序章

# GCCとは何か？



GCC(GNU Compiler Collection)のことを説明するにあたって，Linuxのことも簡単に説明しておいたほうが話は早いでしょう．

本書を読む皆さんの多くはLinuxシステムをお使いだと思います．そのカーネルは1991年にLinus Torvalds氏によって公開されました．そしてその著作権は「GNUライブラリ一般公有使用許諾」(GNU GPL：GNU General Public Lisence)によって保護されています．

皆さんはLinuxシステムでどのようなことを行いますか．インターネット接続やプログラム開発，あるいはサーバとして使用したり，いろいろなアプリケーションを動作させると思います．

Linux Kernelだけでは何もできません．コンパイラや圧縮ツール，メール配送，そのほかのいろいろなプログラムが必要です．Linus Torvalds氏がLinux Kernelを開発した時点で，それらのアプリケーションはすでにこの世に存在していました．Linuxシステムはそのアプリケーションやツールを Kernelに組み合わせて完成させたものです．そのアプリケーションやツールは一般的にGNUツールと呼ばれています．もちろん，この著作権もGPLによって保護されています．このようにGNUとLinuxは不可分のものであり，本来はGNU/Linuxシステムと呼ばれてしかるべきものです．

Linus Torvalds氏は規定によって公開されているGNUツールのソース・コードをkernelに組み込み，現在のLinuxシステムが完成したのです．

### ● GNUプロジェクトについて

GNUプロジェクトは，1984年にRichard M. Stallman氏を中心として作られました．

GNUとは「GNU is Not Unix」の頭文字を取った略号であり，UNIXと上位互換性のあるフリーな総合ソフトウェア・システムの名称です(**図1**)．

kernelも本来はGNUプロジェクトで開発されるはずでしたが，Linuxがあるので現在はそれが使われています．

### ● GPLに関して

GNUプロジェクトが提唱するフリー・ソフトウェアのライセンスのことをGPLと呼びます．ソフトウ

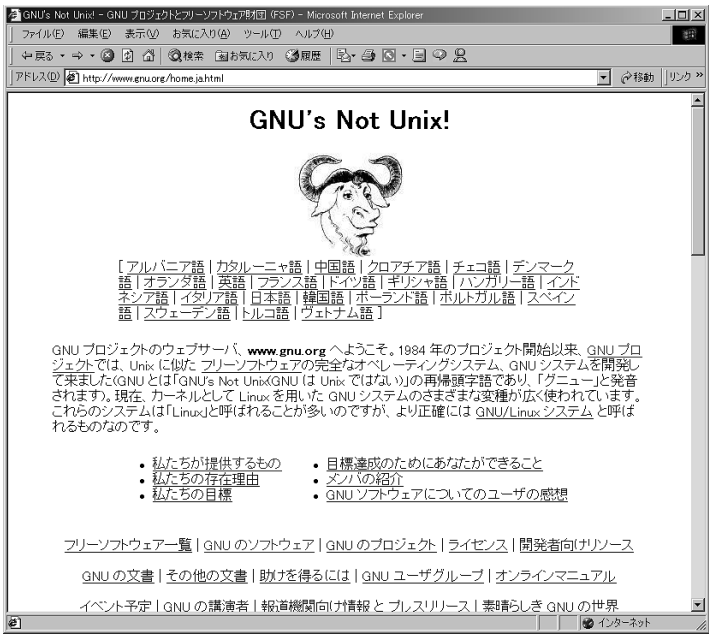


**図1**
**GNUプロジェクト**
(http://www.gnu.org/home.ja.html)

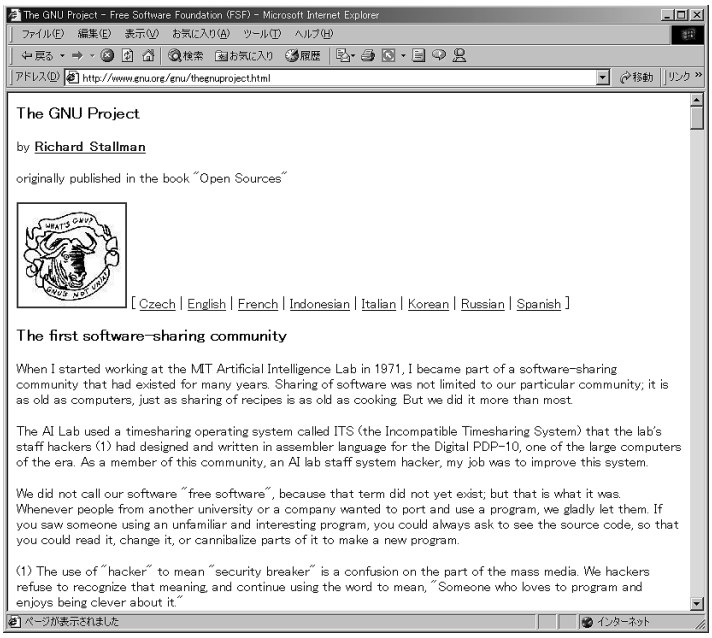


**図2**
**GNU プロジェクトに関する詳細**
(http://www.gnu.org/gnu/thegnuproject.html)

ェアとそれを使用するユーザに，使用・複製・変更・再配布の自由を与えることを目的としています．

フリーソフトウェア財団(Free Software Foundation：FSF)がソフトウェアにさまざまな自由を与える権利として提唱しているコンセプトである「Copyleft」と，そしてそれを保証するためのライセンスがGPLです．GNUプロジェクトの成果物は，そのほとんどがGPLによって配布されています．

