DVEVM Getting Started Guide

Literature Number: SPRUE66A
August 2006

TEXAS INSTRUMENTS

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 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.

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.

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
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
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
Low Power Wireless	www.ti.com/lpw	Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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

EVALUATION BOARD/KIT IMPORTANT NOTICE

Texas Instruments (TI) provides the enclosed product(s) under the following conditions:

This evaluation board/kit is intended for use for **ENGINEERING DEVELOPMENT, DEMONSTRATION, OR EVALUATION PURPOSES ONLY** and is not considered by TI to be a finished end-product fit for general consumer use. Persons handling the product(s) must have electronics training and observe good engineering practice standards. As such, the goods being provided are not intended to be complete in terms of required design-, marketing-, and/or manufacturing-related protective considerations, including product safety and environmental measures typically found in end products that incorporate such semiconductor components or circuit boards. This evaluation board/kit does not fall within the scope of the European Union directives regarding electromagnetic compatibility, restricted substances (RoHS), recycling (WEEE), FCC, CE or UL, and therefore may not meet the technical requirements of these directives or other related directives.

Should this evaluation board/kit not meet the specifications indicated in the User's Guide, the board/kit may be returned within 30 days from the date of delivery for a full refund. THE FOREGOING WARRANTY IS THE EXCLUSIVE WARRANTY MADE BY SELLER TO BUYER AND IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE.

The user assumes all responsibility and liability for proper and safe handling of the goods. Further, the user indemnifies TI from all claims arising from the handling or use of the goods. Due to the open construction of the product, it is the user's responsibility to take any and all appropriate precautions with regard to electrostatic discharge.

EXCEPT TO THE EXTENT OF THE INDEMNITY SET FORTH ABOVE, NEITHER PARTY SHALL BE LIABLE TO THE OTHER FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES.

TI currently deals with a variety of customers for products, and therefore our arrangement with the user **is not exclusive**.

TI assumes **no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein**.

Please read the User's Guide and, specifically, the Warnings and Restrictions notice in the User's Guide prior to handling the product. This notice contains important safety information about temperatures and voltages. For additional information on TI's environmental and/or safety programs, please contact the TI application engineer or visit www.ti.com/esh.

No license is granted under any patent right or other intellectual property right of TI covering or relating to any machine, process, or combination in which such TI products or services might be or are used.

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

FCC Warning

This evaluation board/kit is intended for use for **ENGINEERING DEVELOPMENT, DEMONSTRATION, OR EVALUATION PURPOSES ONLY** and is not considered by TI to be a finished end-product fit for general consumer use. It generates, uses, and can radiate radio frequency energy and has not been tested for compliance with the limits of computing devices pursuant to part 15 of FCC rules, which are designed to provide reasonable protection against radio frequency interference. Operation of this equipment in other environments may cause interference with radio communications, in which case the user at his own expense will be required to take whatever measures may be required to correct this interference.

Preface

About This Guide

The DVEVM (Digital Video Evaluation Module) is an evaluation platform that showcases the DM644x architecture and lets users evaluate the power and performance of the DM644x as a Multimedia engine. The intended audience is the user who is developing Linux-based software on the DM644x ARM core.

The DVEVM does not expose the DSP core for software development, but rather treats it as a "black box" for running off-the-shelf codecs. The DVSDK upgrade allows you to add the ability to develop applications for the DSP side. In addition, the DVSDK adds a full Linux license.

This guide gives you overview information about the board and the software provided with the board. It is intended to be used as the initial "getting to know you" document for the DVEVM. Other documents provide more in-depth information. See the DVEVM documentation index for a complete list of documents that have been included with the product.

Additional Documents and Resources

You can use the following sources to supplement this user’s guide:

- ❏ DaVinci EVM Home at Spectrum Digital:
  http://c6000.spectrumdigital.com/davincievm/
- ❏ TI Linux Community for DaVinci Processors:
  http://linux.davincidsp.com
- ❏ *Codec Engine Application Developer's Guide* (SPRUE67)

Notational Conventions

This document uses the following conventions:

- ❏ Program listings, program examples, and interactive displays are shown in a `mono-spaced font`. Examples use **`bold`** for emphasis, and interactive displays use **`bold`** to distinguish commands that you enter from items that the system displays (such as prompts, command output, error messages, etc.).
- ❏ Square brackets ([and]) identify an optional parameter. If you use an optional parameter, you specify the information within the

brackets. Unless the square brackets are in a **bold** typeface, do not enter the brackets themselves.

