Interface増刊 “ゼロ”から学び，プログラミングを楽しもう

見本

インターフェースZERO

No. 01

http://zero.cqpub.co.jp/

プログラミングからデバッグまでを一通りマスタしよう

やさしい組み込みCプログラミング

CPUの仕組みとプログラムの役割を理解する

プログラムはなぜ動く

プログラム開発に必要な基礎知識を理解する

プログラミング言語とは何か

ハードウェアを制御できるC言語の特徴とその使い方

なぜC言語を使用するのか

CQ出版社

第1章

CPUの仕組みと
プログラムの役割を理解する

プログラムはなぜ動く

大中 邦彦

1.1 コンピュータという道具はスゴイ！

人類が石を道具として使ったのは今から200万年も前のことだそうです．その後，人類は道具から道具を作り出すことによって次々と新しい道具を発明し，3000年前には車輪を発明しました．車輪は水車などに発展し，勝手に動き続ける機械も現れました．

近年になると人類はコンピュータを発明します．初期のコンピュータは，入力した数値に対して決められた計算を行い，その結果を自動的に出すというものでした．電卓のような「入力した二つの値を足した値を出力する」という単純な計算ではなく，「与えられた10個の数値の中から最大値と最小値を選び，その平均を出し…」というように，一連の手続きを必要とする計算をすることができたのです．

コンピュータの「一連の手続きをこなせる」という機能は，人類が過去に発明した，ほかの道具や機械とは一線を画す，非常に大きな特徴です．コンピュータが登場する以前にも，「からくり人形」のように一連の動作を順序通りにこなす機械が作られていました．しかしそれらは，機械的にあらかじめ組み込まれた仕掛けによって動作しているため，機械を作った時に考えられていなかった動きをさせることはできませんでした．

それに対しコンピュータは，プログラムを入れ替えるだけで全く違った計算ができます．プログラムを工夫すれば，コンピュータを設計した人が考えていなかったような全く新しい機能を持たせることができるのです．

「使う人がその道具の機能を後から変えられる」という特徴を持つコンピュータは，人類史上初めてのものといえるでしょう．

1.2 「プログラム」はすごろくだ

子どものころに「すごろく」というゲームで一度は遊んだことがあるでしょう．すごろくは，スタート地点に自分の駒を置き，サイコロを振って出た目の数だけ駒を進めていきます．そして，ゴールまで早くたどりついた人が勝ちになるゲームです．

ここで，**図1.1**のようなすごろくを考えてみましょう．このすごろくで，サイコロの目が全て

第2章

プログラム開発に必要な
基礎知識を身に付ける

プログラミング言語とは何か

増田 晃章，吉田 悦康，奈良 久美子

2.1 コンピュータと会話するには

● 自然言語と機械語

日本語や英語などは，人間同士が意思の伝達に使用するための言語で，これらは「自然言語」と呼ばれます．自然言語を使って直接コンピュータへ指令を出す研究も行われていますが，現在のところは自然言語でコンピュータを自由に制御することはできません．そのため，コンピュータに思いどおりの仕事をさせるためには，プログラミング言語が使用されます（**図 2.1**）．

コンピュータが直接的に理解できるのは，与えられる命令が‘0’であるか‘1’であるかだけです．例えば，「‘1’ならスイッチを入れる」，「‘0’ならスイッチを切る」という動作が可能です．これだけなら二つの動作しかできませんが，2桁で‘00’，‘01’，‘10’，‘11’とすれば四つの動作ができるようになります．そして，‘0’と‘1’の数を3桁，4桁，…，というように増やしていけば，コンピュータはさまざまな命令を実行できるよう

（a）プログラミング言語　　（b）自然言語

図 2.1　言語による違い
人とコンピュータはしゃべる言語が違うので，直接会話はできない．

第3章

ハードウェアを制御できる
C言語の特徴とその使い方

なぜC言語を使用するのか

大中 邦彦

3.1 マイコン上で動作するプログラム

リスト3.1に示すプログラムは，第1章の**リスト1.1**に紹介した電卓のプログラムをARMというCPUの命令を使って書き直したものです．ARMプロセッサにはI/Oポートを入出力する命令がないので，メモリマップドI/O方式を使って周辺装置とデータを受け渡します．

