第1章

TOPPERS/JSPのよりどころとなっているμITRON4.0仕様

邑中 雅樹

第1章である本章は，TOPPERS/JSPカーネルのベースとなっているμITRON4.0仕様スタンダード・プロファイルの全体像について説明します．

μITRON4.0仕様では，その仕様書の中に【仕様決定の理由】という項目があり，ほかのオペレーティング・システム（OS）仕様よりも「なぜそうなっているのか？」ということがわかりやすい構成になっています．ただし，それが徹底しているわけではなく，仕様策定当時カーネル仕様の検討WG（Working Group）のメンバだった方々に直接聞いてみて初めて理解できることも多々あります．

本章では，TOPPERS/JSPを使いこなすうえで，μITRON4.0仕様書を読み解く前に知っておいたほうが良い話題をなるべく多く取り上げます．なお，μITRON4.0仕様書は，無償でダウンロードできます[注1]．興味のある方は，この機会にぜひご一読ください．

1. OSの3大機能

少し大局的にOSというものを考えてみましょう．OSが提供する機能とは何でしょうか．一般論として，

①ハードウェア・リソースの抽象化
②抽象リソースの管理
③リソースの利用効率の向上

の三つがあると言われています．

上記の三つの機能から独立しているわけではありませんが，

④抽象リソースのアクセス保護

を強調する意見もあります．

世の中には，目的別にたくさんのOSが存在しています．そのいずれも，目的別に最適化された形で，上記の①～③の三つの機能を提供しています．

ここで，①～③の機能のいずれにも「リソース」（資源）ということばが関与していることに注意してください．つまり，OSの機能とは，有限個しかないリソースを複数の仕事が取り合ったりしないよう，最適に割り振るための調停役として存在しているのです．

2. μITRON4.0仕様における方針

続いて，OSが提供するこれらの機能について，μITRON4.0仕様はどのような方針をとっているのか，順に見ていきましょう．

● ハードウェア・リソースの抽象化

LinuxやWindowsなどデスクトップ・パソコンに由来するOS，およびQNXやVxWorksなどPOSIXの影響を受けている組み込みOSは，周辺デバイスも含めたシステム全体のハードウェア抽象化機能を提供しています．そのような世界では，デバイス・ドライバを定義するモデルが存在し，アプリケーションからは統一された方法で――たとえばopenシステム・コールを使ってファイルをオープンするのと同じ方法で，抽象化されたハードウェア・リソースにアクセスすることができます．

しかし，μITRON4.0仕様が提供するハードウェアの抽象化とは，CPUコアに限定した，緩い抽象化にとどまっています．

TOPPERS/JSPカーネルの開発者であり，μITRON4.0仕様の編者でもある名古屋大学の高田広章教授は，ITRONは「CPUコアのデバイス・ドライバである」と表現することがあります．これは，CPUコア以外の周辺デバイスはμITRON仕様OSの管理の外である

注1：http://www.ertl.jp/ITRON/SPEC/mitron4-j.html

第2章

TOPPERSプロジェクトの概要と展開

高田 広章

本書では，μITRON4.0仕様に準拠したオープン・ソースのリアルタイムOS（RTOS）であるTOPPERS/JSPカーネルなど，TOPPERSプロジェクトの開発成果を題材に，リアルタイムOSとその周辺技術を解説します．

TOPPERS/JSPカーネルのソース・コードは，TOPPERSプロジェクトのWebサイト（http://www.toppers.jp/）からダウンロードできます．シミュレーション環境はパソコン上で動作するので（**写真1**），本書をきっかけに，ぜひ自分で動かしてみていただきたいと思います．

本章では，TOPPERSプロジェクトのねらいや目的，これまでの成果や今後の計画と，TOPPERSプロジェクトの最初の開発成果であるTOPPERS/JSPカーネルについて紹介します．

