SAMSUNG ELECTRONICS

Installation Manual for UBI

S3C6410

May 09, 2008

(Preliminary) REV 1.00

Important Notice

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. Samsung assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained herein.

Samsung reserves the right to make changes in its products or product specifications with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

This publication does not convey to a purchaser of semiconductor devices described herein any license under the patent rights of Samsung or others.

Samsung makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Samsung assume any liability arising out of the application or use of any product or circuit and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

"Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by the customer's technical experts.

Samsung products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, for other applications intended to support or sustain life, or for any other application in which the failure of the Samsung product could create a situation where personal injury or death may occur.

Should the Buyer purchase or use a Samsung product for any such unintended or unauthorized application, the Buyer shall indemnify and hold Samsung and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorized use, even if such claim alleges that Samsung was negligent regarding the design or manufacture of said product

S3C6410 RISC Microprocessor
Installation Manual for Unsorted Block Image

Samsung Electronics Co., Ltd.
San #24 Nongseo-Dong, Giheung-Gu
Yongin-City Gyeonggi-Do, Korea
446-711

Home Page: http://www.samsungsemi.com/

E-Mail: mobilesol.cs@samsung.com

Printed in the Republic of Korea

Revision History

Revision No	Description of Change	Refer to	Author(s)	Date
1.00	- Master Copy	-	Jongpill Lee	May 09, 2008

NOTE: Revised parts are written in blue.

CONTENTS

Figures

1 Introduction

1.1 Purpose

Ubi is general purpose flash management layer which has similar functionality as the Logical Volume Manager. The basic idea is that the UBI layer provides the management for multiple logical volumes, which can contain either static data or dynamic contents e.g. file systems.

This guide will explain the initial steps for Installing UBI(Unsorted Block Image) and how to compile Linux Kernel.

1.2 Overview

UBI Supports as follows:

- Bad Block Management
- Wear leveling across the device(s)
- Logical to Physical block mapping
- Volume information storage
- Device information

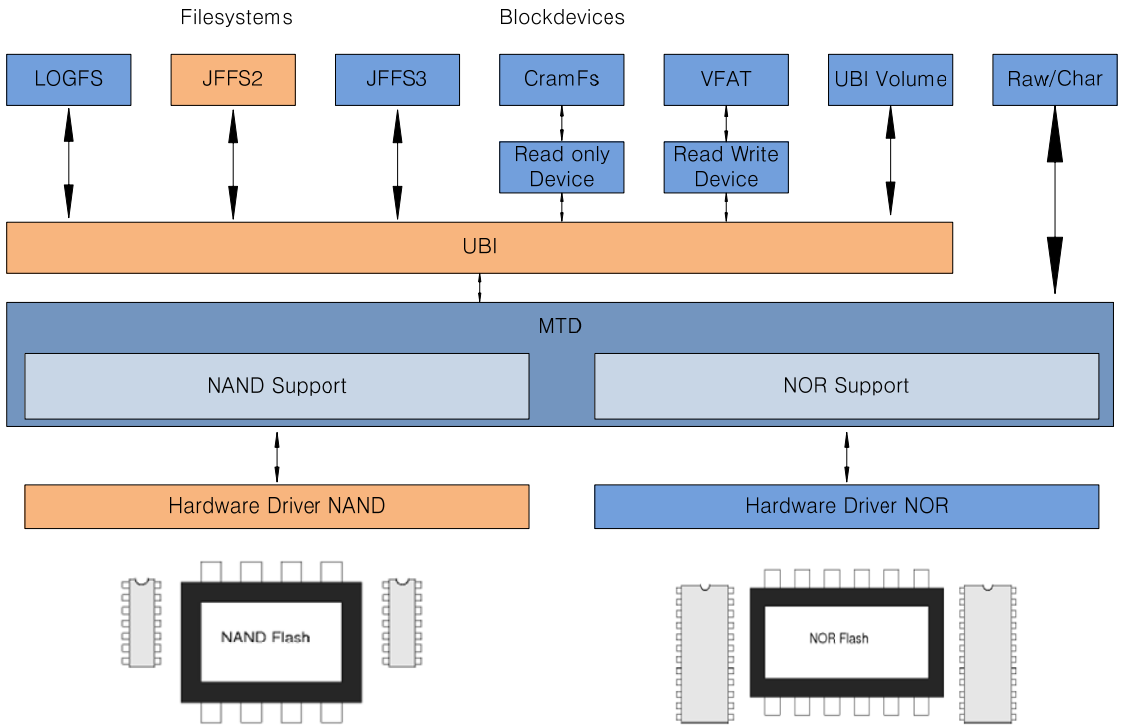


FIGURE 1 UBI/MTD Integration

2 Kernel Compilation

2.1 Root file System

Make sure you have 2.6 compliant versions of *insmod, rmmod,* and *lsmod*. At least you have 2.6 compliant versions of *insmod*.

