過渡故障耐性を持つ Out-of-Order スーパスカラ・プロセッサの評価

†有馬 慧　‡岡田 崇志　‡堀尾 一生　‡喜多 貴信　‡塩谷 亮太　‡五島 正裕
‡坂井 修一

†東京大学工学部　電子情報工学科　　‡東京大学大学院　情報理工学系研究科

1 はじめに

近年のプロセッサ設計において，微細化に伴うばらつきの増加が大きな問題となっている．ばらつきには何種類かあるが，静的か動的かで分類できる．静的なばらつきでは，製造ばらつき，特に不純物密度分布のばらつきによって各トランジスタの閾値がばらつく影響が深刻であり，チップ内ばらつきの激化として顕著に現れている．これは，微細化によってトランジスタや配線が原子のサイズに近づいたことによるものであって，原理的に防ぐことが難しい．動的なばらつきには，ロジック動作による局所的な電源電圧低下や温度上昇などがあり，これらのばらつきも微細化の影響で増加している．その上，近年の省電力指向のため一般化しているDVFS (Dynamic Voltage and Frequency Scaling) によって，意図的に引き起こされる動的ばらつきもある．

回路遅延がばらつくと，同期式回路の設計におけるタイミング制約を満たすことが難しくなる．回路遅延の動的な変化によって信号のタイミングに齟齬が生じ，設計者の想定外の動作が生じることがある．これを動的なタイミング・フォルト (TF) という．

ばらつきが増大していくと，従来の最悪値に基づいたマージンをとる設計手法は悲観的になりすぎる．動的なTFを動的に検出・回復する技術を用いて，静的・動的の両面からばらつきを吸収する手法が注目されている．この手法では，シビアな動作環境下で動的なTFが例外的に発生することを認め，不要なマージンを削って効率的に動作させる．生じた動的なTFに対しては，適切な処理によって安定動作を保証する．本研究は，このアプローチの回復技術について，既存の手法に対する改良・検討を行うものである．

2 関連研究

動的なTFの検出/回復に関する代表的な技術に,RazorII[1]がある．本節ではこの技術について説明する．

2.1 RazorII（検出）

RazorIIは動的なTFを動的に検出するラッチである(図1)．動的なTFによってDから古い値がサンプリングされると，それを検出してエラー・ビットをたてる．

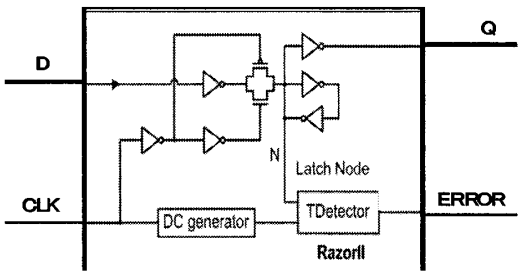

図1: RazorII

2.2 RazorII（回復）

RazorIIを利用した回復方法を説明する．

この方法ではパイプラインを，動的なTFが起こってもよいSpeculative領域と，起きてはいけないNon-speculative領域とに分ける（図2）．RazorIIのエラー・ビットは各ステージでorされて，次のステージへと伝わる．動的なTFが起こると，エラーの通知がパイプラインに沿って流れていく．パイプラインに沿って伝わることでTFの影響よりもその通知の方が必ず先にコミットステージに届く．つまり，誤ったデータがコミットされることはない．

TFの通知がコミットステージに届くと，プロセッサは動作を停止する．パイプラインをフラッシュし，正しいと保証されているアーキテクチャ・ステート（最後にコミットされた状態の論理レジスタファイル（LRF）とＰＣ）より命令を再開する．この再実行の機構自体は，分岐予測ミスに対してもともとプロセッサに備わっているので，新たに準備する必要はない．

2.3 RazorIIの回復方法の問題点

この方法は制御系のパスに起こる動的なTFを考慮していない．たとえばOut-of-Orderスーパスカラ・プロセッサには多数のFIFOのバッファが使われており，こ

†Satoshi ARIMA　‡Takashi OKADA ‡Kazuo HORIO　‡Takanobu KITA ‡Ryota SHIOYA ‡Masahiro GOSHIMA　‡Shuichi SAKAI
†Dept. of EEIC Eng, the Univ. of Tokyo
‡Dept. of Information and Communication Eng, the Univ. of Tokyo

