VT-CMOSキャッシュの性能低下をアドレス予測を用いて低減する先行起動機構

小林 良太郎[†] 藤岡 涼[†,☆]
安藤 秀樹[†] 島田 俊夫[†]

近年，CMOS回路の低電圧化が進むにつれて，従来は無視できるほど小さかったリーク電流が急速に増大している．本論文ではキャッシュをVT-CMOSで構成し，電力制御をセット単位で行うDLC（Dynamic Leakage Cut-off）と呼ばれる手法において，性能低下を抑制する方式を提案する．DLCは，リーク電流を大幅に削減できるが，選択されたセットを動作状態に活性化するために大きな遅延を生じ，その結果性能低下を引き起こす．これに対し本方式は，アドレス予測により，参照されるセットを予測し，参照前に活性化を行うことで遅延を隠蔽し，性能低下を緩和する．SPEC2000ベンチマークを用いて性能を測定したところ，32KB，2-way構成において従来のDLCキャッシュでは平均で，SPECint2000では17.2%，SPECfp2000では5.8%の性能低下を引き起こしていたが，我々の機構を使うことによりそれぞれ，8.9%，1.1%にまで抑制することができた．また，静的消費電力はDLCキャッシュと比べるとわずかに増加するものの，依然としてDLCを用いない通常のキャッシュが消費する電力の1%にまで削減することができる．先行起動機構の動的消費電力によって全消費電力が増加するが，我々の機構は，通常のキャッシュが消費する全消費電力に対し，全消費電力を70nmプロセスでは23.7%，35nmプロセスでは6.9%にまで抑制することができる．

A Preactivating Mechanism for Suppressing the Performance Degradation in a VT-CMOS Cache using Address Prediction

Ryotaro Kobayashi,[†] Ryo Fujioka,[†,☆] Hideki Ando[†]
and Toshio Shimada[†]

As the supply voltage of CMOS circuits goes low, the leakage current, which has been negligibly small, rapidly increases. This paper proposes a mechanism that suppresses performance degradation when the dynamic leakage cut-off (DLC) scheme, which controls power consumption by the set, is adopted to a VT-CMOS cache. Although the DLC significantly reduces leakage current, it causes a long delay to activate circuits in the selected set, leading performance degradation. Our mechanism predicts a reference set with address prediction and activates the set in advance. As a result, the delay is hidden and performance degradation is alleviated. Our results show that, in a 2-way, 32KB cache, our mechanism can suppress performance loss to 8.9% and 1.1% on average for SPECint2000 benchmark and SPECfp2000 benchmark, respectively, while the conventional DLC cache degrades performance by 17.2% and 5.8%, respectively. The results also show that our mechanism slightly increases the leakage power over the DLC cache, but can still reduce the static power to approximately 1% of that of the usual non-DLC cache. While the dynamic power of our mechanism increases the total power, our mechanism can reduce the total power to approximately 23.7% and 6.9% of the usual non-DLC cache in 70nm and 35nm process technology, respectively.

1. はじめに

近年では，マイクロプロセッサを設計する上での重要な要素として，性能やコストの他に消費電力があげられる．これまで，消費電力削減の主要な目的は，電子携帯機器のバッテリー駆動時間の延長であった．しかし，デスクトップやサーバに用いられる最近の高性能マイクロプロセッサでは，チップの電力密度が許容不能なレベルに近づき，電力消費を抑えることが非常に重要な事項となって来た[15]．

CMOS回路で消費される電力は，負荷容量の充放電により消費される動的消費電力とトランジスタの

† 名古屋大学大学院工学研究科
Graduate School of Engineering, Nagoya University
☆ 現在，日本電気株式会社
Presently with NEC Corporation

リーク電流によって消費される静的消費電力に分類できる．従来，動的消費電力が消費電力の大部分を占めていた．一方，静的消費電力の中で最も大きなサブスレッショルド電流は，トランジスタの閾値電圧の低下に対し，急速に増大することが知られている．このため，消費電力に占める静的消費電力の割合は近年急速に増加し，現在では無視することができないレベルに達している．今後，この状況はさらに悪化していくため，何らかの対策が必要とされている．

これに対し，リーク電流を削減する種々の回路技術が研究されている．代表的なものとして **VT-CMOS**（Variable-Threshold CMOS）[10] と **MT-CMOS**（Multi-Threshold CMOS）[14] が提案されている．これらの技術はいずれも回路を待機状態にすることにより，リーク電流を大幅に削減できる．一方で，回路を待機状態から動作状態にする活性化時間は長く，性能に悪影響を与えるという欠点がある．このため，これまでは活性化の頻度が低いブロック，つまり，使用頻度の低いブロックに対してこれらの技術を適用することが考えられていた．しかし，これは明らかに大きな制約である．プロセッサのリーク電流を大幅に削減するためには，使用頻度の高いブロックに対し，性能に与える悪影響を最小化しつつ，電力を制御する方式を見いだすことが求められている．

使用頻度の高いブロックのうち，チップ上で最も多くのトランジスタを使用しているキャッシュは，リーク電流削減の対象として非常に重要である．これに対し，キャッシュを VT-CMOS で構成し，セットを最小単位として電力制御を行う **DLC**（Dynamic Leakage Cut-off）と呼ばれる手法が提案されている[8]．DLC はアドレス・デコード後に，選択されたセットのみを動作状態にする．これにより，リーク電流を大幅に削減することができるが，長い活性化時間により，キャッシュのアクセス時間を大幅に延ばしてしまう．

本論文では，キャッシュに DLC を適用し，かつ，アーキテクチャレベルでプロセッサの性能低下を抑制する**先行起動機構**[5),6)] を提案する．提案機構では，アドレス予測を用いて，アクセスされるキャッシュセットを予測し，そのセットを実際のアクセスよりも前の段階で動作状態にしておく．こうすることにより，VT-CMOS の活性化時間による遅延を隠蔽することができる．

以下第 2 章では本研究で提案する機構について述べ，第 3 章で評価を行う．第 4 章で関連研究について述べ，第 5 章でまとめる．

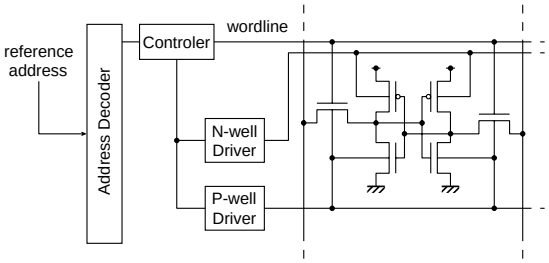


図 1 従来の DLC キャッシュのセット構成
Fig. 1 Structure of a set in the original DLC cache.

2. DLC キャッシュにおける先行起動機構

本節では，まず従来の DLC キャッシュについて述べ，次に先行起動機構で用いるアドレス予測について述べる．最後に DLC キャッシュにおける先行起動機構について説明する．

2.1 従来の DLC キャッシュ

キャッシュは，複数のセットを持ち，セット内にデータを保持する．セットは複数の SRAM セルで構成され，1 つの SRAM セルは 1 ビットのデータを記憶する．キャッシュでは，参照アドレスが入力されると，アドレスデコーダによって参照アドレスに対応する 1 つのセットを選択し，そのセットに保持されているデータを読み出す．

本論文では，キャッシュに対し，セットを最小単位として電力制御を行うとする．キャッシュでは，セットを最小単位としてデータの参照を行う．電力制御の最小単位を，データを参照する制御と同じにすれば，ハードウェアをより容易に実現することができる．

図 1 に従来の DLC キャッシュのセット構成を示す[8]．DLC キャッシュにおいて，セットは Controler，N-well Driver，P-well Driver，複数の SRAM セルを持ち，SRAM セルは VT-CMOS で構成される．SRAM セルの状態は，閾値電圧の高い待機状態と閾値電圧の低い動作状態の 2 つがある．SRAM セルを待機状態にすることにより，リーク電流を削減することができる．

DLC キャッシュの動作を説明する．初期状態では，全てのセルを待機状態とする．アドレスデコーダによりセットが選択されると，Controler は N-well Driver，P-well Driver により，そのセットの SRAM セルを動作状態へ活性化し，その直後にワードラインを立ち上げる．データが読み出された後，それらのセルを再び待機状態に戻す．

DLC キャッシュでは，アクセスされるセットのセルのみを動作状態にするので，キャッシュサイズが大きい場合，ほとんどのセットが待機状態となり，リーク

電流を大幅に削減することができる．しかし，通常のキャッシュに比べて，セットを動作状態へ活性化するための遅延時間が余分に必要となる．そのため，DLC キャッシュのアクセス時間は通常のキャッシュよりも長くなり，性能低下を引き起こす原因となる．

