2. 新しいデザインバランス

1

省電力とプロセッサ

佐藤 寿倫
九州工業大学
tsato@ai.kyutech.ac.jp

背景

プロセッサ性能は半導体技術の恩恵を受け向上し続けているが，まだ十分とは言えない．情報機器の利用における安全性の確保は重要性を増しており，依然としてプロセッサ性能の向上が要求されている．たとえば，ウイルスチェックは非常に負荷の大きな処理であり，現状では電子メールの送受信に大きな時間を要し，ユーザがいらいらを感じることも多い．現在では1チップ上に10億個ものトランジスタが集積され，これはスーパーコンピュータを1チップで実現可能な規模である．プロセッサの性能改善は容易だと思える．しかし微細化の進展はこれまで顧みる必要のなかったさまざまな課題を提示しており，その実現は容易ではない．その1つが省電力である．

消費電力はかつて90年代初めに注目された．1991年には International Workshop on Power And Timing Modeling Optimization and Simulation（PATMOS）が，1994年には Symposium on Low Power Electronics（SLPE）と International Symposium on Low Power Design（ISLPD）が始まった．著者は SLPE に2度参加したが，第1回目は活気立っていた．省電力をプロセス技術や回路技術だけに頼るのではなく，アーキテクチャでも考慮しなければならないと強く印象づけられた．ところが SLPE は早くも第2回で失速した．参加者は減り，この会議は終わりだといった声も聞かれ，実際翌1996年には ISLPD とマージされて International Symposium on Low Power Electronics and Design（ISLPED）となった．恐らく当時はまだ消費電力の問題をプロセス技術や回路技術のみで解決可能であり，アーキテクチャ研究者の関心を繋ぎとめることができなかったのだろう．アーキテクチャ分野での消費電力への無関心ぶりは，この分野の代表的な2つの国際会議 International Symposium on Computer Architecture（ISCA）と International Symposium on Microarchitecture（MICRO）のプログラムを眺めれば一目瞭然である．**図-1** にこれら2つの会議で発表された消費電力関連の論文の割合をまとめた．1996年までは1件の発表もなく，その後も1999年までは発表があっても1件だけという状況だった．ところが2000年に変化が起こった．1999年の MICRO でインテルの Fred Pollack 氏が基調講演“New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies”を行い，プロセッサの電力密度が原子炉なみであるという有名なスライドを見せた．また2000年から2001年にかけて，カリフォルニアで電力危機があった．1998年から2000年にかけて ISCA や MICRO に併設してワークショップやチュートリアルを催すという地道な活動があったことも忘れてはいけない．これらを契機に，アーキテクチャ研究者の消費電力に対する関心が高まったのだろう．現在では，省電力はアーキテクチャ上の重要な関心事である．ISLPED は今では権威の高い会議に成長し，多数のアーキテクチャ論文が発表されている．

