第5章　CPUメーカ純正環境ならできる！アドレス情報の節約

# コンパイラとリンカが連携！グローバル変数アクセスの最適化

鹿取 祐二

本稿では，コンパイラやリンカがそれぞれ独立なGCCなどの汎用コンパイル環境ではなかなか実現できない，CPUメーカであるルネサス純正コンパイラに備えられた，外部（グローバル）変数のアドレス情報最適化のしくみを紹介します．（編集部）

## 外部変数アクセス最適化…構造体化して相対アドレスにする

ここでは，ルネサス製C/C++コンパイラ・パッケージに備えられた外部変数（グローバル変数）のアドレス情報に対する最適化オプションを紹介します．ターゲット・マイコンはRXファミリです．

### ● 外部変数アクセスの問題点

RXファミリに限らず，マイクロコンピュータは外部変数（グローバル変数）の操作が苦手です．その理由はC言語では，外部変数は必ずメモリに割り付くからです．

目的の変数にアクセスするためには，アドレス情報が命令中のどこかに必要となります．

RXファミリは4Gバイトのメモリ空間をサポートしていますから，アドレス情報は32ビットです．結果，外部変数をアクセスするたびに32ビットのアドレス情報を汎用レジスタにロードしなければなりません．

**リスト1**（b）のコンパイル結果で「r」の部分が外部変数のアドレス情報です．外部変数アクセスが6回あり，32ビットをフルで使っており，効率が悪いです．

### ■ プログラム・サイズ削減策その1…プログラマによる構造体化

プログラム・サイズを削減するためには，**図1**に示すように，プログラマ自らが外部変数を構造体化する必要があります．構造体になっていれば，目的の構造体の先頭アドレスだけを汎用レジスタにロードすればよくなります．先頭以外のメンバは先頭アドレスからのオフセットで操作できるようになり，アドレスのロード回数を削減できます．結果としてスピードも向上します．

**リスト1　外部変数はアドレスがフルの32ビット必要になるので命令長が長い**

```
 1: short a, b;
 2: int c;
 3: extern short x, y;
 4: extern int z;
 5:
 6: void main(void)
 7: {
 8:     c = a + b;
 9:     z = x + y;
10: }
```

（a）ソース・プログラム

```
 1: FB42rrrrrrrr _main: MOV.L #_a,R4
 2: FB32rrrrrrrr   MOV.L #_b,R3
 3: DC45           MOV.W [R4],R5
 4: DC34           MOV.W [R3],R4
 5: FB32rrrrrrrr   MOV.L #_y,R3
 6: 4B54           ADD   R5,R4
 7: FB52rrrrrrrr   MOV.L #_c,R5
 8: E354           MOV.L R4,[R5]
 9: FB42rrrrrrrr   MOV.L #_x,R4
10: DC45           MOV.W [R4],R5
11: DC34           MOV.W [R3],R4
12: 4B54           ADD   R5,R4
13: FB52rrrrrrrr   MOV.L #_z,R5
14: E354           MOV.L R4,[R5]
15: 02             RTS
```

外部変数はアドレスが長い

（b）コンパイル結果

### ■ プログラム・サイズ削減策その2…コンパイラ・オプションを使う

外部変数の構造体化をプログラマに代わって行うのがルネサス製コンパイラの「外部変数アクセス最適化を行う」オプションです．このオプションを使えば外部変数の宣言を変更する必要はなく，コンパイラが自動的に外部変数を構造体化し，アドレス・ロードの回数を削減します．

目的のオプションは**図2**に示す「外部変数アクセス最適化を行う」のドロップダウン・メニューで指定します．なお，設定には「モジュール内で最適化」と「モジュール間で最適化」の二つがあり，「モジュール間」の場合は**図3**に示すダイアログが表示されますが，構わずOKしてください．このダイアログの意味は各設定の効果と共に紹介します．

### ● オプションその1…「モジュール内で最適化」の効果

「外部変数アクセス最適化で行う」オプションを「モジュー