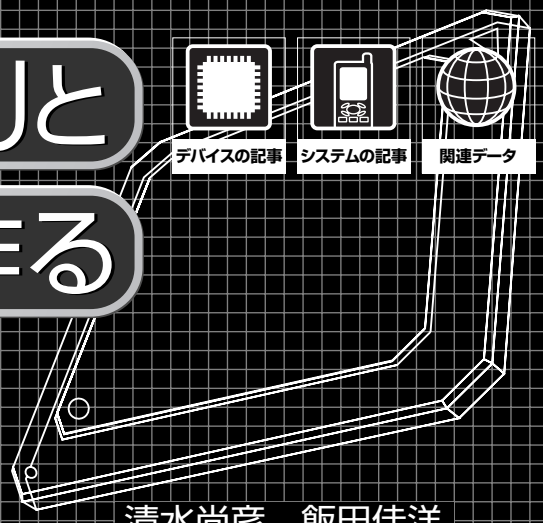
FPGA搭載用 CPUとソフト開発環境を作る



第2回

Cコンパイラ，アセンブラの開発（gcc/gasの移植）

清水尚彦，飯田佳洋

前回は，組み込み機器向けの16ビットCPUのソフト・マクロを開発しました．今回は，そのCPU用の開発ツールを用意します．GNU Cコンパイラには PDP-11移植版が存在するのですが，これはコンパイラのmakeすら通らないものです．そこで，そのPDP-11移植版を修正しました．また，修正したツールを使って簡易モニタ・プログラムを開発しました．（筆者）

PDP-11を動かそうと考えたとき，真っ先に頭に浮かんだのはUNIX V6を動作させることでした．実際にCPUの設計が進み，思ったよりコンパクトにまとまってくると，これは組み込み用途や教育用におもしろい展開ができるかもしれないと思いました．

しかし，世の中にあるPDP-11用の開発ツールは古すぎて，実用にはほど遠いものです．UNIX V6のソース・コードを読んだことのある方ならご存じだと思いますが，Kernighan & Ritchie[注1]スタイルのCはとても現代の開発言語として使えるものではありません．またOSも，UNIXに代わるコンパクトなものが必要です．組み込み用途を考えれば，リアルタイムOSを用いたいところです．

PDP-11用のANSI C互換コンパイラを探していたところ，GNU Cコンパイラ（gcc）がPDP-11に対応していることを知りました．そこでgccを使おうと考えたのですが，そんなに世の中は甘くありません．gccのPDP-11移植版は不完全で，コンパイラのmakeすら通らないものでした．

そこで，gccのPDP-11移植版の修正を行いました．また，移植の過程でGNUのアセンブラ（gas）のPDP-11用移植版にもバグがあることがわかり，こちらも合わせて修正しました．どちらもゼロからの移植ではありませんが，コンパクトなプロセッサへの移植経験は，独自アーキテクチャへの移植のためのステップにもなります．

gccのバグを修正する

本記事では，gccのversion 3.2を移植のベースとしました．GNUのアーカイブはRING Server（http://www.ring.gr.jp/）などに保存されています．以下のURLからgcc-3.2のフルセット（gcc-3.2.tar.gz）を入手します．

http://www.ring.gr.jp/archives/GNU/gcc/

また，アセンブラやリンカなどのツール群binutilsも利用するので，こちらも入手します．binutilsはversion 2.13（binutils-2.13.tar.gz）を使用しました．

http://www.ring.gr.jp/archives/GNU/binutils/

注1：Brian Kernighan氏とDennis M. Ritchie氏は，C言語の開発者である．

〔図1〕パッケージの展開

```
tar xzf binutils-2.13.tar.gz   ← binutils-2.13.tar.gzを展開
mkdir build-binutils           ← build-binutilsディレクトリを作成
tar xzf gcc-3.2.tar.gz         ← gcc-3.2.tar.gzを展開
mkdir build-gcc                ← build-gccディレクトリを作成
```

```
/home/nshimizu/
└ gcc
   ├ binutils-2.13
   ├ build-binutils
   ├ gcc-3.2
   └ build-gcc
```

〔図2〕ディレクトリ構造

筆者は/home/nshimizu/の中にgccディレクトリを作成し，その下で作業を行った．

〔図3〕binutilsのインストール

```
cd build-binutils          ← build-binutilsディレクトリに移動
../binutils-2.13/configure --target=pdp11-none-aout
                           ← configureコマンドの実行
make                       ← makeコマンドでコンパイルを行う
make install               ← インストールを行う(root権限で)
```

