6G-05

FPGA 上での 4ALU プロセッサの並列・連鎖演算ユニットの設計

岩見佑一郎† 古川晋也† 孟林‡ 山崎勝弘‡

立命館大学大学院 理工学研究科† 立命館大学 理工学部‡

1. はじめに

近年、GPU やマルチコアプロセッサなどの性能が急速に向上し、並列化技術が必須となっている。我々は、細粒度の演算レベル並列性の検証として、複数の ALU を有するマルチ ALU プロセッサの設計と、FPGA 上での実装と評価を行ってきた[1]～[3]。

本稿では、4ALU プロセッサの並列・連鎖演算の分類と、並列性を動的に判断する並列・連鎖演算ユニットの設計および命令列を並べ替える命令スケジューラについて述べる。

2. 4ALU プロセッサの構成

2.1 システム構成

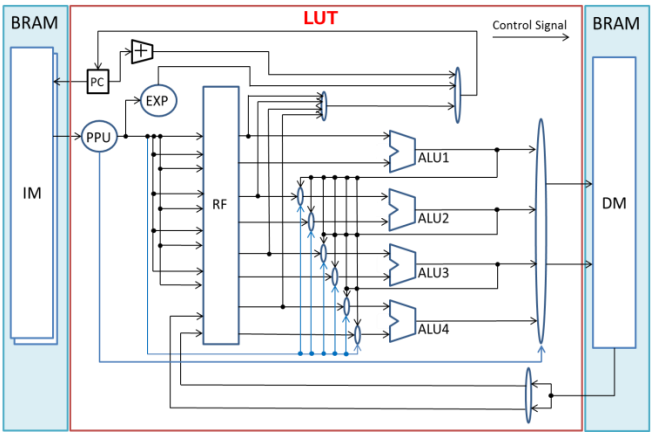

図 1. 4ALU プロセッサの構成

図 1 に 4ALU プロセッサの構成を示す。本プロセッサは 4 つの算術論理演算回路(ALU)を有した、レジスタ (RF) 数 32 個の 32 ビットプロセッサである。また、命令セットは MIPS に準拠し、命令メモリ(IM)とデータメモリ(DM)が分離したハーバード・アーキテクチャである。IM は 2 ポート BRAM を 2 重にし、4 命令同時フェッチを可能としている。また、命令を並列処理する PPU (Parallel Processing Unit) は、4 命令間の並列実行判定部分及び制御部である。

2.2 並列・連鎖演算の分類

図 2 に 4 命令間の並列・連鎖演算の分類を示す。これは 4 命令間に存在し得る並列、連鎖のパターンを示している。2 並列・2 連鎖×2 のパターンは、(a)と(b)に分類されている。その区別としては、1 命令間に連鎖が 2 つ存在しているかそれぞれ独立して連鎖が存在しているか

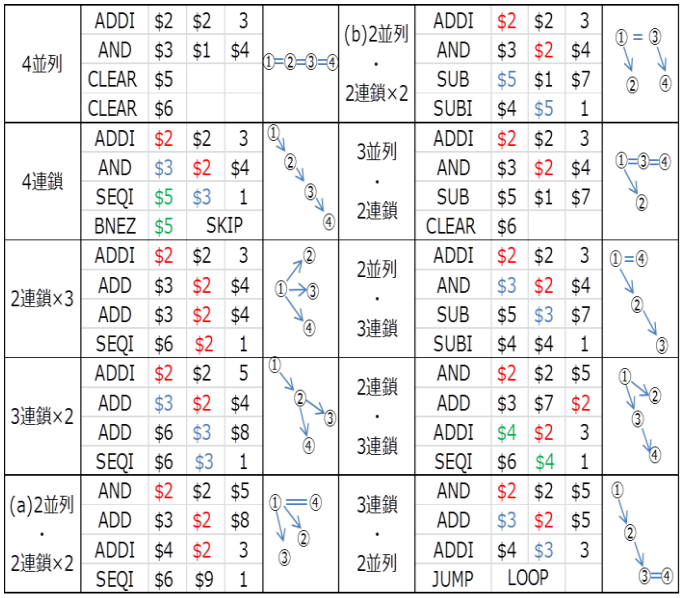

図 2. 並列・連鎖演算の分類

である。1 次 Booth 乗算における並列・連鎖演算の分類を図 3 に示す。ここでは、被乗数 x =2、乗数 y=(-3)とし、それぞれ 4 ビットであり、①～⑧の順で実行される。4 命令間で 2 並列・3 連鎖が検出(②)されており、また、3 命令間でも 2 連鎖・2 並列が検出(④、⑧)されている。

