第 8卷第 4期 南京师范大学学报 (工程技术版) Vol. 8 No. 4
2008年 12月 JOURNAL OF NANJING NORMAL UNIVERSITY (ENGINEERING AND TECHNOLOGY EDITION) Dec, 2008

Transplant uClinux Based on S3C44B0X Through U-Boot

Wang Liwei, Yang Houjun, Fan Yanbing

(College of Information Engineering, Qingdao University, Qingdao 266071, China)

Abstract In the development of an embedded system, it is necessary to build a complete embedded operation system to achieve a faster development period of application program and device driver. Above all, it is imperative to construct U-Boot-lead program. Once a U-Boot-lead program for the operation system kernel is prepared, the self-contained starting environment is used to carry out the transplant of uClinux. This is based on a S3C44B0 board. Finally, from the aspect of a transplant onto a board, the transplanting process is systematically analyzed and described by combining with a hardware platform.

Key words: U-Boot, S3C44B0, uClinux, transplant

CLC number: TP 316, **Document code**: A, **Article ID**: 1672-1292(2008)04-0150-05

通过 U-Boot实现基于 S3C44B0X的 uClinux移植

王丽伟, 杨厚俊, 范延滨

(青岛大学 信息工程学院, 山东 青岛 266071)

[摘要] 在嵌入式系统开发中, 为了加快应用程序以及设备驱动的开发周期, 需要建立一个完备的嵌入式操作系统. 首先, 建立 U-Boot引导程序是必不可少的. 在 U-Boot引导程序为操作系统内核准备好环境之后, 利用它实现了 uClinux基于 S3C44B0开发的移植. 最后, 从板级移植的角度, 结合硬件平台结构系统地分析和讲解了移植过程.

[关键词] U-Boot, S3C44B0, uClinux, 移植

S3C44B0X developed by Samsung is a 16/32 bit RISC embedded MPU. It is based on ARM 7TDMI kernel, and is widely applied in China. The ARM 7TDMI function module integrates lots of periphery function module. In addition, it is a low-cost, high-powered MPU which is developed for handhold equipment and other all-purpose equipments. Due to the low-cost and brief design, it is useful for an application with high requirement of cost and power consumption[1].

This subject uses processor S3C44B0X, which is composed of the core board and extensive function modules. The core board includes processor S3C44b0X, a Flash (2MB) and a SDRAM (8MB). The extensive modules include a Flash (16MB), two Uart, a Jtag interface, a Ethernet interface, and display.

The function structure is shown in Fig. 1.

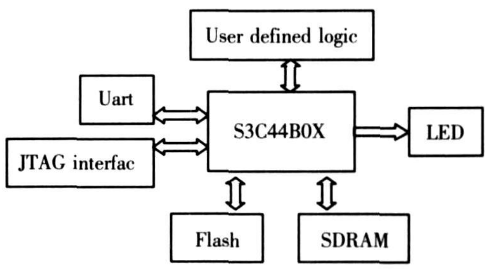


Fig.1 Function Structure

Presently, uC/OS and uClinux are two main embedded operating systems which are based on non-MMU MPU[2]. Derived from Linux, uClinux keeps most advantages of Linux operating system, and supports multitask. Besides this, uClinux is rewrited in order to be the same with non-MMU MPU, so its kernel is small, commonly being used in S3C44B0 processor. Therefore, this subject

Received date: 2008-06-18.
Foundation item: 863 Program (2006AA01Z110).
Corresponding auther: Yang Houjun, associate professor, field of study: computer system structure. E-mail: hjyang666666@126.com

chooses uClinux as the operating system of embedded system.

1 Cross Compile Environment

1.1 Introduction

Cross compile is a process of creating a target code that is created in a computer and run in another one, through the way of using compiler to cross compile a program. Cross development tool mainly includes IDE integrated development environment and Makefile method[3]. Normally, the application exploitation uses integrated exploitation enviromment, kernel compiling uses the later one. The final purpose of this subject is compiling uClinux kernel, so Makefile method is chosen.

The development environment includes binutils, gdb, gcc and glibc which will be used for kernel compile[4]. Cross compile tool chain can be both self-made and downloaded. Making a cross compile tool chain is a complicated process and easily make mistakes. In order to succeed in compile kernel, this subject chooses to download a finished cross compile tool chain from internet.

1.2 Construct cross compile enviromment

Because uClinux is the operating system which is needed to transplant, it is necessary to install a corresponding cross compile tool before the transplant. Arm-elf-tools-20030314.sh is used here. Execute command ./arm-elf-tools-20030314.sh, and an uClinux-arm cross compile environment can be automatically set up in the host[5].

