嵌入式操作系统

4 SkyEye 简介

陈香兰（xlanchen@ustc.edu.cn）

计算机应用教研室 @ 计算机学院
嵌入式系统实验室 @ 苏州研究院
中国科学技术大学

/media/SAMSUNG/work/6 实验室相关

Outline

SkyEye 简介 I

- SkyEye is an **Open Source Software Project** (GPL Licence).
  - Origin from **GDB/Armulator**,
  - Skyeye 的网站：
    - http://www.skyeye.org/index.shtml
- SkyEye 的起源和发展
  - **陈渝：** 做一个用软件实现的**嵌入式开发硬件模拟器**，可以在模拟器上运行各种操作系统，这样就可以在没有开发板的情况下学习和研究操作系统

SkyEye 简介 II

- SkyEye 的目标：
  to provide an integrated simulation environment in Linux and Windows,
  simulates/emulates typical Embedded Computer Systems
- Now the following OS and system softwares can run in SkyEye:
  - uC/OS-II-2.x with network support
  - uClinux based on Linux2.4.x/2.6.x with Network/LCD/TouchScreen/Flash Mem support
  - ARM Linux 2.4.x/2.6.x with Network/LCD/TouchScreen/Flash Mem support
  - Nucleus
  - Rtems
  - Ecos

SkyEye 简介 III

- lwIP on uC/OS-II
- applications on uC/OSII, uClinux,ARM Linux

可对上述软件系统进行源码级的分析、调试和测试。

SkyEye 简介 IV

Processor List supported by SkyEye

Processor name	*Core*	*Architecture*	*Current status*	*Running OS*
S3C4510	ARM7TDMI	ARM	stable	uClinux
S3C44B0	ARM7TDMI	ARM	stable	uClinux
AT91	ARM7TDMI	ARM	stable	uClinux
S3C3410	ARM7TDMI	ARM	stable	uClinux
LPC2210	ARM7TDMI	ARM	not finished	uClinux
EP7312	ATM720T	ARM	stable	linux
Ep9312	ARM9	ARM	stable	linux
AT91RM9200	ARM9	ARM	stable	linux
S3C2410	ARM9	ARM	stable	linux
CS89712	ARM9	ARM	stable	linux
SA1100	StrongArm	ARM	stable	linux
PXA25x	XSCALE	ARM	stable	linux
PXA27x	XSCALE	ARM	stable	linux
bf533	bf53x	Blackfin	testing	uClinux
bf537	bf53x	Blackfin	testing	uClinux
CF5249		Coldfire	stable	uClinux
CF5272		Coldfire	testing	uClinux
Au1100	R4K	MIPS	not finished	linux
MPC8560	E500	PowerPC	testing	linux
MPC8572	E500	PowerPC	testing	linux
leon2	Sparc v8	Sparc	testing	RTEMS

SkyEye 模拟硬件介绍 I

- Now the following hardwares can be simulated by SkyEye:
  - **CPU CORE**: ARM7TDMI, ARM720T,StrongARM, XScale, Blackfin
  - **APPLICATION CPU**: Atmel AT91X40/AT91RM9200,Cirrus CIRRUS LOGIC EP7312/EP9312 CS89712,Intel SA1100/SA1110, Intel PXA 25x/27x, Samsung 4510B/44B0/2410/2440 , Sharp LH7xxxx, NS9750, Philips LPC22xx, BF533
  - **MEMORY**: RAM, ROM, Flash
  - **Peripheral**: Timer, UART,NIC chip,LCD, TouchScreen, etc.

