社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2005－ARC－165（14）
2005／12／1

めりはり型実行モデルに基づくアーキテクチャ

近藤正章[†]　中村宏[†]

近年，リーク電流による消費電力の増加が問題となっている．本稿では，特に実行時のリーク電流を削減するべく，めりはり型実行モデルに基づくアーキテクチャの重要性を主張する．めりはり型実行とは，プロセッサにおける処理を空間的・時間的に閉じ込め，処理を行う際はできるだけ一度に大量の処理を，またストール時にはできるだけ長い間ストールする，というように「めりはり」のついた実行の振舞いを意味する．めりはり型実行により，細粒度の電源電圧供給制御が効率的に行えるため，大幅なリーク電流削減が期待できる．本稿では，めりはり型実行の概要を説明し，それを実現する上でのアーキテクチャの課題について述べる．

A New Architecture Based on Fully-Modulated Execution Model

Masaaki Kondo[†] and Hiroshi Nakamura[†]

As semiconductor technology scales down, the leakage power becomes dominant in the total power consumption of LSI chips. To reduce runtime leakage power, we propose a new architecture based on fully-modulated execution model. Fully-modulated execution indicates behavior where sets of processing are put into temporally and spatially packed regions. This feature allows fine grain power supply control, and thereby leakage power can be reduced effectively. This paper described the overview of the fully-modulated execution and future work for an architectural implementation.

1. はじめに

近年，消費電力・消費エネルギーの削減は，LSI を設計する上での最も重要な課題となっている．モバイル計算機のバッテリ駆動時間の延長という要求はもちろんのこと，ハイエンドプロセッサにおいても，放熱の問題から消費電力削減は必要不可欠である．LSI チップの消費電力には，トランジスタのスイッチングによるダイナミック消費電力と，リーク電流によるリーク消費電力があるが，半導体プロセスの微細化によりリーク消費電力が増大し，今後はチップ全体の消費電力の半分以上を占めるとさえ予測されている．そのため，リーク電流を削減するための技術開発が早急の課題となっている．

従来より，特にモバイル用途のプロセッサにおいて，待機時やアイドル時のリーク電流を削減するための手法は多く提案されており，それらを用いることで大幅にリーク消費電力を削減することができる．しかし，将来的なリーク電流増大を考えると，待機時やアイドル時だけでなく，アプリケーション実行中のリーク消費電力も無視することはできない．したがって，性能低下なく実行時リーク電流を削減するための手法が必要となる．

これまでにも，キャッシュにおける実行時のリーク電流を削減する手法は多く提案されている[2),3)]．プロセッサでは，トランジスタの多くがキャッシュに費やされており，またリーク消費電力はトランジスタ数に比例して増大することから，キャッシュのリーク電流を抑えることはチップ全体の消費電力削減に有効である．一方，文献[1)] のリーク消費電力モデルによると，演算器などの組み合わせ回路は，トランジスタ数は少ないものの特性が異なるため，トランジスタあたりのリーク消費電力が大きく，無視することはできないと考えられる．したがって，キャッシュのみならず，演算器を含めたプロセッサ全体のリーク消費電力削減を考えることが重要である．

リーク電流を削減するための回路手法としては，しきい値電圧を上げる，あるいは電源電圧の供給を停止する (Vdd ゲーティング) などが存在するが，一般的にそれらの手法はリーク電流削減と，スイッチング速度の低下，あるいは動作や情報の保持が不可になるという間のトレードオフがある．したがって，性能低下なく実行時リーク電流を削減するためには，通常の動作を行うモード (高リークモード) と，リーク電流を削減するためのモード (低リークモード) を各ユニットの実行状況に合わせて動的に切り替えつつ実行を行う必要がある．

効率的な実行時リーク電流削減のためには，時間的・

† 東京大学 先端科学技術研究センター
Research Center for Advanced Science and Technology, The University of Tokyo

