3J－01　並列アルゴリズムクラスに基づくハードウェア自動生成

立命館大学大学院　理工学研究科
上平　祥嗣　山崎　勝弘

1　はじめに

並列プログラム動作させることができる最適なハードウェアの自動生成について述べる。また、並列アルゴリズムクラスの一つであるプロセッサファームエンジンの動作原理と実装について述べる。

2　並列処理とハードウェア自動生成

2.1　プロセッサファーム

並列アルゴリズムクラスの一つ。問題を、複数の独立な計算に分割し、それらの結果を統合して最終的な解を得る。マスターが問題を複数の独立したタスクに分割し、各タスクを各スレーブに実行させて、結果を統合する。並列アルゴリズムの中で並列効果が期待できる[1]。

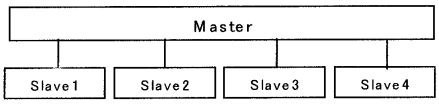

図 1　プロセッサファーム

2.2　システム構成

POSIX スレッドや Open-MP で記述された並列プログラムの機械語コード生成し、その機械語コードとハードウェアの規模、目標性能などの条件に基づいて最小、かつ最適なハードウェアを自動生成する。

本システムは、様々なプロセッサを対象とできるが、本研究には、KITE マイクロプロセッサを用いる。コンパイラは、プロセッサに対応した機械語コードを生成する。ハードウェア生成器によって生成されたハードウェア構成データは FPGA 上でハードウェアとして実現させる。

"Automatic Hardware Generation For Parallel Algorithm　Classes"
Yoshitsugu Uedaira and Katsuhiro Yamazaki
Ritsumeikan Univ.

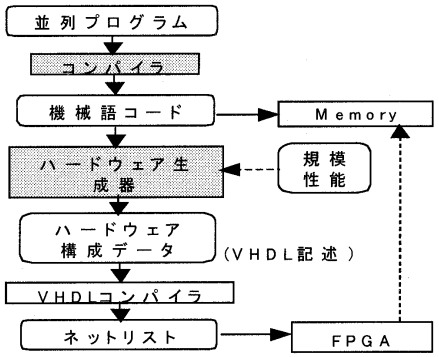

図 2　システム構成図

2.3　CPU の最適化

CPU の生成に関しては、CPU モデルから不要な命令を省いて最小化した CPU を生成する。最適化できる可能性のある CPU 機能を以下に述べる。

- シーケンサのステートマシン記述の削減
- ALU の機能削減
- レジスタの削減
- レジスタのビット数の削減（IX,SP）
- 使用されるアドレス方式による MUX の削減

2.4　KITE マイクロプロセッサ[2]

KITE マイクロプロセッサは、データ幅 16bit、アドレス幅 12bit 教育用の逐次処理型マイクロプロセッサである。

- 汎用レジスタ 4 つ
- 命令数 31
- ゲート数 6000 ゲート程度
- 5 種類のアドレス方式

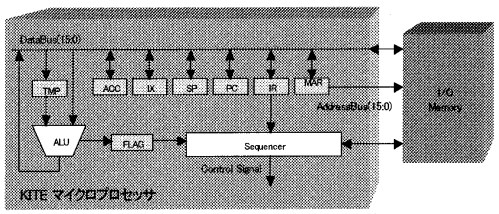

図 3　KITE マイクロプロセッサ

3 プロセッサファームエンジン

プロセッサファームを実行できる最適なハードウェアを自動生成する。それぞれのスレッドがアクセス要求を出した場合に、アクセス競合が起きるので、その問題を解決する。

3.1 プロセッサファームエンジンの動作原理

プログラムが実行されると、まず Master が動作を開始する。Slave は、待ち状態となっている。Master が前処理を終了すると Slave に Start Signal を送る。この Signal に伴って、Slave は動作を開始する。Slave はそれぞれ独自の Start アドレスを持っている。Master も Slave と同時に動作を開始する場合もある。Master が終了すると、Slave 終了待ち状態になる。Slave は終了すると Master に Halt Signal を送る。各 Slave が全て Halt Signal を転送すると、Master は各 Slave に対して Reset Signal を送り、後処理に移る。

3.2 プロセッサファームエンジンの内部仕様

プロセッサファームエンジンでは、Slave 動作中は、Memory Access Request が衝突する。この問題を解決するために、MMU を用意する。MMU は、Master または Slave からの MAR に対して均衡にアクセス許可を出す。メモリアクセスへの優先順位が Priority Register に格納されている。Priority Status Unit は、Priority Register の示す優先度毎に Master または Slave の MAR の有無を示す。

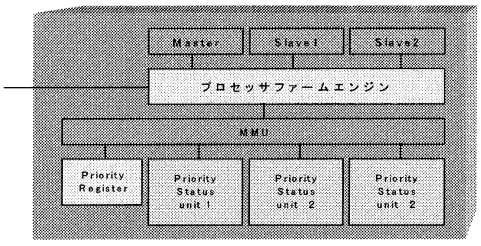

図 4 プロセッサファームエンジンの内部仕様

3.3 MMU の動作原理

複数のスレッドから要求があった場合、LRU に基づいて、適切な処理装置（Master or Slave n）にアクセス許可を行なう。メモリアクセス要求が途切れると即座に他の処理装置にアクセス権が移るようにした。

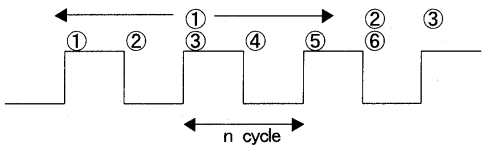

図 5 MMU のクロックサイクル

① Master、または Slave からアクセス要求が出る。
② アクセス許可信号出力
③ アクセス開始
④ Priority Register の値変更
⑤ アクセス要求解除
⑥ アクセス許可解除

3.4 FPGA での動作検証

プロセッサファームで動作するハードウェア構成データを手動で生成し、KITE マイクロプロセッサボードで動作検証した[3]。3 章で述べたハードウェア構成で正常動作することを確認した。

4 今後の研究方向

以前、KITE マイクロプロセッサを設計したことがあり、設計データがあるので、プロトタイプの意味を兼ねて、KITE マイクロプロセッサをターゲットとして研究を行っている。今後は、ハードウェア自動生成を行い、ハードウェア規模や性能評価を行う。しかし、KITE マイクロプロセッサは、キャッシュを搭載していないため、メモリアクセスの際にアクセス競合が頻発し、並列効果が出づらいと考えられる。そのためキャッシュを持つプロセッサをターゲットとして研究を進める必要がある。

5 おわりに

本研究では、並列アルゴリズムクラスに基づいたハードウェア自動生成に関して述べた。プロセッサファームを実行するハードウェアの動作検証を終えて、自動生成を行っている。

参考文献

[1] 安藤、山崎：“類似事例を用いた並列プログラミング” *bit*,Vol.30,No.7,PP23～30,1998.

[2] KITE マイクロプロセッサプロジェクト：“KITE マイクロプロセッサリファレンス・マニュアル Version 1.00” 1993.

[3] KITE マイクロプロセッサプロジェクト：”KITE マイクロプロセッサボード PLUS＋取扱説明書” ,1995.