VHDLワンポイント講座⑨

スケーラブル・デザイン

竹本　悟

回路を設計していると，同じような回路の設計を繰り返し行うことがあります．それらの回路はどこかしら違っているのですが，機能はほとんど同じか，よく似ていたりします．VHDLでは，同じような機能をもった回路を一つのentityとして記述できます．

1.generic文を使う

カウンタを考えてみましょう．16進，35進，24進というように，さまざまなカウンタが1枚のボードに載ることがよくあります．回路図を書きながら，「あ〜あっ，めんどうくさい．カウント値が違うだけなら，同じように一括して書けないかしら？」と思ったものです．ディジタル回路ではそのようなことが多く起こります．回路図ではそれぞれ別の回路を書かなければなりません．CADで回路図入力している場合には同じような部分をコピーすることもできますが，コピーした回路はそれぞれに編集しなければなりません．

また，かりにそれぞれに対応するライブラリを作ったとすると，種々雑多なライブラリができてしまいます．こんどは設計者のほうがライブラリを覚えるのがたいへんで，けっきょくだれも使わなくなってしまう…というようなことになります．それに，作るのもたいへんです(**図1**)．

しかしVHDLでは違います．同じような機能をもった回路をたった一つのentityとして記述できるのです．しかも前回説明したように，componentとして宣言すれば，一つのentityの中でさまざまな動作をさせることができます．ライブラリの数は少なく，機能は豊富にという記述ができるのです(**図2**)．

前回の階層の話でも述べましたが，VHDLでは，階層間で入出力信号のほかに，回路の動作を決めるいろいろなパラメータを渡すことができます．architectureの中ではそのパラメータを使うことで動作を変更することができますし，入出力ポートの数も変更できます．この受け渡しに使われるのがgeneric宣言です．今回はgenericについて少し詳しく説明します．

generic宣言で渡されるパラメータは，どのようなタイプでもよく，VHDLで定義できるものなら何でもかまいません．ただし，入出力ポートと違って変数ではありません．必ず定数でなければならないのです．

〔図1〕ライブラリは多すぎると使えない

〔図2〕Entityは変わる

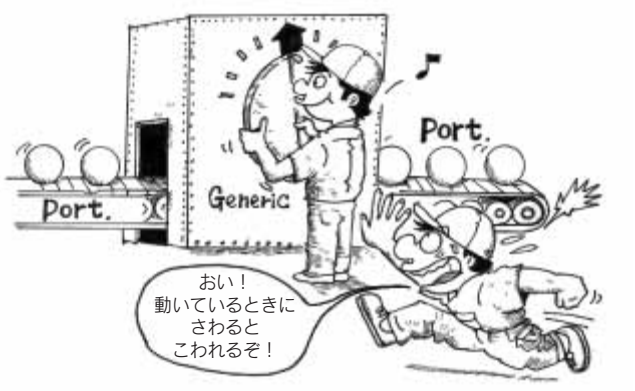
2.generic宣言の効用

entityが受け取るパラメータは，generic宣言で定義されます．ここで定義されたオブジェクトは，entity内ではconstantで宣言された定数と同じように扱われます．変数ではないことに注意してください．動的に変化する変数はパラメータとしては渡せません．もし渡したいなら，ポートとして渡さなければなりません．generic宣言をするときには，通常，初期値を設定します．パラメータが渡されないときには，この値がパラメータとして採用されます．

generic宣言されたパラメータの使い方は，定数としての使い方となんら違いはありません．そこでさまざまな使い方が可能になるわけです．前回の説明で例示したような，配列で定義された入出力ポートの数を変更することもいとも簡単にできるわけです．ただし新たにポートを増やしたり，ポートを削除したりするようなことはできません．

また，if文やcase文を使ってパラメータの値をチェックすることにより，さまざま動作を定義することができます．一つのentityの定義で，あるときはカウンタ，あるときはシフト・レジスタ，というように回路を記述することも可能です．

generic宣言で渡されたパラメータは，その記述が論理合成可能な記述であるかぎり，論理合成結果に影響を与えることはありません．論理合成ツールがVHDLコードを読み込んだとき，generic宣言で渡されるパラメータは定数と同じように扱われるからです．論理合成ツールはgeneric宣言されたオブジェクトを渡されたパラメータの定数を入れ替えて処理するからです(図3)．



〔図3〕genericは変化してはならない

3.generic文の使用例

カウンタを例にします．よく使われる同期式カウンタを考えてみます(リスト1)．任意のカウント値になるようになカウンタを記述してみましょう．

まずは入出力信号です．カウント値は整数で指定してもかまわないのですが，簡単にするため出力信号のビット数で指定できるようにします．つまりnビット・カウンタの記述を考えてみます．

まず，generic宣言で出力信号のビット数を定義します．ビット数ですから，当然，自然数です．そして初期値の設定を忘れないでください．これがデフォルト値になります．そして出力信号の定義を，このgenericで定義されたオブジェクトを使って行います．出力信号は配列(通常はstd_logic_vector)を使って定義します．配列の範囲がgenericで渡された数値によって決まりま

〔リスト1〕カウント値が変更可能なカウンタの例

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity  n_count is
generic( cnt:natural:=4);
                          --上位階層から渡されるパラメータ
port (CLK,EN ,RESET:in std_logic;
     Qout:inout std_logic_vector(cnt-1 downto 0);
     --パラメータによってビット数が変わる
     C:out std_logic
     );
end n_count;

architecture B_count of n_count is
  constant FULL:std_logic_vector(cnt-1 downto 0):=
                                          (others=>'1') ;
begin
 process(RESET,CLK)
   begin
     if RESET='0' then
       Qout<=(others=>'0');     --Qoutはすべて'0'
     else
       if CLK'event and CLK='1' then
         if EN='1' then
           Qout<= std_logic_vector(unsigned(Qout)+1);
         end if;
       end if;       -- end clock
     end if;         -- end reset
   end process;
   C<='1' when Qout=FULL else
      '0';
 end b_count;
```