GPLのソフトウェアは必ずソース・プログラムとともに頒布，複製されます．もしソース・プログラムを付けずに頒布する場合は，ソース・プログラムを確実に入手できる手段を提供することが義務付けられています．

そして，下記のような条件が付けられます．

① ソフトウェアを，使用・複製・変更・頒布したり，新しいオリジナルなフリー・ソフトウェアの一部として利用できること
② 変更・改良されたソフトウェアはGPLに従って頒布されること
③ プログラムの全部あるいは一部を用いて作られたソフトウェアはGPLに従って頒布されること
④ 基本的に無保証であり，そのソフトウェアが原因でトラブルが生じても作者に責任はないこと

なお，LGPL(GNU Lesser General Public License)というライセンスもあり，これは非フリーなモジュールとのリンクを認めています．

### ● GNUツールに関して

GCCを使うことはできても，それを使いこなしている人はまだまだ少数だと思います．

たとえばGCCのコマンド・オプションについての情報はあっても，それを使うとどうなるのか，またどのようなプログラムができるのかといった情報はインターネット上でさえも皆無に等しい状態です．英語で書かれた海外の情報については深く調査していませんが，それを見つけても翻訳しながら使うのはなかなかめんどうなことです．

本書ではGNUツールの中でGCCに関して徹底的に使い込んでみたいと思います．

GNUツールは，1984年にプロジェクトが開始されたそうです．詳しいことは，

```
http://www.gnu.org/home.ja.html
```

でいろいろな情報が得られます．

なお，GNUプロジェクトについてはRichard M. Stallman氏の著作である書籍『フリーソフトウェアと自由な社会』(アスキー)に詳しく書かれています．

GNUの「フリー・ソフトウェア」に関する考え方をそこから抜粋します．

- いかなる目的でもそのプログラムを実行する自由がある
- 自分のニーズに合わせてそのプログラムを修正する

自由がある（この自由を実際に活用するためには，ソース・コードを自由に入手できなければならない．ソース・コードなしにプログラムに変更を加えるのは極めて困難なため）

- その複製を無料でも有料でも再配布する自由がある
- コミュニティがあなたの改善作業によって恩恵が受けられるよう，修正したプログラムを配付する自由がある

よく誤解されているのですが，「フリー」の意味があります．これは自由のことをいうのであって，値段のことではありません．

**図2**に示すWebページにGNUプロジェクトに関する文章があります．興味のある方はお読みください．

GNUツールの中には，たくさんの言語コンパイラやそのほかのツールがあります．はじめに皆さんが一番使うであろうGCCに関する事柄について述べます．

## GCCの機能

現在の最新バージョンは2005-07-07に公開された4.0.1です．本書では，主として3.3.5のオプションを扱っていますが，4.0との違いを表と文章でも説明しています．以降，特に断りがない場合，GCCはFSF版GCCのことを指します．GCCの公式サイトは，

```
http://gcc.gnu.org/
```

です．

また，GCCはC言語だけのコンパイラではありません．GCCコンパイラは，C言語，C++，Objective C，Fortranで書かれたプログラムをコンパイルすることも可能です．本書では，C言語の機能について説明していきます．なお，GCCの機能を有するのはFSFのGCCだけではありません．

まず最初に，EGCS（Experimental/Enhanced GNU Compiler System）が挙げられます．これはgcc-2.8.xをベースとして数値演算関係の最適化能力を中心に，さまざまな改良を施したコンパイラです．GCCのβ版としてEGCS Steering Committeeによって開発・改良が行われていました．1999年4月27日にFSFは，Cコンパイラ『egcs』を開発しているEGCS運営委員会（EGCS Steering Committee）が，GNUのメンテナとして，GNU Cコンパイラ（GCC）の開発を引き継ぐことを発表しました．これによって，EGCS運営委員会はGCC運営委員会（GCC Steering Committee）と名称を変え，今後のGCCのリリースに責任を負うことになりました．

二つ目のPGCC（Pentium GCC）は，GCCをベースとしてPentiumプロセッサやその互換チップでの性能向上に重点を置いたものです．pgcc-1.1.3ではMMXオプションが追加されました．

ちなみに，Linuxの配布パッケージの一つである「Stampede GNU/Linux」は，そのすべてのバイナリがPGCCでコンパイルされています．Stampedeによれば「一般の配布パッケージと比較して20%～30%ほどパフォーマンスが上がっている」とのことです．