2.2 Kernel Compile

In this chapter, you will understand the initial steps for Installing S3C BSP on Linux operating system. Kernel source is compressed by the name of **"s3c-linux-2.6.21.tar.bz2"**. Extract this bz2 file by executing the following command. After extracting the kernel tarball file **"s3c-linux-2.6.21"** directory will be generated. Go to **"s3c-linux-2.6.21"** directory and edit **".cross_compile"** file.

```
[root@localhost test]# tar jxvf s3c-linux-2.6.21.tar.bz2
[root@localhost test]# cd s3c-linux-2.6.21
[root@localhost s3c-linux-2.6.21]# vi .cross_compile
```

Here edit your cross compiler path on first line depending on your environment, in our case cross compiler path is "**/usr/local/arm/release-4.0.3/bin/arm-linux-".**
You can load default-configuration-file that is composed with values optimized to target board. In Kernel, default-configuration-files are located in **"s3c-linux-2.6.21"** directory.
Load configuration file executing following command. Configuration file can be found at the directory of "**s3c-linux-2.6.21/arch/arm/configs/"**.
You should use this board-specific default configuration file for the SMDK board you are working on. For SMDK6410 board,

```
[root@localhost s3c-linux-2.6.21]# make smdk6410mtd_defconfig
```

Then, type as follows:

```
[root@localhost s3c-linux-2.6.21]# make menuconfig
```

Select **'Device Drivers'** from the **'Linux Kernel Configuration'**.