電池駆動のモバイル機器を長時間使用したいという動機から，省電力の検討が始まった．しかし2つの側面から，ハイエンド・プロセッサにおいても消費電力は無視

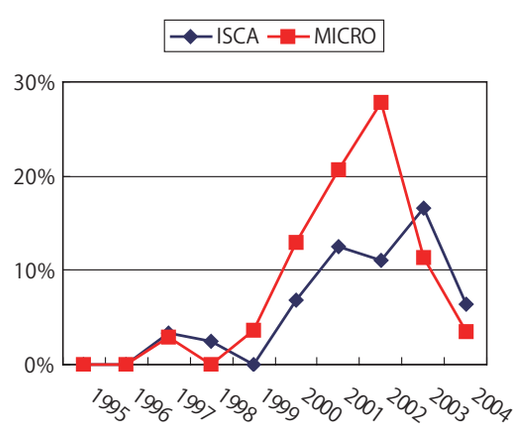

図-1　消費電力関連論文の割合

できないものとなった．1つは依然として存在する高性能コンピュータの要望である．高性能コンピュータは大きな電力を必要とする．たとえば，地球シミュレータには専用の変電施設が設けられている．大きな電力を消費する高性能コンピュータを必要とする分野で興味深いのが，現在隆盛を誇っている情報サービス産業である．情報検索サービスやネット通販に代表されるインターネット上でのサービスにはPCクラスタが用いられており，多くのプロセッサを必要とする．現実に省電力なPCクラスタの研究が盛んである．もう1つは半導体の微細化に伴う副作用である．これまで微細化はプロセッサの性能向上に貢献していた．しかし現在，消費電力の過密化による発熱が致命的な問題である．微細化に伴ってより小さな領域により小さなトランジスタをより多数集積できるようになり，より小さな領域でより大きな電力が消費されている．一方でLSIはその構造上，放熱が困難である．そのためのファンやフィンも研究されているが，コストが大きいために情報家電に採用するには敷居が高い．ましてやモバイル機器では，機器の大きさの制約から採用できない．インテルが予告していたプロセッサの発売をキャンセルしたのは記憶に新しいが，これは発熱の問題とそれに伴う信頼性低下の問題を解決できなかったためであると考えられている．

CMOS回路で消費される電力は，ダイナミック電力，貫通電流電力，そしてリーク電流電力に分けられる．ダイナミック電力は負荷の充放電時に消費される電力，貫通電流電力はpMOSとnMOSの切り替え時に瞬時に両者が同時に通電することによる貫通電流によって消費される電力，そしてリーク電流電力は半導体の性質上トランジスタがオフの時にも流れてしまうリーク電流によって消費される電力である．このなかで貫通電流電力は，バランスのよい設計がされていれば消費電力全体の10%以下に抑えることが可能である．以下では，従来は消費電力の大部分を占めていたダイナミック電力と，近年深刻になったリーク電流電力について，それらの代表的な削減技術を取り上げる．

ダイナミック電力対策

CMOS回路で消費されるダイナミック電力は以下の式で表される．

$$P_{active} = \alpha \cdot C_{load} \cdot f \cdot V_{dd}^2 \quad \cdots \quad (1)$$

ここで，αはスイッチング確率，C_{load}は負荷容量，fはクロック周波数，V_{dd}は電源電圧である．ダイナミック電力を削減するためには，これらの項を小さくすればよい．

■回路・デバイス技術

スイッチング確率を低減する方法として，ゲーテッドクロック方式がよく知られている．データ転送不要時にクロック供給を止めることで消費電力を削減する方法である．トランジスタやローカル配線の負荷容量は，基本的にはスケーリング則に従って縮小される．回路のスタイルを変更することでトランジスタ数を削減したりスイッチング確率を削減したりする方法など，回路レベルでの工夫で消費電力を削減する方法は多々あるが本稿では省略する．興味のある読者はたとえば文献1）などを参照されたい．以下では，クロック周波数と電源電圧を変えることで消費電力を削減する，DVFS（Dynamic Voltage Frequency Scaling）に焦点を当てる．

式（1）より分かるように電源電圧を下げることがダイナミック電力の削減には効果が大きい．しかし電源電圧を下げると以下の副作用を生じる．CMOS回路の遅延時間は以下の式で近似される．

$$D \propto \frac{V_{dd}}{(V_{dd}-V_{th})^{1.5}} \quad \cdots \quad (2)$$

ここでV_{th}はトランジスタの閾値電圧である．式（2）より電源電圧を下げると回路の遅延時間が増大することが分かる．これによりクロック周波数を低くする必要を生じ，プロセッサの性能を低下させる．不用意に電源電圧を下げるわけにはいかないが，$f=\frac{1}{D}$であるから消費されるエネルギー（電力遅延積$P_{active} \cdot D$）はV^2_{dd}に比例し，要求仕様を満足できる範囲で電源電圧をできるだけ下げることが望ましい．処理の空間的あるいは時間的偏りによって生じる無駄な電力消費を除くために，高いクロック周波数が必要ない場合には，低いクロック周波数とそれを満足できるだけの最小限の電源電圧を供給する．この考えを空間的に適用したものが多電源方式であり，時間的に適用したものがDVFSである．DVFSでは処理量の小さな期間にクロック周波数と電源電圧を下げる．インテルのSpeedStepとFoxtonテクノロジや，トランスメタのLongRun，ARMのIEM（Intelligent Energy Management）で，DVFSが利用されている．

■マイクロアーキテクチャ支援

マイクロアーキテクチャからの取り組みでも，式（1）の各項を小さくする工夫によってダイナミック電力の削減を試みる．1度行った処理を再利用するとスイッチング確率を下げることができる．たとえば，デコード後の命令を保持するトレースキャッシュが考えられる．処理の対象となるデータのビット幅を小さくできれば負荷容量を小さくすることに相当する．パイプライン処理や並列処理を行えばクロック周波数を下げてもスループットを維持でき，クロック周波数と電源電圧を下げることでダイナミック電力を削減できる[1)]．他にもマイクロアーキテクチャでの工夫で消費電力を削減する方法はあるが

