

任务**23**：看门狗

3120104198

张瑞祥

实验目的

- 1.掌握看门狗的概念;
- 2.掌握Acadia或RPI或WRTnode上编写看门狗程序的方法;

实验器材

硬件

wrtnode板一块;

5V/1A电源一个;

microUSB线一根;

USB-TTL串口线一根（FT232RL芯片或PL2303芯片

以下为自备（可选）器材:

PC(Windows/Mac OS/Linux)一台;

以太网线一根（可能还需要路由器等)

软件

PC上的USB-TTL串口线配套的驱动程序;

PC上的串口终端软件，如minicom、picocom、putty等;

PC上的SSH软件，如putty等

实验步骤

1 理解看门狗的概念

linux man page上对watchdog的解释是

The Linux kernel can reset the system if serious problems are detected. This can be implemented via special watchdog hardware, or via a slightly less reliable software-only watchdog inside the kernel. Either way, there needs to be a daemon that tells the kernel the system is working fine. If the daemon stops doing that, the system is reset.

watchdog is such a daemon. It opens `/dev/watchdog`, and keeps writing to it often enough to keep the kernel from resetting, at least once per minute. Each write delays the reboot time another minute. After a minute of inactivity the watchdog hardware will cause the reset. In the case of the software watchdog the ability to reboot will depend on the state of the machines and interrupts.

The watchdog daemon can be stopped without causing a reboot if the device `/dev/watchdog` is closed correctly, unless your kernel is compiled with the `CONFIG_WATCHDOG_NOWAYOUT` option enabled.

所以我理解的linux下的看门狗是用于监视系统运行的程序，包括一个内核 **watchdog module** 和一个用户空间的 **watchdog** 程序。内核 **watchdog** 模块通过 `/dev/watchdog` 这个字符设备与用户空间通信。用户空间程序一旦打开 `/dev/watchdog` 设备（俗称“开门放狗”），就会导致在内核中启动一个1分钟的定时器（系统默认时间），此后，用户空间程序需要保证在1分钟之内向这个设备写入数据（俗称“定期喂狗”），每次写操作会导致重新设定定时器。如果用户空间程序在1分钟之内没有写操作，定时器到期会导致一次系统 **reboot** 操作。通过这种机制，我们可以保证系统核心进程大部分时间都处于运行状态，即使特定情形下进程崩溃，因无法正常定时“喂狗”，Linux系统在看门狗作用下重新启动（**reboot**），核心进程又运行起来了。多用于嵌入式系统。

2 编写程序并进行实验

配置内核中的硬件看门狗，使得一定时间内不喂狗就重启Acadia或RPI或WRTnode，写一个程序或脚本保持一定频率的喂狗，当关闭这个程序或脚本时形成重启。实验报告要记录和表现出重启。

首先启动硬件上的看门狗

```
sudo modprobe bcm2708_wdog
```

```
pi@raspberrypi: ~  
pi@raspberrypi ~$ su  
Password:  
su: Authentication failure  
pi@raspberrypi ~$ su -i  
su: invalid option -- 'i'  
Usage: su [options] [LOGIN]  
  
Options:  
-c, --command COMMAND      pass COMMAND to the invoked shell  
-h, --help                  display this help message and exit  
-, -l, --login              make the shell a login shell  
-m, -p,  
--preserve-environment      do not reset environment variables, and  
                             keep the same shell  
-s, --shell SHELL           use SHELL instead of the default in passwd  
  
pi@raspberrypi ~$ su  
Password:  
su: Authentication failure  
pi@raspberrypi ~$ sudo modprobe bcm  
bcm203x      bcm2708_wdog  bcm3510  
bcm2708-rng  bcm2835-v4l2  bcma  
pi@raspberrypi ~$ sudo modprobe bcm2708_wdog  
pi@raspberrypi ~$
```

然后在 /etc/modules 末尾添加bcm2708_wdog

```
pi@raspberrypi: ~  
# /etc/modules: kernel modules to load at boot time.  
#  
# This file contains the names of kernel modules that should be loaded  
# at boot time, one per line. Lines beginning with "#" are ignored.  
# Parameters can be specified after the module name.  
  
snd-bcm2835  
bcm2708_wdog  
:  
:
```

