第1章 昔の教科書を暗記したままで大丈夫ですか？

シンプル・イズ・ベスト

なんてシンプル！新時代のオープンソース・コンパイラの全体像

村井 和夫

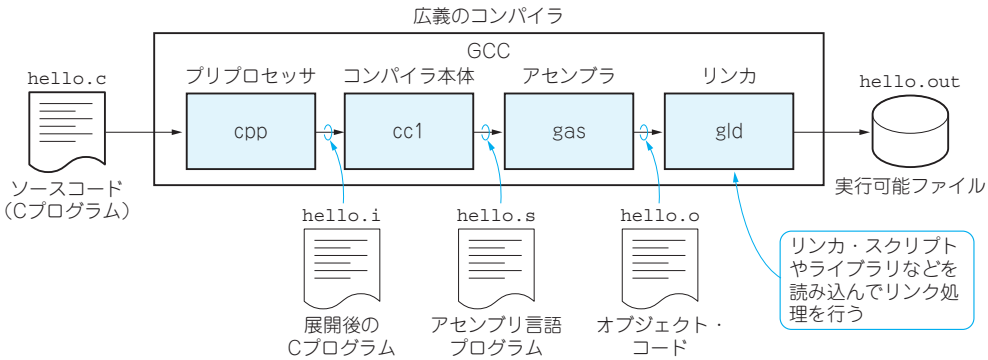

図1 これだけは教科書を思い出しておこう…コンパイラ（広義）の基本構造
従来のGCCの例

教科書のおさらい… 従来のコンパイラの基本構造

まず，一般的なコンパイラの例として，オープンソースのコンパイラとして広く使われてきた従来のGCCの構成について説明します（最近のGCCはコラム1を参照）．

広義のコンパイラ（GCC）の全体構成は，4段階に分けられます．具体的にはプリプロセッサ，コンパイラ本体（狭義のコンパイラ），アセンブラ，リンカの4段階です．**図1**に従来のGCCの全体構成を示します．GCCでは次のように4段階に分けてコンパイルできます．

第1段階：gcc -E cppのみ実行し結果を標準出力に表示
第2段階：gcc -S コンパイラ本体cc1まで実行してhello.sを作成
第3段階：gcc -c アセンブラgasまで実行してhello.oを作成
第4段階：gcc リンカgldまで実行してhello.outを作成

各段階で行う処理内容を**表1**に示します．最も重要なコンパイラ本体は第2段階にあたります．

表1 コンパイラの各段階ごとの処理内容

項目	一般名称	処理内容
第1段階	プリプロセッサ	コンパイル前にソースコードの処理を施す．#includeや#defineなどを処理する
第2段階	コンパイラ本体	ソースコードにコンパイル処理を施し，使用するプロセッサ用のアセンブリ言語に変換する
第3段階	アセンブラ	アセンブリ言語からオブジェクト・コード（機械語）を生成する
第4段階	リンカ	各々バラバラに生成されたオブジェクト・コードやライブラリを一つのファイルに結合して実行可能ファイルを生成する

もうちょっと掘り下げてみる

● キモとなるコンパイラ本体の構造

第2段階のコンパイラ本体では，ソースコードにコンパイル処理を施し，使用するプロセッサ用のアセンブリ言語に変換します．

コンパイラ本体の内部構造は，**図2**に示すとおり，フロントエンド，最適化（Optimizer）とバックエンドの3段階に分かれています．それぞれの処理内容は次の通りです．

▶その1…フロントエンド

フロントエンドでは**図3**に示すように，ソースコードを読み込んで字句解析や構文解析を行って構文木を