第2章

OSを動かすまでの技術を総覧する

ブート・ローダの役割と開発の流れ

日高 亜友

ミュンヒハウゼン男爵は，18世紀のドイツの軍人で，若いころにロシアのフリードリッヒ大王の騎兵隊長を務めた後，ヨーロッパ各地を渡り歩き，狩猟家，冒険家としても知られています．しかし彼を有名にしたのは，軍役を終えて故郷に帰ってから彼が周囲に聞かせた物語です．

この数々の話は「ほらふき男爵の冒険」として後に出版され，また映画化もされました．また彼の名を取ったミュンヒハウゼン症候群は，健康であるにもかかわらずに自分があたかも病気であるかのように訴えたり，病院巡りをしたりする精神病の症状として知られています．

この物語では，男爵は砲弾に乗るという驚異的な経験をし，それで月にも旅行し，あるときには自分で自分の髪の毛を引っ張って沼から這い出たとあります．そして後年出たこの話の別版では，湖から自分を引っぱり出すために，自分のブートストラップ（ブーツを履く時に引っ張り上げる「つまみ革」）を使用したとあります．「bootstrap」は，「自力で成し遂げる」という英語の意味がありますが，このほらふき男爵の冒険の話から出た言葉であるという説が有力なようです．

1．ブート・ローダを理解するための各種用語

● IPL，ブート・ローダ，MBR

コンピュータに電源を投入したときに，最初に起動して，ほかのプログラムやオペレーティング・システム（以下OS）を動かすプログラムをIPL（Initial Program Loader）と呼びます．文献によっては使い分けている例もありますが，多くの場合はこのIPLを「bootstrap loader」とも呼んでいます．ブート・ローダ（bootloader）という用語はこれの短縮形で，一般にはIPLと同意に用いられています．

また，IPLやブート・ローダという語は，パソコン（IBM PC互換機）の環境においては，ディスク・ドライブの最外周にあるMBR（Master Boot Record）部分に書かれている，BIOSが起動して最初に実行するディスク上のプログラム（別名マスタ・ブート・コード）のことを指す場合もありますが，もともとは前述のより広い意味で使用されるものでした．

パソコンでは電源投入時に，最初にROMに搭載されたBIOSが実行されます．パソコンでさまざまな種

コラム 1　BIOS とは

古い時代のシステムでは，デバイス・ドライバの一部とブート・ローダに相当するものを BIOS（Basic Input Output System）として ROM に組み込み，システムの抽象度を上げ，アプリケーションや各種 OS をより汎用的な環境で動作させてきました．

米国 IBM 社製パソコンとその互換機にもこの BIOS が採用され，初代 IBM 機とは異なるハードウェア構成でも，BIOS が機器構成の違いを吸収し，同じ MS-DOS や Windows を起動できるという互換性を維持してきています．さらに BIOS は，ACPI によるパワー・マネジメント機構や PnP（Plug and Play）などの OS に依存しないシステム固有の制御コードを組み込むためにも不可欠です．

また，近年では，BIOS 内で USB キーボードを PS/2 キーボードと同等に処理するというように，旧来（レガシ）のインターフェースに BIOS で新技術をマップすることで，進化し続けてきています．

GNU プロジェクトの推進者の 1 人に FSF（Free Software Foundation）のリチャード・ストールマン氏がいます．同氏は 2005 年 2 月に，パソコン用のフリー BIOS の開発を呼びかけました．その理由は，BIOS 開発メーカやパソコン・メーカによって BIOS に実装されているパワー・マネジメントやシステム管理などの最新技術のコードが公開されないため，Linux をはじめとするオープン・ソース系 OS の開発に支障があること，BIOS のノウハウが公開されず，開発が一部の BIOS 専門メーカに限られていているため，導入コストが非常に高くつくことです．そのほかにも，従来からの BIOS には次のような問題があります．

- 周辺機器が多い場合は特に初期化コードの実行に時間がかかる
- リアル・モードで動作するため実行するプログラムに制限がある
- システム起動時や BIOS 更新時に MS-DOS 互換 OS または機能を必要とする
- BIOS が管理するパラメータのインターフェースが公開されていない

結局同氏の思いは GNU としては実現できず，2006 年 12 月に志を同じくする LinuxBIOS プロジェクトに引き継がれました．もともと LinuxBIOS は，サーバ・マシン向け Linux 専用オープン・ソース BIOS として開発されていました．OLPC（One Laptop Per Child；開発途上国の子供たちを中心に 1 人 1 台 100 ドル程度のパソコンを供給する NPO による国際プロジェクト）に採用され，開発が進みました．

2008 年 1 月に LinuxBIOS は coreboot と名前を変え，サポートの範囲を Linux に限定しなくなりました．coreboot はオープン・ソースで高速な LinuxBIOS の特徴をそのまま受け継ぎながら，現在では BIOS 専門メーカ並みにサポートする豊富なチップセットのリストを公開し，Windows 7 の起動にも成功したと報じられています．

