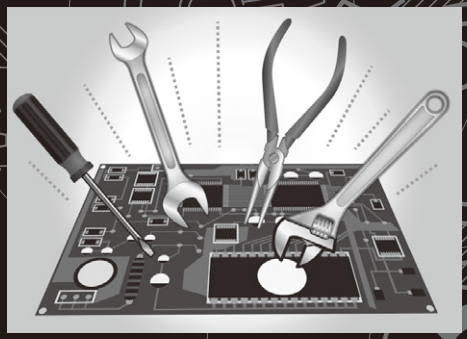
短期連載

組み込みOSチューニング・テクニック



新連載
第1回　移植性：作ったOSの対象とするマイコンを切り替える

鹿取 祐二

```
μT-Kernelソースコード
├─config　コンフィグレーション
├─include　インクルード・ファイル
│　├─sys　システム定義
│　│　└─sysdepend　機種依存部
│　├─tk　OS関連定義
│　│　└─sysdepend　機種依存部
│　├─lib　ライブラリ関連定義
⋮　└─tm　T-Monitor関連定義
```

図1　μT-Kernel3.0ソースコードのフォルダ構成

```
sysdepend　機種依存部
├─＜ターゲット1＞　ターゲット1依存部
├─　…　…
├─＜ターゲットn＞　ターゲットn依存部
└─cpu　CPU依存部
　　├─＜cpu1＞　CPU 1依存部
　　├─　…　…
　　├─＜cpun＞　CPUn依存部
　　└─core　コア依存部
　　　　├─＜core1＞　コア1依存部
　　　　├─　…　…
　　　　└─＜coren＞　コアn依存部
```

図2　機種依存部のフォルダ構成

2023年7月号特集「ゼロから作るOS」で取り上げたTry Kernelは，1500行で学習するため機能を絞り込んであります．ここで紹介するμT-Kernelは，Try Kernelの発展版と言うこともでき，多くの製品に導入されています．今回はあるマイコン向けに作り込んだOSおよび動作プログラムを，ほかのマイコンに載せ替える際に，先人がどのような工夫を施しているのかを紹介します．

本連載では，組み込み向けリアルタイムOSの移植性や応答性の向上などを紹介します．解説にあたり，例としてトロンフォーラムと筆者が公開している組み込みシステム向けのリアルタイムOS μT-Kernel3.0[(1)]のソースコードを取り上げます．

（編集部）

移植性を高める工夫

● C言語を使う

組み込み向けに限らず，OSの移植性を高めるためには，低級言語のアセンブリ言語ではなく，C言語などの高級言語でソースコードが記載されていることが必須となります．近年であれば処理系の最適化技術がかなりのレベルに達していますから，C言語で記述してもアセンブリ言語とほぼ同じ性能を発揮できます．結果，OSの中心的な機能は移植性を重視して，C言語を使って良いと思います．

μT-Kernel3.0のソースコードも上記のような理由から全ソースコードの9割以上がC言語で記述されています．また，必要に応じて下記のようなプリデファインド・マクロを使って，処理系を判別したうえで固有の機能を利用しています．

```
#ifdef __GNUC__
```

どうしても移植性が損なわれる記述を使用する際は，上記のようなプリデファインド・マクロを使って対策することが重要となります．

● 機種依存の部分はフォルダ構成を工夫する

一方，ディスパッチ処理や割り込みのハンドリング処理など，CPU内部レジスタを操作する機種依存の部分はアセンブリ言語を使用せざるを得ないため，その部分に移植性を求めるのは無理と言えます．そのため，機種別にソースコードを用意し，機種依存部分を書くことになります．そのままでは対応機種が増えるごとにソースコードが増加し，混乱してしまいます．

μT-Kernel3.0のソースコードの場合，その機種依存の部分に対しても移植性を配慮しています．それはフォルダ構成です．**図1**に示すように機種依存となる部分は全てsysdependフォルダに纏められており，sysdependフォルダの中は**図2**に示すようにターゲット依存部，CPU依存部，コア依存部のフォルダに分かれています．