in init/main.c

```
enum system_states system_state __read_mostly;
EXPORT_SYMBOL(system_state);
```

in kernel_init()

system_state = SYSTEM_RUNNING;

SYSTEM_RUNNING表示kernel初始化完毕，马上要进入user mode(即运行init).

```
1.  static int __ref kernel_init(void *unused)
2.  {
3.          int ret;
4.
5.          kernel_init_freeable();
6.          /* need to finish all async __init code before freeing the memory */
7.          async_synchronize_full();
8.          free_initmem();
9.          mark_rodata_ro();
10.         system_state = SYSTEM_RUNNING;
11.         numa_default_policy();
12.
13.         flush_delayed_fput();
14.
15.         if (ramdisk_execute_command) {
16.                 ret = run_init_process(ramdisk_execute_command);
17.                 if (!ret)
18.                         return 0;
19.                 pr_err("Failed to execute %s (error %d)\n",
20.                        ramdisk_execute_command, ret);
21.         }
22.
23.         /*
24.          * We try each of these until one succeeds.
25.          *
26.          * The Bourne shell can be used instead of init if we are
27.          * trying to recover a really broken machine.
28.          */
29.         if (execute_command) {
30.                 ret = run_init_process(execute_command);
31.                 if (!ret)
32.                         return 0;
33.                 pr_err("Failed to execute %s (error %d).  Attempting defaults...\
    n",
34.                         execute_command, ret);
35.         }
36.         if (!try_to_run_init_process("/sbin/init") ||
37.             !try_to_run_init_process("/etc/init") ||
38.             !try_to_run_init_process("/bin/init") ||
39.             !try_to_run_init_process("/bin/sh"))
40.                 return 0;
41.
42.         panic("No working init found.  Try passing init= option to kernel. "
43.               "See Linux Documentation/init.txt for guidance.");
44.  }
```

/* Values used for system_state */

extern enum system_states {

```
    SYSTEM_BOOTING,

    SYSTEM_RUNNING,

    SYSTEM_HALT,

    SYSTEM_POWER_OFF,

    SYSTEM_RESTART,

} system_state;
```

可以通过判断system_state的值来知道当前系统运行的状态。有时候会有用。比如只关心或不关心SYSTEM_BOOTING阶段的action，可以用system_state来过滤。