AVR32401: AVR32 AP7 Linux Interfacing DataFlash

8-bit AVR® Microcontrollers

Application Note

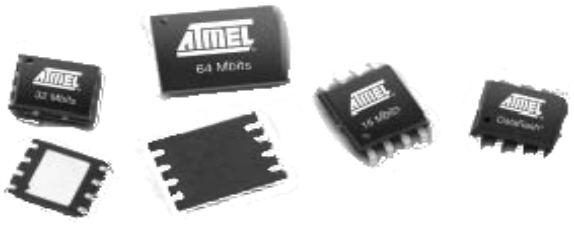
Features

- **JFFS2 file system**
- **Communication through SPI interface**

1 Introduction

This application note serves as an example of how to connect, set up and use DataFlash® from Linux® on the AVR®32. It shows the process of configuring the Linux kernel to set up the SPI communication channel and load the required modules, creating the file system and mounting the DataFlash to be a part of the Linux file system.

Figure 1-1. Atmel DataFlash



Rev. 32055B-AVR32-08/08

2 Hardware

The Serial Peripheral Interface (SPI) supports communication with external devices and memories, such as the DataFlash. The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO): This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK): This control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of baud rates; the SPCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): This control line allows slaves to be turned on and off by hardware.

The DataFlash can be connected to the SPI as shown in Figure 2-1. Hardware connections on page 2.

Figure 2-1. Hardware connections

3 Creating a JFFS v2 root file system

This chapter requires that *mtd-tools* is installed and that the binary *mkfs.jffs2* is available.

3.1 Flash parameters

All flash devices have a given erase size and page size. These must be known because the filesystem will be built in blocks depending on these values.

Before a block in the flash can be reprogrammed, it has to be erased. The erase size is given by the size of an *erase block*, which is the smallest unit that can be erased. Arbitrary bytes can not be written even if they have been properly erased. The smallest unit that can be written at once is a page and the size of a page is given by the page size.

3.2 The mkfs.jffs2 tool

In the *mtd-tools* software package there is a tool called *mkfs.jffs2*. This application is used to make an image from a given root directory.

mkfs.jffs2 should have the following parameters given when used:

- *--output*, where to put the image.
- *--root*, path to the root directory.
- *--big-endian* or *--little-endian*, defines the endianess of the image. AVR32 uses --big-endian.
- *--pagesize*, size of the pages on the flash, defaults to 4KiB.
- *--eraseblock*, size of the erase blocks on the flash, defaults to 64KiB.

For more information about *mkfs.jffs2*, start the application with the --help parameter.

3.3 Making the image

Example of copying a file system on the host computer to the flash on the target:

```
# mkfs.jffs2 --output=/home/user/images/rootfs.img
  --root=/home/user/target_root --big-endian --pagesize=1056
  --eraseblock=8448
```

This implies that the made image should output to */home/user/images/rootfs.img* and the directory */home/user/target root* will be used as the source for the image. The image will be made big endian and 64KiB erase block size and 4KiB page size.

This example conforms to a typical NOR flash, i.e. the AT49BV642D which is used on the ATSTK1000 and ATNGW100.

4 System Setup

The Linux kernel has a lot of drivers that can be loaded on startup or compiled as modules which can be loaded by the user during runtime. The Linux kernel comes with a default set of built-in drivers and modules, and it is up to the user to enable additional features.

4.1 Kernel Configuration

4.1.1 Enabling modules

To enable support for Memory Technology Devices (MTD), enter the Linux kernel source directory, and launch the configuration menu. This is done by executing the following command:

```
# make ARCH=avr32 CROSS_COMPILE=avr32-linux- menuconfig
```

This will open the kernel configuration window where all the modules and drivers can be selected. The options that needs to be enabled are shown in .

```
Device Drivers ->
      SPI support ->
            [X] “SPI support”
            [X] Atmel SPI Controller
      Memory Technology Devices (MTD) ->
```

```
                [X] Memory Technology Device (MTD) support
                [X] MTD partitioning support (1)
                [X] Direct char device access to MTD devices
                [X] Caching block device access to MTD devices
                RAM/ROM/Flash chip drivers -->
                      [X] Detect flash chips by Common Flash Inter... (2)
                      [X] Support for AMD/Fujitsu flash chips (2)
                Self-contained MTD device drivers ->
                      [X] Support for AT45xxx DataFlash
File Systems ->
      Miscellaneous filesystems ->
            [X] Journaling Flash File System v2 (JFFS2) support
```

[Notes] 1. Only needed for partitioning the DataFlash.

2. Not needed by DataFlash, but useful for other types of flash.

4.1.2 Setting up the SPI modules

The SPI controller needs to be set up with the proper chip select, clock frequency and it needs to be told which driver to pass the data on to.

To change these settings, the file linux/arch/avr32/boards/atstk1000/atstk1002.c needs to be modified by adding the following set of variables to the spi_board_info structure:

```
{
    .modalias           = "mtd_dataflash",
       .controller_data = (void *)GPIO_PIN_PB(2),
       .max_speed_hz    = 16000000,
       .bus_num         = 1,
    .chip_select        = 0,
},
```

The first parameter, *modalias*, is matched against the “name” member of registered device_driver structure, and tells the kernel which driver to use The “controller_data” parameter identifies which General Purpose I/O line to use for controlling the peripheral select signal to the SPI chip. The *bus_num* parameter specifies which SPI controller to use.

The last thing that needs to be enabled is the SPI controller 1. This is enabled in the function *atstk1002_init(void)*, in the same file as described earlier. The line:

```
at32_add_device_spi(1);
```

will have to be added to the list of devices to initialize.

4.1.3 Compiling the Kernel

To compile the new kernel with the above changes included, run the following command:

```
# make ARCH=avr32-linux CROSS_COMPILE=avr32-linux-
```

The new uImage file will now be located at *arch/avr32/boot/images/uImage*.

4.2 Device nodes

In order for the user to communicate with the DataFlash, the device files for the Memory Technology Devices (MTD) must be present in the device folder. The following set of commands will create these device nodes:

```
# mknod /dev/mtd0 c 90 0
# mknod /dev/mtd0ro c 90 1
# mknod /dev/mtd1 c 90 2
# mknod /dev/mtd1ro c 90 3
# mknod /dev/mtd2 c 90 4
# mknod /dev/mtd2ro c 90 5
# mknod /dev/mtd3 c 90 6
# mknod /dev/mtd3ro c 90 7
# mknod /dev/mtdblock0 b 31 0
# mknod /dev/mtdblock1 b 31 1
# mknod /dev/mtdblock2 b 31 2
# mknod /dev/mtdblock3 b 31 3
```

These device nodes will also be created with the *mdev* utility.

4.3 Mounting

To be able to write and read to and from the DataFlash, one or more partitions needs to be mounted. Create an empty directory, where the filesystem on the DataFlash will be mounted, and mount it with the following command:

```
# mount -t jffs2 /dev/mtdblock0 /mnt-directory
```

5 Using the DataFlash

The /mnt-directory now provides access to the DataFlash in an easy way – ordinary shell commands will work to read and write to the DataFlash.

```
# cd /mnt
# echo “Hello, DataFlash” > test.file
# cat test.file
Hello, DataFlash
# ls -l
-rw-r--r--    1 root     root           17 Jan  1 00:34 test.file
```

Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
avr32@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Request
www.atmel.com/literature