ブートの仕組みとＬＩＬＯ

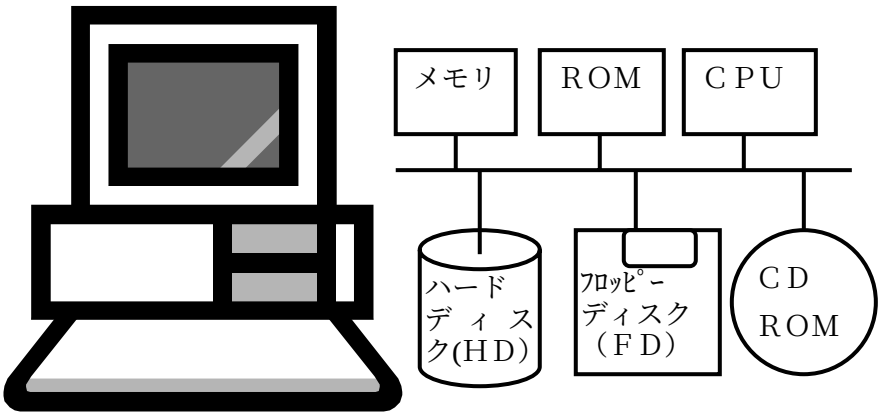
井戸ゼミ

■ はじめに

本文書は、PC の起動（ブート）の仕組みを理解するためのものです。記述の範囲は、ゼミにて行っている、ＬＩＬＯによるＬｉｎｕｘと Windows のデュアルブートの方法を理解出来るまでにとどめています。また、文系の皆さんの直感的な理解のため、若干正確さを欠いた記述もしています。

■ ＰＣの構成

本文書で考える右図のＰＣの構成は、ブートの仕組みを理解する上での最小限のものです。



- プログラムを実行するのは、ＣＰＵ。
- ＣＰＵが実行できるプログラムは、メモリ、もしくは、ＲＯＭ（予め書き込まれていて、変更出来ないメモリ）にある。
- ディスクドライブ（ＨＤ，ＦＤ、ＣＤ）上のプログラムは、ＣＰＵによりメモリ上に持ってきて、はじめて実行出来る。
- 本文書で、“ロード”と言っているのは、メモリにプログラムを持ってくることに相当する。

■ ディスクの構成

ディスク（ＨＤ、ＦＤ、ＣＤ）は、どれも５１２Ｂｙｔｅづつの固まりに分けて使用しています。この固まりを、“セクター（もしくはブロック）”と呼びます。

このうち、ハードディスクはパーティションという単位に仕切られています。Windows で見る、Ｃドライブ、Ｄドライブなどは、パーティションが分かれているわけです。パーティションは、領域という場合もあります。ＰＣの世界では、領域には基本領域と拡張領域（拡張領域の中には論理領域がある）の 2 種類がありますが、詳細については説明しません。

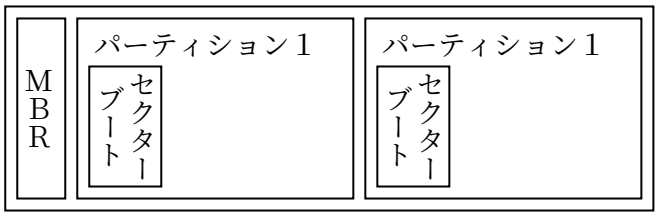
各ディスクには、ＰＣのブートにとって、特別な位置付けのセクターがあります。

＜ＨＤ＞

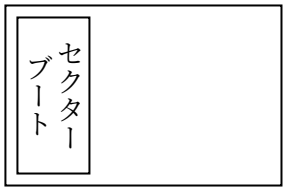
- ＭＢＲ（Master Boot Record）：ＨＤ全体の先頭セクターを指します。
- ブートセクター(Boot Sector)：パーティションの先頭セクターを指します。ＰＢＲ（Partition Boot Record）という場合もあります。