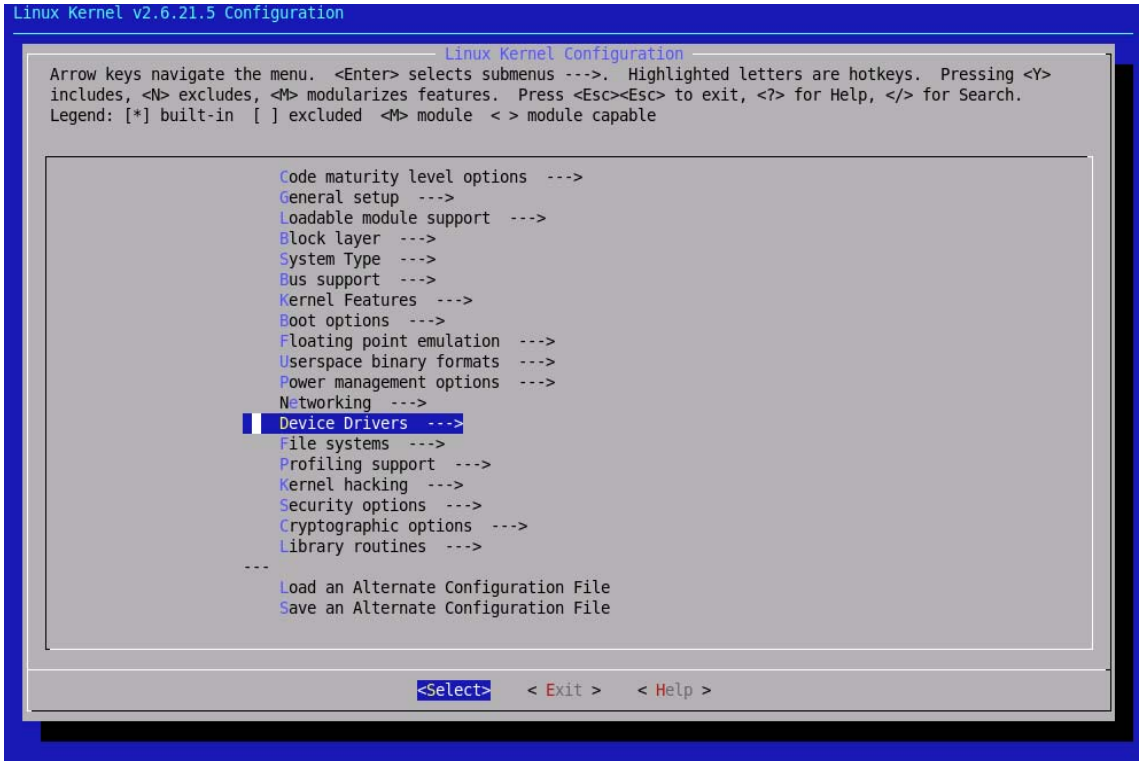


FIGURE 2 Device Drivers

Select **'Memory Technology Devices(MTD)'**

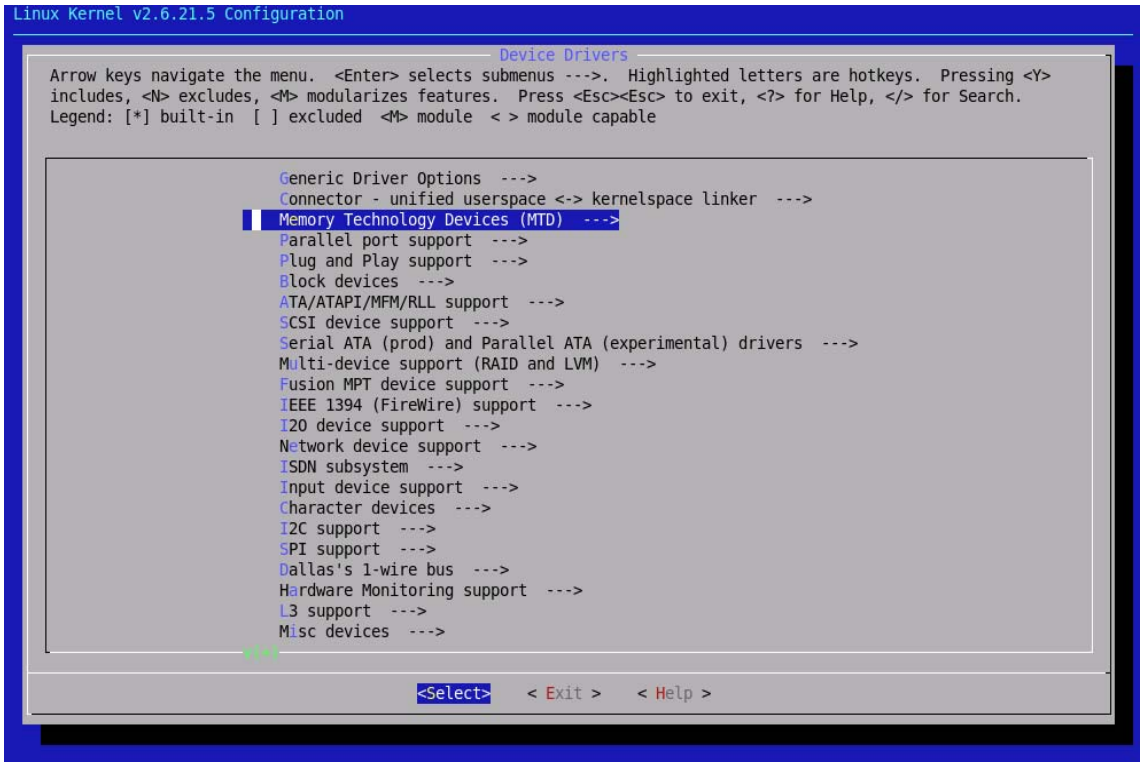


FIGURE 3 Memory Technology Devices

