているが、コンピューティングの本質である、シングルスレッド処理に関しては、近年では大きな変化は見られない。

我々は、従来アーキテクチャでは、将来可能となるトランジスタ資源を有効に活用できないとの立場から、十数年先のデバイス技術を仮定し、次世代アーキテクチャ、大規模データパス（VeryLargeDataPath）アーキテクチャの研究開発を行ってきた [2]。

クロック高速化には遅延による限界があるため、VLDP プロセッサは特に命令レベル並列実行に注目している。

本稿では、VLDP アーキテクチャの最新モデルである VLDP3 プロセッサについて、ベースラインモデル、プロセッサの特徴、検討課題などを簡単にまとめ、VLDP3 研究のインデックスとする。

2. 関連研究

2.1 現行アーキテクチャの問題点

命令レベル並列性を抽出し、動的に並列実行を行うアーキテクチャとしては、現行のスーパスカラが挙げられる。現在の規模では IPC（instruction per cycle：実行並列度）は 2 程度と言われているが、これを大規模化することにより、つまり、メモリ-プロセッサ間の帯域を広げ、命令ウィンドウを拡大し、レジスタのリード/ライトポートを増やし、演算器を充分に配置する、等により、論理的には数倍の IPC をひきだすことができると言われている。

しかし、飛躍的な成長が見込まれているデバイス技術をもってしても、このような「巨大スーパスカラ」は、以下のような理由で投下した資源に見合った性能向上を得られないと考えられる。

まず、実行コード中には分岐命令が存在するため、単に命令メモリ-プロセッサ間の帯域を増やすだけでは、命令フェッチがボトルネックとなる。次に、命令ウィンドウ内から同時発火可能命令を検索し、演算器へ送るロジックは、ウィンドウ内全エントリの相互比較回路であり、大規模化すると時間オーバヘッドが増加する。また、大規模並列実行では頻繁にレジスタ、メモリが参照されるため、これらのポート数、アクセスレイテンシの影響が深刻となる。さらに、拡大したパイプライン段数、各資源の間を結ぶフォワーディングパスが肥大あるいは不足し、データを有効に転送することが難しくなる。

[†]東京大学
[‡](株) 日立製作所

VLDP3ブロック図

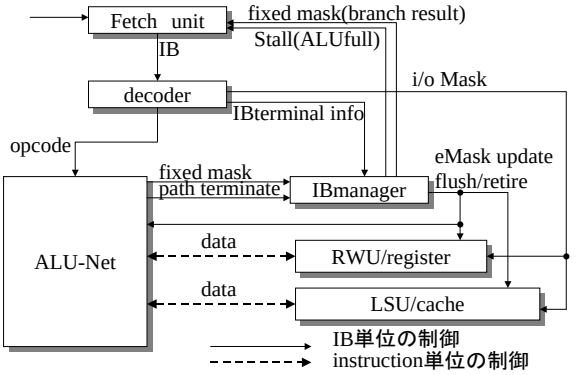

図 1: VLDP3 プロセッサの構成

2.2 VLDP2 アーキテクチャ

このような問題に対し我々は (1) 静的命令クラスタリングを含む、静的解析情報の利用。(2) データ依存の時間的局所性に着目した階層化。を中心とした様様な要素技術の提案を行った。また、1999 年からの VLDP2 プロジェクトではそれら要素技術を盛り込んだアーキテクチャ仕様を策定し、パイプラインシミュレータによる評価を行った [3]。

VLDP2 アーキテクチャは最大 32 命令で構成される命令ブロック (IB) を処理単位としており、このことにより動的負荷を軽減し、大規模フェッチ、高速デコード、大規模レジスタリネーミング等を実現している。また、複数パス実行の制御機構が実装され、分岐予測ミスを隠蔽する。フォワーディングパスは、階層的に構成され、配線量の爆発を防いでいる。同一 IB 内で生成、使用されるデータは IB 内用のフォワーディングパスを経由し、高速に転送される。

VLDP3 アーキテクチャは、これらの知見の上に、データ転送の高速化、制御コストの削減、IB 実行の高速化、静的支援の更なる利用等を目指している。

3. VLDP3 ベースラインモデル

VLDP3 のブロック概念図を図 1 に示す。

3.1 VLDP 実行コード

VLDP3 では実行コードは IB というフォーマットで与えられる。IB は複数の命令をひとかたまりとし、更に静的解析情報を加えたもので、制御の基本単位となる。同じ IB に含まれる命令同士に制御およびデータ依存が存在しても良く、それは実行に反映される。この点では、従来の RISC 型命令の opcode を実行順にそのまま塊に

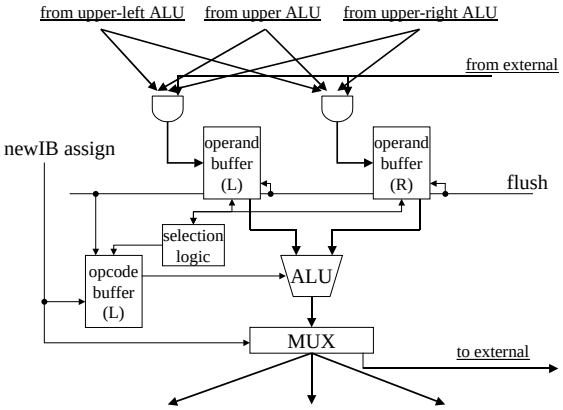

図 2: node ブロック図

したものと言える。命令の塊は、静的分岐情報に拠って作成される。

付加される静的解析情報は様様にあるが、重要なものを以下で説明する。

- IB 内データフローグラフ

VLDP3 の実行機構は、多数の演算器 (node) が接続された ALU-Net となっており、コンパイラによって IB に付加されたデータフローグラフ情報が、そのままアサインされて実行される。