SkyEye 模拟硬件介绍 II

存储器管理单元和缓存单元

- MMU
  Memory Management Unit，
  存储器管理单元，
  是用来管理虚拟内存系统的硬件。
  - MMU 的两个主要功能是：
    1. 将虚地址转换成物理地址；
    2. 控制存储器的存取权限。
  - MMU 关掉时，虚地址直接输出到物理地址总线

SkyEye 模拟硬件介绍 III

- TLB，
  Translation Lookaside Buffers
  在 MMU 中，存放从虚拟地址到物理地址的匹配表
  - 保存的内容包括：
    虚址及其对应的物理地址，权限，域和映射类型。
  - 当 CPU 对一虚拟地址进行存取时，
    首先搜索 TLB 表以查找对应的物理地址等信息，
    如果没有查到，则进行查找 translation table，称为 Translation Table Walk（简称 TTW）。经过 TTW 过程后，将查到的信息保存到 TLB。然后根据 TLB 表项的物理地址进行读写。

SkyEye 模拟硬件介绍 IV

- **CACHE，缓存单元**
  主要用于缓存内存中的数据，其读写速度远快于内存的读写速度，所以可以提高 CPU 的内存数据的访问效率。
- **write/read buffer 硬件单元**
  write/read buffer 硬件单元的作用与 CACHE 的作用类似。
- MMU、CACHE、write/read buffer 一般是高性能 CPU 的重要组成部分，且不同类型 CPU 的 MMU、CACHE、write/read buffer 的逻辑行为也有一定的差异。为了支持模拟多种类型 CPU 的 MMU/CACHE，SkyEye 包含了一个通用的 MMU/CACHE 模拟实现。通过对一些参数的调整可以支持模拟多种类型的 MMU/CACHE 物理结构和逻辑行为。

网络芯片

SkyEye 模拟硬件介绍 V

- 目前 SkyEye 模拟了网络芯片 8019AS，
- 其特点是：NE2000 兼容，内建 16KRAM 缓冲区，10MB 传输速率。
- 虽然目前模拟的开发板上不一定有网络芯片 8019AS，但可以在模拟的开发板上加上网络芯片 8019AS 的模拟。

Outline

SkyEye 的安装

- Linux 操作系统
- Windows+MingW 或 cygwin

在 Kubuntu 上安装 SkyEye

- 主机环境

```
xlanchen@xlanchen-desktop:~$ uname -a
Linux xlanchen-desktop 2.6.28-15-generic #51-Ubuntu SMP
Mon Aug 31 13:33:16 UTC 2009 i686 GNU/Linux
xlanchen@xlanchen-desktop:~$
```

  - uname 命令用于打印系统信息
- 编译器
  - 一开始，主机上只有自带的 gcc，版本 4.3.3
- 在 Kubuntu 上安装 SkyEye 有两种安装方法
  - 下载源代码，编译并安装
  - 直接安装二进制代码
  - sudo apt-get install skyeye
    - 版本：1.2.3

Outline

下载 skyeye 源代码，编译并安装 I

- 最新版本情况
  - skyeye-1.2.8_rc1.tar.gz
  - skyeye-testsuite-1.2.8.tar.gz
- 下载 skyeye-1.2.8_rc1.tar.gz
- 解压缩

  tar zvxf skyeye-1.2.8_rc1.tar.gz

为什么不使用主机自带的 gcc?

下载 skyeye 源代码，编译并安装 II

- 为编译 1.2.8 的 skyeye 准备编译环境

```
5 Tested Environment
X86-32:
        CygWin/MinGW
        gcc-3.4, gcc-4.1.2

        Linux (Debian)
        gcc-3.3 gcc-3.4
        gcc-2.95 gcc-4.0 (doesn't support DBCT)

        BeOS (R5.0.3, Exp-Dano)
        gcc-2.95 + gcc-3.4.3
```

