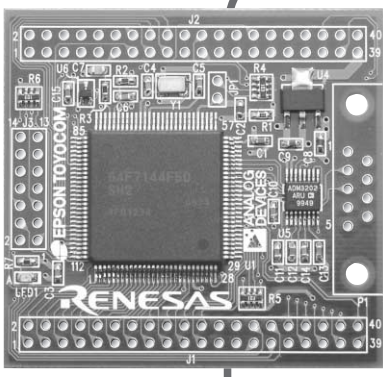
第7章

ロイヤリティ・フリー，ソース・コード提供のRTOS

SH-2付録基板でNucleus PLUSを動作させる

福士 学



本章では，ロイヤリティ・フリー，ソース・コード提供のRTOSであるNucleus PLUSをSH-2付録基板に移植し，サンプル・プログラムを動作させる．Nucleus PLUSはROMが30Kバイト前後，RAMは4Kバイト程度でも動作するほか，μITRONの経験があれば理解しやすいという特徴をもつ．（編集部）

Nucleus PLUSとは

● ロイヤリティ・フリー，ソース・コード提供が特徴

Nucleus PLUSは，EDAベンダであるメンター・グラフィックス製の商用のマルチタスク・リアルタイムOSです．メンター・グラフィックス・ジャパンは本社が米国にあるので，Nucleus PLUSを含む一連の製品は米国製品となっています．

この名前にあまりなじみがないかもしれませんが，Nucleus PLUS自体は日本だけでもすでに10年以上の実績があります．

製品はロイヤリティが不要で，基本的に初期導入費用のみで利用可能です[注1]．

また，ソース・コード提供となっているので，内部処理がどのようになっているのかを手元で確認できます．開発作業の流れとしては，このソース・コードをコンパイラなどのツール・セットを用いてRTOS本体である「RTOSライブラリ」を作成し（RTOSライブラリは最初に一度構築するのみ），ユーザが作成されたアプリケーションとリンクしてビルドを行う，という形になります．

● 対応アーキテクチャとミドルウェア

対応しているCPUアーキテクチャは多岐に渡り，一般的に用いられているARM，MIPS，PowerPC，SHなどのほかにも，DSP系やFPGA系のものも用意しています．

関連製品として周辺ミドルウェア（TCP/IPプロトコル・スタックやFATファイル・システムなど）を多く用意しており，さらにクロス開発環境（デバッガやプロファイラなど），シミュレータ，コンパイラやデバッグ・インターフェースなど，セットに依存しますが，ひととおりをそろえています．

● RTOSとしての特徴

RTOSの特徴として，組み込みに特化してコンパクトかつ高速性や移植性を考慮して設計されています．そして，ライブラリ形式で用いるためスケーラビリティがあり，プライオリティ・プリエンプション方式とタイム・スライス方式を両立することができます．

カーネル自体はスレッド型で，タスクやセマフォなどのオブジェクトは生成数に制限がありませんし，生成数が増加してもそれぞれをコントロールする時間（たとえばタスクを切り替える時間）は左右されず一定となります．また，これらのオブジェクトは動的に生成や削除が可能です．

また，APIについては，名称から理解しやすいものを採用しています．たとえば，タスクを生成するためのAPIは`NU_Create_Task`，キューにデータを送信するAPIは`NU_Send_To_Queue`などとなっているのです．機能から考えるとμITRONの知識があれば，理解しやすいRTOSと言えるでしょう．

RTOSとして必要な資源は，まずメモリ（ROMやRAM）です．利用していない機能はリンク時に入らないため，フット・プリントは，平均してROMが30Kバイト前後，RAMは4Kバイト程度となっています（これらはCPUアーキテクチャやツールに依存している）．もう一つの資源として，タイム・スライスやスリープなどの時間関連のAPIを用いるためには，インターバル・タイマを一つ用意する必要があります．

これらの特徴を**表1**にまとめました．

SH-2付録基板への移植の実際

● 既存の評価ボード向けセットに手を入れるのが簡単

リリースされているNucleus PLUSは，CPUアーキテクチャおよび利用したツールに対応したセットとして入手可能です．これらは基本的に評価ボード（Evaluation Board）に対してポーティングされているので，該当する評価ボードをもっていれば，すぐに実行させることができます．

