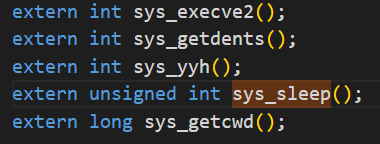
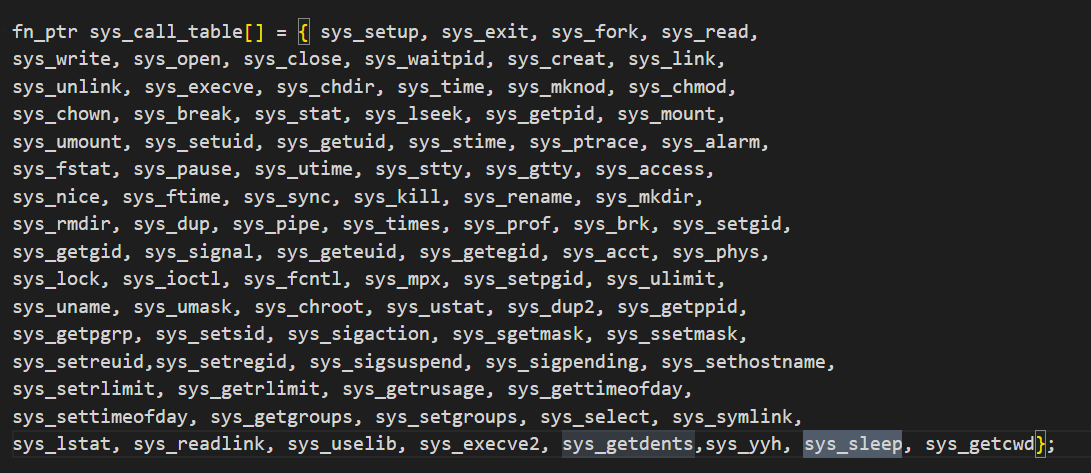
实现了四个系统调用函数，这里分别阐述一下实现思想。