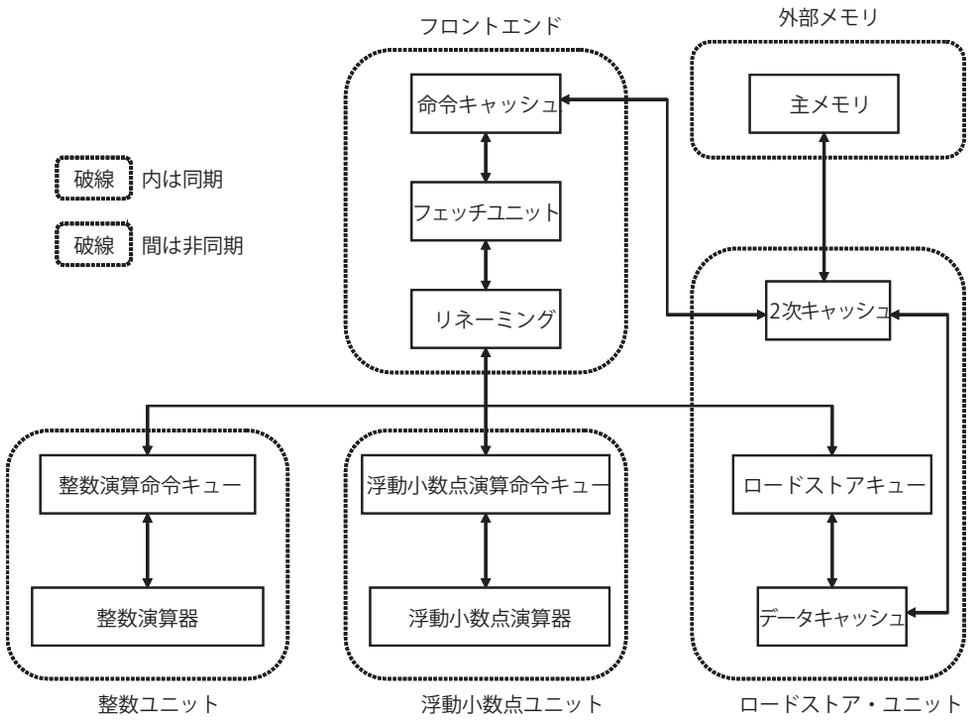

図-2　GALSプロセッサのブロック図

本稿では省略する．興味のある読者はたとえば文献1）～3）などを参照されたい．以下では回路上の工夫と同様，DVFSに焦点を当てる．

無駄な電力消費は空間的・時間的な処理の偏りが原因なので，その特徴を利用してDVFSを適用するのが有効である．米国のカーネギーメロン大（http://www.ece.cmu.edu/~dianam/）やロチェスター大（http://www.ece.rochester.edu/research/acal/），本国の東京大（http://www.hal.rcast.u-tokyo.ac.jp/）などで研究されているGlobally Asynchronous Locally Synchronous（GALS）プロセッサやMultiple Clock Domain（MCD）プロセッサは，この目的に適したプロセッサアーキテクチャである．**図-2**のようにGALSプロセッサでは，チップが複数の比較的大きなブロックに分割される．各ブロックには固有の電源とクロックが供給され，独立してDVFSを適用できる．ブロック間の通信は非同期となるので，同期をとるための非同期キューが必要である．図-2では，フロントエンド，整数ユニット，浮動小数点ユニット，そしてロードストア・ユニットの4つのドメインに分割されている．図にはチップ外の主メモリも書かれている．本来はブロック間通信のためのキューが必要であるが，フロントエンドを除く各ドメインには，アウトオブオーダ命令スケジューリングのための命令キューが元々備わっているので，これらを有効利用することでキューの追加を避けている．

プロセッサで実行されるアプリケーションプログラムの特徴により，各ドメインの処理量は異なる．各ドメインが必要とする処理量に応じたクロック周波数と電源電圧を各ドメイン個別に提供できれば，空間的な処理のばらつきを考慮した省電力化が可能である．またプログラム実行中の各フェーズの特徴によっても，各ドメインの処理量は異なる．DVFSを利用して，プログラムの進行に伴って各ドメインにおけるクロック周波数と電源電圧を変えれば，時間的な処理のばらつきを考慮した省電力化が可能である．したがって，GALSプロセッサでDVFSが効果的に働くためには，各ドメインでの電源電圧とクロック周波数の制御が適応性を持つ必要がある．ロチェスター大のグループは，ハードウェアモニタを利用する方法やプロファイル情報を利用する方法を検討している．