With the installed tool—H-Jtag, the successful compiled bootloader can be loaded into Flash.

2 Bootloader Transplant

Bootloader is the first program after embedded system be electrified, which is a very important component of embedded system. Constructing or transplanting a Bootloader will bring lots of convenience to continue exploitation. Through Bootloader, it can initialize hardware equipment, set up internal spatial map, adjust hardware and software to appropriate situation and prepare proper environment for the transplanting kernel. The main role of Bootloader is to load kernel image from host into board, then go to the kernel portal and startup the kernel.

Actualization of Bootloader depends on the hardware, and different processors with different structures have different Bootloaders. In addition, Bootloader actually relies on the specific configuration of board, so transplant Bootloader aimed at developing board is becoming more important. At present, Bootloader mainly have Blob, vivi and U-Boot in common use. This subject chooses U-Boot to transplant.

2.1 Introduction of U-Boot

U-Boot is an open source project follows GPL terms. In addition to all universal functions of Bootloader, U-Boot also has several advantages such as supporting multiple embedded operating system kernel, supporting multiple processor, high reliability and stability, and plenty of device driver source code.

2.2 Memory space partition

Seek Table 1.

Table 1 Memory space partition

Content	Starting address	Storage medium
Bootloader	0x00000000	Flash
Image load address	0x00040000	Flash
Kernel start address	0x0c008000	SDRAM
Fs load address	0x0c600000	SDRAM

2.3 Transplant U-Boot

2.3.1 Startup flow analysis

U-Boot startup includes two phases, code of phase 1 is defined in file "start.S", which includes the part of electrifying system that executes at 0x0, configure registers and copy phase 2 to RAM. Phase 2 starts at main.c, and it is to examine memory mapping of system, read kernel image and root file system image from Flash to RAM, set up the startup parameter and transfer kernel.

By this token, transplanting U-Boot in allusion to S3C44B0 mostly just need to modify the hardware configure of S3C44B0, and set up the startup parameter for kernel. Besides, in order to ensure the kernel works successfully, it is necessary to do some specific modifications

2. 3. 2 Transplant analysis

Due to B2 board in U-Boot 1. 1. 4 adopts S3C44B0X processor The transplant of U-Boot will be consulted of B2 board

1. Construct the board head file wlw44b0 h

• Change S3C44B0 main frequency to 66

• Append RTL8019 network card

# define CONFIG_ DRIVER_ RTL8019

# define RTL8019_ BASE　0x0a000600

• In order to save enviromment variable into Flash, append Flash parameters

# define CFG_ ENV_ IS_ IN_ FLASH　1

# undef CFG_ ENV_ IS_ IN_ NOWHERE

• Configure the size of storage space and start address for SDRAM and Flash

2 Modify hardware configuration

According to S3C44B0 hardware configuration, modify register value in function board_ init

3 Modify startup file start S

Start S is the first file that the CPU processes, so it is important to ensure its stability In order to ensure its stability and reliability, some modification should be done as follows

• Delete seven interrupt jump sentences, append jump sentences for seven abnormal interrupts ldr pc, abnormal interrupt address

• Modify cpu_ init_ criti part, change the value PLLCON which correspond to MCLK 66 to 0x7c041

4 Modify lowlevel_ init S

Change REFRESH register value to 0x9603fd, refresh register, and improper configure will impact serial output

5 Modify the file config mk which under the board directory

In order to avoid conflict of startup parameter and kernel reflection, change TEXT_ BASE to 0xc700000, let it at the high address of memory

6 Modify the file Makefile which under root directory

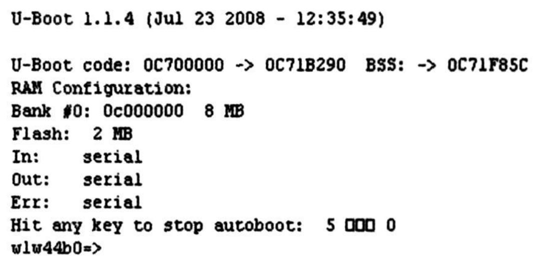
Configure' CROSS_ COMPILE' to ' arm - elf- ', so it will be compiled by the cross compile tool

Thus, it has basically finished the transplant of U-Boot Finally, execute ' make B2_ config' and ' make' command, then get ' u- boot bin' file

2. 4 Download u- boot bin to board

Compile and create ' u- boot bin' file, and then use H - Jtag tool, download it into the board at 0x0 The serial has output and all functions it provided are usable, viz, transplanting U-Boot is successful

Fig 2 shows how U-Boot works



```
U-Boot 1.1.4 (Jul 23 2008 - 12:35:49)

U-Boot code: 0C700000 -> 0C71B290  BSS: -> 0C71F85C
RAM Configuration:
Bank #0: 0c000000  8 MB
Flash:  2 MB
In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  5 □□□ 0
wlw44b0=>
```

Fig.2 u-boot startup

3 Transplant uClinux

3. 1 Instruction of uClinux

uClinux is a modification of Linux, it is developed for non-MMU microprocessor and have been applied broadly to non-MMU microprocessor such as ARM and MIPS In order to apply to non-MMU microprocessor, it has been rewritten, so it is much smaller than Linux In addition it keeps the advantages of Linux like stability, excellent network ability and supporting file system.

