操作系统 实验 2
系统调用

白晋斌
171860607
810594956@qq.com

目录

零、✨重点，精华都在这里

1.读取十六进制数的问题。

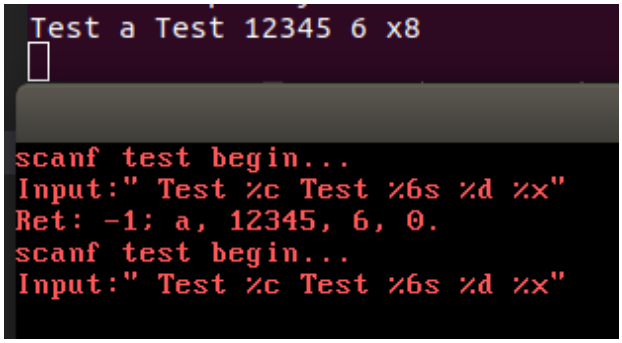
原代码中，十六进制必须要求格式为 0x123 这样，缺少 0x 会读取错误。这里未对框架代码做改动处理。

2.字符串长度的问题。

%6s 实际最多读取 5 个字符，这里以助教所给代码为准。

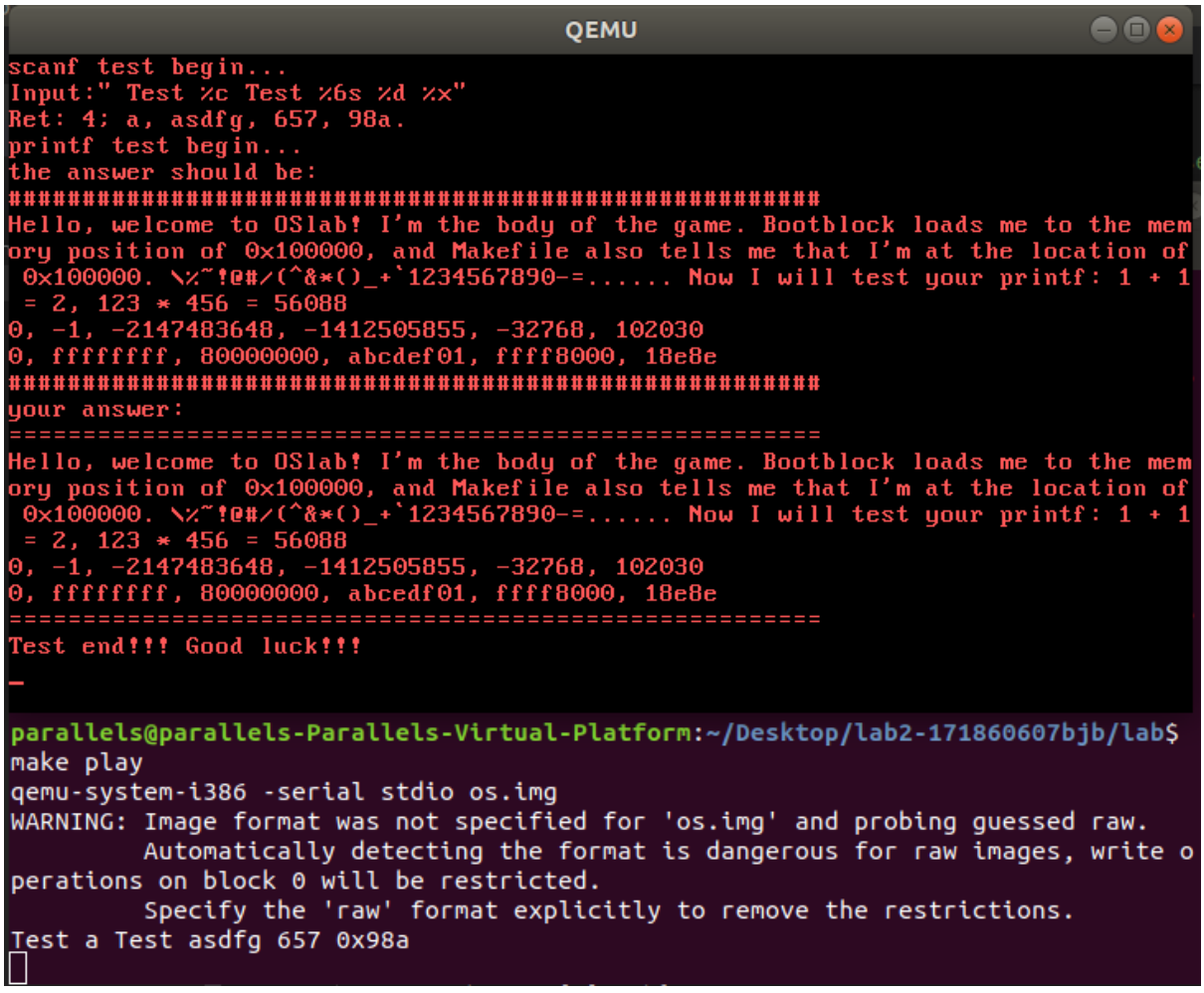
3.scanf 返回值问题。

在匹配过程中，自动去除了 buffer 和 format 中的空格，当匹配其他字符、数字时，如发生匹配不一致，scanf 函数将停止匹配，并返回-1。如图所示。



4.自行测试结果

如图所示，自行输入后结果如下图。



一、实验要求

本实验通过实现一个简单的应用程序，并在其中调用两个自定义实现的系统调用，介绍基于中断实现系统调用的全过程。

1. 实现系统调用库函数 printf 和 scanf。

实验流程如下：

