请查阅教材（表 3.4，3.5 和 3.6）和资料，标出 Linux 的 PCB（task\_struct）中标红字段的解释。

struct task\_struct {

……

# volatile long state; **//表示进程当前状态**

**/\* -1 : unrunnable 0 : runnable >0 : stopped \*/**

**void \*stack;//**

# unsigned int flags;**//表示进程状态的信息（非运行状态）**

**/\* flags值可以表示以下状态**

**PF\_ALIGNWARN 打印“对齐”警告信息。**

**PF\_PTRACED 被ptrace系统调用监控。**

**PF\_TRACESYS 正在跟踪。**

**PF\_FORKNOEXEC 进程刚创建，但还没执行。**

**PF\_SUPERPRIV 超级用户特权。**

**PF\_DUMPCORE dumped core。**

**PF\_SIGNALED 进程被信号(signal)杀出。**

**PF\_STARTING 进程正被创建。**

**PF\_EXITING 进程开始关闭。**

**PF\_USEDFPU 该进程使用FPU(SMP only)。**

**PF\_DTRACE delayed trace (used on m68k)。**

**\*/**

**unsigned int ptrace;//ptrace函数提供了一种父进程可以控制子进程运行，这里应该是可以用来控制子进程**

# int prio;//全称应该是priority，表示进程优先级

int static\_prio;

int normal\_prio;

unsigned int policy;

int nr\_cpus\_allowed;

cpumask\_t cpus\_allowed;

# struct sched\_info sched\_info;//表示调度信息的结构体

**struct list\_head tasks;//所有的进程组成的链表**

# struct mm\_struct \*mm;//内存描述符

struct mm\_struct \*active\_mm;

# int exit\_state;//表示进程的终止状态

int exit\_code;

int exit\_signal;

# int pdeath\_signal;//父进程消亡时发出的信号

unsigned sched\_reset\_on\_fork:1;

unsigned sched\_contributes\_to\_load:1;

unsigned sched\_migrated:1;

unsigned sched\_remote\_wakeup:1;

unsigned long atomic\_flags; /\* Flags requiring atomic access. \*/

struct restart\_block restart\_block;

# pid\_t pid;//进程标识号

**pid\_t tgid;//进程组织号**

# struct task\_struct rcu \*real\_parent;//被调试情况下真正的父进程

struct task\_struct rcu \*parent;**//父进程，一般情况下\*parent和\*real\_parent是指同一个进程，但如果\*real\_parent所指的进程终止，\*parent所指的进程就是父进程**

# struct list\_head children;//子进程链表

struct list\_head sibling;

struct task\_struct \*group\_leader;

# struct pid \*thread\_pid;//指向pid结构体

**char comm[TASK\_COMM\_LEN];//进程名**

# struct fs\_struct \*fs;//文件系统描述符

**struct files\_struct \*files;//打开的文件信息**

# struct signal\_struct \*signal;//进程接收到的信号

**struct thread\_struct thread;//CPU的部分状态(寄存器)保存在thread中**

};