Freescale Semiconductor
Application Note

Document Number: AN3757
Rev. 0, 09/2008

Using MCF5329EVB NAND Flash to Host μClinux Root File System

by: Bruno Nunes and Renato Frias
Applications Engineering
Campinas, Brazil

This application note provides instructions on how to set up a standalone μClinux system on the MCF5329EVB, using NAND flash to host a JFFS2 root file system. The document shows how to use Freescale's Linux Target Image Builder (LTIB) to create a JFFS2 image which is copied to the NAND flash after system boots through the network file system (NFS). After the flash is programmed, we show how dBUG can be updated to support boot from flash. Using the steps provided in this document, the reader will have a system with dBUG and a μClinux kernel on the boot flash and the root file system on the NAND flash.

Contents

1 Introduction

This application note is based on ColdFire MCF5329 and its evaluation board (MCF5329EVB). A host computer using Linux with LTIB installed is also required. Fedora Core 7.0 was used to support the writing of this document, but different distributions may be used. In

case dBUG needs to be updated, a P&E Micro BDM and a Windows host are also needed.

1.1 Processor

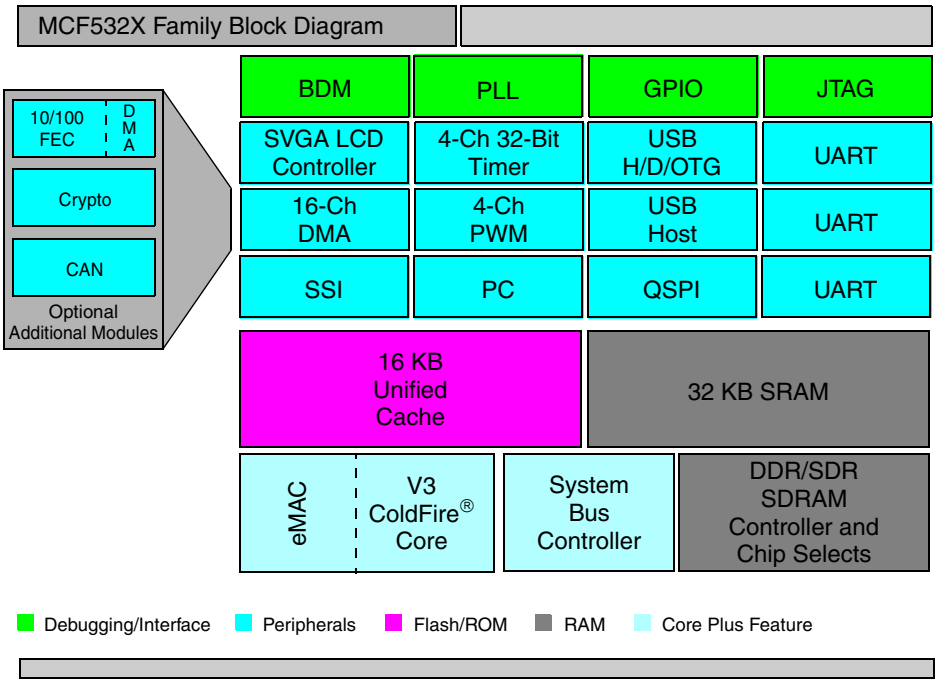

Figure 1. MCF5329 Block Diagram

The V3 ColdFire core of MCF5329 delivers up to 211 (Dhrystone 2.1) MIPS at 240 MHz, and includes these features:

- 32 KB RAM
- 16 KB I/D cache
- Enhanced MAC module (manages DSP-like instructions)

It has many integrated peripherals, such as:

- USB host/USB On-the-Go (OTG)
- FlexCAN 2.0B
- 10/100 Fast Ethernet controller (FEC)
- 3 UARTs
- I^2C bus interface
- QSPI
- Serial synchronous interface (SSI)
- 4 channels PWM
- Real time clock
- 16 channels DMA
- 16-bit DDR/32-bit SDR SDRAM controller

1.2 Evaluation Board

The MCF5329EVB evaluation system includes everything necessary to begin development with the MCF532x processor family. The development kit features the MCF5329-based Fire Engine, a plug-in system-on-module containing 32 MB of DDR-SDRAM, 16 MB of NAND flash, 2 MB of boot flash, an audio codec, a touch screen controller, and other interfaces. The baseboard, which contains the various serial ports and connectors, combined with the Fire Engine and best-in-class ColdFire debug module, offers a complete solution that enables fast design support for the MCF532x device family.

