3K-1

Androidを搭載したマルチコアプロセッサシステム向けのOpenCLライブラリの試作

望月 秋人† 佐藤 未来子† 並木 美太郎‡
東京農工大学大学院工学府†/東京農工大学大学院工学研究院‡

1. はじめに

近年，携帯端末をはじめとする高性能な組込みデバイスにはマルチコアプロセッサを搭載したものが増加している．また，ホモジニアスなマルチコアCPUだけでなく，GPUのようなアクセラレータを搭載したものも既に一般的なものとなっており，並列演算を行うためのデバイスは組込み分野においても広がりを見せている．

しかし，ハードウェアが充実し始めている一方で，ソフトウェア面は充実しているとは言い難い．特にGPUのようなヘテロジニアスな構成のものは，ハードウェアとしては主要な並列演算環境に対応しているものであっても，プログラマやユーザにアプリケーションの開発・実行環境が提供されていないことも多い．このように，一般に普及している組込みデバイス向けにヘテロジニアスな並列演算資源を容易に利用するための環境はほとんど無い．

一般的なPCやHPC分野を見てみると，ヘテロジニアスな並列演算環境として，GPUを用いたNVIDIA CUDAやベンダに依存しない共通化された並列演算のためのフレームワークであるOpenCL[1]が既に存在している．OpenCLは組込み向けのプロファイルが規定されており，標準的な並列演算環境として組込み用途にも広まる可能性があると考えられる．

2. 目的

本研究では，並列演算資源を利用するプログラマやユーザの利便性の向上を図ることを目指すため，今後最も増加が見込まれる携帯端末のプラットフォームであるAndroidを対象として，OpenCLに対応した並列演算デバイスをアプリケーションから利用するための機能を提供するJava OpenCLライブラリを作成する．

3. システム概要

本研究で提案するシステムの概要を図1に示す．本システムは，OpenCLデバイスが搭載されたAndroid端末を対象としている．本システムにおけるJava OpenCLライブラリはOpenCLが搭載されたAndroid端末であれば対象は選ばないことを目指している．本研究では，このライブラリを用いることでJNIを通してOpenCLデバイスをアプリケーションから操作するための環境を提供する．

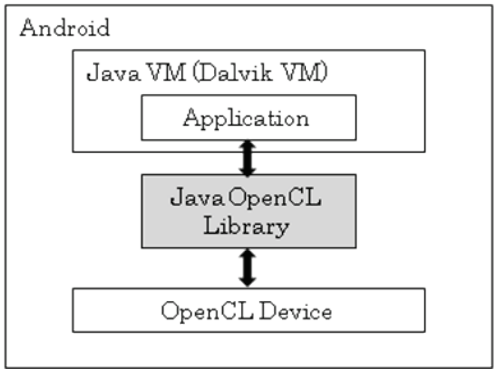

図 1: システム全体概要

4. 試作対象とするOpenCLデバイス

対象とするOpenCLデバイスを含むターゲット全体の概要を図2に示す．実装するJava OpenCLライブラリは，最終的には実行環境は限定しないことを目指しているが，今回は試作として，SH-4Aアーキテクチャのホモジニアスな4コアのマルチコアCPUを，1コア（Android）＋3コア（OpenCLデバイス）のヘテロジニアス構成なシステムとして構築し，この環境で動作するライブラリを試作する．

本研究では，上記のマルチコアCPUの3コア（Core 1～3）を演算専用のOS（Future+MULiTh[3]）が動作する並列演算デバイスとして利用する．この専用OSと汎用OS間の通信を行

A Prototype of OpenCL Library for Android Platform on Multi-core Processor
† Akito Mochizuki
† Mikiko Sato
Graduate school of Engineering, Tokyo University of Agriculture and Technology
‡ Mitaro Namiki
Institute of Engineering, Tokyo University of Agriculture and Technology

う OS 連携機構[2]を介し，専用 OS 上でアプリケーションを実行する．本 OS 間の通信手順を OpenCL API にすり合わせることで，専用 OS による並列演算環境を OpenCL デバイスとして利用できるようにする．なお，本試作では，SH アーキテクチャに対応したコンパイラでコンパイルされたネイティブカーネルのみの対応とした．そのため，今回作成する OpenCL 実装では，ネイティブカーネル実行に必要な API のみを提供する．

