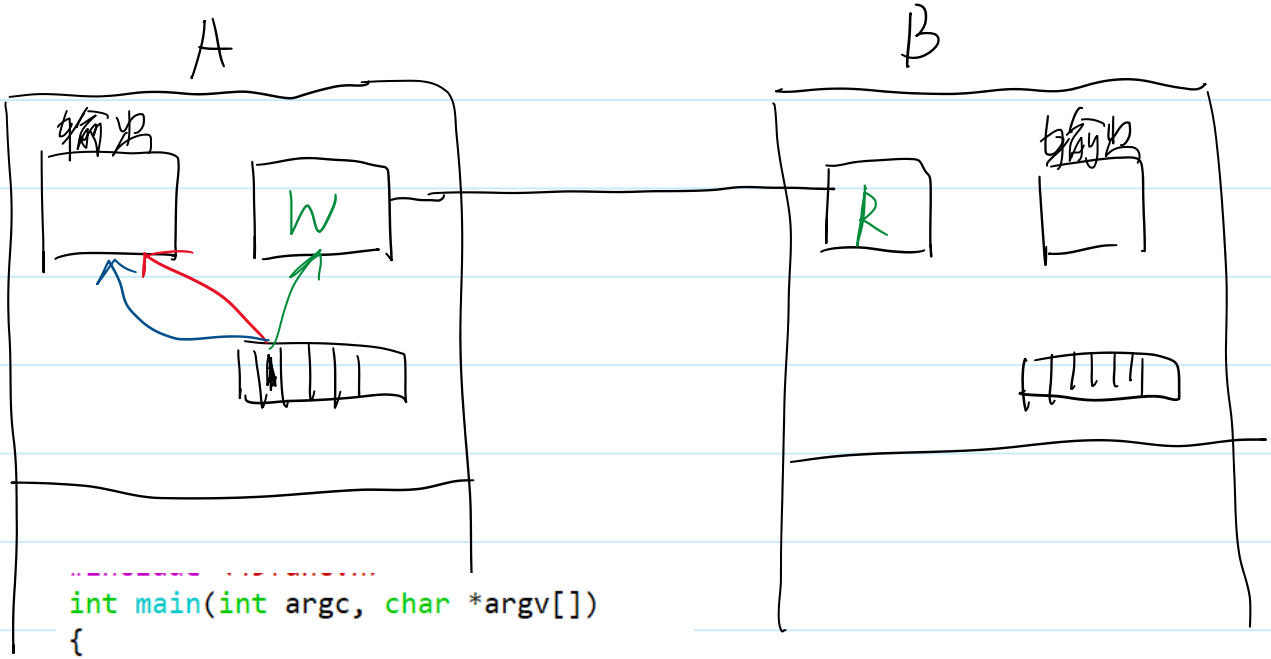
作业

2023年4月22日 9:30



1 printf

2 printf

3 printf

```c
int main(int argc, char *argv[])
{
    // ./0_homework_A 1.pipe
    ARGS_CHECK(argc,2);
    int fd = open(argv[1],O_WRONLY);
    ERROR_CHECK(fd,-1,"open");
    printf("1 helloworld!\n");

    int newfd = 10;
    dup2(STDOUT_FILENO,newfd);
    dup2(fd,STDOUT_FILENO);
    printf("2 helloworld!\n");

    dup2(newfd,STDOUT_FILENO);
    printf("3 helloworld!\n");
    close(fd);
    return 0;
}
```

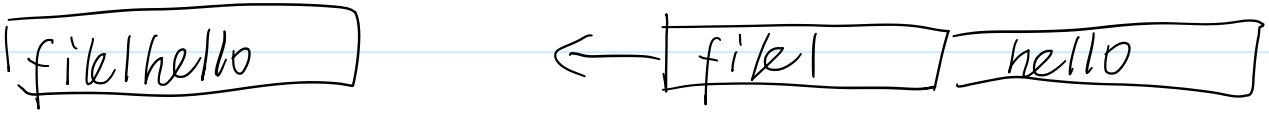
```c
int main(int argc, char *argv[])
{
    // ./0_homework_B 1.pipe
    ARGS_CHECK(argc,2);
    int fd = open(argv[1],O_RDONLY);
    ERROR_CHECK(fd,-1,"open");
    char buf[4096] = {0};
    read(fd,buf,sizeof(buf));
    printf("buf = %s\n",buf);
    close(fd);
    return 0;
}
```

