マイコンを正しく操縦するための作法

基礎から学ぶC言語講座

岡田 好一
Yoshikazu Okada

第6回　メモリ割り当てとスタック管理を攻略しよう

本稿で使用するR8C/1BのROMは16Kバイト＋2Kバイト，RAMは1Kバイトです．

マイコンでは記憶域が限られているため，意識してROMとRAMの使い分けが必要です．素直にコンパイルするとC言語のプログラムはROM上に配置されるので，データをどうするかが問題になります．まず，主記憶の構成から復習しましょう．

本連載で使用中のRenesas Starter Kit for R8C/1Bの入手方法は稿末をご覧ください〈編集部〉．

主記憶の構成

R8C/1B(16KバイトROMタイプ)のメモリ配置図を**図6－1**に示します．

● 周辺機能を制御するためのレジスタSFR

アドレス空間の最初の768バイト(00000h～002FFh)はSFR(Special Function Registers)と呼ばれ，入出力ポートやタイマなどの周辺機能を制御するためのレジスタが配置されています．いわゆるメモリ・マップト入出力です．

割り込み系を除くと，ここはプログラムにとっては外界との唯一の窓口です．ポート(港)の言葉が言い得て妙です．

● ROM上のプログラム

R8Cのようなマイコンでは，プログラムはROMに書き込むのが普通で，その前提でICもコンパイラも設計されています．

C言語には，「ソース・プログラムから翻訳された機械語プログラムは，実行中は不変である」という前提があります．そのため，ROMとの相性は良いのです．

本連載で使用しているR8C/1Bの16Kバイト版では，0C000h～0FFFFhまでの16Kバイトにプログラム用のROMが用意されています．16Kバイトの容量は，多数の固定データを詰め込もうとしない限り，1システムとしては使いでがあります．

もちろん，RAMにプログラムを置くのは反則ではないし，動くはずです．クロック数が厳しい場合など，プログラムが自分で機械語プログラムを書いて制御を渡すという(自己書き換えプログラム)，アクロバット的なプログラムが思い浮かぶかもしれません．が，あくまで例外でしょう．

RAMとその使われ方

R8C/1Bの16KバイトROM版では，0400h～07FFhに1KバイトのRAMが用意されています．R8Cではの速度とROM/RAMの速度は均衡し

Keyword 1　**仮想記憶(virtual memory / virtual storage)**

複数のプログラムが並行して走るマルチタスクの場合，互いの干渉を避けるためにはメモリを分離する必要があります．仮想記憶は，一般のソフトウェア，つまりプログラムから見える記憶領域で，別のプログラムで同じ番地を使用しても，別の記憶として扱われるしかけです．

OSから見ると，個々のプログラムのアドレスと物理メモリと退避のためのディスク領域を結びつけるしかけです．プログラムのアドレスと物理メモリの対応はハードウェア化されていて，MMU(Memory Management Unit)と呼ばれるハードウェア機構が管理します．プログラムが実働するには実メモリが必要ですから，プログラムが仮想記憶にアクセスすると実メモリが割り当てられているかが判定され，退避されていれば実メモリに戻されますし，退避されていなければ実領域が生成されます．この定義では，仮想記憶と実記憶の大きさは限定されていないので，仮想記憶の方が狭い場合もありえます．