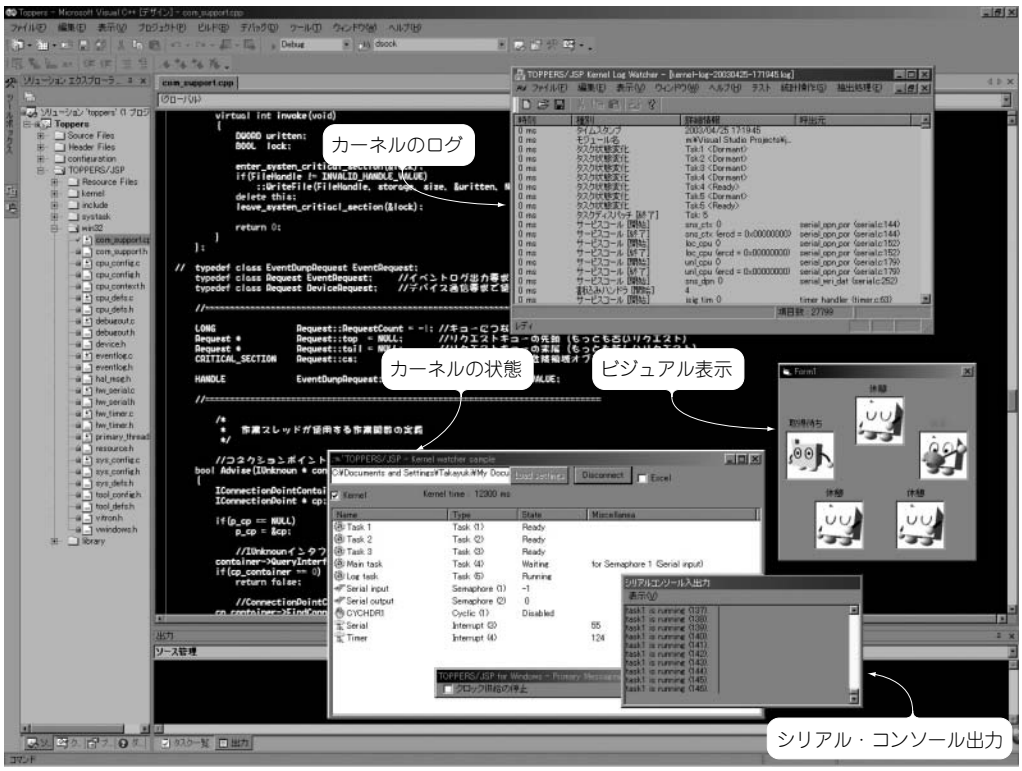


写真1　TOPPERS/JSPのWindows上のシミュレーション環境の動作例
この画面イメージは，JSPカーネルのWindows上のシミュレーション環境の動作例．背景にあるのがVisual C++のウィンドウで，手前には，カーネルの現在状態を表示するウィンドウ，カーネルのログを表示するウィンドウ，シリアルI/Oに送信した文字を表示するためのウィンドウ，プログラムの動作状態をビジュアルに表示するためのウィンドウがある．最後のウィンドウを表示しているプログラムは，Visual Basicで記述してあり，JSPカーネルのシミュレータとはCOMを使って通信している．

第3章

SH-2用 TOPPERS開発ツール

邑中 雅樹

ルネサス テクノロジのSH-2プロセッサ上で，フリーのμITRON実装であるTOPPERS/JSPが動作するようになりました．Webサイトで配布されるTOPPERSのカーネル本体とアプリケーション，開発ツールEclipseなどを利用すれば，だれでもμITRONを使った組み込み開発について学習できるようになります．

そこで本章では，SH-2プロセッサ上でTOPPERSを動作させる方法と，TOPPERSを使った開発について解説します．なお，SH-2プロセッサ（SH7144F）を搭載した開発用ボードとして，ここではInterface誌2006年6月号に付属した基板を使用しました．

1. TOPPERSの開発環境

● 開発ツールに対する要望が高まってきた

TOPPERSプロジェクトは，カーネルの開発だけを目的としたプロジェクトではありません．教育や高信頼性検証といった領域からもTOPPERSカーネルが注目される中，ユーザ・フレンドリなオープン・ソース・ツールを求める声が徐々に高まってきました．