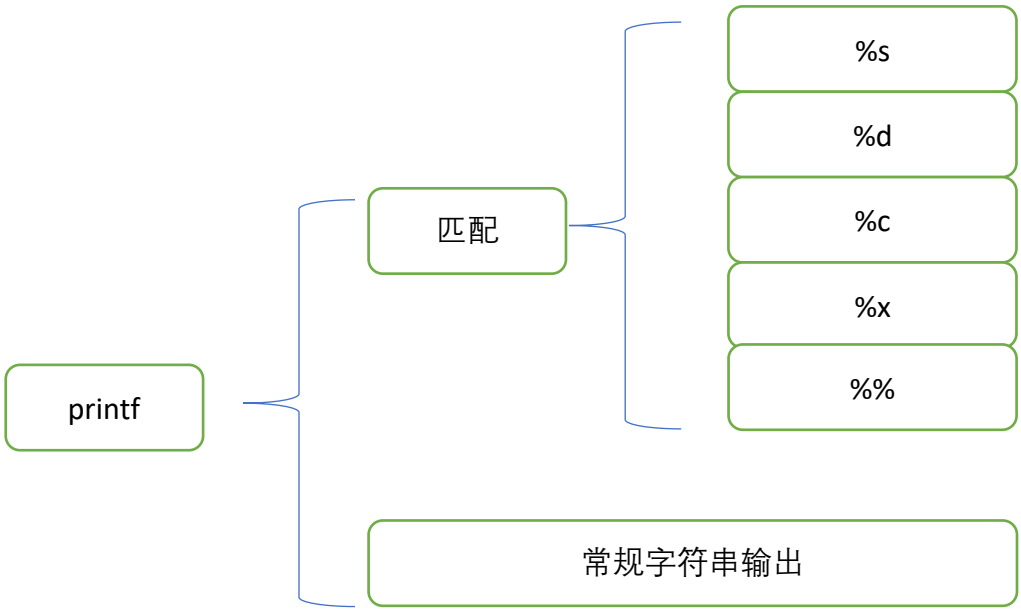
用户程序调用自定义实现的库函数 scanf 完成格式化输入和 printf 完成格式化输出。scanf 基于中断陷入内核，内核扫描按键状态获取输入完成格式化输入。printf 基于中断陷入内核，由内核完成在视频映射的显存地址中写入内容，完成字符串的打印。

2. 完善 scanf 和 printf 的格式化输入输出。

二、实验过程

1. 给 utils 文件夹下的两个.pl 文件可执行权限.

2.补充完成 printf 函数



如图所示。框架代码主要分为两大部分六小部分。通过 while 循环不断获取 ormat 中应该输出的字符，并存入 buffer。当 buffer 满或输出完成时，调用 syscall。

3.补充完成 syscallscan 函数

通过 getkeycode 获取键盘码，当值为 0x1c 终止获取，继续执行框架代码部分；当值不为 0（无按键按下），将键值存入 keybuffer，后续将 keybuffer 通过 getchar 转化为字符。

