システムプログラム概論
デバイスI/Oとファイルシステム

2007/4/24

門林　雄基
奈良先端科学技術大学院大学

Devices

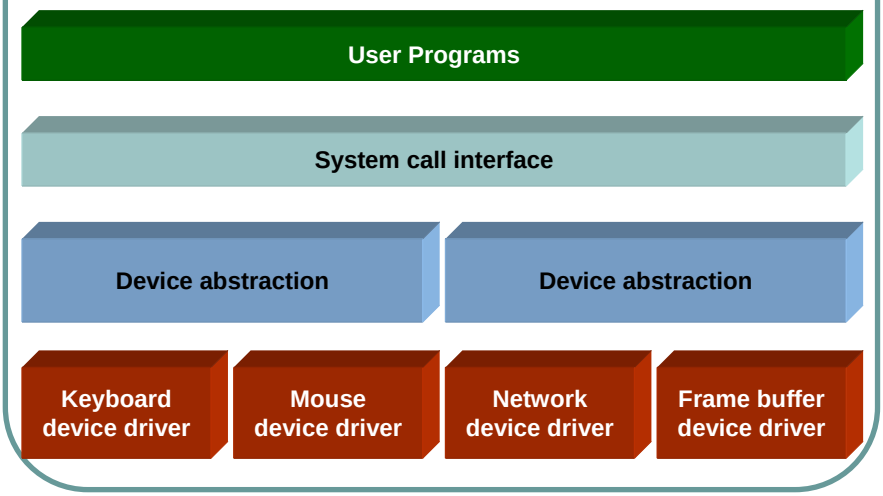
- keyboard, mouse, audio
- printer, scanner, frame buffer (video)
- network
- disk, dvd writer
- actuator, sensor, camera...

- **計算機と実世界とのつながり**

Device driver

- 数千種類のデバイス
- デバイスI/Oをモデル化し、アクセス方法を統一する必要がある
- カーネル以外からデバイスに直接アクセスすることを禁止
- カーネルでアクセス制御

Device drivers and the rest of the world



講義のポイント

- プロセッサとデバイスのやりとり
- デバイスI/O のゴール
- デバイスモデル - デバイスの抽象化
- ドライバモデル
- 高度な抽象化　ー　ファイルシステム

プロセッサとデバイスのやりとり

- I/O port, memory-mapped I/O
  - MOS p. 273
- DMA
  - MOS p. 277, 278
  - DMA channel
  - Word-at-a-time mode / block mode
- interrupt
  - MOS p. 279
- polling

デバイスI/O のゴール

- device independence
- uniform naming
- uniform API to user programs
- error handling
- blocking / non-blocking
- buffering

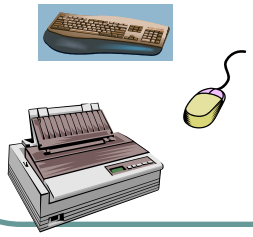
Simple, universal and adaptable

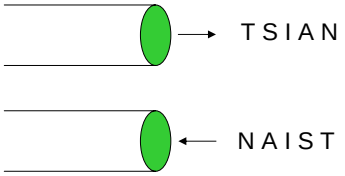
デバイスモデル

- device independence
- **数千種類のデバイス**
- **デバイスから独立であるためには一様なインターフェースを定義する必要がある**
- MOS p.294

Unix device model (1)

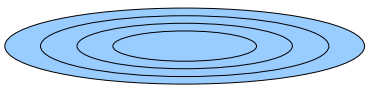
- character device
  - **ランダムアクセスが不可能なもの**
    devices that cannot be accessed from arbitrary point
    - keyboard, serial, mouse, etc.
    - open(), read(), write(), close()

