1A-1

FPGAベースアクセラレータ向けメモリプリフェッチ機構の検討

高前田（山崎） 伸也 †1†2　吉瀬 謙二 †1
東京工業大学 †1　日本学術振興会 特別研究員 †2

1 はじめに

より高い性能とより良い電力効率の達成を目的に，従来のCPUに加えてGPUやFPGAなどのアクセラレータを組み合わせたヘテロジニアスな計算機がスーパーコンピュータをはじめとして，広く普及しつつある．

FPGA上のアクセラレータで，FPGAがチップ内にもつローカルメモリの容量よりも大きなデータを扱う場合，チップ外部とチップ内メモリとの間でデータの入れ替えを行う必要がある．そのため，チップ外のメモリを強く意識したハードウェア構成を取る必要があり，高性能なアクセラレータを容易に開発することを困難にする要因の一つとなっている．

本稿では，FPGAアクセラレータにおけるアプリケーションに特化したプリフェッチ機構を提案する．FPGAに搭載されるアクセラレータのRTL記述を静的に解析することにより，メモリアクセスに関連する部分を複製し，アプリケーションに特化したプリフェッチャーの回路を生成する．そしてそれを元のアプリケーションのカーネルと独立に動作させることにより，チップ外へのメモリアクセスを先行させるハードウェアについて議論する．

2 アプリケーション特化プリフェッチ機構

図1に，我々が提案するアプリケーションに特化したプリフェッチ機構を持つFPGAアクセラレータの構成を示す．アクセラレータはアプリケーションのカーネルとキャッシュ，そしてアプリケーション特化プリフェッチャー(ASP: Application Specific Prefetecher)から構成される．キャッシュを用いることでメモリシステムをアプリケーションカーネルに対して抽象化している．カーネルはキャッシュを介して計算に必要なデータを外部から取得し，またキャッシュに書き戻す．

本稿では，アプリケーション特化プリフェッチャーをアプリケーションのカーネル中で，メモリアクセスに関連する部分を切り出した回路と定義する．そのため，元々のカーネルよりも小規模で，より高い周波数で動作させることが可能であると考えられる．特に，カーネルのアクセスの系列がメモリ上のデータに依存しない場合，アプリケーション特化プリフェッチャー側ではカーネルが本来含む大規模な演算ユニット群を持たないため，小型化が達成しやすい．

本研究では，アプリケーション特化プリフェッチャーを

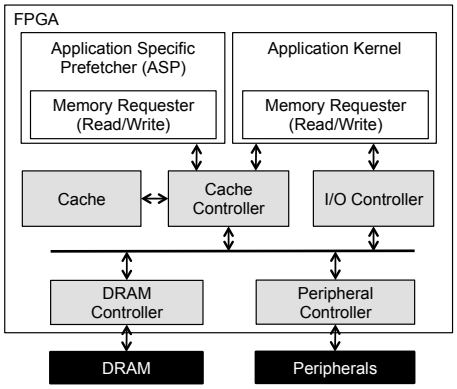

図1: アプリケーション特化プリフェッチ機構をもつFPGAアクセラレータ

カーネルやキャッシュよりも高い周波数で動作させ，キャッシュに対するメモリリクエストを先行的に発行することにより，メモリ性能の向上を目指す．以下にアプリケーション特化プリフェッチャーの生成方法について述べる．まず，カーネルのHDL記述から抽象構文木(AST)を生成し，そこから各変数の定義木を生成する．次に，キャッシュアクセスに用いられるアドレス信号・リクエスト信号・応答信号などの定義木に含まれる変数の和集合を生成する．そして，その集合に含まれる変数の定義木が含む変数を集合に追加していき，変数の集合が変化しなくなるまで，繰り返す．最後に，集合中のそれぞれの変数の定義木をHDLコードに変換し，プリフェッチャーのHDLソースコードを生成する．

3 評価

本稿では，初期評価として，Verilog HDLで記述した簡単なベンチマークを用いて，提案手法による性能向上の度合いを評価する．性能およびキャッシュヒット率をIcarus Verilog[1)]を用いてシミュレーションにより評価する．ベンチマークにはベクター加算を用いた．キャッシュには，C++で記述したサイクルレベルのタイミングシミュレータをVPI (Verilog Programming Interface)を介してHDLシミュレーションに組み込み使用した．キャッシュの構成は，ラインサイズを64バイト，ウェイ数を4，キャッシュ容量を64Kバイト，アクセスレイテンシを1とした．メインメモリには，アクセスレイテンシは64サイクル固定としたシンプルなモデルを用いた．ベクター加算の扱うデータのメモリフットプリントは192Kバイトとした．アプリケーション特化プリフェッチャーの動作周波数をカーネル・キャッシュの1.2倍から2.0倍まで変化させて，