リスト3.1では，「電卓のボタン」が「メモリの0x40000000番地」に割り当てられており，「7セグメントLED」が「メモリの0x40000004番地」にあるものとしてプログラムを作成してあります．

リスト3.1の右側にある数値の一覧は，このCPUが直接理解することができる機械語です（16進法で表記）．このような数値を見ても人間には分からないので，アセンブリ言語を使ってもう少し分かりやすく記述したのが真ん中の列です．

● アセンブリ言語を読んでみよう

このアセンブリ言語で書かれた機械語を，少し詳しく見てみましょう．まず，日本語の文章で書かれた「レジスタA」，「レジスタB」，「レジスタC」は，それぞれアセンブリ言語の中のR1，R2，R3に相当します．最初の2行にあるR4とR5もレジスタなのですが，これらは電卓のボタンと7セグメントLEDにアクセスするための番地を覚えておくために使用しています．例えば，

```
    LDR    R4, Button_Addr
```

という命令は，プログラムの最後の方に書かれている，

```
    Button_Addr: DCD 0x40000000
```

という値をR4というレジスタに読み込む（ロードする）という命令です．これで，R4に0x40000000が入ります．同様に，R5には0x40000004が入ることになります．

続くMOV命令は，レジスタに定数を覚えておくための命令です．すなわち，

```
    MOV    R1, #0
```

は，R1に0を入れる命令になります．LDR命令もレジスタに値を入れる命令でしたが，LDRはプログラムの別の場所に書かれた値を読み込む命令なのに対し，MOV命令は#0という値を直接代入する命令です．

次のSTR命令は，レジスタの内容をメモリに書く命令です．例えば，

```
    STR R3, [R5]
```

という命令は，「R3の内容をR5が覚えている番地に書く」という命令です．R5にはI/Oの1番

第4章

制御文やデータの型，演算子などの
基本文法を理解する

C 言語の文法―基礎の基礎―

三好 健文

本章では，C 言語で自由にプログラミングをするために絶対に知っておかなければならない C 言語のデータ型，演算子，制御構文について説明します．これらは，C 言語を理解する上でどうしても知っておかねばならない基礎の基礎です．

C 言語の文法を網羅して解説することは紙面の都合でできませんので，詳細については他の書籍を参考にしていただきたいのですが，きっと，この基礎の基礎を知っておくことで理解がすすむと思います．

4.1 C 言語におけるデータの取り扱い

● 数字と文字の表し方

定数，すなわち C 言語のプログラムの中で使用することができる値には，「数字」と「文字」があります．

数字は，10 進数，8 進数，16 進数で書くことができます．10 進数は，10 になると 1 桁が繰り上がるおなじみの数です．8 進数と 16 進数は，それぞれ 8 と 16 になると 1 桁が繰り上がる数です．この桁が上がり下がりする数のことを「基数」といいます．

表 4.1 に，「65」という数を 10 進数，8 進数，16 進数で記述した例を示します．8 進数であることを示す場合は頭に「0」を付け，16 進数の場合は頭に「0x」を付けます．16 進数では 16 で桁が上がるため，各桁は 1 ～ 15 までの数を表現する必要があります．そのため，0 ～ 9 までは 10 進数と同じ数を用い，10 ～ 15 までを表現するためにアルファベットの A，B，C，D，E，F を用います．

C プログラムの中で数を表現する場合，表記は大文字でも小文字でもかまいません．すなわち，16 進数の 0xF と 0xf は，どちらも 10 進数の 15 を意味します．

表 4.1　定数表現の例（65 の場合）

	記法	記述例	ビット列（8 ビットの場合）
数字	10 進数	65	01000001
	8 進数	0101	01000001
	16 進数	0x41	01000001
文字		‘A’	01000001

