实验一 同步与异步write的效率比较

姓名：刘梦杰 学号：22920212204168 时间：2023.10.15

一、实验内容

计算 write 耗费的时间，来比较同步写和异步写的性能差异。显示的时间应当尽量接近write操作过程所花的时间，不要将从磁盘读文件的时间计入显示结果中。

二、实验原理及思路

- 根据输入命令按同步或异步方式 open 文件。
- 将打开的文件以 BUFFSIZE 的大小读入缓冲区，每次读的过程中执行 write 操作，循环执行到文件全部读完。
- 每次写之前调用时间函数记录一次时间，写之后再调用一次时间函数测量时间，两者之差就是 write 的时间。
- 令BUFFSIZE从256开始起算直至128K，打印出不同 BUFFSIZE 下所用的时间。
- 关键代码：
  - 异步或同步打开文件

```
if (argc == 2){ // async
    if ((fd = open(argv[1], O_RDWR|O_CREAT|O_TRUNC, FILE_MODE)) < 0){
        err_sys("Cannot open");
    }
}
else { //sync
    if ((fd = open(argv[1], O_RDWR|O_CREAT|O_TRUNC|O_SYNC, FILE_MODE)) <
0) {
        err_sys("Cannot open");
    }
}
```

  - 以不同 BUFFSIZE 写入文件，并计算时间

```
for (BUFFSIZE = 1024; BUFFSIZE ⩽ 524288; BUFFSIZE *= 2) {
    if (lseek(fd, 0, SEEK_SET) == -1){
        err_sys("lseek error");
    }
```

```
    LoopsCount = length / BUFFSIZE;
    ClockStart = times(&tmsStart);
    for (int i = 0; i < LoopsCount;i++){
        if (write(fd, buff + i * BUFFSIZE, BUFFSIZE) != BUFFSIZE){
            printf("%d\n", BUFFSIZE);
            err_sys("write error");
        }
    }
    // if remain
    int remain = length % BUFFSIZE;
    if (remain){
        if (write(fd, buff + LoopsCount * BUFFSIZE, remain) != remain){
            err_sys("write error");
        }
        LoopsCount++;
    }
    ClockEnd = times(&tmsEnd);
    UserTime = (double)(tmsEnd.tms_utime - tmsStart.tms_utime) / ticks;
    SysTime = (double)(tmsEnd.tms_stime - tmsStart.tms_stime) / ticks;
    ClockTime = (double)(ClockEnd - ClockStart) / ticks;
    printf("%8d\t %8.2f\t %8.2f\t %8.2f\t %8d\n", BUFFSIZE, UserTime,
SysTime, ClockTime, LoopsCount);
}
```

三、实验结果

需要的库文件放在 include目录下

源程序为 `timewrite.c`

可执行程序为 `timewrite`

1. 使用命令 `gcc timewrite.c -o timewrite` 编译生成可执行程序

```
[cs214168@mcore 1]$ gcc timewrite.c -o timewrite
[cs214168@mcore 1]$
```

2. 执行命令 `./timewrite <1.jpg 2.jpg` 异步write，输出文件 2.jpg 不用O_SYNC 打开

```
[cs214168@mcore 1]$ ./timewrite <1.jpg 2.jpg
BUFFSIZE         User CPU        System CPU      Clock time      Loops Count
     256             0.00             0.01            0.01             4327
     512             0.00             0.00            0.02             2164
    1024             0.00             0.00            0.01             1082
    2048             0.00             0.00            0.00              541
    4096             0.00             0.00            0.00              271
    8192             0.00             0.00            0.00              136
   16384             0.00             0.00            0.00               68
   32768             0.00             0.00            0.00               34
   65536             0.00             0.01            0.00               17
  131072             0.00             0.00            0.00                9
```

3. 执行命令 `./timewrite 1.jpg sync <2.jpg` 同步 write，输出文件1.jpg 用O_SYNC 打开

```
[cs214168@mcore 1]$ ./timewrite 1.jpg sync <2.jpg
BUFFSIZE         User CPU        System CPU      Clock time      Loops Count
     256             0.00              0.10          111.58             4327
     512             0.00              0.03            8.83             2164
    1024             0.00              0.02            4.39             1082
    2048             0.00              0.00            2.20              541
    4096             0.00              0.01            1.10              271
    8192             0.00              0.00            0.56              136
   16384             0.00              0.00            0.29               68
   32768             0.00              0.00            0.22               34
   65536             0.00              0.00            0.09               17
  131072             0.00              0.01            0.06                9
```

四、体会和建议

实验过程中，我们观察到异步 write 所用的时间要明显少于同步 write 所用时间，可以得出异步 write 的效率要明显高于同步 write 的结论。同时也学会了库函数 times 的相关使用方法。实验代码的比较容易理解，但是由于不够熟练，许多函数用法及参数不够了解，写代码时参考了许多资料。