≫ 文法の曖昧さを理解して確実性と再利用性を高める

マイコンC言語 転ばぬ先のつえ

第30回　動的スタック領域の削減②…引数領域を削減する

鹿取 祐二

表1　各マイコンにおける引数格納規則の一例
一部を記載．詳細を知りたい場合は各処理系のマニュアルを参照のこと

項　目	RL78	H8	RX	SH
引数格納に利用する汎用レジスタ	AX，BC，DE	ER0，ER1	R1，R2，R3，R4	R4，R5，R6，R7
引数格納には利用しない汎用レジスタ	HL	ER2～ER6	R5～R15	R0～R3，R8～R15
汎用レジスタのサイズ	2バイト	4バイト	4バイト	4バイト
格納可能な個数	2～6個	2～6個	4個	4個

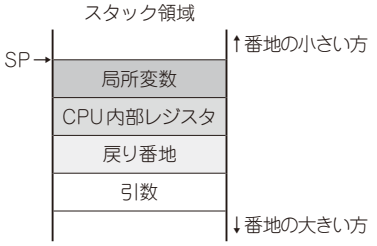

図1　スタック領域に保存されるデータの内容
関数が呼び出されたときの保存内容

本連載では，C言語の言語仕様（文法）の曖昧な部分と，それにより起こる問題を解説します．再利用性と効率が高く，安全かつ安心して使えるソフトウェアが開発できるようになることを目指します．

前回（2024年2月号）から，動的領域であるスタック領域の削減について紹介しています．前回はネスト・レベルを減らすことでスタック領域を削減する方法を紹介しました．

今回は，スタック領域に格納されるデータの1つである，引数の領域とその削減方法について解説します．（編集部）

引数は汎用レジスタに格納できる個数に抑えよう

● スタック領域に格納される情報を考える

ネスト・レベル以外でスタック使用量を削減する方法は，スタック領域に格納されるデータを個別に考えるしかありません．以前に紹介した通り，スタック領域には**図1**に示すデータが格納されます．この中で削減対象から外れるものは戻り番地です．関数からの戻り番地だけは固定のサイズであり，現状市販されているマイコンだと2バイトか4バイトとなります．どのように関数を記述しても変化しないサイズですから，削減対象からは除外します．

結果，考えるべきは引数，CPU内部レジスタ，局所変数として使用される3つの領域となります．本節では，その中で引数のための領域について削減方法を紹介します．

● 汎用レジスタに割り付くかどうかが問題

近年の処理系は，関数呼び出し時の引数は可能な限り汎用レジスタに格納して目的の関数に渡します．もちろん，個数や格納可能な引数の型は処理系や対象のマイコンによって変化します．本連載で取り上げている4つのマイコン，処理系もルネサス エレクトロニクス社純正のものであれば，大まかには**表1**のようになっています．なお，「大まかに」と記載したのには意味があり，引数の格納規則はかなり複雑だからです．**表1**は大まかな記載であるため，詳細は各処理系のマニュアルを参照ください．

幾つか具体例を見てみましょう．例えばRL78の場合，**図2**のように**表1**の規則を満たしていれば，引数がスタック領域に格納されることはありません．しかし，**図3**のように**表1**の規則から外れると，外れた引数はスタック領域に格納され，その結果，スタック領域を消費することになります．

● マイコンや処理系ごとの違いはどうするのか？

表1から分かるように引数の格納規則はマイコンや処理系により異なります．先の**図3**(a)のプログラムもRXが対象の場合は**図4**のように変化し，全ての引数が汎用レジスタに格納され，スタック領域は使用しません．

つまり，「引数は何個までなら大丈夫」とか「型はど

第15回　演算子⑥…関係演算子と等値演算子（2022年5月号）
第16回　演算子⑦…論理演算子（2022年6月号）
第17回　演算子⑧…条件演算子とカンマ演算子（2022年10月号）