山东大学 软件 学院

**操作系统课程设计** 实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202000300125 | 姓名：贾星宇 | | 班级：计软20.5 |
| 实验编号：Lab1 | | | |
| 实验题目：Nachos实验环境准备、安装与源码分析 | | | |
| 实验学时：5 | | 实验日期：2022.11.05 | |
| 实验目的：  1. Nachos开发环境的安装测试(不含Linux系统本身及虚拟机软件的安装测试)。  2. Nachos实验代码框架(源码目录)的基本分析。  3. Nachos Makefile的基本分析。  4. 硬件机制模拟部分的实现原理分析，包括中断、时钟、CPU指令执行。 | | | |
| 硬件环境：  HUAWEI matebook14 2020笔记本  Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.11 GHz CPU  8GB内存  512GB SSD | | | |
| 软件环境：  宿主机：Windows 10 21H2 64位  虚拟机软件：VMware Workstation Pro 16.2.4 build-17966106  Linux：Ubuntu 14.04.6 LTS Desktop i386 (Trusty Tahr)  gcc/g++：(Ubuntu 4.8.4-2ubuntu1~14.04.4) 4.8.4  MIPS交叉编译器：gcc-2.8.1-mips.tar.gz  Nachos：Nachos-3.4-UALR-2022 | | | |
| 实验步骤与内容：   1. 开发工具安装过程：   （1）安装虚拟机所需要的环境vmvare workstation：由于旧版本的vmvare无法支持网盘中的虚拟机文件，因此更新虚拟机到16.2.4版本    （图：vmvare workstation 16 pro）  （2）在vmvare中打开网盘中下载的文件，并成功启动虚拟机。用户名为u1，密码为u1，联网进行安装。      （图：安装并启动虚拟机）   1. Nachos安装过程：   打开terminal，并输入如下指令：   1. 更新Ubuntu的源：sudo apt update      1. 安装g++：sudo apt install g++，由于先前已安装测试，因此此次安装仅检查是否更新：      1. 安装用于MIPS的交叉编译器：   复制压缩包gcc-2.8.1-mips.tar.gz到主目录，并进行解压：     1. 安装Nachos：创建文件夹oscp，并将压缩包**nachos-3.4-ualr-2022.tar.gz**复制到此文件夹中并进行解压，由于先前解压过，展示解压后的文件：      1. 测试Nachos threads：进入nachos的threads文件夹，进行clean后make：       运行nachos：     1. 生成实用程序coff2noff和coff2flat：       发现已生成：     1. 生成能加在用户进程的nachos：   cd ../userprog  make clean  make    (8) 生成用户mips程序halt、matmult、shell：  cd ../test  make clean  make  ../userprog/nachos -x halt.noff     1. Nachos实验代码框架（源码目录）的基本分析   Nachos本身作为Linux宿主主机的一个进程，包含一个MIPS虚拟机用于执行用户程序，且用户程序为c程序，它通过交叉编译后生成mips可运行程序。其中，nachos kernel负责用户程序的装在和运行。  分析Nachos文件夹：    其中c++example文件夹存放了有关C++的介绍和实例，code文件夹存放了各部分代码，包含了MIPS可执行文件转换实用工具。Doc文件夹为原作者的简单介绍和原有的作业要求。  重点关注Code文件夹：    其中Makeifle.dep为平台相关文件，包括x86平台，为各个Makefile所共用。  Makefile.common为各个Makefile所共用的文件部分。  bin文件夹为用户MIPS程序目标码变换的实用程序及源码  filesys文件夹为文件系统管理部分的代码  machine文件夹为MIPS虚拟机模拟代码  network为网络管理部分代码  test为一些测试用MIPS用户应用程序  threads为线程管理部分的代码  userprog为用户程序部分的代码  demo0为用于演示Makefile的例子  demo1为使用信号量解决生产者消费者同步问题的例子  lab2~lab7为一些待完成的实验2~7代码，放在对应文件夹内，包含了相应的Makefile  其次，还有C++example和doc文件夹如下：      补充  Lab2-lab7不必多说，主要目的是为了做到保存原有的系统代码，将实验过程中需要修改的文件和类放到相应的文件夹下，做到隔离的作用，防止将系统因为意外情况和误操作而改崩。  Demo0和demo1分别描述了线程调度和生产者消费者问题，用类似于入门指引的操作指导着我完成实验一和实验二，同时也进一步了解nachos系统的原理和实验的基本过程。  Filesys主要进行一些与文件系统相关的操作，实验四、五的相关内容。nachos的初始文件系统是单索引文件系统，其中初始扇区代表比特图和文件目录，表现为BitMap和Directory。