2パス限定投機システムの提案 – マルチスレッド制御機構 –

佐藤 和史† 十鳥 弘泰†† 福田 明宏†† 米田 淳一† 大津 金光† 横田 隆史† 馬場 敬信†
†宇都宮大学大学院工学研究科情報工学専攻 ††宇都宮大学工学部情報工学科

1 はじめに

半導体の集積技術の向上により計算機システムにおける利用可能なハードウェア資源は増大しているが，一方でクロック速度の向上が飽和する状況になりつつある．現在は，半導体素子の改良やクロック速度の向上に頼ることなく，アプリケーションプログラムの実行を高速化する手法が求められている．この課題に対して，我々の研究室ではプログラム中の経路（パス）を投機の対象とする2パス限定投機方式の提案を行っている[1]．この実行方式について，これまでトレースベースのシミュレーションにより性能評価を行い，有効性を示してきた．

本研究では，2パス限定投機方式の実現アーキテクチャ“PALS”を設計する．特に、2パス限定方式を実現する上で最も重要なスレッドの管理制御を行うスレッド制御機構 (TMU: Thread Management Unit) について設計する．

2 2パス限定投機方式

2パス限定投機方式について述べる．まず，2パス限定投機方式はプログラムの実行前にプログラム内の各ループの実行パスの実行頻度についてプロファイルを行う．プロファイル情報をもとに実行頻度の高い上位二つのパスについて最適化したコード (投機スレッドコード) を生成する．投機スレッドコードは実行パス以外のコードとそれらへの分岐命令を削除したコードであり，もとの逐次コードと比べ高速に実行が可能である．プログラムの実行時には1イテレーション毎にどちらのパスを実行するかをパス予測器によって予測し，それを処理ユニットの一つに通知してスレッド実行を開始する．この実行方式ではループイテレーションの順番がスレッドの順番となる．実行中のスレッドの中で最もイテレーションの順番が早いスレッドを先頭のスレッドとし，最もイテレーションの順番が遅いスレッドを末尾のスレッドとする．

スレッドは投機的に実行されるため，パスが正しく実行されていることを保証する必要がある．これはassert命令によって行う．もし，処理ユニットが実行パスが間違っていること (パス違反) を検出した場合はもう一方のパスの投機スレッドコードを実行する．もう一方のパスも誤っていた場合は元の逐次処理のコードを実行する．パス違反を検出した処理ユニットはパス予測器からの予測結果を必要とせず，すぐに次のパスを投機実行を行うことができる．これにより実行再開までのディレイを削減できる．先行するスレッドがパス違反を起こした場合は後続のスレッドは実行を破棄し，新たにパス予測器からの予測結果を受け取り，スレッド実行を行う．本実行方式ではこのような動作を複数のスレッドに対して並列に行うことで，高速化を達成する．このような投機的マルチスレッド実行モデルを実現できるようにハードウェアを設計する．

3 2パス限定投機システム “PALS”

図1に2パス限定投機システム “PALS” のハードウェア構成を示す．

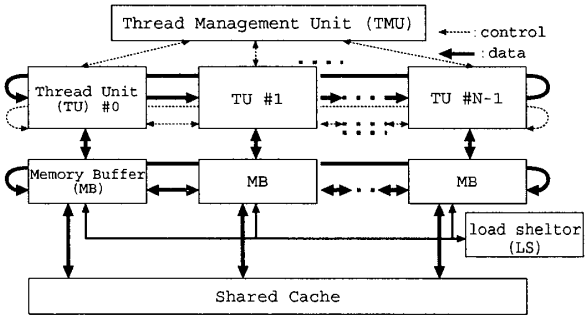

図 1: 2パス限定投機システムのハードウェア構成

図中の各ユニットの機能は以下の通りである．

- スレッド制御機構 (TMU)：プログラム実行時にパス予測を行い，スレッド起動を行う制御機構
- スレッドユニット (TU)：スレッドを実行する処理ユニット
- メモリバッファ (MB)：各TUに備わった投機的なメモリアクセスを処理するバッファ
- ロードシェルター (LS) :MBへの投機的なストアデータの書き込みを保証するためのロードデータの退避用メモリ

