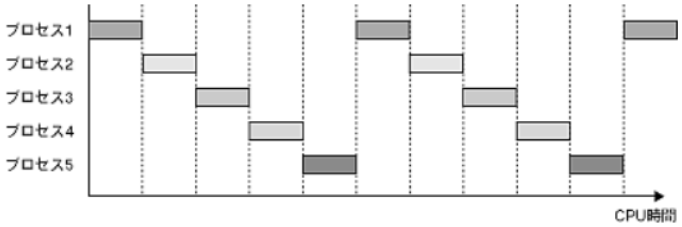
J3 Network Exercise

#02 04/21 (Thu) 10:35-12:05

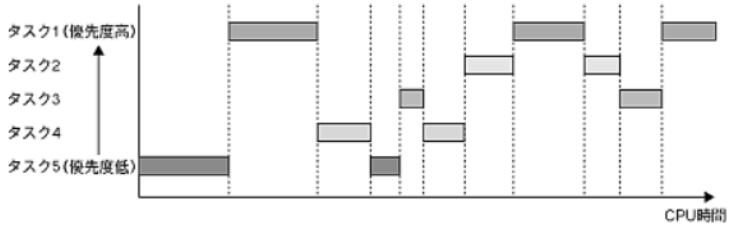
Yasuyuki SAITO
National Institute of Technology, Kisarazu College

基本事項の説明

- プロセス
  - 実行中のプログラムのことを指す。
  - マルチタスクOS は、複数のプロセスがCPUを使う時間を分け合って、同時並行的に動いている。
    - タイムシェアリング処理：ある時間で区切ってタイムスライスを形成し、あるタイムスライスの間にあるプロセスがCPU を用いる。
    - プリエンティブ処理：プロセスに優先順位を設けてCPU を割り当てる。



（a）タイムシェアリング処理



（b）プリエンプティブ処理

図の出展 http://www.kumikomi.net/archives/2001/05/03postpc.php?page=7

基本事項の説明

- プロセス間通信（inter-process communication）
  - あるプロセスとあるプロセスの間の通信のことを指す。
  - UNIX shell での例
    - ls -l | grep ‘^d’ | lv
      ↑ ↑
      これは、パイプ記号（日本語キーボードの場合、Shift + ¥ で入力）
    - パイプを用いた場合は、親子関係にあるプロセス同士で通信できる。
    - プロセスの親子関係については、今後の授業で説明。今は、「あるプロセス（親プロセス）から別のプロセス（子プロセス）を生成する」という理解を。
- ソケットを用いたプロセス間通信
  - 同じホストで動いているプロセス同士で通信できる。
  - ネットワーク上の他のホストで動いているプロセスと通信できる。

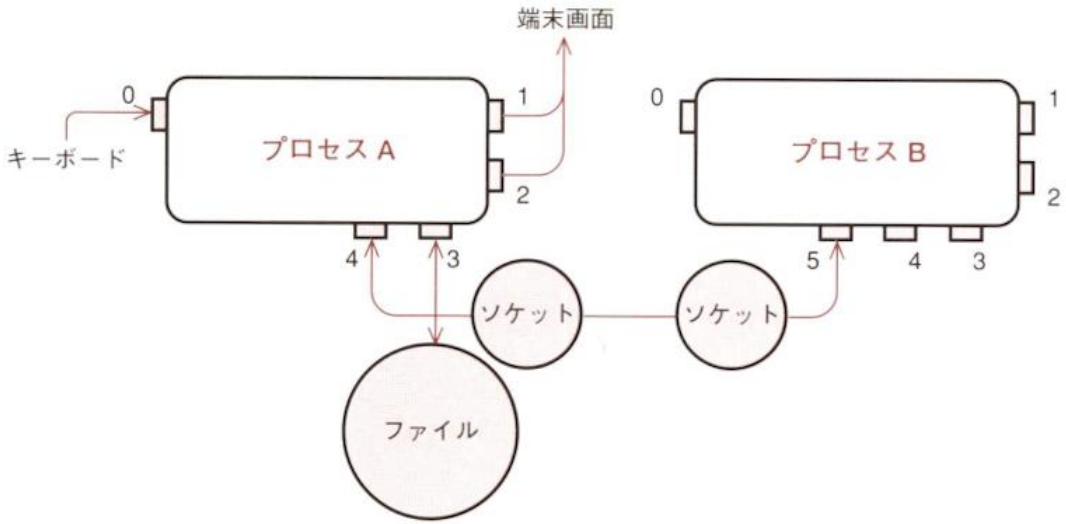
基本事項の説明

- ファイル
  - UNIX の世界においては、「プロセスの外界すべてをファイルシステムに位置づける」という発想。
    - キーボード、マウス、ディスプレイ、プリンタなど全てを「ファイル」として扱う。
    - HDD 内のデータの塊を「ファイル」として扱うのは想像できるだろうが、全ての機器を「ファイル」として扱うというのは、最初は理解しにくいかもしれない。まずは、「まぁ、そんなもんだ」と思えば十分。
- ファイル記述子（file descriptor）
  - あるファイルに与える「背番号」のようなもの。
    - 0 標準入力（キーボード）
    - 1 標準出力（ディスプレイ）
    - 2 標準エラー出力（出力先はディスプレイだが、エラーメッセージの表示用）

