入出力デバイスの種類

- 補助記憶（２次記憶）装置：HDD, DVD-ROM, USBメモリなど
- ネットワーク機器：有線／無線LANインターフェース、Bluetoothなど
- 表示装置：ディスプレイ、音声出力装置、（熱や匂い、振動も）
- 操作のための装置：キーボード、マウス、タッチパッド
- プリント・スキャン装置：プリンタ、スキャナ、カメラ
- 時計、タイマー
- センサー：加速度センサ、温度／湿度センサ、光センサ、GPS

データの管理

- システム内でのデータの管理の方法によって大きく分けられる
  - ブロック型デバイス
  - キャラクタ型（文字型）デバイス
  - その他レジスタ型，別空間型
- ブロック型はデータを固まりで処理、キャラクタ型はバイト単位で処理
- ブロック型は一定量のやりとりが見込まれる記憶デバイスなどが多い
  - デバイス上で大量の情報が管理されているようなものが対象
- キャラクタ型はデータ量が一定でなく、やりとりの頻度も不確定なものが対象
  - デバイス上でデータが蓄えられないもの→情報は即時処理される

デバイスコントローラ

- デバイスコントローラ：PC本体とのインターフェースとデバイスの直接的な制御を担う
- PC本体からの命令を受ける→命令を読みデバイス上で実施させる
  →得られた結果を返す
- 本体とのハードウェア的な接続→PCIバスなどの共有通信路で接続
- デバイスコントローラとデバイス間
  →規格で定められた信号によって通信
- PCへ結果を返す場合のデータサイズがタイプを分ける（ブロック型、キャラクタ型など）→処理方法が異なる

デバイスの制御

- コンピュータ本体とデバイスとのやりとり
- 一部のメモリ空間がデバイス内の情報と結びついている
- デバイスへの命令→所定のアドレスへ書き出し
- デバイスから情報収集→所定のメモリから読み込み
- デバイスに対する命令や返答は機器に依存
  - メーカが違えば同じ種類のデバイスでも異なる

メモリの共有方法

- デバイスコントローラとの情報共有にはPCが管理するアドレス空間へのマッピングを用いる
  - アドレスを指定すれば相互に連絡可能
- デバイス専用のアドレス空間（I/O空間）を用いる方法
  →I/Oポート
  - 読み書きに専用の命令（I/O命令）、値の判定に手間、通常のメモリとの競合がない
- 通常メモリのアドレス空間の一部を用いる方法
  →メモリマップドI/O
  - 通常のメモリアクセスを利用した命令が可能、キャッシュの制御が必要

ダイレクトメモリアクセス

- メモリの読み書きは通常プロセッサが一つ一つ行う→プロセッサはその作業中は他のことができない
- I/Oに割り当てられたメモリはデバイスの応答速度に依存する（かなり遅い）→プロセッサは応答を待つ間は無駄な時間
- まとめて転送してくれる特別な代理に実施させる仕組み→DMA（ダイレクトメモリアクセス）
- 専用のコントローラ→DMAコントローラ

割り込み（interrupt）

- プロセッサに対する作業の割り込みを伝える仕組み
  - エラーが出た時やデバイスからの要求で発生する
  - プログラムから意図的に発生させることもできる
- 割り込みが発生すると、プロセッサは処理中の作業を止め、割り込み専用の処理を実施しなくてはならない→割り込みハンドラ
- プロセッサによっては番号付きの割り込みハンドラを複数持つことができる→状況によって異なる割り込み処理が可能
- 割り込みハンドラの処理が終わると元の作業に復帰する

ベクタ割り込み

- 割り込みハンドラを一つしか持てないプロセッサもある→ポーリング割り込み
- ポーリングによって定期的なデバイスの状態把握

デバイスへの依存を無くす

- 入出力デバイスはデバイス毎に制御命令が違う
  →制御用ソフトウェアは個別に必要
- 制御関数の仕様を好き勝手に作ると、その上で動くアプリケーションもデバイス毎に対応が必要
- できればデバイスの種類が同じなら同じ呼び出し方がしたい→デバイス独立性

I/Oソフトウェア

- プログラム的には同期的な入出力の方が書きやすい場合が多い
  →例）writeの後の処理はwriteし終えた前提で処理が進む
- 実際には書き込みをDMAに任せて非同期化して、
  書き込み終わるまで他のことをプロセッサは行いたい
  →この処理を意識的に書かなくて良い仕組み
- アプリケーションから共通した手続きで、
  デバイスへのアクセスがプログラムの動作を妨げず、
  個々のデバイスに対して的確な命令を実施させる

デバイスドライバ

- 上位のプログラムが望む処理を個々のデバイス向けの命令に変換して実施
- デバイス固有の手法が必要→一般的にはデバイスメーカが作成し提供
- OSに組み込まれて使用される
  →OSの要望（仕様）に沿ったインターフェイス（関数）として用意
- メーカはOS毎にデバイスドライバを用意することに
  - バージョンが違っても、ドライバへの要求仕様が違う場合がある
  - Windows ではVxD (3.x, 9x系), WDM (2000, XP系), WDF (Vista以降)などの仕様がある
- ドライバは元々カーネルの一部→デバイスが多様化したことで、モジュール化して必要なものだけ読み込むように

上位のソフトウェア

デバイス独立ソフトウェア

- デバイスドライバの上位にあって、プログラムから見えるデバイスを抽象化するための機能
  - バッファリング：ソフトウェアの間に小さな記憶領域（バッファ）を置いて、プログラムとデバイスのデータのやりとりを仲介する→お互いのデータの読み書きを非同期化
  - エラー処理：対処可能な処理をドライバ内で処理→円滑なデータのやりとりを実現
  - 読み書き単位の抽象化：プログラム–デバイス間でやりとりするデータをそれぞれに適した大きさに調節して伝送→デバイスの持つ制約を意識しない操作を実現

ユーザ空間ソフトウェア

- ほとんどのI/Oソフトウェアは**OS内にある**
  →一部はユーザ空間で動作するライブラリとして提供
- 例）C言語ならprintfとかscanf→ユーザのプログラムが利用できて、画面出力、キー入力を受け付ける
- ユーザ向けのライブラリ以外のカテゴリ→スプーリング
- 常時稼働している特別なプロセス（デーモン）によって特定のデバイスを制御とデータプールの管理を行う→スプーリングデーモン
- ユーザプロセスはデバイス用に処理の依頼をして、プール用のフォルダに伝送するデータを置くと、デーモンが順次転送処理を行う→例えばプリンタデーモン
- 一つのデバイスに対する複数のユーザプログラムからの要求を整然と処理できる