4.5 进程间通讯 - 信号（三）_物联网 / 嵌入式工程师 - 慕课网

> 慕课网慕课教程 4.5 进程间通讯 – 信号（三）涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

- 在 Linux 系统中提供了 alarm 函数，用于设置定时器, 具体信息如下

函数头文件 #include <unistd.h>

函数原型 unsigned int alarm(unsigned int seconds);

函数功能 设置定时器的秒数

函数参数 seconds：定时的时间秒数

函数返回值 返回上一次进程设置定时器剩余的秒数

- 要点:
    - 定时器的定时任务由内核完成, alarm 函数值负责设置定时时间, 并告诉内核启动定时器
    - 当定时时间超时后，内核会向进程发出 SIGALRM 信号

示例一

验证 alram 函数的返回值

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <signal.h>
#include <unistd.h>
#include <sys/wait.h>


int main(void)
{
    unsigned int ret;

    ret = alarm(5);

    sleep(2);

    printf("ret = %d\n",ret);

    ret = alarm(3);

    sleep(1);

    printf("ret = %d\n",ret);


    return 0;
}
```

示例二

设置定时器的定时时间为 3s , 并处理 SIGALRM 信号

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#include <sys/types.h>
#include <signal.h>
#include <unistd.h>
#include <sys/wait.h>
#include <signal.h>
#include <string.h>



void do_alarm(int sig)
{
    printf("Recieve signal < %s >\n",strsignal(sig));
}

int main(void)
{
    __sighandler_t sigret;

    sigret = signal(SIGALRM,do_alarm);

    if (sigret == SIG_ERR){
        perror("[ERROR] signal(): ");
        exit(EXIT_FAILURE);
    }

    alarm(3);

    pause();

    return 0;
}
```

- 问题:
  - 在使用 wait() 函数时，由于阻塞或者非阻塞都非常消耗资源，并且在阻塞情况下，父进程不能执行其他逻辑
  -
- 解决方案
  - 子进程退出是异步事件, 可以利用在子进程退出时，会自动给父进程发送 SIGCHLD 信号
  -

示例

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <signal.h>
#include <unistd.h>
#include <sys/wait.h>
#include <signal.h>
#include <string.h>

void do_sig_child(int sig)
{
    printf("Receive signal < %s >\n",strsignal(sig));
    wait(NULL);
}

int main(void)
{
    pid_t cpid;
    __sighandler_t sigret;

    sigret = signal(SIGCHLD,do_sig_child);
    if (sigret == SIG_ERR){
        perror("[ERROR] signal(): ");
        exit(EXIT_FAILURE);
    }

    cpid = fork();
    if (cpid == -1){
        perror("[ERROR] fork(): ");
        exit(EXIT_FAILURE);
    }else if (cpid == 0){
        printf("Child process < %d > start.\n",getpid());
```

```
        sleep(2);
        exit(EXIT_SUCCESS);
    }else if (cpid > 0){
        while(1){

        }
    }
    return 0;
}
```

练习

- 探测用户是否已经输入，如果用户在 3 秒内没有输入则提示超时一次, 如果超时三次程序自动结束。

全文完