これに対して米国 Intel 社では，IA-64（Itanium）で採用した EFI（Extensible Firmware Interface）という従来の BIOS に変わる新しい規格の普及を目指し，他社からも賛同を得ています．EFI はインテル・アーキテクチャの Macintosh も搭載し，x64 向け OS 側の対応は，Windows Vista SP1 から行われています．もともと EFI は Intel 社で開発されましたが，現在は普及を促進するために仕様の管理権限を Unified EFI Forum へ移しました．そのため，UEFI の規格として広く知られ，採用されるようになってきています．

さらに最近では，メーカ製 BIOS を利用せずに PC アーキテクチャ・システムを起動するという技術も，各所で研究・開発されています．たとえば Linux カーネル 2.6 の KEXEC は，BIOS を介さず迅速にシステムの再起動を行うため，システム再構成やカーネル更新時に，エンタープライズ・サーバの停止時間を短縮できます．また IBM 社では，自社マシン向けに本文中で紹介した Open Firmware を積極的に導入しています．

参考 URL

(1) FreeBIOS；http://sourceforge.net/projects/freebios/
(2) coreboot；http://www.coreboot.org/
(3) Linux のためのオープン BIOS；http://www.ibm.com/developerworks/jp/linux/library/l-bios.html

類の OS を起動したり，あるいはインストールしてある複数の OS を選択起動することができるのは，BIOS から呼び出されて動作する，MBR に書いてあるこの IPL と，それに続いて起動される一連のプログラムの働きによります．このように，Windows 2000/XP の起動時に使用される NTLDR や，Linux の起動で利用される LILO や GRUB は，ブート・ローダとも呼ばれるので注意が必要です．また，システムコマンダーなどの各種 OS の切り替え起動を目的とするツールは，ブート・マネージャと呼ぶこともありますが，同様に MBR の IPL 起動のしくみを利用して実装しています．

コラム２　以前はROM，最近はフラッシュ・メモリに搭載

ブート・ローダは，マスクROMやPROM，紫外線消去タイプのEPROM(Erasable and Programmable ROM，別名UV-EPROM)に代わり，最近では一般的に，ボード上に実装したまま書き換えができるフラッシュ・メモリに搭載されています．フラッシュ・メモリは，電気的にメモリの一部または全部を消去して再書き換えできるEEPROM(Electrorically Erasable and Programmable ROM)の一種で，高速にアクセスし，大容量化しやすく，主にNOR型とNAND型に分類できます．

NOR型はランダム・アクセス性能に優れ，バイト単位で書き込みが可能なため，ブート・ローダやファームウェアといったプログラムや，システム・パラメータの保管に使用されます．

NAND型は，ブロック単位での読み書きしかできませんが，ビット当たりの単価が安く，消去や書き込みの速度も速く大容量化に向いているため，デジタル・カメラなどの携帯機器の可搬型外部記憶デバイスとして使用されています．ただし，NAND型は，読み出し時にビット・エラーが発生する場合があるため，冗長ビットを付加してエラー訂正を行い，利用セルの再配置を行うなどの工夫をして信頼性や寿命を向上させる必要があるため，制御用のハードウェア回路やソフトウェアが必要になります．

最近のLinuxでは，MTD(Memory Technology Device)としてROMを扱い，ファイル・システムとしてマウントして使用できるようになりました．NAND型フラッシュ・メモリもサポートされていますが，これを利用するブート・ローダやファームウェアの開発には注意が必要です．

そのほか，16ビット程度までの比較的小規模の組み込み系CPUでは，CPUチップにフラッシュ・メモリやRAMまで内蔵し，外付けメモリがなくてもシステムを構築できるように配慮しているものも普及してきています．また，大容量の不揮発性メモリを必要とするシステムの開発では，比較的コストを安価にできるバッテリ・バックアップRAMを使用する場合もあります．

いずれの場合もブート・ローダはシステムの起動時に一度しか実行されませんが，OSをもつコンピュータ・システムには不可欠なものです．

● ブートROM，ファームウェア

パソコンのBIOSを含めた広い意味のブート・ローダは，不揮発性メモリ(ROM)に格納されていることから，このROMをブートROMと呼びます．組み込みシステムでは，ROMの中にブート・ローダだけではなく動作プログラム(アプリケーション)やOSまで格納する場合があります．この場合，システム全体を制御するプログラムが入っているROMをブートROMと呼びます．

ファームウェアは，特定のハードウェアを制御するために機器や周辺コントローラに組み込んで使用するソフトウェアです．一般的にファームウェアはROMに格納されていますが，中には機器の起動時に通信手段を経由して，外部から転送して利用する場合もあります．

● デバッガ，モニタ，モニタ・デバッガ，ROMモニタ

一般に組み込み用のCPUメーカやボード・メーカ，リアルタイム・オペレーティング・システム(以下RTOS)メーカでは，ソフトウェアの開発や移植を行いやすくするために，実機システムのROMに組み込むための，ユーザ・プログラムのロードやデバッグをサポートするプログラム(ROMモニタ)を提供しています．デバッガは特にブレークポイントやトレースなどのデバッグ機能までをもつものを指し，そこまでの機能がないものをモニタと呼びます．実際上は明確な使い分けはなく，総称してモニタ・デバッガと呼ぶ場合もあります．