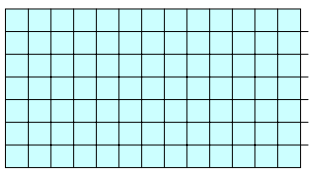




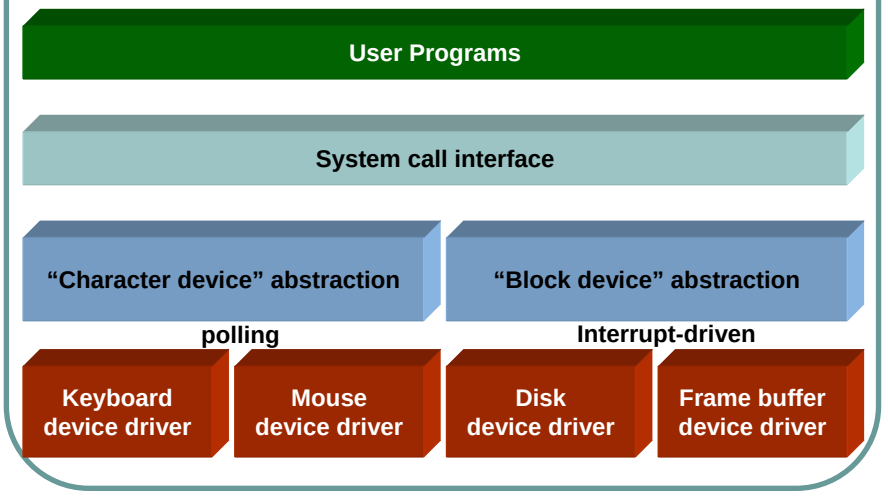
Unix device model (2)

- block device
  - **ランダムアクセスが可能なもの**
    devices that can be accessed from arbitrary point
    - frame buffer, disk, etc.
    - open(), read(), write(), lseek(), close()

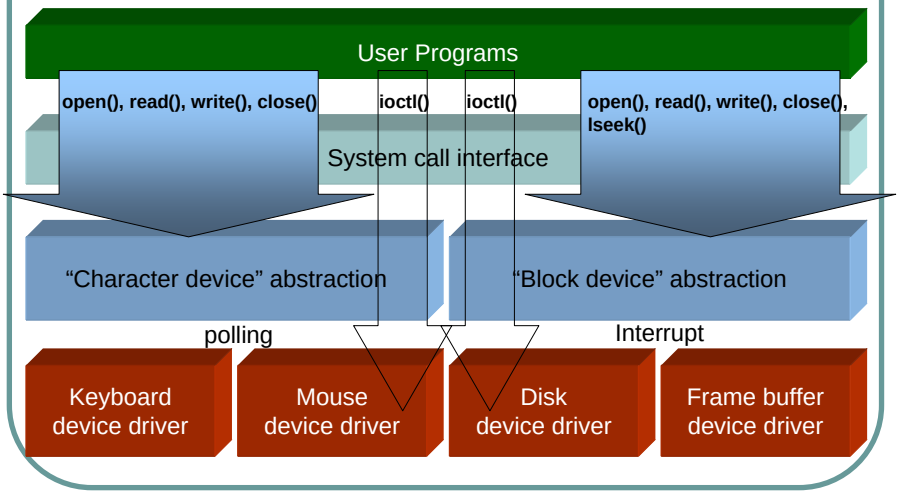




Unix device driver architecture

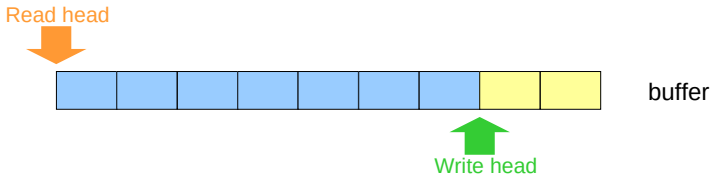


Uniform API to user-level programs



Buffering

- **デバイス速度とプロセス実行速度の差を吸収**
  - printer, network, etc.
- **バッファサイズを越えてプロセスが出力を吐き出したら？**
  - **プロセスをブロック**
- **プロセスがデバイスより遅かったら？**
  - **デバイスへの読み書きを休止**



Blocking / non-blocking

- Blocking read
  - read() **が終わるまでプロセスをブロック**
  - **プログラムは単純に**
  - **プロセス内の処理の並列度は上げられない**
- Non-blocking read
  - read() **を発行し続行**
  - **プログラムは複雑に**
  - **プロセス内の処理の並列化が可能**
- open, read, write, close...