Trademarks

The Texas Instruments logo and Texas Instruments are registered trademarks of Texas Instruments. Trademarks of Texas Instruments include: TI, DaVinci, the DaVinci logo, XDS, Code Composer, Code Composer Studio, Probe Point, Code Explorer, DSP/BIOS, RTDX, Online DSP Lab, DaVinci, TMS320, TMS320C54x, TMS320C55x, TMS320C62x, TMS320C64x, TMS320C67x, TMS320C5000, and TMS320C6000.

MS-DOS, Windows, and Windows NT are trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Solaris, SunOS, and Java are trademarks or registered trademarks of Sun Microsystems, Inc.

All other brand, product names, and service names are trademarks or registered trademarks of their respective companies or organizations.

Contents

This appendix describes optional procedures you may use depending on your setup and specific needs.

Chapter 1

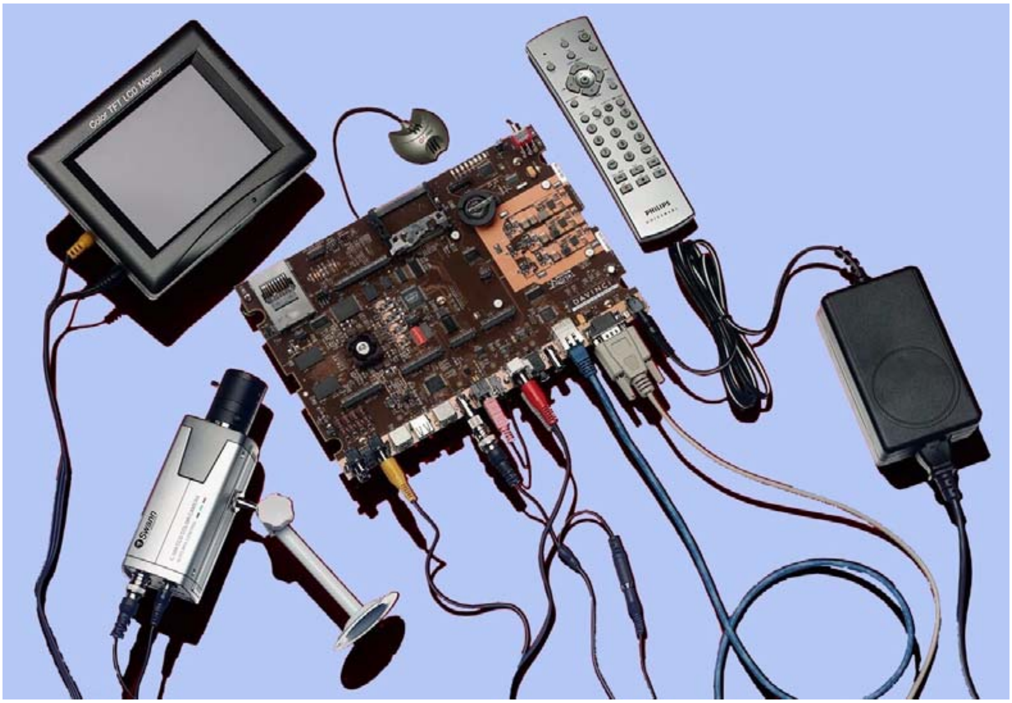
DVEVM Overview

This chapter introduces the DVEVM (Digital Video Evaluation Module).

1.1 Welcome!

Your new DVEVM (Digital Video Evaluation Module) will allow you to evaluate TI's new DaVinci^{TM} Technology and the DM644x architecture.

This technology brings together system-solution components tailored for efficient and compelling digital video and audio.



The intended audience is a user developing Linux-based software on the DM644x ARM core. The DVEVM does not expose the DSP core for software development, but rather treats it as a "black box" for running off-the-shelf codecs. Separate upgrade kits from Texas Instruments will provide tools and software for DSP development.

1.2 What’s in this Kit?

Your DVEVM kit contains the following hardware items. Section 2.1, *Setting Up the Hardware* tells how to connect these components.