図の出展　雪田 修一「UNIX ネットワークプログラミング入門」技術評論社, p.29

プロセス間通信におけるソケットとファイル記述子

- これは、コンピュータ内部での話。
  - 通信路の両端にはソケットが割り当てられる。
  - プロセス側からはファイル記述子を用いてアクセスする。



- 上の図の例では、プロセスA は、プロセスB との通信において、プロセスA のファイル記述子 4 を使う。
- プロセスB のファイル記述子 5 を用いて、プロセスA とアクセスする。
- プロセスA のファイル記述子 3 を用いてファイルへアクセスする。

図の出展　雪田 修一「UNIX ネットワークプログラミング入門」技術評論社, p.30

ホスト間でのプロセス間通信

- ネットワークが関係している話。
  - 相手のホストを特定するには、IP address が必要。
  - 相手が特定できたら、さっそく通信したい。しかし、このままだと、ホストA, B 間のプロセス間通信は、同時に1組のプロセス同士でしかできないし、ホストA はホストB と通信している間に、ホストC, D, E, … とは通信できない（もの悲しい…）。
    - いわんや、ホストA に1組の糸電話だけが用意されているというイメージ。
  - そこで、ポート番号を用いる。
    - 複数のポートを用意しておいて、他のホストとの通信は、各ポートを使う。
    - いわんや、ホストA に複数の糸電話を用意するというイメージ。
    - ポート番号は符号なし16ビット2進数で表現される（0～65,535）。

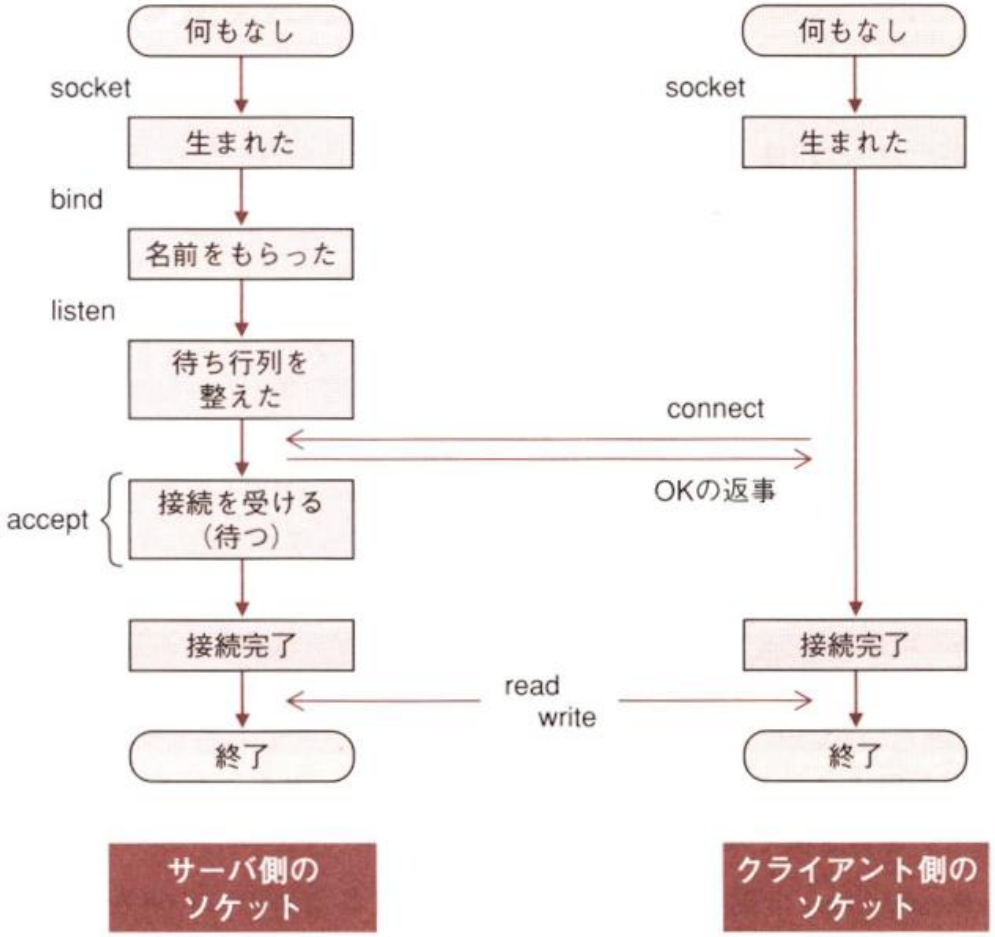
Wikipeida「TCPやUDPにおけるポート番号の一覧」より。
　0番のポートはエニーポート（any port）と呼ばれ、アプリケーションに対して、動的に別番号の空きポートを割り当てるために用意された特殊なポート番号である。別番号のポートの再割り当てを行わずに0番のポートとして使用することは禁止されているため、利用上では注意が必要である。 → 通常は、1 以上のポート番号を使う。