Blocking / non-blocking（例）

- Blocking

```
foo()
{
  fd = open(“/dev/dsp”, ..);
  d = read(fd, buf, 5);

  if (d > 0) {
    // この時点でread完了
  } else {
    // エラー
  }
}
```

- Non-blocking

```
fooA()
{
  fd= open("/dev/dsp", ...);
  fcntl(fd, F_SETFL,
            O_NONBLOCK);
  d = read(fd, buf, 5);

  if (d > 0) {
    // この時点でread完了
  } else {
    // EAGAIN
    //   (data not available)
    // otherwise error
  }
}
```

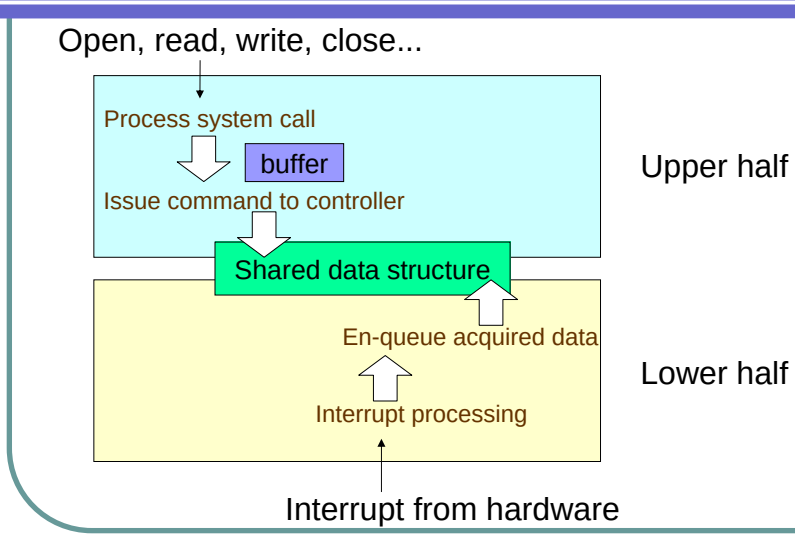

Layering in I/O software subsystem

- MOS p. 288
- p. 290
- p. 299

Unix driver model

- upper half, lower half
- upper half - **システムコール処理、コマンド発行処理、バッファ処理**
- lower half - **割り込み処理**
- **プロセスpで** read() → **ディスクコントローラにコマンド発行 → p 待ち状態**
- **(ディスクアクセス)** → (DMA**転送**) → **割り込み発生 → p 実行状態**

Unix driver model



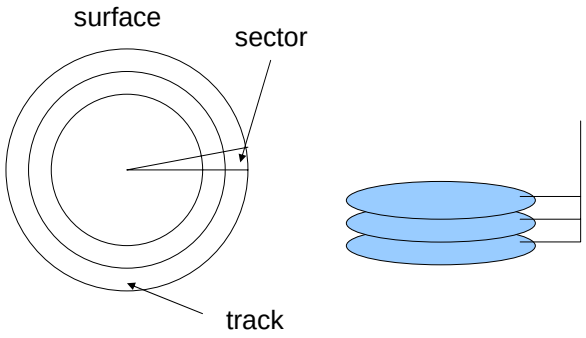
ここまでのまとめ

- **プロセッサとデバイスのやりとり**
  - I/O port, memory-mapped I/O
  - DMA
  - interrupt
- **デバイスI/O のゴール**
  - “Simple, universal, and adaptable”
  - device model, uniform API, buffering, error handling
- **デバイスモデル - デバイスの抽象化**
  - Character device, block device
- **ドライバモデル**
  - Upper half, lower half