2.2 アドレス予測

本研究で利用するアドレス予測について説明する．メモリ命令の参照アドレスをアドレス計算の前に予測することをアドレス予測という．予測には通常，値予測が用いられる．値予測には最終値予測[11),12)]，ストライド予測[24)]，コンテキスト・ベース予測[19)] 等がある．

一般に，それぞれの予測器はメモリ命令毎に参照されたアドレスの履歴を記録する単一または複数のテーブルから構成されている．それらのテーブルに記録された履歴を利用して予測は行われる．このうち，ストライド予測は一定の差分で値が出現するような性質を利用して予測を行う手法である．メモリ命令の参照アドレスは一定の差分で出現することが多く，コスト性能比では，上記のアドレス予測手法の中でもストライド値予測が良い[16)]．一方，コンテキスト・ベース予測は将来の実行結果は過去数回の実行結果のいずれかである確率が高いという性質を利用する手法である．ストライド予測とコンテキスト・ベース予測は予測可能な値の出現パターンが異なるので，これらを組み合わせたハイブリッド予測[24)] はより高い予測精度を得ることができる．

本研究では，ストライド予測を用いた場合と，ストライド予測とコンテキスト・ベース予測を組み合わせたハイブリッド予測を用いた場合について評価する．

2.3 先行起動機構

主記憶とプロセッサの速度差は非常に大きく，このギャップを埋めるため，近年ではキャッシュは複数に階層化されている．一般に下位のキャッシュほど容量が大きいので，DLC の適用による消費電力の削減効果は大きい．一方，L2 以下の階層のキャッシュのアクセス頻度は，一般に非常に小さく，短レイテンシへの要求は小さい．このため，DLC の活性化時間が性能に与える悪影響は小さく，問題がない．これに対して，最上位の L1 キャッシュに DLC を適用した場合は，下位のキャッシュほどの消費電力の削減効果はないものの，使用されているトランジスタは依然として非常に多いので，効果はやはり大きいと考えられる．しかし，下位と異なりアクセス頻度は非常に高く，かつ，短レイテンシへの要求も強いので，DLC の起動時間が性能へ与える悪影響は非常に大きいと考えられる．本論文では，この問題を解決するための方式を提案する．

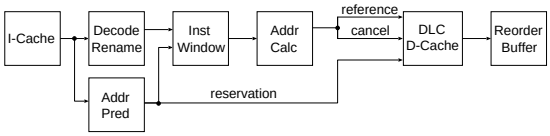


図 2 先行起動機構を備えたプロセッサの構成
Fig. 2 Processor organization with our preactivating mechanism.

図 2 に我々の提案する先行起動機構を備えたプロセッサの構成を示す．通常のスーパースカラプロセッサにおいて，L1 データキャッシュに DLC を適用し，先行起動を行うための機構を追加している．具体的には，まず，アクセスされるセットを予測するために，アドレス予測器を用意する．次に，予測したアドレスを保持するために，命令ウインドウの各エントリに新たなフィールドを設ける．そして，アドレス予測結果を用いて電力を制御するために，データキャッシュへ参照（reference），取消（cancel），予約（reservation）という 3 つのアドレスを与える．

図 3 に提案機構における DLC キャッシュのセット構成を示す．従来の DLC キャッシュに，2 つのアドレスデコーダと**予約カウンタ**と呼ばれるアップダウン・カウンタを追加する．予約カウンタは，各セットに 1 つずつ存在し，対応するセットへのアクセスが予測される回数を記録する．予約カウンタの値は，対応するセットが予約アドレスによって選択されたときに 1 増加させ，取消アドレスによって選択されたときに 1 減少させる．また，分岐予測ミスが判明したときには，全ての予約カウンタを 0 にリセットする．これは，分岐予測ミスが生じると予約カウンタの値を増加させたが取り消しを行っていないメモリ命令の多くがフラッシュされるからである．フラッシュされないメモリ命令も存在し，予約カウンタが 0 になることで，DLC 起動のペナルティを被るが，そのようなメモリ命令は少なく，かつ，分岐予測ミスの頻度は低いので，問題ない．

提案機構の動作を以下に示す．命令デコード時に，アドレス予測器を用いて，メモリ命令の参照アドレスを予測する．そして，命令ウインドウの当該命令を挿入するエントリに，予測結果を書き込む．また，予測結果を予約アドレスとしてデータキャッシュに送信する．データキャッシュでは，予約アドレスをデコードし，選択されたセットの予約カウンタを 1 増加させる．カウンタ値が 0 から 1 へ増加するときに，対応するセットを動作状態に活性化し，カウンタ値が 1 以上である間，動作状態を保持する．

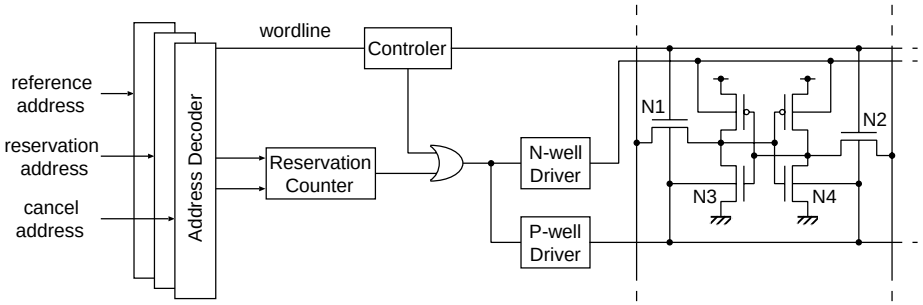


図 3 提案機構における DLC キャッシュのセット構成
Fig. 3 Structure of a set in our proposed mechanism.

数サイクル後，メモリ命令が命令ウインドウから発行され，実効アドレスが計算される．計算された実効アドレスを参照アドレスとして，また，命令ウインドウより読み出された予測アドレスを取消アドレスとしてデータキャッシュに入力する．データキャッシュでは，入力された参照アドレスにより，選択されたセットのデータを読み出す．デコード時に行ったセット予測が正しく，参照アドレスにより選択されたセットが既に動作状態にあるなら，活性化による遅延は発生しない．そうでなければ，従来の DLC キャッシュと同様に動作し，活性化による遅延が発生する．一方，データキャッシュに入力された取消アドレスにより，選択されたセットの予約カウンタを 1 減少させる．カウンタ値が 0 になれば，対応するセットを待機状態にする．

3. 評　　価

最初に，DLC キャッシュにおける活性化の遅延時間を評価する．次に，プロセッサ性能について述べる．そして，静的消費電力の削減効果と先行起動機構の動的消費電力の影響について評価する．最後に，キャッシュの構成を変化させた場合の先行起動機構の有効性について述べる．

3.1 DLC キャッシュにおける活性化の遅延時間

DLC キャッシュにおいて，活性化の遅延時間がアクセス時間に与える影響について評価する．評価環境を以下に示す．キャッシュの構成は，32K バイト，32 バイト・ブロック，2 ウェイ・セットアソシアティブとした．**表 1** に，評価のために仮定した SRAM セルのパラメータを示す．表において λ とは，最小加工寸法の 2 分の 1 のサイズを表す単位である．また，N1 〜 N4 は，それぞれ図 3 に示した SRAM セルのトランジスタ N1〜N4 である．

評価では，通常キャッシュと DLC キャッシュのアクセス時間を測定するために，CACTI[17),18),21),26)] を修正したシミュレータを使用した．CACTI とは，キャッシュサイズ，連想度，ブロックサイズを入力とし，最適なアレイの分割方法，その時のアクセス時間，消費エネルギーを計算するプログラムである．CACTI では，まず，キャッシュの回路を基本的なゲートと RC 等価回路によってモデル化する．ゲート遅延については，Horowitz の遅延モデル[7)] に基づいて解析的に計算する．消費エネルギーについては，電源電圧の 2 乗と負荷容量の積で求める．分割の最適化は，次のようにして行われる．与えられたキャッシュ構成の下で，アレイの分割方法の全ての場合について，アクセス時間，消費エネルギーを計算する．そして，これらの計算結果をパラメータとして，分割の良さを表す評価関数が最善の値となる分割方法を採用する．

表 1 SRAM セルのパラメータ
Table 1 Parameters for an SRAM cell.