The MCF5329EVB comes with a complimentary Special Edition version of CodeWarrior Development Studio for ColdFire Architectures. Open source tool support, including μClinux operating system and Nano-X Window system, will also be available from Freescale. Comprehensive software and tool solutions are available through partnerships with a number of world-class embedded suppliers.

The evaluation board highlights the MCF5329 microprocessor, which contains a superset of the peripherals and interfaces available in the MCF532x device family.

1.3 Host Computer

MCF5329EVB BSP must be installed on the host computer. The latest release of this BSP can be downloaded at www.freescale.com.

Requirements for the host development system are:

- Ethernet
- Serial port
- 1 GB of free disk space
- NFS server
- TFTP server
- rsync
- Perl and software packages (as described later in this application note)

To build Linux target images, Freescale provides a tool in the BSP called the LTIB (Freescale GNU/Linux Target Image Builder) that makes the image-creating process easier. It is a tools framework used to manage, configure, extend, and build Linux software elements, making it easily possible to build a Linux target image and a root filesystem.

After downloading and saving the BSP iso image, you will need to install it. Open a terminal and run this command as root:

```
mount -o loop <your BSP iso folder>/<bsp iso file>.iso /mnt/
```

Then switch to a user account, and on /mnt run the install command: ./install. You will be prompted to define the LTIB installation folder. When install finishes, move to this folder and run the ./ltib command.

A permission problem may occur when running ./ltib for the first time. If that happens, use the visudo command to edit the sudoers file with vi. The sudoers file lists the users that have privileges to execute certain programs.

To edit sudoers as root, enter:

```
# /usr/sbin/visudo
```

When vi opens, type “i” to insert text, then add this line to the end of the sudoers file:

```
<user account name> ALL = NOPASSWD: /bin/rpm, /opt/freescale/ltib/usr/bin/rpm
```

To exit and save changes in vi press the Esc key to exit insert mode, then type “:wq” to write the file and quit the vi text editor.

In some cases, some Linux package updates are needed to complete LTIB installation. If it is needed, use the yum command on Fedora to download and install the missing packages, or the aptitude command on Debian-based distributions. For instance, on LTIB installation a message like the one below may be displayed:

```
Package                 Minimum ver   Installed info
-------                 -----------   ---------------
glibc-headers           0             not installed
glibc-devel             0             not installed
libstdc++-devel         0             not installed
gcc                     2.96          not installed
gcc-c++                 2.96          not installed
zlib-devel              0             not installed
rpm-build               0             not installed
ncurses-devel           0             not installed
bison                   0             not installed
```

To install these packages on a Fedora host open a terminal and type:

```
yum install glibc-headers
yum install glibc-devel
yum install libstdc++-devel
yum install gcc
yum install gcc-c++
yum install zlib-devel
yum install rpm-build
yum install ncurses-devel
yum install bison
```

2 NFS μClinux Boot on Target System

The Freescale application note AN3408, “Building a Sample CGI Application for the μClinux-Targeting ColdFire® MCF5329 Evaluation Board,” provides detailed information about installing and using the open-source μClinux distribution and LTIB-based μClinux distribution for the ColdFire MCF5329 evaluation board. Please refer to AN3408 to learn how to configure the environment to boot MCF5329 via network file system.

3 Generating a JFFS2 File System

After the system is booting via NFS, LTIB can be used to create a JFFS2 image to be programmed on the NAND Flash. Start LTIB by typing:

```
$ ltib -c
```

On the LTIB initial menu select “Target Image Generation Options.” On the next menu select JFFS2 as target image type and set the “jffs2 erase block in KB” option to 16.