数据无边界

2023年4月22日 10:03

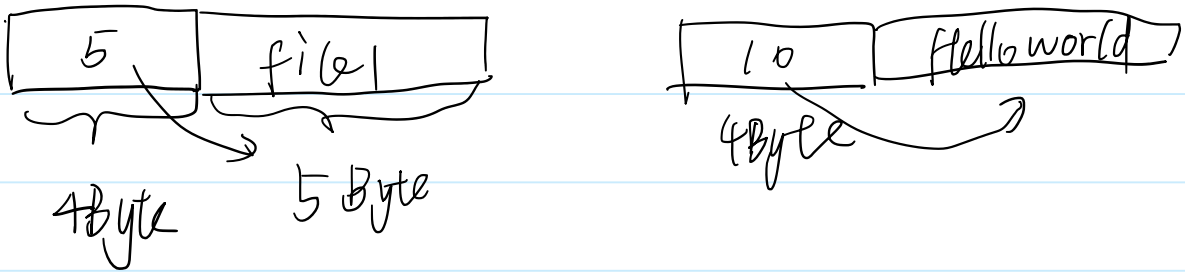
```c
#include <49func.h>
int main(int argc, char *argv[])
{
    // ./2_homework_transB 1.pipe
    ARGS_CHECK(argc,2);
    int fdr = open(argv[1],O_RDONLY);//读管道
    ERROR_CHECK(fdr,-1,"open fdr");
    mkdir("./storage",0777);
    char buf[4096] = {0};
    // 收文件名
    read(fdr,buf,sizeof(buf));
    char filepath[8192] = {0};
    sprintf(filepath,"%s/%s","./storage",buf);
    // 新建文件
    int fdw = open(filepath,O_WRONLY|O_CREAT, 0666);
    ERROR_CHECK(fdw,-1,"open fdw");
    // 收文件的内容
    memset(buf,0,sizeof(buf));
    ssize_t sret = read(fdr,buf,sizeof(buf));
    printf("sret = %ld\n", sret);
    write(fdw,buf,sret);
    close(fdw);
    close(fdr);
    return 0;
}
```

```c
#include <49func.h>
int main(int argc, char *argv[])
{
    // ./2_homework_transA file1 1.pipe
    ARGS_CHECK(argc,3);
    int fdr = open(argv[1],O_RDONLY);
    ERROR_CHECK(fdr,-1,"open fdr"); //读磁盘文件
    int fdw = open(argv[2],O_WRONLY);
    ERROR_CHECK(fdw,-1,"open fdw"); //写管道
    // 发文件名
    write(fdw,argv[1],strlen(argv[1]));
    // 发文件的内容
    char buf[4096] = {0};
    ssize_t sret = read(fdr,buf,sizeof(buf));
    write(fdw,buf,sret);
    close(fdw);
    close(fdr);
    return 0;
}
```



