pipe

テーマ　：　pipe

大事（そう）なところだけ抜粋します

※　意見には個人差があります

※　誤りが含まれている可能性もあります

要点的な

大事なこと

- 各プロセスは異なるメモリ空間で動作しているので
  メモリの共有ができない
  - 今は mmap() でできるけど当時は本当にできなかった？
- プロセス間のデータの受け渡しのために
  pipe を使う
- pipe はファイルシステム（ブロックデバイス）経由で
  データをやり取りする

似た関数

	for file	for pipe
書き込み	writei()	writep()
読み込み	readi()	readp()

似た関数

- close()　：　close() システムコール
- closef()　：　file[] エントリの参照 cnt を -1
- closei()　：　inode[] エントリの参照 cnt を -1

似た関数

- ialloc()　　:　inode を割当て
- falloc()　　:　file[] エントリを割当て
- ufalloc()　　:　u.u_ofile[] エントリを割当て

対の関数

- リリース
  - plock() : inode[] の ILOCK をクリア
  - prele() : inode[] の ILOCK をクリア

11.2. pipe()

struct

- user : P.46
- inode : P.274
- file : P.316

u.u_ofile[]

- そのプロセスがオープンしているファイル
  → file[] の当該エントリへのポインタ（P.315）
- int 型
- 要素数は NOFILE : 15

file[]

- オープンしているファイルの操作情報の構造体配列
- 要素数は NFILE : 100

inode[]

- メモリ上で管理している方の inode 構造体配列
- 要素数は NINODE : 100

iget()

- inode[] エントリを取得する
  - 引数 1 : デバイス番号
  - 引数 2 : inode 番号
- ロックされてたらスリープして待つ
- IMOUNT なら mount[] を取得して
  対応デバイスルートの inode[] エントリを返す
- 子関数
  - bread()
  - brelse()
  - iput()

iput()

- inode[] エントリのカウンタをデクリメントする
- 子関数
  - itrunc()
  - iupdat()
  - prele()

inode 領域の取得／解放

- ialloc()
  - getfs()
  - iget(), iput()
  - bread()
- ifree()
  - getfs()

ialloc()

- ブロックデバイス中の未使用 inode を割り当てる
- 呼び出す人たち
  - maknode()
  - pipe()
- maknode() を呼び出す人たち
  - mknod()
  - creat()

pipe()

- ialloc()　　：　inode[] エントリを獲得
- falloc() x2　：　file[] エントリ read 用 ／ write 用
- iput()　　：　file[] エントリ取れなかったら

- ip　：　rw 共用の inode 構造体ポインタ
- rf　：　read 用の file 構造体ポインタ
- wf　：　write 用の file 構造体ポインタ
- r　：　u.u_ofile[] エントリの fd を退避

writep()

- plock() / prele() : inode[] エントリのロック＆解除
- writei()　：　ファイル書き込み
- wakeup() :　受信側プロセスを起床

- rp　：　file[] の当該エントリ（ip 代入時のみ使用）
- ip　：　対応する inode[] エントリ

readp()

- plock() / prele() : inode[] エントリのロック＆解除
- readi()　　:　ファイル読み込み
- wakeup() :　送信側プロセスを起床

- rp　　:　file[] の当該エントリ（最後にも使用）
- ip　　:　対応する inode[] エントリ

plock()

- i_flag に ILOCK をセットする
- sleep()　：　空いてなかったら
- rp　：　inode[] エントリ

prele()

- i_flag の ILOCK をクリアする
- wakeup()　：　待ってる人がいたら
- rp　：　inode[] エントリ

closef()

- ファイルをクローズする

- 今日事前に勉強する予定なので
  もう知ってるよね？

pipe の使い方

- 1. pipe()
- 2. fork()
- 3. 親： 片方を close()
  子： 片方を close()

dup()

- getf()　　：　file[] エントリを取得
- ufalloc()　：　u.u_ofile[] の最初の空きエントリを獲得

- fp　：　ユーザープログラムから来た file[] エントリ

第11章　完

EOF

プロセス概要

- 基本的なルールは Linux と一緒
- 管理情報
  - proc 構造体
    - メモリ上の特定領域に常に存在
    - プロセス最大数は NPROC (= 50)
  - user 構造体
    - PPDA 領域内
    - グローバル変数 u でアクセス

1 プロセスにつき
それぞれ 1 つずつ
紐付けられる。

つまりこう？

APR

UISD (0177600) : PDR[0] – [7] for Text

(UDSD) (0177620) : PDR[0] – [7] for Data

UISA (0177640) : PAR[0] – [7] for Text

UDSA (0177660) : PAR[0] – [7] for Data