After these selections are made, exit LTIB. After it runs, a JFFS2 image named jffs2.rootfs will be created in the LTIB installation folder. Figure 2 depicts the configuration to be applied. This file must be copied to a folder inside ltib_install_path/rootfs/ so the target can access it. The folder ltib_install_path/rootfs/boot will be used in this application note. In the ltib install folder, type:

```
$ cp jffs2.rootfs rootfs/boot
```

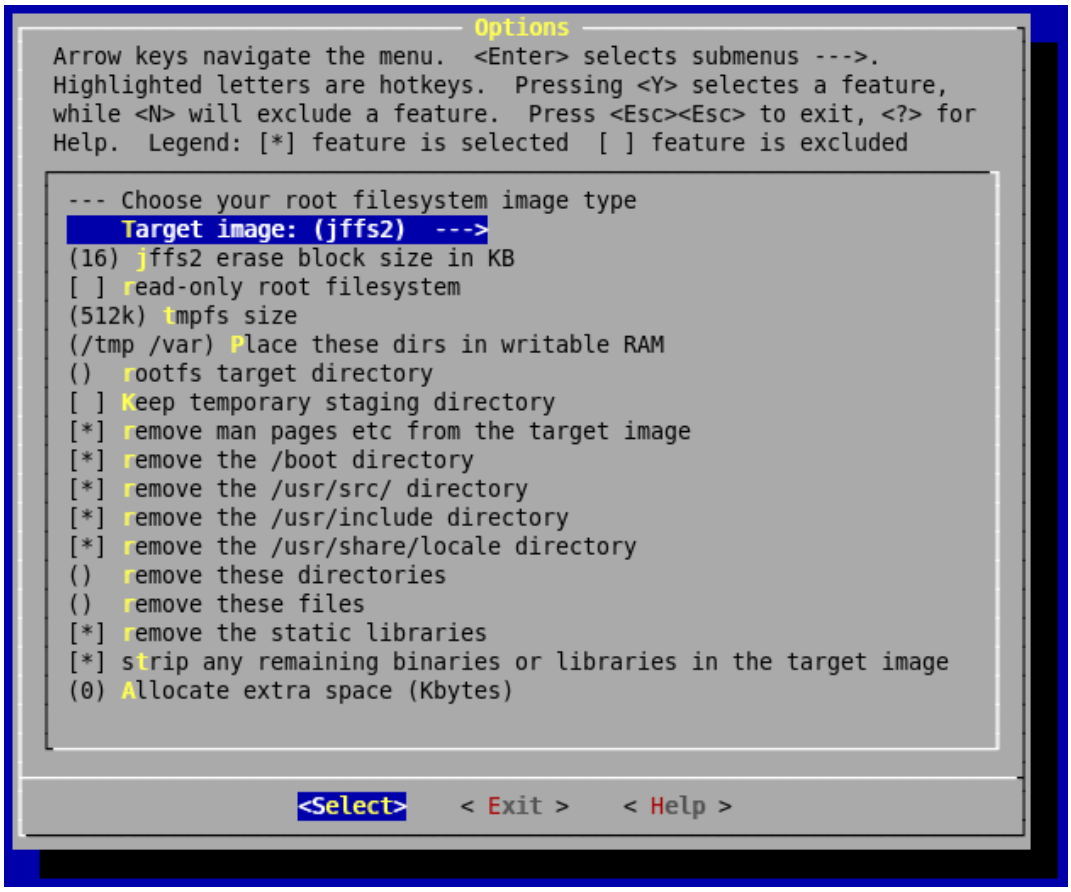

Figure 2. LTIB — Target Image Generation Options Screen

3.1 Erasing and Programming NAND Flash

As mentioned earlier, the NAND flash on MCF5329EVB will be used to host the Linux root file system. Thus, bootloader and kernel will remain in boot flash, but the Linux root file system will reside on NAND flash, allowing more space for OS applications and files.

The first step in doing this is to configure μClinux NAND flash support.

Run LTIB, $ ./ltib –c and select the checkbox “Configure the kernel.”

Next, on Linux Kernel Configuration, the “Device Drivers” option must be chosen to enable NAND flash support. Select the “Memory Technology Device (MTD) support” checkbox and at this submenu, select

"NAND Device Support," then select "NAND Flash device on M5329/M5373 board." These steps are shown in Figure 3.