用户自己设计边界

2023年4月22日 10:30




```c
#include <49func.h>
int main(int argc, char *argv[])
{
    // ./2_homework_transB 1.pipe
    ARGS_CHECK(argc,2);
    int fdr = open(argv[1],O_RDONLY);//读管道
    ERROR_CHECK(fdr,-1,"open fdr");
    mkdir("./storage",0777);
    char buf[4096] = {0};
    int length;
    // 收文件名
    read(fdr,&length,sizeof(int));
    read(fdr,buf,length);
    char filepath[8192] = {0};
    sprintf(filepath,"%s/%s","./storage",buf);
    // 新建文件
    int fdw = open(filepath,O_WRONLY|O_CREAT, 0666);
    ERROR_CHECK(fdw,-1,"open fdw");
    // 收文件的内容
    memset(buf,0,sizeof(buf));
    read(fdr,&length,sizeof(int));
    ssize_t sret = read(fdr,buf,length);
    printf("sret = %ld\n", sret);
    write(fdw,buf,sret);
    close(fdw);
    close(fdr);
```

```c
#include <49func.h>
int main(int argc, char *argv[])
{
    // ./2_homework_transA file1 1.pipe
    ARGS_CHECK(argc,3);
    int fdr = open(argv[1],O_RDONLY);
    ERROR_CHECK(fdr,-1,"open fdr"); //读磁盘文件
    int fdw = open(argv[2],O_WRONLY);
    ERROR_CHECK(fdw,-1,"open fdw"); //写管道
    // 发文件名
    int length = strlen(argv[1]);
    write(fdw,&length,sizeof(int));
    write(fdw,argv[1],length);
    // 发文件的内容
    char buf[4096] = {0};
    ssize_t sret = read(fdr,buf,sizeof(buf));
    length = sret;
    write(fdw,&length,sizeof(int));
    write(fdw,buf,sret);
    close(fdw);
    close(fdr);
    return 0;
}
```

select的原理

2023年4月22日 11:03

关心 fd_set

$$\frac{1024}{8 * sizeof(fdmask)} * sizeof(fdmask) = \frac{1024}{8} Byte$$

```
typedef struct
  {
    __fd_mask __fds_bits[1024 / (8 * (int) sizeof (__fd_mask))];
  } fd_set;
```

= 1024 bit

```
[liao@ubuntu ~]$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 31346
max locked memory       (kbytes, -l) 65536
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
```

bitmap 位图.

(监听集合)

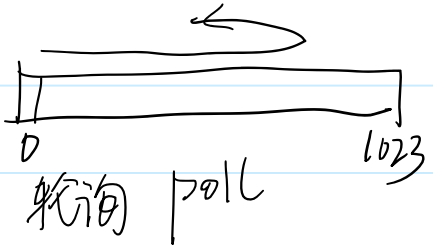
① 在用户态创建位图

② 把位图从用户态拷贝到内核态

③ OS 轮询 0~nfds-1

④ 得到就绪集合，放在之前监听集合的位置

⑤ FD_ISSET 看对应bit是否1.



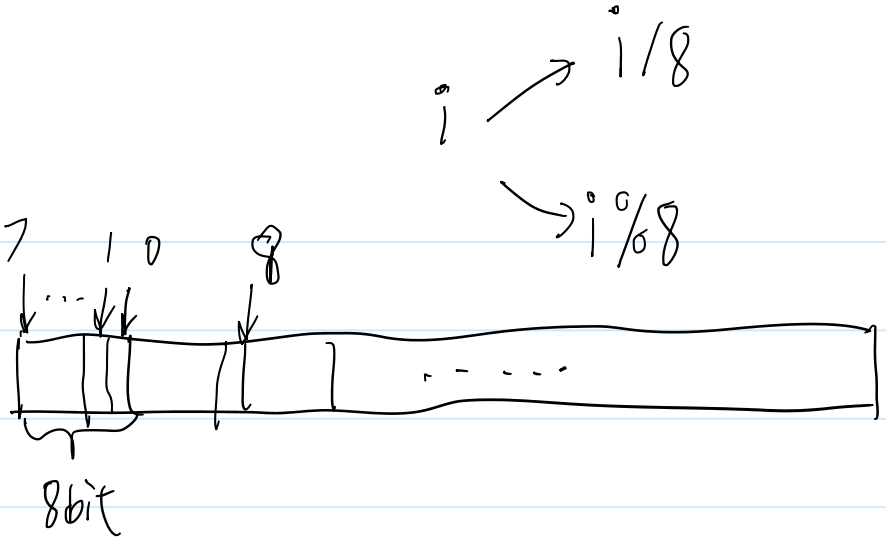
位图

2023年4月22日 11:11

1024个文件描述符 0,1

Char bitmap[1024/8]

int bitmap[1024/32]