第5章

開発環境を構築し，
実際にプログラムを実行しよう

はじめての C 言語プログラミング

佐藤 亨

5.1 C 言語の開発環境を作ってみよう

前章までに学んだC言語の知識のおさらいも兼ねて，手(とプログラム)を動かしてみましょう．

C言語の開発環境は，多くの場合OSに用意されているので，例えばLinuxやMac OS Xを使える環境が手元にあれば，すぐに試してみることができます．ここでは，Windows7にC言語の開発環境をセットアップする方法を紹介します．Windows XPまたはVistaの場合は，フォルダ構成が若干異なるので，読み替えてください．

今回インストールするのは，Microsoft社が提供している無償の開発環境「Visual Studio 2010 Express」の中の「Visual C++ 2010 Express」(以下，VC++)です．Visual C++ 2010 Express以外にも「Visual Basic 2010 Express」，「Visual C# 2010 Express」などがあります．

これらのツールは，有償のVisual Studio 2010に比べると機能が限定されていますが，プログラミングを体験してみるには十分すぎるほどの機能を持っています．将来，Express Editionでは物足りなくなったときは，有償版を購入してください．

ところで，VC++は「C++言語」用の開発環境です．でも大丈夫．C++言語はC言語の(ほぼ)上位互換になっているのです．たいていのC++コンパイラはC言語にも対応します．VC++は，ソース・ファイルの拡張子を見て，それに適したコンパイルをしてくれます．

● Visual C++ 2010 Expressのインストール

では，早速VC++をインストールしてみましょう．まず，以下のURLに行きます．

```
http://www.microsoft.com/japan/msdn/vstudio/express/
```

そこで，Visual C++ 2010 Expressの「Webインストール(ダウンロード)」をクリックし，セットアップ・プログラム`vc_web.exe`をダウンロードします．このセットアップ・プログラムを実行させて指示通りにクリックしていけば，インストールは完了です．

VC++本体で150Mバイト，オプションを含むとさらに数百Mバイトのダウンロード量になりますので，それに耐えうるネット環境でセットアップを行ってください．ダウンロード後のインストールも時間がかかります．

もし，開発環境とインターネット環境が別々の場合は，直接インストールするのではなく，いったんDVDにしてからインストールする方法もあ

第6章

関数呼び出しに伴うプロセッサの
内部動作を理解する

C言語における関数定義と呼び出し方

三好 健文

6.1 関数の基本的な記述方法

関数は，C言語における処理の単位です．**図6.1**に示すように，関数に「入力」を与えると「出力」を返します．C言語では，入力する値を「引数」，出力する値を「戻り値」と呼びます．

関数は，次のように記述します．

```
戻り値の型　関数名(仮引数)
{
 処理の本体
    ...
 return 戻り値 ;
}
```

ここで，「戻り値の型」には，int や char などのプリミティブ型のほかに，構造体やポインタ型を用いることができます．ただしC言語では，関数の戻り値は一つだけに限られています．

「関数名」は，変数名と同じようにアルファベット，数字，「_（アンダースコア）」を使用して名前を付けます．ただし，最初の1文字を数字にすることはできません．

「仮引数」は，必要な変数の型と名前を定義します．複数の引数を受け取る場合は，仮引数を「,（カンマ）」で区切って列挙します．

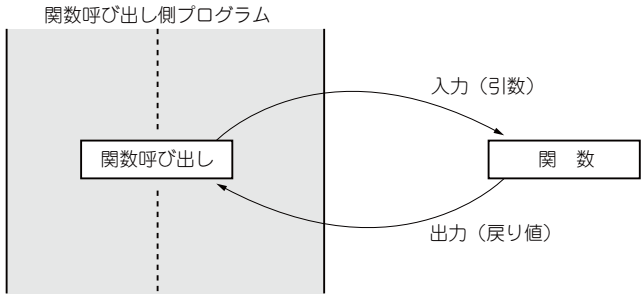

図6.1　関数呼び出しの「入力」と「出力」の関係

第 7 章

プログラム実行に必要な
プロセッサ内部の機能

スタックと割り込みを理解する

大山 将城

本章では，C 言語で開発したプログラムがシステム上で動く際に必要な，「スタック」と「割り込み」と呼ばれる仕組みについて説明します．スタックと割り込みは，C プログラムの文法上では見えないながら，プログラムを実行する上で重要な役割を果たしています．

