文章编号:1673 - 0062(2012)01 - 0034 - 04

μClinux2.4.17 中添加 SATA 硬盘控制器驱动的方法研究

陆银丽,于红利,黄智伟*

(南华大学 电气工程学院,湖南 衡阳 421001)

摘 要:大部分开源的嵌入式操作系统内核中已经添加了对 SATA 硬盘控制器 sil3114 驱动的支持,而一些老版本的内核不支持 SATA 硬盘控制器.根据实际项目需求,本文采用驱动移植的方法,根据内核的编译原理,生成二进制内核文件,实现了在实时操作系统 μClinux2.4.17 中添加 sil3114 的驱动.经测试,该内核可以很好的支持 SATA 硬盘运行.

关键词:μClinux2.4.x;驱动移植;sil3114;SATA 硬盘

中图分类号:TP311 **文献标识码**:B

Research on the Method of Adding SATA Hard Disk Driver into μClinux2.4.17

LU Yin-li, YU Hong-li, HUANG Zhi-wei*

(School of Electric Engineering, University of South China, Hengyang, Hunan 421001, China)

Abstract: SATA hard disk sil3114 driver has been added into most of open source embedded operating system, but some early version kernels do not support SATA hard disk controller. According to the requirement of practical project, the article adopts the method of driver transplantation, according compiling principle of kernel to generate binary kernel file. It is realized that the driver of sil3114 is added into real-time operating system μClinux2.4.17. After tested, the kernel can support SATA hard disk well.

key words: μClinux2.4.x; drivers transplantation; sil3114; SATA hard disk

0 引 言

嵌入式数字视频录像机由于其高性能、低成本、更稳定的优势逐渐全面进入小区、楼宇、金融等监控领域[1],而 SATA 硬盘以其传输速度高、抗干扰能力强、可靠性高等特点,成为嵌入式数字视频监控系统中实现存储功能设备的首选.本项目中的嵌入式数字视频录像系统采用 S3C2510 微处理器、Flash、SDRAM、PCM1801、TVP5154、DM648、Silicon Image 公司的 SATA 硬盘控制器

收稿日期:2011 - 11 - 16

作者简介:陆银丽(1987 -),女,河南西峡人,南华大学电气工程学院硕士研究生.主要研究方向:集成电路与系统设计. * 通讯作者.

sil3114 等外围设备来实现嵌入式视频监控. 厂家提供的内核为实时操作系统 μClinux2.4.17(以下称为目标内核),支持 S3C2510 微处理器,但不支持 SATA 硬盘控制器 sil3114. 使用时需要将 SATA 硬盘控制器的驱动添加到内核中,使操作系统在启动时就能够支持 SATA 硬盘.

直到 μClinux2.4.27(以下称为源内核)中才添加 sil3114 的驱动,而本系统使用的内核为 μClinux2.4.17,两个版本的内核只有修订版本号不同,表示存在支持驱动的多少、结构体的成员变化、变量的定义不同等一些变动. 本文采用驱动移植的方法,根据内核的编译原理,生成二进制内核文件,实现了在目标内核中添加 SATA 硬盘控制器 sil3114 的驱动. 有效地提高了效率,缩短了开发周期.

1　内核编译

内核编译根据不同的情况步骤会有所不同,但主要分为内核配置、建立依赖关系、生成内核三个步骤[2-3].

1)内核配置

用户根据具体的硬件配置来选择对应的编译选项. 内核配置系统分为三个部分:Makefile,分布在内核的各级目录,用来定义内核的编译规则;配置文件(2.4 内核为 config. in,2.6 内核为 Kconfig),向用户提供配置选项;配置工具,包括配置命令解析器和配置用户界面. 通过内核配置,最终会生成. config 的配置文件.

2)建立依赖关系

读取配置过程中生成的配置文件,来创建对应的依赖关系树,从而决定某段代码是否会被编译进内核.

3)生成内核

执行编译规则,生成内核. 内核镜像分为压缩的内核镜像和未压缩的内核镜像,压缩的内核镜像通常为 zImage,而未压缩的内核镜像为 vmlinux.

2　sil3114 驱动移植的方法

sil3114 是 SATA 硬盘控制器,需要编译到目标内核里,使目标内核在启动时就能够支持 SATA 硬盘. 由于目标内核不支持 sil3114 的驱动,而源内核中提供 sil3114 的驱动,直接把它添加到目标内核中,可能会造成整个内核的崩溃. 为了避免出现这种情况,可先将源内核中 sil3114 的驱动代码编译成模块,最后再添加到目标内核中.