＜ＦＤ＞

・ブートセクター：ＦＤ全体の先頭セクターです。



＜ＨＤの特別なセクター＞



＜ＦＤの特別なセクター＞

ＨＤが２つあれば、それぞれにＭＢＲがあります。これらの特別なセクターに、決められたプログラムとデータを置くことで、ＰＣの起動を行えるようにしています。

■ Windows のブート

（１） 初期化プログラム

ＰＣの電源が投入されると、実行出来るプログラムは、ＲＯＭのプログラムだけです。ここには、ＢＩＯＳというソフトウェアが固定的に置かれており、それを実行するよう、決められています。最初に動くプログラムは、ＰＯＳＴ(Power On Self Test)と呼ばれ、基本的なデバイスの初期化を行います。

（２） 起動ドライブ検索

ＢＩＯＳは、決められた順序でディスクドライブを検索していきます。通常はＦＤが１番ですが、これが装着されれていないと、2 番目のＨＤに行きます。ここで、起動ドライブがＨＤに決まったとします。

（３） ＭＢＲロード

ＨＤの最初のセクターにある、ＭＢＲをロードします。ＭＢＲには、“ブートストラップローダ”と呼ぶプログラムが含まれており、これを起動します。

（４） パーティションテーブルの検査

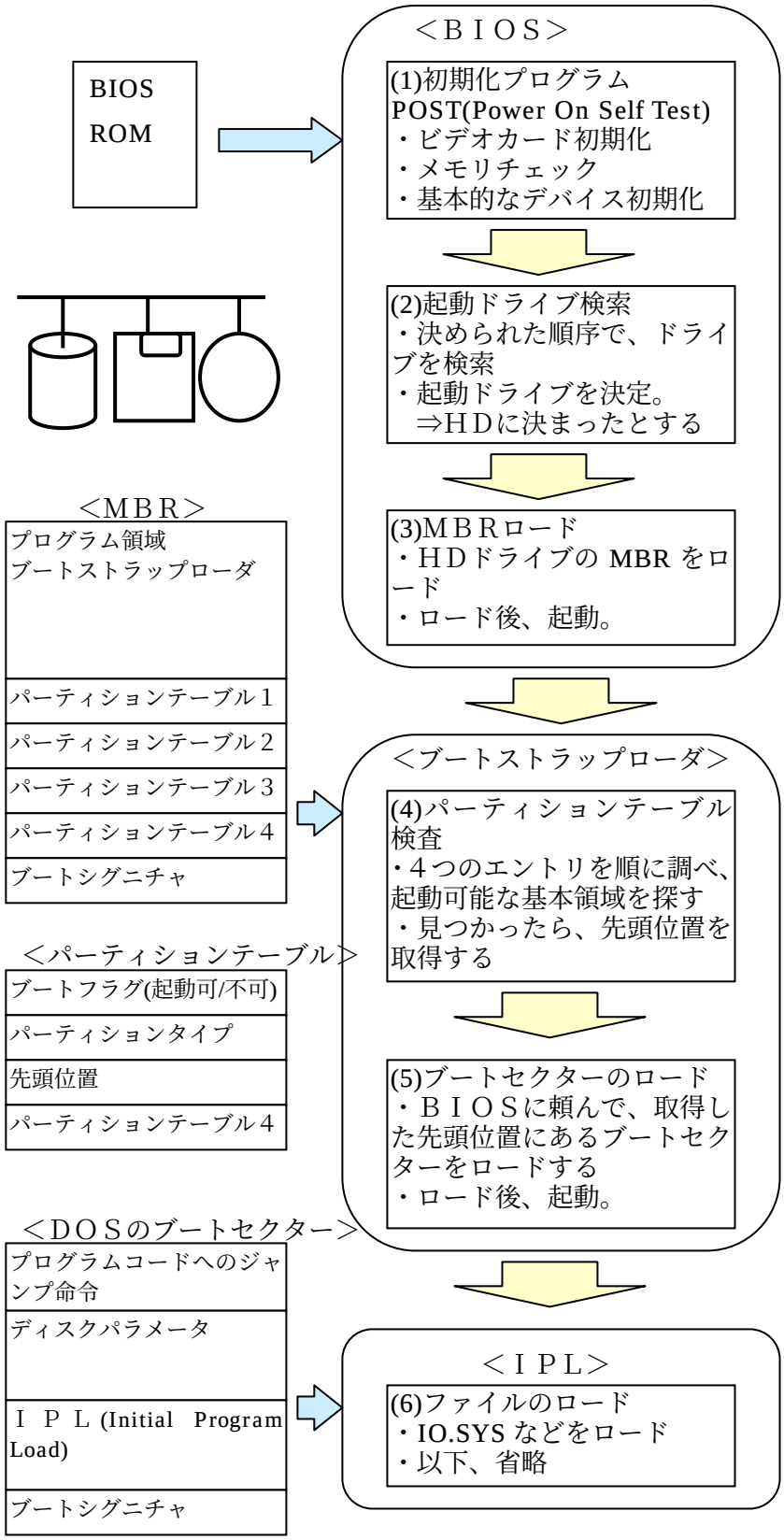
ブートストラップローダは、パーティションテーブルを順に調べ、起動可能なパーティション(すなわち、ＯＳとして動くプログラムが格納されたパーティション)を探します。パーティションテーブルには、パーティションの先頭位置が書かれていますから、これを取得しておきます。

