1Q-09　VLDP アーキテクチャにおける実行パス制御方式の検討

高峰 信, 中村 友洋, 吉瀬 謙二, 辻 秀典, 安島 雄一郎, 坂井 修一, 田中 英彦
東京大学大学院 工学系研究科

1　はじめに

近年、プロセッサの高速化に対する要求は高まるばかりであるが、従来のアーキテクチャでは、命令レベル並列度の利用に関してその限界が指摘されている。当研究室では、多数の演算器をネットワークで接続し (ALU-Net)、ALU-Net 上でデータパスの構築を行うプロセッサ (VLDP プロセッサ) の開発を行っている [1]。VLDP プロセッサ中の機構で、投機的にフェッチした命令間のデータ依存関係を解析し、ALU-Net に複数のデータパスを構築する機構をデータパス先行展開と呼ぶ。本稿では、データパス先行展開の実現に向けて問題となる点について検討する。

2　基本的なデータパス先行展開モデル

データパス先行展開は、制御依存関係が解消された命令群から、各制御パス毎にデータ依存関係に基づいたデータパスを作成し、実行に先駆けて ALU-Net 上に割り付けを行う機構である。ALU 間のネットワークを効率良く利用するためには、フェッチ、デコード、発行、実行、レジスタへの書き戻し、という流れに加えて、次のような手順が必要となる。(1) デコードされた命令間でのデータ依存解析。(2)ALU-Net への命令の割り付け。(3)ALU-Net での実行パスの作成。データパス先行展開では、デコード→データ依存解析→割り付け→実行パス作成の動作を制御する。図 1はこの動作を模式的に表したものである。以降で複数実行パスを ALU-Net 上に構築する動作について、図 2に示すような従来のプロセッサ [2] を拡張したモデルを用いた場合の実行の様子を示し、実現可能性とボトルネックについて考察を行う。

ここで、データ依存解析を行う命令列は、すでにデコードされ制御依存が解消されているものとして考える。実際は、コントロールフロー先行展開でフェッチされた命令群に対して、十分な速度でデコードを行う回路は複雑かつ大規模なものとなり、デコーダ回路による全体の速度低下も十分に考えられる。

A Study of the data path construction unit on VLDP architecture
Makoto Takamine, Tomohiro Nakamura, Kenji Kise, Hidenori Tsuji, Yuichiro Ajima,
Shuichi Sakai, Hidehiko Tanaka
Graduate School of Engineering, The University of Tokyo

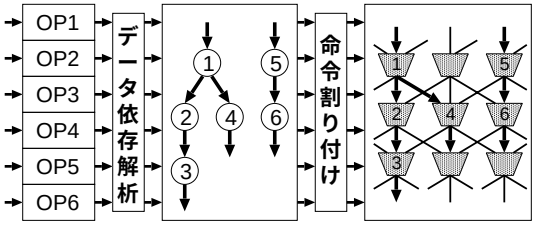

図 1: データフロー先行展開イメージ図

2.1　データ依存解析

VLDP アーキテクチャでは、オペランドの供給をできる限り ALU-Net 上で行うために、命令間でのデータ依存解析が重要となる。解析する命令列は制御依存が解消された複数のデータパスを含むものとする。デコードされた命令列は命令ウィンドウに入れられ、データ依存解析が行われる。単純なデータ依存解析の機構を考えれば、各命令ウィンドウ中のソースオペランドに対し、最新のデータを作成する命令が命令ウィンドウ内に存在するかどうかを検索すれば良い。さらにデータフロー先行展開では、ALU-Net に対して先行割り付けを行っているため、ALU-Net 上にも未実行の命令が存在する。このような命令についても、割り付けられた ALU の番号とデスティネーションレジスタの値を保持したテーブルを用意し、同時に依存関係を解析する。この検索によって各命令はソースオペランドを提供する命令が入っている命令ウィンドウの番号、もしくは ALU の番号を示すタグを得る。この時点でソースオペランドがレジスタを示しているものは、現在のレジスタの値が命令ウィンドウに転送される。

2.2　ALU-Net への割り付け

命令ウィンドウ中でデータ依存が解析され、オペランドがそろった命令は ALU-Net に割り付けられる。オペランドがそろった命令とは、ソースオペランドとして実際の値か ALU の番号を示すタグを持っている命令である。命令を新しく割り付けるには、ALU 同士の接続を示したテーブルを参照する。このテーブルは当該命令がタグとして持っている ALU の番号を入れると、それに接続されている空き ALU の番号を返す。空き ALU があれば命令はその ALU に割り付けられる。その際、割り付けられる命令が入っていた命令ウィンドウの番号を