微細化の進展とクロック周波数の上昇に伴い配線遅延の問題がクローズアップされているが，GALSプロセッサはこの問題に対しても解決策の1つである．消費電量と配線遅延の2つの問題を同時に解決できるという意味で，将来のプロセッサアーキテクチャの有望な候補といえよう．

リーク電流電力対策

電源電圧を下げることがダイナミック電力を削減する最も有効な方法であることは，前章で述べたとおりである．一方，スケーリングが進んでもトランジスタ内部の電界を限界値以下に抑えるためには，電源電圧をスケーリングする必要がある．以上の2つの理由から電源電圧を下げることは必然であるが，式（2）より電源電圧を下げるとプロセッサ性能が低

下する．これを防ぐためには，閾値電圧も同様にスケーリングしなければならない．しかし閾値電圧を低減すると，ソース・ドレイン間のサブスレッショルド・リーク電流が増大する．加えてスケーリングに伴ってゲート絶縁膜が薄くなると，トンネル電流が流れるようになりゲート・リーク電流が増大する．さらに，スケーリングが進んでチャネル濃度が高くなると，ゲート電極下のドレイン端に高い電界がかかることによりドレインから基板へ Gate Induced Drain Leakage（GIDL）と呼ばれる接合リーク電流が流れる．**図 -3** にこれらのリーク電流をまとめた．以上のように微細化によりリーク電流が増大し，ハイエンド・プロセッサではリーク電流・電力が電力消費量全体の 50 ～ 90%を占めると言われている．

ゲート・リーク電流に対しては，比誘電率の高い High-K 絶縁膜材料を導入することで解決が試みられている．電気的には薄膜化と同じ効果を得つつ物理的には絶縁膜を厚くでき，トンネル電流を抑えることができるからである．GIDL についても，チャネルを低濃度にできるゲート絶縁膜材料の導入やトランジスタ構造の改良により解決が試みられている．一方，サブスレッショルド・リーク電流は材料や構造を変えても解決が難しく，回路やアーキテクチャの工夫により解決が試みられている．以下ではサブスレッショルド・リーク電流に焦点を当て，その削減方法を紹介する．

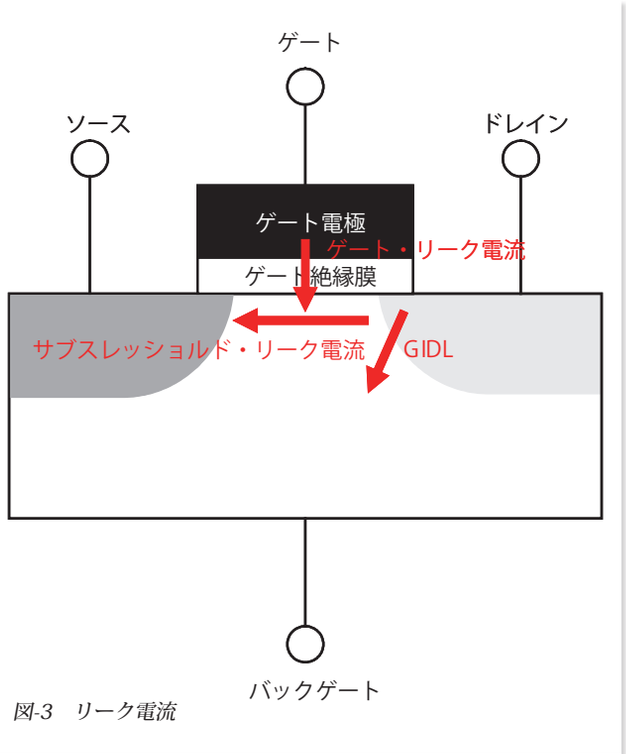

図-3　リーク電流

■回路・デバイス技術

サブスレッショルド・リーク電流によって消費される電力は以下の式で与えられる．

$$P_{leakage} = I_{leakage} \cdot V_{dd} = I_0 \cdot 10^{\frac{V_{th}}{s}} \cdot V_{dd} \quad \cdots \quad (3)$$