- **DVEVM Board** (SDI P/N 702050). This board contains a DaVinci TMS320DM6446 dual-core device with an ARM9 and C64+ DSP for development of applications that use both a general-purpose processor and an accelerated DSP processor.
- **Hard Disk Drive** (Samsung P/N MP0402H). The hard drive provided with the DVEVM is a 2.5" Spinpoint drive with 40 GB of storage. The drive speed in 5400 RPM and it has an 8MB cache. The drive is an Ultra ATA 66/100/133 IDE. Software is preloaded on the DVEVM board’s hard disk drive.
- **CCD Camera** (Swann P/N SW-C-C500R). This camera provides NTSC or PAL video imaging for DaVinci applications.
- **LCD Display** (Delvcam P/N DELVPRO56 (NTSC) / DELVPRO56PL (PAL)). The Delvcam LCD display provided with the DVEVM kit has a 5.6" screen and 320x240 pixels. Cables and a power supply are provided. The NTSC version supports has a 110 VAC power supply. The PAL version has a 220 VAC power supply.
- **PC Desktop Microphone** (Labtec Verse 333). The microphone provides a way to capture audio for use by DaVinci applications.
- **IR Remote Control** (Phillips Magnavox P/N PM4S EFM7ND). This universal remote control is included to provide a user interface to the demo applications.
- **A/V Cables.** Cables used to connect the DVEVM board to peripheral devices and to a host Linux workstation used for development are provided in the kit.

The DVEVM kit also comes with the following software CDs. Information about how to use the software components is provided in Chapter 4.

- DaVinci Digital Video Evaluation Kit.
- TI DaVinci Demonstration Version of MontaVista Linux Pro v4.0 Target.
- TI DaVinci Demonstration Version of MontaVista Linux Pro v4.0 Tools.
- Additional software CDs are provided with DVDP upgrade kits.

DVDP

1.3 What's on the Board?

The DVEVM comes loaded with peripherals your multimedia applications may need to make use of. The hard drive on the board also comes preloaded with demonstration software. The following block diagram shows the major hardware components.