下面就是喂狗的程序

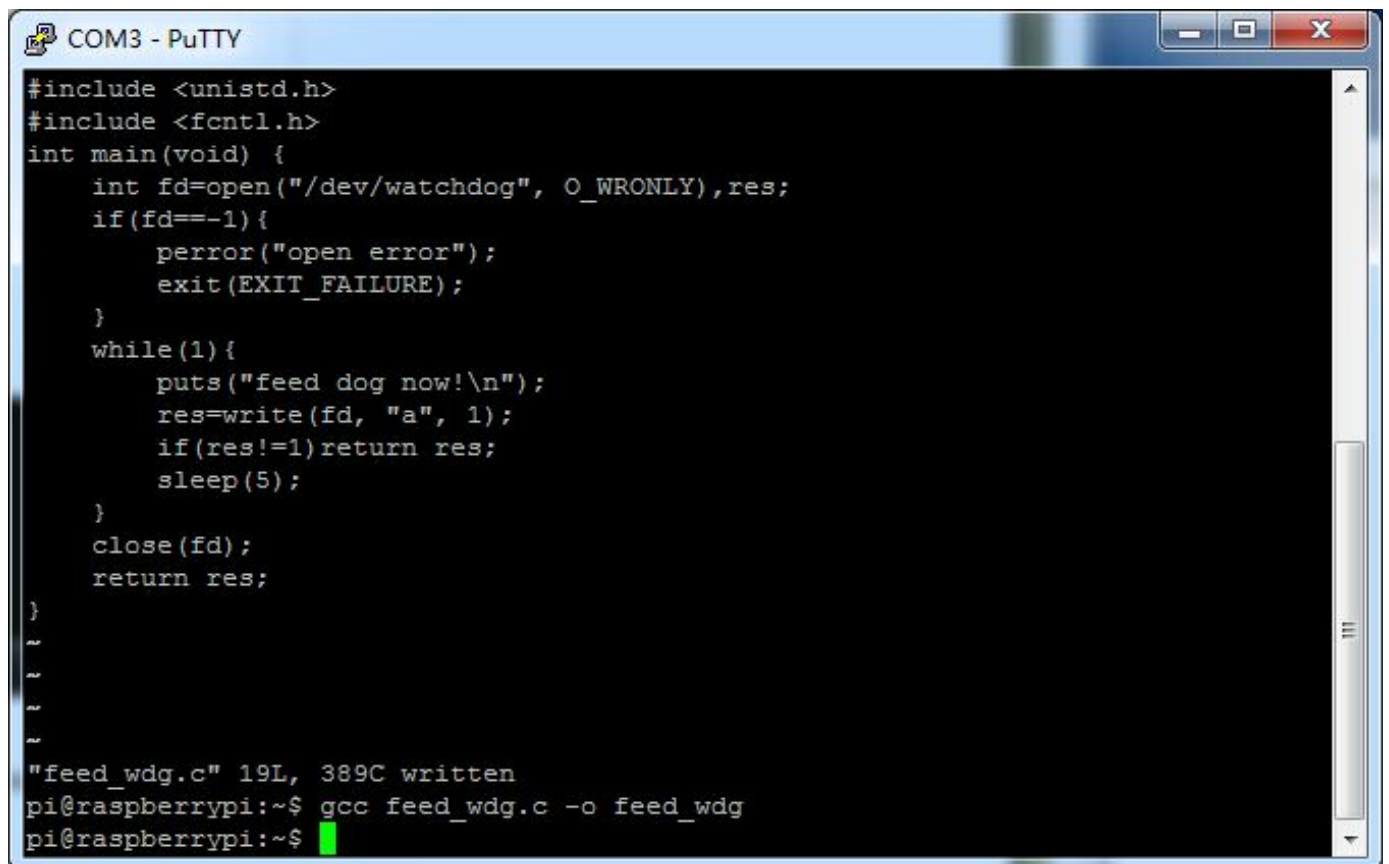
```
#include <stdio.h>  
#include <stdlib.h>
```

```

#include <unistd.h>
#include <fcntl.h>
int main(void) {
    int fd=open("/dev/watchdog", O_WRONLY),res;
    if(fd==-1){
        perror("open error");
        exit(EXIT_FAILURE);
    }
    while(1){
        puts("feed dog now!\n");
        res=write(fd, "a", 1);
        if(res!=1)return res;
        sleep(5);
    }
    close(fd);
    return res;
}