select的缺陷

2023年4月22日 11:24

1. 文件数量上限1024，不好改

2. 存在大量内核态和用户态之间的拷贝

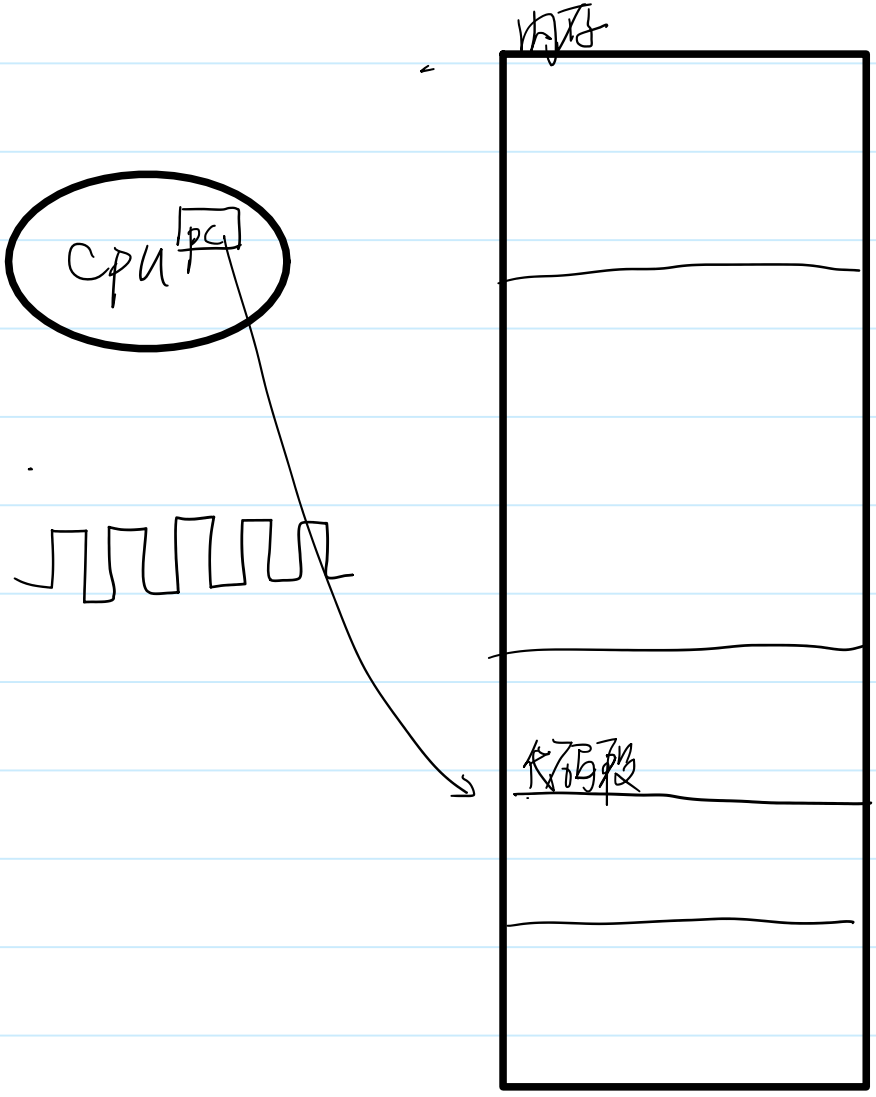
3. 将所集合和就绪集合耦合在一起

4. 海量连接，少量就绪。

进程

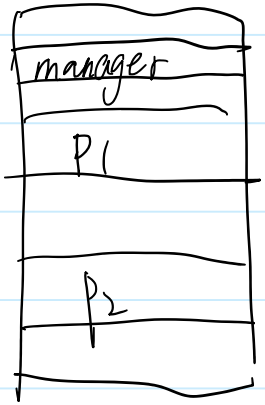
2023年4月22日 11:33

进程是一个正在运行的程序（动态）



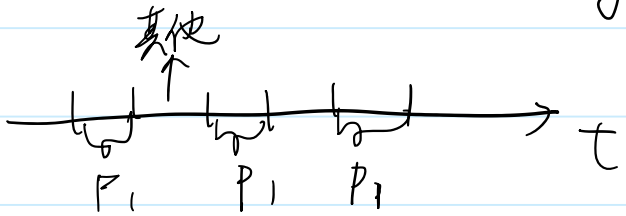
批处理系统（batch）

多道程序设计（multiprogramming）



驻留

分时系统（time-sharing）