- 安装 gcc-3.4

  sudo apt-get install gcc-3.4

- 可能需要安装：build-essential 和 texinfo

下载 skyeye 源代码，编译并安装 III

- 简单编译 1.2.8

  ./configure CC=gcc-3.4

  make

```
xlanchen@xlanchen-desktop:~/workspace/skyeye-1.2.8_rc1$ ls
aclocal.m4       [config.h].in  dbct          Makefile      REPORTING-BUGS
arch             config.h.in    depcomp       Makefile.am   skyeye
AUTHORS          config.log     device        Makefile.in   skyeye.o
autom4te.cache   config.status  INSTALL       memory        stamp-h1
binary           config.sub     install-sh    misc          third-party
ChangeLog        configure      LICENSE       missing       TODO
config.guess     configure.in   ltmain.sh     NEWS          utils
config.h         COPYING        MAINTAINERS   README
xlanchen@xlanchen-desktop:~/workspace/skyeye-1.2.8_rc1$
```

下载 skyeye 源代码，编译并安装 IV

- 安装

  sudo make install

[可选，可能出错，报 mkinstalldirs 没有，不安装或者寻找一个 mkinstalldirs 脚本拷贝到 third-party 目录下]

- 判断编译 / 安装是否成功
  - 下载测试集
    最新的 skyeye-testsuits，版本为 1.2.8
    skyeye-testsuite-1.2.8.tar.gz
  - 解压缩：

    tar zfxf skyeye-testsuite-1.2.8.tar.gz

下载 skyeye 源代码，编译并安装 V

- 进入到 linux 目录，修改符号连接 **skyeye**指向编译好的 skyeye

  rm skyeye
  ln -s ../../skyeye-1.2.8_rc1/skyeye skyeye

- 运行如下命令：

  ./exec_skyeye.sh *s3c2410/s3c2410x-2.6.14/ ../../*
  或者运行下列命令
  cd s3c2410/s3c2410x-2.6.14/
  ../../skyeye -c skyeye.conf -e vmlinux

下载 skyeye 源代码，编译并安装 VI

- 注意
  由于前面对 skyeye 的编译是缺省编译，在运行 testsuite 下的例子时，有些可能会出错。

Outline

在 Linux 的根文件系统中添加 hello I

1. 准备交叉编译环境
   - 下载**3.4.1**的交叉编译器到你的工作目录 $WDIR
     - arm-linux-gcc-3.4.1.tar.bz2
   - 在工作目录 $WDIR 中解压缩（$WDIR 也可以是根目录）

     tar jvxf arm-linux-gcc-3.4.1.tar.bz2
   - 查看是否 ok

```
xlanchen@xlanchen-desktop:~/09FallEOS$ usr/local/arm/3.4.1/bin/arm-linux-gcc -v
Reading specs from /home/xlanchen/09FallEOS/usr/local/arm/3.4.1/bin/../lib/gcc/a
rm-linux/3.4.1/specs
Configured with: /work/crosstool-0.27/build/arm-linux/gcc-3.4.1-glibc-2.3.2/gcc-
3.4.1/configure --target=arm-linux --host=i686-host_pc-linux-gnu --prefix=/usr/l
ocal/arm/3.4.1 --with-headers=/usr/local/arm/3.4.1/arm-linux/include --with-loca
l-prefix=/usr/local/arm/3.4.1/arm-linux --disable-nls --enable-threads=posix --e
nable-symvers=gnu --enable-__cxa_atexit --enable-languages=c,c++ --enable-shared
 --enable-c99 --enable-long-long
Thread model: posix
gcc version 3.4.1
xlanchen@xlanchen-desktop:~/09FallEOS$
```

在 Linux 的根文件系统中添加 hello II

2. 编写简单的 hello.c，并编译

```
1 #include <stdio.h>
2
3 int main(void){
4         printf("HELLO WORLD!\n");
5         return 0;
6 }
```

- 使用 3.4.1 版本的 arm-linux 交叉编译器，静态编译得到 elf 格式的可执行文件：

  $WDIR/usr/local/arm/3.4.1/bin/arm-linux-gcc -static -o hello hello.c

```
xlanchen@xlanchen-desktop:~/09FallEOS/skyeye-testsuite-1.2.8/linu
x/s3c2410/s3c2410x-2.6.14$ file hello
hello: ELF 32-bit LSB executable, ARM, version 1, statically link
ed, for GNU/Linux 2.4.3, not stripped
xlanchen@xlanchen-desktop:~/09FallEOS/skyeye-testsuite-1.2.8/linu
x/s3c2410/s3c2410x-2.6.14$
```