```

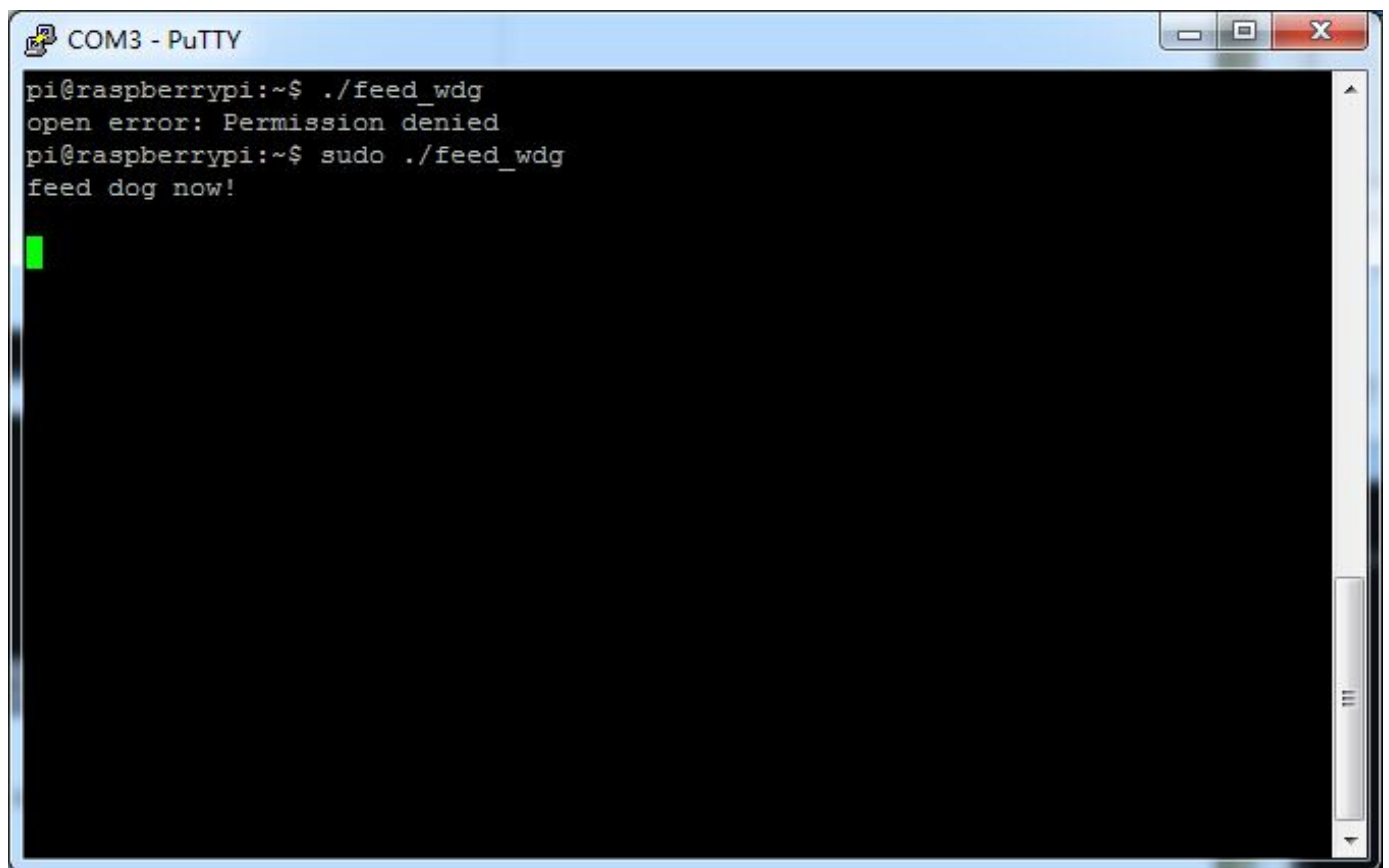
程序每隔5秒钟喂一次狗，编译运行



```

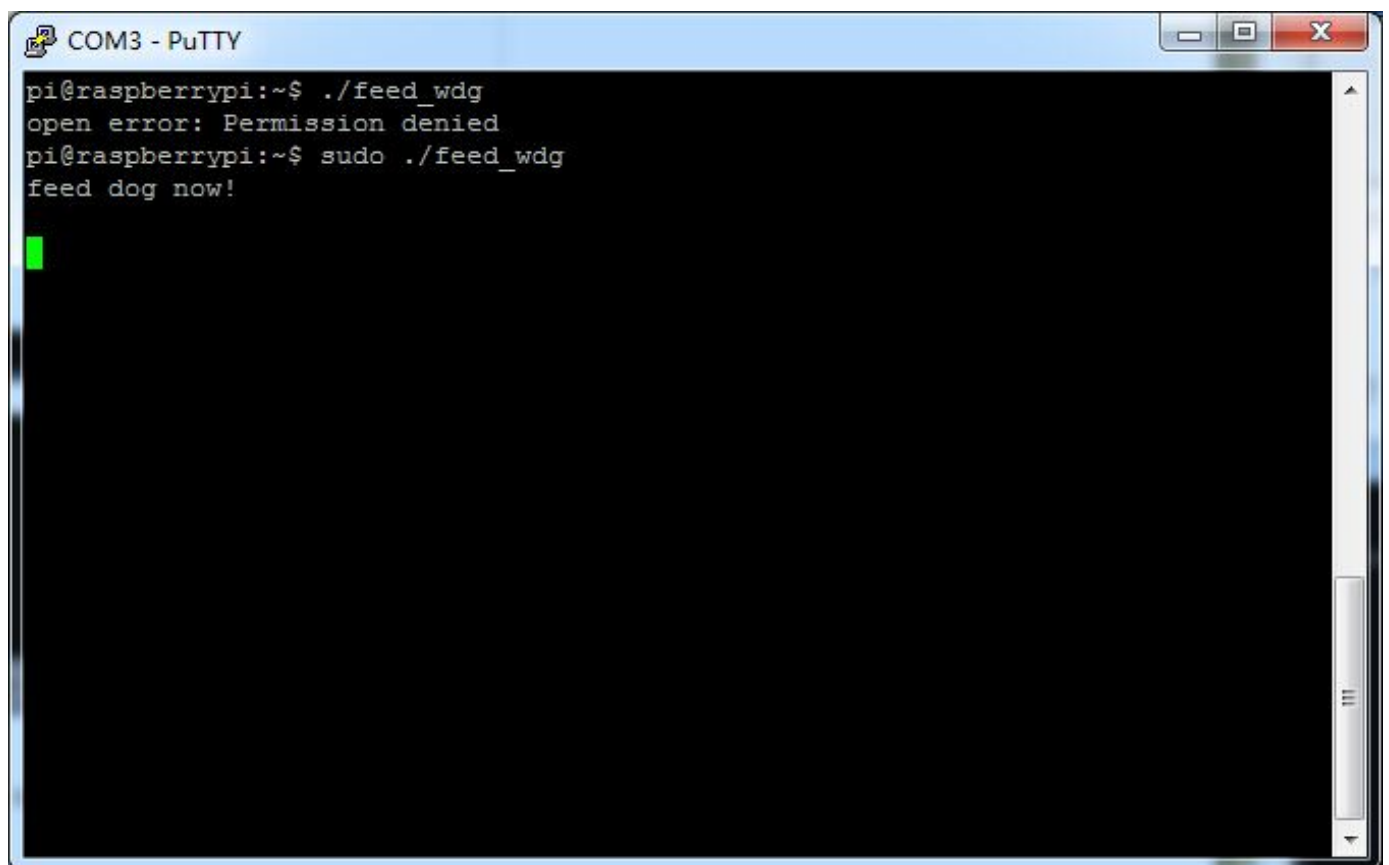
COM3 - PuTTY
#include <unistd.h>
#include <fcntl.h>
int main(void) {
    int fd=open("/dev/watchdog", O_WRONLY),res;
    if(fd==-1){
        perror("open error");
        exit(EXIT_FAILURE);
    }
    while(1){
        puts("feed dog now!\n");
        res=write(fd, "a", 1);
        if(res!=1)return res;
        sleep(5);
    }
    close(fd);
    return res;
}
~
~
~
~
"feed_wdg.c" 19L, 389C written
pi@raspberrypi:~$ gcc feed_wdg.c -o feed_wdg
pi@raspberrypi:~$ █

