第3章 ISAの理解は性能向上やデバッグの容易さにつながる

Armv8-M命令セット・アーキテクチャの概要

金丸 敦礼

● Armv8-MのISAはArmv6-MやArmv7-Mの上位互換

本章では，Armv8-M命令セット・アーキテクチャ（Instruction Set Architecture：ISA）について説明する前にまず，Armv6-MとArmv7-MのISAを解説します．次に，それらの上位互換として位置するArmv8-Mベースラインと Armv8-Mメインラインの ISAについて説明します．そこで，本章でのArmv6-M，Armv7-M，Armv8-Mと表記している場合は「ISA」を示します．

命令セット・アーキテクチャを理解すべき理由

● ISAはソフトとハードのインターフェース

ISAとは，アーキテクチャを構成する最も重要な要素の1つであり，そのCPUが実行可能な命令の仕様を定義したものです．

ソフトウェア開発者がプログラムを作成し，それをコンパイルすると，そのISAが定義する命令を用いたバイナリ・コードが生成されます．そして，CPUはその命令を解釈し，実行します．

このように，ISAは，ソフトウェアとハードウェア（演算器）の間のインターフェースとして機能します．もし仮に，あるCPUのISAと互換性がある別のCPUを使用する場合，どちらのCPUも同じ命令コードで実行させることができます．つまり，あるCPU向けにコンパイルしたバイナリ・コードをそのまま別のCPUで実行させることができます．これにより多くのソフトウェア資産を再利用できます．

● ISAは処理したい内容で選択

ISAは演算処理の内容，つまり算術演算（加算，減算など），論理演算（AND，OR，XORなど），メモリ・アクセス（ロード，ストアなど），ジャンプ（条件付き，無条件など）などの演算命令の仕様を定義します．また演算内容以外にも，使用レジスタやメモリ・アクセス方法などの仕様も定義します．実行できる命令が少ない場合，CPUの回路規模を小さくできますが，複雑な処理に時間を要することになります．一方で，実行できる命令が多いと，CPUの回路規模は大きくなりますが，複雑な処理も時間をかけずに実行できます．ただし，複雑な処理が必要なければ不要な回路を抱えることになります．いずれにせよ，処理したい内容に適したISAを選択する必要があります．

● ISAの理解は性能向上やデバッグの容易さにつながる

今のコンパイラは非常に優れているため，ソフトウェア開発者はISAの詳細を理解せずとも，CPUの性能を引き出せているかもしれません．しかし，実際にCPUを動かしているのはISAにて定義されている命令である以上，ISAを理解しておくことで，さらなる性能向上や効率性の改善，デバッグの容易さにつながることは間違いありません．

1．Armv6-MとArmv7-MのISA

● Armv7-MはArmv6-Mを包括を備える

Armv6-MはCortex-M0/M0+のISAです．また，Armv7-MはCortex-M3/M4/M7のISAになります．Armv7-MはArmv6-Mを包括しています．また，同じArmv7-MでもCPUによってサポートしている命令セットが異なります．**図1**にArmv6-MとArmv7-Mの一覧を示します．小さな四角が16ビット命令，大きな四角が32ビット命令を示しています．例えば，`WFE`命令や`WFI`命令は16ビットで，`DSB`命令や`ISB`命令は32ビットです．

● 一般的な命令と高度なデータ演算命令を備える

Armv6-Mは一般的な分岐命令やデータ演算命令，ロード・ストア命令をサポートしています．

一方，Armv7-MはArmv6-Mがサポートしている命令に加え，Cortex-M3向けに追加されたデータ演算命令やビット操作命令，Cortex-M4向けに追加されたDSP命令や浮動小数演算命令，Cortex-M7向けの追加の浮動小数点演算命令をサポートしています．