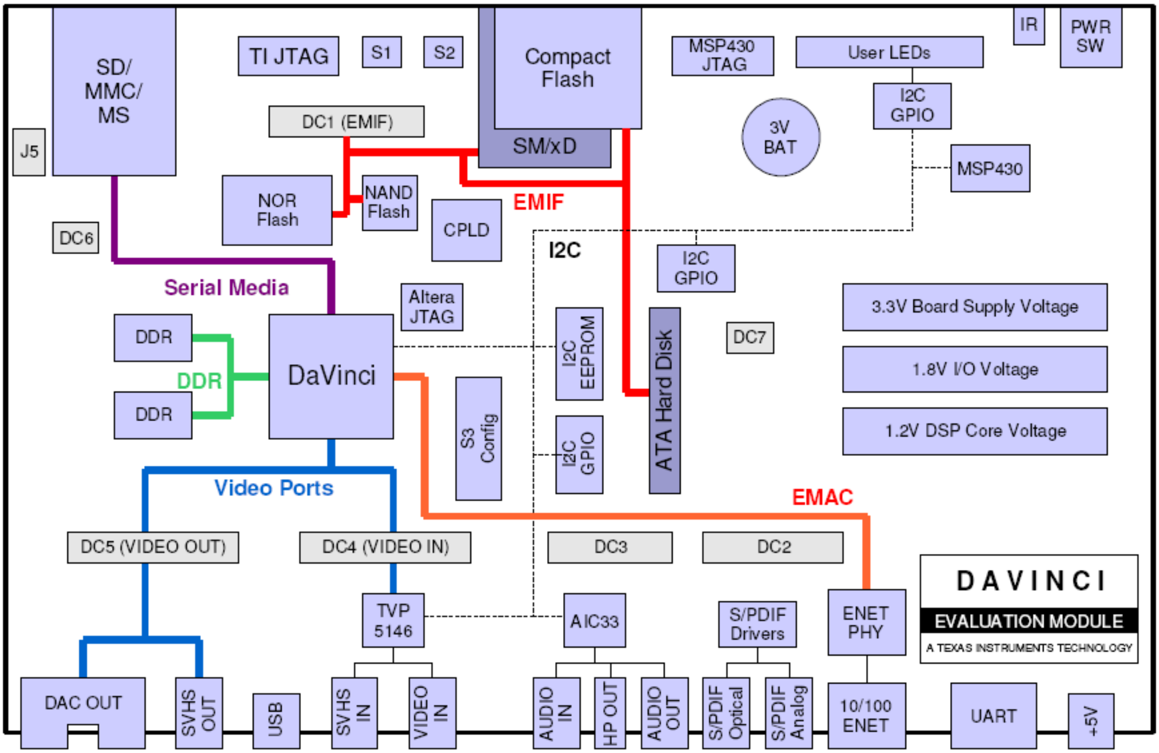


Diagram provided courtesy of Spectrum Digital Inc.

Figure 1–1 DVEVM Hardware Block Diagram

For more information about the DVEVM hardware, see the DaVinci EVM website at http://c6000.spectrumdigital.com/davincievm/.

The DaVinci EVM incorporates a battery holder to provide backup power to the MSP430's real-time clock when the power is not applied to the board. The battery is not included in the kit. See the Spectrum Digital DaVinci EVM Technical Reference for suggested battery part numbers.

1.4 What’s Next?

To get started evaluating the DVEVM and developing applications for the DM644x, begin by using this Getting Started guide. It will step you through connecting the hardware, testing the software, and beginning to develop applications.

When you are ready for more information about DaVinci Technology and the DM644x architecture, see the following:

❏ DaVinci EVM Home at Spectrum Digital:
http://c6000.spectrumdigital.com/davincievm/

❏ TI Linux Community for DaVinci Processors:
http://linux.davincidsp.com

❏ *Codec Engine Application Developer's Guide* (SPRUE67)

❏ Other PDF documents on the CDs included with the DVEVM

Chapter 2

DVEVM Hardware Setup

This chapter tells you how to set up the DVEVM hardware.

Topic **Page**

2.1 Setting Up the Hardware

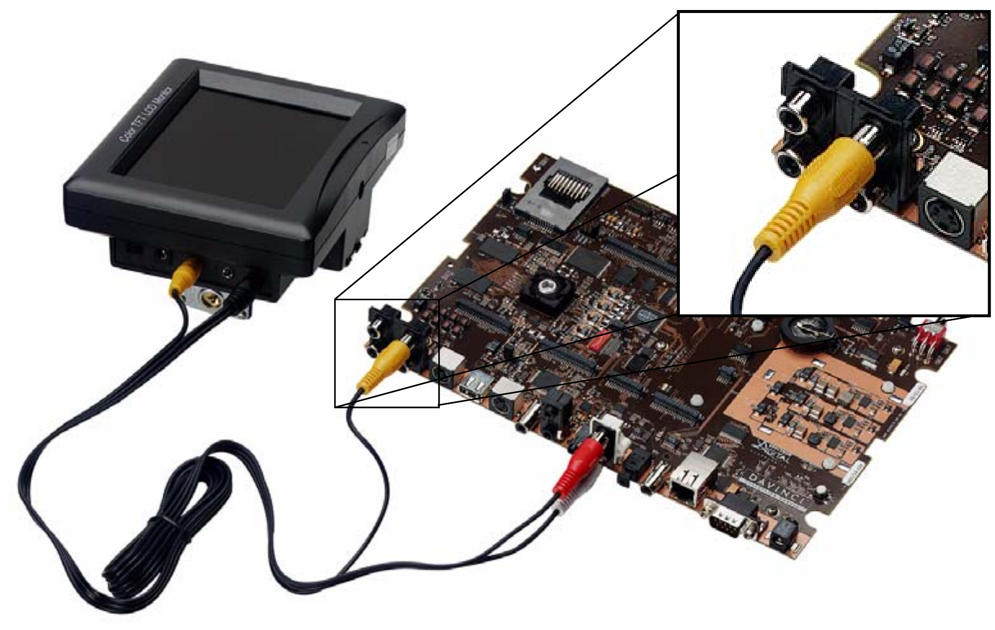
To set up the hardware provided with the DVEVM, use the steps in the sections that follow. You may skip sections if you do not need to access a particular peripheral. For example, if you do not need to use the serial cable, skip that section.