Select **'UBI – Unsorted block images'**

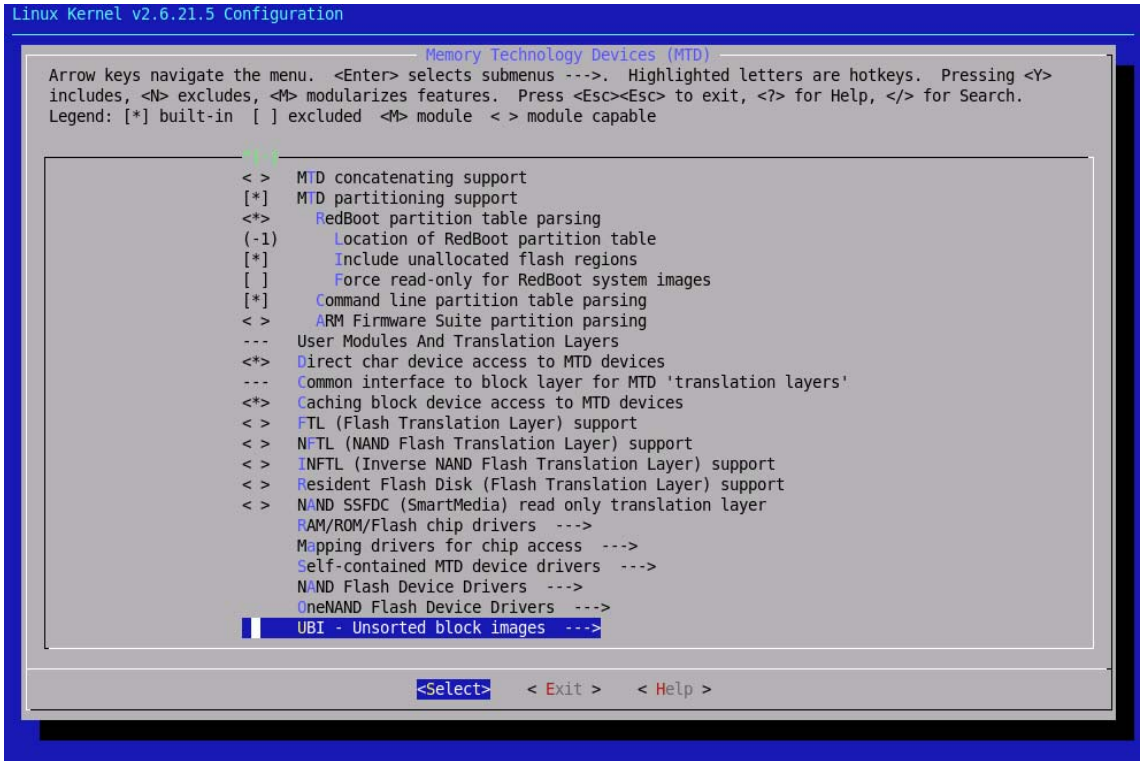


FIGURE 4 UBI – Unsorted Block Images

Select **'Enable UBI' to Module Type**

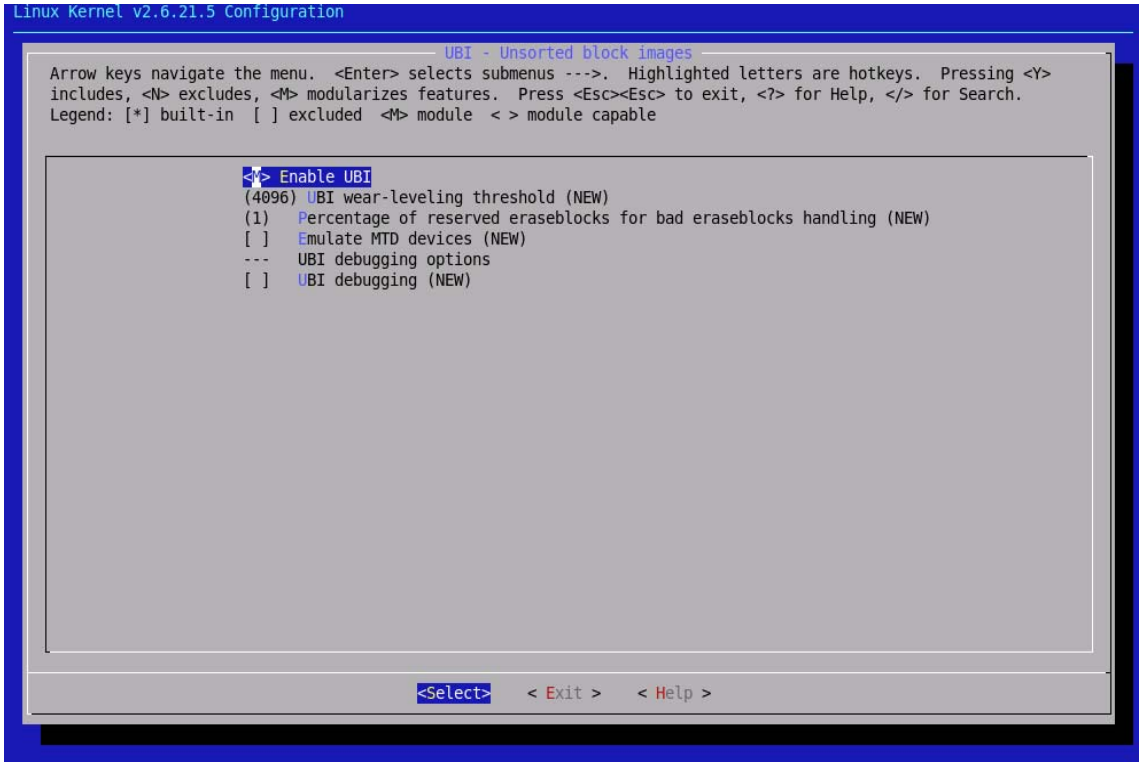


FIGURE 5 Enable UBI