- IB の入力、出力情報

異なる IB 間に跨るデータ依存やメモリアクセスは、動的に変化するため、ALU-Net によって完全に吸収することはできず、論理レジスタやメモリを介して値を受け渡す。IB にはこのような、ALU-Net の外部との値の授受についての情報が含まれる。また、IB 内には複数の制御分岐が含まれるため、IB の実行結果にも複数のパターンが存在する。このため、IB には、IB 内各命令出力の有効/無効を指定する mask が複数用意され、実行時に動的に選択、適用される。

このような仕様により、VLDP3 の IB はプログラムの種類によらず、効率よく充填することができる。

3.2 処理の流れ

命令フェッチ〜デコード〜ALU-Net 各 node への命令アサインはパイプライン処理され、1IB/cycle のスループットを持つ。IB フェッチを続けるため、フェッチアドレスを予測して投機実行を行う。従来の分岐予測にあたるものとして、mask 予測を行う。デコードは、IB に含まれる静的情報によって高速に行われる。opcode や接続情報が送信され、IB のデータフローグラフが ALU-Net 上に構成される。一方で、入出力情報が RWU(Read Write Unit)、LSU(Load Store Unit) へ送られ、それぞれのユニットではデータ転送処理を開始する。

ALU-Net にアサインされた各命令は、既に様様な依存が解消された状態になっているので、必要なデータが揃い次第演算を行い、指定された方向へデータを出力すれば良い。このため、各 node は非常にシンプルになる (図 2)。各 node は独立して動作し、データフローグラフの指定に従い並列実行が行われる。

ALU-Net には同時に複数の IB がアサインされ、IB レベルでも並列実行が行われる。これらの命令の IB 多重 out-of-order 実行を可能とするため、レジスタ、メモリでは動的なリネーミング処理を行う。VLDP3 はこのように IB 内並列性と IB 間並列性の双方を利用する。

また、メモリアクセス機構では、高速化のため、ストア-ロード依存投機実行を行い、キャッシュに先行アクセスする。LSU では、この時の予測ミス検出や、投機ストア値の保持などを行う。

4. 今後の課題

4.1 ベースラインモデルの評価

いくつかの予備評価をフィードバックしながら、ベースラインモデルをサイクルレベルのシミュレータにより作成し、前述したような VLDP3 アーキテクチャの特徴を評価する。また、最適なパラメタの検討や処理オーバヘッドの検出を行う。

4.2 データ転送オーバヘッドの削減

VLDP3 ではデータ転送についてオーバヘッドの削減を図っている。生成されたデータがすぐ用いられるようなケース (これは転送レイテンシによるオーバヘッドが大きくなり得るケースである) は、基本的に IB 内依存となり、コンパイラが直接転送経路を指定し、LocalWire による高速な転送が行われる。また、IB に着目した階層化により、フォワーディングパスの配線量の爆発的増加を防いでいる。

一方で、オーバヘッドとなる可能性のある転送として、レジスタやメモリを経由する転送が挙げられる。特に、隣接 IB 間のレジスタ依存や、メモリアクセスは僅かな転送レイテンシでも性能に大きな影響を与えてしまう恐れがある。これらについて、レジスタを介さない IB 間フォワーディング手法や、依存投機手法によるメモリアクセス機構、静的支援によるメモリ・フォワーディングなどを検討している。

また、演算器と記憶バッファと接続網の集まりである ALU-Net は、積極的に活用することによって、大部分のデータ転送を LocalWire 上の転送に吸収する潜在性を持っている。このような ALU-Net 利用の可能性と、手法の一般化についての検討を行っていく。

5. まとめ

本稿では、次世代命令レベル並列実行アーキテクチャとして我々が研究を行っている VLDP3 について概要を示した。VLDP3 は複数命令のかたまりである IB を処理単位として制御の高速化を図り、多数演算器の結合である ALU-Net 上に同時に多数の IB のデータフローを展開して並列実行を行う。各 node のロジックが非常にシンプルになること、データ転送が IB によって階層化されていること等が特徴である。

参考文献

[1] Summary of the International Technology Roadmap for Semiconductors, Semiconductor Industry Association, February 6,2002

[2] 田中英彦: ここいらで、計算機アーキテクチャを再考しよう, 情報処理学会研究報告 計算機アーキテクチャ研究会, 94-ARC-108, Vol 94, No. 91, pp. 33-40

[3] 辻秀典, 安島雄一郎, 坂井修一, 田中英彦: 大規模データパス・プロセッサの提案, 情報処理学会研究報告 計算機アーキテクチャ研究会, 2000-ARC-139, Vol. 2000, No. 74. pp. 55-60