TCP とUDP

- TCP
  - 簡単にいうと、送信したデータが相手に届いたことを確認しつつ通信する方式。
  - 欠落しては困るような重要なデータを扱うときに用いる。
- UDP
  - 簡単にいうと、相手がきちんとデータを受け取ったことを確認しない方式。
  - 処理速度を優先して、データに欠落があっても構わない場面で用いる。
    - たとえば、動画ストリーミング。多少コマ落ちしても、人が動画の内容を理解する上では問題ない。

TCP クライアントとサーバの処理手順の流れ図

- ソケットを用いたクライアントとサーバの処理は、以下のように図示される。



図の出展　雪田 修一「UNIX ネットワークプログラミング入門」技術評論社, p.49

TCP クライアントの処理手順

- 通常、以下のような手順を踏む（教科書 p.32）。
  - socket() を実行してTCP ソケットを作成する。
  - connect() を実行してサーバへの接続を確立する。
  - send() と recv() を実行して通信を行う。
    - または write() と read() を使う。
  - close() を実行して接続をクローズする。

TCP サーバの処理手順

- 通常、以下のような手順を踏む（教科書 p.38）。
  - socket() を実行してTCP ソケットを作成する。
  - bind() を実行してソケットにポート番号を割り当てる。
  - listen() を実行し、割り当てたポートへ接続を作成できることをシステムに伝える。
  - 以下の手順を繰り返し実行する。
    - クライアントから接続要求を受けるたびに、accept() を呼び出してソケットを新しく取得する。
    - send() と recv() または write() と read() を実行しながら、作成したソケットを介してクライアントとやり取りをする。
  - close() を実行してクライアントとの接続をクローズする。

今回のミッション

- サンプル・プログラムのコンパイル、実行、課題提出
  - 細かい指示は、jes のファイルを参照。
    /home/class/j3/network/expr/src/01_TCPecho/00_readme.txt

- 注意事項（超重要）
  - emacs は授業が終わるまで、終了しないこと。
    - 万一、コマンド操作ミスや Makefile の記述ミスなどでファイルが消えた場合、emacs のバッファにファイルが残っているので、バッファからファイルを呼び出し、何か編集して（たとえば、文字を入力して消す。このときは、undo しては意味がない）、C-x C-s で保存すれば、ファイルを復活できる。
    - emacs を終了していてファイルが消えた場合は、間違いなく good-bye forever なので、ゼロからやり直すしかない。
    - これはエディタを使う上で常識。
    - コマンドプロンプトに戻るには、C-x C-z でsuspend する。
    - suspend したemacs を呼び出すには、fg コマンドを用いる。

今回のミッション

- 注意事項(超重要)
  - emacs を複数 起動しないこと。
    - ファイルバッファの操作をすれば、ファイルを切り替えられる。
    - 複数のファイルを同時に見たい? emacs 上で画面分割せよ。他のファイルからのコピペも単一の emacs 上ならば簡単。
  - 終了する前に、全ての端末で jobs を実行すること。
    - jobs を実行して すぐにコマンドプロンプトが返ってきた場合は、プログラムをバックグラウンドで実行 あるいは suspend していない。exit を実行して安全。
    - 何か表示された場合は fg でプログラムを再開し、そのプログラムを終了させる。念のため、もう一度 job を実行して確認してから、exit で端末を終了。
    - 全ての端末を exit コマンドで安全かつ確実に終了させることも、コンピュータを使う上で常識。

今回のミッション

- 注意事項(超重要)
  - Makefile の書き方について
    - 環境変数の定義や、コンパイル ルールの塊の境界には空行を置こう。
    - = の前後には空白やタブを置こう。
    - 環境変数BIN に与えるターゲット名や依存関係のオブジェクト ファイル名は、バックスラッシュを用いて複数行を用いて記述しよう。
    - バックスラッシュの前には空白を置こう。
    - ターゲット名やオブジェクト ファイル名の位置を合わせよう。
    - : の後には空白やタブを置こう。