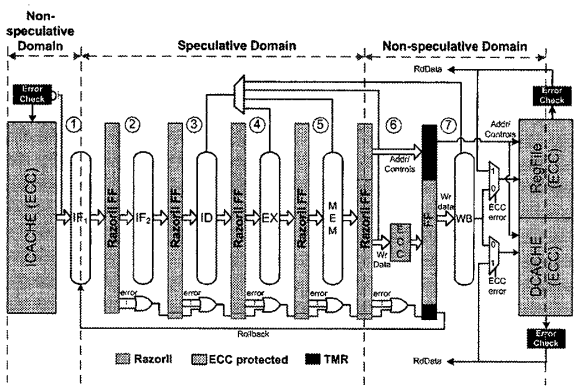

図 2: RazorII を利用する, 動的な TF からの回復

の制御系のポインタに動的な TF が起こる可能性があるが, これはパイプラインフラッシュでは回復できない.

3 提案手法

制御系のパス上での動的な TF を考慮にいれた回復方法として, 当研究室はリセットによる回復[2] を提案している.

3.1 リセットによる回復手法

基本的な考え方は 2.2 項の方法と同じであるが, 再実行前の処理が異なる. 2.2 項の方法ではパイプラインをフラッシュしていたのに対し, 提案手法ではアーキテクチャ・ステートを残してプロセッサをリセットする. 正しいアーキテクチャ・ステートがあれば, 論理的には命令の再実行を開始することができる. リセットによって, プロセッサは一貫性のとれた初期状態に戻っているため, 制御系の動的な TF にも対応している. また, この回復手法は一般に過渡故障に対応できる.

3.2 改良・検討

[2] では, ロード/ストアキュー（LSQ）での動的な TF を考慮していない点が問題である. LSQ は物理的に 1 つのモジュール内部, にコミットしているストア命令とそうでないストア命令をもつ. そのため, 内部で動的な TF が起きてしまうと, コミットされた状態を確実に保持することが困難である. また, リセットも単純にはできない.

そこで, 本研究ではコミット周辺のステージを整理し, コミットされた誤りのない状態を LSQ から分離することを考える（図 3）. 実行が終了した命令はリオーダバッファ（ROB）からインオーダで次のステージ（単なるバッファ）に進む（完了）. ストア命令も対応して LSQ から先へ進む. 完了ステージ以降は動的な TF が起こらないように設計する. これで動的な TF の影響がコミットされないことを保証する. ストア命令がコミットされる時には, 同時に対応するストア要求が BUF へ格納される. その後, キャッシュポートが利用可能な時にデータキャッシュ（D$）へ書き込まれる.

なお, 完了ステージ以降のバッファ内にある値は参照されない. それらの値は LSQ や ROB が保持する. 参照の機能のためにバッファが複雑化することを避けるためである. D$や LRF へ書き込みが完了した命令はリタイアし, LSQ や ROB からエントリが解放される.

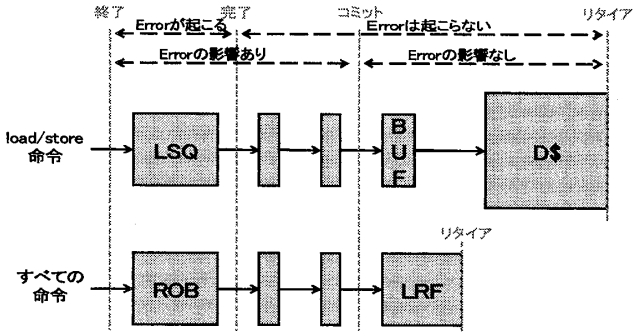

図 3: コミット周辺ステージの整理

4 まとめ

本研究は動的な TF の回復技術に関するものである.

RazorII の方法では回復できない制御系のパス上での動的な TF まで対応できる, リセットによる回復手法を提案した. 本稿では, そのうちコミット周辺のステージを整理し, これまでに考慮されていなかったストア命令のコミットを明確にし, その前後を物理的に分離した.

現段階ではモデルでしかないため, 実際に設計する際のハードウェア量の検討や, その他プロセッサ性能へ与える影響の考察が今後の課題である. 問題がなければ将来的に実装・評価まで考えたい.

参考文献

[1] S. Das, C. Tokunaga, S. Pant, W. H. Ma, S. Kalaiselvan, K. Lai, D. M. Bull, and D. T. Blaauw. Razorii: In situ error detection and correction for pvt and ser tolerance. *Solid-State Circuits, IEEE Journal of*, Vol. 44, No. 1, pp. 32–48, 2009.

[2] 杉本健. タイミング・フォルト耐性を持つスーパスカラ・プロセッサ. 東京大学修士論文, 2009.