Linux Porting on an ARM and Application Development

Saurabh Kabra

ECE, VIT University, Vellore.
E-mail : Saurabhkabra2009@gmail.com

***Abstract* – In this paper I have discussed about how to port or boot Linux operating system on an ARM based board and has discussed about application development on the same ARM based board. So basic idea of this paper is to take latest Linux kernel code and modify it according to your board and application requirement. After modification I compiled the modified kernel code with ARM cross compiler for Linux and generate a ".bin" file. Using flashkit for my board I have flashed this file into my board and generated a Linux image file using universal boot loader, which further in result produced a command prompt window i.e. my board has been booted.**

After booting of my board I developed an application which will interface my keyboard with ARM board and show the output at my board display through UART.

Keywords – Cross Compiler, Flashkit, Kernel, UART.

I. INTRODUCTION

As we can see in present scenario market survey denotes that many smart phones are there which have been installed with android. So my idea is like why we can't boot the same mobile platform using simple Linux OS rather than going for android version. The reason why I am emphasizing on Linux OS is that its kernel code is easily available from their official website. Anyone can download it from www.kernel.org for free and can use it even for different application development. But this feature is not there with android OS as it is not freely available for everyone. Google publishes most of the code under the apache license 2.0.

II. RELATED CONCEPTS

A. DB8500 series board:-

The process of Linux porting which I will be discussing here is generalized one but some of the commands are specifically for DB8500 board manufactured by ST Ericsson. This DB8500 has following features

1. It has cortex A9 processor on board.
2. It provides WCDMA/EDGE modem support.
3. It has on chip ROM of 180Kbytes and eSRAM of 640Kbytes.
4. It has Smart Video Accelerator (SVA) which provides 1080HDTV video encoding.
5. It has Smart Imaging Accelerator (SIA) which can provide 20MP snap shots.

B. ARM cortex A9 processor:-

Features of cortex A9 processor are:

1. It is dual core processor with 128 interrupts.
2. It has 512 Kbytes 8 way L2 cache controller
3. Runs on different domains like Run mode, Idle Mode, off mode, Retention Mode etc..

C. Linux Kernel:-

Linux is free open source software under GNU general public license. Kernel is the most important part of any Linux code. It is the one program which always runs. All the modification according to our boards are made in this part only.

D. Fedora 17:-

Linux has different flavors.Here for my project I have used FEDORA version 17. Its features are:

1. Using kernel version 3.6.
2. Support for network manager.
3. More GUI based interface.

E. Flashkit:-

Flashkit is a software tool used to flash software, achieves or loaders o the concern mobile platform. It has 5 parts:

a. Assemble tool
b. Sign tool
c. Flash script engine

F. UART:-

UART stands for Universal Asynchronous Receiver Transmitter. It means we can use same UART for receiving and transmitting the serial or parallel data simultaneously.

G. U-Boot:-

The U-boot utility is a multi-platform, open source universal boot loader for loading boot images like Linux image. It is also called as primary boot loader.

III. PROCESS

Process for Linux porting is shown in Fig: 1

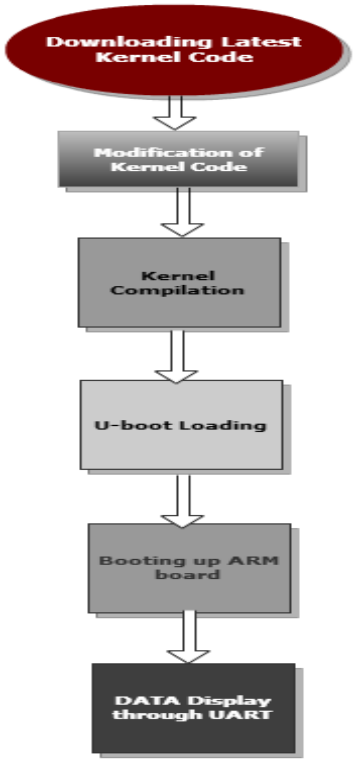

Fig. 1 : Linux porting process

Now I will discuss each step in detail.

1. Downloading kernel code:-

It is first and foremost step to choose latest and stable Linux kernel. This latest kernel can be easily downloadable from www.kernel.org. This will be zipped version. If you downloaded it on a PC with window installed then you can directly use any zip utility to unzip it.

For Linux machine just open the terminal and type following command:

$ tar xfv linux-3.7.5.tar.bz2

2. Modification of kernel code:-

Once you unzipped your .tar file your kernel code is ready to be modified. At first open the terminal window in your Linux machine and go to Linux directory using CD command. For example

$ cd home/user/Documents/Linux-3.7.5

After that you need to open “Kbuild” file using command

$ vi Kbuild

This will in turn open a file where you need to define your architecture as

SRCARCH=arm

And then open make file using command

$ vi Makefile

In this file you need to define your cross compiler as:

CROSS_COMPILE=arm-unknown-linux-gnueabi-

This will define your cross compiler. Cross compiler is a compiler which runs on a machine but produces a code for another machine.