在 Linux 的根文件系统中添加 hello III

3. 将 hello 复制到 linux 的根文件系统映像 initrd.img 中
   - 挂载根文件系统映像

     mkdir root

     sudo mount -o loop initrd.img root

   - 将 hello 拷贝到根文件系统中

     sudo cp hello root/bin

   - 卸载根文件系统映像

     sudo umount root

在 Linux 的根文件系统中添加 hello IV

4. 使用 skyeye 启动 linux，运行 hello

   skyeye -c skyeye.conf -e vmlinux

   - 进入 armlinux 之后，进入 bin 目录，运行 hello

在 Linux 的根文件系统中添加 hello V

```
Welcome to

 _      _____      __   __    _      _
/ \    /  __ \    /  \_/  \  | |    |_|
 / _ \   | |  | |  / /\   /\ \ | |      _ ____  _   _  _  _
/ /_\ \  | |__| | / /  \_/  \ \| |     | |  _ \| | | |\ \/ /
/ /___\ \ | |__\ \ | |       | || |___ | | |_| | |_| |/    \
/_/     \_\| |   \_\|_|       |_||_____||_|_| |_|\____|\_/\_/

ARMLinux for Skyeye
For further information please check:
http://www.skyeye.org/



BusyBox v1.4.1 (2007-02-10 01:19:06 CST) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

/bin/ash: can't access tty; job control turned off
/ $ cd bin
/bin $ hello
HELLO WORLD!
/bin $
```

Outline

命令行安装 skyeye

- 使用命令行安装
  - sudo apt-get install skyeye
- 使用 sk<tab> 可以看到：

```
xlanchen@xlanchen-desktop:~$ sk
skill   skyeye
xlanchen@xlanchen-desktop:~$ sk
```

启动 uclinux I

- 仍然使用 skyeye-testsuite-1.2.8
- 进入 uClinux 目录
- 进入 at91/uclinux_cs8900a/ 目录
- 运行

skyeye -c skyeye.conf -e linux

启动 uclinux II

```
Welcome to

          ____ _   _
         /  __| ||_|
 _    _| |   | | _ ____  _    _  _  _
| | | | |   | || |  _ \| | | |\ \/ /
| |_| | |__| || | | | | |_| |/    \
|  ___\____|_||_|_| |_|\____|\_/\_/
| |
|_|

GDB/ARMulator support by <davidm@snapgear.com>
For further information check:
http://www.uclinux.org/

Command: /bin/ifconfig eth0 up 10.0.0.2
SIOCGIFFLAGS: No such device
SIOCSIFADDR: No such device
pid 11: failed 512
Execution Finished, Exiting
init: Booting to single user mode

Sash command shell (version 1.1.1)
/>
```

Outline

添加 hello 到 μCLinux 的根文件系统中 I

- hello 的源代码同上

```
1 #include <stdio.h>
2
3 int main(void){
4         printf("HELLO WORLD!\n");
5         return 0;
6 }
```

- 对于 μCLinux，使用 arm-elf-工具链
  - 下载 arm-elf-tools-20030314.sh
  - 运行./arm-elf-tools-20030314.sh 安装
  - 使用 arm-elf-<tab> 查看是否安装成功

添加 hello 到 μCLinux 的根文件系统中 II

- 编译 hello

  arm-elf-gcc -elf2flt -o hello hello.c

```
xlanchen@xlanchen-desktop:~/09FallEOS/skyeye-testsuite-1.2.8/uC
linux/at91/uclinux_cs8900a$ file hello
hello: BFLT executable - version 4 ram
xlanchen@xlanchen-desktop:~/09FallEOS/skyeye-testsuite-1.2.8/uC
linux/at91/uclinux_cs8900a$
```

