テストノート

2023年11月17日 金曜日 9:26

オペレーティングシステム

- ハードウェアの抽象化
  統一的で単純化された利用方法を提供.

- 資源管理

	char	short	int	long	longlong	ポインタ
16 bit OS	8	16	16	32		16
32 bit OS			32		64	32 (多くの処理系)
lcp 32			16		64	32
lp 32						
64 bit OS						
LLP64			32	32	64	64 bit Windows
LP 64			32	64	64	64 Unix OS
ILP 64			64	64	64	64

システムコール

write()
└ 指定したサイズのデータをファイルに書き込む

utsname 構造体

```
struct utsname {
    char sysname[ ];
    char nodename[ ];
    char release[ ];
    char version[ ];
    char machine[ ];
}
```

プロセスの終了

exit();
正常終了 0 or EXIT_SUCCESS
異常終了 1 or EXIT_FAILURE

エラーメッセージ
perror — エラーメッセージ表示

マスターブート と ブートブロック
↑
ディスク全体の最初のセクタ

オペレーティングシステムを起動するプログラムが格納されている

- ~~[illegible]~~
- Signal()
- ~~[illegible]~~
- ~~[illegible]~~ → 待機, 割込の受け入れ準備
- ~~[illegible]~~ → プロセス終了
- ~~[illegible]~~

socket() : socket() の返したファイルディスクリプタ
bind() : 最下位ポート番号

```
int count = 0;
pthread_mutex_t i;
void func (void *arg) {
pthread_mutex_lock (&i);
    count++;
pthread_mutex_unlock(&i);
}
func ( i );
```

① 割込 タイミング
[競合]
複数のスレッドを処理するとき,
その処理中に別スレッドの処理が割り込むことがあるため。

スーパーブロック

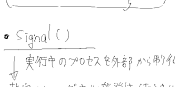
- ファイルシステムの大きさ
- 空きブロックに関する情報
  - 空きブロックの数, 次ブロックのアドレス
- inodeリストに関する情報
  - inodeリストの大きさ, inodeリストの数
  - 空きinodeリスト
- ファイルの種類
  - 一般/ディレクトリ/デバイス
- ハードリンク
- ファイルの所有権
  - ユーザ/グループ-IDとID
- ファイルアクセス権
  - 所有者/グループ/他者
- 時間
  - 最終アクセス時間
- ファイルサイズ
- データブロックへのポインタ

ファイル
*様々なデータを格納するための論理的な単位
*記憶装置の一つ.

I/O

アドレス変換バッファ
テーブルの内容を高速にアクセス可能なアドレス変換バッファ(TLB)にチャッシュすることで主記憶装置へのアクセス回数を削減.
↓
一般にCPU内部に存在

プロセッサ(CPU)のコンテキスト情報
- プログラムカウンタ
- レジスタ
- フラグ
- スタックポインタ
- ベースレジスタ インデックスレジスタ・スケールファクタ



- Signal()
↓
実行中のプロセスを外部から制御する手段
指定したシグナルが発生したときに自動的にハンドラを起動するように登録.
- ヘッダファイル #include <signal.h>
  signal(シグナル番号, ハンドラ)
- 引数
  - シグナル番号
    シグナルの種類(整数)の指定.
  - シグナルハンドラ
    ※シグナルが発生したときに動作する関数を指定.
    ※指定できる関数は引数・戻り値なし
    ※デフォルトの動作をこの指定した動作に置きかえる

- メッセージドリブン
  システムやソフトウェアの設計やアーキテクチャにあたって処理のトリガーとしてメッセージを使用する手法.
  → プログラムの異なる部分プロセス間や外部からのメッセージによって動作
  これ、それにfifoとして処理が進む.

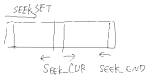
※多重サーバ/並行サーバ
反復サーバ
順次クライアントの要求を処理

並行サーバ
クライアントからの要求毎に子プロセスを生成し, その子プロセスに処理させる

SEEK_SET
先頭からオフセットの場所まで移動

SEEK_CUR
現在位置からオフセット分移動

SEEK_END
ファイル末尾からオフセット分移動




```
int main(){
  int i,n;
  char buffer[512];
  while (n=read(STDIN_FILENO, buffer, sizeof(buffer))>0){
      for (i=0;i<n;i++){
          if (buffer[i]>='a'){
              buffer[i]-='A';
          }
      }
      write(STDOUT_FILENO, buf, n);
      }
  }
```

```
int main(){
  pid_t cp;
  pid_t gp;
  int p;
  cp = fork();
  if (cp==0){
  gp = fork();
  if (gp==0){
    printf("gp:%d, cp:%d", getppid(), getpid());
    }
    else { wait(&p);
    printf("cp:%d, gp:%d, p:%d", getppid(), getpid(), p);
    }
  }
  else { wait(&p);
    printf("pid%d, cp:%d", getpid(), cp);
    }
```