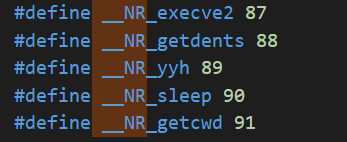
1. 首先仿照系统调用的函数格式进行修改
   1. 添加系统调用的接口程序



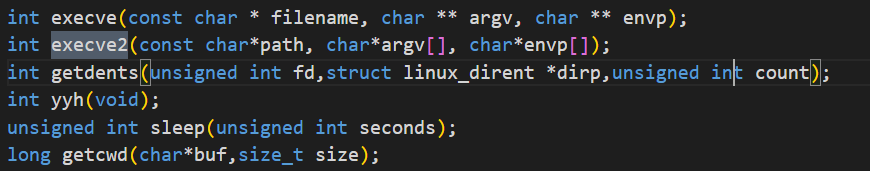
* 1. 在中断向量表中添加系统调用函数名



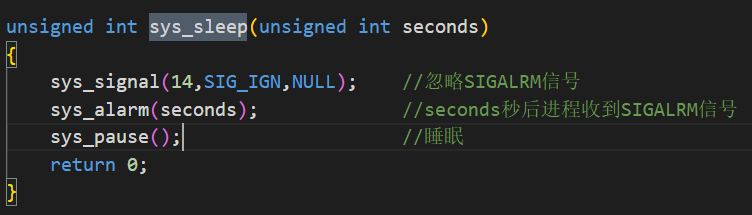
* 1. 添加系统调用号



* 1. 仿照，添加函数声明

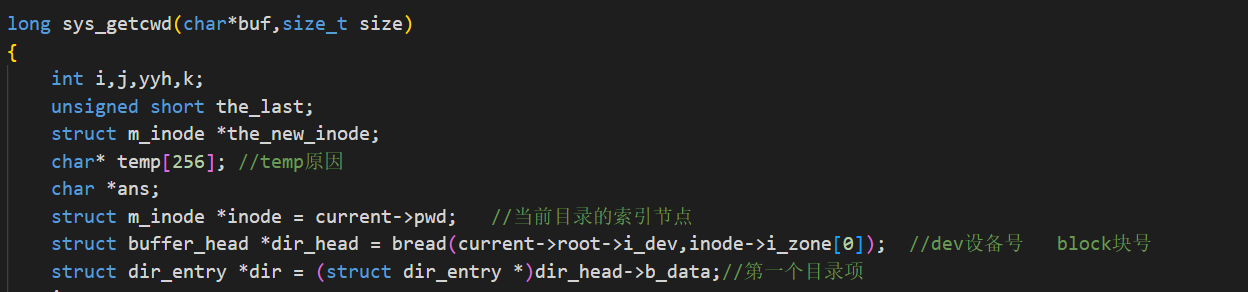


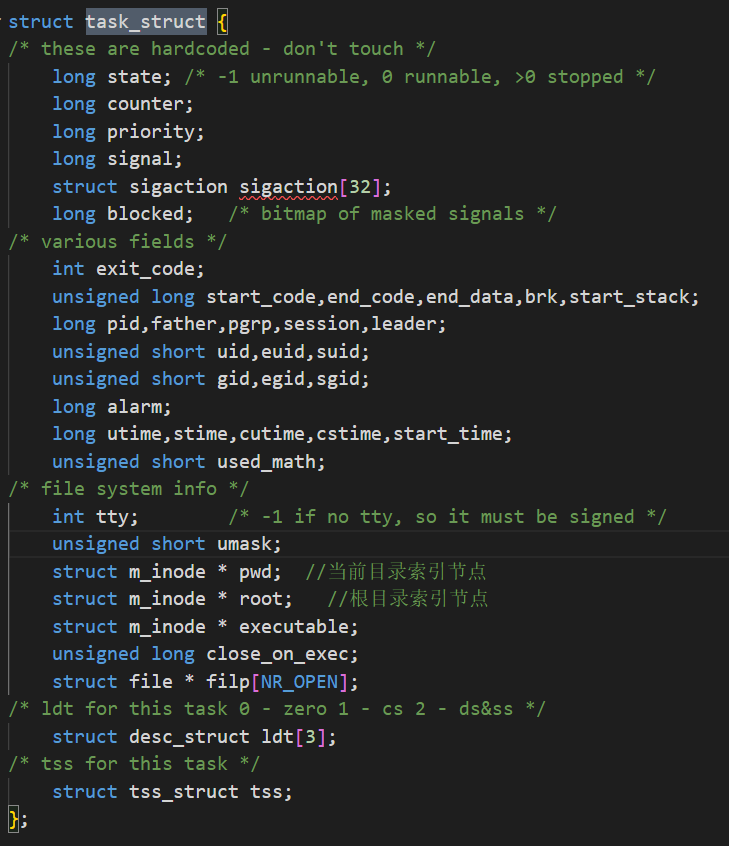
1. sys\_sleep函数的实现



使用定时器，首先要安装SIGALRM信号。如果不安装SIGALRM信号，则进程收到SIGALRM信号后，缺省的动作就是终止当前进程。因此开头安装SIGALRM信号，即接受到时忽略（SIGALRM信号对应的编号为14）。这样最终会正常返回0。

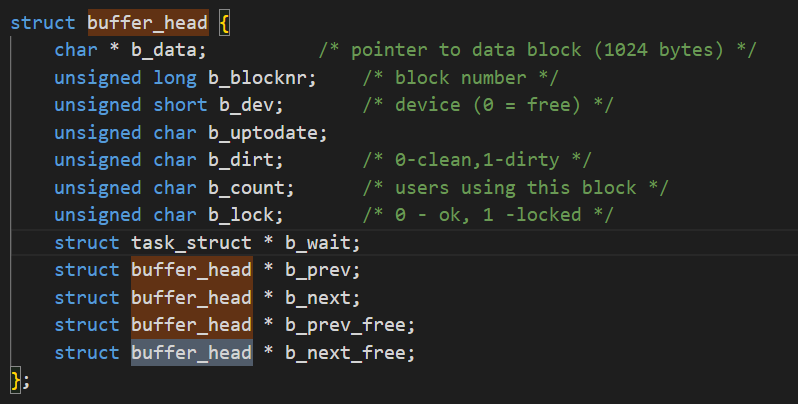
1. sys\_getcwd函数的实现
   1. 运行时一直出现一个错误，后来发现是char \*temp[1024]数组开的有点大，后来改为char \*temp[256]后运行正确了。