TOPPERSカーネルは可能な限りコンパイラやツールに依存する部分としない部分を分離するように設計されています．主要なカーネル開発者は，GNUのツール群の利用を前提としてカーネルの開発を行っており，TOPPERSカーネルを用いたOSの構築においては，GNUのツール群を用いることがもっとも情報を集めやすい選択肢であると言えます．

● GUI環境，開発検証環境の固定，GCCのビルド

しかし，GNUのコンパイラであるGCCはコマンド・ライン・ツールであり，DOS時代を知らない最近の若いエンジニアに使い方を教えようとすると，GCC以前にコマンド・シェルの使い方から教える必要があります．ここには大きな壁が存在しています．

また，高信頼性検証の領域では，検証環境をいかに固定するかが鍵となります．具体的には同一のソース・コードとツールのバージョンを使い続けるわけですが，その一方で，GCCは毎日進歩を続けている典型的なオープン・ソース・ソフトウェアであり，ツールのバージョンを固定することによる損失は小さくありません．ここに大きなギャップが存在しています．

実製品開発へTOPPERSカーネルを利用する場合にもギャップはあったかもしれません．GCCは，基本的にはソース・コード配布なので，製品開発者が自分の使うCPU用のクロス環境を自分でビルドすることになります．最近のGCCはかなりましになりましたが，かつてはGCC自身のバグのせいでビルドさえ通らないケースもありました．まさにTOPPERSカーネルを用いた開発にたどり着くまでに，本質的でない部分で多くの時間を割かれる格好です．趣味ならいざしらず，業務では許されないでしょう．

インターネット上ではクロス開発用のGCCがダウンロード可能になっているサイトもありますが，複数の環境をインストールしているうちに，いつの間にか動かなくなるということもまれにありました．Windows用として提供されているクロス用GCCの多くが，特定のエミュレーション・ライブラリに依存し，そのライブラリがまれにバージョン非互換であるためです．

カーネルの完成度としては，製品への搭載を検討できるほどのものになったTOPPERSカーネルですが，さらなる普及のためには，ツールを含めた統合開発環境を提供しなければならないという状況にありました．

第4章

Windows上で動作するシミュレータ環境を使って実際に動かしてみよう！

今井 和彦

本書でここまでの章を読んだ方は「へぇ，TOPPERSって，フリー・ソフトウェアなんだ．試しにやってみようかな」と思われたことでしょう．リアルタイムOSに限らず，ソフトウェアに対する理解の早道は，本を読むだけでなく実際に動かしてみることです．幸い，TOPPERS/JSPカーネルにはシミュレーション環境が含まれており，実機がなくてもひととおりのことが試せます．今回は，これを用いて実際にプログラムを書いて，動かしてみましょう．

初心者がリアルタイムOSを学ぶ際のポイントがいくつかあります．まずは個々のサービス・コール[注1]単体ではなく，その背後にある考え方を意識して，それが使われる状況とセットで理解することが大切です（**図1**）．同じサービス・コールでも使う状況によっては意味が異なってくるのです（ちょうど，外国語を学ぶ際のイデオムや文脈に似ている）．それを知らないとμITRONの仕様書や参考書を読んでも，「このサービス・コールはどうやって使うのだろう？」，「どうして私が使いたい機能は用意されていないの？」といった疑問が残ります．これではせっかくリアルタイムOSを導入しても，開発効率がアップするどころではありません．

1. 使用するツールとインストールの手順

最初にTOPPERS/JSPカーネルのシミュレーション環境について述べます．JSPカーネルにはWindows用のシミュレーション環境が用意されています．実機がなくてもデバイス・ドライバを含めてシミュレーションできるように，自分でデバイスをモデリングするしくみが提供されています（**図2**）．

本文中で扱う例題は，`http//www.cqpub.co.jp/interface/techi/techi.html`からダウンロードできるので，ぜひ自分のパソコン上で動かして体験してください．