「スタック」は，第 6 章でも説明しましたが関数の作業領域としてメモリの一部を利用する仕組みです．「割り込み」は，実行の流れを切り替えるための仕組みです．どちらも C 言語の文法ではありませんが，プログラムが動く仕組みを知っておくと，特にデバッグで役に立ちます．

7.1 プログラムが実行される流れを知ろう

ここでは，順次実行，分岐，関数呼び出しという 3 種類の C 言語プログラムを例に，プログラムが実行される流れを見ていきます．まず，順次実行と分岐に関して，C 言語のソース・レベルの動作とメモリ上の動作を見ていきましょう．

● 順次実行と分岐の実行順序

リスト 7.1 に，main 関数のみで作成した順次実行されるプログラムを示します．内容は，処理 A と処理 B です．このプログラムを実行したときの順序は処理 A →処理 B になります．

次に，**リスト 7.2** に if 文により分岐するプログラムを示します．ここでは，引数 ch の値によってローカル変数 result に代入する値を変え，最後は result を戻り値として関数を終了します．このプログラムを実行したとき，分岐が行われて実行される順序は以下のようになります．

リスト 7.1
main 関数のみのプログラム

```
void    main()
{
    /* 処理A */
    /* 処理B */
}
```

リスト 7.2
if 文を含むプログラム

```
int FuncA( char ch )
{
    int result;

    if(ch == 'A'){
        result = 1;
    }
    else{
        result = 2;
    }

    return result;
}
```

第8章

ポインタを制する者は
C 言語を制す

便利で効率の良いポインタと仲良くなる

増田 晃章，吉田 悦康，奈良 久美子

筆者がC言語に初めて触れたとき，先輩から「C言語はポインタが難しくて山場だぞ．でもポインタを制すればC言語を制したも同然だ！」と脅されました．確かに，今でもこの先輩と同じような意見を聞いたり，「ポインタ」というと嫌な顔をする人に出会う機会が多いことは事実です．それでは，この難しいといわれているポインタと仲良くなるにはどうすればよいのでしょうか．

本章では，それにチャレンジしてみたいと思います．

8.1 C言語とメモリの関係を知る

● メモリって何者ですか

ポインタ（pointer）を一言で表現すると，「アドレスを用いて変数や関数などを操作する機能」ということができます．それでは，アドレスとは一体何でしょうか． ポインタの役目を確認する前に，アドレスの謎を解いていきましょう．そのために，まずはコンピュータが動作する仕組みとコンピュータの部品の一つであるメモリについて確認していきます．

私たちがコンピュータと呼んでいるものは，必ず共通した基本構成（5大要素）を持っています．具体的には，**表 8.1** に示す「入力」，「制御」，「記憶」，「演算」，「出力」という五つの機能です．そして，これらの機能をつかさどる装置を「5大装置」と表現し，コンピュータが処理を行うときは，これらの装置が必ず動作しています．

例えば，私たちがエディタで文字を入力する作業を考えてみます．コンピュータは，①入力装置（キーボード）から入力データ（文字）を受け取り，⑤出力装置（ディスプレイ）に結果を反映します．

表 8.1　コンピュータの各機能と装置の対応 (1)

	機　能	装　置	対応機器	人間で考えるならば
①	入力	入力装置	キーボード，マウス など	目，耳など（外部から指令を受ける）
②	制御	制御装置	CPU（Central Processing Unit）など	神経（命令を伝える，情報を伝える）
③	演算	演算装置		脳（考える，処理を行う）
④	記憶	記憶装置	（主記憶装置）メモリなど （補助記憶装置）ハードディスクなど	脳（記憶する）
⑤	出力	出力装置	ディスプレイ，プリンタなど	口，手足など（外部へ結果を伝える）

