2H-5

# 投機メモリシステムのハードウェア実装

北 直樹 † 横田 隆史 † 大津 金光 † 馬場 敬信 †
† 宇都宮大学大学院工学研究科情報システム科学専攻

## 1 はじめに

プログラム実行中に、分岐を含む後続の処理を予測し、投機的に実行することで、高速にプログラムを実行できる。本研究室では、投機的な処理の手法として、2パス限定投機方式と、同方式を実現するアーキテクチャ PALS (PAth Limited Speculation) を提案した[1]。しかし、ハードウェアの詳細な構造は検討されていないため、ハードウェアではシミュレータで想定されない問題が発生する可能性がある。PALS のハードウェア上で発生する問題を調査・解決することを目的として、PALS 特有の機構である Memory Access Unit をハードウェアとして実装するために、Verilog-HDL による設計を行う。

## 2 2パス限定投機方式

2パス限定投機方式では、実行頻度の最も高い#1パスと、2番目に実行頻度の高い#2パスのコード、およびループイテレーションと同じ処理を行う逐次コードの3つのコードを生成する。#1パスと#2パスのうち、どちらのパスが実行されるかをスレッドごとに予測し、予測したパスの命令コードを投機実行する。投機に成功した場合、各イテレーションでの分岐を先行スレッドの終了を待たずに判断する分、高速に実行できる。投機実行に1度失敗した際は、後続のスレッドも含めて実行結果を破棄し、もう一方のパスを改めて投機対象として実行する。2つのパスのどちらも予測失敗となった場合は、逐次コードを非投機で実行する。

## 3 2パス限定投機システム PALS

本研究室は、前述の2パス限定投機方式を実現するため、図1の PALS アーキテクチャを提案した。

PALS では、投機スレッドの予測情報を、TMU (Thread Management Unit) から各 TU (Thread Unit) に送信する。TU では指定されたパスを投機実行し、その際に行われたメモリアクセスを MB (Memory Buffer) に記録する。スレッドの実行が終了した場合、先頭スレッドの TU は自 MB 内のストアデータを Data Cache に反映させてから、TMU に成功を伝え、直後のスレッドを先頭スレッドとし、自身は次のスレッドが TMU から指定されるまで待機する。

容量の限界で MB にデータを格納できなくなった場合、投機ストアデータ格納できる領域が無くなるため、以降の命令を実行できなくなる。このため、先頭スレッドのみ LS (Load Shelter) にロードの履歴を退避させ、投機ストアデータ用の領域を確保することが許されている。

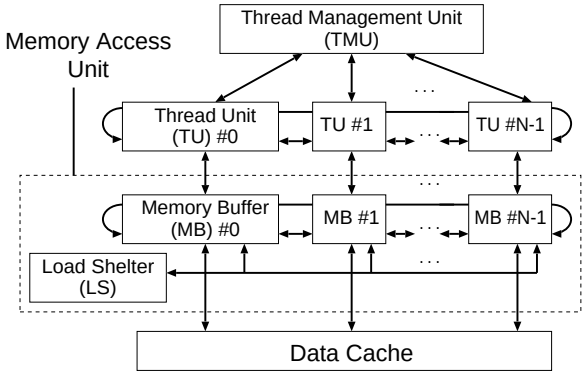


図 1: PALS 構成図

## 4 Memory Access Unit

投機的な処理を行うアーキテクチャでは、投機実行に失敗した場合、投機的に行われた処理を破棄し、スレッドが実行される前の状態に戻す回復処理が必要となる。回復処理を短時間で行うために、PALS では、スレッドごとの投機ストアデータを格納する記憶装置として MB を用意する。MB では、スレッドの投機成功が確定するまでメモリアクセスの内容を保持し、投機成功が確定すれば Data Cache にストアデータを反映させる。これにより、投機実行中は Data Cache 内のデータを書き換えることが無く、スレッドの投機実行失敗時、該当するスレッドの MB の内容を全て消去する処理のみで、回復処理が行える。

また、後続スレッドの実行の際に、実行中の先行スレッドでストアされたデータが必要になる場合があるため、MB 間でのデータ通信も行えるようにする。後続スレッドの実行に必要なデータを読み出す際、自 MB から順に先行 MB 内を探索し、実行中の先行スレッドでストアされたデータから、必要なデータを探索する。どの MB にもデータが存在しなければ、Data Cache から必要なデータを読み出す。

上記の MB 間での通信を行った後、後続スレッドによって読み出されたデータと同じアドレスにストアが行われた際、メモリ依存違反が発生したことになる。このため後続 MB にメモリ依存違反の発生を通知し、実行開始時点に戻すことで、整合性を保つ。

また、後続 MB からデータの読み出しが要求されても、データを返答する前にスレッドが終了する場合がある。その場合、後続 MB に読み出し要求を返却し、

Hardware Implementation of Speculative Memory System
†Naoki Kita, Takashi Yokota, Kanemitsu Ootsu and Takanobu Baba
Department of Information Systems Science, Graduate School of Engineering, Utsunomiya University (†)

