操作系统课程设计调研报告

主要目标

网络部分

在上学期完成的MIPS ucore的基础上, 加入网络扩展功能, 能够实现简单的TCP协议, 可以与远端机器有线连接并传输文件.

具体的:

- 完成以太网接口的驱动;
- 完成在内核态实现的FTP协议，使得可以进行文件传输(12周以前)
- 尝试进行进一步的扩展: 在内核态或用户态实现其它TCP协议, 如HTTP/SMTP等(12周以后及暑期)

GDB远程调试接口部分

在上学期MIPS ucore和thinpad的基础上实现调试功能, 主要是:

- 实现ucore的汇编级调试
- 实现ucore的源码级调试(可选)
- 实现用户态的调试(可选)

文件系统部分

以上学期使用的MIPS ucore作为基础, 移植本学期使用的X86 ucore的lab1到lab8, 但经过调研之后, 具体的工作主要有以下两条:

- 完成FLASH驱动, 将当前的文件系统搬到FLASH上.
- 完成lab1中kern panic下的调试功能.

调研情况

网络部分

接口驱动

ThinPAD硬件部分含有一个DM9000以太网芯片, 之前在计算机组成原理实验中, 我们并没有实现这个芯片的功能以及所需的驱动, 因此在本实验中需要实现.

这个接口及其驱动的实现方法详见《GDB远程调试接口暨硬件调研设计报告》的以太网芯片部分.

网络标准：TCP/IP协议栈

经过调研, 我们决定按照TCP/IP的标准来实现本实验中的网络功能。

调试部分

串口

解决跨时钟域同步即可, 已做过实验.

DM9000AEP以太网芯片

在调研前, 先重新温习了一些基础的网络原理知识. 然后找到的芯片的datasheet, 通过阅读芯片Feature, 内部布图, 接口说明, 时序等理解了如何将DM9000AEP与CPU对接.

实际上, DM9000AEP很好的实现了物理层以及数据链路层的算法. 为了控制芯片的运行, 只需要往芯片中存在几十个寄存器写数据就可以了. 具体来说, 第一个写入操作必须是需要操作的寄存器8位地址, 第二次可以选择访问或写入这个寄存器的值.

读写地址和读写数据的时序是一样的.

GDB调试部分

这个部分的调研是难度最大的. 首先通过一些讲解嵌入式系统远程调试和ucore实验指导书了解了gdb远程调试的基本概念. 然后通过一些文章了解到了GDB远程调试接口——RSP的内容. 通过了解gdb调试信息的编译过程, 比如stabs格式等也加深了对于gdb的理解. 还通过kgdb了解到linux中的调试stub. 之后, 就把工作重心放在GDB的线程级, 进程级调试的学习上, 需要相应的指令达到目标. 通过查阅RSP的manual, 可以知道这个功能可以通过RSP的扩展, 异步调试包实现.

简而言之, 只要在硬件中提供一些相对简单的功能, 比如读写寄存器, 读写内存, 设置硬断点等, 并提供一个, 就可以实现GDB的汇编级调试. 同时, gdb可以直接加载host机上kernel得到符号表信息, 这样应该是能实现源码级调试的. 并且通过在操统的调试stub上支持对于进程, 线程的操作, 有望实现用户态代码调试.

文件系统部分

lab部分对比

由于文件系统部分的任务主要是将当前X86 ucore的几个实验移植到MIPS ucore上, 因此, 这部分的调研方式主要是将现有的MIPS ucore与X86 ucore在几个lab的完成度上进行对比, 结果如下:

lab1

lab1中其中一个任务是要求完成中断机制, 这在MIPS ucore中已经完成. 但另一个任务, 在kernel panic状态下实现简单的debug命令, 包括kerninfo和backtrace就没有被实现, 因此我们需要在MIPS ucore当中重新加入这一功能.

lab2

lab2是关于连续物理地址分配的实验, 在MIPS ucore当中已经以buddy system的形式进行了实现.

lab3

lab3是关于虚拟内存管理的实验, 但在MIPS ucore当中, 并没有启动虚存, 因此也没有所谓的页面置换算法.

lab4 lab5 lab6

这三个实验都与进程的调度有关, MIPS ucore中已经实现的很好.

lab7

这一实验是关于进程间通信的, 在MIPS ucore当中只实现了信号量, 但没有实现管程. 但管程事实上只是信号量的一种封装, 如无必要, 不会实现.

lab8

在MIPS ucore中已经完全地移植了现在X86 ucore当中的文件系统, 这是调研之前没有预料到的, 因此我们在文件系统方面的目标现在稍有改变.

