5E–2

チップマルチプロセッサにおける 動的なキャッシュ-メインメモリ間データ圧縮の検討

滝田 裕, 坂井 修一, 田中 英彦
東京大学大学院工学系研究科
{takita,sakai,tanaka}@mtl.t.u-tokyo.ac.jp

1 はじめに

半導体技術の進歩により、2003 年頃には約一億個のトランジスタを一つのチップに載せることが可能になる。この大幅に増加した半導体資源を利用して CPU の性能を向上させる方法として、単一チップに複数の CPU を載せた Chip Multi-Processor(CMP) が注目されている。CMP では逐次プログラムを適当な大きさのスレッドに分割し、それらのスレッドを投機実行することで実行の高速化を行うが、CPU 台数に見合った性能向上が得られにくい。また、CMP では複数の CPU が同時に load/store を行うため、メインメモリへのアクセスは通常の CPU と比較して多くなる。

そこで本稿では、CMP 内の一部の CPU を利用してキャッシュ-メモリ間を流れるデータを圧縮しメモリバンド幅を仮想的に拡大させる手法を検討する。

2 Chip Multi-Processor

CMP は 1 つのチップに複数の CPU を集積したもので、既存の CPU デザインを流用できるため現在の CPU に広く用いられている Super Scalar Processor よりも容易に設計できる。また、今後の CPU の主流になると言われている VLIW と異なり、コンパイラによる静的なスケジューリングを行わなくてもそれなりの実行性能が得られる。

ここで扱う CMP は Stanford Univ. の Hydra[1] と同様、一つのチップの上に複数の CPU を載せバスで結合したものを想定しており、一次キャッシュが各 CPU 毎に存在し、二次キャッシュは CPU バスを通じて共有されている (図 1 参照)。

Symmetric Multi Processor(SMP) として動作するが、CPU および共有二次キャッシュ間のデータ転送が高速に行えるため、一般的な SMP よりも台数効果を出しやすい。また、各 CPU のレジスタファイルを転送するバスを用意することで、高速なタスクの生成が可能である。

A Study of On-the-fly Data Compression between Cache and Main Memory on Chip Multi-Processor
Hiroshi TAKITA, Shuichi SAKAI, Hidehiko TANAKA
Graduate School of Engineering, University of Tokyo

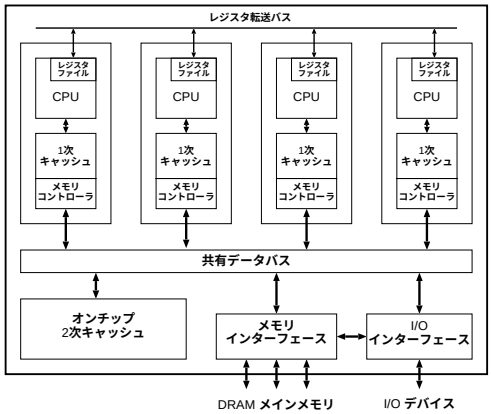

図 1: Chip Multi-Processor の構成

CMP で逐次プログラムを高速化する場合、与えられたプログラムを適当な大きさのスレッドに分割し、各スレッドを投機的に CPU に割り付けて並列実行しているが、投機実行が成功する率が低いため多くの CPU の実行時間は無駄に使われている (図 2 参照)。

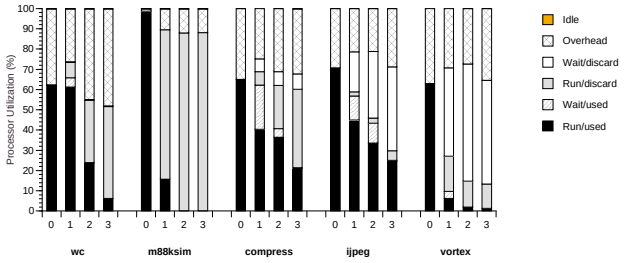

図 2: Hydra[1] における CPU 利用率

3 デバイステクノロジー予測

Moore's Law が今後も維持される場合、CPU の性能は 10 年後に現在の 100 倍程度なると予想される。

SIA の半導体技術予測 [?] によると、10 年で 1 チップに入るトランジスタ数は約 30 倍、ローカルクロックは 5.6 倍向上するため、チップ内において CPU 性能を 100 倍にすることは可能である。しかし、同じ 10 年間でチップ外バスクロックは 2 倍、パッケージからボードに出せるピン数は 2.8 倍にしかならない。周波数の向上とピン数の増加だけでは Moore's Law に沿ったチッ