（５） ブートセクター

見つかった起動可能なパーティションの先頭位置にあるＤＯＳのブートセクターをロードします。ロードしたブートセクターの先頭は、プログラムコードへのジャンプ命令ですが、これを起動します。

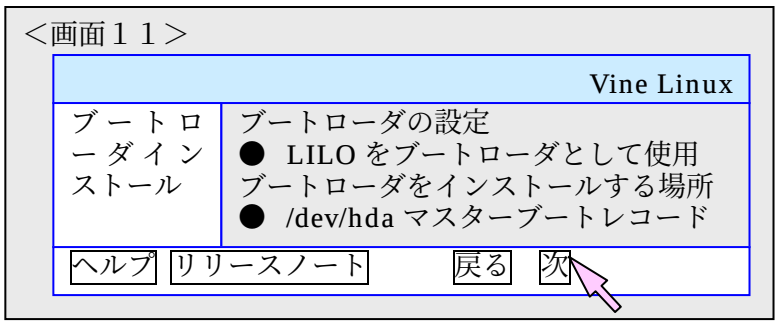
（６） ＩＰＬによるファイルのロード

処理はブートセクター中のＩＰＬ(Initial Program Load)に渡り、ＤＯＳの世界のファイルが次々とロードされ、Windows が起動されていきます。



■ ＬＩＬＯのしていること

＜画面１１＞



私たちがＰＣにＶｉｎｅＬｉｎｕｘをインストールしている時、右図のような設定を行いました。ＬＩＬＯをインストールする場所を、マスターブートレコードに設定しています。すなわち、我々のインストールでは、ＭＢＲを書き換えており、上記のブートストラップローダが動くかわりに、ＬＩＬＯが動くことになります。ＬＩＬＯはブートストラップローダと異なり、最初に見つけた起動可能な基本領域のブートセクターを決め打ちするのではなく、メニューを表示して利用者に起動するＯＳを選択させるようになっているという訳です。

ＬＩＬＯは、どうやって起動出来るＯＳを知るのでしょうか？普通にインストールした際は、その過程でインストーラが既に起動可能な Windows があることを調べて、メニューに出しています。どのようなＯＳがメニューに出されるかは、Linux 上のファイル、”/etc/lilo.config”に書かれています。右図でその様子を見てみると、Ｌｉｎｕｘと Windows のそれぞれの記述が見て取れます。

