デバイスの記事

アサーションを活用して不ぐあい箇所を即座に特定

後編

――複雑なシーケンスを簡潔に表現するテクニック

赤星博輝

前回は，SERE（Sequential Extended Regular Expression）と呼ばれる記法によって長いシーケンスを表現する方法を紹介した．今回は，シーケンスに名まえを付けて階層的に呼び出す方法や，シーケンスどうしの演算などについて解説する．また，アサーションのチェックを途中でキャンセルする方法についても説明する．（編集部）

最近は「検証が大事」と言われていますが，まだ，検証を設計の“おまけ”と考える人がいることも事実です．その一方で，設計対象が大規模化・複雑化している現状では，新しい検証手法などの導入が不可欠になっていることもまちがいありません．

それでは「検証」とは何なのでしょうか？ ここでは，「RTL記述が設計仕様どおり正しく設計されているかどうかをチェックすること」について考えてみます．このチェックを行うには，どのような方法があるのでしょうか？

RTL設計に対して，考えうるすべてのシミュレーション・パターンをチェックできればよいのですが，現実にはこれを行えないケースがほとんどです．その理由は，現在の大規模な設計では膨大な量のシミュレーション・パターンを作る必要があり，その作成がたいへんということと，その膨大なシミュレーション・パターンを使ってシミュレーションすると膨大な時間が必要になり，限られた設計期間ではシミュレーションを終了することができないということです．

ここでもう1歩進めて，検証は「シミュレーション・パターンを流すこと」ではなく，「設計時に潜り込んだバグを探し出すこと」が目的であると考えてみましょう．そうすると，シミュレーションだけでなく，設計レビューやアサーション，フォーマル・ベリフィケーション（形式的検証）などの手法も有効になってきます．バグ（虫）を捕まえるのに一つの方法だけを使うと，どうしても同じタイプのバグばかり発見する傾向があります．設計時に潜り込んださまざまなタイプのバグを見つけ出すには，違った視点からチェックすることが必要となります．残念ながら，「アサーションを覚えたらすべて安心」ということにはなりません．

それでは，アサーションにはどのような効果があるのでしょうか？ シミュレーションを漁船に例えると，シミュレーション・パターンはどこの漁場をどのように動くのかを決定することに対応します（**図1**）．これに対して，期待値チェックという網で魚を捕まえるのですが，出力を見るだけなので網の目が粗いと考えることができます．ここではすべての魚（バグ）を捕まえたいわけですから，網の目を細かくすることが有効になります．アサーションは，内部の信号やインターフェース部の動きをチェックします．つまり，より細かい網で魚（バグ）を捕まえることができます．

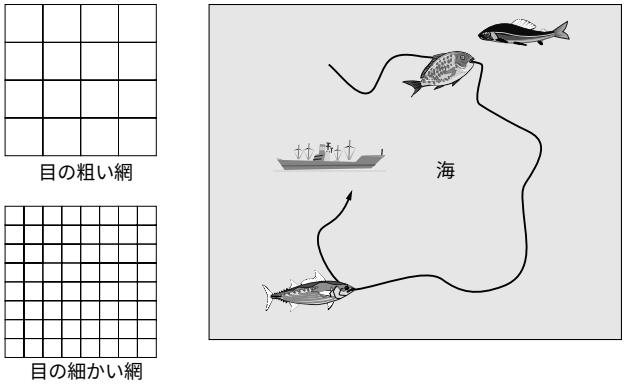


図1　バグの検出とは

漁船の行動に例えると，魚をとることがバグの検出と言える．シミュレーション・パターンは，漁船が海の上をどのように進むか（航路）を決定する．漁船は「期待値チェック」という網で魚をとるが，アサーションによって網の目を細かくすることができる．

アサーションなどを用いて網の目を細かくしても，シミュレーション・パターンの作りかたが悪いと，魚のいない場所を動き回って疲れるだけになりますし，網の目を細かくしすぎると抵抗が増えて進むのが遅く（シミュレーション時間が長く）なります．アサーションを使う場合には，よく考えて検証する必要があります．

● PSL/Sugarの三つの階層の概念を理解する

これまでは，PSL/Sugarを使った実例を挙げながら説明してきましたが，ある程度複雑なことをするためには基本となる考えかたを知っておく必要があります．今回はPSLの基本となる三つの階層（レイヤ）の説明から始めます．