1) The DVEVM is sensitive to static discharges. Use a grounding strap or other device to prevent damaging the board.

   Be sure to connect communication cables before applying power to any equipment.

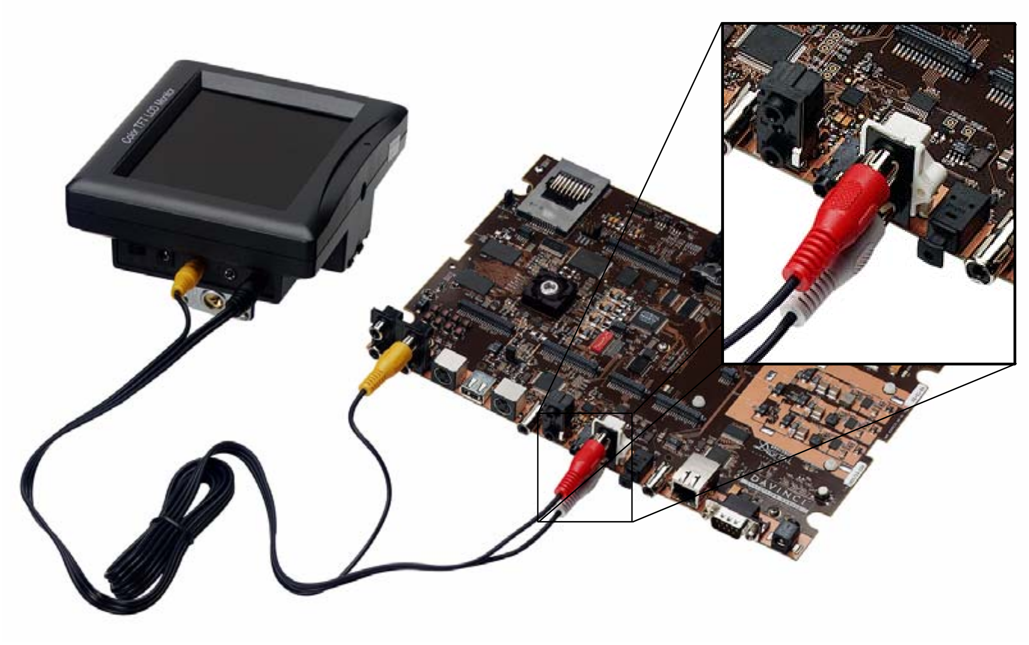
2) If you use PAL video, set switch 10 on the S3 (USER) bank of switches to On. If you use NTSC video, set this switch to Off. See Figure 1–1 for S3 switch bank location.

3) Connect the yellow video cable to the upper-right Video Out jack on the DVEVM and the LCD display Video Input as shown below.



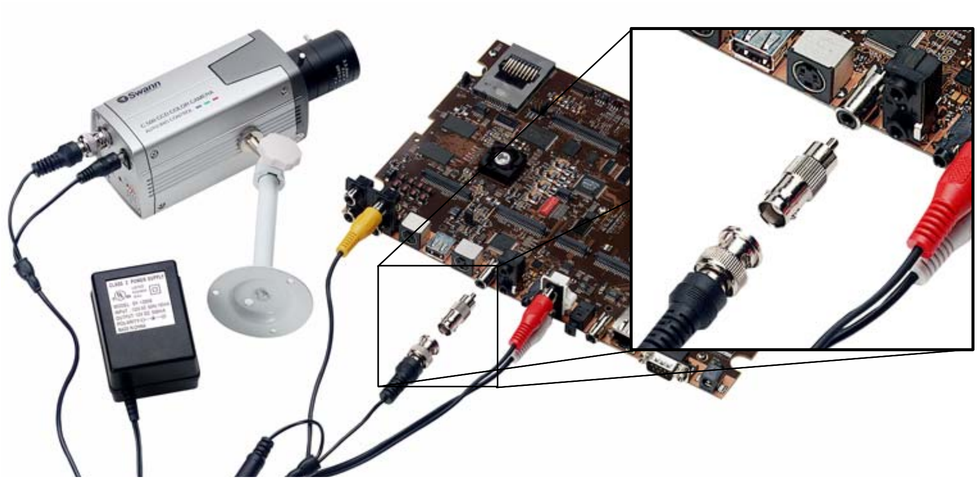
See Section A.2, *Changing the Video Input/Output Methods* for information about using S-Video or Component video.