三つめのAGCC（Athlon GCC）は，PGCCをベースとしてAthlonプロセッサをターゲットに最適化を施したコンパイラです．同時にPGCCのバグ・フィックスも行われているので，Athlonユーザでなくとも使う価値はあるでしょう．

ここで述べたEGCSやPGCC，AGCCはFSFが直接かかわっているものではありません．前述の「フリー・ソフトウェア」の考え方によって別のプロジェクトとして作られています．

しかし，PGCCやAGCCの性能が優れているため，現在では本来のGCCに吸収され，プロジェクトも統合されています．

なお，WindowsではGNUツールを動作させるエミュレータとしてCygwinがあります．この上でGCCを動かすことが可能です．

GCCは多くのプロセッサで動作するため，組み込み分野でも広く使われています．GCC本体とGCCでコンパイルしたバイナリは16ビットから64ビットまで数多くのCPUで動作します．もちろん標準的なLinuxディストリビューションにはGCCが付属していますが，組み込み用の場合は自分でインストールしなければなりません．

最新の4.0では以下のようなハードウェア・コンフィグレーションが可能です．

```
M680x0 Options
M68hc1x Options
VAX Options
SPARC Options
ARM Options
MN10300 Options
M32R/D Options
IBM RS/6000 and PowerPC Options
Darwin Options
MIPS Options
```

```
Intel 386 and AMD x86-64 Options
HPPA Options
Intel 960 Options
DEC Alpha Options
DEC Alpha/VMS Options
H8/300 Options
SH Options
Options for System V
TMS320C3x/C4x Options
V850 Options
ARC Options
NS32K Options
AVR Options
MCore Options
IA-64 Options
D30V Options
S/390 and zSeries Options
CRIS Options
MMIX Options
PDP-11 Options
Xstormy16 Options
FRV Options
Xtensa Options
```

## C言語とGCC

C言語は1972年にアメリカのAT&Tベル研究所において，Dennis M. Ritchie氏によって設計されました．参考となった言語はMartin Richards氏が開発したBCPL言語，そしてそれを参考にしてKen Thompson氏が開発した「B言語」です．

当初の目的はミニオペレーティング・システムであるUNIXを開発することでした．しかし，UNIXをはじめとする各種OS上で動くアプリケーション開発にも用いられるようになり，世界的に普及したため，ほかのOSやメインフレームにも普及しました．初期のパーソナル・コンピュータにおいても，BASICにあきたらない人々がC言語でプログラムを作ったので，普及に弾みがつきました．

もちろんUNIXやLinuxだけではなく，Windows，そしてMacOSもC言語で開発されています．あらゆるアーキテクチャに広まったため，初期の頃は，使用するコンパイラによって仕様が異なるという問題もありました．

それは，C言語の標準とされていた文献である「プログラム言語C」(共立出版)にいくつかあいまいな部分が存在していたためです．

そこで，ISOとANSIは共同でC言語の規格の標準化を進めました．その結果，1989年12月に規格が策定されました．これは「C90」と呼ばれています．

1999年12月には，ISOで規格の改定が行われ，新たに規格が策定されました．これは「C99」と呼ばれています．「C99」については第13章で解説します．

このような歴史のあるC言語に新たな潮流を作り出したのが，Richard M. Stallman氏です．GNUプロジェクトの創始者である彼は数々のフリー・ソフトウェアを作る道具としてGCCを企画しました．

それはフリー・ソフトウェアを商用ソフトウェアでコンパイルすると著作権の問題が発生するためです．

1980年代中期から開発されたGCCは，1994年にバージョン2がリリースされて実用的レベルに達しました．もちろんRichard M. Stallman氏だけではなく，Steve Jobs氏のNextComputer社，また世界中の優秀な「ハッカー」(クラッカーではない)の手に助けられてリリースされたことは言うまでもありません．

## GCCと組み込み

GCCは組み込み分野でも大いに役立っています．かつて，組み込み機器の開発にはアセンブラが使用されていました．極限までプログラムのサイズを小さくするためであり，商用のC言語コンパイラが高価だったためでもあります．時と場合によっては「ハンド・アセンブル」でコードを書かなければなりませんでした．現在はGCCのターゲットCPUが多彩になったため，クロス・コンパイラさえ構築すれば，開発工数を大幅に減少させることができます．また，独自OSで動作する組み込みが当たり前だったため，デバッグにも苦労したものですが，今はGDB(GNU Debugger)が使えるため，かなり楽になりました．

実用的な話としては，Linuxザウルスで動作するアプリケーションを作りたい場合は，XScale向けのクロス・コンパイル環境を別のパソコンに構築すれば十分です．小さなWebサーバを動作させて楽しむのもまた一興です．

もちろんQtopiaの実行環境もパソコン上に構築できるのでGUIツールを作った場合のテストも問題なくできます．