- 由于 romfs.img 不同于 initrd.img，需要采用新的加载方法
  - 挂载

    mkdir romfs_dir

    sudo mount -o loop romfs.img romfs_dir

添加 hello 到 μCLinux 的根文件系统中 III

- 拷贝 hello？报错!! **WHY**?

```
xlanchen@xlanchen-desktop:~/09FallEOS/skyeye-testsuite-1.2.8/uC
linux/at91/uclinux_cs8900a$ cp hello romfs_dir/bin/
cp: 无法创建普通文件“romfs_dir/bin/hello”: 只读文件系统
xlanchen@xlanchen-desktop:~/09FallEOS/skyeye-testsuite-1.2.8/uC
linux/at91/uclinux_cs8900a$
```

- 复制根文件系统到一个新的目录中

  sudo cp -r romfs_dir/* new_romfs/

- 拷贝 hello

  ***sudo** cp hello new_romfs/bin*

- 生成新的 romfs 映像

  sudo genromfs -f romfs_new.img -d new_romfs/

添加 hello 到 μCLinux 的根文件系统中 IV

- 修改 skyeye.conf，使之使用新的 romfs 映像 romfs_new.img

```
1 #skyeye config file sample
2 cpu: arm7tdmi
3
4 mach: at91
5
6 mem_bank: map=M, type=RW, addr=0x00000000, size=0x00004000
7 mem_bank: map=M, type=RW, addr=0x01000000, size=0x00400000
8 mem_bank: map=M, type=R,  addr=0x01400000, size=0x00400000, file=./romfs_new.img
9 mem_bank: map=M, type=RW, addr=0x02000000, size=0x00400000
10 mem_bank: map=M, type=RW, addr=0x02400000, size=0x00008000
11 mem_bank: map=M, type=RW, addr=0x04000000, size=0x00400000
12 mem_bank: map=I, type=RW, addr=0xf0000000, size=0x10000000
13 #set nic info
14 #net: type=cs8900a, base=0xfffa0000, size=0x20,int=16, mac=0:4:3:2:1:f, ethmod=t
15 net: type=cs8900a, ethmod=tuntap, hostip=10.0.0.1
16 #dbct: state=on
```

添加 hello 到 μCLinux 的根文件系统中 V

- 运行 skyeye

skyeye -c skyeye.conf -e linux

```
Welcome to
                ____ _  _
               /  __| ||_|
    _    _| |   | | _ ____  _   _  _  _
   | |  | | |   | || |  _ \| | | |\ \/ /
   | |_| | |__| || | | | | |_| |/    \
   |  ___\____|_||_|_| |_|\____|\_/\_/
   | |
   |_|

GDB/ARMulator support by <davidm@snapgear.com>
For further information check:
http://www.uclinux.org/

Command: /bin/ifconfig eth0 up 10.0.0.2
SIOCGIFFLAGS: No such device
SIOCSIFADDR: No such device
pid 11: failed 512
Execution Finished, Exiting
init: Booting to single user mode

Sash command shell (version 1.1.1)
/> cd bin
/bin> hello
HELLO WORLD!
/bin>
```

Outline

准备交叉编译环境 I

- 下载arm-linux-tools-20061213.tar.gz
- 在根目录下解压缩

  cd /

  sudo tar zvxf arm-linux-tools-20061213.tar.gz

准备交叉编译环境 II

- 查看安装是否成功

```
xlanchen@xlanchen-desktop:~$ arm-linux-
arm-linux-addr2line         arm-linux-gcov
arm-linux-addr2name.awk     arm-linux-gnatbind
arm-linux-ar                arm-linux-grepjar
arm-linux-arm-linux-gcjh    arm-linux-jar
arm-linux-as                arm-linux-jcf-dump
arm-linux-c++               arm-linux-jv-scan
arm-linux-c++filt           arm-linux-ld
arm-linux-cpp               arm-linux-ld.real
arm-linux-elf2flt           arm-linux-nm
arm-linux-flthdr            arm-linux-objcopy
arm-linux-g++               arm-linux-objdump
arm-linux-g77               arm-linux-ranlib
arm-linux-gcc               arm-linux-readelf
arm-linux-gcc-3.4.4         arm-linux-size
arm-linux-gccbug            arm-linux-strings
arm-linux-gcj               arm-linux-strip
arm-linux-gcjh
```

