次は Pythonで 高位合成だ！

Pythonで記述したソースから Verilog HDLソースを出力する高位合成ツール プログラミング言語Pythonではじめる FPGA開発入門

片岡 啓明 Hiroaki Kataoka

いわゆるソフトウェア屋さんの間では，Pythonは手軽なプログラミング言語として人気があります．そんなソフトウェア屋さん達をFPGAの世界に引き込みたいと考え，筆者は高位合成ツールPolyphonyを開発しました．ここではPythonで記述された簡易なMIPSコアをFPGAに実装して動かすまでを紹介します．まずはPythonインタプリタで実行し，高位合成結果をシミュレータで確認し，最終的に実機に実装して動作確認します．

1 Pythonで高位合成

● Pythonベースのコンパイラ Polyphony

今回紹介するのは，プログラミング言語Python（ポリフォニー）で記述されたソースを高位合成できるPolyphonyで，筆者が開発したオープン・ソースのツールです．

Polyphonyでは，Pythonで書いたアルゴリズムを論理合成可能なHDL（現時点ではVerilog HDLのみ）に変換が可能です．ここでは，Polyphonyの特徴と，Polyphonyを使ってFPGAボード上で動作する簡易MIPSプロセッサを実装してみた手順を紹介します．

● Pythonはどんな言語？

本書の読者の中には，Pythonにはなじみのない方も多いと思います．Pythonがどういう言語か簡単に触れておきます．

Pythonの特徴（良い点）として，一般的によくいわれるのは次のような点です．

・コードが読みやすい（可読性）

一般的に，Pythonは可読性の高いプログラミング言語だといわれます．特徴的なのはインデントが構文規則の一部になっており，コードのブロックを規定する点です．これにより誰が書いたコードでも見た目が統一され，読みやすくなる効果があります．

また，Pythonの文化として，書いたコードの処理の理解のしやすさや読みやすさが常に優先されます．もちろん書き手の技量にもよるのですが，こういったことから他の人の書いたコードが他の言語に比較して読みやすいものになる傾向があります．

・難しくない（学習コストの低さ）

Pythonは構文的にはC言語にわりと近い言語です．既にC/C++言語を使っている方であれば，Pythonを使ったことがなくてもそのコードがどういう処理を行っているのか理解するのは難しくないと思います．さらに，書籍やネット上の日本語のドキュメントも豊富で，たとえ分からないことがあってもネットで調べれば大抵のことはすぐに見つかります．実用的なコードを書けるまでおそらくそう時間はかからないでしょう．

蛇足ですが，PythonはMIT（マサチューセッツ工科大学：Massachusetts Institute of Technology）などの大学でもコンピュータ・サイエンス教育用の言語として採用されたりもしているようです．それだけ入門者にもやさしい言語だといえそうです．

・標準ライブラリが充実

Pythonでは言語の標準関数とライブラリが豊富で，すぐに実用的なコードを書き始められます．また，言語標準の要素としてのリストや辞書，それらと組み合わせる標準関数は非常に強力で，Pythonでコードを書く上では欠かせない道具になります．加えて，豊富なライブラリはアルゴリズムの検証やテストを行う際に特に便利に感じるはずです．

標準で高い機能を持つ結果，HDLはもちろんのことC/C++系言語などに比べてコードは自然と短くなります．短いコードは全体の見通しを良くし，それによりバグを混入させる可能性も低く抑えることができるでしょう．

2 Polyphonyの特徴

● Polyphonyを使った作業の流れ

Polyphonyが想定するFPGA開発の流れは次の通りです．

(1) Pythonでアルゴリズムを書く
(2) Pythonコードでの検証
(3) 高位合成（ここでPolyphonyを使用）
(4) 出力後のHDLでの検証

リスト1　階乗計算処理

```
def fact(x):
    y = 1
    n = x + 1
    for i in range(1, n):
        y = y * i
    return y
```

（a）Pythonで記述されたソース