



中国科学技术大学

University of Science and Technology of China

操作系统实验一

实验环境搭建 & Nachos系统调用

任课老师：李永坤

助教：张强、郭帆、吴加禹

2018/3/23

目录



中国科学技术大学
University of Science and Technology of China

- **Nachos介绍与安装**
 - **什么是Nachos**
 - **Nachos体系结构**
 - **Nachos工作模式**
 - **实验平台搭建**
 - **Nachos编译安装**
- **Linux源码阅读**
 - 阅读工具
 - 阅读要求
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

什么是Nachos



中国科学技术大学
University of Science and Technology of China

- Nachos
 - Nachos的全称是“Not Another Completely Heuristic Operating System”，它是一个可以修改和跟踪的操作系统教学软件
- Nachos
 - 美国加州大学伯克利分校在操作系统课程中使用的操作系统课程设计平台，使用C++编写，是一款教学用操作系统。
 - 在美国很多大学和国内一些高校中得到了应用，如：加州大学、纽约大学、北京大学、南开大学、四川大学...

什么是Nachos



中国科学技术大学
University of Science and Technology of China

- Nachos

- 给出了一个支持多线程和虚拟存储的操作系统骨架，可让学生在短时间内对操作系统中的基本原理和核心算法有一个全面和完整的了解

- Nachos

- Nachos建立在一个软件模拟的虚拟机之上，模拟了MIPS R2/3000的指令集、主存、中断系统、网络以及磁盘系统等操作系统所必须的硬件系统

什么是Nachos



中国科学技术大学
University of Science and Technology of China

- Nachos在操作系统教学方面具有以下优点：
 - 采用通用虚拟机
 - 面向对象性
 - 简单且易于扩展功能
 - 实现操作系统的基本功能和模块
 - 确定性调试比较方便

什么是Nachos



中国科学技术大学
University of Science and Technology of China

- 参考资料:
 - Nachos主页（包含Nachos介绍、源码）
 - <http://www.cs.washington.edu/homes/tom/nachos/>
 - 纽约大学的 Nachos英文教程《Nachos Project Guide》
 - 南昆士兰大学的 Nachos英文教程《Nachos Study Book》
 - 北京大学编写的《Nachos中文教程》
 -

目录



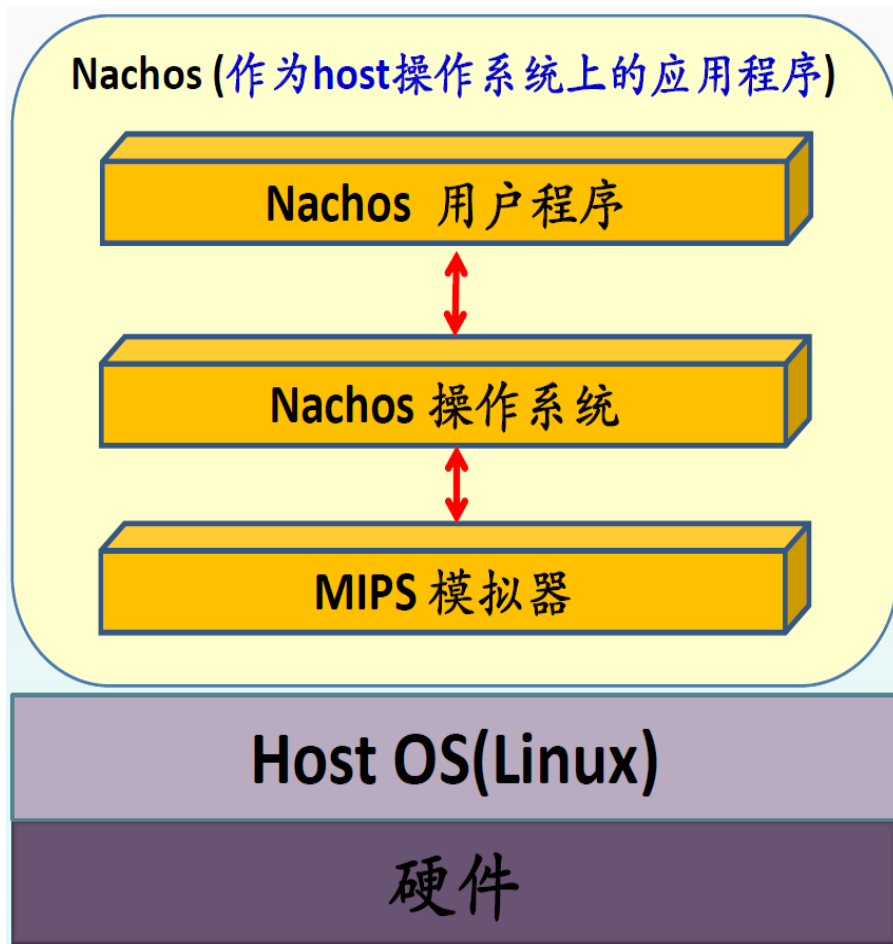
中国科学技术大学
University of Science and Technology of China

- **Nachos介绍与安装**
 - 什么是Nachos
 - **Nachos体系结构**
 - Nachos工作模式
 - 实验平台搭建
 - Nachos编译安装
- **Linux源码阅读**
 - 阅读工具
 - 阅读要求
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

