mytop实现思路

一． mytop实现中CPU使用百分比思路（简要）：

1. 计算进程和任务总数nr\_total，读取/proc/kinfo
2. 读取每个进程的信息，包括type，endpoint，pid，cycles\_hi,cycles\_lo，state。遍历/proc/pid/psinfo

如果type==task，则进程标记p\_flags |=istask

如果type==system，则进程标记p\_flags |=issystem

如果state！=state\_run,则标记p\_flags |=blocked

连续读CPUTIMENAMES次cycles\_hi,cycle\_lo（三个cputimenames），然后拼接成64位，放在p\_cpucycles[]数组中。

1. 计算每个进程proc的滴答，通过proc和当前进程prev\_proc做比较，如果endpoint相等，则在循环中分别计算

for(i = 0; i < CPUTIMENAMES; i++) {

if(!CPUTIME(timemode, i))

continue;

if(proc->p\_endpoint == prev\_proc->p\_endpoint) {

t = t + prev\_proc->p\_cpucycles[i] - proc->p\_cpucycles[i];

} else {

t = t + prev\_proc->p\_cpucycles[i];

}

}

1. 计算总的cpu使用百分比，遍历所有的进程和任务，判断类型，计算systemticks，userticks（由于kernelticks和idleticks为0不用计算）。

if(!(proc2[p].p\_flags & IS\_TASK)) {

if(proc2[p].p\_flags & IS\_SYSTEM)

systemticks = systemticks + tick\_procs[nprocs].ticks;

else

userticks = userticks + tick\_procs[nprocs].ticks;

}

二． mytop中可以用到的函数

观察在minix中输入top可看到内存和进程信息，都是通过该top.c实现的。本实验可以移植其中的函数来实现。一中的实现思路也可用以下函数来实现功能。

在main()函数中实现了while循环，所有的显示信息都是showtop()打印的。其中getinfo()函数读取/proc/kinfo得到总的进程和任务数nr\_total。

在showtop()函数中，打印top上显示的信息。本实验需要的是print\_memory()和print\_procs(),打印出内存和CPU使用情况。print\_memory()需读取/proc/meminfo文件信息，print\_memory()则要获取到每个进程和任务的信息，通过get\_procs()函数将所有需要的信息放在结构体数组proc[]中，每个元素都是一个进程结构体。

get\_procs()函数，首先记录当前进程，赋值给prev\_proc，然后通过parse\_dir()函数获取到/proc/下的所有进程pid,再通过parse\_file()函数获取每一个进程信息，即读取/proc/pid/psinfo文件。

在parse\_file()函数中读取的信息需要判断是否可用。比如version是否为1，如果不是该进程不需要记录。在判断slot时，需要用到SLOT\_NR(endpt)函数，不过该函数有些问题，判断出来的slot大于nr\_total,所以自己修改一下slot的赋值。在源码下有一句p = &proc[slot];所以了解到slot就是该进程结构体在数组中的位置，可以简单通过其他赋值，比如slot++，只要在数组中不会重复即可。该函数会给进程结构体变量赋值，看源码即可。也可按照源码一样全部变量都赋值，供后面使用。

再创建一个tp结构体，这个结构体包含了进程指针p和ticks，对应的就是某个进程和滴答。

在cputicks()函数中，计算每个进程的滴答。滴答并不是简单的结构体中的滴答，因为在写文件的时候需要更新。需要通过当前进程来和该进程一起计算，这里需要用到p\_cpucycles,在前面赋值的时候已经写进进程结构体了。

print\_procs()函数中就输出CPU使用时间，这是最后也是最重要的。这里创建了一个tp结构体的数组tick\_procs。对所有的进程和任务(即上面读出来的nr\_total)计算ticks，具体看源码。把kernelticks，userticks，systemticks相加就可以得到CPU的使用百分比。在计算idleticks时因为IDLE已经大于nr\_total,所以计算出的idleticks恒为0.