```



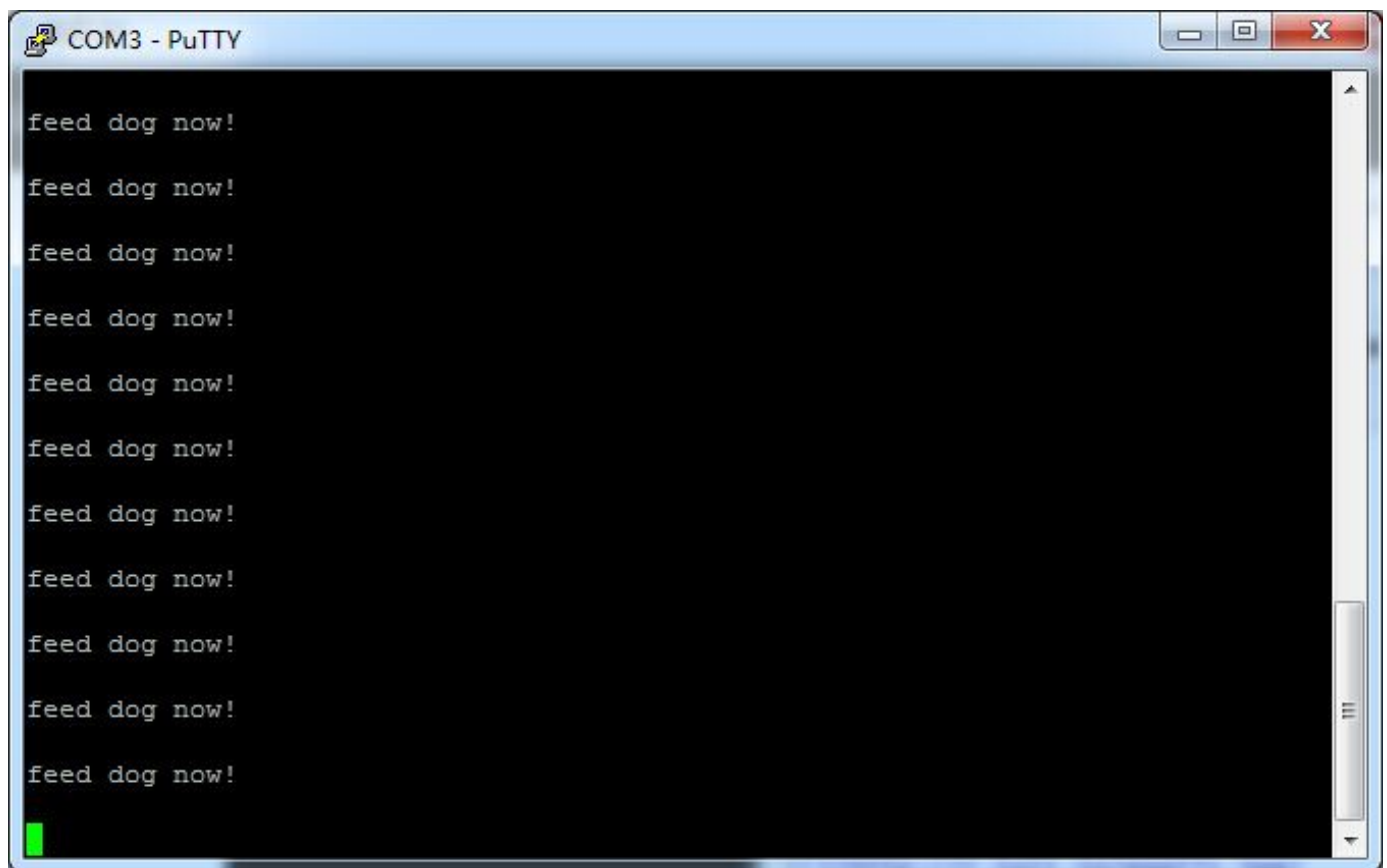
A terminal window titled "COM3 - PuTTY" with a standard Linux prompt. The user runs the command `./feed_wdg`, which results in a "Permission denied" error. They then attempt to run the same command with `sudo`, which also results in a "Permission denied" error. The prompt returns to the user, and a green cursor is visible on the line following the error message.

```
pi@raspberrypi:~$ ./feed_wdg
open error: Permission denied
pi@raspberrypi:~$ sudo ./feed_wdg
feed dog now!
```