ここで I_0 は定数，s はサブスレッショルド・スイング・パラメータと呼ばれ次式で与えられる．

$$s = \ln 10 \cdot \frac{kT}{q\eta} \quad \cdots \quad (4)$$

ここで k はボルツマン定数，T は絶対温度，q は電子の電荷，η はデバイスで決まる定数である．式（3）から分かるように，サブスレッショルド・リーク電流は閾値電圧が下がるに伴い指数関数的に増大する．ここにリーク電流の問題の深刻さがある．

サブスレッショルド・リーク電流の削減に有効な技術をまとめると

- トランジスタのサイズを最適化する方式
- 多電源電圧を用意する方式
- 多閾値電圧を用意する方式
- スタック効果を利用する方式
- 電源電圧を動的に制御する方式
- 閾値電圧を動的に制御する方式
- スリープトランジスタを利用する方式

となる．以下で簡単に説明する．関心のある読者は文献 4）などを参考に原論文にあたっていただきたい．

トランジスタのサイズを最適化する方式では，機能は同じだがトランジスタのサイズが異なるセルを用意する．大きな駆動力を必要としないところにチャネル幅の小さなトランジスタを使用すれば，リーク電流を削減できる．この方法は，従来用いられている CAD ツールのライブラリを充実させるだけで採用できるが，リーク電流削減の効果はそれほど大きくない．

多電源電圧を用意する方式では，チップ上のブロックごとに複数の電源電圧を使い分ける．たとえば，クリティカルパス上のトランジスタにのみ高電圧の電源を供給し，残りのトランジスタには低電圧の電源を供給する．式（3）より，電源電圧の削減に比例したリーク電流電力の削減が期待できる．この方法では，電源配線の引き回しが煩雑になること，低電圧部から高電圧部への信号線にレベルシフタ回路を必要としその分だけ遅延が増すこと，などの欠点がある．

多閾値電圧を用意する方式も同様の考えに基づいており，電源電圧ではなく閾値電圧を使い分ける．同様に，クリティカルパス上のトランジスタの閾値を低くし，残りのトランジスタの閾値を高くする．マスクを追加する必要はあるが，閾値の異なるトランジスタを使ったセルをライブラリに登録するだけで従来の CAD ツールを利用できるので，現在でも多くの LSI メーカで採用されている．

スタック効果を利用する方式は，縦積みされたトランジスタの 1 つ以上がオフであると，リーク電流が$\frac{1}{5}$から$\frac{1}{10}$まで低減する効果を利用する．これまで説明した方式と異なり，採用のためのコストがかからない方式である．

