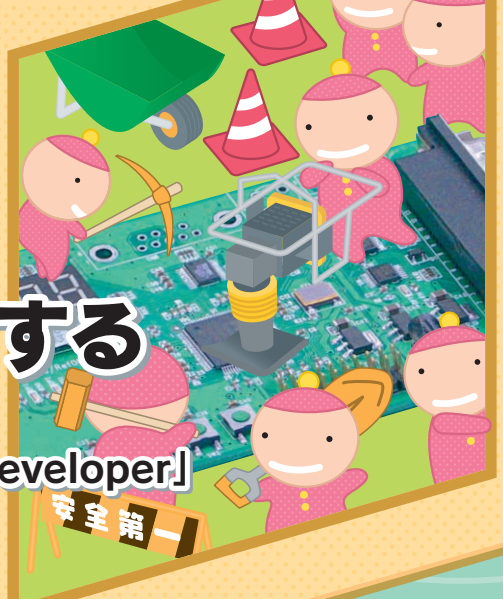
第1章

ANSI C言語によるハードウェア設計を体験する

Impulse Accelarated Technologies社のハードウェア/FPGA設計ツール「ImpulseC/CoDeveloper」

仲野 巧



C言語でハードウェアを設計するときに理解しておかなくてはならないことは，「ソフトウェアは逐次的で能動処理，ハードウェアは並列的で受動処理」であるということである．ハードウェア設計ツールを使って，ANSI C言語で設計を試みる．使用するツールは，米国Impulse Accelarated Technolgies社の「ImpulseC/CoDeveloper」である．このツールは，本誌付属のDVD-ROMに評価版が収録されている．（編集部）

ソフトウェア技術者が，C言語でハードウェア設計を行うにはどうすればよいでしょうか．それは，C言語でハードウェアを記述するには，次のような二つの違いを理解する必要があります．

一つは，逐次処理と並列処理です．ソフトウェアはCPUが命令を逐次実行しますが，ハードウェアは並列に動作します．もう一つは，能動的処理と受動的処理です．ソフトウェアはCPUで実行しているため，すべてのメモリ，すべてのI/Oなどの資源を利用できます．しかし，ハードウェア（周辺回路）はCPUが制御しているため，直接に資源を利用することはできません．

そこで，C言語でハードウェアを設計するには，並列処理のプロセス構造と，さらに受動的なC言語の記述に変更する必要があります．逆にいえば，この変更を理解していれば，簡単にC言語からハードウェア設計が可能になります．

本稿では，これらのプロセス構造とC言語の記述を例に，ANSI C言語によるハードウェア/ソフトウェア協調設計ツール「ImpulseC/CoDeveloper」を用いたハードウェア設計の進め方を説明します．

図1にC言語からハードウェア化するイメージを示します．この図は，ソフトウェアのアルゴリズムからハードウェアの並列処理へ変換するには，そのための記述の変更が必要であることを意味します．

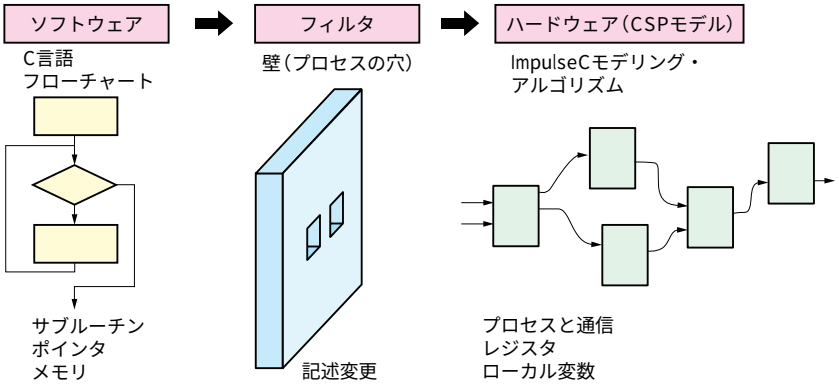


図1
C言語からハードウェア化へのイメージ
アルゴリズム（シーケンシャルのフローチャート）からハードウェア向きの並列処理のプロセス記述に変換するため，壁を通す（整形する）必要がある

