Subject: Lecture on Operating System Lecture Exercise 10.4
From: IKENOYA Katsutoshi <j05002@ie.u-ryukyu.ac.jp>
Date: Sat, 27 Jan 2007 12:45:21 +0900
To: Shinji KONO <kono@ie.u-ryukyu.ac.jp>

学籍番号：055702B

問題10.4

mmap_test.c は、共有メモリを使用するmmapの例である。
man mmap を参考にしながら、このプログラムの動作を説明せよ。
プログラムを変更し、リアルタイムで、共有された情報が変化する
ことを確認せよ。

・ソース

ソースの場所：/home/y05/j05002/OS-Lecture10/Lec10-4/mmap_test.c

```
#include <stdio.h>
#include <sys/mman.h>
#include <sys/types.h>
#include <fcntl.h>

#define SHARE_SIZE (BUFSIZ*16)

main(int ac,char *av[])
{
int fd = -1 ;
caddr_t share,s;
int *p;
int i;

/*
マップしたページの変更を共有する
msync()が呼ばれると更新される
*/
int map = MAP_SHARED;

if(ac==1) {
//領域を新たに作成
map |= MAP_ANON;
} else if(ac!=2) {
fprintf(stderr,"Usage: %s [newfile]",av[0]);
exit(0);
} else {
if((fd=open(av[1],O_CREAT|O_RDWR,0666))==0) {
fprintf(stderr,"can't open %s¥n",av[1]);
}
}

/*
ファイル記述子fdで指定されたファイルのオフセット(0)から
SHARE_SIZEの範囲をメモリにマップする。
メモリ保護モードを読み書き可能に指定
*/
share = mmap(0L,SHARE_SIZE,PROT_READ|PROT_WRITE,
map,fd,(off_t)0);
if(share!=NULL) {
fprintf(stderr,"mapped %08x¥n",share);
}
*((int *)share) = 0;
```

```
if(fork()) {
/* parenet */
p = (int *)share;
*p=0;
sleep(1);

for(i=0;i<10000;i++) {
(*p)++;

//インクリメントする度に表示するときに使用
//fprintf(stderr,"parent %d¥n",*p);

//pをインクリメントする度に更新を伝える
//MS_ASYNC:更新を予定に組み込む
// msync(share,SHARE_SIZE,MS_ASYNC);
}

/*
上記のfor loopでmsyncを使用しない場合
ここで初めて同期をとる。
*/
msync(share,SHARE_SIZE,MS_ASYNC);
fprintf(stderr,"end parent %d ",*p);
wait(NULL);
fprintf(stderr,"%d¥n",*p);
} else {
/* child */
p = (int *)share;
fprintf(stderr,"start child %d¥n",*p);
while(*p<1000) {
// msync(share,SHARE_SIZE);
}
fprintf(stderr,"end child %d¥n",*p);
exit(0);
}
}
```

・実行結果(for loop中のmsyncを使用しない場合)

```
[j05002@Lec10-4]% ./mmap_test mmap_test.c
mapped 00006000
start child 0
end parent 10000 end child 10000
10000
[j05002@Lec10-4]%
```

for loop中のmsyncを使用しない場合、shareの同期が親プロセスがpを10000回インクリメントした後にしか行われないので、子プロセスは更新情報を途中で受け取ることができない。そのため、pの値が1000以上になっても子プロセスは終了せずに、親プロセスと同じ値(10000)で終了してしまっている。

・実行結果(for loop中のmsyncを使用した場合)

```
[j05002@Lec10-4]% ./mmap_test mmap_test.c
mapped 00006000
start child 0
end child 2929
end parent 10000 10000
[j05002@Lec10-4]%
```

親プロセスがpをインクリメントする度に同期がとられるので、親プロセスよりも先に子プロセスが終了していることが分かる。
ただし、本来ならば子プロセスはpが1000の時に終了するはずが実行結果では、2929となっている。これはpの計算速度が速く、pの値が1000以上と認識することに子プロセスが追いついてないからであると考えられる。

・実行結果(リアルタイム)

```
[j05002@Lec10-4]% ./mmap_test mmap_test.c
mapped 00006000
start child 0
parent 1
parent 2
parent 3
parent 4
parent 5
省略
parent 1148
parent 1149
end child 1149
parent 1150
parent 1151
省略
parent 9996
parent 9997
parent 9998
parent 9999
parent 10000
end parent 10000 10000
[j05002@Lec10-4]%
```