Nachos体系结构

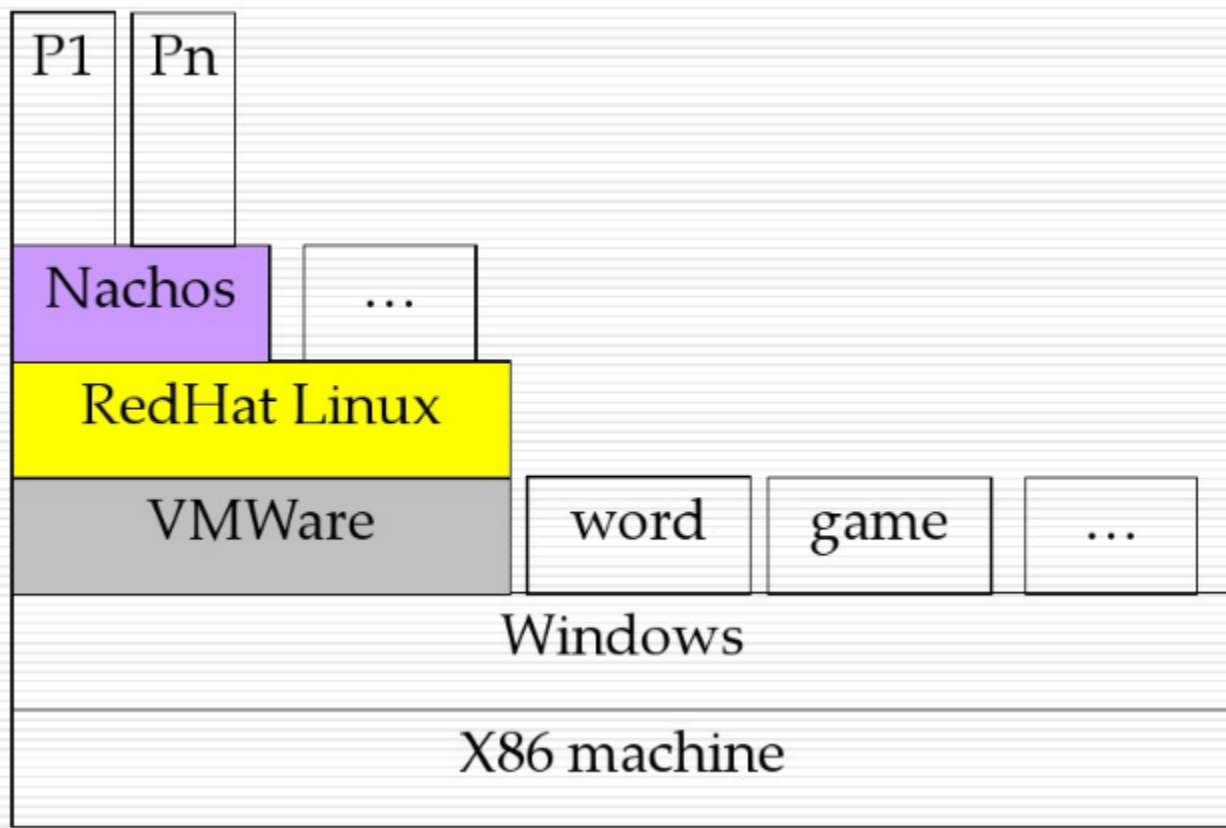


中国科学技术大学
University of Science and Technology of China



- Nachos本身是Linux宿主主机的一个进程
- Nachos包含一个MIPS虚拟机，用于执行用户程序
- 用户程序是c程序经过交叉编译后生成的mips可运行的程序
- Nachos Kernel负责用户程序的装载以及在MIPS模拟器上的执行