当前进程current是一个指向task\_struct的指针，观察task\_stuct的结构体成员



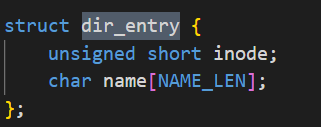
current->pwd是当前目录的索引节点，current->root是根目录的索引节点。

bread(dev, block)返回一个指向缓冲头的指针。（buffer\_head）

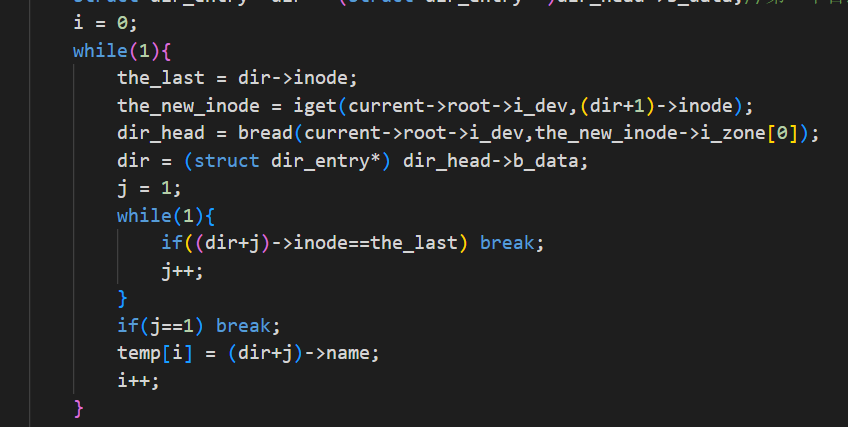


buffer\_head中的b\_data即对应的第一个逻辑块。

之后进行dir\_entry\*强制类型转换。

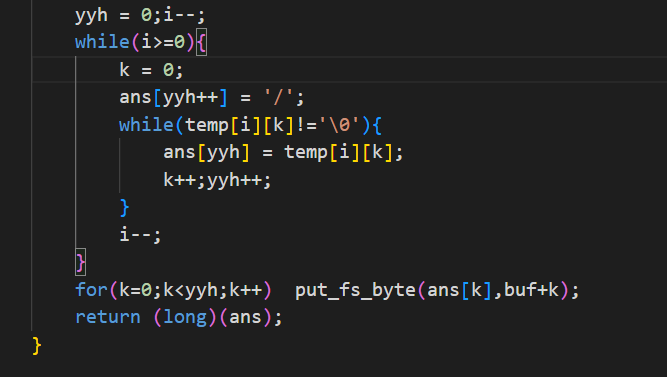


最终得到第一个目录项。



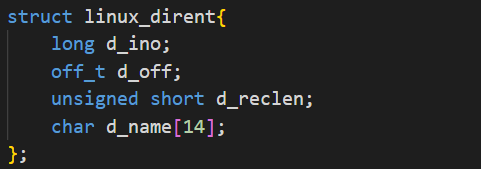
不断向上寻找，同时根据节点找到对应的名字。这里j从1开始，即从目录项中第二个开始（一般是上一级目录），当j==1时，表明当前是根目录，退出即可。