这一部分 0x1c 与标准库不太一样，我们通过查询框架代码可以得知 0x1c 是回车所对应的键码。

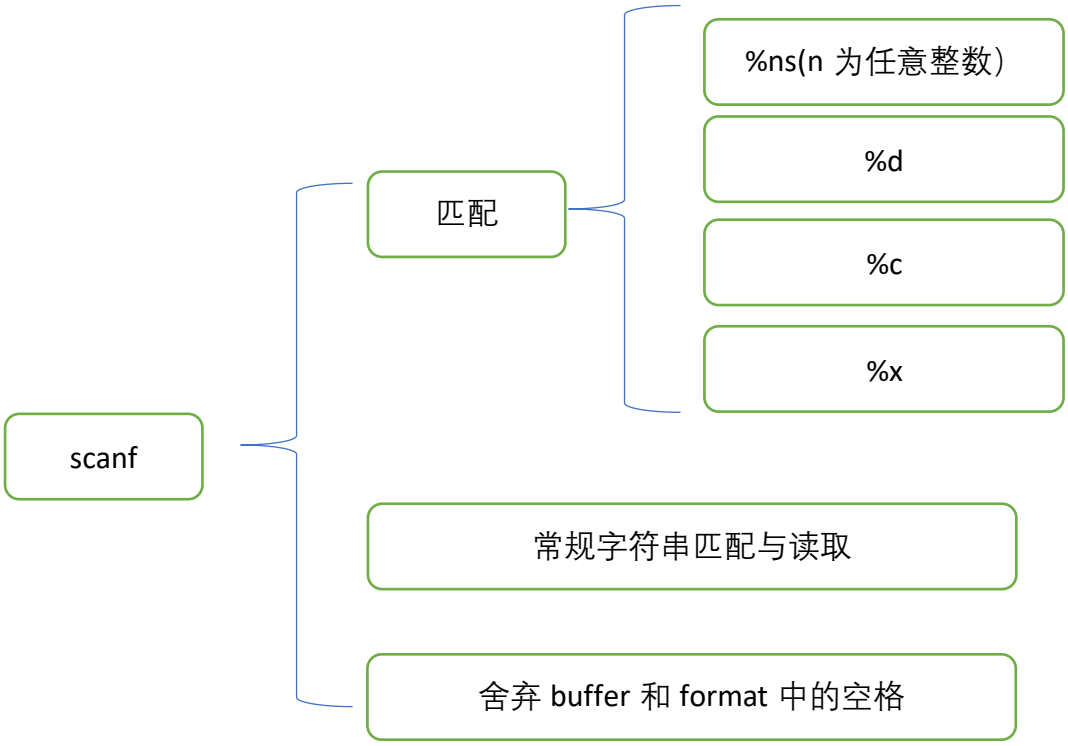
```
while(1){
        uint32_t k=getKeyCode();
```

```
        if (k==0x1c) {
            keyBuffer[bufferTail]=k;
            bufferTail=(bufferTail+1)%MAX_KEYBUFFER_SIZE;
            putChar('\n');//huixian
            break;
        }
        else if (k!=0) {
            keyBuffer[bufferTail]=k;
            bufferTail=(bufferTail+1)%MAX_KEYBUFFER_SIZE;
            putChar(getChar(k));//huixian
        }
    }
```

此外，还新增了回显功能，方便在 scanf 时观察自己的输入。

4.补充完成 scanf 函数

scanf 似乎是这三块中最复杂的，但依靠助教强有力的 API，写起来也很简单。大致结构如下。



基本逻辑即去除 buffer 和 format 中的空格，然后逐个扫描 format，遇到常规字符依次与自己所输入的进行比对，遇到%开头则读取 buffer 中的内容并写入 args。当遇到错误输入时，终止扫描返回-1，否则返回%匹配的参数个数。经测试可正常扫描。截图见第零部分。

三、拓展功能

1.新增了回显功能

方便在 scanf 时观察自己的输入。

```
putChar(getChar(k));
```

2.提高了代码的鲁棒性

检测到输入异常时，scanf 终止并返回-1。

四、代码逻辑

1.Bootloader 从实模式进入保护模式,加载内核至内存，并跳转执行

2.内核初始化 IDT（中断描述符表），初始化 GDT，初始化 TSS（任务状态段）

3.内核加载用户程序至内存，对内核堆栈进行设置，通过 iret 切换至用户空间，执行用户程序

4.用户程序调用自定义实现的库函数 scanf 完成格式化输入和 printf 完成格式化输出

5.scanf 基于中断陷入内核，内核扫描按键状态获取输入完成格式化输入

6.printf 基于中断陷入内核，由内核完成在视频映射的显存地址中写入内容，完成字符串的打印

五、友情鸣谢

感觉助教大大和各位群友鼎力相助！！！