Application Report
SPRAAT3–February 2008

Making a Cramfs File System Image

Chase Maupin

ABSTRACT

This application report demonstrates how to build a compressed ROM file system (cramfs) file system image and add support for cramfs in the Linux kernel. The goal is to build a small file system that can be placed in Flash memory. Since cramfs is a read-only file system with limited size, it is mostly useful in embedded devices where there is a limited amount of storage and there is no need to write data back to the file system to be saved between reboots.

Contents

1 Overview of cramfs

The cramfs is a read-only Linux file system designed to have a small footprint. The main difference between cramfs and a compressed image of a traditional file system is that the cramfs file system can be used without the need to decompress it first. This is accomplished by compressing the file system one page at a time, which allows for random read access. However, this also prevents writing to the file system.

There are some additional limitations to cramfs aside from the read-only attribute. The maximum file size in a cramfs file system is 16MB. The maximum size of the total file system is a little over 256MB. This is because the last file on the file system must begin before the 256MB mark but may extend past the 256MB mark.

2 Adding cramfs to the Linux Kernel and Building the Kernel Image

The following steps assume that you have already installed the DVEVM software packages to obtain the ARM Linux tool chain, the Linux support package, and the example RAM disk file system. If you have not done so, see the *DVEVM Getting Started Guide* (SPRUE66) before continuing.

For your embedded system to be able to use the cramfs file system image that you create, the Linux kernel used by your system must know how to read from the cramfs file system. This requires enabling the cramfs file system support in the Linux kernel configuration. There are multiple ways to edit the kernel configuration, but for this example, xconfig is used. This example assumes that you have made a copy of the default installed kernel tree (from the DVEVM packages) in the /home/user/workdir directory.

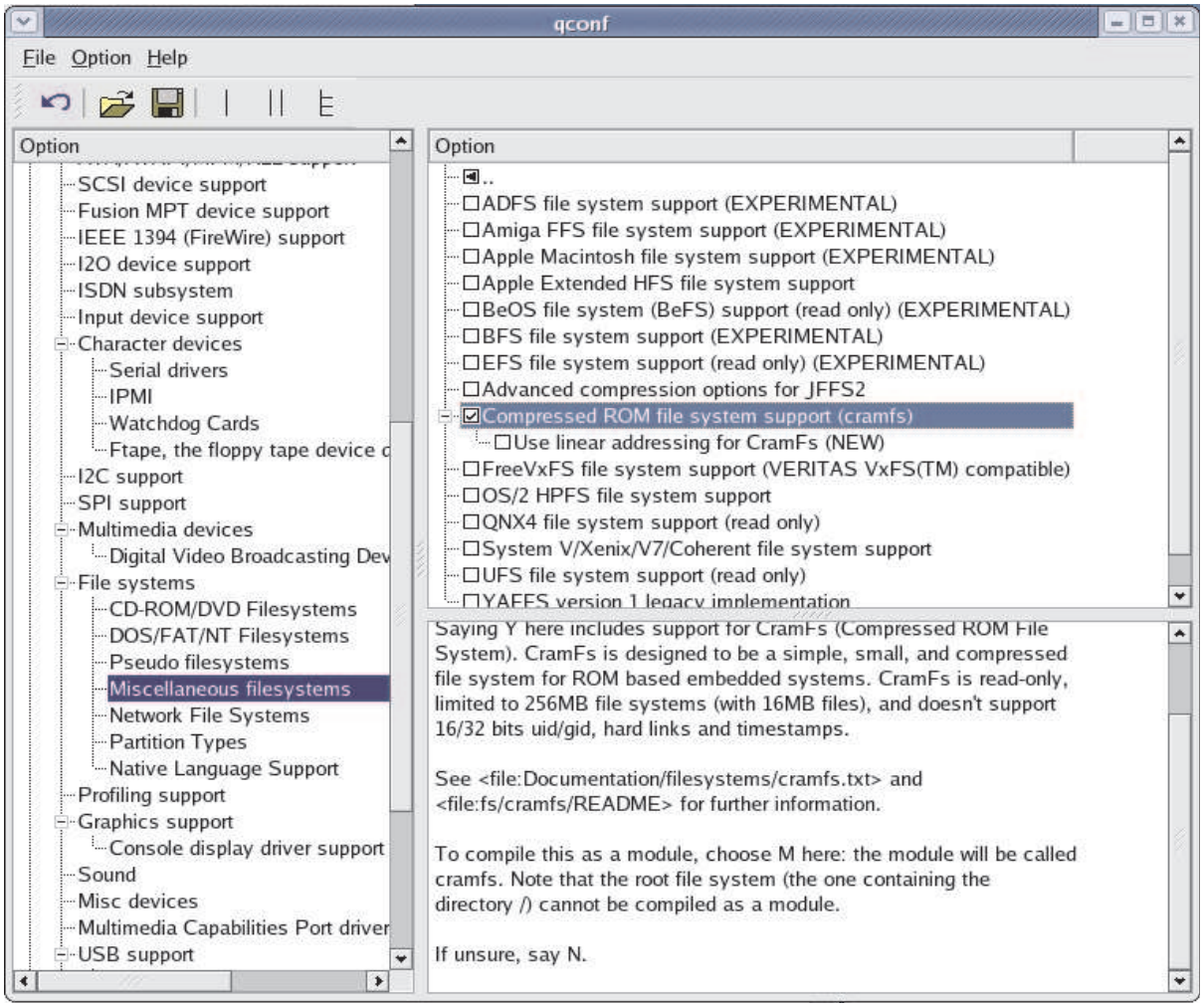
1. Change the directory to the base directory of the kernel tree:

```
host $ cd /home/user/workdir/ti-davinci
```