A terminal window titled "COM3 - PuTTY" with a standard Linux prompt. The user runs the command `./feed_wdg`, which results in a "Permission denied" error. They then attempt to run the same command with `sudo`, which also results in a "Permission denied" error. The prompt returns to the user, and a green cursor is visible on the line following the error message.

```
pi@raspberrypi:~$ ./feed_wdg
open error: Permission denied
pi@raspberrypi:~$ sudo ./feed_wdg
feed dog now!
```



```
COM3 - PuTTY

feed dog now!

feed dog now!

feed dog now!

feed dog now!

feed dog now!

feed dog now!

feed dog now!

feed dog now!

feed dog now!

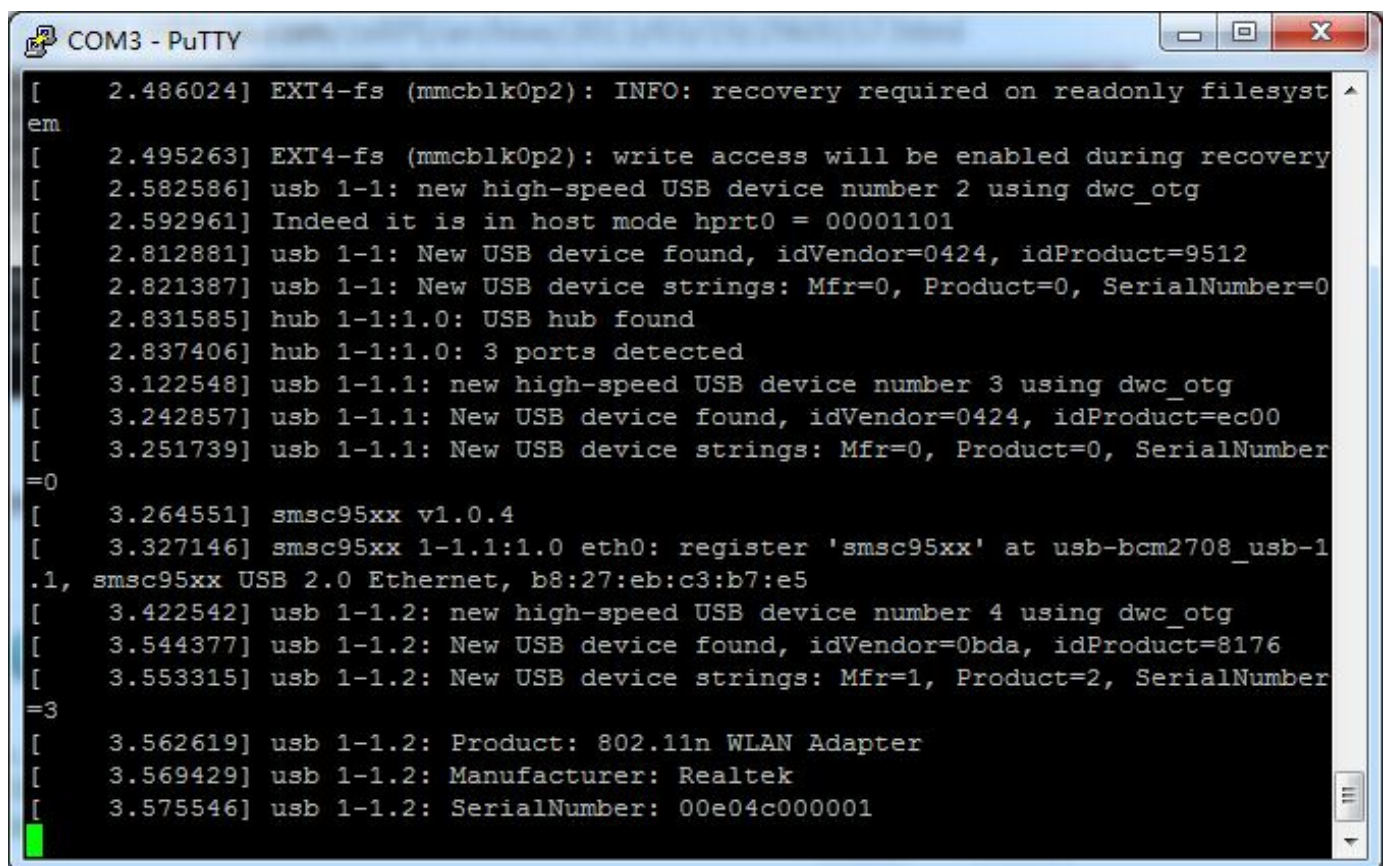
feed dog now!

feed dog now!

feed dog now!

feed dog now!
```