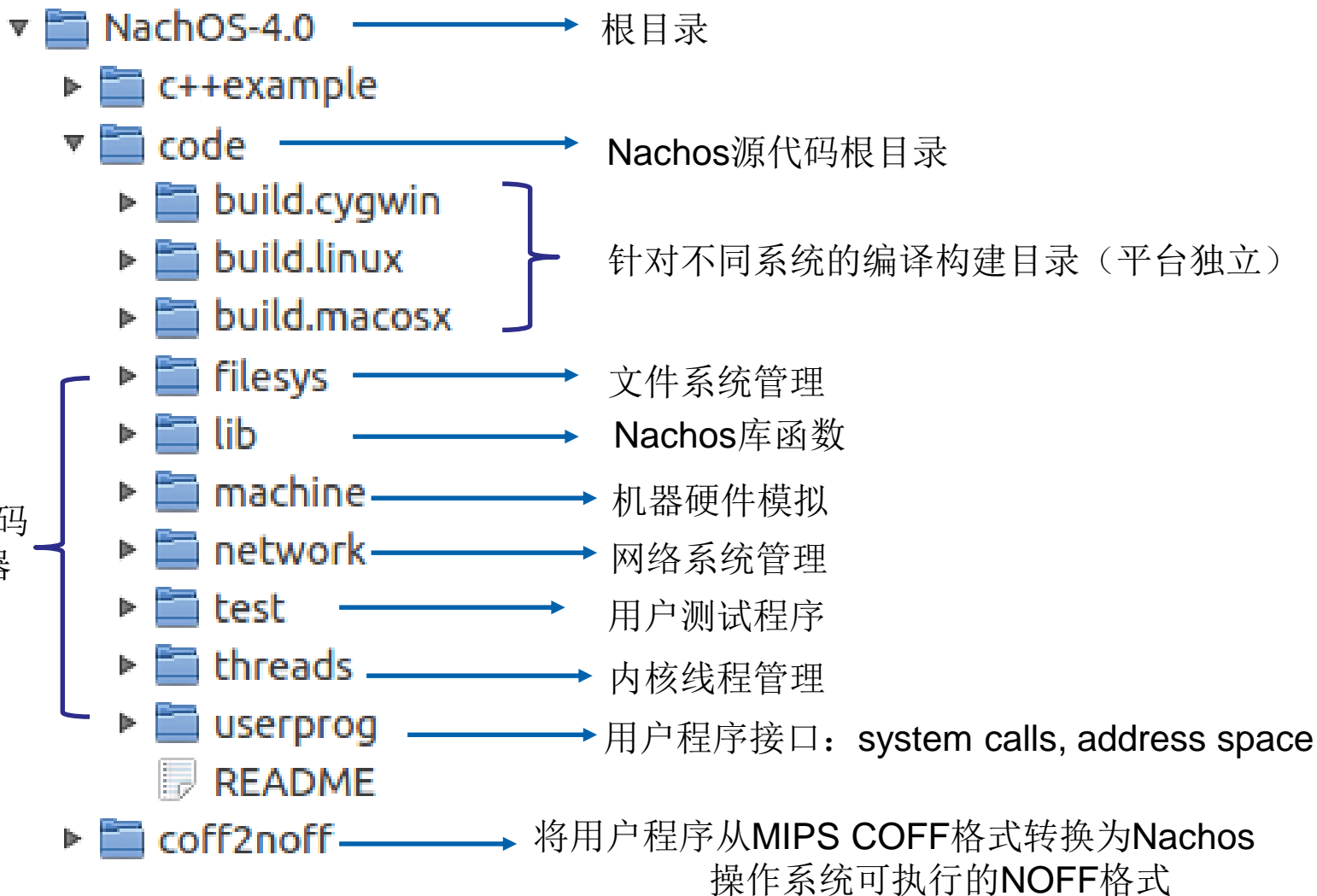
- Nachos与Linux/VMWare/Windows的关系



Nachos体系结构

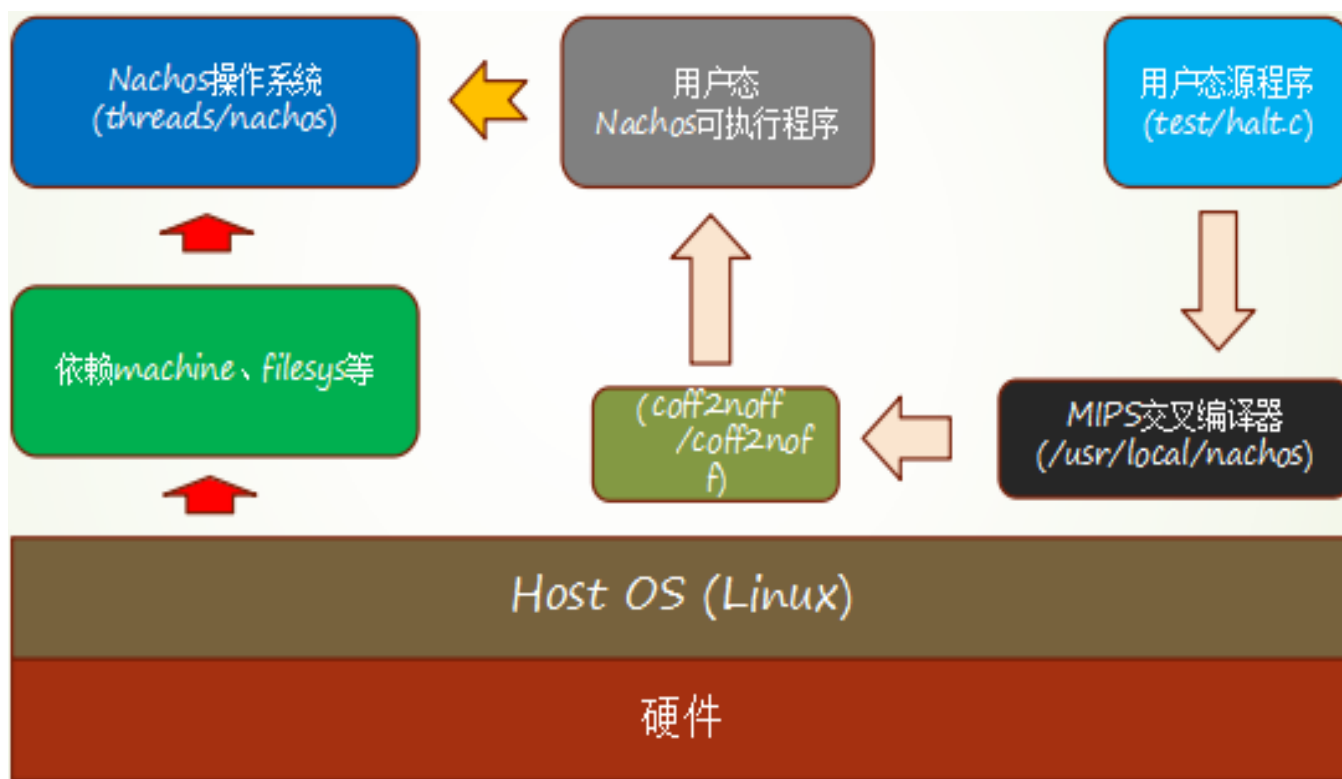


中国科学技术大学
University of Science and Technology of China



Nachos内核源码
和MIPS模拟器

- Nachos用户态程序与目录结构间的关系

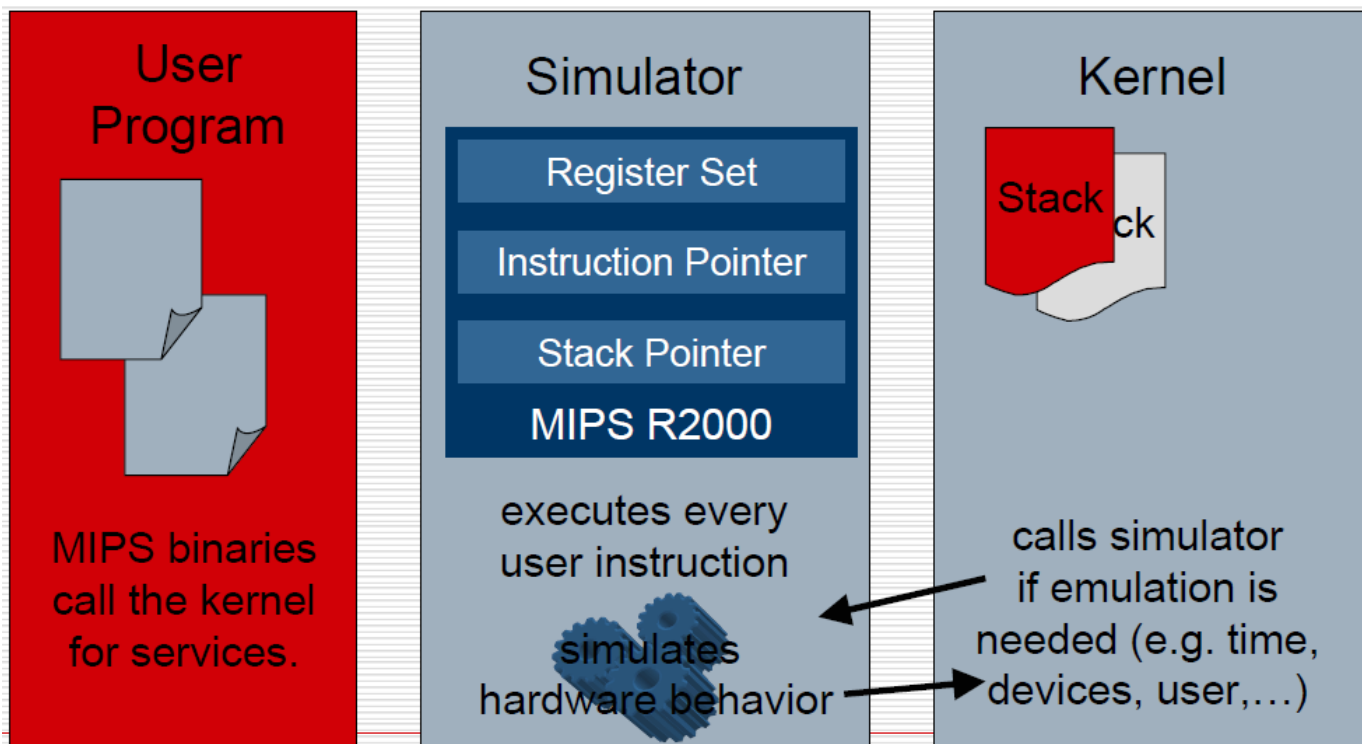


- **Nachos介绍与安装**
 - 什么是Nachos
 - Nachos体系结构
 - **Nachos工作模式**
 - 实验平台搭建
 - Nachos编译安装
- **Linux源码阅读**
 - 阅读工具
 - 阅读要求
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