3.2 Transplant uClinux

Transplant uClinux generally includes three levels: structure level transplant, platform level transplant and board level transplant.

If the structure of CPU which is going to be transplanted is different from any CPU that uClinux supports, then modify related CPU structure file under "linux/arch", this is called structure level transplant. If uClinux supports the CPU, create directory and write code under related system structure directory, this is called platform level transplant. Lastly, if uClinux supports the CPU, only board level transplant is enough. Board level transplant is creating related developing board directory under "linux/arch", then modify it according to board.

uClinux has already supported S3C44B0X processor, so just need to do board level transplant.

This subject chooses uClinux-dist-20040408 to transplant. The linux-2.4.x in it is actually the kernel 2.4.20. Although it supports s3c44b0 much well, the patch 'uClinux-20040408-ARMSYS.patch' is still necessary, which is made according to s3c44b0.

Now present the transplant process of uClinux in details.

1. Append 44B0 directory and file

Create 44B0 directory under "vendors/Samsung", copy all the file from directory 4510 to 44B0.

2. Patch the file

Copy the patch file to uClinux-dist root file, and then execute the command "patch —p1 < uClinux-20040408-ARMSYS.patch".

3. Append the image file

Modify "Makefile" under "vendor/Samsung/44B0", append the related content about image file, so "image.rom" and "image.ram" can be got.

4. Modify kernel configure file

According to the idiographic configuration of S3C44B0, modify the address and size of SDRAM and Flash, main frequency and external clock, and startup parameter "CONFIG_CMDLINE". Neglecting these will cause kernel runs irregularly.

5. Append romfs file system

Because of the shortage of linux-2.4.20 for supporting romfs file system, it is important to add romfs file system manually.

• Append related file image information

In order to create the romfs file system, modify "Makefile" which under "vendor/Samsung/44B0":

```
(CROSS_COMPILE) ld —r —o$ (ROOTDIR) /$ (LINUXDIR) /romfs.o —b binary$ (ROMFSIMG)
```

• Modify the file "blkmem.c" under "linux-2.4.x/drivers/block"

Append variables "romfs_data[ ]" and "romfs_data_end[ ]". And append "{0, romfs_data, -1}" in order to point the address romfs placed.

• Modify file "vmlinux-armv.lds.in" which under "linux-2.4.x/arch/armnommu"

Append romfs.o, and capture its start and end address——"romfs_data" and "romfs_data_end".

6. Append serial driver

Linux2.4.20 kernel only implements simple serial driver for S3C44B0X, so appending serial driver program is important.

• Append configure option of serial

Append serial interface option in the file "Config.in" under "linux-2.4.x/driver/serial", in order to make sure that it has the option when compiling the kernel.

• Define serial port number

Append definition of "PORT_S3C44B0" in "serial_core.h", define the value to 38.

• Append objective file in "Makefile" under "linux-2.4.x/driver/serial":

obj-$ (CONFIG_SERIAL_S3C44B0X) + = serial_s3c44b0. o

3.3 Configure and compile kernel

Compile and configure kernel after finish the basic transplant. First, execute command "make menuconfig" under directory "uClinux-dist", choose "Samsung" as producer and board "44B0", use "linux2.4.x" kernel and "uClibc" library.

And it's necessary to select the options "Customize Kernel Settings" and "Customize Vendor/User Settings", which means that the kernel and vendor/user option will be configured manually.

Configure file will be set to "config_linux_2.4.x" and "config_vendor_2.4.x", then make sure the configure and append other functions needed, mainly as follows:

- The base address and size of SDRAM and Flash defined in the "System Type" options.
- Notice definitions of main frequency and external clock frequency which under System Type options.
- Another point needed to pay attention to is "Default kernel command string" in General setup options.

