≫ 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13回　演算子④…算術演算子の正しい使い方

鹿取 祐二

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．

今回は，C言語において算術演算子の交換法則が成り立たないこと，C言語とマイコンとで乗除算の仕様が異なる場合に発生する問題とその対策を解説します．〈編集部〉

20 整数型だと数学の交換法則は成り立たない

乗算の「*」，除算の「/」，剰余の「%」などの算術演算子は注意しなければならない項目が多いのが特徴です．その代表例は整数型の場合，数学における交換法則が成り立たないことです．

● その①…乗算と除算の場合

数学では乗算と除算は順不同となっていますが，C言語では整数型の除算は割ったときの商であり，余りは切り捨てです．そのことを常に考慮しなければなりません．

かく言う筆者も，そのことを忘れた失敗をたびたび犯した記憶があります．例えば，以下は10ビット精度のA-D変換結果を0～100%に補正する式です．

```
signed char volume;
volume = ADDR / 1023 * 100;
```

ADDR（unsigned short型）がA-D変換結果を格納したレジスタだと思ってください．10ビット精度だと変換結果は0～1023の間なので，結果を1023で除算して重みを求め，それを100倍すれば完成です．数学上は問題のない計算式ですが，C言語ではNGです．「/」は除算の商を導くので，変換結果が0～1022までは全て0，最大値の1023のときだけ1となります．それを100倍したら，最終的な結果は0か100にしかなりません．正しくは乗算を先に行う次の記述が正解です．

```
signed char volume;
volume = ADDR * 100L / 1023;
```

乗算を先に行うと符号付き16ビットの限界値である32,767を超える可能性があるため，定数値の100に接尾子のLを付けました．第1回（本誌2021年1月号）で紹介した「符号付きと符号なしを一緒に演算しない」の教えには従っていませんが，A-D変換結果が0～1023の間で推移する限り，移植性は担保されるので問題はありません．

● その②…加算と減算の場合

以上のように整数型の場合，乗算と除算には交換の法則が成り立ちません．このことは加算と減算に関しても，ある意味同じことが言えます．ただし，加算と減算の場合，式の途中でけたあふれが生じても，最終的な結果が，代入先の変数に格納可能な値であれば事実上は問題ありません．例えば，大ざっぱな例ですが次のような計算式です．

```
int result;
result = 20000 + 20000 - 10000;
```

もし，int型が16ビットの場合，20,000と20,000を加算した時点でオーバフローを起こしますが，次に10,000を減算しているので結果は30,000となります．30,000ならばint型にもちゃんと格納できるので正しい結果が求められます．ただし，上記は符号付き整数型の内部表現を2の補数に限定した場合の記述であって，文法的にはNGの記述です．本来ならば減算を先に行うか，接尾子を使った次のような記述が文法的にも許された記述となります．

```
int result;
result = 20000  - 10000 + 20000;
result = 20000L + 20000 - 10000;
```

最終的にC言語では「数学の交換法則は成り立たない」と常に考えた方が良いでしょう．

21 乗除算の仕様はC言語とマイコンとでは異なる

先ほどまでの内容は特に組み込み系に特化したものではありませんが，今度は違います．特に16ビット・クラスのマイコンを使う方は，これから紹介する内容