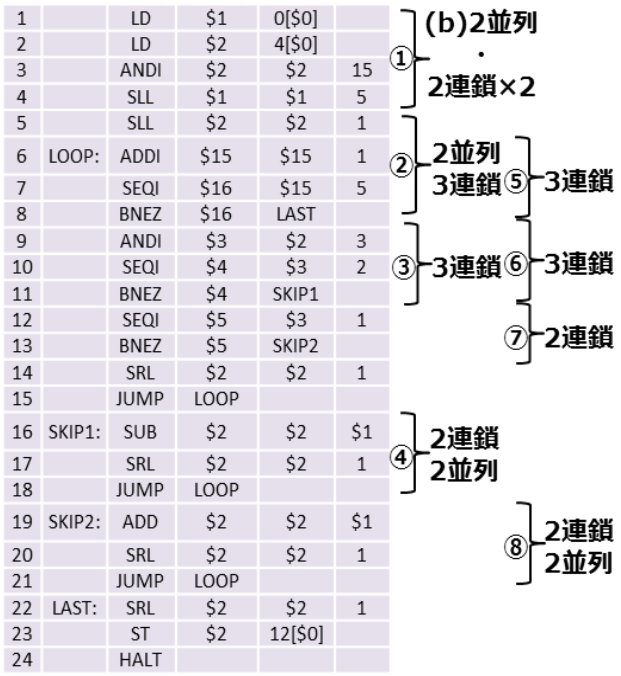

図 3. 1 次 Booth 乗算のアセンブリプログラム

Design of Parallel/Chaining Operation Unit for 4ALU Processor on a FPGA Board.

Yuichiro Iwami†,Shinnya Furukawa†, Lin Meng‡, and Katsuhiro Yamazaki‡

†Graduate School of Science and Engineering, Ritsumeikan University.

‡College of Science and Engineering, Ritsumeikan University.

2.3 PPU のアルゴリズム

図 4 に PPU のアルゴリズムを示す。

① 同時実行可能な並列演算と連鎖演算を実行時に検出する。

② 分岐命令までの一連の演算を並列処理する。最上位命令の場合、単一実行を行う。

③ データ依存がある場合、依存先を示すフラグを立て、依存先の ALU の前のマルチプレクサを制御する。

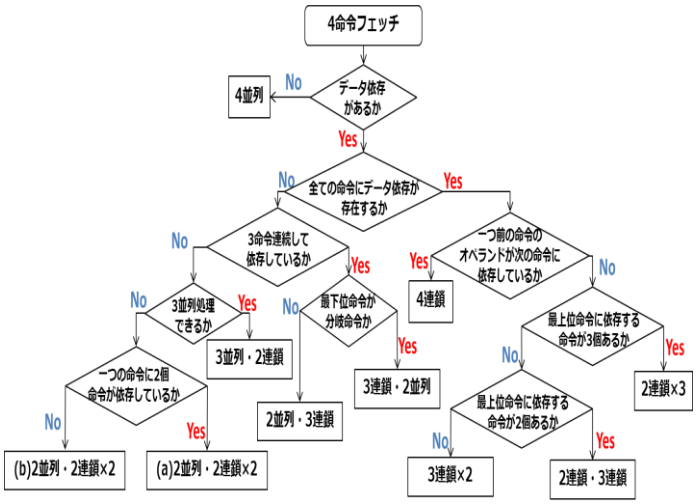

図 4. PPU のアルゴリズム

3. 並列・連鎖演算ユニットの設計

図 5 に PPU と ALU の制御パスとプログラム例を示す。まずフェッチされた 4 命令を PPU 内にて解析し、それらのオペランドを調べる。そこで、データ依存が存在していれば、依存フラグを 1 とする。この依存フラグは、ALU2～4 の前のマルチプレクサと対応しており、どの命令と依存関係にあるかを示している。これにより、依存している ALU の演算結果を、次の ALU の入力先に送る。依存がない場合、レジスタファイルから値を ALU に送る。