＜/etc/lilo.config の内容＞

```
prompt
timeout=50
default=linux
（中略）
lba32

image=/boot/vmlinuz-2.14.18-Ovl3
        label=linux
        read-only
        root=/dev/hda6
（中略）
other=/dev/hda1
        optinal
        label=win
```

image=～ ～root=～ ： Linux の記述

other=～ ～label=win ： Windows の記述

□ＬＩＬＯのメニューを変えたこと

Ｌｉｎｕｘのインストールを終えた何人かの方は、ＰＣの電源を入れたときに表示されるＬＩＬＯのメニューを少し変更しました。ひとつは、タイムアウトしてデフォルトのＯＳが立ち上がってくるまでの時間を長くしました。もうひとつは、デフォルトのＯＳをＬｉｎｕｘから Windows に変更しました。これをどうやってやったかと言うと、”/etc/lilo.config”の内容を次のように書き換えたのです。どのような変更であるのかは、理解しやすいでしょう。それでは、これでメニューが変わるかというと、これだけでは変わりません。ＬＩＬＯはＰＣの電源投入直後、ファイルなど読めない環境で動いている訳ですから、”/etc/lilo.config”を読んでその通りに動くことは出来ません。ハードディスクから直接読み出せるようにこのような情報が書かれていて、それをＬＩＬＯが見ている訳です。“/etc/lilo.conf”は言わば覚え書きみたいなもので、その内容は、“/sbin/lilo”コマンドを打つと、ＬＩＬＯに反映されるようになります。

＜/etc/lilo.config の変更＞

```
timeout=500
default=win
```

□ＬＩＬＯが起動できなかったＰＣへの対処

Ｌｉｎｕｘのインストールを終えた何人かの方は、ＰＣの電源を入れたときにＬＩＬＯが立ち上がらず、Windows が立ち上がってしまいました。これは、“/etc/lilo.config”中の、”lba32”とあるところが、“linear”となっていたためです。”lba32”と“linear”とは、セクターの番号の数え方を指定しているものです。“linear”は昔の数え方で、最大８ＧＢにあたるセクターしか数えられません。よって、Linux の/boot が先頭から８ＧＢよりも後ろにあると、Ｌｉｎｕｘを起動することが出来ないわけです。そのため、ＬＩＬＯは起動出来るＯＳは Windows のみと判断し、メニューを表示しませんでした。これの解決方法は、次の通りです。まず、インストール時に作成しておいたＬｉｎｕｘのブートＦＤから、

Ｌｉｎｕｘを立ち上げます。Ｌｉｎｕｘが使える環境になったら、“ /etc/lilo.config ”を編集して、“ linear ”を ” lba32 ”に書き換えます。この後、 “ /sbin/lilo ”コマンド（マップインストーラーと呼ぶことがあります）を打って、解決です。しかしながら、同じようにインストールして、どうして lilo.config の内容が“ linear ”となるＰＣがあるのかは、不明です（私には解りません）。

□Ｌｉｎｕｘをインストールしてから Windows９８/Ｍｅをインストールすると。。。

Windows９８やＭｅをインストールすると、問答無用でＭＢＲを Windows 本来の形に書き換えてしまいます。よって、ＭＢＲにインストールしたＬＩＬＯは働かなくなってしまいます。Windows９８をインストールしてから、Ｌｉｎｕｘをインストールする必要があるわけです。Windows２０００やＸＰは、このようなＭＢＲ書き換えを行いません。すなわち、Ｌｉｎｕｘをインストールしてから、ＸＰをインストールすることが可能です。

■ おわりに

ゼミのみなさんは、まだＵｎｉｘの勉強を始めたばかりですから、立ち上がらなかったＬｉｎｕｘを立ち上げてみせた時、手品のように感じたかも知れません。しかしながら、実際はこの文書に記したとおり、理屈から言って当たり前のことです。本文書に記した対処方法、自分で実行できますか？Ｕｎｉｘに慣れ親しみ、“ 理屈 ”を覚えていくことを期待しています。

なお、本文書は、次のサイトを参考に作成しました。

http://www.tkcity.net/~nobusan/boot_hdd.html

“ 理屈 ”は、Ｗｅｂで見ることが出来る訳です。

―以上―