Select **"Emulate MTD devices"**

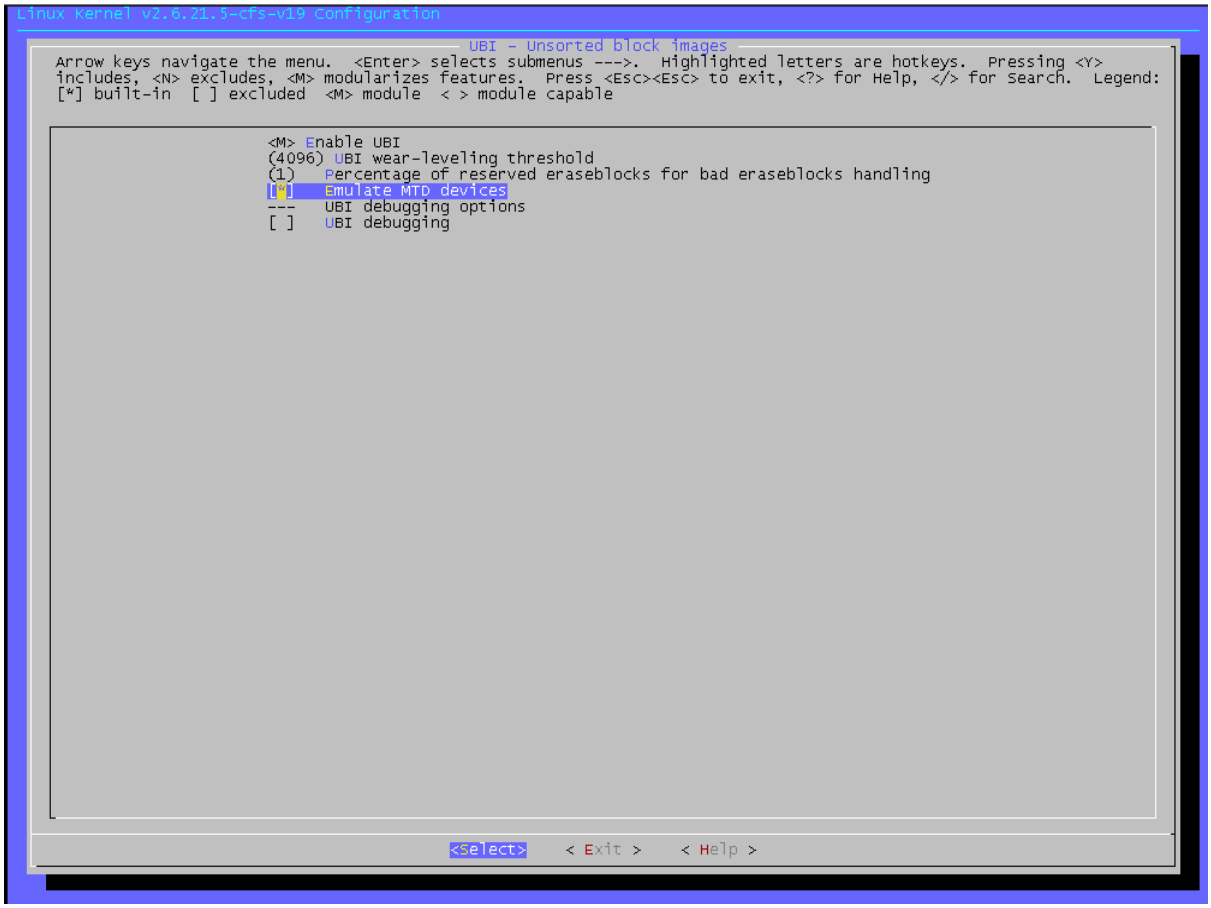


FIGURE 6 Select "Emulate MTD Devices"

Finally select *'Exit'* and then *'Yes'* to save your new kernel configuration.