4) Connect the red and white audio cables to the DVEVM Audio Output and the LCD display R/L Audio Input jacks as shown below:



5) Connect the BNC-to-RCA connector to the coax cable. Then connect the coax cable to the video camera and the DVEVM Video Input.

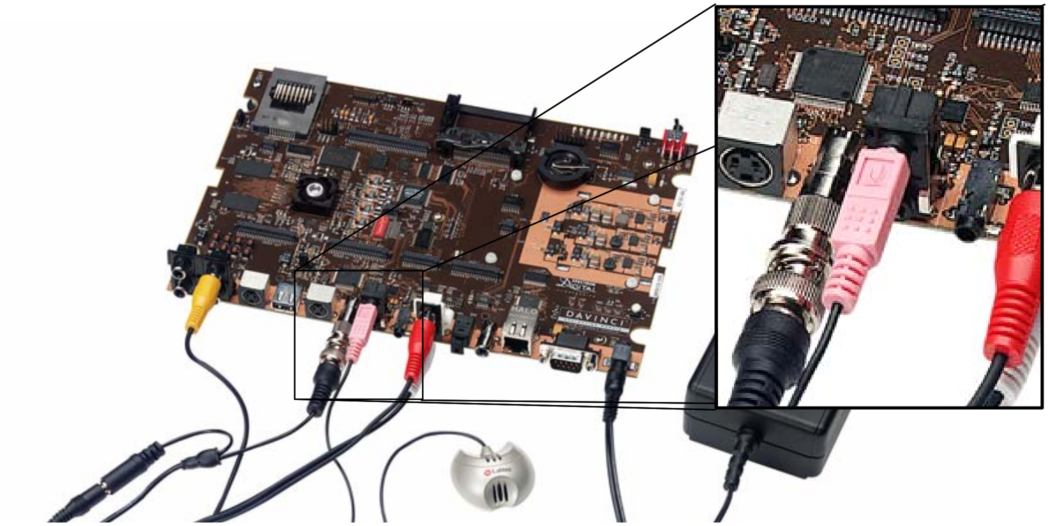
6) Connect the power jack for the video camera. To be ESD safe, do not plug in the other end of the camera power cord until the later step that instructs you to do so.



See Section A.2, *Changing the Video Input/Output Methods* for information about using S-Video or Component video.

7) Connect the microphone to the DVEVM.

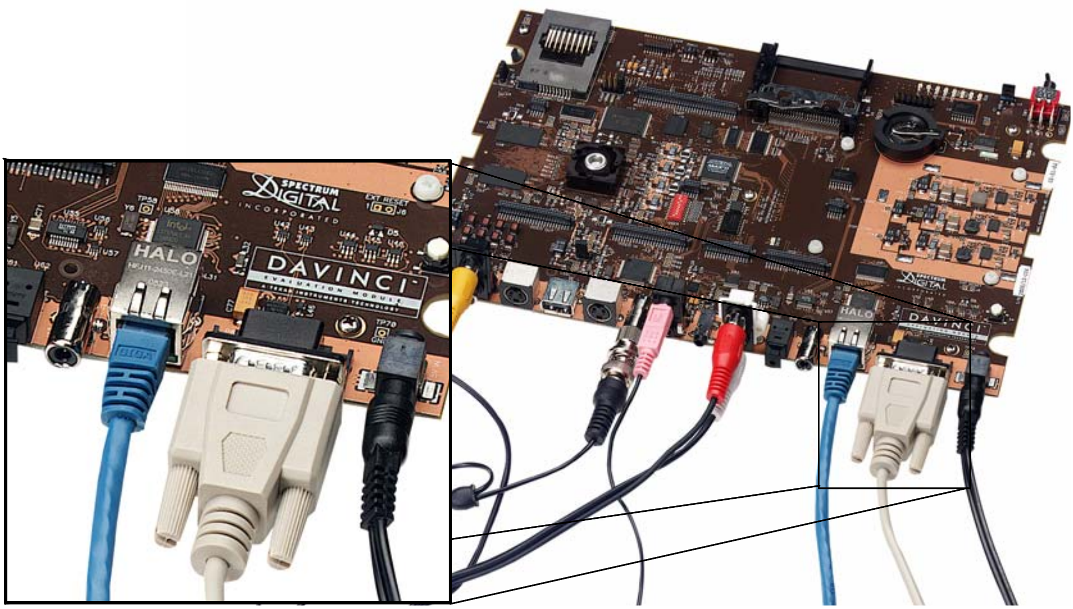
8) Connect the power cable to the DVEVM Power Jack on the board. To be ESD safe, do not plug in the other end of the cable yet.



