連 載

SystemC言語入門

（第２回）

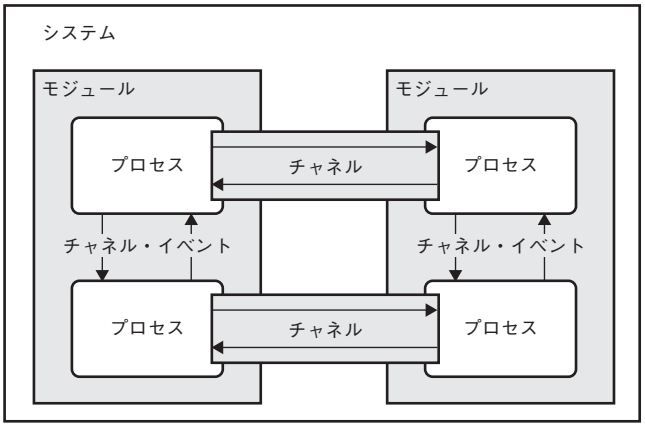
SystemC 2.0の基本構文と記述例

柿本勝，大河内俊雄

本誌2001年12月号のpp.65-74（特集 第4章）では，SystemC言語の特徴やリファレンス・シミュレータの使いかたなどを紹介した．また，本誌2002年1月号の付属CD-ROMにはリファレンス・シミュレータ（SystemC v2.0 for WinNT/2000）を収録した．今回は，記述例を示しながら，SystemC 2.0の基本構文について解説する．時間の概念を含まないUTF（untimed functional）レベルのモデルと，サイクル精度のBCA（bus cycle accurate）レベルのモデルを記述する．（編集部）

本誌2001年12月号に掲載された連載第1回では，SystemCの成り立ち，言語の構成と特徴，リファレンス・シミュレータの入手方法とその使用方法を紹介しました．SystemCはシステム・レベルのモデリングをサポートする言語として開発されているため，ハードウェア向けからソフトウェア向けまで，広範な記述をサポートしています．現時点では，ハードウェアについては，抽象度の高いシステム・レベルからRTL（register transfer level）までのモデリングが可能です．ソフトウェアについては，ネイティブ環境で実行することで，システム・レベルの検証をサポートしています．

連載第2回の今回は，SystemC 2.0の言語の基本構文とその記述方法について，主にハードウェア・モデリングに焦点を当てて説明します．



〔図1〕SystemCのシステム構成

SystemCでは，いくつかのモジュールの組み合わせと，それらをつなぐチャネルによって機能動作を表現する．処理機能を表すモジュールとその間の通信を表すチャネルを分けて定義することで，モジュールの独立性を保てるようにしている．モジュールは階層化が可能で，処理単位であるプロセスで構成される．プロセス間は，チャネルに加えてイベントによる通信が可能となっている．

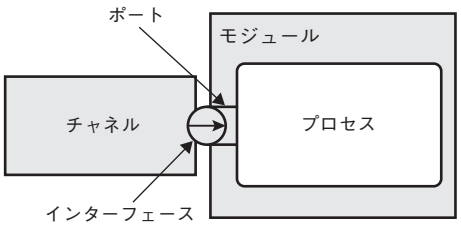
1. SystemCによるモデリングとは

まず，SystemCによるモデリングの考えかたとファイル構成について説明します．

●処理機能のモジュールと通信のチャネルを分離独立

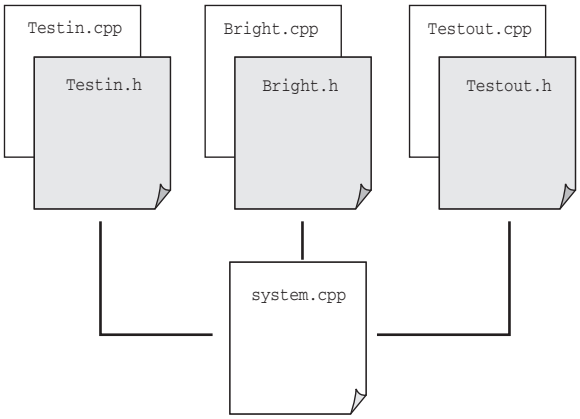
SystemCのモデリングでは，システムをいくつかの「モジュール」とそれらをつなぐ「チャネル」で構成します（**図1**）．処理機能を表すモジュールとその間の通信を表すチャネルを明確に定義して分離することで，モジュールの独立性を保てるようにしています．このため，システムを最適化する過程で通信プロトコルを変更しても，機能モジュールを変更することなく設計作業を進められます．また，モジュールの流用性が高まるので，異なるシステムの間で共通のモジュールを利用しやすくなります．さらに，処理機能を変えず，通信手順だけを変更して性能を上げるといったことも行えます．