第9章

ポインタを使うプログラム例から，
その挙動を理解する

いろいろなポインタとその使い方

岡崎 光隆

前章では，ポインタの基本的な事柄について説明しました．本章では実際にプログラムを書いて，ポインタを使ったプログラムがどのように動くのかを目視しながら動作を確認し，ポインタの挙動を観察していくことにします．

9.1 C言語の変数とメモリの関係

プログラムが動作するとき，プログラムが扱うデータはメモリに格納されます．しかし，実際にメモリ上にデータが格納されている様子を見たことがある人は少ないと思います．データがどのようにメモリに入っているかを知っていると，ポインタの概念はぐっと分かりやすくなります．

そこでまず，プログラムのデータがメモリに格納されている様子を見てみましょう．**リスト9.1**のプログラムを見てください．このプログラムは4個の変数を定義して，その変数が格納されているメモリ上の場所を表示するプログラムです．

図9.1は，**リスト9.1**のプログラムを統合開発環境（IAR Embedded Workbench）上でデバッグ実行した様子です．プログラムが完全に終了して

リスト9.1　変数の格納位置を表示するプログラム

```
#include <stdio.h>
int main(void) {

  int i = 0x12345678;
  int j = 0x11223344;
  char s[] = "Hello Pointer World!";
  char *p = s;

  printf("&i=0x%08p¥n",&i);
  printf("&j=0x%08p¥n",&j);
  printf(" s=0x%08p¥n",s);
  printf("&p=0x%08p¥n",&p);

  return 0;
}
```

%08pはprintf関数の書式指定．引数をメモリ上のアドレスと見なして16進8桁で表示する

変数に&を付けると，その変数をメモリ上に格納しているアドレスを取得できる

配列変数の場合，配列の名前がそのまま配列が格納されているメモリの先頭要素のアドレスを表す

第10章

動く＆読めるプログラムを
作るテクニックを学ぶ

バグの入り込みにくいプログラムの書き方

山崎 辰雄

10.1 バグとは何か？

一般に，「プログラムが期待通りに動かないこと」を「バグ（虫）；bug」と呼んでいます．しかしそれは，いったい誰の「期待」でしょうか．作成者（あなた）の？ それとも使う人の？ 最初にプログラムを依頼した人の？ いろいろな立場によって，バグの内容は異なってくるものです．

もし，プログラムが期待通りに動かなかったとき，「あっ，それはバグだ」という一言で終わらせてしまっては駄目です．単にバグというと，「勝手に入り込んでしまったもの」という印象を持ち，責任者不在になってしまいやすいからです（**図 10.1**）．そのため，ソフトウェア・テストの専門家は「バグ」という言葉を使わずに，「誤り」や「欠陥」という言葉を使うようにしています．