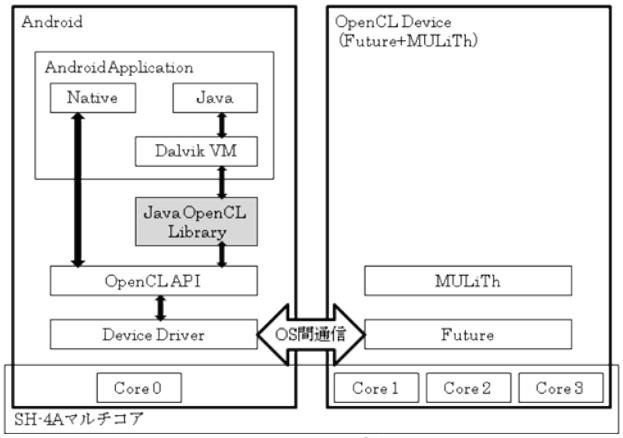

図 2: 実行環境概要

4.1 Java OpenCL ライブラリ設計概要

Java OpenCL ライブラリはネイティブカーネル実行のための API を JNI を介して呼び出す．Java アプリケーションへ提供する API に関しては，実装する OpenCL API と一対一で対応づける．そのうえで，関数名や変数名は OpenCL の仕様と同じ名前のものを Java で利用できるようにする．

4.2 OS 連携機構と OpenCL API の関係

OS 連携機構を用いて専用 OS でプログラムを実行するには，大きく分けて表 1 の左側の手順が必要となる．これらの操作を表 1 の右側に示す OpenCL API で行えるようにすることで OpenCL デバイスとして実装した．たとえば，ホストで用意したカーネルのアドレス，カーネル自身がとる引数を指定し，clEnqueueNativeKernel を呼ぶことで OpenCL デバイスへのプログラムの転送と実行を行う．

表 1: OS 連携機構と関連する OpenCL API の対応

OS 連携機構の実行手順	OpenCL API
デバイスオープン	clGetDeviceIDs
汎用・専用 OS の共有メモリ領域にバッファ確保	clCreateBuffer
演算対象のデータを転送	clEnqueueWriteBuffer
専用 OS プロセス生成 バイナリ転送・実行	clEnqueueNativeKernel
演算結果読み出し	clEnqueueReadBuffer

4.3 対象環境での JNI のオーバヘッドの評価

実行環境において，OS 連携機構を介して専用 OS 側のプログラムを実行する機能を C 言語から呼び出して実行した場合と，Dalvik VM 上のアプリケーションから呼び出して実行した場合の実行時間を比較した．評価に用いた専用 OS 側のプログラムは同じものであり，専用 OS 側の実行結果を文字列として返すものである．

結果を表 2 に示す．実験環境においては JNI を用いた場合のオーバヘッドは 130msec (7%) 程度であった．この値は比較的大きいが，プログラム全体に占めるデバイス側での処理量が増えるほど，オーバヘッドが占める比率が下がると考えられる．

表 2: 実行時間の比較

	実行時間 (sec)
C 言語からの呼び出し	1.837
JNI を介した呼び出し	1.968

5. まとめ

本研究の成果として，Android アプリケーションから OpenCL デバイスを利用できるようにするためのライブラリを作成した．このライブラリを導入することで，これまで Android アプリケーションでは利用することのできなかった OpenCL デバイスによる並列演算環境を実現することができるようになった．

しかし，今回のシステムでは Android アプリケーションのプログラミングについて，Java の設計方針であるオブジェクト指向が考慮されていない．今後はプログラミングの選択肢を増やし，プログラマの利便性を向上させるためにもオブジェクト指向を取り入れた設計も検討したい．

参考文献

[1] OpenCL
http://www.khronos.org/opencl/ (2011)
[2] 磯部泰徳，佐藤未来子，並木美太郎，マルチコア CPU における OS の資源管理方式の研究，情報処理学会第 72 回全国大会 (2010)
[3] 佐藤未来子，磯部泰徳，十山圭介，野尻徹，入江直彦，内山邦夫，並木美太郎，”汎用ホモジニアスマルチコアプロセッサにおける OS とスレッドライブラリ”，情報処理学会第 20 回コンピュータシステムシンポジウム (2008)