トランジスタ N1，N2 の幅	6λ
トランジスタ N3，N4 の幅	12λ
セルの高さ	40λ
セルの幅	20λ
セルあたりの p ウェルの高さ	40×4/6λ
セルあたりの p ウェルの幅	20λ

DLC キャッシュにおける活性化の遅延時間とは，p ウェルを駆動する時間である．p ウェルの総駆動容量は，駆動ドライバから p ウェルまでの配線容量と，p ウェルの p-n 接合部の容量の合計である．p-n 接合部の容量とは p ウェルー n ウェル容量と p ウェルー n^+ 容量の和である．評価では，CACTI に修正を加え，ワードラインの遅延時間の測定と同様の方法を用いて，p ウェルを駆動する時間を測定した．

評価のために仮定したトランジスタと配線のプロセス・パラメータを，それぞれ**表 2** と**表 3** に示す．プロセス・パラメータのうち，電源電圧は文献 20) に示されたものを使用した．トランジスタに関しては，CACTI で仮定されているパラメータを，プロセスに対してスケーリング[25)] したものを使用した．配線に

表 2 各プロセスでのパラメータ（トランジスタ）
Table 2 Parameters for each process (Transistor).

Process (nm)	Gate Cap. (fF/μm)		pwell-n$^+$ Cap.		pwell-nwell Cap.		Transistor Res. (KΩ-μm)		Supply Voltage (V)
	cell	pass	(fF/μm^2) Area	(fF/μm) Line	(fF/μm^2) Area	(fF/μm) Line	Nch	Pch	
70	1.56	1.16	1.56	0.27	2.57	0.10	1.4	3.3	0.7
50	1.56	1.16	2.19	0.27	3.60	0.10	0.8	2.0	0.6
35	1.56	1.16	3.13	0.27	5.14	0.10	0.5	1.2	0.5

表 3 各プロセスでのパラメータ（配線）
Table 3 Parameters for each process (Wire).

Process (nm)	Width (nm)	Res. (mΩ/μm)	Cap. (fF/μm)
70	120	500	0.278
50	80	1020	0.294
35	60	1760	0.300

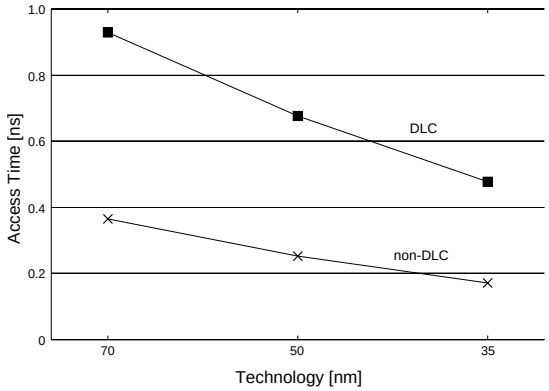


図 4 アクセス時間
Fig. 4 Access time.

関しては，文献 1) に示されたものを使用した．

図 4 に各プロセスにおける，DLC キャッシュと通常キャッシュ（non-DLC）のアクセス時間を示す．図において，活性化の遅延時間は，DLC キャッシュと通常キャッシュのアクセス時間の差である．図から，DLC キャッシュは活性化時間が長く，アクセス時間を大幅に延ばしてしまうことが分かる．通常キャッシュに比べ，DLC キャッシュのアクセス時間は 2.6 倍〜2.8 倍となる．

3.2 プロセッサ性能

まず，評価環境について述べる．次に，セット予測精度を示す．最後に，先行起動機構による性能改善について評価する．

3.2.1 評 価 環 境

SimpleScalar Tool Set[2] のスーパスカラ・プロセッサ用シミュレータに本機構を組み込み測定した．命令セットは MIPS R10000[13] を拡張した SimpleScalar/PISA である．ベンチマーク・プログラムは SPEC2000 の 16 種類を使用した．ベンチマーク・プログラムのバイナリは，GNU GCC version 2.7.2.3（コンパイルオプション: -O6 -funroll-loops）を用いて作成した．**表 4** にそれぞれのベンチマーク・プログラムの入力セット，実行命令数を示す．

表 4 ベンチマーク・プログラム
Table 4 Benchmark programs.

	プログラム	入力	実行命令数
SPEC int2000	bzip2	input.source 2	1645M
	gcc	cccp.i	1586M
	gzip	input.log 2	1010M
	mcf	inp.in	173M
	parser	test.in	224M
	perlbmk	scrabbl.in	218M
	vortex	lendian.raw	810M
	vpr	place.in	709M
SPEC fp2000	ammp	ammp.in	3923M
	applu	applu.in	535M
	apsi	apsi.in	812M
	art	c756hel.in	1721M
	equake	inp.in	679M
	mesa	mesa.in	307M
	mgrid	mgrid.in	577M
	swim	swim.in	891M

以下のモデルについて評価した．

- base モデル
- conventional モデル
- preactivating モデル
- simple モデル

base モデルは通常のキャッシュを備えたモデルである．conventional モデルは従来の DLC キャッシュを備えており，キャッシュを参照するときに，常に活性化による遅延が生じるモデルである．一方，preactivating モデルは，DLC キャッシュに加え先行起動機構を備えているモデルである．このモデルでは，セットが正しく予測できれば，活性化による遅延は生じない．simple モデルは，DLC キャッシュにおいて，セットの状態を制御する方式として，文献 4) で提案されているシンプル方式を用いるモデルである．このモデルでは，最初に全てのセットを待機状態にする．待機

状態のセットが参照されるときに当該セットを動作状態にし，その状態を保つ．そして，一定の間隔で全てのセットを待機状態にする．

表 5 に，各評価モデルに共通するプロセッサ構成を示す．本論文では，各プロセスにおいて base モデルの備える通常キャッシュのアクセス時間をサイクル時間として，各モデルのデータキャッシュのヒットレイテンシを定めた．base モデルの場合，データキャッシュのヒットレイテンシは 1 サイクルとした．conventional モデルの場合，3.1 節の結果より，3 サイクルとした．一方，preactivating モデルの場合，セットの予測が正しければ 1 サイクル，そうでなければ 3 サイクルとした．また，simple モデルの場合，参照したセットが動作状態であれば 1 サイクル，そうでなければ 3 サイクルとした．conventional モデル，preactivating モデル，simple モデルにおいて，データキャッシュへのアクセスに複数サイクルを要するとき，そのアクセスの間であっても，それに後続するアクセス要求に応えることができるとした．preactivating モデルにおいて，予約カウンタのビット数を 6 ビットとした．取消アドレスは，パイプラインにおいて，メモリ命令が命令ウインドウから発行されるときに命令ウインドウから実効アドレスを計算する機能ユニットに渡され，実効アドレスが計算された後に機能ユニットからデータキャッシュに入力される．simple モデルにおいて，全てのセットを待機状態にする間隔（update window size）を 1000，2000，4000 サイクルとし，それぞれの場合について測定した．**表 6** に各ベンチマーク・プログラムにおける base モデルの IPC を示す．

preactivating モデルでは，2 つのアドレス予測手法について評価した．コスト性能比の高い手法としてストライド予測を用い，性能の高い手法としてハイブリッド予測を用いた．ストライド予測には，VHT（Value History Table）のサイズが 1K エントリのストライド予測器を使用した．ストライド予測器において，ストライドを検出していれば予測を行い，そうでなければ予測を行わないとした．ハイブリッド予測には，VHT のサイズが 1K エントリのストライド予測器と，VHT のサイズが 1K エントリ，値 4，履歴 6，PHT（Pattern History Table）のサイズが 4K エントリ，飽和カウンタの最大値が 12 のコンテキスト・ベース予測器とのハイブリッド予測器を使用した[24]．ハイブリッド予測器において，2 つの予測器の内どちらか一方の予測結果を選択する方法は，文献 24) に示されたものを使用した．この方法では，まず，コンテキスト・ベース予測器の PHT のカウンタ値に閾値を設定する．そして，予測時に参照した PHT のカウンタ値がこの閾値を越えていれば，コンテキスト・ベース予測器の結果を選択する．カウンタ値が閾値を越えておらず，かつ，ストライドを検出していれば，ストライド予測器の結果を選択する．上記のいずれでもなければ，予測を行わない．閾値の値は文献 24) より 6 とした．

表 5 プロセッサ構成
Table 5 Processor configurations.

パイプライン段数	7 段
命令デコード幅	最大 8 命令
命令発行幅	最大 8 命令
命令コミット幅	最大 8 命令
命令ウインドウ	RUU(Register Update Unit) 128 エントリ，LSQ(Load/Store Queue) 64 エントリ
機能ユニット数	iALU 8，iMULT/DIV 2，fpALU 4，fpMULT/DIV/SQRT 2
命令キャッシュ	完全，ヒットレイテンシ 1 サイクル
データキャッシュ	32KB，2 ウェイ・セットアソシアティブ，32 バイト・ブロック，4 ポート，ミスペナルティ6 サイクル
2 次キャッシュ	2MB，4 ウェイ・セットアソシアティブ，64 バイト・ブロック，ミスペナルティ36 サイクル
分岐予測機構	BTB 2048 エントリ，4 ウェイ・セットアソシアティブ，gshare 6 ビット履歴 8K エントリ PHT，RAS(Return Address Stack) 16 エントリ，分岐予測ミスペナルティ5 サイクル

表 6 base モデルの IPC
Table 6 IPC of the base model.