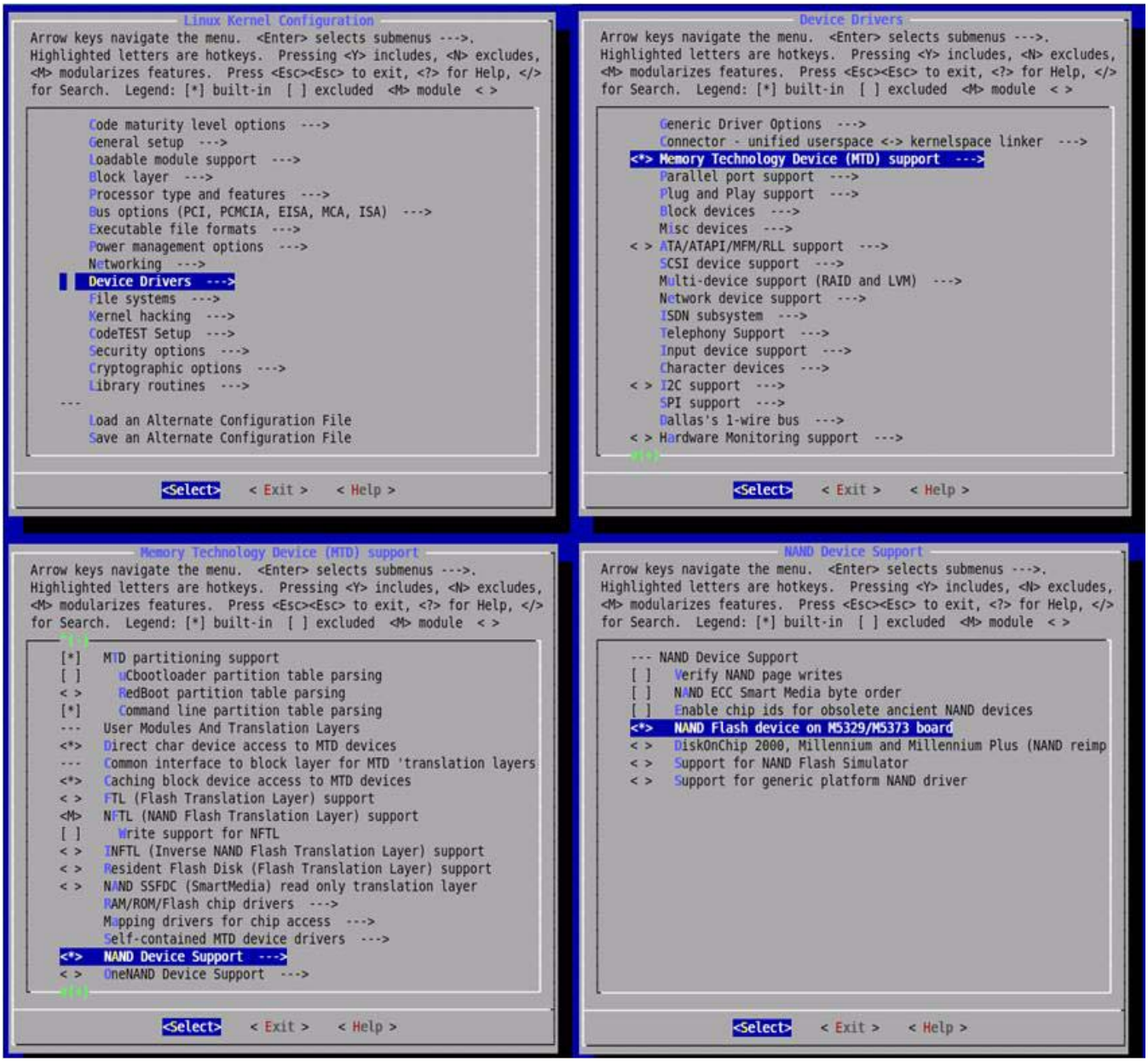

Figure 3. NAND Flash Support

Exit LTIB and boot the system through NFS as described in previous sections. If the NAND configuration was set properly, this message will be displayed when the system boots:

```
...
NAND device: Manufacturer ID: 0x20, Chip ID: 0x73 (ST Micro NAND 16MiB 3,3V 8-bit)
Scanning device for bad blocks
Creating 1 MTD partitions on "NAND 16MiB 3,3V 8-bit":
```

0x00000000-0x01000000 : “M53xx flash partition 1”
...