モジュールは，同時並行動作を行う「プロセス」と，その間の同期などを行う「チャネル・イベント」で構成され



〔図2〕チャネルとモジュールの接続関係

チャネルは，モジュール間またはプロセス間のコミュニケーションを定義するもので，インターフェースとモジュールに定義するポートを利用して通信を実現する．チャネルには`read( )`と`write( )`などの手続きがインターフェースとして定義されており，モジュールに定義するポートを介してインターフェースを参照して接続する．



〔図3〕SystemCのファイル構成

モジュール記述はヘッダ・ファイルとインプリメンテーション・ファイルで構成される．ヘッダ・ファイルには，モジュールの定義，ポートの定義，使用するプロセスの宣言，初期設定（C++のコンストラクタの部分）を記述する．インプリメンテーション・ファイルには，モジュールの動作をプロセスに分けて記述する．また，最上位階層として，システム・ファイルが必要である．ここでは，メイン関数`sc_main()`を定義して各モジュールとチャネルの宣言を行い，テストベンチとモジュール・インスタンスの接続やシミュレーション実行命令などを記述する．

ます．実現したい処理（機能）はプロセス内に記述します．また大規模なシステムを表現するため，階層化することができます．

通信を受け持つチャネルは，「ポート」と「インターフェース」で構成されます．チャネルはモジュール間またはプロセス間のコミュニケーションを定義するもので，インターフェースとモジュールのそれぞれに定義されるポートを介して通信を行います（**図2**）．

例えばSystemCで用意されているFIFO（first-in first-out）チャネルには，インターフェースとして`read()`と`write()`という手続きが定義されており，それぞれの関数が呼ばれることでFIFOを実現するように動作（プロトコル）が記述されています．モジュールがこのチャネルを使用するときは，FIFOチャネルに直接接続するのではなく，インターフェースを参照することで接続します．インターフェースには`read()`，`write()`などの手続きの宣言のみが記述されているので，モジュール側からはチャネルの実現方式を気にすることなく，FIFOチャネルを使用できます．

●ファイル構成はC++と同じ

SystemCはC++のクラス・ライブラリで構成されているので，C++の基本を理解していると，わかりやすいと思います．モジュール記述はヘッダ・ファイルとインプリメンテーション・ファイルで構成されます（**図3**）．一般にヘッダ・ファイルを“モジュール名.h”，インプリメンテーション・ファイルを“モジュール名.cpp”とします．ヘッダ・ファイルには，モジュールの定義，ポートの定義，使用するプロセスの宣言，初期設定（C++のコンストラクタの部分）を記述します．インプリメンテーション・ファイルには，モジュールの動作をプロセスに分割して記述します．

また，最上位階層としてシステム・ファイルが必要です．ファイル名は通常system.cpp，top.cppなどとします．ここでは，メイン関数`sc_main()`を定義して各モジュールと各チャネルの宣言を行い，テストベンチやモジュール・インスタンスの接続を行い，シミュレーション実行命令などを記述します．

2. SystemCの構文解説

次に，サンプル回路の例を示しながら，処理機能をSystemCでどのように記述していくかについて説明します．

SystemC 2.0では，従来のバージョンより抽象度の高いシステム記述を行えるようになっています．まず機能だけに着目してUTF（untimed functional）レベルで検証を行います．その後，時間概念を盛り込んだTF（timed functional）レベルやサイクル精度のBCA（bus cycle accurate）レベルへと詳細化し，タイミング精度の高い検証を行います．

●サンプル回路はBRIGHTフィルタ

サンプル回路として画像の明るさを制御するBRIGHT