プログラムが期待通りに機能しなければ，裁判になることもあります．そのときには，期待通りに機能しない「機能不全」を「瑕疵（かし）」と呼んでいます．

図 10.1 「バグ」が勝手に入り込んだ？

第11章

実行ファイル生成の流れを理解し，
トラブル解決に役立てる

ツール・チェーンとプリプロセッサを理解する

森 孝夫

最近のプログラミング環境は大変充実しており，極論すればワンクリックでソース・コードの作成から実行ファイルの作成までができるといっても過言ではありません．

しかし，実際に実行ファイルがどのような過程で作成されるのかを理解していれば，コンパイル・エラーに悩まされることなく，トラブルの解決に役立てることができます．

11.1 組み込みプログラミングとパソコン・プログラミング

C言語で作ったプログラムを実行させるには，以下に示す二つの処理が必要です．

(1) ビルド（build）：C言語のプログラムを実行形式に変換する

(2) ロード（load）：実行形式をハードウェア環境に読み込ませ，実行可能な状態にする

ただし，これらは組み込みプログラミングの場合もパソコン上で実行されるプログラミングの場合も必ず必要な処理ですが，これを行う手順は大きく異なります．

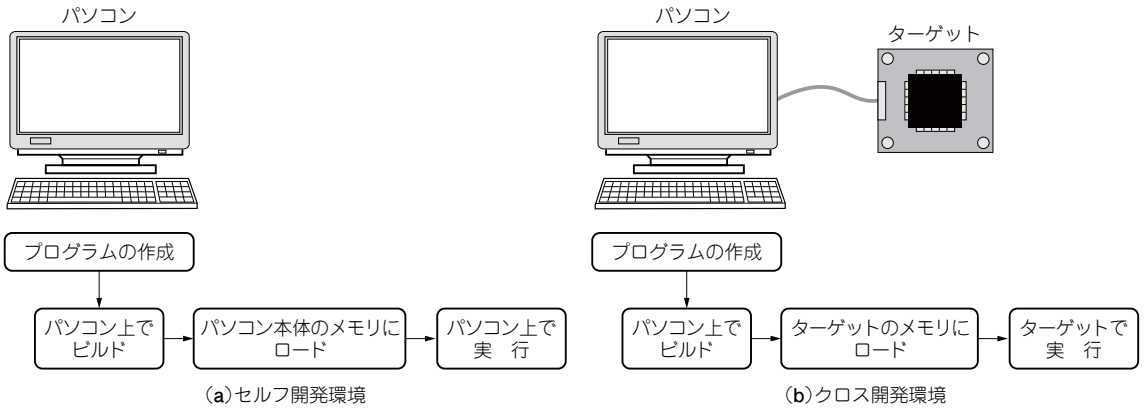

図 11.1　プログラムが実行される手順

第12章

知ってて得する修飾子から，
誰もがつまずく落とし穴まで

C プログラムに役立つノウハウ

舘 伸幸

本章では，C 言語のプログラミングに役立つノウハウを紹介します．ただし，C 言語はさまざまな環境のもとで使われているので，ここに掲載した内容が全ての環境で有効であるというわけではありません．

「使用できるメモリ容量が少ない場合に有効」，「コプロセッサが存在しない場合に有効」など，それぞれ活用できる前提が異なります．じっくり読んで，自分の環境ではどれが「使える技」なのかを見極めてください．

12.1 volatile を知っておこう

リスト 12.1 に示すプログラムは，①で変数 iFlag をクリアし，②でタイマ機能を開始する関数 voStartTimer() を呼んでいます．一定時間が経過すると，タイマ割り込みが発生し，その割り込みハンドラ・ルーチンで変数 iFlag に 1 を代入するようにしてあります．そのことを前提に，③のループでは iFlag が 1 になるまで，すなわちタイマが一定時間経過するまで待ち続けています．

このプログラムは，正常に動作するでしょうか？

コンパイラは，iFlag が割り込み先で書き換わる変数とは知りません．なので，最適化処理で③は単なる無限ループと同じと解釈し，while 文の条件などを消してしまうことがあります．このことを知らずにデバッグを始めると，いつまでたっても voWaitTime() が終了せず，タイマの設定や割り込みの設定をさんざん見直して悩むことになります．

● 困った「最適化」を防ぐ volatile

これを防ぐには，勝手に書き換わる変数 iFlag に volatile という修飾子を付けて，コンパイ

リスト 12.1　一定の時間が経過するまで待つプログラム

```
int iFlag;

void voWaitTime()
{
    iFlag = 0;          ←①
    voStartTimer();     ←②

    while( iFlag == 0 ){ ←③
        ;
    }

    return;
}
```

Interface増刊 インターフェースZERO No.01

CQ出版社 〒170-8461 東京都豊島区巣鴨1-14-2 TEL.03-5395-2141（販売部）

定価1,890円 本体1,800円

雑誌　01620-3
Ⓛ 2012.4/25

4910016200324
01800