3. Kernel Compilation:-

For kernel compilation first you need to set your specification in configuration command window. For that you need to type following command in your terminal window:

$ make menuconfig

This command will present a window (in fig 2) which will have options like processor type,

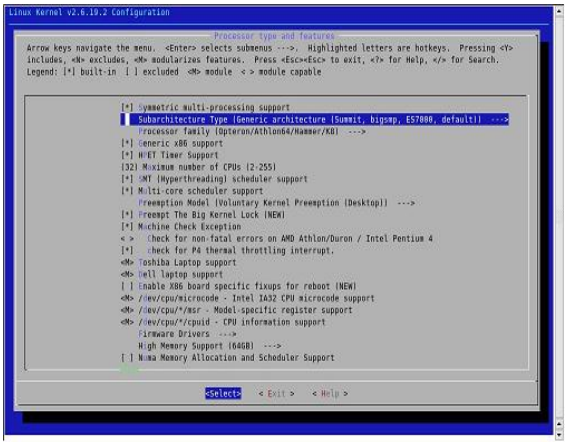

Fig. 2: Menuconfig Window.

device drivers etc. in short you can define all your board and architecture specifications here and using which you can configure your code.

After saving the .config fileyou just need to type

$ make Image ARCH=arm

CROSS_COMPILE=arm-unknown=linux-gnueabi-

This will generate a .bin file in your Linux library named as "VMLINUX.bin" as well as it will also generate "system.map" and "VMLINUX.o".

4. U- boot loading:-

This the most important step before the execution of ARM boot up using Linux. As we already got VMLINUX.bin file. Now using any zip utility we will convert this ".bin" file into ".gz" file i.e. VMLINUX.gz.

Now I will use flash kit for DB 8500 board to flash the board. This flashkit work can be done on either Linux machine or window machine. I did it on window machine. First of all I installed the driver for flashkit in a window PC. Then I opened command prompt and using CD command I went to my flashkit directory.

There at first I typed following command:

>flasher.bat hrefp_v42_v20_db8500b0_secst –L –e

This command erased the memory of my board. And then I typed the following command:

>flasher.bat hrefp_v42_v20_db8500b0_secst –L –c

After the execution of following command this will ask to connect the hardware.

For hardware connection I used controlled power supply of 4V as my board cannot afford more than 4v voltage. And then using USB cable I connected my board to my PC.

When I connected my hardware it flashed it and then I loaded my U-boot with my compressed Linux image into my hardware which generated a uncompressed image of my Linux image. And it is stored in my board DDR(Double Data Rate).

5. Booting up of board:-

After the uncompressed image has been generated next will be booting up my ARM processor based board. For this purpose I will take my uncompressed Linux image code and after compilation of this, my board will show a command prompt screen on its display. This displays is actually running on Linux OS which shows that our board has been booted with Linux OS.

6. Data display through UART:-

As my board has been booted-up using Linux OS the next step is to development of an application which shows the input data (which has been entered using keyboard) through UART to my board display.

For that at first I interfaced one USB keyboard to my board using simple interfacing assembly language program and I have to install the device driver to my board flash memory using simple flash kit application. After the installation of keyboard and its interface with board I changed the make file configuration in kernel code which in turned directed the keyboard to take its data from user and send this data to UART port 2. And this UART port is again already interfaced with display mounted on my board. So it is showing the same data on my board's display.

IV. CONCLUSION

So finally I booted my ARM processor based with display of data on my board's display. So from this paper I can conclude that using Linux based porting I can easily boot my board. The major advantage of this technique is that it will bring down cost of mobile as for mobile OS companies need not to pay anything to anyone because Linux offers Royalty free licensing.

And another advantage is that it will allow a simple basic programmer to design his own application and either he can launch it to any company or he can sell it to any mobile giants. And I am quite sure in future it will be the most popular technology for the mobile OS porting.

V. REFERENCES

[1] Linux-Kernel-Programming in a nutshell by Greg Kroah-Hartman from OREILLY publication

[2] Building Embedded Linux Systems by Karim Yaghmour, Jon Masters, Gilad Ben-Yossef, andPhilippe Gerum from OREILLY publication.

[3] www.kernel.org

[4] www.fedoraforum.org

[5] www.gnuarm.org

[6] www.linux-arm.org/pub/LinuxKernel/ WebHome/aleph-porting.pdf

[7] www.ijera.com/papers/Vol2_issue2/ JO2216141618.pdf

◈◈◈