With the NAND flash working, the next step is to erase it. After the system boots, type:

```
(target) # /usr/bin/flash_eraseall /dev/mtd1
```

After NAND flash is erased we can copy the root file system to it.

```
(target) # cd /boot
(target) # cp rootfs.jffs2 /dev/mtdblock1
```

3.2 Creating a Compressed Kernel Image

The NOR flash on the standard CPU module has only 2 MB and some of it is used by the bootloader. For this reason it is useful to create a compressed image in the tftpboot directory.

The first step is to copy the latest generated kernel image to the tftpboot directory on the host machine:

```
$cd ltib_installation_path
$ cp rootfs/boot/vmlinux.bin /tftpboot
```

Now, to compress the image:

```
$ cd /tftpboot
$ gzip vmlinux.bin
```

The procedure above will create a compressed kernel image, vmlinux.bin.gz. Later in this document we will show how to use it to program the boot flash.

4 Bootloader — dBUG

dBUG is a monitor with a command line interface that can be used to load code on MCF5329EVB. For additional details check the dBUG firmware development document in the dBUG package in the docs folder. Later on in this section are instructions on how to extract the dBUG package from the LTIB.

dBUG is the default bootloader on MCF5329EVB. There are different versions of dBUG — in case an upgrade is needed, CFFlasher and P&E BDM Multilink may be used. The most recent version of CFFlasher is available at www.freescale.com. Please note that CFFlasher is a Windows application and not a Linux application.

To allow boot from flash, a recent version of dBUG must be used. To update dBUG on the board the steps in Section 4.1, “Building dBUG,” and Section 4.2, “Upgrading dBUG,” can be performed. If the dBUG version on your board already supports boot from flash you can skip to Section 5, “Making it Work.”

4.1 Building dBUG

First run LTIB on the host machine, $ ./ltib –c. In the LTIB menu select the “Build a boot loader” option. For bootloader choice pick dBUG, then exit the menu.

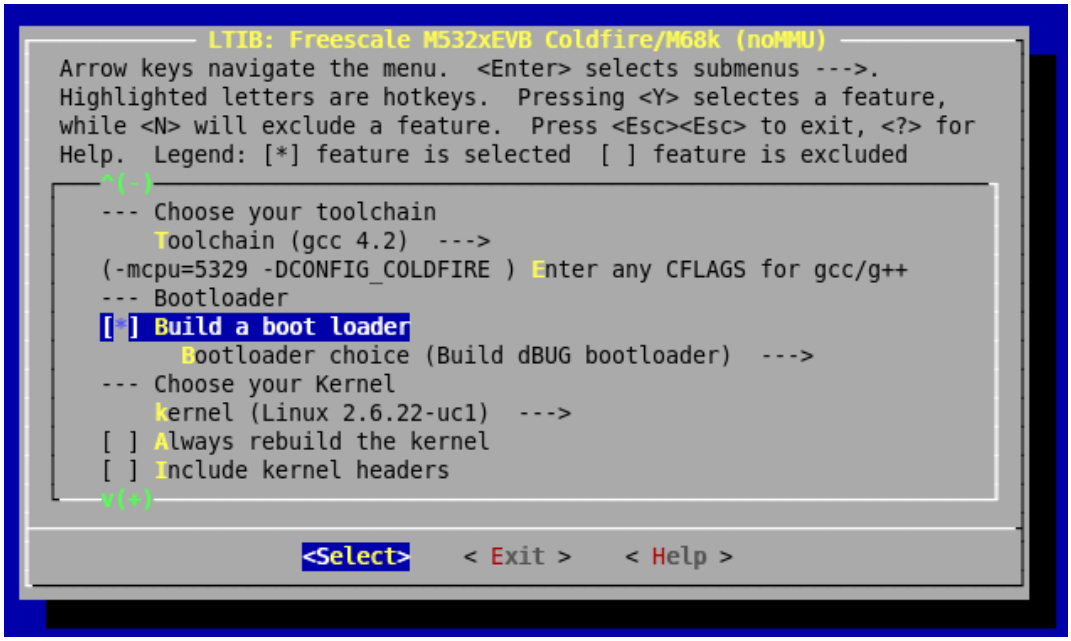

Figure 4. Building a Boot Loader

At this point, if you don't have dBUG on LPP (local package pool), LTIB will look for it on GPP (global package pool). Visit www.bitshrine.org for more information on LPP and GPP. If access to GPP is needed, the host computer must be connected to the Internet — otherwise the build will fail.