Nachos工作模式



中国科学技术大学
University of Science and Technology of China



- Nachos的应用程序是MIPS指令格式的 (交叉编译生成)
- MIPS模拟器模拟硬件行为，并执行用户程序的每条指令
- 用户程序可触发系统异常，模拟器收到异常，转到kernel处理
- Nachos kernel实现系统服务

目录



中国科学技术大学
University of Science and Technology of China

- **Nachos介绍与安装**
 - 什么是Nachos
 - Nachos体系结构
 - Nachos工作模式
 - **实验平台搭建**
 - Nachos编译安装
- **Linux源码阅读**
 - 阅读工具
 - 阅读要求
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

- VMware Workstation Player的下载安装

- 下载链接:

- https://my.vmware.com/cn/web/vmware/free#desktop_end_user_computing/vmware_workstation_player/14_0

VMware Workstation 14.1.1 Player for
Windows 64-bit Operating Systems.

(exe | 90.64 MB)

[显示详细信息](#)

下载 ↓

关于本产品

描述

VMware Workstation 14.1.1 Player

文档

发行说明

VMware Workstation 14.1.1 Player for Linux
64-bit.







(bundle | 110.47 MB)

[显示详细信息](#)

下载 ↓

- Ubuntu内核镜像的下载安装（32位）

- 下载地址: <http://releases.ubuntu.com/trusty/>

	ubuntu-14.04.5-desktop-amd64.iso	2016-08-03 17:49	1.0G	Desktop image for 64-bit PC (AMD64)
	ubuntu-14.04.5-desktop-amd64.iso.torrent	2016-08-04 20:43	41K	Desktop image for 64-bit PC (AMD64)
	ubuntu-14.04.5-desktop-amd64.iso.zsync	2016-08-04 20:43	2.1M	Desktop image for 64-bit PC (AMD64)
	ubuntu-14.04.5-desktop-amd64.list	2016-08-03 17:49	4.5K	Desktop image for 64-bit PC (AMD64)
	ubuntu-14.04.5-desktop-amd64.manifest	2016-08-03 17:41	60K	Desktop image for 64-bit PC (AMD64)
	ubuntu-14.04.5-desktop-amd64.metalink	2016-08-04 20:46	44K	Ubuntu 14.04.5 LTS (Trusty Tahr)
	ubuntu-14.04.5-desktop-i386.iso	2016-08-03 17:52	1.0G	Desktop image for 32-bit PC (i386)
	ubuntu-14.04.5-desktop-i386.iso.torrent	2016-08-04 20:43	42K	Desktop image for 32-bit PC (i386)
	ubuntu-14.04.5-desktop-i386.iso.zsync	2016-08-04 20:43	2.1M	Desktop image for 32-bit PC (i386)

目录



中国科学技术大学
University of Science and Technology of China

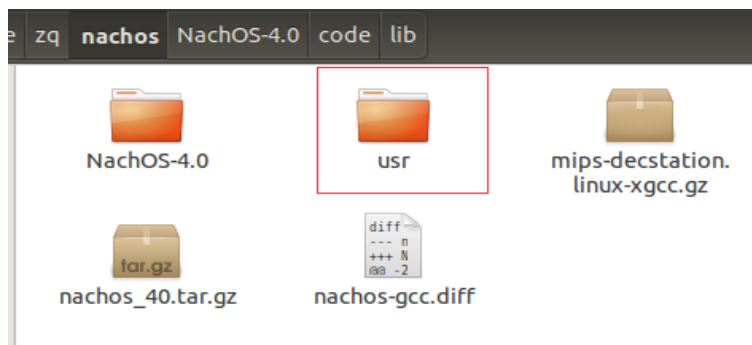
- **Nachos介绍与安装**
 - 什么是Nachos
 - Nachos体系结构
 - Nachos工作模式
 - 实验平台搭建
 - **Nachos编译安装**
- **Linux源码阅读**
 - 阅读工具
 - 阅读要求
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

- 创建 “nachos”文件夹
 - “Ctrl+Alt+T”打开终端窗口, “mkdir nachos”创建nachos文件夹并进入

```
zq@ubuntu:~/zq$ mkdir nachos  
zq@ubuntu:~/zq$ cd nachos/  
zq@ubuntu:~/zq/nachos$
```

- 下载源码
 - Nachos4.0源码
 - wget
http://mll.csie.ntu.edu.tw/course/os_f08/assignment/nachos_40.tar.gz
 - MIPS交叉编译器源码
 - wget http://mll.csie.ntu.edu.tw/course/os_f08/assignment/mips-decstation.linux-xgcc.gz
 - patch 文件
 - wget http://mll.csie.ntu.edu.tw/course/os_f08/assignment/nachos-gcc.diff

- 解压源码压缩包(解压到nachos目录下)
 - `tar zxvf nachos_40.tar.gz`
 - `tar zxvf mips-decstation.linux-xgcc.gz`



– MIPS交叉编译器的作用

- 如果需要在Nachos操作系统运行用户态程序，就需要用交叉编译器进行编译，生成Nachos下可执行文件。简单地说，就是在宿主机的Linux平台上生成Nachos操作系统中的可执行代码

- Nachos的编译安装
 - Patch Makefiles文件: `patch -p0 < nachos-gcc.diff`

```
zq@ubuntu:~/zq/nachos$ patch -p0 < nachos-gcc.diff
patching file NachOS-4.0/code/build.linux/Makefile
patching file NachOS-4.0/code/lib/list.cc
patching file NachOS-4.0/code/lib/sysdep.cc
patching file NachOS-4.0/code/lib/sysdep.h
patching file NachOS-4.0/code/test/Makefile
patching file NachOS-4.0/code/test/Makefile.dep
patching file NachOS-4.0/code/test/start.s
patching file NachOS-4.0/code/test/start.S
patching file NachOS-4.0/code/threads/switch.s
patching file NachOS-4.0/code/threads/switch.S
patching file NachOS-4.0/coff2noff/coff2noff.c
patching file NachOS-4.0/coff2noff/Makefile
zq@ubuntu:~/zq/nachos$
```