	プログラム	IPC
SPEC int2000	bzip2	3.04
	gcc	2.60
	gzip	2.14
	mcf	2.44
	parser	2.81
	perlbmk	2.49
	vortex	4.51
	vpr	2.93
SPEC fp2000	ammp	2.47
	applu	4.49
	apsi	3.22
	art	2.94
	equake	2.69
	mesa	4.25
	mgrid	5.02
	swim	2.84

3.2.2 セット予測精度

先行起動方式では，参照するセットを正しく予測できれば，活性化による遅延は発生せず，性能への悪影響を緩和することができる．本節では，preactivating

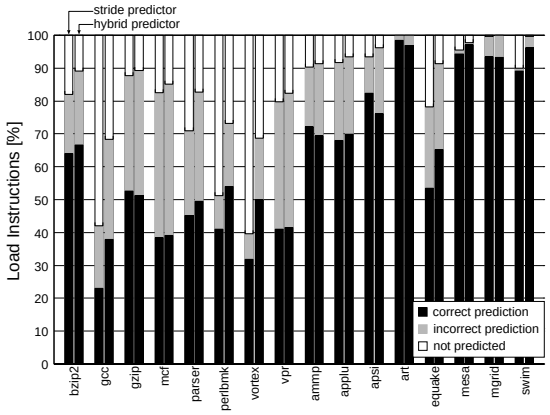


図 5 セット予測精度
Fig. 5 Set prediction accuracy.

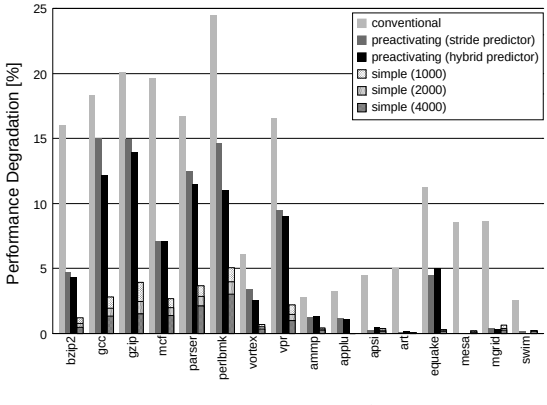


図 6 base モデルに対する性能低下
Fig. 6 Performance degradation from the base model.

モデルにおいて，ストライド予測とハイブリッド予測のセット予測精度を評価した．

図 5 に評価結果を示す．縦軸は全動的ロード命令の予測の正誤における割合を示し，横軸はベンチマークを示す．図には 2 本の棒グラフがあり，左側がストライド予測の場合，右側がハイブリッド予測の場合である．棒グラフは 3 つの部分からなり，下から順に，セット予測精度，セット予測ミス率，予測を行わなかった割合である．ここで，セット予測精度とは，全動的ロード命令数に対するセット予測の正しい命令数の割合である．

図から分かるように，SPECint2000 と SPECfp2000 を比較すると，SPECfp2000 の方がセット予測精度が高い傾向にある．また，多くのベンチマークで，ハイブリッド予測の方がストライド予測よりもセット予測精度が高く，予測精度の差は SPECint2000 の場合の方がより大きい傾向にある．セット予測精度は，ストライド予測の場合 SPECint2000 で平均 42.2%，SPECfp2000 で平均 81.4%，ハイブリッド予測の場合 SPECint2000 で平均 48.7%，SPECfp2000 で平均 83.0%である．ストライド予測，ハイブリッド予測では，それぞれ 3，4 つのベンチマークにおいて，ロード命令の 90%以上の参照先セットを正しく予測することができた．

3.2.3 先行起動機構による性能改善

図 6 に base モデルに対する性能低下率を示す．縦軸は性能低下率を示し，横軸は各ベンチマークを示す．図には 4 本の棒グラフがあり，左から順に conventional モデル，ストライド予測を行う preactivating モデル，ハイブリッド予測を行う preactivating モデル，simple モデルである．simple モデルでは，3 本の棒グラフが重なっており，奥から順に update window size が 1000，2000，4000 サイクルの場合である．

図より，conventional モデルは活性化の遅延により性能が大きく低下していることが分かる．性能低下率は，SPECint2000 で平均 17.2%，SPECfp2000 で平均 5.8% となる．

これに対し，preactivating モデルは，性能低下を大きく抑制することができる．また，セット予測精度が高いほど，性能低下の抑制効果は高い．ストライド予測の場合，性能低下率が SPECint2000 で平均 10.2%，SPECfp2000 で平均 0.9%にまで抑えられる．一方，ハイブリッド予測の場合，性能低下率が SPECint2000 で平均 8.9%，SPECfp2000 で平均 1.1%と，さらに抑制することができる．これは conventional モデルに対しては，SPECint2000 で平均 8.2%ポイント，SPECfp2000 で平均 4.7%ポイントの性能改善である．特に，bzip2，mcf，perl で 11% ポイント以上の改善を示している．

simple モデルは，preactivating よりも更に大きく性能低下を抑制することができる．また，update window size が大きいほど，性能低下の抑制効果は高い．update window size が 1000 サイクルの場合，性能低下率が SPECint2000 で平均 2.8%，SPECfp2000 で平均 0.3%となる．update window size を 4000 サイクルに増加させると，性能低下率が SPECint2000 で平均 1.4%，SPECfp2000 で平均 0.1%に改善する．

3.3 静的消費電力の削減

静的消費電力は，動作状態にあるサイクルあたりの平均セット数に比例する．本節では，これを 3.2.1 節で述べたシミュレーションにより求め，各モデルの静的消費電力を比較する．**図 7** に測定結果を示す．縦軸は base モデルでの静的消費電力を 100%としたと

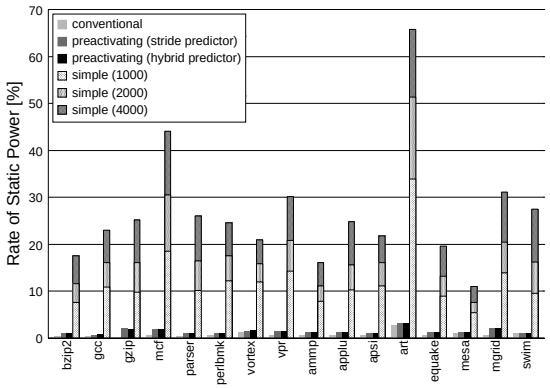


図 7 base モデルに対する静的消費電力
Fig. 7 Static power to the base model.

きのそれぞれのモデルの静的消費電力の割合を示し，横軸はベンチマークを示す．図には 4 本の棒グラフがあり，左から順に conventional モデル，ストライド予測を使った preactivating モデル，ハイブリッド予測を使った preactivating モデル，simple モデルである．simple モデルでは，3 本の棒グラフが重なっており，奥から順に update window size が 4000，2000，1000 サイクルの場合である．

図から分かるように，conventional モデルは base モデルに対し，静的消費電力を SPECint2000 で平均 0.6%，SPECfp2000 で平均 1.0%にまで削減できる．

preactivating モデルは，conventional モデルと同様，静的消費電力を大幅に削減できる．どちらの予測手法の場合も，conventional モデルとの差は平均でわずか 0.8%ポイント以下であり，base モデルに対し静的消費電力を平均で 100 分の 1 にまで削減できる．これは，キャッシュの全セットにおいて，これから参照されると予測される少数のセットだけを活性化しており，無駄に電力を消費することがほとんどないためである．

一方，simple モデルは，他のモデルと異なり，静的消費電力の削減効果が低い．また，図 6 と図 7 より，update window size が大きいほど性能低下の抑制効果はより高くなるが，静的消費電力の削減効果はより低くなる．update window size が 1000 サイクルの場合，base モデルに対して，静的消費電力を SPECint2000 で平均 11.9%，SPECfp2000 で平均 12.6%にしか削減できない．update window size を 4000 サイクルに増加させると，base モデルに対する静的消費電力の削減率は，SPECint2000 で平均 24.6%，SPECfp2000 で平均 28.8%とさらに悪化する．この原因は，simple モデルでは，一度参照されたセットは，参照されなくなった後であっても，全てのセットを待機状態にするまで動作状態を保つことにある．

preactivating モデルにおいて，アドレス予測器自体が消費する静的消費電力について述べる．アドレス予測器は PHT や VHT と呼ばれる SRAM で構成されるテーブルを持つ．これらのテーブルにもキャッシュと同様に DLC を適用すれば，アドレス予測器の静的消費電力は非常に小さくなり，図 7 の結果から容易に推測できるように，十分無視することができる．