多任务操作系统

2023年4月22日 11:51

进程 资源分配的基本单位

→ 内存

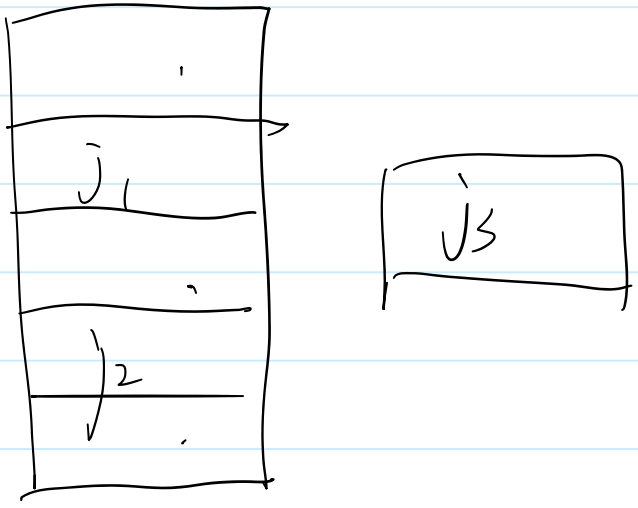
计算机解决不了问题，加一个中间层

多道程序设计的缺陷

内存资源：① 相对地址

② 没有隔离

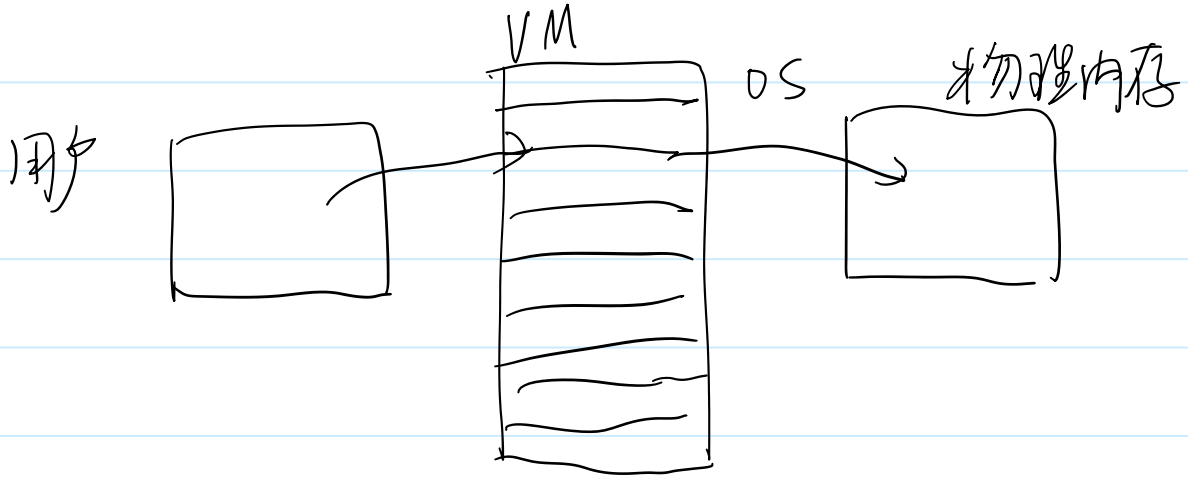
③ 利用率低



虚拟内存

2023年4月22日 11:57

virtual



每个进程一个独立的虚拟内存.(隔离)

地址 0 ~ MAX

分页机制

内存 → 固定大小,4096B

分配内存到以分页为一页.

只给正在使用的虚拟内存页 分配物理页

页 页框

如果页框数量不够

2023年4月22日 12:06