When LTIB finishes, a dBUG image, m5329evb_flash.s19 will be copied to the boot directory, <LTIB installation> /rootfs/boot. This file can be flashed on m5329_evb using CFFlasher.

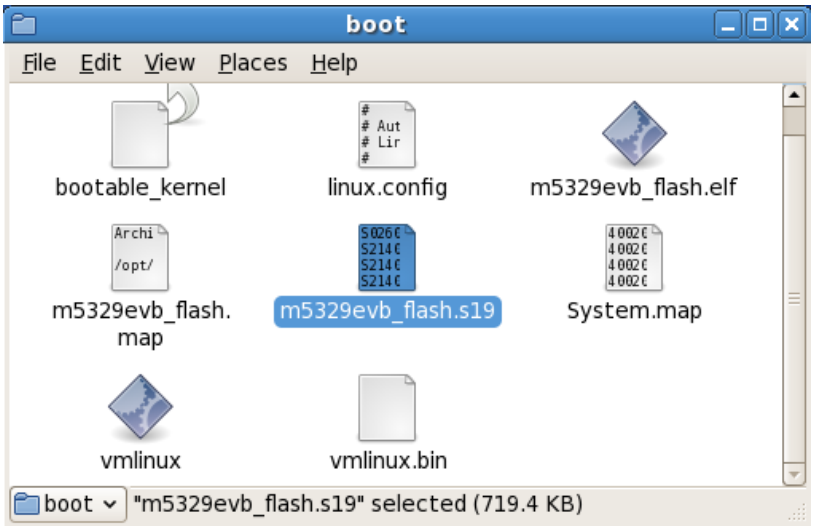

Figure 5. Boot Loader s19 File

If there is a need to modify dBUG code, you can use the command below to extract dBUG source code. On the host machine, in the LTIB installation folder, type:

```
$ ./ltib -p dbug -m prep
```

By executing this command LTIB will extract the dBUG package to <LTIB install path>/rpm/BUILD. Then the source code can be edited. As an example for this application note, a date was added on the dBUG prompt. After editing the code, you can build it by executing the command:

```
$ ./ltib -p dbug -m scbuild
```

To update the package status and copy the dBUG images to <LTIB installation> /rootfs/boot, enter:

```
$ ./ltib -p dbug -m scinstall
$ ./ltib -p dbug -m scbuild
```

Now that the dBUG image is ready we can flash the board using CFFlasher. This must be done on a Windows host.

4.2 Upgrading dBUG

Install and open CFFlasher, then choose the Target Config option.

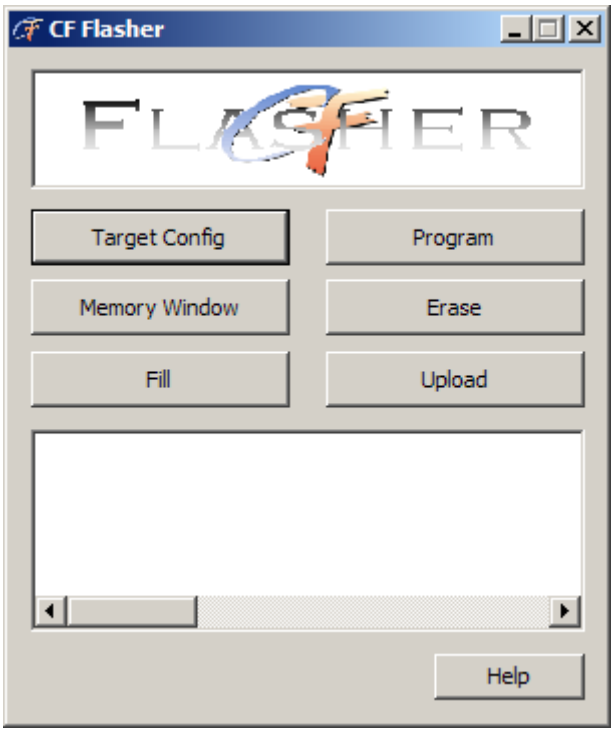

Figure 6. CFFlasher Window

In target configuration select MCF5329EVB. On BDM Communication, pick either USB or parallel port BDM. Click OK. Now select “Program” on the CFFlasher main screen, browse the image file in the File Select field, then click the Program button.