DLC を適用したテーブルでは，選択されたセットが待機状態であれば，そのセットを活性化する必要がある．しかし，DLC の活性化に要するレイテンシは，アドレスを予測してからメモリ命令を実行するまでの時間に比べて十分に短いので問題ない．

また，アクセス・レイテンシが増加しても，テーブルに修正を加えることで，アドレス予測器に必要な高いスループットを維持することができる．まず，セットの状態に関わらず，活性化に要する遅延を常に与え，アクセス・レイテンシを一定にする．これにより，予測結果がアウト・オブ・オーダで出力され，後の処理が複雑になることを防ぐ．次に，毎サイクル，アクセス要求を受け付けるようにする．セットごとに，電力制御とデータの参照を行っているので，起動中にない異なるセットへのアクセス要求は独立に受け付けることができる．一方，起動中のセットへのアクセス要求は，すでに受け付けられている要求とマージし，読み出したセットから必要なデータを取り出す論理を加えることで，受け付けることが可能となる．

3.4 先行起動機構の動的消費電力の影響

前節で示したように，従来の DLC キャッシュは大きな性能低下を招くという大きな欠点があるが，先行起動機構を加えることにより，静的電力を小さくできるという長所を維持したまま，欠点を小さくすることができる．しかし，ハードウェアの追加が必要なため，動的消費電力は増加する．これにより，先行起動機構が simple モデルより優れているかどうかは，これまでのところ明らかでない．

一般に，与えられた回路の動的消費電力は，プロセスの進歩に伴いスケーリング・ファクタの 2 乗で急速に減少していく[25]．一方，1 章で述べたように，静的消費電力は，プロセスの進歩に伴い急速に増大する．このため，動的消費電力の増加という先行起動機構の欠点は，将来に向かって小さくなっていき，静的消費電力の大幅な削減という長所は大きくなっていくと考えられる．

本節では，ITRS などが行った将来のプロセスの予

測に基づき，静的消費電力と動的消費電力の両方を加えた全消費電力を求め，simple モデルと比較する．また，プロセッサ性能と消費エネルギーの総合的な評価を行う．

3.4.1 全消費電力の評価

静的消費電力と動的消費電力を加えた全消費電力を評価する．評価する電力は以下の 6 種類に大別できる．

- キャッシュの静的消費電力
- キャッシュの動的消費電力
- 予約アドレスと取消アドレスのデコーダの動的消費電力
- アドレス予測器の動的消費電力
- p ウェルを駆動する回路の動的消費電力
- 命令ウインドウの予測アドレスフィールドの動的消費電力

この他，分岐予測ミス時の予約カウンタのリセットにより消費される動的消費電力があるが，評価に含めていない．この理由は，分岐予測ミスの発生頻度は 3% であり，予約カウンタのリセット回数は非常に低いことに加え，予約カウンタのビット数はキャッシュのビット数の 1%であり，予約カウンタを 1 回リセットするために必要な消費エネルギーは非常に小さいことにある．

キャッシュの静的消費電力については，計算により求めた．その他の消費電力については，3.1 節で述べた CACTI と 3.2 節で述べたスーパスカラ・プロセッサ用シミュレータを用いて求めた．

静的消費電力の計算方法について述べる．一般に，静的消費電力は，単位ゲート幅あたりのリーク電流 I_{leak}，リーク電流が流れるトランジスタのゲート幅，および，電源電圧の積で求められる．I_{leak} については，将来のプロセスに渡り ITRS が予測を行っている．しかし，残念ながらその値は p-n 接合温度が 25 ℃の時の値でしかない．実際にチップが動作している時の p-n 接合温度は 100 ℃程度となるので，電力計算にはその時の I_{leak} が必要である．そこで，次のようにして 100 ℃の時の I_{leak} を求めた．

まず，一般に I_{leak} は，以下の式で表される[23]：

$$I_{leak} = \frac{\mu\, C_{ox}\, (a-1)\, v_t^2}{L}\, exp^{\frac{-V_{th}}{a v_t}} \qquad (1)$$

ここで，μ は移動度，C_{ox} はゲート酸化膜容量，a は定数，$v_t = k \cdot T/q$，k はボルツマン係数，T は絶対温度，q は単位電荷，L はチャネル長，V_{th} は閾値電圧である．μ は $T^{-1.5}$ に比例[22] することから，$\mu = \mu_0 \cdot T^{-1.5}$ と表すことができる．また，サブスレッショルドスロープ S は，$S = a \cdot v_t \cdot \ln 10$ と表すことができる．これらより式 (1) は以下のように変形することができる：

$$I_{leak} = \frac{\mu_0\, T^{-1.5}\, C_{ox} \left(\frac{S}{v_t \ln 10} - 1\right) v_t^2}{L}\, 10^{\frac{-V_{th}}{S}} = \frac{\mu_0 C_{ox}}{L} \left(\frac{S v_t}{\ln 10} - v_t^2\right) T^{-1.5} 10^{\frac{-V_{th}}{S}} \qquad (2)$$

表 7 各プロセスでの閾値電圧，サブスレッショルドスロープ，リーク電流（25 ℃）
Table 7 Threshold voltage, subthreshold slope, and leakage current for each process (25 ℃).

Process (nm)	V_{th} (V)	S (mV/decade)	I_{leak} (μA/μm)
70	0.16	80	1
50	0.13	80	3
35	0.11	75	7

表 7 に，以下の計算のために仮定した 25 ℃における V_{th}，S，I_{leak} を示す．表の V_{th} は文献 3) の結果を外挿して求めた．また，S と I_{leak} は文献 20) より得た．

式 (2) と表 7 に示した値を用いて 100 ℃の場合の I_{leak} を計算する．まず，μ_0 と C_{ox} は温度に依存しないので，式 (2) に表 7 の V_{th}，S，I_{leak} の値を代入し，μ_0 と C_{ox} の積を求める．次に，S は温度に比例し，V_{th} は温度が 1 ℃上昇すると 1mV 下がる[23] ことから，表 7 の値より，100 ℃における S と V_{th} の値を求める．最後にこれらの値を式 (2) に代入し，100 ℃のときの I_{leak} を求めた．その結果，70nm，50nm，35nm の各プロセスでの I_{leak} として，それぞれ，15.8μA/μm，40.0μA/μm，97.2μA/μm が得られた．

動的消費電力の求め方について述べる．動的消費電力は，各回路の動的消費エネルギーをプロセッサの実行時間で割ることにより得られる．各回路の動的消費エネルギーは，各回路の動作回数と 1 動作あたりの動的消費エネルギーの積である．一方，プロセッサの実行時間は，実行サイクル数とサイクル時間の積である．本論文では，base モデルの備える通常キャッシュのアクセス時間をサイクル時間とした．プロセッサ・シミュレータにより，プロセッサの実行サイクル数と，各回路の動作回数を求めた．CACTI により，通常キャッシュのアクセス時間と，各回路の 1 動作あたりの動的消費エネルギーを求めた．命令ウインドウは，キャッシュと同様，SRAM で構成されるテーブルなので，CACTI に命令ウインドウの構成を与えることで，予測アドレスフィールドの動的消費エネルギーを求めることができる．**表 8** に，各回路の 1 サイクルあたりの動作回数をベンチマーク平均で示す．また，**表 9** に，各回路の 1 動作あたりの動的消費エネルギーを示す．

表 8 各回路の 1 サイクルあたりの平均動作回数
Table 8 Average number of operations per cycle for each circuit.

Model	Number of Operations per Cycle				
	Cache	Preactivating Mechanism			
		Address Decoder	Predictor	Well Driver	Address Field
base	2.96	-	-	-	-
preactivating (stride predictor)	2.92	4.16	3.68	2.40	2.92
preactivating (hybrid predictor)	2.92	4.16	3.69	2.48	2.92
simple (1000)	2.95	-	-	-	-
simple (2000)	2.95	-	-	-	-
simple (4000)	2.96	-	-	-	-

表 9 各回路の 1 動作あたりの動的消費エネルギー
Table 9 Dynamic energy per operation for each circuit.

