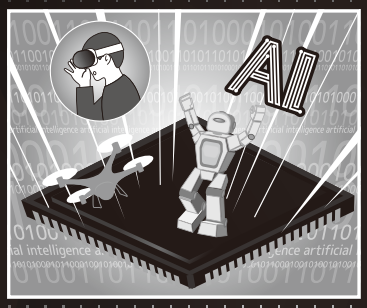
2nm時代到来!

AI，RISC-V，ベクトル・プロセッサ…

新時代のプロセッサ開発技術



第4回　アクセラレータ向け開発環境の今

西村 成司

● 連載概要

本連載では電子システムのアプリケーションを支える屋台骨のプロセッサについて，市場と技術の最新動向を解説します．半導体の中でも，その最先端を走るプロセッサを支える技術を知ることにより，半導体，ひいては電子システムのトレンドを把握することができ，ソフトウェアを開発する立場でも，より多面的にハードウェアからソフトウェアまで含めたシステムを捉えることができるようになると思います．

● 今回のテーマ

FPGA（Field Programmable Gate Array）やAIプロセッサの登場など近年ではハードウェアも大きく進化し，それに伴って，CUDAなどのアクセラレータ・プログラミング環境も進化を遂げています．HDL（Hardware Description Language：ハードウエア記述言語）でカーネル関数（アクセラレータで処理する関数）を記述し，煩雑な独自のAPI（Application Programming Interface）を使ってアクセラレータ上のカーネル関数を起動する…といった時代はとうに終わりを告げ，特定のアクセラレータに依存しない標準化された手段でソフトウェアを記述するということが当たり前の時代となりました．本稿ではSYCLを中心に，アクセラレータ向けのソフトウェア開発環境の最新動向についてまとめます．

アクセラレータ向けSDKの歴史と進歩

筆者が最初に触れたアクセラレータ環境は2004年に登場したCray XD1というシステムでした(11)．このシステムは2ソケットのデュアルコアOpteronプロセッサが1つのFPGA（Vertex-II Pro，旧ザイリンクス）を共有するもので，HDLを使ってカーネル関数を開発して専用のAPIでカーネル関数を呼び出すものでした．アクセラレータを使うにはハードウェアの開発と同じ手順を踏む必要があり，当時の率直な感想としてはソフトウェア屋にとっては，とても使いにくそうなシステムだということでした．C++から直接アクセラレータを呼び出すことを可能にした2006年のエヌビディアのCUDAの登場は非常に革新的な進歩でした．

● アクセラレータ・プログラミング環境「OpenCL」

OpenCLは，2008年にアップルによって提案され，Khronos Groupによって標準化が進められている低レベルなアクセラレータ・プログラミング環境です(1)．特定のベンダによらない，アクセラレータ向けの標準APIとして今日では一般的に用いられています．C99ベースの独自言語OpenCL Cでカーネル関数を記述し，ホストAPIを通じてホスト・プロセッサからアクセラレータの制御を行います．ホストAPIにはアクセラレータの初期化，メモリ管理，カーネル関数のコンパイル・リンク，ホスト-アクセラレータ間のデータ転送，カーネル関数の起動，ホスト-アクセラレータ間の同期を行うための関数群が用意されています．OpenCLの登場により初めて，移植性の高いアクセラレータ・プログラムが実現可能となりました．

今日普及しているOpenCL処理系の多くは2020年に発表されたOpenCL 3.0規格に基づいています(2)．この規格ではアクセラレータがサポートすべき最大公約数的な機能としてOpenCL 1.2規格相当が定められており，SVM（Shared Virtual Memory）やデバイスサイド・キューなどOpenCL 2.0以降に標準規格へ採り入れられた機能は全てオプションとされています．また，カーネル関数の記述にはC++言語の機能が利用できるようにC++ for OpenCLが導入されました（もちろん，従前のOpenCL Cも使える）．

● OpenCLが抱える課題

OpenCLは全てのアクセラレータに万能な究極のソリューションである…と言いたいところですが，残念ながら課題もあります．OpenCLは（競合するCUDAと比べて）比較的低レベルなAPIであるため，移植性が高い反面，プログラミングの記述量が多くなりがちであるという課題がありました．例えば，アクセラ

第1回　売れるプロセッサの狙い目（2023年10月号）
第2回　プロセッサ市場の新たな動き（2023年11月号）
第3回　RISC-Vの命令トレース仕様にみるデバッグ＆トレース機能（2024年1月号）