这时候结束程序运行，等待一会儿后系统就发生错误，然后重启



```
COM3 - PuTTY

[ 2.486024] EXT4-fs (mmcblk0p2): INFO: recovery required on readonly filesystem
[ 2.495263] EXT4-fs (mmcblk0p2): write access will be enabled during recovery
[ 2.582586] usb 1-1: new high-speed USB device number 2 using dwc_otg
[ 2.592961] Indeed it is in host mode hprt0 = 00001101
[ 2.812881] usb 1-1: New USB device found, idVendor=0424, idProduct=9512
[ 2.821387] usb 1-1: New USB device strings: Mfr=0, Product=0, SerialNumber=0
[ 2.831585] hub 1-1:1.0: USB hub found
[ 2.837406] hub 1-1:1.0: 3 ports detected
[ 3.122548] usb 1-1.1: new high-speed USB device number 3 using dwc_otg
[ 3.242857] usb 1-1.1: New USB device found, idVendor=0424, idProduct=ec00
[ 3.251739] usb 1-1.1: New USB device strings: Mfr=0, Product=0, SerialNumber=0
[ 3.264551] smsc95xx v1.0.4
[ 3.327146] smsc95xx 1-1.1:1.0 eth0: register 'smc95xx' at usb-bcm2708_usb-1.1, smc95xx USB 2.0 Ethernet, b8:27:eb:c3:b7:e5
[ 3.422542] usb 1-1.2: new high-speed USB device number 4 using dwc_otg
[ 3.544377] usb 1-1.2: New USB device found, idVendor=0bda, idProduct=8176
[ 3.553315] usb 1-1.2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[ 3.562619] usb 1-1.2: Product: 802.11n WLAN Adapter
[ 3.569429] usb 1-1.2: Manufacturer: Realtek
[ 3.575546] usb 1-1.2: SerialNumber: 00e04c000001
```

实验感悟

通过这次实验，对看门狗的运行机制有了一定的了解。