Process (nm)	Dynamic Energy per Operation (pJ)					
	Cache	Preactivating Mechanism				
		Address Decoder	Stride Predictor	Hybrid Predictor	Well Driver	Address Field
70	28.4	9.2	14.5	30.9	0.6	16.7
50	15.6	4.9	8.1	17.0	0.3	8.0
30	7.8	2.4	4.1	8.5	0.2	1.3

図 8 に base モデル，preactivating モデル，simple モデルについて，キャッシュと先行起動機構の全消費電力のベンチマーク平均を示す．縦軸は 70nm プロセスにおける base モデルの全消費電力を 100%としたときのそれぞれのモデルの全消費電力の割合を示し，横軸はプロセスを示す．図には 6 本の棒グラフがあり，左から順に base モデル，ストライド予測を行う preactivating モデル，ハイブリッド予測を行う preactivating モデル，update window size が 1000, 2000, 4000 サイクルである simple モデルの場合である．棒グラフは，base モデルと simple モデルの場合，2 つの部分からなり，下から順に，キャッシュの静的消費電力，キャッシュの動的消費電力である．一方，preactivating モデルの場合，6 つの部分からなり，下から順に，キャッシュの静的消費電力，キャッシュの動的消費電力，予約アドレスと取消アドレスのデコーダの動的消費電力，アドレス予測器の動的消費電力，p ウェルを駆動する回路の動的消費電力，命令ウインドウの予測アドレスフィールドの動的消費電力である．base モデルの棒グラフに付加した数字は，キャッシュの全消費電力（total）と動的消費電力（dynamic）の割合を示す．

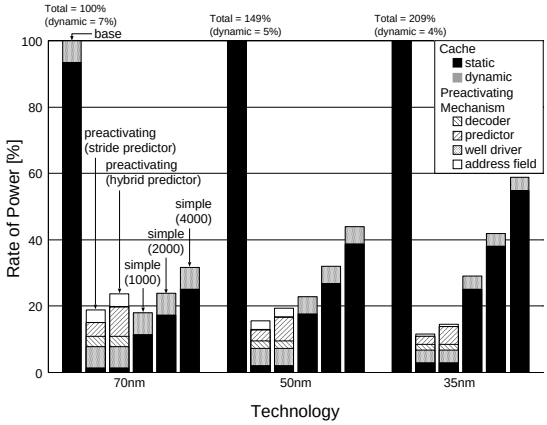


図 8 キャッシュと先行起動機構の全消費電力
Fig. 8 Total power of the cache and our preactivating mechanism.

図より，preactivating モデルは，先行起動機構の動的消費電力によって，全消費電力が増加することが分かる．70nm プロセスでは，preactivating モデルにおける先行起動機構の動的消費電力は，ストライド予測の場合 base モデルの全電力の 11.1%，ハイブリッド予測の場合 15.9%に達する．そのため，全消費電力は，ストライド予測の場合 18.9%，ハイブリッド予測の場合 23.7%にまで増加し，それぞれ，update window size が 1000，2000 サイクルである simple モデルとほぼ同等となる．

しかし，preactivating モデルでは静的消費電力を大幅に抑制できるのに加え，動的消費電力はプロセスの進歩に伴い減少していくため，プロセスの進歩に伴い全消費電力の割合が大幅に低下し，simple モデルよりも小さくなる．35nm プロセスにおいて，preacti-

vating モデルでは，先行起動機構の動的消費電力は，ストライド予測の場合 4.9%，ハイブリッド予測の場合 7.6%に減少する．そのため，全消費電力の割合はストライド予測の場合 11.6%，ハイブリッド予測の場合 14.5%と大幅に低下する．これは，update window size が 4000 サイクルである simple モデルよりも，それぞれ，47.2%ポイント，44.3%ポイント小さく，update window size が 1000 サイクルである simple モデルよりも，それぞれ，17.4%ポイント，14.5%ポイント小さい．

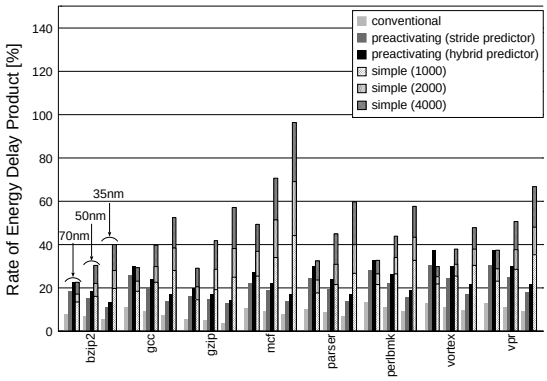
プロセスの進歩に伴い，base モデルの全消費電力が増加する一方で，preactivating モデルの全消費電力は減少するので，base モデルに対する先行起動機構の消費電力の削減効果は非常に大きくなる．70nm プロセスにおける base モデルの全消費電力を基準とすると，35nm プロセスでは，base モデルの全消費電力が 209%に増加する一方で，preactivating モデルの全消費電力は 11.6 〜14.5%に減少する．そのため，35nm プロセスでは，消費電力の削減効果はより大きくなり，preactivating モデルは全消費電力を，base モデルに対して，ストライド予測の場合 5.6%，ハイブリッド予測の場合 6.9%にまで削減することができる．

3.4.2 プロセッサ性能と消費エネルギーの評価

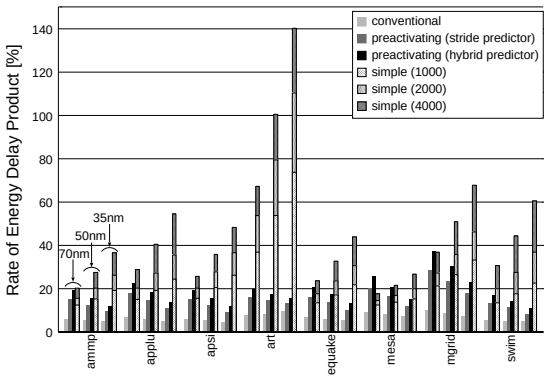
本節では，プロセッサ性能と消費エネルギーの総合的な評価を行う．最初に，広く評価指標として使われているエネルギー遅延積（ED 積）で評価する．次に，より性能に重点を置いた ED 積と遅延の積（以下 ED^2 積と呼ぶ）で評価をする．

図 9 に，ED 積の評価結果を示す．図 9 (a) は SPECint2000 の場合，(b) は SPECfp2000 の場合である．縦軸は 70nm プロセスにおける base モデルの ED 積を 100%としたときの ED 積の割合を示し，横軸は各ベンチマークを示す．図には 3 組の棒グラフがあり，左から順にプロセスが 70nm，50nm，35nm の場合である．各棒グラフの組は，4 本の棒グラフからなり，左から順に，conventional モデル，ストライド予測を行う preactivating モデル，ハイブリッド予測を行う preactivating モデル，simple モデルである．simple モデルでは，3 本の棒グラフが重なっており，奥から順に update window size が 4000，2000，1000 サイクルの場合である．**表 10** に，図 9 に示した ED 積の割合を，ベンチマーク平均で示す．

図より，conventional モデルは，他のモデルよりも常に，ED 積が小さくなることが分かる．しかし，図 6 から分かるように，conventional モデルは，DLC の起動時間が性能へ与える悪影響が非常に大きいという



(a)SPECint2000



(b)SPECfp2000

図 9 base モデルに対する ED 積
Fig. 9 ED product to the base model.

欠点を持つ．このため，ED 積が最小であることにのみ着目し，conventional モデルが最もよいモデルであるとすることはできない．

70nm プロセスでは，preactivating モデルは，update window size が 1000 サイクルである simple モデルよりも，ED 積が大きいことが分かる．しかし，プロセスの進歩に伴い，preactivating モデルは ED 積を simple モデルよりも削減できるようになる．これは，図 8 より preactivating モデルはプロセスの進歩に伴い消費電力の割合を大幅に低下させることができるためである．35nm プロセスでは，preactivating モデルの ED 積の割合は，update window size が 1000 サイクルである simple モデルよりも，12%ポイント〜15%ポイント低い値である．

ED^2 積の評価結果について述べる．**表 11** に，70nm プロセスにおける base モデルの ED^2 積を 100%としたときの ED^2 積の割合を，ベンチマーク平均で示す．

表 11 より，conventional モデルは，他のモデルよりも常に，ED^2 積が小さくなることが分かる．しかし，

表 10 base モデルに対する ED 積（ベンチマーク平均）
Table 10 ED product to the base model (average of the benchmarks).

Model	Rate of ED Product		
	70nm process	50nm process	35nm process
conventional	8.5%	7.4%	6.4%
preactivating (stride predictor)	20.3%	16.8%	12.6%
preactivating (hybrid predictor)	25.2%	20.6%	15.5%
simple (1000)	17.7%	22.2%	27.8%
simple (2000)	23.1%	30.5%	39.5%
simple (4000)	30.4%	41.8%	55.5%

表 11 base モデルに対する ED^2 積（ベンチマーク平均）
Table 11 ED^2 product to the base model (average of the benchmarks).