准备交叉编译环境 III

- 查看版本信息

```
xlanchen@xlanchen-desktop:~$ arm-linux-gcc -v
Reading specs from /usr/local/lib/gcc/arm-linux/3.4.4/specs
Configured with: ../configure --target=arm-linux --disable-shar
ed --prefix=/usr/local --with-headers=/home/gerg/new-wave.ixdp4
25/linux-2.4.x/include --with-gnu-as --with-gnu-ld --enable-mul
tilib
Thread model: posix
gcc version 3.4.4
xlanchen@xlanchen-desktop:~$
```

编译 armlinux I

- 从 kernel.org 上找到 china 的镜像网站，下载linux-2.6.26.tar.bz2
- 在工作目录 $WDIR 中解压缩，后执行如下命令

```
cd linux-2.6.26
make ARCH=arm CROSS_COMPILE=arm-linux- s3c2410_defconfig
make ARCH=arm CROSS_COMPILE=arm-linux- dep
make ARCH=arm CROSS_COMPILE=arm-linux- menuconfig
```

- 修改如下:

编译 armlinux II

- Device Driver→Character Driver→Serial Driver，**取消**8250/16550 and compatible serial support 选项
- 修改 include/asm-arm/arch-s3c2410/map.h
  #define S3C2410_CS6 (**0xC0000000UL**)
- 修改 include/asm-arm/arch-s3c2410/memory.h
  #define PHYS_OFFSET (**0xC0000000UL**)
- Boot options→Default kernel command string:
  mem=32M console=ttySAC0 root=/dev/ram initrd=0xc0800000,0x00800000 rw
- [可选]File systems→ 中，**仅仅保留 ext2**
- [可选]Networking→，**取消 Networking support**
- [可选]Device Driver→，**取消 I2C support 和 USB support**

编译 armlinux III

- 编译

make ARCH=arm CROSS_COMPILE=arm-linux- zImage

需要等待较长时间

```
  LD      vmlinux
  SYSMAP  System.map
  SYSMAP  .tmp_System.map
  OBJCOPY arch/arm/boot/Image
  Kernel: arch/arm/boot/Image is ready
  AS      arch/arm/boot/compressed/head.o
  GZIP    arch/arm/boot/compressed/piggy.gz
  AS      arch/arm/boot/compressed/piggy.o
  CC      arch/arm/boot/compressed/misc.o
  LD      arch/arm/boot/compressed/vmlinux
  OBJCOPY arch/arm/boot/zImage
  Kernel: arch/arm/boot/zImage is ready
xlanchen@xlanchen-desktop:~/workspace/linux-2.6.26$
```