④

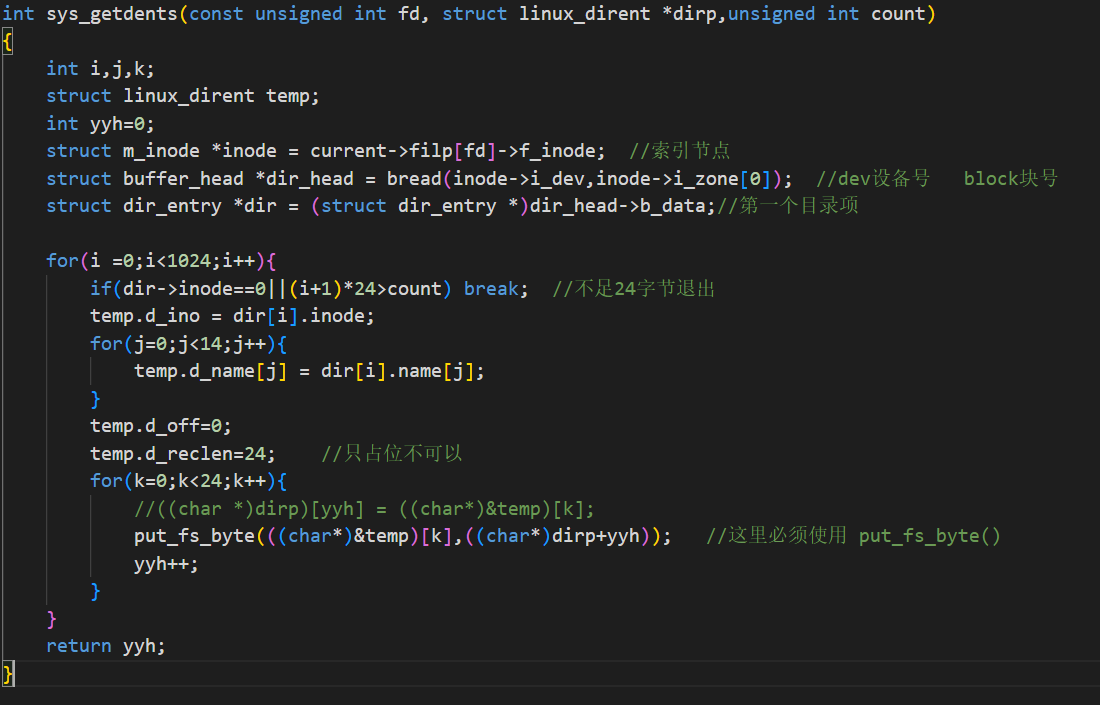


最后添加上’/’即可

1. sys\_getdents函数的实现



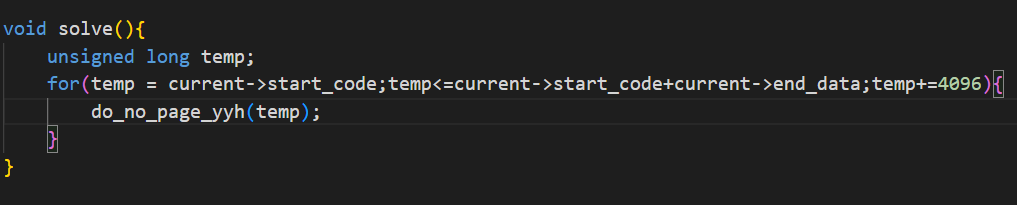
声明linux\_dirent结构体，否则在使用 struct linux\_dirent temp时编译会出错。



同sys\_getcwd前面一样，得到第一个目录项。之后进行for循环，1024足够，读取每一个目录项并用temp存储，然后利用put\_fs\_byte()函数。这里24字节要注意，否则运行时会出错。put\_fs\_byte()可以实现从核心态拷贝一个字节的数据到用户态。

1. sys\_execve2函数的实现

主要是仿照execve函数，添加相应的东西，大同小异，不作赘述。



核心在于do\_execve2需要提前进行读取代码段和数据段，避免页故障。从current->start\_code到current->start\_code+current->end\_data线性地址（页大小4096字节）调用do\_no\_page\_yyh。其中do\_no\_page\_yyh与do\_no\_page函数类似，只是删掉了printk以及第一个参数不用。

最终结果：