Model	Rate of ED^2 Product		
	70nm process	50nm process	35nm process
conventional	9.4%	8.2%	7.1%
preactivating (stride predictor)	21.4%	17.7%	13.3%
preactivating (hybrid predictor)	26.4%	21.7%	16.2%
simple (1000)	18.0%	22.6%	28.2%
simple (2000)	23.3%	30.8%	39.9%
simple (4000)	30.7%	42.1%	56.0%

先でも述べたように，conventional モデルは，DLC の起動時間による性能への悪影響が非常に大きいため，最もよいモデルであるとすることはできない．

表 10 と表 11 より，どのモデルにおいても，ED^2 積は，ED 積とほぼ同一の傾向を示すことが分かる．ED 積と ED^2 積の割合の差は，ベンチマーク平均でわずか 2%ポイント未満である．この原因は，各モデルの遅延の増加率は，エネルギーの削減率を大幅に下回るので，ED 積や ED^2 積の割合にほとんど影響を与えないことにある．

3.5 キャッシュ構成を変更した場合の影響

3.1 節〜3.4 節の評価では，キャッシュの構成は，32K バイト，2 ウェイ・セットアソシアティブ，32 バイト・ブロック，4 ポート，ミスペナルティ 6 サイクルとした．本節では，これらのうち，way 数，あるいは，容量を変更した場合の影響について述べる．

以下の構成のキャッシュについて，性能と消費電力を評価した．

- 32K バイト，4 ウェイ・セットアソシアティブ（32KB，4-way）のキャッシュ
- 64K バイト，2 ウェイ・セットアソシアティブ（64KB，2-way）のキャッシュ

表 12 各キャッシュ構成での base モデルに対する性能低下（ベンチマーク平均）
Table 12 Performance degradation from the base model for each cache configuration (average of the benchmarks).

Model	Performance Degradation		
	32KB 2-way	32KB 4-way	64KB 2-way
conventional	11.3%	11.3%	11.4%
preactivating (stride predictor)	5.4%	5.3%	5.5%
preactivating (hybrid predictor)	4.9%	4.8%	5.0%
simple (1000)	1.5%	1.3%	1.8%
simple (2000)	1.1%	0.8%	1.3%
simple (4000)	0.8%	0.5%	1.0%

表 12 に，各キャッシュ構成における，同一構成の base モデルに対する性能低下のベンチマーク平均を示す．表より，どのモデルにおいても，キャッシュ構成の変更による性能低下率の差は，ベンチマーク平均で 0.5%以下と極めて小さいことが分かる．このことから次のことが言える．

preactivating モデルでは，容量が同じで way 数が増加すると，セット数が減少し偶然によりセット予測率が向上すると予想されたが，セット数の減少が半分程度では，そういう偶然はほとんど生じないといえる．また，アクセスしたセットが最近アクセスされたセットと同一である偶然がより多く生じれば，性能は向上するが，way 数が倍になってもそのような偶然が大きく増加することはないといえる．simple モデルでも，way 数が増加すると異なる way 間でのアクセスで偶然起動ペナルティが回避される可能性が高まるが，そのような状況は少ないといえる．

way 数が同じで容量を増加させた場合，preactivating モデルでは，前述したことと逆のことが偶然生じうるが，偶然はほとんど生じていないので，性能低下率は変わらない．また，simple モデルでは，way 数が同じならば，性能低下率がほとんど変わらないのは自明のことである．

キャッシュの構成を変更した場合について，各モデルの静的消費電力を評価した．**表 13** に，32KB，2-way キャッシュにおける base モデルの静的消費電力を 100%としたときの静的消費電力の割合を，ベンチマーク平均で示す．

表より，どのモデルにおいても，静的消費電力の割合は，容量を増加させてもほとんど変化せず，way 数の増加に伴い増加することが分かる．この理由は以下のとおりである．静的消費電力は，動作状態にあ

表 13 各キャッシュ構成でのベースモデルに対する静的消費電力（ベンチマーク平均）
Table 13 Static power to the base model for each cache configuration (average of the benchmarks).

Model	Rate of Static Power		
	32KB 2-way	32KB 4-way	64KB 2-way
conventional	0.8%	1.6%	0.8%
preactivating (stride predictor)	1.4%	2.8%	1.4%
preactivating (hybrid predictor)	1.5%	2.9%	1.5%
simple (1000)	12.3%	22.3%	13.1%
simple (2000)	18.6%	31.5%	20.6%
simple (4000)	26.8%	42.7%	31.7%

るサイクルあたりの平均セット数と，セットあたりの SRAM セルの数に比例する．容量を増加させた場合，キャッシュの全セット数は増加するが，容量が十分であれば，動作状態にあるセット数はほとんど変化しない．一方，way 数を増加させた場合，セットあたりの SRAM セルの数が増加し，消費電力が増加する．

way 数の増加により消費電力が増加した場合でも，conventional モデルと preactivating モデルでは，simple モデルと異なり，動作状態にあるセット数は少ないので，その影響は小さい．conventional モデルと preactivating モデルの静的消費電力の割合は，ベンチマーク平均で 3%未満と非常に低い．これは，update window size が 1000 サイクルである simple モデルより，19%ポイント〜21%ポイントも低い値である．

4. 関 連 研 究

これまでに提案されている L1 データキャッシュの静的消費電力削減手法について述べる．

Kaxiras らは使用されないキャッシュラインの電源電圧を MT-CMOS を用いて停止することでリーク電流を削減する cache decay を提案した[9]．MT-CMOS を用いてキャッシュラインを停止した場合にはセルの記憶を保持することはできないため，ラインが最後に参照される時点（last access）を予測し，ラインを停止する．last access の予測には，ラインの参照はキャッシュに取り込まれた直後に集中し，一旦参照されなくなると，以後長い間参照されることがないという性質を利用する．ラインへの参照と参照の間隔（access interval）を監視し，一定サイクル以上の間，同一ラインへの参照が行われなければそのラインへ再びアクセスされることはないと予測し，電源供給を停止する．この手法では last access 以降の静的消費電力は削減できるが，それ以前の静的消費電力は削減できない．このため，彼らの評価では L1 データキャッシュの静的消費電力を 4 分の 1 にしか削減できない．

Flautner らは動的に電源電圧を変化させる DVS（Dynamic Voltage Scaling）をキャッシュに適用し，ライン単位で静的消費電力を制御する drowsy cache を提案した[4]．drowsy cache では，DVS を用いてラインの電源電圧を通常より低い状態（drowsy mode）にすることでリーク電流を削減する．しかし，ラインの電源電圧を drowsy mode から通常の状態にする時間は長く，性能に悪影響を与える．これに対し，Flautner らはキャッシュラインの状態を制御するいくつかの手法を提案している．これらの手法の中で最も有効であるシンプル方式は，あるラインへの参照は時間的に集中する特性があることを利用している．この手法では，最初に全てのラインを drowsy mode にする．drowsy mode のラインが参照されるときに当該ラインを動作状態にし，その状態を保つ．そして，一定の間隔で全てのラインを drowsy mode にする．動作状態になったラインは，参照されなくなった後であっても全てのラインが drowsy mode にされるまで動作状態を保つので，無駄に電力を消費してしまう．このため，3.3 節で評価したように，この手法では L1 データキャッシュの静的消費電力を 8 分の 1 にしか削減できない．

5. ま と め

DLC を適用したキャッシュにおいて，回路を動作状態に活性化する遅延による性能低下を緩和する方式を提案した．回路の動作状態への活性化には大きな遅延が生じる．活性化の遅延により，従来の DLC キャッシュでは，32KB，2-way 構成において平均で，SPECint2000 で 17.2%，SPECfp2000 で 5.8%性能が低下することが分かった．この性能低下を抑えるために，アドレス予測を用いて，メモリ参照以前にキャッシュラインを動作状態にする方式を提案した．この方式ではアドレス予測としてハイブリッド予測を用いた場合，性能低下率を平均で，SPECint2000 で 8.9%，SPECfp2000 で 1.1%にまで抑制することができる．また，静的消費電力は DLC キャッシュと比べるとわずかに増加するものの，依然として DLC を用いない通常のキャッシュが消費する電力の 1%にまで削減することができる．我々の機構は先行起動機構の動的消費電力によって全消費電力が増加し，70nm プロセスでは，通常のキャッシュが消費する全消費電力に対する消費電力の削減率は，23.7%に悪化する．しかし，我々の機構は静的消費電力を大幅に抑制できるのに加

