3A-07

OSによるマルチコアアクセラレータ上でのパイプライン並列処理の実行時制御機構

小柴 篤史† 濱田 槙亮† 大城 研治† 天野 英晴‡ 並木 美太郎†

東京農工大学† 慶應義塾大学‡

1 はじめに

マルチメディア等のストリーミング処理の高速化にはアクセラレータによるパイプライン並列処理が有効である．しかし従来の OS では，デバイスドライバの制御オーバヘッドが処理性能を低減させる，パイプライン並列処理の開発・実行環境が限られている等の課題がある．本研究では，複数アクセラレータをカーネル内で統一的に制御するパイプライン並列処理の実行時制御機構を提案する．提案機構は OpenCL で記述されたプログラムからパイプラインを構築し，パイプライン中のタスクの依存関係に応じてデータ転送とタスク実行を制御する．このカーネル単体の実行時制御によりコンテキストスイッチを抑制する．本評価では，提案機構がデバイスドライバの制御オーバヘッドを 86.2%削減することを示す．

2 アクセラレータ上でのパイプライン並列処理と課題

近年，特定の処理に特化したアクセラレータを用いたヘテロジニアス・コンピューティングが注目されている．特にアクセラレータを用いたパイプライン並列処理は，JPEG デコーダ等の複数のタスクで構成されるストリーミングアプリケーションの高速化に有効である．各タスクを適切なアクセラレータに割り当て，アクセラレータ間でパイプライン式にデータを渡しながらタスクを並列実行することで高い処理性能を達成する．

Linux などの汎用 OS でパイプライン並列処理を行う際，従来のデバイスドライバによるアクセラレータ制御が課題となる．パイプライン並列処理では複数のアクセラレータ，DMA を頻繁に制御する必要があるが，従来の OS ではこれらをユーザプロセスがデバイスドライバを介して行う，そのためユーザ/カーネル間のコンテキストスイッチが頻発し，制御オーバヘッドが増大する．また，パイプライン並列処理のプログラミング環境・実行環境として OpenCL ではパイプライン並列 API の仕様が規定されているが，仕様に基づく OpenCL ランタイム機能の実装はベンダに依存する．そのため既存のランタイム機能は適用できるアクセラレータが限られ，パイプライン処理で用いられる多種アクセラレータへの適用は難しい．

既存研究においてアクセラレータを対象とした資源管理機構やプログラミングモデルが提案されている．しかし，これらは GPU のみを対象にしている[1]，パイプライン並列制御はサポートしていない[2]等の課題がある．

本研究は，汎用 OS 環境におけるマルチコアアクセラレータを用いた高効率なパイプライン並列制御の実現を目標として，パイプライン並列処理向けの新しいアクセラレータの資源管理機構と，パイプライン並列プログラムの開発・実行をサポートする OpenCL ランタイムライブラリを提案する．

3 パイプライン並列処理の実行時制御機構

図 1 に，本研究で想定するヘテロジニアスマルチコア環境と，提案する OS 機構である Pipeline Parallel Manager (PPM) および OpenCL ランタイムの全体構成を示す．本環境では，ホスト CPU とアクセラレータはそれぞれキャッシュおよびローカルメモリを持ち，メインメモリを共有する．また，各メモリ間のデータ転送のための DMA を持つ． OpenCL

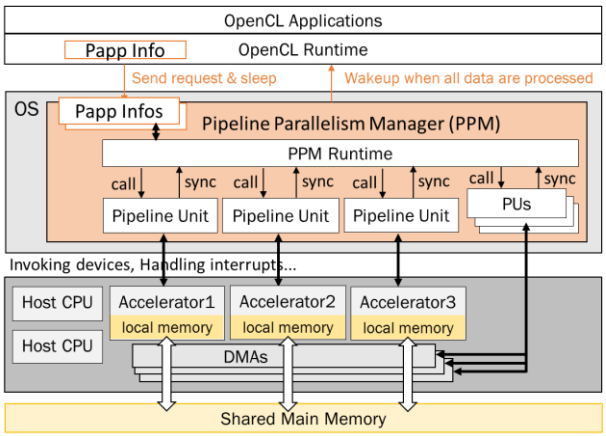

図 1 提案手法の全体構成

Towards Kernel-based Runtime Management of Multi-core Accelerators for Pipeline Processing

Atsushi KOSHIBA†, Shinsuke HAMADA†, Kenji OHSHIRO†, Hideharu AMANO‡, Mitaro NAMIKI†