- NachOS-4.0/code/build.linux目录下依次执行`make depend/`
`make`命令编译安装

- 注：首次编译出现错误时，修改lib/list.cc文件,将所有的IsInList()函数改为this->IsInList()

```
../lib/list.cc:259:25: note: declarations in dependent base 'List<int>' are not
found by unqualified lookup
    ASSERT(IsInList(item));
    ^
../lib/debug.h:65:9: note: in definition of macro 'ASSERT'
    if (condition) {} else {
    ^
../lib/list.cc:259:25: note: use 'this->IsInList' instead
    ASSERT(IsInList(item));
    ^
../lib/debug.h:65:9: note: in definition of macro 'ASSERT'
    if (condition) {} else {
    ^
make: *** [libtest.o] Error 1
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/build.linux$
```

```
../threads/synchlist.cc:27:15: warning: deprecated conversion from string consta
nt to 'char*' [-Wwrite-strings]
    listEmpty = new Condition("list empty cond");
    ^
g++ -m32 -P -I../network -I../filesystem -I../userprog -I../threads -I../machine -I
../lib -I- -Dx86 -DLINUX -c ../threads/switch.S
cc1: note: obsolete option -I- used, please use -iquote instead
g++ bitmap.o debug.o libtest.o sysdep.o interrupt.o stats.o timer.o console.o ma
chine.o mipssim.o translate.o network.o disk.o alarm.o kernel.o main.o scheduler
.o synch.o thread.o addrspace.o exception.o synchconsole.o directory.o filehdr.o
filesystem.o pbitmap.o openfile.o synchdisk.o post.o switch.o -m32 -o nachos
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/build.linux$
```

- /NachOS-4.0/code/build.linux目录下执行`./nachos` 测试Nachos是否成功安装

```
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/build.linux$ ./nachos
Machine halting!

Ticks: total 10, idle 0, system 10, user 0
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/build.linux$
```

- coff2noff的编译安装: /NachOS-4.0/coff2noff目录下执行`make`命令编译

```
zq@ubuntu:~/zq/nachos/NachOS-4.0/coff2noff$ make
gcc -DRDATA -m32 -c -o coff2noff.o coff2noff.c
gcc -m32 coff2noff.o -o coff2noff.x86Linux
strip coff2noff.x86Linux
zq@ubuntu:~/zq/nachos/NachOS-4.0/coff2noff$
```

– 检查Nachos环境是否成功配置：

- test目录下执行`make`命令编译用户程序

```
usr/local/nachos/decstation-ultrix/bin/ -c sort.c
../../../../usr/local/nachos/bin/decstation-ultrix-ld -T script -N start.o sort.o
o sort.coff
../../../../coff2nooff/coff2nooff.x86Linux sort.coff sort
numsections 3
Loading 3 sections:
    ".text", filepos 0xd0, mempos 0x0, size 0x390
    ".data", filepos 0x460, mempos 0x390, size 0x0
    ".bss", filepos 0x0, mempos 0x390, size 0x1000
../../../../usr/local/nachos/bin/decstation-ultrix-gcc -G 0 -c -I../userprog -I../l
ib -B../../../../usr/local/nachos/lib/gcc-lib/decstation-ultrix/2.95.2/ -B../../../../
usr/local/nachos/decstation-ultrix/bin/ -c segments.c
../../../../usr/local/nachos/bin/decstation-ultrix-ld -T script -N start.o segments
.o -o segments.coff
../../../../coff2nooff/coff2nooff.x86Linux segments.coff segments
numsections 4
Loading 4 sections:
    ".text", filepos 0xf0, mempos 0x0, size 0x1e0
    ".rdata", filepos 0x2d0, mempos 0x1e0, size 0x20
    ".data", filepos 0x2f0, mempos 0x200, size 0x10
    ".bss", filepos 0x0, mempos 0x210, size 0x20
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/test$
```

- NachOS系统中运行用户程序，执行命令`../build.linux/nachos -x halt`

```
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x halt
Machine halting!

Ticks: total 22, idle 0, system 10, user 12
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/test$
```

目录



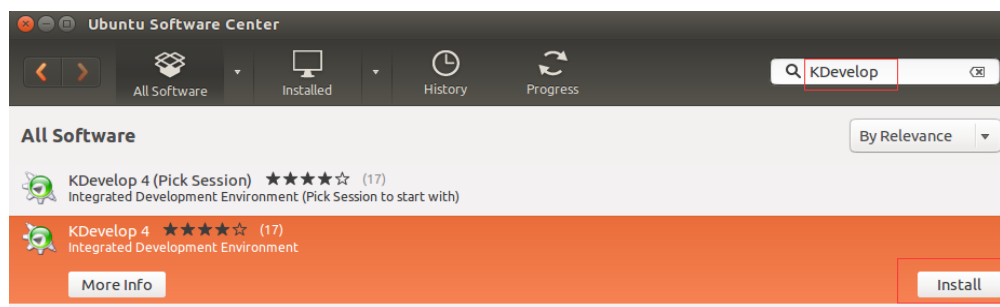
中国科学技术大学
University of Science and Technology of China

- **Nachos介绍与安装**
 - 什么是**Nachos**
 - **Nachos**体系结构
 - **Nachos**工作模式
 - 实验平台搭建
 - **Nachos**编译安装
- **Linux源码阅读**
 - **阅读工具**
 - 阅读要求
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

- Linux下使用gedit或vim编写代码，效率十分低下，为提高编程效率，推荐KDevelop/Sublime/SourceInsight等工具。

– KDevelop /Sublime Text (免费)

- Ubuntu软件中心可免费下载安装Kdevelop（推荐）



– SourceInsight（收费）

- 非常适合阅读Linux内核源码
- 可参考博客：<http://blog.csdn.net/jin13277480598/article/details/50387759>

- 阅读Linux内核源码

- 在线Linux内核阅读

- LXR工具: <https://lxr.missinglinkelectronics.com/linux>

- 下载Linux内核

- 下载地址: <https://mirrors.edge.kernel.org/pub/linux/kernel/>
 - 有V1.0-V4.x所有版本的内核源码
 - 这里演示下载Linux-4.3内核

Index of /pub/linux/kernel/

linux-4.3.5.tar.xz	31-Jan-2016 19:34	83M
linux-4.3.6.tar.gz	19-Feb-2016 22:39	126M
linux-4.3.6.tar.sign	19-Feb-2016 22:39	819
linux-4.3.6.tar.xz	19-Feb-2016 22:39	83M
linux-4.3.tar.gz	02-Nov-2015 00:23	126M
linux-4.3.tar.sign	02-Nov-2015 00:23	473
linux-4.3.tar.xz	02-Nov-2015 00:23	83M
linux-4.4.1.tar.gz	31-Jan-2016 19:34	127M
linux-4.4.1.tar.sign	31-Jan-2016 19:34	819
linux-4.4.1.tar.xz	31-Jan-2016 19:34	83M
linux-4.4.10.tar.gz	11-May-2016 15:03	127M

