

程序 14-36 linux/include/sys/wait.h

```
1 #ifndef SYS_WAIT_H
2 #define SYS_WAIT_H
3
4 #include <sys/types.h>
5
6 #define LOW(v)          ((v) & 0377)           // 取低字节 (8 进制表示)。
7 #define HIGH(v)         (((v) >> 8) & 0377)    // 取高字节。
8
9 /* options for waitpid, WUNTRACED not supported */
10 /* waitpid 的选项, 其中 WUNTRACED 未被支持 */
11 // [注: 其实 0.12 内核已经支持 WUNTRACED 选项。上面这条注释应该是以前内核版本遗留下来的。]
12 // 以下常数符号是函数 waitpid(pid_t pid, long *stat_addr, int options) 中 options 使用的选项。
13 #define WNOHANG        1           // 如果没有状态也不要挂起, 并立刻返回。
14 #define WUNTRACED       2           // 报告停止执行的子进程状态。
15
16 // 以下宏定义用于判断 waitpid() 函数返回的状态字 (第 20、21 行的参数*stat_loc) 的含义。
17 #define WIFEXITED(s)   (!((s)&0xFF))    // 如果子进程正常退出, 则为真。
18 #define WIFSTOPPED(s)  (((s)&0xFF)==0x7F) // 如果子进程正停止着, 则为 true。
19 #define WEXITSTATUS(s)  (((s)>>8)&0xFF)   // 返回退出状态。
20 #define WTERMSIG(s)    ((s)&0x7F)       // 返回导致进程终止的信号值 (信号量)。
21 #define WCOREDUMP(s)   ((s)&0x80)       // 判断进程是否执行了内存映像转储 (dumpcore)。
22 #define WSTOPSIG(s)    (((s)>>8)&0xFF)   // 返回导致进程停止的信号值。
23 #define WIFSIGNALED(s) (((unsigned int)(s)-1 & 0xFFFF) < 0xFF) // 如果由于未捕捉信号而
24 // 导致子进程退出则为真。
25
26 // wait() 和 waitpid() 函数允许进程获取与其子进程之一的状态信息。各种选项允许获取已经终止或
27 // 停止的子进程状态信息。如果存在两个或两个以上子进程的状态信息, 则报告的顺序是不指定的。
28 // wait() 将挂起当前进程, 直到其子进程之一退出 (终止), 或者收到要求终止该进程的信号,
29 // 或者是需要调用一个信号句柄 (信号处理程序)。
30 // waitpid() 挂起当前进程, 直到 pid 指定的子进程退出 (终止) 或者收到要求终止该进程的信号,
31 // 或者是需要调用一个信号句柄 (信号处理程序)。
32 // 如果 pid= -1, options=0, 则 waitpid() 的作用与 wait() 函数一样。否则其行为将随 pid 和 options
33 // 参数的不同而不同。(参见 kernel/exit.c, 142)
34 // 参数 pid 是进程号; *stat_loc 是保存状态信息位置的指针; options 是等待选项, 见第 10, 11 行。
35
36 pid_t wait(int *stat_loc);
37 pid_t waitpid(pid_t pid, int *stat_loc, int options);
38
39 #endif
40
```