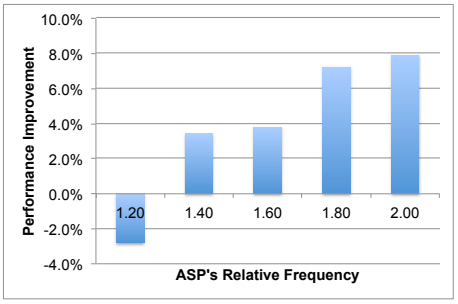

図 2: 性能向上率

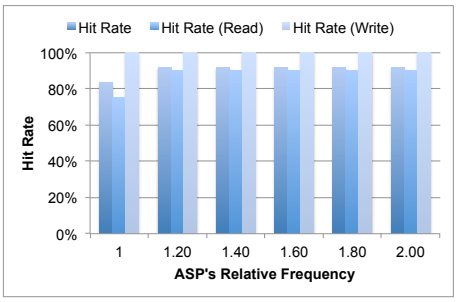

図 3: キャッシュヒット率

性能の変化を測定した．

図 2 にアプリケーション特化プリフェッチャーを用いない場合の性能を基準とした，それぞれの動作周波数におけるアプリケーション特化型プリフェッチャーによる性能向上率を示す．また，図 3 にそれぞれの構成におけるキャッシュヒット率を示す．アプリケーション特化プリフェチャーの動作周波数が 1.2 倍の場合を除き，すべての場合で性能向上を達成した．動作周波数が上昇するにつれて性能向上率が向上し，2.0 倍の場合においては 7.9%の性能向上を達成した．動作周波数が 1.2 倍の場合において性能が低下した理由としては，動作周波数が低いためにプリフェッチの開始を十分に早期に開始することができず，カーネルのリクエストに対して有効な支援ができなかったことと，アプリケーション特化プリフェチャーを追加したことにより発生したキャッシュのアクセスポートに対する競合が発生したことが挙げられる．後者を回避するには，カーネルのリクエストを優先し，カーネルからリクエストが発行された場合には，プリフェッチャー側の処理をアボートするなどの処置を施すことなどが必要である．図 3 に示すとおり，プリフェッチャーの追加によりキャッシュヒット率自体は向上しているため，ポートやラインなどの競合を回避することにより，より高い性能を達成できるものと考えられる．

4 関連研究

FPGA 向けのメモリシステムの最適化の研究としては，Samuel ら [2] による，高位合成言語で記述されたカーネルのコースコードを解析し，オフチップ SDRAM へのメモリアクセスを並べ替えることにより，メモリバンド幅を有効利用する方式や，Eric ら [3] による抽象度の高いメモリモデルを用いてアプリケーションを記述し，外部メモリとのカーネルの間にキャッシュとデータ転送機構を自動的に挿入するフレームワークの CoRAM などが挙げられる．前者は，高位合成系をターゲットしており，またループ中のインデックスにのみ着目して最適化を施す点で本研究と異なる．後者は，アプリケーションの記述を容易にする点では本研究とは類似しているが，キャッシュのデータの先読み等は行わない点で本研究とは異なる．

また，マルチコアや SMT プロセッサ上で，本来のアプリケーションのスレッドとは別に，キャッシュのプリフェッチを行うことを目的としたヘルパースレッディングという手法がある [4, 5]．本研究は，FPGA アクセラレータのアプリケーションに対するヘルパースレッディングととらえることも可能であり，これらの研究で提案された手法は同様に活用できると考えられる．

5 まとめ

本稿では，FPGA アクセラレータ向けアプリケーション特化プリフェチャーの生成手法および，プリフェチャーによる性能向上率の初期評価を行った．今後の課題として，より現実的なアプリケーションを複数用いた評価を行うこと，プリフェチャーの回路面積などの評価などを行うことが不可欠である．また，既存のプリフェッチ技術に対する優位性を定量的に評価する必要がある．加えて，プリフェチャーの動作周波数を上昇させずに先行実行させるために，カーネルの HDL 記述からメモリアクセスに関連する状態遷移を切り出し，カーネルの状態の先読みを行う手法の検討を行いたい．

参考文献

1) Stephen Williams and Michael Baxter. Icarus verilog: open-source verilog more than a year later. *Linux J.*, Vol. 2002, No. 99, pp. 3–, July 2002.

2) Samuel Bayliss and George A. Constantinides. Optimizing sdram bandwidth for custom fpga loop accelerators. In *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*, FPGA '12, pp. 195–204, New York, NY, USA, 2012. ACM.

3) Eric S. Chung, James C. Hoe, and Ken Mai. Coram: an in-fabric memory architecture for fpga-based computing. In *Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays*, FPGA '11, pp. 97–106, New York, NY, USA, 2011. ACM.

4) Jiwei Lu, Abhinav Das, Wei-Chung Hsu, Khoa Nguyen, and Santosh G. Abraham. Dynamic helper threaded prefetching on the sun ultrasparc cmp processor. In *Proceedings of the 38th annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 38, pp. 93–104, Washington, DC, USA, 2005. IEEE Computer Society.

5) Md Kamruzzaman, Steven Swanson, and Dean M. Tullsen. Inter-core prefetching for multicore processors using migrating helper threads. In *Proceedings of the sixteenth international conference on Architectural support for programming languages and operating systems*, ASPLOS '11, pp. 393–404, New York, NY, USA, 2011. ACM.