[../](#)
[Historic/](#)
[SillySounds/](#)
[crypto/](#)
[next/](#)
[people/](#)
[ports/](#)
[projects/](#)
[testing/](#)
[uemacs/](#)

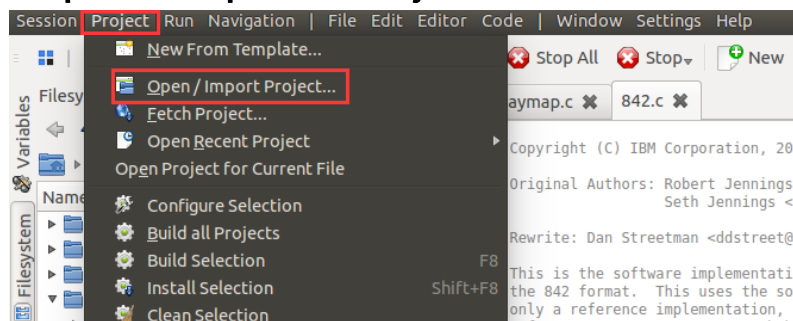
[v1.0/](#)
[v1.1/](#)
[v1.2/](#)
[v1.3/](#)
[v2.0/](#)
[v2.1/](#)
[v2.2/](#)
[v2.3/](#)
[v2.4/](#)
[v2.5/](#)
[v2.6/](#)
[v3.0/](#)
[v3.x/](#)
[v4.x/](#)

20-Mar-2003 22:38	-
26-Jun-2017 21:57	-
24-Nov-2001 14:54	-
20-Mar-2018 07:29	-
17-Apr-2017 18:13	-
13-Mar-2003 01:34	-
18-Sep-2012 20:27	-
14-Feb-2002 05:32	-
20-Mar-2003 23:31	-
20-Mar-2003 22:58	-
20-Mar-2003 22:58	-
20-Mar-2003 22:58	-
20-Mar-2003 23:02	-
08-Feb-2004 09:17	-
20-Mar-2003 23:12	-
24-Mar-2004 19:22	-
20-Mar-2003 23:23	-
01-May-2013 14:14	-
14-Jul-2003 03:50	-
08-Aug-2013 19:12	-
19-Mar-2018 20:08	-
19-Mar-2018 20:08	-
19-Mar-2018 08:19	-

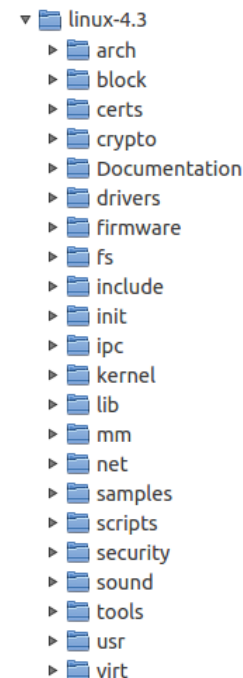
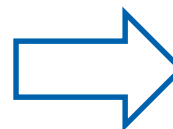
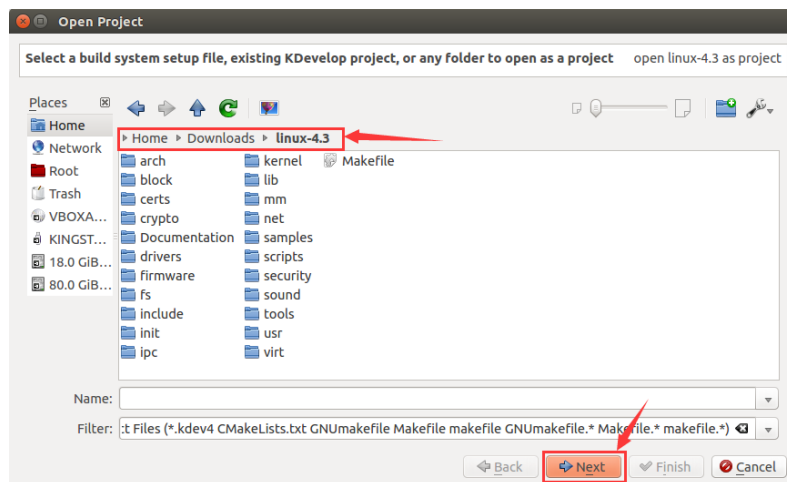
- 解压内核源码包

- `tar xvJf linux-4.3.tar.xz`

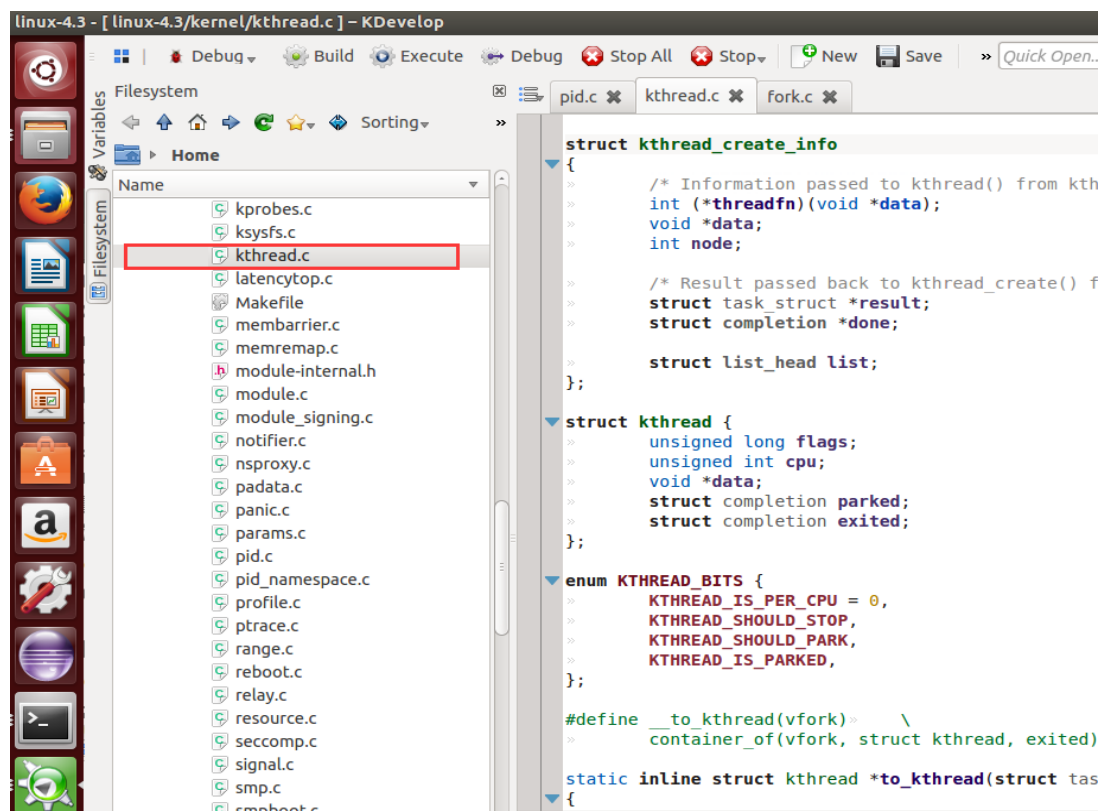
- 阅读Linux内核源码
 - 打开KDevelop软件,导入内核源码
 - Project->Open/Import Project