え，動的消費電力はプロセスの進歩に伴い減少していくため，プロセスの進歩に伴い全消費電力が大幅に低下し，35nm プロセスにおいて，消費電力の削減率を6.9%にまで改善させることができる．

謝辞

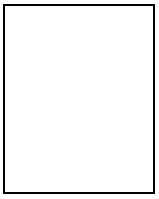
本研究の一部は，文部科学省科学研究費補助金基盤研究（C）課題番号 15500036，文部科学省 21 世紀 COE プログラム，財団法人栢森情報科学振興財団研究助成，財団法人堀情報科学振興財団研究助成の支援により行った．

参 考 文 献

1) Agarwal, V., Hrishikesh, M. S., Keckler, S. W. and Burger, D.: Clock Rate versus IPC: The End of the Road for Conventional Microarchitectures, *Proc. 27th Annual International Symposium on Computer Architecture*, pp. 248–259 (June 2000).
2) Burger, D. and Austin, T. M.: The Simplescalar Tool Set Version 2.0, Technical Report 1342, Department of Computer Sciences, University of Wisconsin-Madison (June 1997).
3) Butts, J. A. and Sohi, G. S.: A Static Power Model for Architects, *Proc. 33rd Annual International Symposium on Microarchitecture*, pp. 191–201 (December 2000).
4) Flautner, K., Kim, N. S., Martin, S., Blaauw, D. and Mudge T.: Drowsy Caches: Simple Techniques for Reducing Leakage Power, *Proc. 29th Annual International Symposium on Computer Architecture*, pp. 148–157 (May 2002).
5) 藤岡 涼，片山 清和，小林 良太郎，安藤 秀樹，島田俊夫：VT-CMOS を用いたデータキャッシュでの性能低下をアドレス予測を用いて低減する手法，電子情報通信学会技術研究報告，DSP2001-110〜124，pp. 75–82 (2001 年 10 月).
6) Fujioka, R., Katayama, K., Kobayashi, R., Ando, H. and Shimada, T.: A Preactivating Mechanism for a VT-CMOS Cache using Address Prediction, *Proc. International Symposium on Low Power Electronics and Design*, pp. 247–250 (August 2002).
7) Horowitz, M.: Timing Models for MOS Circuits, Technical Report, SEL83-003, Integrated Circuits Laboratory, Stanford University (1983).
8) Kawaguchi, H., Itaka, Y. and Sakurai, T.: Dynamic Leakage Cut-off Scheme for Low-Voltage SRAM's, *Symposium on VLSI Circuits Digest of Technical Papers*, pp. 140–141 (June 1998).
9) Kaxiras, S., Hu, Z. and Martonosi, M.: Cache Decay: Exploiting Generational Behavior to Reduce Cache Leakage Power, *Proc. 28th Annual International Symposium on Computer Architecture*, pp. 240–251 (June 2001).
10) Kuroda, T., Fujita, T., Mita, S., Nagamatsu, T., Yoshioka, S., Suzuki, K., Sano, F., Norishima, M., Murota, M., Kako, M., Kinugawa, M., Kakumu M. and Sakurai, T.: A 0.9-V, 150-MHz, 10-mW, 4mm^2, 2-D Discrete Cosine Transform Core Processor with Variable-Threshold-Voltage (VT) Scheme, *IEEE Journal of Solid-State Circuits*, Vol. 31, No. 11, pp. 1770–1779 (November 1996).
11) Lipasti, M. H., Wilkerson, C. B. and Shen, J. P.: Value Locality and Load Value Prediction, *Proc. Second International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 138–147 (October 1996).
12) Lipasti, M. H. and Shenet, J. P.: Exceeding the Dataflow Limit via Value Prediction, *Proc. 29th Annual International Symposium on Microarchitecture*, pp. 226–237 (December 1996).
13) MIPS Technologies, Inc.: MIPS R10000 Processor User's Manual, Version 2, (October 1996).
14) Mutoh, S., Douseki, T., Matsuya, Y., Aoki, T. and Yamada, J.: 1-V Power Supply High-Speed Digital Circuit Technology with Multi Threshold-Voltage CMOS, *IEEE Journal of Solid-State Circuits*, Vol. 30, No. 8, pp. 847–854 (August 1995).
15) Pollack, F.: New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies, Keynote Presentation to the 32nd Annual International Symposium on Microarchitecture (November 1999).
16) Reinman, G. and Calder, B.: Predictive Techniques for Aggressive Load Speculation, *Proc. 31st Annual International Symposium on Microarchitecture*, pp. 127–137 (December 1998).
17) Reinman, G. and Jouppi, N. P.: Extensions to CACTI, Unpublished document.
18) Reinman, G. and Jouppi, N. P.: CACTI 2.0: An Integrated Cache Timing and Power Model, Compaq WRL Report 2000/7 (February 2000).
19) Sazeides, Y. and Smith, J.E.: The Predictability of Data Values, *Proc. 30th Annual International Symposium on Microarchitecture*, pp. 248–258 (December 1997).
20) International Technology Roadmap for Semiconductors (ITRS): International Roadmap for Semiconductors 2001 Edition, Process Integration, Device, and Structures and Emerging Re-

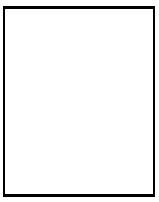
search Devices, Technical Report (2001).
21) Shivakumar, P. and Jouppi, N. P.: CACTI 3.0: An Integrated Cache Timing and Power, and Area Model, Compaq WRL Report 2001/2 (August 2001).
22) Sze, S. M.: Semiconductor Devices: Physics and Technology, John Wiley and Sons, Inc. (1986).
23) Taur, Y. and Ning, T. H.: Fundamentals of Modern VLSI Devices, Cambridge University Press (1998).
24) Wang, K. and Franklin, M.: Highly Accurate Data Value Prediction using Hybrid Predictors, *Proc. 30th Annual International Symposium on Microarchitecture*, pp. 281–290 (December 1997).
25) Weste, N. H. E. and Eshraghian, K.: Principles of CMOS VLSI Design, A System Perspective, Second Edition, Addison Wesley (1993).
26) Wilton, S. J. E. and Jouppi, N. P.: An Enhanced Access and Cycle Time Model for On Chip Caches, WRL Research Report 93/5 (July 1994).

(平成？年？月？日受付)
(平成？年？月？日採録)



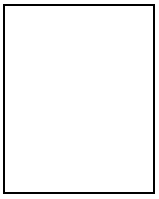
小林良太郎（正会員）

1995 年名古屋大学工学部電子情報学科卒業．1997 年名古屋大学大学院工学研究科電子情報学専攻博士過程前期過程修了．2000 年名古屋大学大学院工学研究科電子情報学専攻博士過程後期過程満了．工学博士．2000 年名古屋大学大学院工学研究科電子情報学専攻助手．1999 年情報処理学会山下記念研究賞受賞．2002 年情報処理学会論文賞受賞．計算機アーキテクチャの研究に従事．



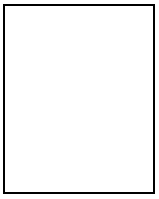
藤岡　涼

2001 年名古屋大学工学部電子情報学科卒業．2003 年名古屋大学大学院工学研究科電子情報学専攻博士過程前期過程修了．同年，日本電気株式会社に入社．



安藤　秀樹（正会員）

1959 年生．1981 年大阪大学工学部電子工学科卒業．1983 年大阪大学大学院修士課程修了．京都大学工学博士．1983 年三菱電機 (株)LSI 研究所．ISDN 用ディジタル信号処理 LSI，第 5 世代コンピュータ・プロジェクトの推論マシン用プロセッサの設計に従事．1991 年 Stanford 大学客員研究員．1997 年名古屋大学大学院工学研究科電子情報学専攻講師．1998 年名古屋大学助教授．1998〜2001 年東京大学大学院理学系研究科助教授併任．2004 年名古屋大学教授．1998 年，2002 年情報処理学会論文賞受賞，電子情報通信学会/情報処理学会 2003 年先進的計算基盤システムシンポジウム優秀学生論文賞．計算機アーキテクチャ，コンパイラの研究に従事．ACM，IEEE，電子情報通信学会会員．



島田　俊夫（正会員）

1968 年東京大学工学部計数工学科卒業．1970 年東京大学大学院修士過程修了．同年電子技術総合研究所入所．1993 年より名古屋大学大学院工学研究科電子情報学専攻教授．人口知能向き言語，LISP マシン，データフロー計算機の研究に従事．最近はマイクロプロセッサのアーキテクチャやチップ内並列処理の研究を行っている．1988 年度市村賞，1994 年度情報処理学会論文賞，1995 年注目発明，2002 年度情報処理学会論文賞受賞．工学博士．