TMUは各TUと1対1で接続しており，各TUはMBを持ち，TU同士，MB同士はそれぞれ個別にリングバスで接続している．また，各MBはLSと共有キャッシュに接続している．スレッドの実行が確定したらMB内の投機的なストアデータをメモリに反映させる．

以上のようなハードウェア構成とするまでの検討過程を以下に示す．まず，2パス限定投機方式を実装する上で重要となるのが，パス予測器である．このパス予測器の配置については二通りある．TU毎に用意する分散型と一つのパス予測器に全てのTUがアクセスする集中型である．前者の方法を採用した場合，予測結果を遅延無しで全てのパス予測器に反映させるには処理ユニットを完全結合で接続する必要がある．これはハードウェアコストの面から現実的ではない．パス予測器は集中型を採用する．

続いてスレッド制御方法の検討を行う．スレッド制御とは，マルチスレッド実行において，プログラムの整合性を保つための動作である．PALSで想定するスレッド制御はスレッドの起動，停止，終了，破棄である．スレッドの起動はスレッド実行開始の動作，スレッドの停止は依存のある変数の値を後続のスレッドから

Proposal of Two-Path Limited Speculation System - Multithread Controll Unit -

†Kazufumi Satou, Junichi Yoneda, Kanemitsu Ootsu, Takeshi Yokota and Takanobu Baba
††Hiroyoshi Jutori, Akihiro Fukuda,
Department of Information Science, Graduate School of Engineering, Utsunomiya University (†)
Department of Information Science, Faculty of Engineering, Utsunomiya University (††)

送られてくるまで待つ動作，スレッドの終了はスレッド実行が正しく終了したことを他の TU 及びパス予測器に通知する動作，スレッドの破棄はパス違反によるスレッド実行の実行結果の無効化処理である．

多くの投機的マルチスレッド実行モデル [2] で実現しているように，PALS でもソフトウェアとハードウェアの両方を用いてスレッド制御を行う．先ほどの検討でスレッドの起動に必要な機構であるパス予測器について集中型を採用した．これを前提としてスレッドの起動をソフトウェアで行う場合とハードウェアで行う場合で考える．ソフトウェアで行う場合は TU がスレッド起動用の命令を検知し，パス予測器へ通知を送り，パス予測器は他の TU へスレッド実行開始の通知を送る．ハードウェアで行う場合は TU からの通知を待たずにパス予測器が自動で TU へ通知を送る．パス予測器が一つに対して TU は複数を想定しているため，ボトルネックとなる．そこで PALS ではパス予測器との通信を抑えるため，ハードウェアでの自動スレッド起動を行う．

ただし，パス予測器のみでは TU の状態を把握することができない．また，PALS ではスレッド間の依存関係解消のために TU 同士を接続し，データを送受信できるようにする．これを実現する構成はリング構造である．リング構造でランダムに TU へスレッドの起動を行っては依存関係を解消するための通信コストが増大する．そこで，パス予測器を内包し，TU の状態を管理するためのハードウェア機構としてスレッド制御機構 (TMU) を用意し，TMU が自動でリング構造に沿って順番にスレッド起動を行う．

他のスレッド制御については，パス予測器の通知を必要とせず，TU が起点となる動作である．TMU との通信を抑えるためにできるだけ TU 間の通信で対処し，TMU への通信は必要最低限に抑えるようにする．

