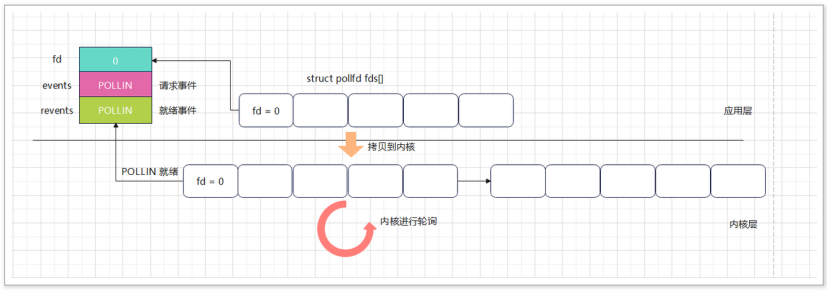
12.4 多路复用 io-poll（一）基本原理与应用_物联网 / 嵌入式工程师 - 慕课网

> 慕课网慕课教程 12.4 多路复用 io-poll（一）基本原理与应用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。

- 多路复用 poll 的方式与 select 多路复用原理类似，但很多地方不同，下面是具体的对比
  - 在应用层是以结构体（struct pollfd）数组的形式来进行管理文件描述符，在内核中基于链表对数组进行扩展，select 方式中文件描述符集合为 1024
  - 

  - poll 将请求与就绪事件通过结构体进行分开，不同重复对文件描述符数组进行赋值
  - select 将请求与就绪文件描述符存储在同一个集合中，导致每次都需要进行重新赋值才能进行下一次监控
  - 在内核中仍然使用的是轮询的方式，与 select 相同，当文件描述符越来越多时，则会影响效率
  -
- ## poll 多路复用实现主要调用 poll 函数的基本信息 如下:

函数头文件 #include <poll.h>

函数原型 int poll(struct pollfd *fds, nfds_t nfds, int timeout);

函数功能 监控多个文件描述符的变化

函数参数 fds ： sturct pollfd 结构体指针

nfds : fds 结构体的数量

timeout : 超时时间，单位为 ms

函数返回值

- 成功 :
  - > 0 返回就绪的文件描述符数量
  - = 0 超时返回，没有文件描述符就绪
- 失败 :
  - –1 : 发生错误，并设置 errno

- 参数相关说明
    - struct pollfd 结构体说明
    - ```
      struct pollfd {
         int   fd;
         short events;
         short revents;
       };
      ```
    - nfds_t 类型定义
    - ```
      typedef unsigned long int nfds_t;
      ```

- poll 事件说明

事件定义	说明
POLLIN	普通数据可读
POLLOUT	普通数据可写
POLLRDNORM	普通数据可读
POLLERR	发生错误

示例

使用 poll 函数监控标准输入，如果有输入，则获取标准输入的内容并打印

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

int main(void)
{
    int ret,maxfd = 0;
    struct pollfd pfds;
    char buffer[64] = {0};
    pfds.fd = 0;
    pfds.events = POLLIN;

    maxfd = pfds.fd;

    for(;;){
        ret = poll(&pfds,1,1000);
        if (ret == -1){
            perror("[ERROR] poll(): ");
            exit(EXIT_FAILURE);
        }else if (ret == 0){
            printf("Timeout.\n");
        }else if (ret > 0){
            if (pfds.revents & POLLIN){
                fgets(buffer,sizeof(buffer),stdin);
                printf("buffer :  %s ",buffer);
            }
        }
    }

    return 0;
}
```

练习

使用 poll 监听有名管道，当有名管道有数据时，读取数据并打印

全文完