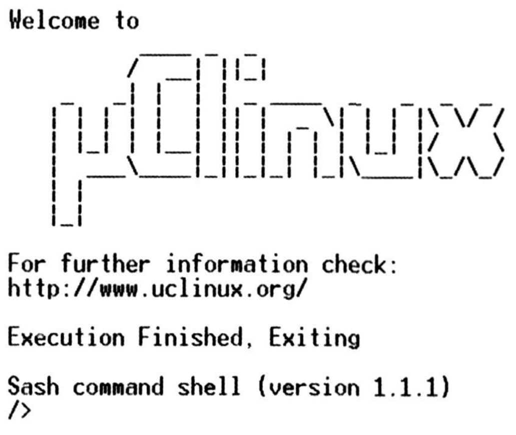
Set up parameters which are used when startup kernel. Set up it as" root= /dev/ram0 init= /linuxrc", "root" designates root file system on "/dev/ram0", set "init= /linuxrc" in order to load root file system correctly.

Thus, configure of kernel is basically finished, according to practical circumstance to do other configures. After that, execute commands of "make dep", "make lib_only", "make user_only", "make romfs", "make linux", "make image" orderly, then the image file will be generated.

3.4 Debug and download

U-Boot provides two different operating modes: startup and load mode, download mode. Under startup and load mode, U-Boot loads and run operating system into RAM, users can't intervene the process; under download mode, U-Boot downloads kernel image and root file system into RAM from Host through communication method such as serial or network. This mode will not engross too much memory space, this subject use network interface to download executable file.

Install tftp server in host, use the "tftp" function, download "image.ram" and "romfs.img" to RAM at "0xc008000" and "0xc600000", then implement "go 0xc008000" order and startup kernel.



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

For further information check:
http://www.uclinux.org/

Execution Finished, Exiting

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

Fig.3 uclinux startup

4 Conclusions

This paper describes transplant u-boot leading procedure and transplant process of uClinux kernel that based on s3c44b0, being master of the transplant process of u-boot and uClinux is important for subsequent equipment drive transplanting and writing.

According to the transplant includes three levels, this paper particularly presents the transplant process of uClinux from "transplant based on board" point of view. In addition, pay attention to the relationship between software and hardware during the process, it elicits subsequent transplant of other board, and makes it easier.

(下转第 172页)

[5] 求是科技. Visual C++ 数字图像处理典型算法及实现 [M]. 北京: 人民邮电出版社, 2006: 315-333.
Qiushi Keji. Visual C++ typical algorithm & implementation in digital image processing[M]. Beijing: Posts & Telecom Press, 2006: 315-333. (in Chinese)
[6] 刘瑞祯, 于仕琪. OpenCV教程基础篇 [M]. 北京: 北京航空航天大学出版社, 2007: 243-326.
Liu Ruizhen, Yu Shiqi. OpenCV course(part 1: fundamental) [M]. Beijing: Beijing University of Aeronautics and Astronautics Press, 2007: 243-326. (in Chinese)
[7] 程军娜, 姬光荣, 冯晨. 基于数学形态学的藻细胞图像预处理 [J]. 中国海洋大学学报, 2008, 38(3): 513-516.
Cheng Ju'na, Yi Guangrong, Feng Chen. Algae cell image preprocessing based on mathematical morphology[J]. Periodical of Ocean University of China, 2008, 38(3): 513-516. (in Chinese)
[8] Kumar S, Ong S H, Ranganath S, et al. A luminance-and contrast-invariant edge-similarity measure[J]. IEEE Trans Pattern Anal Mach Intell, 2006, 28(12): 2 042-2 048.
[9] Serra J, Vincent L. An overview of morphological filtering [J]. Circuits System Signal Process, 1992, 11(1): 47-108.

[责任编辑: 孙德泉]

(上接第 154页)

[References]

[1] Tian Ze. The Development and Application of Embedded System [M]. Beijing: BeiHang University Press, 2005.
[2] Xiong Chunjie, Zhong Jiaqi, Chen Qioumei. Transplant uclinux based on s3c44b0 development platform[J]. Control & Automation, 2007, 23(2): 46-48.
[3] Li Yan, Wang Weibing, Sun Yongchun. Transplant uClinux OS kernel based on S3C44B0X[J]. Journal of Harbin University of Science and Technology, 2006, 11(2): 73-75.
[4] Wu Rui, Zhu Jun, Wang Yuwang, et al. Transplant uClinux based on UP-NETARM 3000 platform[J]. Computing Technology and Automation, 2006, 25(4): 228-232.
[5] Chang Yinxia, Zhang Zhendong, Tang Jiying. Transplant uClinux to S3C44B0 through Bootloader[J]. Journal of Hebei University of Technology, 2005, 34(2): 100-104.
[6] Chen Weijun, Li Zhengming, Sun Jun, et al. The design and implement of S3C44B0 leading program based on U-Boot[J]. Embedded Software Application, 2007, 1(2): 113-115.

[责任编辑: 顾晓天]