# 簡易オペレーティングシステムの設計

064 番　本　武尊

## 1. 研究課題

windows や linux など既存のオペレーティングシステムの内部構成を理解し、理解をより深めるため、実際に現在最も普及している Intel 系 CPU 上で（FD から起動する）簡易 OS を設計・実装してみる.

## 2. 設計計画

簡易版なので、以下の機能に絞って設計していく.

① IPL(ブート処理部)の設計
電源が入ってから、CPU がディスクの先頭 512byte をメモリに読み込み、そこで実際にカーネル(OS の核)を読み込むことによって OS が起動する.

② VRAM 描画ルーチンの設計
VGA で描画する基本的なルーチンを設計

③ 割り込み処理部の設計
ここではキーボードからの割り込みとタイマー割り込みのみ設計する.

④ メモリ管理部の設計
フラットモデル.ページングは使用しない.ファーストフィット方式、連結リストを用いて領域割り当て及び開放を行う.

⑤ タスク管理部の設計
プロセスの生成、切り替えを実験してみる.

## 3. 設計内容

保護モードでのメモリ管理
[リアルモード]
[プロテクトモード]
[グローバルディスクリプタテーブル]
[セグメント・セレクタ]
[セグメント・ディスクリプタ]
メモリ管理サブシステムの設計
基本フラットモデル
開放は可変分割方式ファーストフィット方式
空き領域の管理方式にはリスト（チェイン）方式
割込みと例外の処理
IRQ と割込み
[割り込みディスクリプタテーブル]
タスク管理
タスクの実行
[タスク・ステート・セグメント（TSS）]
タスクの切り替え
カーネルの構築

## 4. プログラム(OS)の動作

Start_kernel();
_sys_init_vga();
_sys_init_io();
_sys_init_idt();
_sys_mm_init();
set_tss();
set_seg_desc();
LLDT 命令
process1();

```
void process1() {
   while (1) {
      _sys_printf("PROCESS1 ");
      switch_task(0x30);
   }
}
```

```
void process2() {
   while (1) {
      _sys_printf("process2 ");
      switch_task(0x20);
   }
}
```

実行結果

PROCESS1　process2　PROCESS1　process2
PROCESS1　process2　PROCESS1　process2
PROCESS1　process2　PROCESS1　process2
PROCESS1　process2　PROCESS1　process2
PROCESS1　process2　PROCESS1　process2
PROCESS1 …

## 5. まとめ

オペレーティングシステムの基本的な内部構成を設計・実装することによって、OS のブラックボックス的なものは結構解消された.ブート処理部はアセンブラで記述しなければならないためアセンブラの学習にもなった.またブート処理部も含め、割り込み処理、メモリ管理、タスク生成・切り替えの設計・実装などハードウェアに密接する部分の設計ではターゲットとする CPU 及び基盤 BIOS のアーキテクチャの理解が不可欠であることがわかった.

参考文献

[1] インテル・アーキテクチャ・ソフトウェア・デベロッパーズ・マニュアル 下巻
http://ime.nu/www.intel.co.jp/jp/developer/design/pentium/manuals/
[2] 「詳解　LINUX カーネル」　O'REILLY
[3]「オペレーティングシステム　第 2 版　設計と理論および MINIX による実装」　A・S・タネンバウム＋A・S・ウッドハル