- 选择内核Linux-4.3文件夹，Next->Finish



- 阅读Linux内核源码
 - 阅读内核源码
 - 核心代码在kernel目录下，比如查看线程模块源码kthread.c



- **Nachos介绍与安装**
 - 什么是**Nachos**
 - **Nachos**体系结构
 - **Nachos**工作模式
 - 实验平台搭建
 - **Nachos**编译安装
- **Linux源码阅读**
 - 阅读工具
 - **阅读要求**
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

- 阅读内容--查看Linux内核源码
 - 第一次主要熟悉Linux内核的组成架构、核心功能模块
 - 如Linux内核由哪几大模块构成？进程/线程模块、内存管理模块的所在目录？
- 阅读要求
 - 阅读指定模块的内核源码后，需提交一份**源码阅读报告**！

第一次Linux内核源码阅读的具体内容和阅读报告要求
下周通知！！！！

目录



中国科学技术大学
University of Science and Technology of China

- **Nachos介绍**
 - 什么是**Nachos**
 - **Nachos**体系结构
 - **Nachos**工作模式
- **Nachos安装**
 - 实验平台搭建
 - **Nachos**编译安装
 - 源码阅读工具
- **实验内容**
 - **系统调用概述**
 - 实验要求
 - 实验提示
 - 测试程序的编写与运行

- 系统调用的概念
 - 将一些同硬件或者和操作系统功能有密切联系的一部分常用指令序列抽取出来，使它们成为操作系统本身的一部分，这就是所谓的系统调用
- 系统调用
 - 只有用户态的程序才能进行系统调用，进行系统调用后的状态变成系统态，一般程序在系统态运行会有较高的优先级。系统调用作为操作系统的一部分，长驻内存。
 - 操作系统提供的系统调用太少，会影响用户程序的执行效率和用途；因为不是每种应用都会用到所有的系统调用，系统调用数目太多，则会使得操作系统的核心过于庞大而同样影响系统的效率。

- Nachos 系统调用

- Nachos的系统调用函数定义在userprog/ksyscall.h头文件中，其运行机制与linux 的系统调用相似。
- 在该文件中定义了两个系统调用，分别为：
 - SysHalt(): 实现中断操作
 - SysAdd(): 实现两个数的相加操作
- **Nachos的系统调用实现过程分为5步:**
 1. 定义系统调用号和系统调用接口
 2. 添加进入内核系统调用的接口
 3. 在内核中修改中断入口处理函数
 4. 在内核中实现系统调用函数
 5. 编写用户测试程序

系统调用概述



中国科学技术大学
University of Science and Technology of China

- 以添加Sub系统调用为例进行分析：
 - 第一步：定义系统调用号和系统调用接口
 - code/userprog/syscall.h中添加

```
#define SC_Sub    43
int Sub(int op1, int op2);
```

- 第二步：添加进入内核系统调用的接口
 - code/test/start.S中添加

```
.globl Sub
.ent Sub
```

Sub:

```
addiu $2,$0,SC_Sub
syscall
j      $31
.end Sub
```

在r2寄存器中存放系统调用号SC_Halt

触发系统中断调用



跳转到中断返回寄存器



— 第三步：在内核中修改中断入口处理函数

- 系统调用引发中断的异常处理函数是code/userprog/exception.cc 中的ExceptionHandler(), 异常类型为SyscallException。
- 中断处理函数中添加相应操作:

```
void ExceptionHandler(ExceptionType which){ ← CPU执行到syscall指令时触发中断异常  
    int type = kernel->machine->ReadRegister(2); ← 从r2寄存器读取系统调用号  
    switch (which) {  
        case SyscallException:  
            switch(type) {  
                case SC_Sub:  
                    int subResult, op1, op2;  
                    op1= (int)kernel->machine->ReadRegister(4); ← 从r4、r5寄存器读取  
                    op2= (int)kernel->machine->ReadRegister(5); ← 系统调用参数  
                    subResult =SysSub(op1,op2); ← 调用内核系统调用函数完成运算  
                    kernel->machine->WriteRegister(2, (int) subResult);← 返回参数写入r2寄存器  
                    printf("%d-%d=%d\n",op1,op2, subResult);  
                    /* Modify return point */  
                    {  
                        /* set previous programm counter (debugging only)*/  
                        kernel->machine->WriteRegister(PrevPCReg, kernel->machine->ReadRegister(PCReg));
```

系统调用概述



中国科学技术大学
University of Science and Technology of China

```
/* set programm counter to next instruction (all Instructions are 4 byte wide)*/  
kernel->machine->WriteRegister(PCReg, kernel->machine->ReadRegister(PCReg) + 4);  
/* set next programm counter for brach execution */  
kernel->machine->WriteRegister(NextPCReg, kernel->machine->ReadRegister(PCReg)+4);  
}  
return;  
ASSERTNOTREACHED();  
break;  
...  
}  
}
```

- 注：对于有参数的系统调用，MIPS编译器的参数传递规则如下.

参数1:	r4寄存器
参数2:	r5寄存器
参数3:	r6寄存器
参数4:	r7寄存器
系统调用号/返回参数:	r2寄存器
中断返回跳转到:	r3寄存器

– 第四步：在内核中实现系统调用函数

- userprog/ksyscall.h文件中添加

```
int SysSub(int op1, int op2)  
{  
    return op1 - op2;  
}
```

– 第五步：编写用户测试程序

- code/test/目录下添加sub.c文件

```
#include "syscall.h"  
int main(){  
    int result;  
    result = Sub(42, 23);  
    Halt();  
}
```

— 第五步：编写用户测试程序

- 修改用户程序目录（code/test）下的Makefile文件

1) 添加测试程序名称

change this if you create a new test program!