電源電圧を動的に制御する方式は，ダイナミック電力

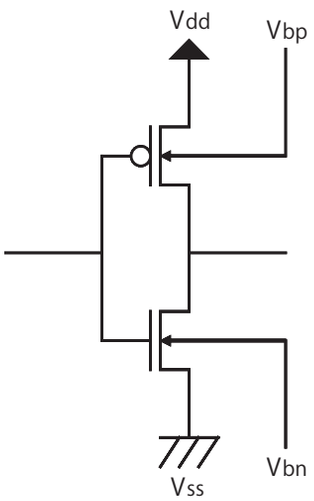

図-4 VTCMOS

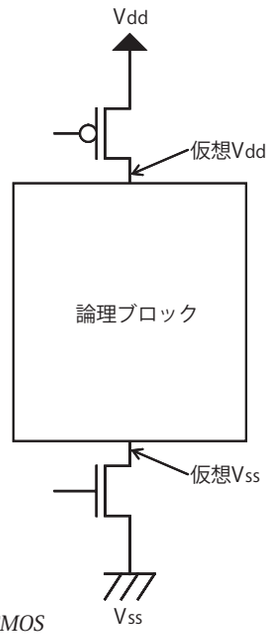

図-5 MTCMOS

対策の章で説明したDVFSである．リーク電流電力も電源電圧に比例するので，DVFSによりリーク電流電力を削減できる．

閾値電圧を動的に制御する方式はVTCMOS（Variable Threshold voltage CMOS）等と呼ばれている．**図-4**にVTCMOSを示す．この方式では基盤（図-3のバックゲート）にバイアス電圧V_{bp}とV_{bn}を印加することで閾値電圧を制御し，その結果リーク電流を削減する．nMOSでは，基盤に逆バイアスを与えると閾値電圧が上昇し，逆に順バイアスを与えると閾値電圧は低下する．逆バイアスを利用する場合には，製造時に全体の閾値を低めに設定しておき，待機時などリーク電流を抑えたい時に基盤に逆バイアスを与える．一方順バイアスを利用する場合には，製造時には全体の閾値を低めに設定しておき，高速動作を必要とする時に基盤に順バイアスを与える．もちろん，順逆両方の基盤バイアスを使うことも可能である．またnMOSのみ，あるいはpMOSのみに基盤バイアスを印加することも可能である．インテルの検討によればリーク電流が$\frac{1}{2}$から$\frac{1}{10}$まで削減される．基盤バイアスを制御するための追加の回路が必要になるが，リーク電流を削減する方式として有望視されている．たとえばトランスメタのLongRun2で利用されている．VTCMOSを利用すると閾値電圧を本来の設計値に近づけることができるので，近年問題となっているトランジスタの性能ばらつきを解消する方法としても期待されている．ただし次の問題がある．基盤バイアスの印加によりGIDLが増大し，逆にリーク電流の総量が増えてしまうという現象が生じることがある．

スリープトランジスタを利用する方式は，MTCMOS（Multi Threshold voltage CMOS）やGated-Vddと呼ばれている．MTCMOSは待機時のリーク電流を削減することを目的としている．待機時に動作させる必要のないブロックへの電源供給を遮断する．**図-5**にMTCMOSの動作原理を示す．本来の論理ブロックと直列に，電源電圧側と接地側にスリープトランジスタが挿入されている．図では両側にスリープトランジスタを接続しているが，どちらか一方のみが使用される場合もある．論理ブロックは低閾値トランジスタで作られ，スリープトランジスタでは高閾値トランジスタを使う．スリープトランジスタを，論理ブロックの動作時にはオンにし待機時にはオフにする．論理ブロックにはスリープトランジスタを介して電源が供給される．事実上の電源と接地をそれぞれ仮想V_{dd}，仮想V_{ss}と呼ぶ．スリープトランジスタは閾値電圧が高いのでリーク電流が小さい．キルヒホッフの法則よりスリープトランジスタに流れる電流と論理ブロックに流れる電流は等しいから，論理ブロックに流れるリーク電流はスリープトランジスタに流れる電流で抑えられることになる．インテルの評価によると，リーク電流を$\frac{1}{2}$から$\frac{1}{1000}$までに削減可能である．MTCMOSでは待機時に電源供給が遮断されるので，レジスタやメモリの内容などの論理ブロック内の状態を失いたくないものについては，特別な対策が必要である．最も簡単な方法はMTCMOSの対象から外すことである．また，直列にスリープトランジスタが挿入されるために，動作速度が低下する問題もある．

■マイクロアーキテクチャ支援

現在のプロセッサはその面積の大部分をキャッシュメモリが占めている．ロジック部と比較してメモリ部は密なので，面積比以上にトランジスタ数比はメモリ部が大きくなる．リーク電流は動作に関係なく流れるので，その大きさとトランジスタ数には強い相関がある．したがって，必然

的にキャッシュメモリに流れるリーク電流を削減しようとする研究が多く，演算器などに流れるリーク電流を削減しようとする試みはまだ少ない．これまでに提案されたキャッシュのリーク電流削減手法をまとめると

- DRI キャッシュ
- ディケイ・キャッシュ（Decay cache）
- ドラウジー・キャッシュ（Drowsy cache）

となる．以下で簡単に説明する．関心のある読者は文献 5）などを参考に原論文にあたっていただきたい．

DRI キャッシュはスリープトランジスタを利用してリーク電流を削減する．基本的なアイディアは，プログラムが必要とするキャッシュ容量だけを提供するように，不要なキャッシュへの電源供給を遮断してキャッシュ容量を可変にしようとするものである．そのために，あらかじめ設定された期間ごとにキャッシュミス回数を評価する．ミス回数が閾値よりも大きければキャッシュ容量を増し，小さければ容量を減らす．

ディケイ・キャッシュもスリープトランジスタを利用している．下位のメモリ階層からデータを読み込んだキャッシュラインは，局所性のためにしばらくの間は頻繁に参照されるが，時間が経つとまったく参照されなくなる．次にこのラインが参照されるのは新しく別のデータが読み込まれる時である．参照されない期間にそのラインへの電源供給を遮断すれば，パフォーマンスにまったく影響することなくリーク電流を削減できる．インテルの Yonah に採用された Dynamic Smart Cache Sizing はいまだ情報に乏しく詳細が分からないが，一種のディケイ・キャッシュと推察される．

