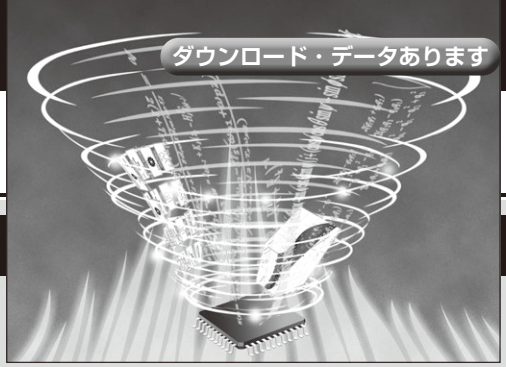

ノイズ除去を例に設計の流れを体験

MATLAB実習 初めてのディジタル・フィルタ作り

篠原 規将

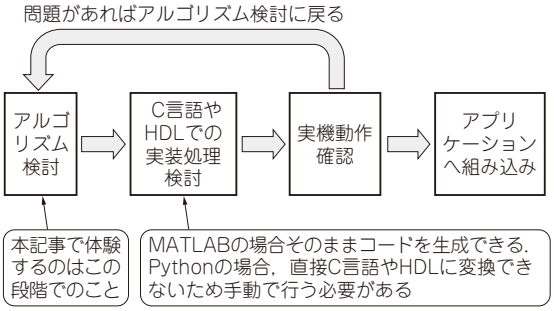

図1　PC上でシミュレーションした後にマイコンなどに実装するアプリケーションの開発フロー

PCでシミュレーション後にマイコンへ移植しやすい

● CやPythonとの比較

MATLABはSimulinkツールを使ってグラフィカルにシミュレーションしたり，プログラミング言語を用いたコードベースでのシミュレーションができます．特にさまざまなノイズやモデルが用意されているため，評価検証にかかる準備が少ないです．

対してC言語だけでシミュレーション環境を構築しようとすると，計算モデルを一から作っていく必要があり，この作業でかなりの時間を要してしまいます．

Pythonの場合，さまざまな処理は関数化されており簡単に扱える一方で，中身がブラックボックス化されているため，HDL（ハードウェア記述言語）に変換してFPGAへ実装したり，C言語に変換してマイコンに実装する場合，一から検討を行う必要があります．

マイコンやFPGA上で動くアプリケーションの開発フローを**図1**に示します．MATLABを使う場合，アルゴリズム検討を行い，その後C言語やHDLへの変換をMATLABの機能を使って行えます．そのためPC上でのシミュレーションをした後に，マイコンやFPGAといった実機への移植が行いやすい環境になっています．

リスト1　MATLABとPythonのコード比較（グラフ表示を行う例）

```
step = 0.001;

x = -pi:step:pi;
y1 = sin(x);
y2 = cos(x);

fig = figure;
grid on;
hold on
plot(x,y1)
plot(x,y2)
xlim([-pi pi]);
ylim([-1 1]);
xlabel('φ [rad]');
ylabel('Amp');
legend('sin(x)','cos(x)');
title('Sin関数、Cos関数グラフ')
saveas(fig,"test01","jpg");

close(fig);
```

（**a**）MATLAB

```
import numpy as np
import matplotlib.pyplot as plt
import japanize_matplotlib

step = 0.001

x = np.arange(-np.pi,np.pi,step)
y1 = np.sin(x)
y2 = np.cos(x)

plt.figure()

plt.plot(x,y1,label = "sin(x)")
plt.plot(x,y2,label = "cos(x)")
plt.xlim([-np.pi,np.pi])
plt.ylim([-1,1])
plt.xlabel("φ [rad]")
plt.ylabel("Amp")
plt.legend()
plt.grid()
plt.title("Sin関数，Cos関数グラフ")
plt.savefig("test01.png")
plt.show()

plt.close()
```

（**b**）Python

● 文法はPythonに似ている

MATLABのMスクリプトとPythonの文法は比較的似ています．そのため事前にPython言語を学んでいるとMATLABのスクリプト作成が行いやすいです．