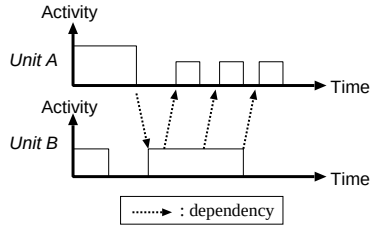

図 1 従来の実行

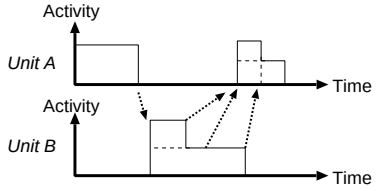

図 2 めりはり型実行

空間的に細粒度に電源電圧供給制御を行うことが望ましく，オーバヘッドが小さな Vdd ゲーティング手法が有望であるが，それでもモードの切り替え時にはある程度のオーバヘッドが生じる．そこで本稿では，Vdd ゲーティング手法により効率的にリーク消費電力を削減するべく「めりはり型実行モデルに基づくアーキテクチャ」の重要性を主張する．めりはり型実行とは，処理を空間的・時間的に閉じ込め，処理を行う際はできるだけ一度に大量の処理を，またストール時にはできるだけ長い間ストールする，というような「めりはり」のついた実行の振舞いを意味する．本稿では，めりはり型実行となるよう処理の制御を行うプロセッサアーキテクチャについて検討する．

2. めりはり型実行モデル

2.1 従来のプロセッサにおける実行

従来のマイクロプロセッサでは，命令レベルの並列性抽出技術，あるいはレーテンシ隠蔽技術に代表されるように，できる限り暇なユニットが存在しないように，また各命令を早期に実行可能状態にし，かつ実行可能な処理はできる限りすぐに実行されるようにアーキテクチャ的な工夫や命令スケジューリングが行われている．

図 1 は従来のプロセッサの実行の様子を示したものである．図は時間の経過にともなう 2 つのユニットの稼働率を表し，ユニット間の矢印は処理の依存関係を表している．従来型の実行では，Unit-B の命令実行に依存する Unit-A の命令は，依存関係が解決次第すぐに実行される．そのため，もし Unit-A で処理すべき命令が十分にない場合，すなわち命令レベル並列度が十分でない場合,「実行」と「ストール」フェーズをを短い時間間隔で繰り返すことになる．このように短い間隔で実行とストールが繰り返される場合，ストール時に Unit-A の電源供給を停止することは難しいと考えられる．

2.2 めりはり型実行

従来型の実行では，時間の経過にともない各ユニットが常に少しずつ処理を行っていくのに対し，めりはり型実行ではできる限り処理が時間的・空間的に集中するように制御を行う．

図 2 は，めりはり型実行を行うプロセッサでの実行の様子を示したものである．めりはり型実行の場合は，依存が解決し実行の準備ができた命令もあえて実行せず，後でまとめて実行する．これにより，処理とストールのフェーズをはっきりと区別することができるようになる．ストール時に Unit-A や Unit-B の電源供給を停止することを考えると，めりはり型実行では，より長い期間電源供給を停止することができるため，リーク電流の大幅な削減が期待できる．また，高リーク／低リークモードの切り替え回数も少なくてすみ，モード切り替えにともなうオーバヘッドの影響も小さくなる．

3. めりはり型実行アーキテクチャの課題

前節では，めりはり型実行モデルによりリーク消費電力を効率的に削減できる可能性があることを述べた．しかし，めりはり型実行モデルを実現する命令実行制御方式をを考える場合，アーキテクチャとして解決すべき課題がいくつか存在する．特に，

- 後回しにするべき命令の選択
- 後回しにした命令の実行再開のタイミング
- できる限り多くの処理をまとめて行うための命令実行方式

などについて検討しなければならない．また，処理を後回しすることは性能低下に結び付きやすく，いかに性能低下なくめりはり型実行を行えるかが重要となる．さらに，電源電圧供給制御行う空間的な粒度，すなわち Vdd ゲーティングを行うユニットの単位も重要な検討課題である．

今後のプロセス微細化にともなうリーク電流増大を考えると，実行時リーク消費電力削減は取り組むべき重要な問題であり，めりはり型実行モデルに基づくアーキテクチャはその解決に向けた一つの候補として十分に期待できるものであると考えられる．今後は，上記の課題を軸に，効率的にリーク消費電力を削減できるアーキテクチャを検討していく予定である．

参 考 文 献

1) J.AButts and G.S.Sohi, “A static power model for architects” *In Proc. the 33rd MICRO*, pp.191–201, 2000.
2) S.Kaxiras, et al., “Cache decay: exploiting generational behavior to reduce cache leakage power”, *In Proc. the 28th ISCA*, pp.240–251, 2001.
3) K.Flautner, et al., “Drowsy caches: simple techniques for reducing leakage power” *In Proc. the 29th ISCA*, pp.148–157, 2002.