ドラウジー・キャッシュでは，DVFS を利用してリーク電流を削減する．一定期間ごとにすべてのキャッシュラインへの電源電圧を低下させ，参照されたラインのみ電源電圧を上昇させる．定期的にキャッシュ全体がスリープモードに移り，リーク電流を削減できる．この方法が採用可能である理由は，スリープトランジスタと比較してスリープモードからの回復に必要な時間がきわめて短いことである．

以上は最初の提案に従って各アーキテクチャで特定の回路方式を利用する場合を説明したが，必ずしも上述の組合せを用いる必要はない．たとえば VTCMOS を利用して DRI キャッシュを構成することも可能である．

今後の動向予測

戦略的に研究されるべきテーマを提案したい．1 つ目は温度の見積り手法である．式（4）より，サブスレッショルド・リーク電流は温度に大きく影響されることが分かる．温度が上昇すればリーク電流は増大し，リーク電流電力が増大すれば温度が上昇する．両者を同時に考慮した評価を可能にするために，温度を正確に見積もることのできる手法を考えなければならない．

2 つ目はマルチコア GALS プロセッサである．本特集の「1.5 チップ・マルチプロセッサ」（前月号掲載）で解説されているとおり，マルチコアプロセッサは有望なプロセッサアーキテクチャである．インテルのペンティアム D では，クロック周波数はコアごとに独立に定めることができるが，電源電圧はチップ全体で統一されている．各コアでの処理量に大きなばらつきがあると無駄な電力が消費されてしまう．各コアをドメインとした GALS プロセッサであればこの問題を解決できる．

最後に，ばらつきを考慮することの重要性を指摘したい．スケーリングが進むにつれて，製造ばらつきの性能ばらつきに与える影響が大きくなっている．ばらつきが大きいと，電源電圧すなわち信号振幅の低減により信号対雑音比が悪化する．1 チップに集積されるトランジスタは増えているから，チップ全体の誤動作率が上がる．省電力には電源電圧の低減が必要だが，信頼性を損なってしまう．したがって信頼性の確保と省電力とのトレードオフに配慮する必要がある．本特集の「2.2 信頼性・安全性とプロセッサ」で解説されている通り，プロセッサにおける信頼性の確保は重要である．ばらつきが存在したとしても，消費電力を削減しつつ正しい動作を保証できるアーキテクチャが必要である．その 1 つが，設計制約の最悪条件を満足するのではなく，典型的なケースに最適化し例外的なケースには安全装置を備える Better-than-worst-case 設計である．関心のある読者にはぜひ文献 6）を読んでいただきたい．

低消費電力プロセッサに求められる性能が上がるにつれて，「性能を犠牲にして消費電力を削減する」から「性能を維持したままで消費電力を削減する」へ，そして現在は「電力消費量を維持したままで性能向上を図る」というように研究の意味づけが変化してきたような気がする．誤解してはいけないことは，省電力は設計における制約条件であって目的関数ではないことである．消費電力を無視してはならないが，アーキテクチャ研究者に本来求められているのはプロセッサの性能向上である．

参考文献

1）Chandrakasan, A. P., Sheng, S. and Bordersen, R. W. : Low-Power CMOS Digital Design, IEEE Journal of Solid-State Circuits, Vol.27, No.4 (1992).

2）Murakami, K. and Magoshi, H. : Trends in High-Performance, Low-Power Processor Architectures, IEICE Transactions on Electronics, Vol. E84-C, No.2 (2001).

3）Inoue, K., Moshnyaga, V. G. and Murakami, K. : Trends in High-Performance, Low-Power Cache Memory Architectures, IEICE Transactions on Electronics, Vol.E85-C, No.2 (2002).

4）Kao, J., Narendra, S. and Chandrakasan, A. : Subthreshold Leakage Modeling and Reduction Techniques, International Conference on Computer-Aided Design (2002).

5）Li, Y., Parikh, D., Zhang, Y., Sankaranarayanan, K., Stan, M. R. and Skadron, K. : State-Preserving vs. Non-State-Preserving Leakage Control in Caches, Design, Automation and Test in Europe (2004).

6）Special Issue on Better than Worst Case Design, IEEE Computer, Vol.37, No.3 (2004).

（平成 17 年 9 月 1 日受付）