二次記憶デバイスの抽象化と管理: ファイルシステム

二次記憶デバイス ― ディスク

- Surface
- Track
- Sector
- Cylinder



ディスク･スケジューリング
Disk scheduling

- FCFS
  - **到着順、fair だがシーク時間が長くなる可能性**
- SSTF (shortest seek time first)
  - **現在のヘッド位置から最も近いブロックへのアクセス要求を処理**
- SCAN (Elevator algorithm)
  - **進行方向で、最も近いブロックへのアクセス要求を処理**
- MOS p. 319〜

ディスク･スケジューリング
Disk scheduling

- Q. SSTF **では飢餓状態** (starvation) **になりうる。なぜか？**
- Q. SCAN **では飢餓状態にならない。なぜか？**

二次記憶
Secondary storage

二次記憶
- **固定長ブロック単位での読み書き**
- **セクタ番号でアクセス**
- **アクセス制御なし**
- **ディスククラッシュ時のデータ損失**

ファイルシステム
- **バイト単位での読み書き**
- **名前を指定してアクセス**
- **アクセス制御あり**
- **ディスククラッシュに強い**

ファイルシステムの機能
Functions of file system

- ディスク･スケジューリング
- ディスクブロック管理
  disk block management
- 名前管理
  namespace management
- アクセス制御
  access control
- 耐故障性
  failure resilience

} 今回は割愛

ディスクブロック管理
Disk block management

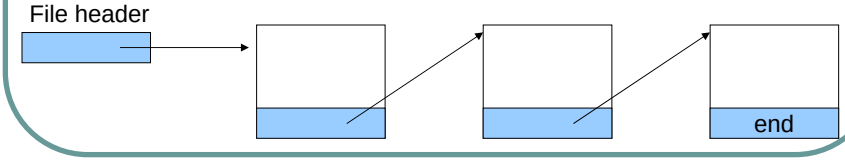
- 空きブロックの管理にはビットマップを用いる
  - 1ブロックあたり1ビット
- リストを用いる方法もあるが...
  - MOS p. 413

ディスクブロック管理のゴール
Disk block management goals

- minimize seeks, rotational delay
- minimize fragmentation
- fast sequential access
- fast random access

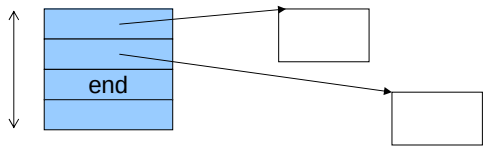
Linked files (Alto)

- ブロックが次のブロックを指す
- + ファイルサイズの延長が容易
- + 空きブロック管理が容易
- - ランダムアクセスが遅い
- - 信頼性が低い
  - ブロック破損 => ファイル末尾まで破損



Indexed files (VMS)

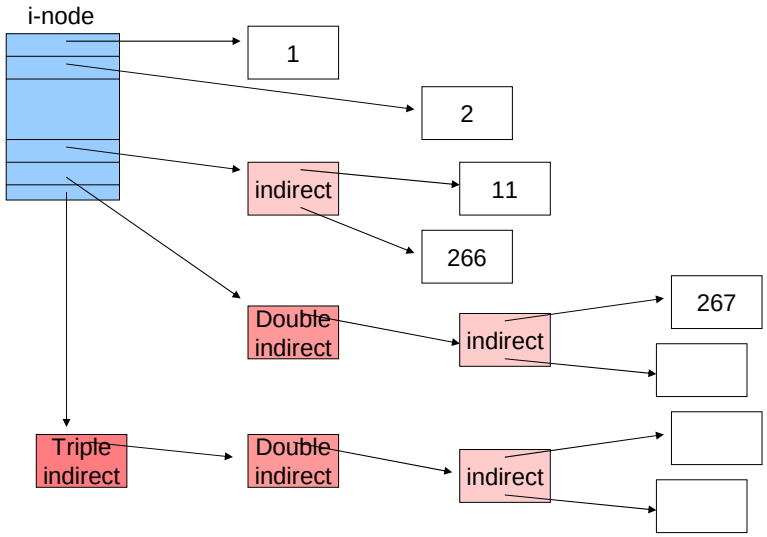
- ユーザは最大ファイルサイズを宣言する。
- システムはそれに相当するブロック数、ポインタが入るようにファイルヘッダを割り当てる。
- + ファイルサイズの延長が容易 (最大までなら)
- + ランダムアクセスが速い
- - 最大ファイルサイズを越える延長が面倒