ロード処理の実行を保証する。

スレッドの投機実行中に、MBの容量が不足することで、メモリアクセスの内容を記録できなくなり得るが、この場合には、メモリ依存違反検出のために必要となる、後続MBからのロード要求の情報、あるいは目的のデータが発見されていないロード要求の情報をLSに移動させることで、先頭MBの動作を保証する。ただし、LSにデータを移動できるのは先頭スレッドのMBのみである。

## 5 MBのハードウェア構成

MBは図2のように、MB内の処理を行うMB_COREと、入力ポート、出力ポートで構成される。

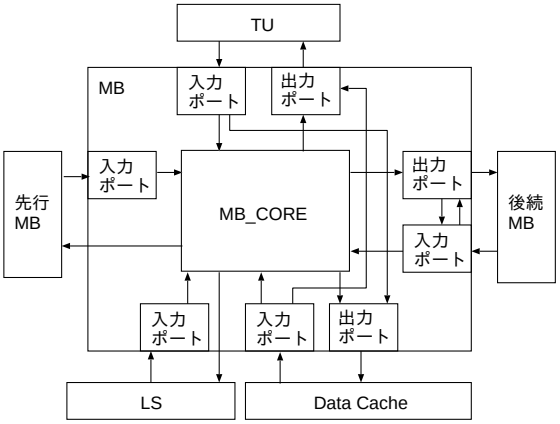


図2: MBの構成

TU，隣接するMB，LS，Data Cacheから MBに動作要求が入力されるが、入力された内容は同時に処理できないため、先に行われる処理が完了するまで、要求の内容を保存する入力ポートが必要となる。

また、ループ以外の処理を実行するため投機的な処理を行わない場合、MBにデータを格納する必要が無いので、TUとData Cacheの間でデータのやり取りを行う。ロード要求の返却を行う際には、入力ポート内のロード要求を後続MBに出力する。これらの動作においてメモリアクセス要求の経路を制御するため、3つの出力ポートが用意されている。

中央のMB_COREは、MBへの要求を処理するモジュールである。その構成は図3のようになる。

複数の入力ポートに要求が格納されている場合、優先度判定回路により、最優先で処理する要求を判定し、検索するエントリの内容を決定し、CAMunitに送る。

エントリを管理する記憶領域には、CAMunitとして設計した連想メモリを用いる。CAMunitはMB用に設計した連想メモリである。検索するエントリ記憶されているエントリの検索条件は、優先度や処理中の要求の内容などからCAMunit内で判断する。

出力制御装置では、CAMunitの検索結果をもとにその後の動作を決定する。TU，隣接MB，LS，Data Cacheへの要求の出力、CAMunit内に格納されたエントリの更新を制御する。

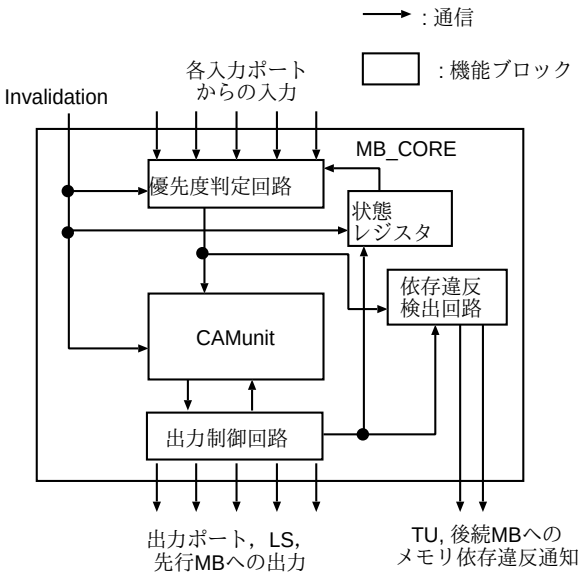


図3: MB_CORE内の機能モジュール

上記3つのモジュールにより、各要求の処理は、エントリの検索、出力またはエントリの更新、という一連の動作によって行われる。この一連の処理を行わない動作は、LSとの通信を行うために、LSからの返答があるまで待機するか、自身が先頭スレッドになるまで待機するかのいずれかである。よって、通常の検索動作、LSからの返答待機、先行スレッドの終了待機の3つの状態によって動作を制御する。これらの状態を管理するため、MBに状態制御回路を用意する。

メモリ依存違反の検出と処理は、メモリ依存違反検出回路で行う。自MBでメモリ依存違反を検出した場合は後続MBに、先行MBからのメモリ依存違反通知が入力された場合は後続MBに、メモリ依存違反通知を出力する。

各モジュールは、Invalidation信号によって初期化する。CAMunit内では全てのエントリを同時に初期化することで、短時間で回復処理を行えるようにした。

## 6 おわりに

MBのハードウェア構造が決定し、現在Verilog-HDLで設計を行っている。 設計が完了後、FPGAボード上に実装し、ハードウェア上で発生する問題について調査する。

**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 (C)20500047，同 (C)21500049，同 (C)21500050) および宇都宮大学若手萌芽的研究プロジェクトの援助による．

**参考文献**

[1] 十鳥 弘泰，大津 金光，横田 隆史，馬場 敬信: “2パス限定投機方式を実現するマルチコアプロセッサPALSの提案”，信学技法，Vol.109，No.319 (CPSY2009-46)，pp.19-24，2009.