†Tokyo University of Agriculture and Technology
184-8588, Tokyo, Japan

‡Keio University
223-0061, Yokohama-shi, Japan

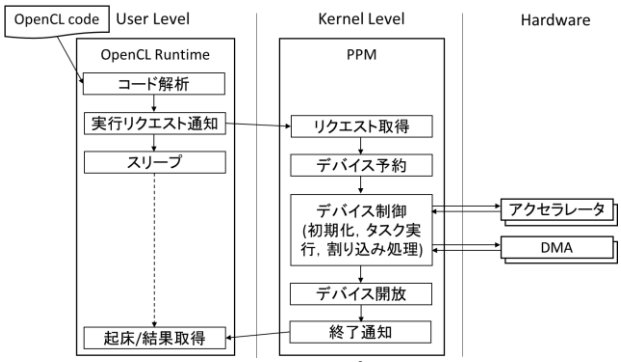

図 2　PPM を用いたパイプライン処理の流れ

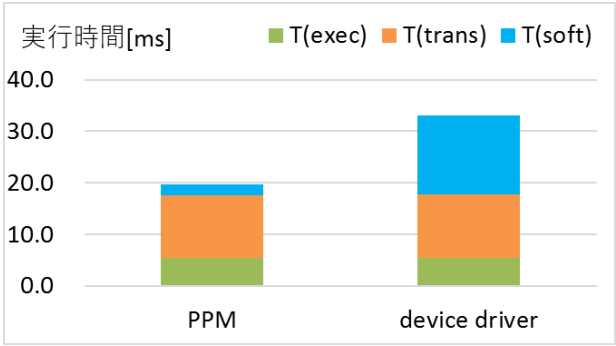

図 3　アルファブレンダの実行時間

ランタイムは，OpenCL で記述されたパイプライン並列プログラムの解析と，PPM へのタスク実行通知を担う．PPM は実行時制御部 (PPM Runtime) とハードウェア依存部 (PU) で構成され，PPM Runtime は OpenCL ランタイムから得たタスク情報に基づきデバイス (アクセラレータ，DMA) を制御する．PU は各デバイスのハードウェア依存の処理 (初期化，タスク実行，割り込み処理) を共通の命令 (init(), exec(), done()) に抽象化した関数群で，PPM Runtime は PU を用いて種類の異なるアクセラレータを統一的に制御する．

図 2 に PPM を用いたパイプライン処理の実行フローを示す．まず，ユーザはプログラムを実装する際，OpenCL API を用いてパイプライン並列処理の実行情報 (パイプラインを構築するデバイスの数と種類，入出力データ，タスク間データフロー，実行一回ごとの演算データサイズ) を記述する．OpenCL ランタイムは実行プログラムの OpenCL API 呼び出しを解析してこれらの情報を抽出し，PPM に実行リクエストとして通知する．PPM Runtime はリクエストを取得し，パイプライン処理に使うデバイスを予約，実行を開始する．実行中は PU を介して各デバイスを制御し，全てのデータを処理したのち OpenCL ランタイムへ通知する．このように各デバイスをカーネル内で直接制御することで，ユーザ/カーネル間通信のオーバヘッドを削減する．

パイプライン並列処理では各タスクの出力データが後続タスクの入力データとなるため，タスク間の依存関係を考慮した実行時制御が必要となる．そこで PPM Runtime は，実行中のデバイスの状態を Running，依存タスク待ちを Wait で表現し，状態遷移に基づき制御する．実行時，あるデバイス上でタスクが終了した時，次のデータが処理可能な場合は Running，他のタスクの終了を待つ必要がある場合は Wait に遷移する．このとき，そのタスクと依存関係にあるタスクで実行可能になった Wait タスクも Running に遷移する．これにより円滑な並列実行を可能にする．

4　実装と評価

本実験では，Zynq の ARM Cortex-A9 プロセッサと画像処理アクセラレータの実チップで構成されるヘテロジニアス・マルチコアプロセッサの評価環境を構築し，ARM プロセッサをホスト CPU として動作する Linux カーネル 4.4.0 のカーネルモジュールとして PPM のプロトタイプを実装した．そして，PPM を用いてアクセラレータ上で 3 段パイプラインの画像処理プログラム (アルファブレンダ) を実行した時の処理速度を計測し，デバイスドライバを用いた場合と比較した．

結果(図 2)より，PPM を用いた制御はデバイスドライバによる制御オーバヘッドを 86.2%削減し，処理速度を 1.66 倍向上している．これは，PPM がユーザ/OS 間のコンテキストスイッチに起因するオーバヘッドをほぼ排除できるためである．

5　おわりに

本研究では，マルチコアアクセラレータを用いたパイプライン並列処理向けの資源管理機能を提案，評価した．今後の課題として，OpenCL ランタイムの実装，マルチコアアクセラレータを用いた場合の処理性能の評価が挙げられる．

謝辞

本研究は，特別研究員奨励費 16J06711 および JSPS 科研費 25220002 の助成を受けたものである．

参考文献

[1] K. Menychtas, K. Shen, and M. L. Scott. Disengaged scheduling for fair, protected access to fast computational accelerators. International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS'14, pp. 301-316. 2014.

[2] C. J. Rossbach, J. Currey, M. Silberstein, B. Ray, and E. Witchel. Ptask: Operating system abstractions to manage gpus as compute devices. In Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles, SOSP '11, pp. 233-248. 2011.