Multilevel indexed files (4BSD)

- 目標ー小さいファイル、大きなファイル双方を効率よく扱う
- ファイルヘッダ(i-node)で 13個のポインタをもつ。
- ポインタのうち、最初の10個はデータブロックを直接指す。
  - (ファイルが10ブロック以下なら、残りのポインタは NULL)
- 11番目のポインタは indirect block へのポインタ
  - indirect block: データブロックへのポインタを含むブロック
  - 256 ブロックまで
- 267番目のブロックを割り当てるときは？
- double indirect block -- 12番目のポインタ
  - indirect block へのポインタを含むブロック
- さらに大きなファイルには 13番目のポインタを使う。
  - triple indirect block

Multilevel indexed files



Multilevel indexed files

- + ファイルサイズの延長が容易
- + ランダムアクセスが速い
- + 小さなファイルへのアクセスが特に高速
- - 大きなファイルでは、indirect block アクセスに時間を費す
- - シークが多い

名前管理 (Unix を例として)
Naming

- "/usr/bin/perl" → ディスクブロック番号系列
- システム内で、ファイルは (デバイス, i-node) で識別される
- i-node: Unix におけるファイル管理のための情報
  - デバイス内で一意な番号 (i-node 番号) により識別
  - i-node にブロック番号系列が記録される
- i-node は i-node ブロックに記録される。
- i-node ブロック数: ファイルシステム作成時に決定

名前管理
Naming

- ディレクトリ
  - (ファイル名, i-node) の系列を含むファイル
  - OS で特別扱い -- 通常のファイルのような書き込みは禁止
  - MOS p. 406

Unix における名前解決
Name resolution in Unix

- "/usr/bin/perl" の名前解決:
  - "/" -> i-node 番号 2 (固定)
  - i-node 2 が指すディスクブロック読み込み
    - ディレクトリ /
  - "usr" をディレクトリ / にて検索 -> i-node 番号 100
  - i-node 100 が指すディスクブロック読み込み
    - ディレクトリ /usr
  - "bin" をディレクトリ /usr にて検索 -> i-node 番号 500
  - ...
  - "perl" をディレクトリ /usr/bin にて検索 -> i-node 番号 9000

デバイスの識別 – ファイルシステム上での

- デバイスの識別
  - major device number, minor device number
- デバイスへのアクセス
  - デバイスファイル
  - “/dev/ad0”, “/dev/mouse” 等
  - ファイルシステムのアクセス制御を利用

アクセス制御
Access control

- ファイルヘッダにアクセス制御情報を保存
  - i-node (Unix)
- OSがアクセス制御情報にもとづいて読み込み、書き込み、実行などを許可/禁止
- OS固有のアクセス制御モデル

- 例)
  - Unix: (owner, group, rwxrwxrwx)
    - 読み込み、書き込み、実行
    - オーナ、グループ、その他の三階層に分け、アクセス制御
  - VMS, NT 等: (user/group, action...)
    - 読み込み、書き込み、実行、ファイル生成, ファイル消去、...
    - 複数のグループを指定可能

システム資源の名前付けとアクセス制御
Naming resources

- ファイルだけでなく、デバイスもアクセス制御したい。
  - マウス、キーボードはデスクトップ利用者のみ
  - ディスクのバックアップは管理者のみ
  - プリンタは特定のシステムプログラムのみ
  - ...

- Unix ではデバイスもファイル、ディレクトリと同様のアクセス制御を行う
  - "/dev/kbd0", "/dev/da0"...

ファイルシステムーまとめ

- ディスク･スケジューリング
- ディスクブロック管理と名前管理
  - Multi-level indexed files
  - i-node
- アクセス制御