これらは多くの場合，ブート・ローダの機能ももつため，そのままか，または改造してブート・ローダとして使用します．

デバッガやモニタは，無料で配布されたり，各メーカのWebサイトなど，インターネット上で公開されている場合が多いのですが，製品に組み込んで使用する場合には，利用条件やライセンス内容について確認する必要があります．

● OSのブート・コード

OSの種類によっても異なりますが，起動時にOSのすべてのコードをメモリに展開して実行するのではなく，段階的に起動する場合があります．OSのコードの中でも，次のようなカーネル本体の起動時だけに実行する部分をOSのブート・コードと呼びます．組み込みLinuxなどでは，カーネルのロード方法や動作環境に合わせてOSのブート・コードを修正または作成する必要があり，この項目とブート・ローダが移

植の際には技術的なポイントとなります．

a）カーネルの動作に必要なメモリ領域，CPU，RAM，割り込みベクタの設定
b）起動に必要なファイルをメモリ上に展開するRAMDISK を作成
c）圧縮したカーネルやRAMDISK イメージを読み込んでメモリに展開
d）ファイル・システムのマウントに必要な外部記憶装置やネットワークとUI の初期化

● リセット，ハードウェア・リセット，ソフトウェア・リセット

電源ON 時は，CPU ごとに実行する命令のアドレスが決まっています．ハードウェア・リセットは，リセット・ボタンなどの外部からの要求で，システムを構成するCPU をはじめとする各種コントローラに電気信号的にリセット信号を送ることにより，電源ON 直後と同様の状況を作り，起動動作から再実行することをいいます．

これに対して，ソフトウェア・リセットは，リセット信号の助けを借りずにソフトウェアでCPU をはじめとする各種コントローラを初期化して，電源ON 時と同じアドレスから起動し直すことをいいます．このときにどの程度の初期化を行うかについては，ソフトウェア・リセットを実行するコードに依存しますが，通常はソフトウェア・リセットを行っても問題なく動作するようにシステムを開発します．

一般にリセットというと，ハードウェア・リセットを指します．本章では，以降リセットと電源ON が同じ動作をすると仮定して解説します．

コラム 1 にBIOS に関する簡単な解説を，コラム 2 にはROM とフラッシュ・メモリの解説を載せました．以降は，パソコン・アーキテクチャにはあまりとらわれずに，より広い意味でのブート・ローダ，特に組み込み系システムでのブート・ローダを中心にして説明していきます．

2. ブート・ローダの種類と現状

それでは実際に，どのようにしてブート・ローダは開発されているのでしょうか．その前に，以下に一般的にブート・ローダと呼ばれるものや，ブート・ローダの機能を実現しているものを例示して，コードの入手方法の観点から種類別に整理します．まず，**表 1** に主なブート・ローダをまとめて整理しました．

● 開発，配布，流通形態による分類

▶オープン・ソース系（例：RedBoot，Das U-boot）

主にボランティアの手によって開発され，インターネットで公開されているブート・ローダです．もともとブート・ローダとして開発されているため，完成度は高いのですが，対応するアーキテクチャはある程度限定されています．

▶CPU メーカ提供（例：ルネサス テクノロジH8 モニタ，Motorola BUG）

CPU やボード・メーカが主に製品の評価目的に提供するROM モニタやデバッガです．無償または比較的安価で入手できますが，機能や拡張性が乏しい場合があります．

▶OS メーカ製提供（例：VxWorks，Windows CE など）

組み込み系OS メーカでは，自社のOS を起動するためのブートROM に搭載するブート・ローダを供給しています．実行・デバッグ環境との相性もよく，サポートも受けられますが，それなりの費用がかかります．

▶ボード・メーカ提供（例：E!Kit-1100 など）

マイコン評価ボードを購入すると，サポートするOS を起動するためのブートROM が付属します．オープン・ソース系CPU メーカ提供のものに，独自に手を入れたものが多いのですが，中には独自にブートROM を開発しているメーカもあります．

▶専用メーカが提供（例：パソコンのBIOS，BASIC）

今ではCPU，ボードやOS を作らずにブート・ローダ（BIOS）を開発，販売しているのは，Award 社を買収した米国Phoenix Technologies 社と米国AMI（Advanced Micro Instruments）社のパソコンのBIOS メーカ 2 社に絞られています．かつては米国Microsoft 社が開発したMS Basic のROM 版も，パソコン・アプリケーションのブート・ローダとして利用されていました．

▶独自開発

最初から独自に開発する場合，社内や外注先で以前作ったものの流用や改造，本来はブート・ローダではなかったファームウェアやOS を流用して改造することがあります．

▶明確なブート・ローダがない場合

ファームウェアのようにシステム構成が比較的簡単で，システムを起動するコードと主要な制御を行うコードが明確に分かれてない場合もあります．