PSLは正確には四つの階層を持っていますが，ここではアサーションを使用するために最低限必要な知識として，三つの階層についてのみ説明します．この階層の知識がないと，PSLの演算子とHDL（VHDL，Verilog HDL）の演算子のどちらを使うのかわからなくなったりするので，以下の三つの階層を頭に入れておきましょう．

- **論理式層（boolean layer）**
- **テンポラル層（temporal layer）**
- **検証層（verification layer）**

図2にこれらの階層と記述を示します．層とPSLの対応を見ていただければ，理解しやすいと思います．

1）論理式層

論理式層は，論理式を記述する層です．この層の記述が単体で存在することはなく，ほかの層から使用されます．論理式なので，ある時点でのプロパティを記述することになります．この層はHDLで記述されるものが基本となります．HDLとしてはVerilogフレーバとVHDLフレーバが使えるようになっています．

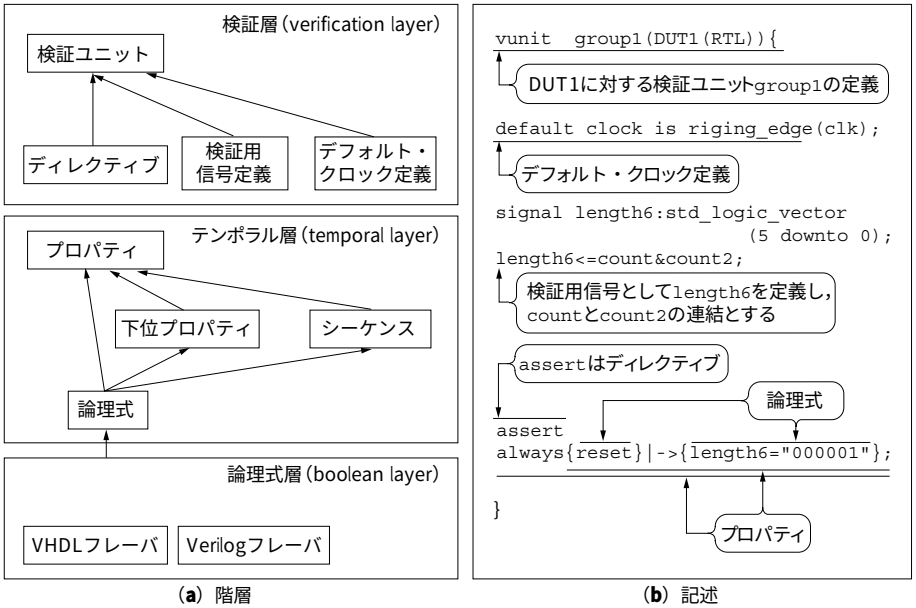
2）テンポラル層

テンポラル層では，複数クロックにまたがる回路のプロパティ（性質）を記述します．そのために，プロパティは以下の三つから構成されます．

- **論理式**
- **シーケンス**
- **下位のプロパティ**

論理式は論理式層で記述するものになり，先ほど述べたようにVerilogフレーバやVHDLフレーバなどを使用します．シーケンスは複数サイクルにまたがる回路の動作になり，前回説明したSERE（Sequential Extended Regular Expression）などで記述します．このシーケンス記述を使うと，シーケンスに名まえを付けて呼び出し，再利用したり，短いシーケンスを組み合わせて長いシーケンスを定義したり，シーケンスに対する演算を行うことで複雑なシーケンスを記述したりすることができます．また，プロパティも階層的に定義することができます．

これらの三つを組み合わせてプロパティの定義を行うことができます．注意する点として，論理式層は各HDLのフレーバを使用しますが，シーケンスなどではPSL/Sugarの



（a）階層　　（b）記述

図2
PSL/Sugarの階層と実際の記述

アサーションを使用するために最低限必要な知識として，三つの階層の概念を理解する必要がある．論理式層は，論理式を記述する層である．テンポラル層では，複数クロックにまたがる回路のプロパティ（性質）を記述する．検証層では，検証ツール（シミュレータやプロパティ・チェッカなど）に「何をどのようにチェックしなさい」という指示を与えるためのディレクティブを記述したり，そのディレクティブをグループにまとめたり，チェックするときの基準となるデフォルト・クロックを定義したりする．