マイコンでも処理が複雑かつ高度化されるにつれ，仮想記憶が必要とされるでしょう．

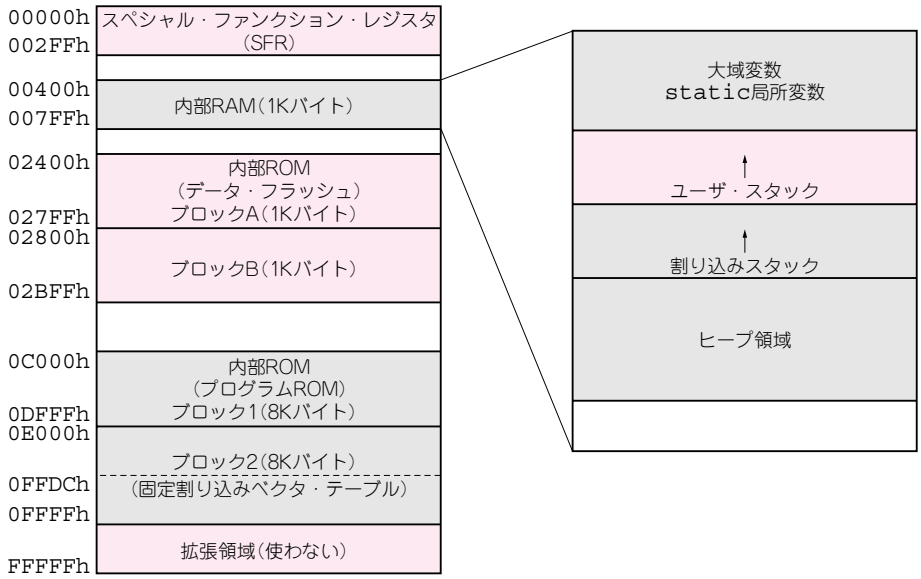

図6-1　R8C/1B(16Kバイト ROM タイプ)のメモリ配置図
R8C は上位機種の M16C の CPU を引き継いでいるので，データは 16 ビット・アドレス内(64Kバイト)，プログラムは 20 ビット・アドレス内(1Mバイト)に配置されている

ていますから，速度に関しては使用上の注意はありません．

● システム・スタック

R8C 自体の動作に最小限必要な RAM 領域は，割り込みスタック領域です．簡単なシステムですべてアセンブラで組むならば，最初に割り込みスタック・ポインタ(ISP)の値を設定し，そのまま割り込みスタック領域を使い続けるのがシンプルです．

システム・スタックの第1の用途は，サブルーチンや割り込み処理ルーチンの終了時に，復帰すべき命令へのアドレスの一時記憶です．復帰するときには，そのアドレスがプログラム・カウンタに格納されます．特に設定しない限り，通常のサブルーチンでも割り込みルーチンでも ISP が使われ，ユーザ・スタック・ポインタ(USP)は使われません．

第2の用途は，CPU 内のレジスタで作業記憶が足りない場合のデータの一時退避用です．この場合，機械語の push 命令と pop 命令が使われます．

● ユーザ・スタック

本稿で使用している純正の C 言語による開発システムでは，一般ユーザのスタック領域が別に設定されます．通常のプログラムはユーザ・スタック領域，割り込みルーチンは割り込みスタック領域と使い分けます．

ユーザ・スタックを利用するには，フラグ・レジスタ(FLG)の U フラグをセットします．セットするコードが，自動生成されるスタートアップ・プログラムに入っています．

スタックの使い分けはハードウェアが行います．割り込み処理ルーチンは，自動的に割り込みスタック領域を使うことになります．割り込み処理が終了すると，

Keyword 2　自己書き換えプログラム

通常の計算機言語では，プログラムとデータは分離しているので，実行中にプログラムが変わりうるのは，やむを得ぬ離れ業として以外は，にわかには信じ難い話に思えます．

しかし，歴史的にはごく初期からデータとプログラムの境界をなくす方向の動きがあり，計算機言語にも対応するしかけがあります．

例えば，LISP ではインタプリタ自身を eval 関数で呼び出すことができます．それどころか，PROLOG では自分自身が作ったプログラムを自分に追加することができます(もっとも，OS レベルではアップデートなどと称して，日常茶飯事のできごとだが)．

マイコンでプログラムの固定が可能なのは，アドレス・レジスタなどの間接指定のしくみがあるからです．経験上，配列で済む場合はその範囲で，もう少し柔軟に対応したい場合はポインタを利用します．

プログラムでインタプリタやシミュレータを書くのはもちろん可能ですから，結局は程度問題になります．