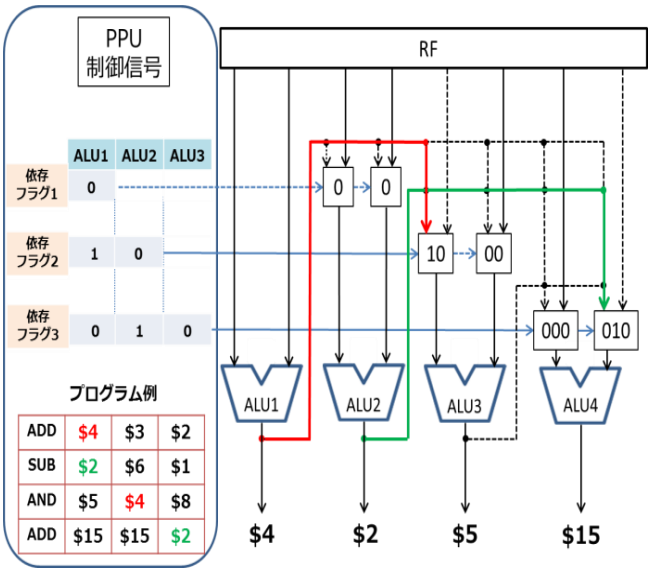

図 5. PPU による並列・連鎖演算

プログラムの例に沿って述べると、まず最上位命令の格納先レジスタ$4 が、第 3 命令のソースオペランドに用いられているため、PPU 内の ALU3 と対応する依存フラグ 2 の ALU1 のフラグを 1 にする。これにより、最上位命令の演算結果$4 を直接 ALU3 に送ることができる。第 2 命令も同様に、第 4 命令と依存関係にあるので依存フラグ 3 の ALU2 のフラグを 1 とし結果を送る。このように、どの命令が依存関係にあるかを PPU 内で判別、制御することにより依存命令を動的に処理することができる。

4. 命令スケジューラの設計

4ALU プロセッサの DM は 2 ポート BRAM で実現されており、メモリアクセス命令(MA)が 3 つ以上連続して存在する場合、4 命令同時処理が不可能である。そこで、命令列の並列性を上げるため、命令入れ替えを行う命令スケジューラを設計した。

① フェッチされた 4 命令に、メモリアクセス命令（LD, ST）が 3 つ以上連続して存在する場合、3 つ目以降の命令を入れ替えられる命令があるか探索する。

② メモリアクセス命令を含むセグメント内に、メモリアクセス命令とデータ依存のない命令があり、移動させても影響がなければ、命令の入れ替えを行う。そのセグメント外でも定数生成のように移動できる命令がある。ここでのセグメントとは、先頭から最後まで順に実行される命令系列を示す。

5. 考察

4 並列演算と 4 連鎖演算の実行時間は異なるので、マルチサイクル設計、最適な段数のパイプライン設計、及び並列・連鎖演算時の最適なクロック数の検討が必要である。また、全体のハードウェア量を明らかにする必要がある。これらの課題を解決した上で、今後、テストプログラムを増やし演算の種類が多いパターンの検証をすることで、高速化が図れると考える。

6. おわりに

本研究では、4ALU プロセッサによる並列・連鎖演算ユニットの設計、及び命令スケジューラの設計を行った。これにより、4 命令間の並列・連鎖演算を動的に処理する手法を提案した。今後、様々なプログラムを用いてテストを行い、連鎖が効果的なパターンの検証、4 連鎖時のクリティカルパスをどのように処理するか、また、より効率化を図るため、パイプライン設計を加えることの検討を進めていく必要がある。

参考文献

[1] 石川、杵川、境、孟、山崎：演算レベル並列処理用マルチ ALU プロセッサの設計と実現、FIT2013、C-005、2013.

[2] 杵川、石川、岩見、孟、山崎：4ALU プロセッサの設計と並列・連鎖演算の検出、電子情報通信学会総合大会、2015.

[3] 岩見、石川、杵川、孟、山崎：マルチ ALU プロセッサによる Booth 乗算の並列性評価、電子情報通信学会総合大会、2015.