程序 14-25 linux/include/linux/mm.h

```
 1 #ifndef _MM_H
 2 #define _MM_H
 3
 4 #define PAGE_SIZE 4096     // 定义 1 页内存页面字节数。注意高速缓冲块长度是 1024 字节。
 5
 6 #include <linux/kernel.h> // 内核头文件。含有一些内核常用函数的原型定义。
 7 #include <signal.h>       // 信号头文件。定义信号符号常量，信号结构以及信号操作函数原型。
 8
 9 extern int SWAP_DEV;      // 内存页面交换设备号。定义在 mm/memory.c 文件中，第 36 行。
10
   // 从交换设备读入和写出被交换内存页面。ll_rw_page()定义在 blk_drv/ll_rw_block.c 文件中。
   // 参数 nr 是主内存区中页面号；buffer 是读/写缓冲区。
11 #define read_swap_page(nr,buffer) ll_rw_page(READ,SWAP_DEV,(nr),(buffer));
12 #define write_swap_page(nr,buffer) ll_rw_page(WRITE,SWAP_DEV,(nr),(buffer));
13
   // 在主内存区中取空闲物理页面。如果已经没有可用内存了，则返回 0。
14 extern unsigned long get_free_page(void);
   // 把一内容已修改过的物理内存页面映射到线性地址空间指定处。与 put_page()几乎完全一样。
15 extern unsigned long put_dirty_page(unsigned long page,unsigned long address);
   //释放物理地址 addr 开始的 1 页面内存。
16 extern void free_page(unsigned long addr);
17 void swap_free(int page_nr);
18 void swap_in(unsigned long *table_ptr);
19
   //// 显示内存已用完出错信息，并退出。
   // 下面函数名前的关键字 volatile 用于告诉编译器 gcc 该函数不会返回。这样可让 gcc 产生更好
   // 一些的代码，更重要的是使用这个关键字可以避免产生某些（未初始化变量的）假警告信息。
20 extern inline volatile void oom(void)
21 {
   // do_exit()应该使用退出代码，这里用了信号值 SIGSEGV(11) 相同值的出错码含义是“资源
   // 暂时不可用”，正好同义。
22         printk("out of memory\n\r");
23         do_exit(SIGSEGV);
24 }
25
   // 刷新页变换高速缓冲宏函数。
   // 为了提高地址转换的效率，CPU 将最近使用的页表数据存放在芯片中高速缓冲中。在修改过
   // 页表信息之后，就需要刷新该缓冲区。 这里使用重新加载页目录基址寄存器 cr3 的方法来
   // 进行刷新。下面 eax = 0，是页目录的基址。
26 #define invalidate() \
27 __asm__("movl %%eax,%%cr3"::"a" (0))
28
29 /* these are not to be changed without changing head.s etc */
   /* 下面定义若需要改动，则需要与 head.s 等文件中的相关信息一起改变 */
   // Linux 0.12 内核默认支持的最大内存容量是 16MB，可以修改这些定义以适合更多的内存。
30 #define LOW_MEM 0x100000                    // 机器物理内存低端（1MB）。
31 extern unsigned long HIGH_MEMORY;           // 存放实际物理内存最高端地址。
32 #define PAGING_MEMORY (15*1024*1024)        // 分页内存 15MB。主内存区最多 15M。
33 #define PAGING_PAGES (PAGING_MEMORY>>12)    // 分页后的物理内存页面数（3840）。
34 #define MAP_NR(addr) (((addr)-LOW_MEM)>>12) // 指定内存地址映射为页面号。
35 #define USED 100                            // 页面被占用标志，参见 449 行。
36
```

```
    // 内存映射字节图（1 字节代表 1 页内存）。每个页面对应的字节用于标志页面当前被引用
    // （占用）次数。它最大可以映射 15Mb 的内存空间。在初始化函数 mem_init()中，对于不
    // 能用作主内存区页面的位置均都预先被设置成 USED（100）。
37  extern unsigned char mem_map [ PAGING_PAGES ];
38
    // 下面定义的符号常量对应页目录表项和页表（二级页表）项中的一些标志位。
39  #define PAGE_DIRTY      0x40            // 位 6，页面脏（已修改）。
40  #define PAGE_ACCESSED   0x20            // 位 5，页面被访问过。
41  #define PAGE_USER       0x04            // 位 2，页面属于：1-用户；0-超级用户。
42  #define PAGE_RW         0x02            // 位 1，读写权：1－写；0－读。
43  #define PAGE_PRESENT    0x01            // 位 0，页面存在：1-存在；0-不存在。
44
45  #endif
46
```