第6章 試しながらステップ・バイ・ステップ！

Cortex-A時代の新コンパイラ LLVM/Clangのしくみと使い方

村井 和夫

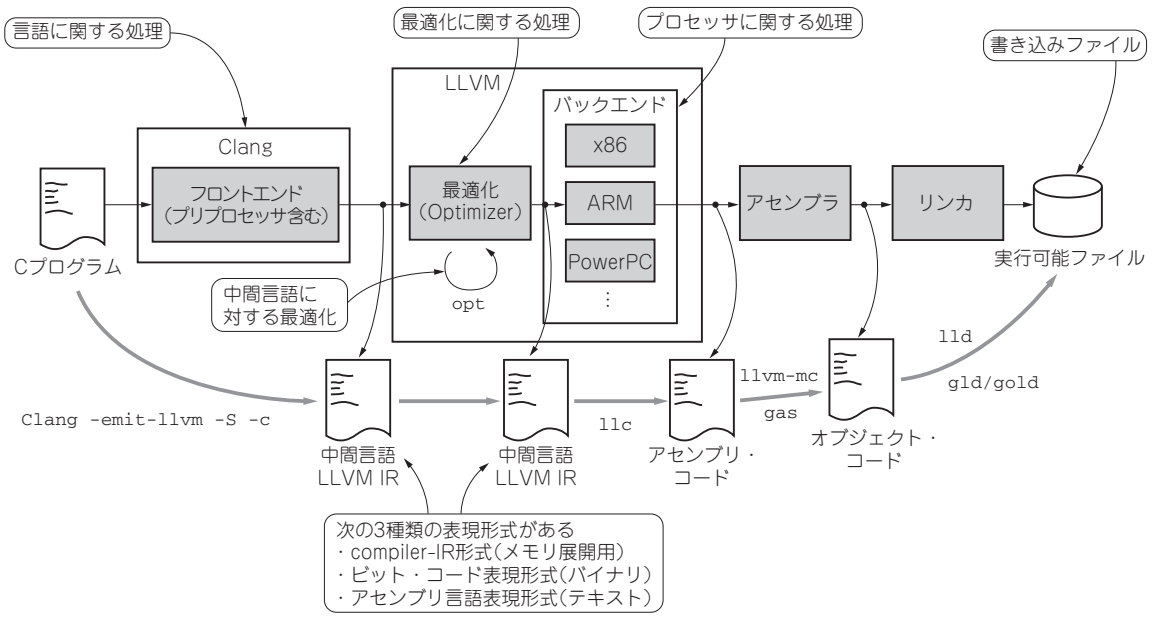

図1 LLVM/Clangコンパイル環境の全体構成

第2部では2000年より静的・動的プログラミング言語の動的コンパイル技術を調査する研究基盤として開発されたLLVMについて紹介します．実際にLLVMを使いながら内部の構造を見ていきます．また，GCCとLLVMのARMクロス開発環境を構築し，生成された実行可能ファイルをCortex-A（BeagleBone Black）やCortex-M（本誌2012年6月号付属FM3マイコン基板）などで実際に動かしてみます．

全体構成

LLVM（広義のコンパイラ）の全体構成を**図1**に示します．全体の処理内容は第1章でも紹介した従来のGCCとほぼ同じですが，コンパイラ本体部周辺の構造が異なります．

LLVMではGCCと同様，各段階ごとに個別にコンパイルすることができます．各段階で行う処理や個別にコンパイルする際に使用する実行コマンドなどを**表1**に示します．

LLVMはGCCと異なり，コンパイラ本体部の三つの機能であるフロントエンド，最適化，バックエンドがそれぞれ独立した構造となっています．

● フロントエンド：言語ごとに用意するプログラミング言語-コンパイラ用言語変換ソフト

プログラミング言語に関する処理を行うフロントエンド部は，LLVMから完全に独立しています．C/C++とObjective-C/C++のフロントエンド部としてClangというコンパイラ・フロントエンドがLLVMと共に提供されています．Clangはソースコードを LLVM IRというLLVMで用いられる中間言語に変換します．

● 最適化：優れた中間言語による独立構造な処理部

LLVMは，Clangなどのフロントエンドで生成され