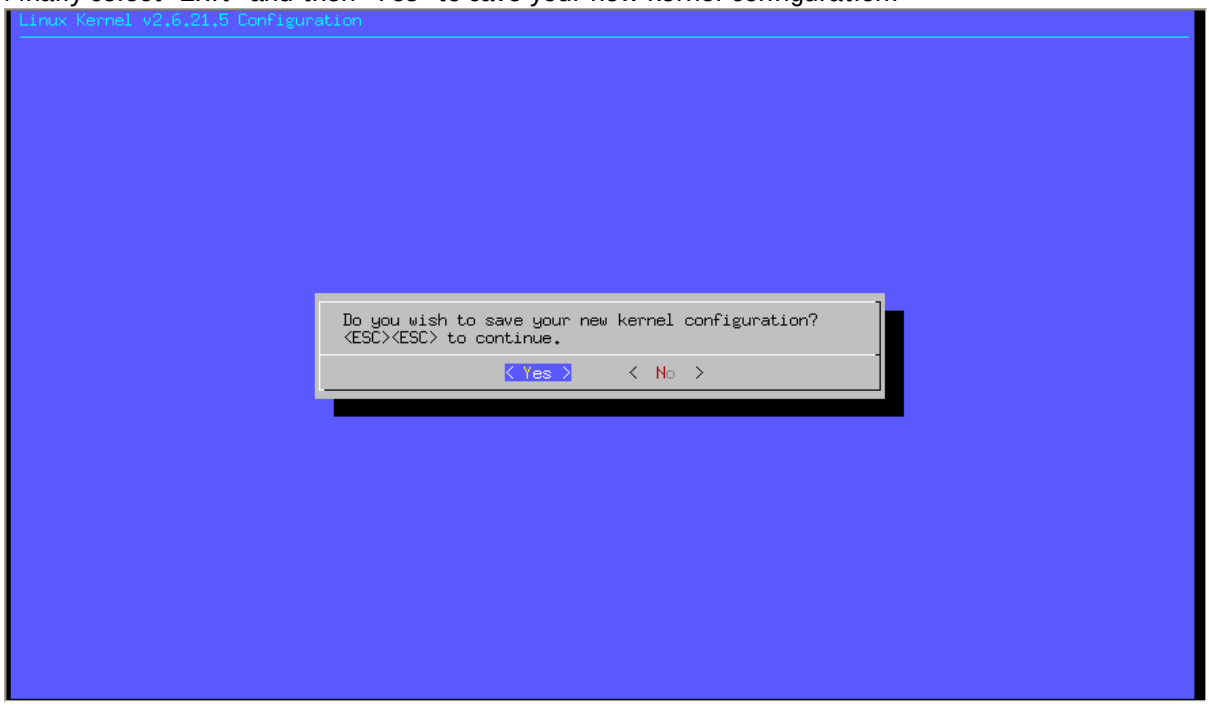


FIGURE 7 Saving New Kernel configuration

Execute the following commands on your host PC, the Kernel image will be created in **"s3c-linux-2.6.21/arch/arm/boot/"** directory by the name of **"zImage"**. Also, UBI Module will be created in "**s3c-linux-2.6.21/drivers/mtd/ubi/**"directory by the name of "**ubi.ko**".