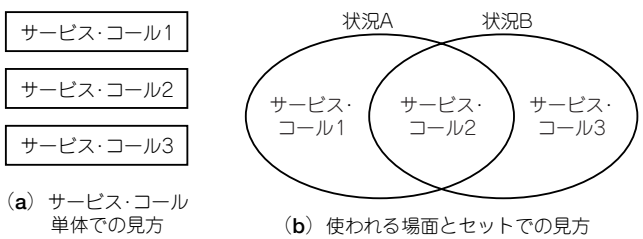
開発は，以下の三つのツールを用いて行います．

- TOPPERS/JSPカーネルWindows用シミュレータ（Windows 9xはサポート外）
- Visual C++ 6.0またはVisual C++ .NET
- Visual Basic 6.0

● インストール

Visual C++とVisual Basicはすでにインストールされているものとします．

●JSPカーネルのコンフィギュレータとデバイス・マ



（**a**）サービス·コール単体での見方

（**b**）使われる場面とセットでの見方

（**a**）サービス・コール単体だけ見てもOS全体は理解しにくい．しかし，（**b**）使われる状況とセットで，さらにその背後にある考え方まで意識すると理解しやすい

図1　リアルタイムOSは難しい？

注1：一般には，OSのAPIはシステム・コールと呼ばれているが，μITRON4.0仕様ではOSのAPIとソフトウェア部品のAPIを合わせてサービス・コールと総称している．本書でもこれに従う．

第5章

XScaleへ移植するための基礎知識

邑中 雅樹

ここから第7章までは，カーネル内部の話題です．TOPPERSカーネル固有の話が多くなるかもしれませんが，どのリアルタイム・カーネルもおおむね似たような構造をもっているので，ほかのリアルタイム・カーネルを扱うときにも参考になるのではないかと思います．

カーネル内部を触わる動機としては，

- サービス・コールを拡張したい
- 新しいボードやCPUに対応したい

などが考えられます．今回は，配布パッケージには含まれていない新しいターゲット・ボードへの移植方法を紹介します．

そのために，まずは移植に必要な道具について解説します．アプリケーション開発とカーネル移植では必要な知識が少し異なり，オープン・ソース固有の事情やコンパイラやデバッガに関する，知られているようで知られていない留意点もいくつかあるため，少しページを割いて解説します．

次に，今回移植する環境の紹介を行い，最低限の環境構築を試みます．

これ以降の章はカーネル本体の移植ということもあり，JSPカーネルに強く依存した話になります．可能な限りソース・コードを引用しますが，お手元のパソコンの画面にJSPカーネルのソース・コードを表示しておくことをお勧めします．

1. JSPカーネルの構成

JSPカーネルは，当初から移植性を重視しているため，C言語で記述されたハードウェアに依存しない部分と，アセンブラが必要だったり周辺デバイスへのアクセスが必要になるターゲット依存部分が，ソース・コードのレベルで明確に分離されています．さらに，ターゲット依存部は，CPU依存部とボード依存部に明確に分かれます．また，Release1.4からは，GCC以外のビルド環境への対応も強化され，一部は分離されました．

これらをまとめると，JSPカーネルが想定するターゲットのモデルは**図1**のようになっていると考えられます．JSPカーネルの移植作業とは，MPUやボード，ツールなどに依存している部分を，使っているビルド・ツールとターゲットに合わせ込む作業といえるでしょう．

では，具体的にどのように分離されているのかをざっと見てみましょう．

図2は，JSPカーネルのディレクトリ構成の一部を抜粋したものです．ターゲット依存部は，`config`ディレクトリにまとめられています．このターゲット依存部は，CPU依存部とボード依存部に分かれています．たとえば，`config/h8/akih8_3048f`というディレクトリがあります．この場合，`config/h8`ディレクトリの中にGCC用のH8に依存する部分が，`akih8_3048f`ディレクトリの中に秋月電子通商製H8/3048ボードに依存する部分が，それぞれ格納されています．

CPU依存部やボード依存部に含まれるコードの分量は，ターゲットによって大きく異なります．SHのように周辺機能も内蔵され，ある程度標準化されてい

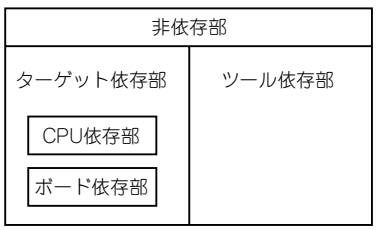


図1　JSPカーネルが想定するターゲットのモデル