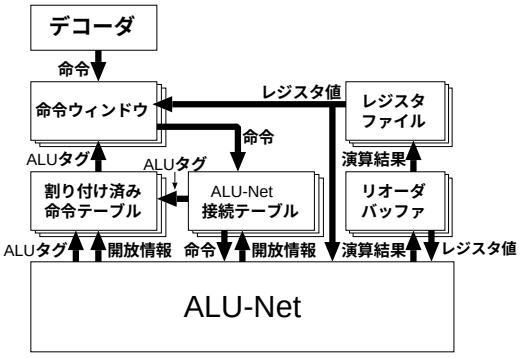

図 2: データパス先行展開モデル

タグとして持っている命令ウィンドウ中の命令のオペランドを、新しく割り付けられた ALU の番号に変更する。テーブル中の 1 つの ALU に対して 2 つ以上の問い合わせが重なった場合、命令間に優先順位を設けて、複数の命令が同じ ALU に割り付けられないようにする。ALU の空きがなければ、適当な空いている ALU に配置され、オペランドはレジスタから供給される。ただしこの場合、解消したデータ依存関係がくずれてしまう場合があるため、レジスタ名前替えの機構を用意する。複数実行パスを扱うため、多重化したリオーダ・バッファによるレジスタ名前替え [3] を使うことで対応する。

演算が終了した命令は、演算結果をレジスタに書き戻す。同時に ALU を解放し、ALU の接続を表すテーブルを更新する。命令ウィンドウ中のオペランドで当該 ALU 番号を指しているものがあれば、レジスタを介してその値に演算結果を転送する。

2.3 実行パスの作成

ALU-Net 中の 1 つの ALU の出力は複数の ALU の入力に接続されている。データパス先行展開では、演算結果が必要な ALU だけに供給されるように、ALU 間の接続の ON/OFF を行ってデータの流れを制御し、実行パスを作成する必要がある。ここで、命令は割り付け時に ALU 間の接続も考慮して配置されているため、この制御は容易に行うことができる。演算結果の出力は接続されている全てのラッチに出力される。オペランドのタグとして ALU の番号を持っている命令についてのみ、接続されたラッチからオペランドを参照する。

3 基本モデルについての考察

このモデルでは、データ依存解析はオペランドの比較によって実現している。しかし、検索は命令ウィンドウ内と ALU-Net 上の命令について行うため、大規模フェッチと大規模 ALU を前提とした VLDP アーキテクチャでは、実現するためのハードウェアのコストを考えた場合、この手法は適当でない。命令ウィンドウのサイズを 32、ALU の個数を 100 とした仮定した場合、1 サイクルで全ての命令に関して比較を行うのに必要な比較器の個数は 1192 と計算でき現実的ではない。広い範囲でのデータ依存解析を高速に行うためには、コンパイラによる静的な依存関係の解析といった支援技術や、全く新しい検索機構の提案が必要である。

ALU-Net への割り付けでは、1 つの ALU が持つファンアウトの制限が問題となる。全ての ALU 同士の接続が可能なモデルでは、依存解析した全ての命令を同時に ALU-Net に割り付けることができる。現実的なモデルではネットワークの制限から、レジスタを介するデータの授受は避けられない。また、今回のモデルでは、データ依存関係のある命令は、ソースオペランドを供給する命令の割り付けが行われないと、割り付けが行われない。1 サイクルで割り付けた命令は、そのほとんどが次の 1 サイクルで演算が完了する。命令ウィンドウでの依存解析が十分な速度で行われ、かつ ALU-Net の個数が十分にあるとすれば、依存関係のある命令を同時に割り付ける機構がないと、割り付け性能が全体のクリティカルパスになる。

4 まとめ

VLDP アーキテクチャでデータ依存解析と ALU-Net への割り付けを行う機構であるデータパス先行展開についてその動作を説明し、従来のプロセッサの拡張によって実現しようとした場合に生じる問題を明らかにした。今回用いたモデルでは最低限の動作の保証はできるが、ハードウェア量が膨大になり、十分な動作速度を得ることはできない。データ依存解析部では、効率的な依存解析を行う新しい機構を導入する必要があり、命令割り付け部は、依存関係のある命令を同時に割り付けることを考えなければ、割り付け動作がクリティカルパスになる。今後はこのボトルネックを解消する手法について研究を行い、VLDP の実装を目指す。

参考文献

[1] 中村友洋, 吉瀬謙二, 辻秀典, 安島雄一郎, 田中英彦. 大規模データパスプロセッサの構想, ARC124-3, pp. 13-18, 1997.

[2] マイク・ジョンソン. スーパースカラ・プロセッサ. 日経 BP センター.

[3] 安島 雄一郎, 中村 友洋, 吉瀬 謙二, 辻 秀典, 田中 英彦. スーパスカラ・アーキテクチャのための複数パス実行機構の提案, 並列分散シンポジウム JSPP'98, Vol.98, No.7, pp.23-30, Jun. 1998.