准备根文件系统

- 在后面的课程中准备

Outline

准备交叉编译环境

- 使用前面安装的 arm-elf-<tools>
- 若没有安装，则
  - 下载 arm-elf-tools-20030314.sh
  - 运行

    ./arm-elf-tools-20030314.sh

编译 μCLinux I

- 下载源代码**uClinux-dist-20040408.tar.gz**
- 解压缩

  tar zvxf uClinux-dist-20040408.tar.gz

- 编译

  cd uClinux-dist
  make memuconfig

在 *vendor/product* 选项中选择 *GDB/ARMulator*
Kernel 版本选择 *2.4.x*
其他选项不变（使用缺省选项）

make dep; make

编译 μCLinux II

- 查看编译出来的内核映像
  - images 目录

```
xlanchen@xlanchen-desktop:~/09FallEOS/uClinux-dist$ ls images/ -la
总用量 3260
drwxr-xr-x  2 xlanchen xlanchen    4096 2009-09-21 12:56 .
drwxr-xr-x 17 xlanchen xlanchen    4096 2009-09-21 12:56 ..
-rw-r--r--  1 xlanchen xlanchen 1657136 2009-09-21 12:56 image.bin
-rwxr-xr-x  1 xlanchen xlanchen   39712 2009-09-21 12:56 linux.data
-rwxr-xr-x  1 xlanchen xlanchen  912912 2009-09-21 12:56 linux.text
-rw-r--r--  1 xlanchen xlanchen  704512 2009-09-21 12:56 romfs.img
xlanchen@xlanchen-desktop:~/09FallEOS/uClinux-dist$
```

编译 μCLinux III

- linux-2.4.x 目录

```
xlanchen@xlanchen-desktop:~/09FallEOS/uClinux-dist$ ls linux-2.4.x/
arch            drivers  kernel       mm              Rules.make
COPYING         fs       lib          mmnommu         scripts
CREDITS         include  linux        net             System.map
crypto          init     MAINTAINERS  README
Documentation   ipc      Makefile     REPORTING-BUGS
xlanchen@xlanchen-desktop:~/09FallEOS/uClinux-dist$
```

- 在 skyeye 上运行 μCLinux
  - 建立 skyeye.conf
  - 拷贝 skyeye-testsuite-1.2.8/uClinux/at91/uclinux_cs8900a/skyeye.conf

编译 μCLinux IV

- 修改后，内容如下：

```
1 #skyeye config file sample
2 cpu: arm7tdmi
3
4 mach: at91
5
6 mem_bank: map=M, type=RW, addr=0x00000000, size=0x00004000
7 mem_bank: map=M, type=RW, addr=0x01000000, size=0x00400000
8 mem_bank: map=M, type=R,  addr=0x01400000, size=0x00400000, file=./images/romfs.img
9 mem_bank: map=M, type=RW, addr=0x02000000, size=0x00400000
10 mem_bank: map=M, type=RW, addr=0x02400000, size=0x00008000
11 mem_bank: map=M, type=RW, addr=0x04000000, size=0x00400000
12 mem_bank: map=I, type=RW, addr=0xf0000000, size=0x10000000
13 #set nic info
14 #net: type=cs8900a, base=0xfffa0000, size=0x20,int=16, mac=0:4:3:2:1:f, ethmod=tunta
15 #net: type=cs8900a, ethmod=tuntap, hostip=10.0.0.1
16 #dbct: state=on
```

编译 μCLinux V

- 运行如下命令

skyeye -c skyeye.conf -e linux-2.4.x/linux

```
Welcome to
                 ____ _  _
                /  __| ||_|
 _   _| |  | | _ ____  _   _  _  _
| | | | |  | || |  _ \| | | |\ \/ /
| |_| | |__| || | | | | |_| |/    \
|  ___\____|_||_|_| |_|\____|\_/\_/
| |
|_|

GDB/ARMulator support by <davidm@snapgear.com>
For further information check:
http://www.uclinux.org/

Execution Finished, Exiting

Sash command shell (version 1.1.1)
/>
```

小结

Project3

- 在 skyeye 上成功跑出 armlinux 和 μCLinux
  - 可以使用现成的映像
  - 〔可选，加分〕可以自己编译
- 分别将 hello 加入到 armlinux 和 μCLinux 的根文件系统中，在 skyeye 上启动 armlinux 和 μCLinux 并运行 hello
- 提交报告，要求要说明
  - 被使用的 armlinux 和 μCLinux 的版本和 Linux 内核的版本
  - 〔可选，加分〕若自己编译了内核，给出编译 armlinux 和 μCLinux 的交叉编译器的版本和编译过程
  - 编译 hello 的交叉编译器的版本和编译过程
  - 给出关键输出的图示
- 难度：

  armlinux>μ*CLinux*> 使用现成的内核

Thanks !

The end.