〔図4〕gccのインストール

```
cd build-gcc               ← build-gccディレクトリに移動
../gcc-3.2/configure --target=pdp11-none-aout
            --with-included-gettext --enable-languages=c
                           ← configureコマンドの実行
make                       ← makeコマンドでコンパイルを行う
```

〔図5〕gccコンパイル時に発生したエラー(一部)

```
cc1: 警告: `-g': 不明またはサポートされない -g オプションです
../../gcc-3.2/gcc/libgcc2.c: 関数 `__muldi3' 内:
../../gcc-3.2/gcc/libgcc2.c:362: 警告: `__x0' はこの関数内で初期化されずに使用される可能性があります
../../gcc-3.2/gcc/libgcc2.c:362: 警告: `__x1' はこの関数内で初期化されずに使用される可能性があります
../../gcc-3.2/gcc/libgcc2.c:362: 警告: `__x2' はこの関数内で初期化されずに使用される可能性があります
../../gcc-3.2/gcc/libgcc2.c:362: 警告: `__x3' はこの関数内で初期化されずに使用される可能性があります
/tmp/ccgb0JUk.s: Assembler messages:
/tmp/ccgb0JUk.s:16: Error: Unknown instruction 'movdi'
/tmp/ccgb0JUk.s:190: Error: Unknown instruction 'movdi'
/tmp/ccgb0JUk.s:256: Error: Unknown instruction 'movdi'
/tmp/ccgb0JUk.s:264: Fatal error: Case value 6 unexpected at line 553 of file "../../binutils-2.13-pdp11/
                                                                              gas/config/tc-pdp11.c"
```

なお，開発環境はLinuxなどのUNIX系OSであることを前提としています．Windows環境で開発したい場合は，Cygwinなどを用いてください．

●とりあえずビルドして問題点を探し出す

まずは，ダウンロードしたパッケージを展開してみます．また，パッケージを展開したディレクトリとは別に，ツールをビルドするためのディレクトリを作成します(**図1**，**図2**)．

次にconfigureコマンドを用いて，PDP-11用のツールを作成します．まずはbinutilsのほうから作成します(**図3**)．これで，とりあえず第1段階のbinutilsが完成します(実はこのバージョンのbinutilsにはバグがあるのだが，詳しくは後述する)．

次に，gccを作成します(**図4**)．するとlibgcc2.cのコンパイルでエラーが発生します(**図5**)．そこで，原因を追求して問題点を解消していく作業に入ります．**図5**を見るとアセンブリ・ソース(ccgb0JUk.s)でエラーを出しているので，アセンブリ・ソースを抽出する必要があります．gccでアセンブリ・ソースを出力するには，-Sオプションを付けます．build-gcc/gccディレクトリに移動し，**図6**のコマンドを入力してアセンブリ・ソース(libgcc2.s．**リスト1**)を出力します．

図5によると，問題となっているのはmovdiの部分です．この記述は明らかにコメントですが，binutilsのプログラムでは;は行の区切りになるだけなので，コメントとは見なさずエラーとなっていたことがわかります．それ以外にも，定数表現などでいろいろと問題になりそうな部分が見つかります．

おそらく，binutilsの最初の移植はDECアセンブラ互換をねらったものだったのでしょう．しかし移植が不十分で，通常のbinutilsの構文とDECアセンブラの構文が中途半端に混在し，問題の原因となっているようです．

●変更方針：組み込み用途に合わせる

gccを修正するにあたり，以下を変更方針とします．

- 定数表現，レジスタ表記などは，binutils標準の記法になるべく沿うように変更する．ただしDECアセンブラを利用したい場合を考慮して，コンパイラの出力はDECアセンブラにも使えるようにしておく．
- 組み込み用途を考えて，拡張命令セット(EIS)や浮動小数点のない構成もオプションで指定できるようにする．これらは，コンパイルのモデル・オプションで指定する．
- 標準ライブラリは用意しない．その代わり，米国RedHat社が提供するnewlibを利用可能にする．

以下のURLに，newlib-1.10.0用のパッチを用意しました(現時点ではsetjump関連のアセンブリ・ルーチンを書いていないため開発途上だが，newlibのコンパイルは通るはず