注1：詳細はメンター・グラフィックス・ジャパン（株）に問い合わせのこと．

表1　Nucleus PLUS の特徴

名　称	Nucleus PLUS
フット・プリント	ROM 30K バイト前後 　今回の SH7144F では最大約 40K バイト， 　最小約 15K バイト RAM 4K バイト前後 　今回の SH7144F では 3K バイト強
必要資源	RTOS を格納および実行するためのメモリ インターバル・タイマ一つ（必要に応じて）
保持機能	タスク管理（サスペンド，レジューム，…） タスク間同期（セマフォ，イベント，キュー，…） メモリ管理（固定長，可変長） 時間管理（ワンショット，定周期） 割り込み管理（ローレベル，ハイレベル）

今回の SH7144F は「SH2（7043）EVAL board」という既存の評価ボード向けのセットがあるので，これを母体として移植を進めます．

● 移植作業内容の検討

単純に書くと，Nucleus PLUS のソース・コードのうち，アセンブラで記述されている部分が移植を検討する部分となります．このアセンブラ部分は，大きく分けて三つあります．

① 初期化部分（ブート・コード，割り込み関連を含む）
② スレッド制御部分
③ タイマ制御部分

このうち②と③については，同じアーキテクチャであればほとんど変更不要なので，検討から除外します．また，①における割り込み関連部分は，それぞれの SH-2 の型番によっては若干ベクタ・テーブルが変わる程度なので，工数はほとんどかからないでしょう．つまり，今回はブート・コードが移植検討の主要部分となります．

このブートの部分を検討する前に，まずは Nucleus PLUS のブート遷移を**図1**として整理してみます．

初期化処理には**図1**中の（**a**），（**b**），（**c**）という 3 段階があります．最後の（**c**）アプリケーション初期化処理については，のちほど出てくるのでここでは割愛します．（**b**）高レベル初期化処理は，RTOS 側が自分を初期化するためのルーチンなので，通常は触れることがありません．また，この二つについては ANSI C での記述となります．

結局アセンブラで記述されている（**a**）低レベル初期化処理においてブート・コードの移植を行い，（**b**）に遷移する前にはアプリケーションを含む通常のプログラムが動作するような状態にしておけば良いのです．

図1の（**a**）で行っていることを**図2**に列挙します．

低レベル初期化処理としてはこれだけです．必要があればメモリ・チェックや全領域初期化，他メモリ空間のための設定（いわゆるバス・コントローラ設定），ほかに必要となるペリフェラルの初期化設定などを追加することになりますが，今回用いるターゲットでは，ほとんど触れなくても良いようです．

実際に元のセットから変更を行った部分を**図3**に示します．

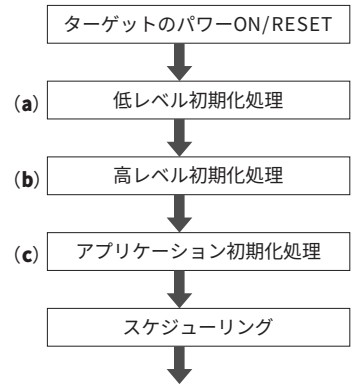


図1
ブートからの動作遷移

- 必要なレジスタの初期化（スタック・ポインタ含む）
- BSS 領域のクリア
- DATA 領域を ROM から RAM へコピー
- ベクタ設定（VBR）
- RTOS 内部変数の初期化
- RTOS システム・スタック設定
- インターバル・タイマ設定

図2　低レベル初期化処理の中身

- システム・スタックの調整
- VBR アドレス変更
- インターバル・タイマ・カウントの調整（クロック依存）
- MSTCR2 などの特殊レジスタ設定
- 管理ベクタの追加
- キャッシュ設定を削除

図3　移植で行った内容

この数点で移植が完了してしまいます．もし SH7144 および SH7043 のいずれにも知識があれば，ほんの 1 時間もしないうちに終わってしまうかもしれません．

● インターバル・タイマについて

移植の際にはカウントの調整のみを行っていますが，今回は CMT（Compare Match Timer）の 0 番を利用しています．この CMT#0 は RTOS で予約扱いとなるので，アプリケーション側から利用してはなりません．また，これに付随して MSTCR2 の CMT も設定しています．

アプリケーションを動作させる

● アプリケーション demo.c

第 6 章で解説したとおり，Nucleus PLUS の移植は最低限ではこの程度で完了してしまいます．

本来 RTOS を移植した後には，当然そのための試験を行わなくてはなりませんが，今回は割愛します．

さて移植が正しく終わると，もちろんユーザ・アプリケーションを動作させることができるようになります．この確認のため，およびアプリケーションの作成方法を学ぶために，標準デモンストレーション・アプリケーションである demo.c が添