现有的文件系统的局限

当前ucore中的文件系统是simple file system, 它主要有两个问题, 一是文件的大小有限制, 不能超过4MB+48KB, 但由于文件系统只是一个教学操作系统, 因此暂时不考虑大文件的储存问题, 当另一个问题就比较严重, 现有的文件系统并不能支持文件的创建和删除, 这对lab8的完成不会造成任何影响, 但会影响到远程文件的传输.

另外, 现有的文件系统是建立在RAM上的, 我们希望通过实现FLASH驱动, 将文件系统从RAM转移到FLASH上.

FLASH及其驱动调研

FLASH操作非常友好, 软件上的驱动可以模仿串口驱动, 只是硬件上在MMU当中需要约定地址映射. thinpad FLASH的资料在上学期计原实验指导书中可以查到. Flash和Ram类似, 就是和CPU协调跑一个状态机. 不同之处在于, 在真正操作之前需要通过写入操作码选择操作模式, 另外, 写入之前必须先擦除.

初步设计部分

网络部分

经过调研, 我们决定按照TCP/IP的标准来实现本实验中的网络功能.

网络接口层

最底层, 直接与硬件交互. 在实现了以太网接口的驱动后, 需要实现ARP协议, 交换两端的MAC地址.

网络层

需要实现IP协议, 定义网络数据包的格式, 将数据封装成数据包, 并实现数据包的发送和接收.

传输层

通过传输协议实现主机之间的通信会话, 传输协议一般有TCP和UDP两种, 我们只需要实现TCP.

应用层

最高层, 直接访问网络, 实现网络功能. 我们需要实现的网络应用有FTP(用于传输文件), 以及HTTP/SMTP(扩展).

调试部分

DM9000AEP以太网芯片

给CPU新增模块使其能往以太网芯片中读写数据, 处理好读写逻辑, 并修改MMU映射关系即可。

由于datasheet完整阅读工作量过大, 并且手写驱动需要对于以太网数据链路层的理解和较好的实现网络层协议的基础, 和高层对接的关键是找到一个可以使用的DM9000AEP芯片驱动.

串口

使用合理的跨时钟域同步实现一个鲁棒的串口, 具体会使用边沿同步和脉冲同步. 可以考虑在超过115200的波特率下运行.

GDB调试

目标定位为实现GDB的汇编级调试. 为了达成这个目标, 首先需要支持GDB控制的读写寄存器, 读写内存, 设置硬断点等操作. 其次需要有能和RSP协议交互的模块, 这个模块一般来说放在操统中实现比较简单. 但我们的目标是尽量不对操统进行修改就达到调试的目的, 可选的方案有硬件解析, 以及通过写入rom中的机器码指令完成RSP协议交互. 选择后者, 但具体细节有待考虑.

之前也提到源码级调试和用户态代码调试, 暂时认为源码级调试可能只需要加载host机上的符号表信息, 而用户态代码调试较为复杂, 需要在操统上进行一定的修改. 这是可选项目.

文件系统部分

文件系统针对远程文件传输的解决方案

我们评估了如果在当前系统当中加入文件系统的工作量. 由于文件系统在本质上可以看成是一个数据库, 添加和删除必然涉及到块的分配和释放, 这个过程的处理可能会比较繁琐. 因此对于远程传输, 我们会采取在ucore编译阶段加入空文件的方法, 收到远端字节流的时候, 将字节流写入到文件当中即可.

FLASH 读写

模仿串口驱动在操作系统中实现一个FLASH驱动, 并在dev层取代ramdisk. 硬件上需要修改MMU, 对地址映射做一些约定.

小组分工

- 网络部分: 梁泽宇 董豪宇
- GDB部分: 李宇轩 董豪宇
- 文件系统部分: 董豪宇

现在已有的参考资料

- 《计算机硬件系统实验教程》
- DM9000AEP datasheet
- ucore实验指导书gitbook qemu+gdb调试部分
- gdb debug 信息 stabs 格式

- gdb调试多进程
- Linux内核驱动开发之KGDB原理介绍及kgdboe方式配置
- Howto: GDB Remote Serial Protocol
- X86 ucore 源码
- ucore plus 源码