2.1　sil3114 驱动在源内核环境下的模块编译

模块编译最重要一步为 Makefile 的编写,为了确保 Makefile 的正确性,可使 sil3114 的驱动在源内核环境下编译生成模块. Makefile 文件如下:

```
OUTPUT = SATA_SIL. o
OUTPUT_DIR = output
KERNEL = /forlinx/μclinux-2.4.27/
CROSSPREFIX = /usr/local/bin/arm-elf-
CFLAGS = -Wall-I $(KERNEL)/include-c
DEST = $(foreach fn, $(OUTPUT), $(OUTPUT_DIR)/$(fn))
ECHO_OK = echo-e" \xlb[40G\xlb[34m[ OK ]\xlb[0m"
ECHO_FAILED = echo-e" \xlb[40G\xlb[31m[ FAILED ]\xlb[0m"
all: $(OUTPUT_DIR) $(OUTPUT_DIR)/$(OUTPUT)

$(OUTPUT_DIR):
  @ mkdir $@
  @ chmod 777-R $@
$(OUTPUT_DIR)/%. o:sata_sil. c
  @ echo-n" Complin $^..."
  @ if $(CROSSPREFIX) gcc $(CFLAGS) $^-o $@;then $(ECHO_OK);else $(ECHO_FAILED);fi
```

在编绎的过程中要注意以下问题:

1)在搜索头文件时,确保按照正确的搜索路径搜索.

在头文件的搜索过程中,先搜索编译选项指定的"-I"或"--include-dir"目录;然后,如果 < prefix >/include(一般是/usr/local/bin 或是/usr/include)存在,make 也会把这些路径作为搜索路径. 由于新生成的源内核,在 include 目录下没有 asm 目录,会使头文件引用错误. 本文中选择的 S3C2510 微处理器不含 MMU(Memory Management Unit),需要将 asm 链接指向 asm-armnommu,实现方法:sudo ln-s asm-armnommu asm.

2)确保生成的[. o]文件与[. c]文件的名字一致

Makefile 的隐晦规则,可以自动推导文件以及文件依赖关系,即 make 看到一个[. o]文件,会自动把[. c]文件添加到依赖文件中,这就要求目标文件和依赖文件的名字完全相同. 以 sil3114 的驱动为例:驱动的[. c]文件为 sata_sil. c,则要求

生成的[.o]文件为 sata_sil.o.

3)确保包含正确的头文件,避免出现隐式声明.

如果没有正确包含结构体、变量、宏定义、函数等所在的头文件,会出现隐式声明错误.以函数为例进行说明:编译器在处理函数调用代码时需要函数原型,因为只有确定函数参数的类型、个数及返回值类型等,才可以判断要生成什么命令.而如果编译器在处理函数调用代码时没有找到原型,只好根据函数调用代码做隐式声明,即隐式声明是从函数调用代码推导而来的.而事实上,函数定义的形参类型与函数调用代码中的实参类型可能不一致;另外,从函数调用代码也无法确认函数的返回值类型,所以隐式声明只能规定返回值的类型为 int 型.这些都会导致在函数的实际引用中出错[4].解决方法:找到变量、函数等所在的头文件,在对应的文件中包含该头文件.

4)避免出现解析错误.

解析错误出现的原因分为以下几种:

(1)语法错误通常表现为如丢失标点符号、大括号不匹配、宏定义的格式不正确等,只要进行相关修改即可.

(2)在函数、结构体、变量等的说明之前使用,也会引起解析错误,和隐式声明错误类似,包含正确的头文件或在文件开头添加声明即可.

5)注意内核里的条件宏定义.

条件编译是按照不同的条件,只编译符合条件的代码,可生成不同的目标代码文件,同时目标代码较小,方便程序的移植和调试[5].在内核中,通常分为两种方法来减小目标代码的大小:

(1)通过控制编译的参数来确定某段代码是否被编译到内核中,常采用类似于__KERNEL__、__ASSEMBLY__等宏变量.在使用时,只需在 Makefile 中的 CFLAGS 中添加编译选项-D__KERNEL__或-D__ASSEMBLY__.

(2)通过在内核中定义条件宏.在内核的某些文件中,会定义相关的变量为 0 或 1,在编译过程中,把这些变量作为条件实现条件编译.因此,在查看错误信息时,需注意是否已经定义了该条件宏.

2.2 sil3114 驱动在目标内核环境下编译成模块

在确保 Makefile 正确性的情况下,把目标内核作为内核路径,编译生成模块,以此确保目标内核不出现类似于以上的编译问题.

此外,还需注意与内建函数类型冲突.内建函数与语言相关(更确切说是与编译器相关),与关键字作用一致,无需说明,也无需包含任何头文件,编译器就可以使用这些函数.为了提高代码的运行效率,内核中使用了大量的内建函数.其中很多是标准 C 库函数的内建版本,如 memcpy,它们与对应的 C 库函数功能相同,而其他内建函数的名字通常以__builtin_开头,通常比较短小且执行速度较快.Gcc 的编译参数-fno-builtin 用来指出在程序的编译过程中,只识别以__builtin_开头的内联函数.在 Makefile 的 CFLAGS 中添加编译选项-fno-builtin,使 Gcc 不要使用以下内建函数:如 memcpy、memset 和 memcmp 等[6].

