<https://blog.csdn.net/stpeace/article/details/73658137>

 上代码：

**[cpp]** [view plain](https://blog.csdn.net/stpeace/article/details/73658137) [copy](https://blog.csdn.net/stpeace/article/details/73658137)

1. #include <stdio.h>
2. #include <unistd.h>
3. #include <stdlib.h>
4. #include <signal.h>
6. **int** main(**int** argc, **char** \*argv[])
7. {
8. alarm(5);
9. sleep(20);
10. printf("end!\n");
11. **return** 0;
12. }

         运行5秒后， 内核向进程发出SIGALRM信息， 进程被终止， 所以上述程序的结果是：Alarm clock

         当然， 我们也可以人为定义信号处理函数， 如下：

**[cpp]** [view plain](https://blog.csdn.net/stpeace/article/details/73658137) [copy](https://blog.csdn.net/stpeace/article/details/73658137)

1. #include <stdio.h>
2. #include <unistd.h>
3. #include <stdlib.h>
4. #include <signal.h>
6. **void** sig\_alarm(**int** sig)
7. {
8. printf("sig is %d, sig\_alarm is called\n", sig);
9. }
11. **int** main(**int** argc, **char** \*argv[])
12. {
13. signal(SIGALRM, sig\_alarm); // 注册alarm信号对应的函数
14. alarm(5);  // 5秒后，内核向进程发出alarm信号， 执行对应的信号注册函数
15. sleep(20);
16. printf("end!\n");
17. **return** 0;
18. }

         结果：

sig is 14, sig\_alarm is called  
end!

        可以看到， 内核向应用进程发出SIGALRM信号， 执行对应的注册函数， 而非杀死进程。