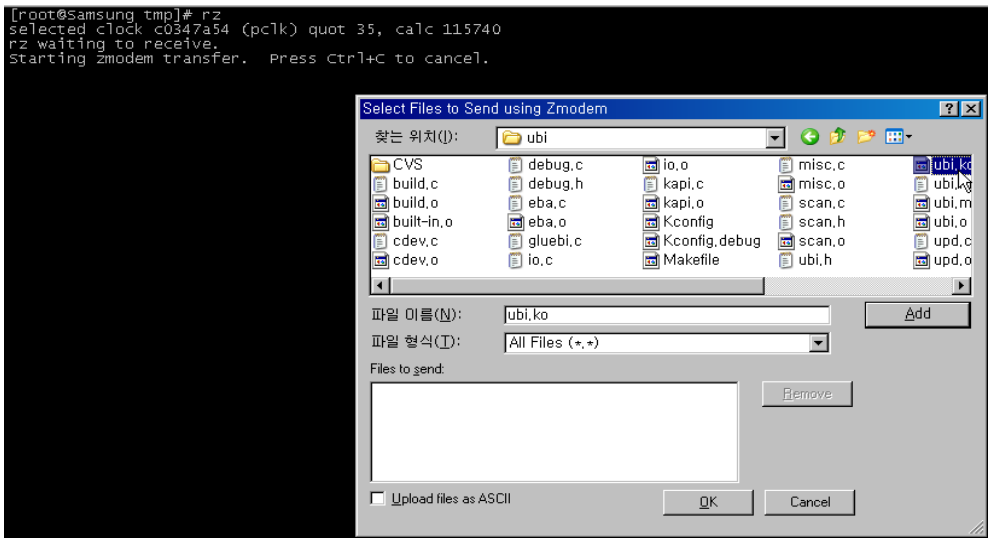
```
[root@localhost s3c-linux-2.6.21]# make
```

3 Test

In this Chapter, you will understand how to test UBI in you system.

3.1 Download “ubi.ko”

First, you have to download “ubi.ko” device module file that is created by make command. You can this device module in “linux/drivers/mtd/ubi/ubi.ko”.



3.2 Load UBI Module

Next, Insert UBI Module(ubi.ko) which is download from your host with insmod command. Input command as following.

```
[root@Samsung tmp]# insmod ubi.ko mtd=3
```

In this command, “mtd=3” is a MTD partition what you want to create UBI Volume. If you want to create UBI volume in another MTD partition, input number of partition. If you success insert UBI module(ubi.ko), you can see ubi information.

```
[root@Samsung tmp]# insmod ubi.ko mtd=3
UBI: empty MTD device detected
UBI: create volume table (copy #1)
UBI: create volume table (copy #2)
UBI: attached mtd3 to ubi1
UBI: MTD device name:            "File System"
UBI: MTD device size:            14 MiB
UBI: physical eraseblock size:   16384 bytes (16 KiB)
UBI: logical eraseblock size:    15360 bytes
UBI: number of good PEBs:        896
UBI: number of bad PEBs:         0
UBI: smallest flash I/O unit:    512
UBI: VID header offset:          512 (aligned 512)
UBI: data offset:                1024
UBI: max. allowed volumes:       89
UBI: wear-leveling threshold:    4096
UBI: number of internal volumes: 1
UBI: number of user volumes:     0
UBI: available PEBs:             885
UBI: total number of reserved PEBs: 11
UBI: number of PEBs reserved for bad PEB handling: 8
UBI: max/mean erase counter: 0/0
UBI: background thread "ubi_bgt0d" started, PID 939
[root@Samsung tmp]#
```

3.3 Create UBI Volume

Now, Make a UBI Volume in MTD partition. First of all, make “sysfs” directory and mount sysfs to sysfs directory.

```
[root@Samsung tmp]# mount sysfs sysfs ./sysfs
[root@Samsung tmp]# mount -t sysfs sysfs ./sysfs
```

You need ubimkvol application which is to make ubi volume. Download ubimkvol application. And Excute ubimkvol as following command.

```
[root@Samsung tmp]# ./ubimkvol -s 10245780 -N Storage -d 0
```

This Command is make UBI Volume in MTD partition. Volume size is 10245780 bytes and Volume name is “Storage” and device node is “ubi0”. If you want to know other option, you can see by input command “./ubimkvol –h”.

```
[root@Samsung tmp]# ./ubimkvol -s 10245780 -N Storage -d 0
UBI Volume's node is /dev/ubi0
UBI Volume make is Success!!
```

And, you want to confirm result, you can see proc file system.

```
[root@Samsung tmp]# cat /proc/mtd
dev:    size   erasesize  name
mtd0: 00040000 00004000 "Bootloader"
mtd1: 001c0000 00004000 "Kernel"
mtd2: 03000000 00004000 "Root - Cramfs"
mtd3: 00e00000 00004000 "File System"
mtd4: 009c9000 00003c00 "Storage"
```

3.4 Mount jffs2

Finally, you want to read and write in UBI Volume, system is mounted temporary directory to /dev/mtdblock4.

```
[root@samsung tmp]# mount -t jffs2 /dev/mtdblock4 mnt/ -o loop
JFFS2 write-buffering enabled buffer (512) erasesize (15360)
```