2.3 sil3114 驱动编译到目标内核中,生成二进制文件

1)sil3114 驱动编译到目标内核时,需要修改 Makefile 和 Config.in.

(1)把相关文件复制到/forlinx/S3C2510/linux2510/对应目录下.

(2)修改/forlinx/S3C2510/linux2510/drivers/scsi/目录下的 Makefile 文件,添加内容如下:

Obj-$(CONFIG_SIL3114) += sil3114.o,保存退出.

(3)修改/forlinx/S3C2510/linux2510/drivers/scsi/目录下的 Config.in,使之在使用命令 make menuconfig 时,生成支持 sil3114 的配置选项.

在 comment"SCSI low-level drivers"后添加以下内容:

bool 'Silcon Image Sil3114 Driver' CONFIG_SIL3114,保存退出.

对于 bool 选项只有两种选择:"Y"和"N",在当选中该选项时表现为[*].

(4)转到 linux2510 目录:cd/forlinx/S3C2510/linux2510/,执行以下命令.

sudo make clean:删除以前步骤留下的文件,避免因无法找到某些文件而出现错误.

sudo make menuconfig:进入 Ncurses 图形配置界面.

利用上下键依次选择"SCSI support"→"SCSI low-level drivers"→"Silcon Image Sil3114 Driver",按"Y"选中 Silcon Image Sil3114 Driver,在退出前,选择"Yes"保存对内核的配置.

sudo make:编译内核.

2)gcc 编译过程

gcc 编译过程主要分为预处理、编译、汇编、链接四个阶段.预处理阶段根据预处理指令来修改源文件的内容,主要完成源程序的"替代"工

作. 编译就是通过词法分析和语法分析,在确认所有的指令都符合语法规则之后,将其翻译成等价的中间代码或汇编代码. 汇编是把汇编代码翻译成目标机器指令的过程. 链接主要是用链接器把生成的目标代码和系统或用户的库链接起来,生成可执行文件[7].

gcc 编译过程应注意以下问题:

(1)避免出现链接错误

在链接过程中,主要是链接函数和全局变量,链接器并不管函数所在的源文件,只需找到函数的中间目标文件,若找不到就会出现链接错误. 由于 SATA 硬盘控制器的驱动文件还依赖于其他的[.c]文件,需要把这些文件都编译到内核里,生成对应的[.o]文件,避免出现链接错误.

(2)符号导出

EXPORT_SYMBOL_GPL 说明的符号必须在模块文件的全局部分导出,不能在函数中导出,这是因为这些宏将被扩展成一个特殊用途的声明,而该变量必须是全局的[8]. 这个变量存储于模块的一个特殊的可执行部分(一个"ELF 段"),在装载时,内核通过这个段来寻找模块导出的变量(可通过 <linux/module.h> 查看详细信息).

3　实验结果及结语

本文采用驱动移植的方法,根据内核的编译原理,生成二进制内核文件,实现了在实时操作系统 μClinux2.4.17 中添加 SATA 硬盘控制器驱动. 实验测试利用 USB 下载线,将该内核下载到开发板上,可以正常的支持 SATA 硬盘操作. 实践证明,采用这种方法可以明显地缩短开发周期,提高效率. 该方法也是在不熟悉内核编码规律以及具体硬件工作原理下,往内核中添加驱动的一种有效方法.

参考文献:

[1] 章立,徐立鸿,姜磊,等. 嵌入式数字视频录像机 GUI 系统的设计与实现[J]. 微电子学与计算机,2006,23(2):58-61.

[2] 商斌. Linux 设备驱动开发入门与编程实践[M]. 北京:电子工业出版社,2009:121-125.

[3] 黄智伟. ARM9 嵌入式系统基础教程[M]. 北京:北京航空航天大学出版社,2008:384-393.

[4] 徐尔贵,丁雷. Visual Basic 教程[M]. 北京:清华大学出版社,2003:34-37.

[5] 谭浩强. c 语言程序设计[M]. 2 版. 北京:高等教育出版社,1998:130-140.

[6] Griffith Arthur. GCC:The complete reference[M]. United States of Amercia:The McGraw-HIll Companies. Inc,2002:529-530.

[7] 杨宗德,邓玉春. Linux 高级程序设计[M]. 2 版. 北京:人民邮电出版社,2009:31-32.

[8] Jonathan Corbet, Alessandro Rubini, Greg Kroah-Hartman. Linux device drivers[M]. 3rd. Sebastopol: O'Reilly Media. Inc,2005:615-617.