Keyword ANSI C，逐次処理，並列処理，能動的処理，受動的処理，アルゴリズム，ハードウェア設計ツール，バブル・ソート，プロセス・モデル，エッジ検出，画像処理

1. ANSI C言語によるハードウェア設計ツールとは

ImpulseC/CoDeveloperは，ANSI Cで記述したアルゴリズムをハードウェア化できるツールです．C言語の開発環境と違和感のないWindowsベースの開発デバッグ環境を提供しています．ここでImpulseCは標準のANSI Cに新しいライブラリを追加したもので，CoDeveloperはハードウェア/ソフトウェア協調設計のシミュレータと動作合成を含む開発環境を意味します(1)．

ソフトウェア技術者は，従来のC言語のアルゴリズムを設計しながら，パソコン上でシミュレーションし，最終的にはハードウェアへの変換が可能です．このあたりが，ソフトウェア設計者向きのハードウェア設計ツールだといえるところだと思います．

C言語のサンプルでハードウェア設計を説明します．

HelloWorldはC言語では，有名なサンプル記述です．

```
#include <stdio.h>

int main (int argc, char *argv [])  {

    printf ("HelloWorld.") ;

    return (0) ;
}
```

このC言語のどこをハードウェアにしたらよいでしょうか．このプログラムはハードウェア化するアルゴリズムがない記述です．ここでいいたいのは，最初に必要なのは，ハードウェア化するアルゴリズムを明確にすることです．

2. アルゴリズムを記述する… BubbleSortを例に

アルゴリズムとデータ構造などでに出てくるバブル・ソートをハードウェア化してみます．

N個のデータが配列dataに格納されている場合のプログラム例を紹介します．バブル・ソートのアルゴリズムは，隣り同士のデータを比較して，大きいデータを交換しながら下に移動することで，1回のループで最大のデータが最下位に移動します．次のループでは，トップから，最下位から2番目までのデータを比較して，大きいデータを最下位から2番目に移動します．同様に，最上位まで繰り返すことで，データが泡のように移動していくソーティングなので，バブル・ソートと呼ばれています．

```
top = N;
k = 0;
tmp = 0;

while ( top >= 1 )  {
        k=0;
        while ( k < top )  {
                if ( data [k]  > data [k+1] )  {
                        tmp = data [k] ;
                        data [k]  = data [k+1] ;
                        data [k+1]  = tmp;
                }
                k=k+1;
        }
        top=top-1;
}
```

バブル・ソートは，アルゴリズムが明確です．ImpulseC/CoDeveloperを利用してバブル・ソートのシミュレーションを行ってみます．

ImpulseC/CoDeveloperは標準のコンパイラにMinGW(GCC)を利用しています．そのほか，Visual Studio 6やVisual Studio.Netなどのソフトウェア環境も利用できます．これもソフトウェア技術者が使いやすいツールになっている点です．

バブル・ソートのプログラムは，付属DVD-ROMのImpulseフォルダの中のSoftwareフォルダに入っています(**リスト1**)．

Softwareフォルダのプロジェクト・ファイルSoftware.icProjをダブルクリックしてImpulseC/CoDeveloperを起動します．ソフトウェアのシミュレーションだけですから，ProjectExplorerにはSoftware.cのみが登録されています．Software.cをクリックすると，C言語のプログラムが表示されます(**図2**)．

アルゴリズムをシミュレーションするためには，ソフトウェアの開発環境と同じようにコンパイルして，exeファイルを作成し，シミュレーションを実行します(**図3**)．

▲コンパイル　「Project」→「Build Software Simulation Executable」

▲シミュレーション　「Project」→「Launch Software Simulation Executable」

このように，ソフトウェア設計者は，ImpulseC/CoDeveloperをアルゴリズムの開発デバッグ環境として利用できます．

3. ハードウェア化のためのソフトウェアの変更

C言語のアルゴリズムをハードウェア化するために，ソ