以上の検討結果から PALS の設計を行った．

4 スレッド制御機構の設計

続いて TMU の設計について述べる．TMU の内部構成を図 2 に示す．

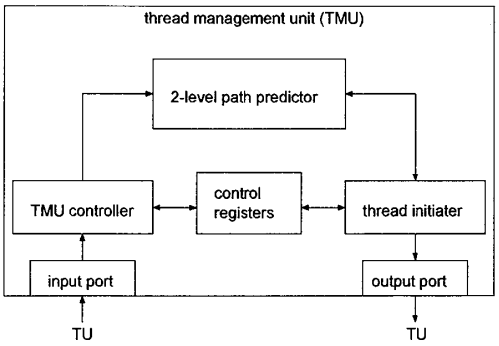

図 2: スレッド制御機構の内部構成

各ユニットの説明は以下の通りである．

- 2-level path predictor：2 レベル分岐予測器を基にした次に実行するパスを決定するパス予測器．
- TMU controller：TU からの通知を受け取り TMU の動作を決定する制御装置．
- thread initiater：2-level path predictor で予測したパスを TU に割り振るための制御装置．
- control registers：TU と TMU の状態を把握するためのレジスタ群．
  - heading thread unit number register：先頭のスレッドを実行中の TU の番号を記録．
  - tailing thread unit number register：末尾のスレッドを実行中の TU の番号を記録．
  - asserting thread unit number register：assert 不成立発生を通知した TU の番号を記録．
  - TU state register：各 TU が idle 状態か busy 状態かを記録．
  - TMU state register：TMU の状態を記録．

TMU の役割はパス予測を行い，予測結果を idle 状態の TU に送ることでスレッドを起動することである．これを実現しつつ TU との通信を抑えるためには TMU は TU の状態を記録する必要がある．そのため，TU state register を用意し各 TU の状態を 1 ビットで記録できるようにした．

また，TU のスレッド起動の順番はリング構造に沿って順番に行う．これを実現するために heading thread unit number register と tailing thread unit number register を用意する．TMU が先頭のスレッドと末尾のスレッドを実行している TU を把握することでスレッドの管理を容易にし，常に末尾のスレッドを実行している TU の次の TU へスレッドを起動することでリング構造に沿ったスレッド起動を可能にした．

さらにパス違反によるスレッドの破棄に対応する必要がある．スレッドの破棄が完了するには数サイクルかかる．この間に別のスレッドでパス違反が発生した場合にはスレッド破棄の通知が重複しプログラムの整合性が保てなくなる．これを防ぐためにパス違反を検知した TU が，先に発生したパス違反によるスレッド破棄が完了していることを把握する必要がある．これを全 TU の状態を記録している TMU に問い合わせることで実現する．TMU に各 TU がのスレッド破棄中かどうかを把握するための asserting thread unit number register と TMU state register を用意する．スレッド破棄の対象となる TU はパス違反を検出した TU から末尾のスレッドを実行している TU までの全ての TU である．パス違反が発生した場合にはスレッド破棄中であることを TMU state register に記録し，対象 TU の全てからスレッド破棄終了通知を受け取ったかどうかでスレッド破棄が完了したか判定する．これらを検討結果からプログラムの整合性を保つ設計を行った．

5 おわりに

2 パス限定投機システム “PALS” の設計について述べた．また，スレッド起動の役割を担う TMU の設計について述べた．今後は，動作検証を行い，2 パス限定投機方式の評価環境を完成する予定である．

謝辞 本研究は，一部日本学術振興会科学研究費補助金（基盤研究 (B)18300014，同 (C)19500037，同 (C)20500047）および宇都宮大学重点推進研究プロジェクトの援助による．

参考文献

[1] 横田隆史ほか: “2 パス限定投機方式の提案”，情報処理学会論文誌:コンピューティングシステム，Vol.46, No.SIG 16 (ACS-12), pp.1-13, 2005.12

[2] Gurindar S. Sohi, Scott E. Breach, T. N. Vijaykumar, “Multiscalar Processors”, In Proceedings of the 22nd Annual International Symposium on Computer Architecture, pages 414–425, June 22–24, 1995.