其余扇区存储文件。文件的大小取决于文件头索引能够索引的大小，因此才有了添加二级索引来扩展文件系统的说法。其中每个文件由文件头和文件体构成，文件头中存储文件基本信息，包括文件体各扇区索引，表现为类FileHeader。一整个文件的表达方式为OpenFile，包括了读、写和写回等操作，属性除了文件头之外，还有当前文件指针指向的位置。文件系统中的synchdisk负责较为底层的一些操作。  Machine文件夹负责MIPS虚拟机、指令执行相关的操作，如interrupt类负责中断的相关操作，如exec、halt，而translate则负责物理页和虚拟页之间的转换表等操作。  Threads文件夹负责了线程相关操作，也就是实验二。如描述一个线程属性及操作的thread，存储线程的列表list、用于线程调度的scheduler等。  Userprog文件夹主要负责用户进程的相关操作，也就是实验六、七。在addrspace中主要负责为用户进程分配地址空间，也就是创建一个合理的页表，这也是实验部分我们重点需要修改的地方；而exception文件则负责处理异常，负责将进程由用户态转为核心态执行；如果要添加系统调用，我们需要相应修改syscal和start.s，同时progtest文件提供了运行用户线程的基本操作方式，供我们参考实现。   1. Nachos Makefile基本分析 2. 基本知识：gcc是GNU推出的多平台编译器，可以将C、C++语言源程序、汇编程序和目标程序编译、链接成可执行文件。gcc执行过程：预处理—编译—汇编—链接。 3. Makefile：实现通过make命令进行自动化编译，makefile文件中主要包含目标文件、依赖文件以及command产生过程。输入make命令后，会在当前目录下寻找makefile文件，找到后找文件中第一个目标文件，并将文件作为最终目标文件。如果目标文件不存在，则执行后面的命令生成此文件。 4. Nachos的makefile结构：      1. Nachos硬件机制模拟部分的实现原理分析，包括中断、时钟、CPU指令执行等： 2. 中断模块：主要作用是模拟底层中断机制，通过该模块启动和禁止中断。文件为interrupt.cc interrupt.h。中断等待队列是Nachos虚拟机中最重要的数据结构之一，它记录了当前虚拟机可以预测的将在未来发生的所有中断。当系统进行了某种操作可能引起未来发生的中断时，系统会将终端插入到中断等待队列中。对于一些定期需要发生的中断，系统会在中断处理后将下一次要发生的中断插入到中断等待队列中。先发生的中断将得到优先的处理。中断发生时调用中断处理程序，作用包括在现有的模拟硬件基础上建立更高层次的抽象。     （图：interupt.cc）   1. 时钟中断模块：模拟时钟中断，每隔一定时间发生一次时钟中断。时钟中断可以进行时间片轮转法的时钟控制、分时系统线程优先级的计算以及线程进入睡眠状态时的时间计算。Nachos利用其模拟的中断机制来模拟时钟中断，间隔由TimerTicks宏决定。     （图：timer.cc）  (3) CPU指令执行：Nachos内部模拟了MIPS指令集，每条指令封装成一个MIPS对象，然后从取指到执行的阶段封装成一个instruction对象。处理器为processor对象，用byte数组模拟内存，int数组模拟寄存器组，共38个，包括一些特殊功能的寄存器。执行指令的过程即 使PC寄存器不断改变，然后取指、解析、执行的过程，通过捕捉异常来执行系统调用和其他中断。    （图：mipssim.cc）   1. 关键代码段落和测试运行结果   Nachos是用C++语言中的类来表示各个对象的，其中Machine类用来模拟计算机主机，提供读写寄存器、读写主存、读写汇编指令等功能。    （图：machine.cc）    （图：machine类结构）  Interrupt类用来模拟硬件中断系统，中断状态有开和关两种，终端类型有时钟中断、磁盘中断、控制台写中断、控制台读中断、网络发送中断以及网络接收中断。在类中有一个记录即将发生中断的中断等待队列，每个等待处理的中断包含中断类型、中断处理程序的地址及参数、中断应当发生的时间等信息。    （图：Interrupt数据类型）  Console类模拟控制台设备，当用户程序向控制台写一个字符时，写程序立即返回，过了给定的时钟周期后I/O操作完成，控制台向CPU发送一个控制台写中断。但是控制台是否有用户输入可供读取是随机的，因此控制台每隔给定的时钟周期向CPU发送一个控制台读中断，周期性地发中断的方法与定时器类似。读中断发生后，如果有用户输入的话，控制台读中断处理过程将控制台输入的字符放入字符缓冲区。    （图：console数据类型） | | | |
| 结论分析与体会：  Nachos是建立在一个软件模拟的虚拟机上的教学实践平台，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需的硬件系统。  RISC指令集指令数目较少且较常用，不必编写编译器，可直接用C/C++语言编写程序。这样降低了开发难度。  其实，许多现代的操作系统都是这样，先用软件模拟硬件，在软件上建立操作系统并调试，随后在真正的硬件上调试。  Nachos也使用了操作系统中的一些新的概念，如网络、线程和分布式应用。Nachos以线程作为一个基本概念讲述。  Nachos确定性调试比较方便，在同样的输入顺序、输入参数的情况下，Nachos的运行结构是完全一样的。  Nachos简单且易于扩展，是让读者在一个框架下发挥自己的创造性来进行扩展。 | | | |