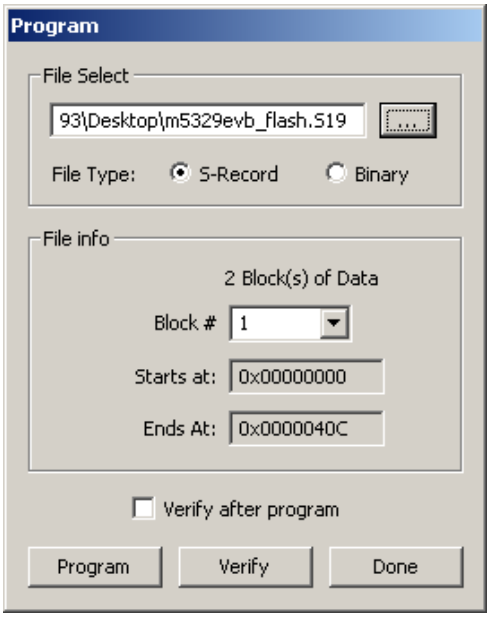

Figure 7. Flashing the Board

After flashing the board you can test if debug was updated by using the version command in dBUG, as shown in Figure 8.

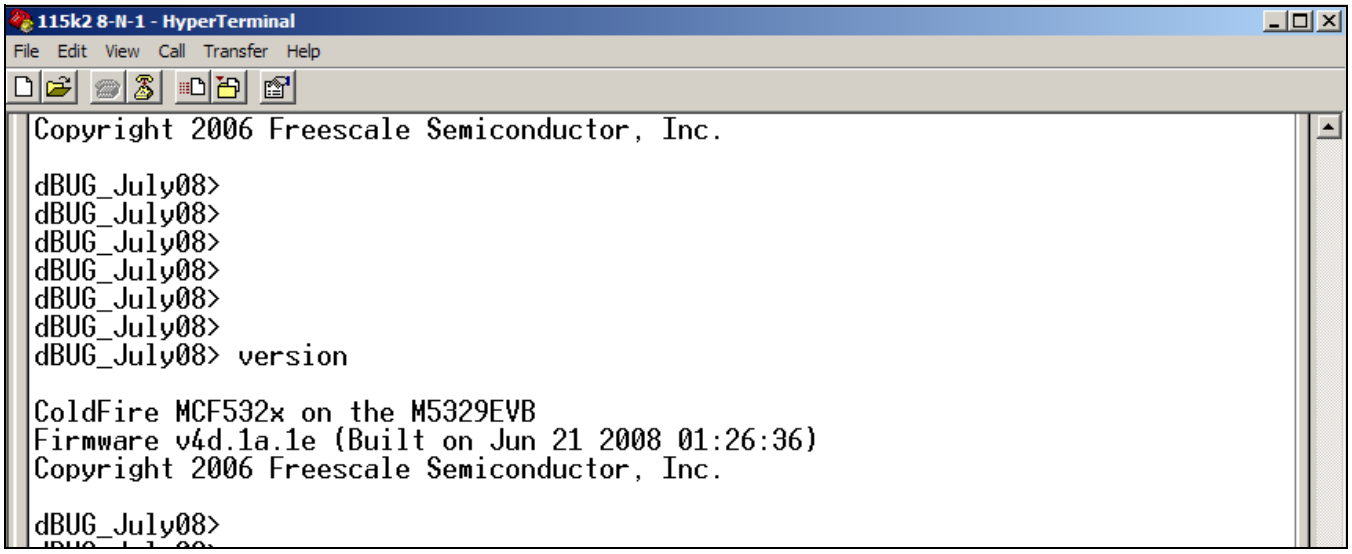

Figure 8. dBUG Version

5 Making it Work

Now that an updated version of dBUG is working, dBUG parameters must be set to allow boot from flash. To do this the set command may be used. For example, in the dBUG prompt, type:

```
dBUG> set <parameter> <value>
```

More specifically, autoboot must be set to “Autoboot from Flash” and kcl to root=/dev/mtdblock1. Also, the network configurations must be set to allow the compressed kernel transfer through TFTP (trivial transfer file protocol) from host to target.

```
dBUG> set autoboot flash
dBUG> set kcl root=/dev/mtdblock1
dBUG> set client <target IP address>
dBUG> set server <server IP address>
dBUG> set gateway <gateway address>
dBUG> set netmask <mask value>
```

An example configuration is shown in Figure 9.