プ外部からのデータ供給能力の向上は困難であるため、データ供給機構に何らかの対策が必要となる。

4 動的なデータ圧縮

図 2 から、CMP では投機実行に使用する CPU の数を少々変化させても、実行性能に大きな影響が出ないと推察できる。そこで外部からのデータ供給能力を向上させるため、キャッシュからメインメモリへデータを送るときに CMP 内の一部の CPU を用いてデータを圧縮し、圧縮したままのイメージをメインメモリに格納する手法を考える。

4.1 システム構成

動的なデータ圧縮を行う CMP の内部構成は図 3 の通り。

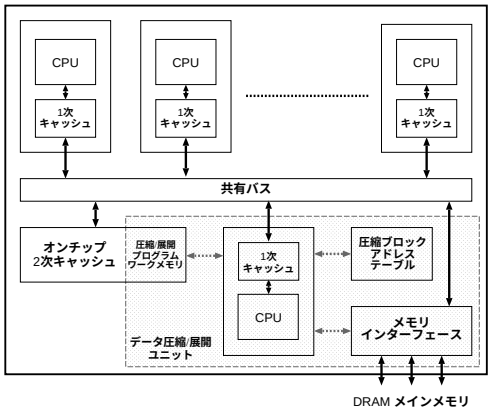

図 3: システム構成

データ圧縮 (展開も) を行うユニットは、CPU と 2 次キャッシュ上のワークメモリ、圧縮ブロックアドレステーブル、メモリインターフェースからなる。

アドレスの管理は、キャッシュ内では一般的な CPU 同様キャッシュライン (64~256bytes) 単位で行うが、メインメモリ上では圧縮ブロック (圧縮効率を考え 16kbytes 以上) 単位で行う。そのため圧縮されたブロックのメインメモリ上での先頭アドレスと圧縮された後のブロックのサイズ、チップ内で使用される物理アドレスを対応付けるためのテーブル (圧縮ブロックアドレステーブル) をデータ圧縮ユニット内に用意する。

データの圧縮/展開ルーチンは 2 次キャッシュ上に用意されたワークメモリにコードを格納しておく。

4.2 データ圧縮/展開の流れ

2 次キャッシュからメインメモリへデータを書き戻す時の圧縮動作は以下の通り。

1. 追い出されるキャッシュラインの属する圧縮ブロックを圧縮ブロックアドレステーブルを元に探し、2 次キャッシュ上のワークメモリに展開して読み込む
2. 展開されたデータに追い出されるキャッシュラインを上書きする
3. できた圧縮ブロックに圧縮をかけ、メインメモリに書き込む
4. 圧縮ブロックの圧縮されたあとのサイズと格納されたメインメモリの先頭アドレス、物理アドレスを圧縮ブロックアドレステーブルに書き出す

また、メインメモリから 2 次キャッシュへデータを読み込む時の展開動作は以下の通り。

1. 読み込むデータのあるキャッシュラインの属する圧縮ブロックを圧縮ブロックアドレステーブルを元に探し、2 次キャッシュ上のワークメモリに展開して読み込む
2. 必要なキャッシュラインだけキャッシュに格納する

5 まとめ

本稿では、投機実行時の CMP の CPU の利用状況とローカルクロックと外部バスクロックの格差に着目し、CMP においてキャッシュ-メモリ間を転送されるデータを CMP 内の一部の CPU を利用して動的に圧縮する方法を検討した。今後は圧縮を適用した場合のアクセスレイテンシの変化や、効率のよい圧縮ブロックサイズ等を調べる予定である。

参考文献

[1] Kunle Olukotun, Basem A. Nayfeh, Lance Hammond, Ken Wilson, and Kunyung Chang. The case for a single-chip multiprocessor. In *Proceedings of the 1996 International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-VII)*, pp. 2–11, October 1996.