PROGRAMS = add halt shell matmult sort segments sub

2) 添加测试程序的交叉编译指令

sub.o: sub.c

\$(CC) \$(CFLAGS) -c sub.c

sub: sub.o start.o

\$(LD) \$(LDFLAGS) start.o sub.o -o sub.coff

\$(COFF2NOFF) sub.coff sub

目录



中国科学技术大学
University of Science and Technology of China

- **Nachos介绍**
 - 什么是**Nachos**
 - **Nachos**体系结构
 - **Nachos**工作模式
- **Nachos安装**
 - 实验平台搭建
 - **Nachos**编译安装
 - 源码阅读工具
- **实验内容**
 - 系统调用概述
 - **实验要求**
 - 实验提示
 - 测试程序的编写与运行

实验要求



中国科学技术大学
University of Science and Technology of China

- 实验目的：
 - Nachos实验平台的搭建与熟悉
 - 掌握系统调用实现的原理与流程
- 实验内容：
 - 基于Nachos实验平台实现Create, Open, Write, Read, Close系统调用，并编写测试程序进行测试，通过上述实现的系统调用完成文件的创建、打开/关闭、读/写。
- 实验环境
 - VMware虚拟机（32位）+ Nachos-4.0

实验要求



中国科学技术大学
University of Science and Technology of China

- 实验检查：
 - 程序正确运行并实现指定系统调用功能
 - 提问3~4个，对相关核心代码的解释说明
 - 检查时间：待定
- 实验提交：
 - 提交内容：实验源码+实验报告（打包成一个压缩包）
 - 命名方式：“学号+姓名+实验一”

具体考核方式和实验提交内容
下周通知！！！！

目录



中国科学技术大学
University of Science and Technology of China

- **Nachos介绍**
 - 什么是**Nachos**
 - **Nachos**体系结构
 - **Nachos**工作模式
- **Nachos安装**
 - 实验平台搭建
 - **Nachos**编译安装
 - 源码阅读工具
- **实验内容**
 - 系统调用概述
 - 实验要求
 - **实验提示**
 - 测试程序的编写与运行

- Nachos实验平台:

- 上述系统调用在/userprog/syscall.h均已声明各自的系统调用号和系统调用接口。
- 实验内容需要在code/test/start.S中使用汇编语言添加进入内核系统调用的接口，修改中断处理函数并编写内核服务函数。

```
#define SC_Halt          0
#define SC_Exit          1
#define SC_Exec          2
#define SC_Join          3
#define SC_Create        4
#define SC_Remove        5
#define SC_Open          6
#define SC_Read          7
#define SC_Write         8
#define SC_Seek          9
#define SC_Close        10
#define SC_ThreadFork    11
#define SC_ThreadYield   12
#define SC_ExecV         13
#define SC_ThreadExit    14
#define SC_ThreadJoin    15

#define SC_Add           42
```

系统调用号

```
/* Note: Create does not open the file. */
/* Return 1 on success, negative error code on failure */
int Create(char *name);

/* Remove a Nachos file, with name "name" */
int Remove(char *name);

/* Open the Nachos file "name", and return an "OpenFileId" that can
 * be used to read and write to the file.
 */
OpenFileId Open(char *name);

/* Write "size" bytes from "buffer" to the open file.
 * Return the number of bytes actually read on success.
 * On failure, a negative error code is returned.
 */
int Write(char *buffer, int size, OpenFileId id);

/* Read "size" bytes from the open file into "buffer".
 * Return the number of bytes actually read -- if the open file isn't
 * long enough, or if it is an I/O device, and there aren't enough
 * characters to read, return whatever is available (for I/O devices,
 * you should always wait until you can return at least one character).
 */
int Read(char *buffer, int size, OpenFileId id);
```

系统调用接口

– Write系统调用

- Nachos内核函数接口：Write(char *buffer, int size, OpenFileId id)
 - 含义：向指定的文件中写入大小为size，存储在buffer中的数据
- 系统调用提示：
 - 从内存中读取数据，并将其写入到文件中

– Read系统调用

- Nachos内核函数接口：Read(char *buffer, int size, OpenFileId id)
 - 含义：从指定的文件中读取大小为size 的数据，并将数据写入到buffer中
- 系统调用提示：
 - 从文件中读取数据，并将其写入到内存中

实现演示

演示系统调用Write的实现
过程.....

实验提示-添加Write系统调用



中国科学技术大学
University of Science and Technology of China

```
/code/test/shell.c
#include "syscall.h"
int main()
{
    .....
    Write(prompt, 2, output);
    .....
}
```

```
/code/userprog/syscall.h
#define SC_Write 8
int Write(char *buffer, int
size, OpenFileId id);
```

/code/test/start.s

```
#include "syscall.h"
start:
    jal main
    move $4,$0
    jal Exit
    .end __start

Write:
    addiu $2,$0,SC_Write
    syscall
    j $31
    .end Write
```

execution starts
from here

**/code/userprog/
Exception.cc**

```
case SC_Write:
    result = SysWrite();
```

**/code/userprog/
ksyscall.h**

```
int SysWrite(){}
```

generate a hardware
trap into nachos kernel

```
RaiseException()
/code/machine/
mipssim.cc
```

目录



中国科学技术大学
University of Science and Technology of China

- **Nachos介绍**
 - 什么是**Nachos**
 - **Nachos**体系结构
 - **Nachos**工作模式
- **Nachos安装**
 - 实验平台搭建
 - **Nachos**编译安装
 - 源码阅读工具
- **实验内容**
 - 系统调用概述
 - 实验要求
 - 实验提示
 - **测试程序的编写与运行**

- 编译：
 - code/build.linux 目录下进行nachos内核的编译： *make*（或 *make clean, 再make*）
 - code/test目录下进行用户测试程序的交叉编译： *make*，产生 nachos用户态可执行程序sub
- 测试运行
 - code/test目录下运行 `../build.linux/nachos -x sub`
 - 或者code/build.linux 目录下运行： `./nachos -x ../test/ sub`

```
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/test$ ../build.linux/nachos -x sub
42-23=19
Machine halting!

Ticks: total 29, idle 0, system 10, user 19
Disk I/O: reads 0, writes 0
Console I/O: reads 0, writes 0
Paging: faults 0
Network I/O: packets received 0, sent 0
zq@ubuntu:~/zq/nachos/NachOS-4.0/code/test$
```



THE END...

THANK YOU~