Lab 0实验过程的一些知识点的记录

ls命令：列出文件

cd命令：切换到某文件夹

pwd命令：显示当前目录

mkdir命令：创建目录

gcc-v:查看gcc版本

gdb:进入gdb,输入quit，退出

到labcodes\_answer/lab1\_result文件夹下，先make clean,然后make就可以重新编译了。

在这时,bin目录下会有kernel这样一个文件，利用file bin/kernel命令查看该文件的属性，可以发现该文件是一个可执行文件，它可以把我们的bootloader给加载进去。而bootloader在bin目录下的ucore.img镜像文件之中，这个磁盘镜像包含了bootloader以及ucore的kernel。

执行make qemu就执行了刚才的镜像文件，通过串口进行输出到屏幕上。按ctrl+c就可以终止刚才程序的执行。

以上是用字符方式执行的，我们也可以用图形化的界面，Eclipse CDT环境

先Clean project，再build project，可以看到这个项目就编译完成了

对该项目进行Debug的方法：点击debug下的debug configuration，到zylin native那项，debugger的stop on startup项关掉

在commands栏initialize commands里敲入命令：

target remote :1234 //用于完成对qemu的连接

file /home/moocos/moocos/ucore\_lab/labcodes\_answer/lab1\_result/obj/bootblock.o

//加载bootloader符号，即bootloader的主体执行程序

break bootmain //在bootmain所对应的虚拟地址里面设置了一个断点

#file /home/moocos/moocos/ucore\_lab/labcodes\_answer/lab1\_result/bin/kernel

#break kern\_init

#break print\_kerninfo

Run commands里填：continue

在debugger的gdb debugger中键入gdb(原来是arm-elf-gdb)

点击带个红点的运行按钮，下面有个External tools configuration:

location里填的是：/usr/bin/make

Arguments处填的是：gdb

Working directory处填的是：/home/moocos/moocos/ucore\_lab/labcodes\_answer/lab1\_result

点击红点运行下的lab1\_ans，可以看到与qemu建立了联系。

然后双击bootmain.c文件的88行，设置断点

然后，首次对该项目debug，要选择debug按钮下的debug configuration中的debug以后该项目就自动出现在debug列表中了，这时就对该项目进行debug的操作了。

然后点击Step over按钮开始Debug

这里注意，如果我们要调试ucore,为此我们需要把ucore的符号信息加载进来，这时我们进入console里面键入file bin/kernel

这里面有一个起止函数kern\_init，这是ucore最开始跑的一个c程序的函数，我们在console中设置中断：

break kern\_init

这时将提示：

Breakpoint 3 at 0x100000: file kern/init/init.c, line 17.

这时点击按钮step return,然后点击step over以继续运行程序，可以发现此时便跑完了bootloader的工作，同时把控制权交给了ucore另外一个程序：程序跳到了刚才设置的kern/init/init.c文件的kern\_init函数处

此时再尝试单步运行几次，然后按resume，使得程序正常运行。

输入meld lab1\_result/ lab2\_result/ 可以比较两次实验中代码的区别。

可以用understand工具更好的分析代码