2. Launch the Linux kernel configuration utility:

```
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- xconfig
```

3. Check the Compressed ROM file system support (cramfs) box to enable the cramfs file system support in the Linux kernel under Device Drivers → File systems → Miscellaneous filesystems.



4. Continue configuring your kernel as needed for your system. For an example on making the smallest kernel possible, see *Building a Small Embedded Linux Kernel Example* (SPRAAH2).
5. Build the Linux kernel image with the following command:

```
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- uImage
```

3 Downloading and Installing Required Tools

The mkcramfs tool must be installed on your host Linux workstation to create the cramfs file system image. If you do not have the mkcramfs installed on your host system, you can obtain the latest version of the tool at http://sourceforge.net/projects/cramfs. After downloading the tools, in this example the tools are downloaded to the /home/user directory, you can perform the following steps to compile and install the tools.

1. Untar the following tools package:

```
host $ cd /home/user/

host $ tar xzf cramfs-1.1.tar.gz

host $ cd cramfs-1.1
```

2. Compile the tools

```
host $ make
```

   This creates the mkcramfs and cramfsck utilities. You can install both, but in most cases, only the mkcramfs utility is necessary.

3. Install the tools

   Since the Makefile does not have an install section, you have to install the utilities. This can be done by copying the tools to the /usr/bin directory.

```
host $ cp mkcramfs /usr/bin/

host $ cp cramfsck /usr/bin (optional)
```

4 Creating a cramfs File System Image

Although this section is not dependent on the previous section, it is assumed that you have already installed the DVEVM software on the Linux host machine according to the steps outlined in the *DVEVM Getting Started Guide* (SPRUE66).

To save time, a RAM disk is provided with the DVEVM ARM Linux software tool chain; use this as a base for your cramfs file system image. It is located under:

```
<tool chain install directory>/pro/devkit/arm/v5t_le/images
```

Note: Later releases of the DVSDK software package provide the sample RAM disk file system image at <dvsdk install dir>/<PSP dir>/bin

In this directory, the RAM disk file is called ramdisk.gz (about 2.1MB gunzipped). In run time, it occupies about 6.3MB in DDR. This file system contains some unnecessary utilities for this project, but is appropriate for a typical embedded system. The following steps demonstrate how to use this example file system to generate a cramfs file system image.

1. Copy the example initial RAM disk to a temporary location:

```
host $ mkdir –p /mnt/def_cd

host $ cp <tool chain install
dir>/pro/devkit/arm/v5t_le/images/ramdisk.gz /mnt/def_cd

host $ cd /mnt/def_cd
```

2. Mount the example initial RAM disk image:

```
host $ gunzip ramdisk.gz

host $ mkdir ram0

host $ mount ramdisk ram0 –o loop
```

3. Modify the example initial RAM disk. You can add new applications or files at this point. For a more detailed example on how to add a new application, see *Building a Small Embedded Linux Kernel Example* (SPRAAH2). For this example, a simple shell script is added to print *Hello World!!!* on start-up.

```
host $ vi ram0/etc/init.d/hello.sh


Add the following 2 lines into the hello.sh file:

#!/bin/sh

echo "Hello World!!!"


host $ chmod +x ram0/etc/init.d/hello.sh

host $ cd ram0/etc/rc.d/rcS.d

host $ ln –s ../init.d/hello.sh S50hello

host $ cd /mnt/def_cd
```

4. Create a cramfs image of the file system:

```
host $ mkcramfs ram0 cramfs.image
```

5. Un-mount the example initial RAM disk:

```
host $ umount ram0
```

5 Summary

This application report demonstrates how to create a cramfs image of a file system called cramfs.image. With this information, you can flash this image into the Flash memory on your board and use it as an initial ramdisk image (initrd) for the kernel you built in Section 2. For more information on flashing the kernel and cramfs file system images to the Flash on your board, see *Building a Small Embedded Linux Kernel Example* (SPRAAH2).

6 References

- *DVEVM Getting Started Guide* (SPRUE66)
- *Building a Small Embedded Linux Kernel Example* (SPRAAH2)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Clocks and Timers	www.ti.com/clocks	Digital Control	www.ti.com/digitalcontrol
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
RF/IF and ZigBee® Solutions	www.ti.com/lprf	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265