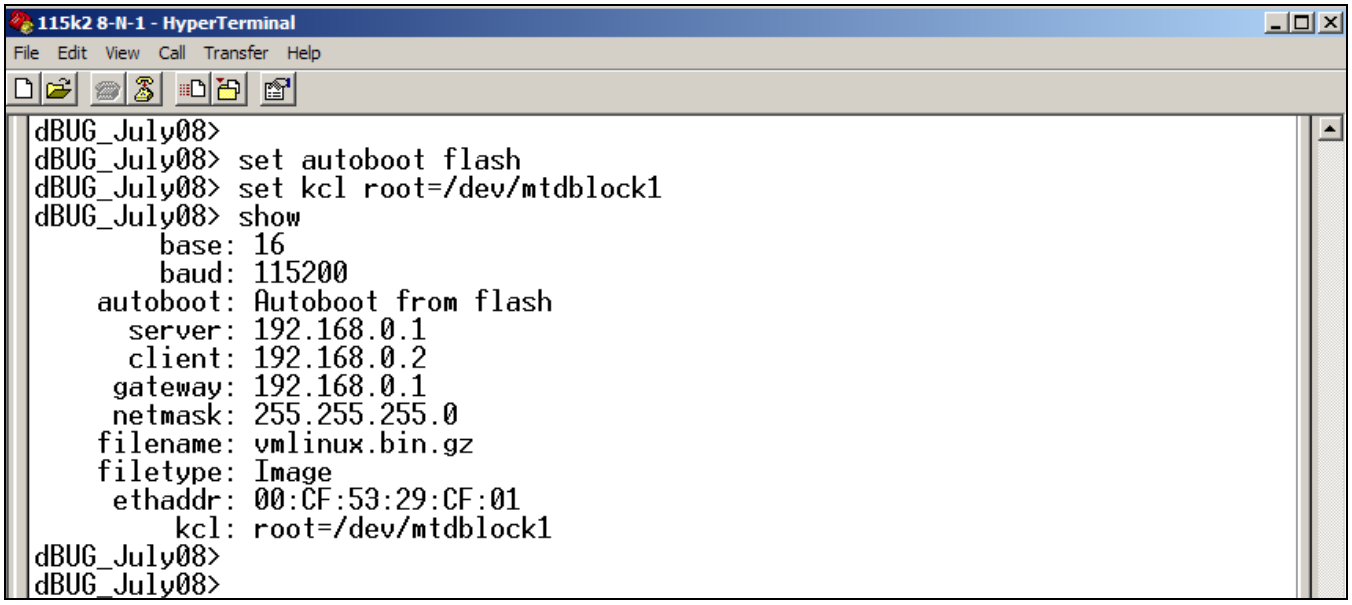

Figure 9. Configuring dBUG

After configuring dBUG parameters, the kernel must be transferred to the target and programmed in flash. At the dBUG prompt, type:

```
dBUG> dnfl
```

This command uses tftp to retrieve the vmlinux.bin.gz file from the host and write it to flash. It only works if TFTP service is working. TFTP was also required to boot using NFS, as mentioned in Section 2, “NFS µClinux Boot on Target System.”

AN3408 provides instructions on how to set the environment to allow the TFTP transfer.

After the kernel is programmed on the boot flash, the ethernet cable can be unplugged and the system can be reset. Now the processor will boot using the kernel from boot flash and the root file system from NAND flash.

6 Conclusion

This application note described how to configure LTIB to create a JFFS2 filesystem and how to program the NAND flash. It also provides instructions about updating the bootloader and programming a kernel image on boot flash, so the system can boot disconnected from a PC.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3757
Rev. 0
09/2008

freescale™
semiconductor