9) If you will use the Ethernet connection, connect the Ethernet cable to the Ethernet Port on the DVEVM and to an Ethernet network port.

If you do not connect the board's Ethernet controller to a computer running a DHCP server, booting the board will take several additional minutes.

10) If you plan to use the UART port for a console window, connect the RS232 null modem cable to the DVEVM UART port and the a COM port on your host Linux workstation. See Section 2.2, *Connecting to a Console Window* for more about using a console window.



11) Plug in the LCD display to a power supply.

12) Plug in the NTSC/PAL video camera to a power supply.

13) Plug in the DVEVM board to a power supply.

14) Power on the LCD display.

15) Power on the DVEVM board.

16) The initial screen of the demo software should be displayed on the LCD display. Use the IR remote to run the software as described in Chapter 3.

    Note that there will be a DHCP-related delay if there is no network connection.

2.2 Connecting to a Console Window

You can open a console window that allows you to watch and interrupt DVEVM boot messages by following these steps:

1) Connect a serial cable between the serial port on the DVEVM and the serial (COM) port on a PC.

2) Run a HyperTerminal session on the PC and configure it to connect to that serial port.

3) When you power on the DVEVM, you will see boot sequence messages. You can press a key to interrupt the boot sequence and type commands in the U-Boot command shell. In this guide, commands to be typed in the U-Boot shell are indicated by an `EVM #` prompt.

Chapter 3

Running the Demonstration Software

This chapter explains how to run the software demos provided with the DVEVM.

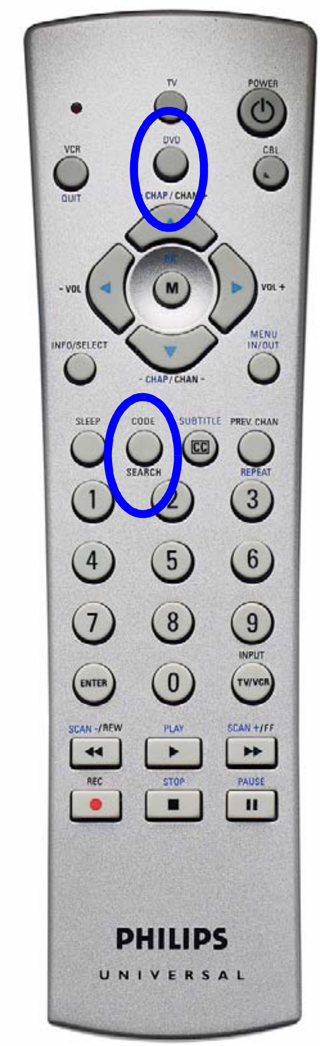
Topic Page

3.1 Default Boot Configuration

Out of the box, the DVEVM starts the demos automatically after a few seconds when you power up the board. It does not require an NFS mount or a TFTP server to run the standard demos. A DHCP server is required by default; the DHCP service provided by most routers is sufficient to run the demos. By default, the DHCP server obtains dynamic IP addresses.

To abort the standard boot, press any key in the console window (see Section 2.2). Also see Section A.5, *Alternate Boot Methods* if you want to change the boot configuration.

3.2 Starting the Standalone Demos



When you connect the DVEVM hardware, the pre-loaded examples run automatically on the LCD display. These examples encode and decode audio, video, and speech. There are two ways to use the demos:

- **Standalone.** This is the default power-on mode. The demos run automatically with no connection to a workstation in the default boot configuration.
- **Command line.** Once you have connected the DVEVM to a workstation and installed the necessary software (as described in Section 4.3.1, *Installing the Target Linux Software*), you can run the demos from the board's Linux command line as described in Section A.1, *Running the Demos from the Command Line*.

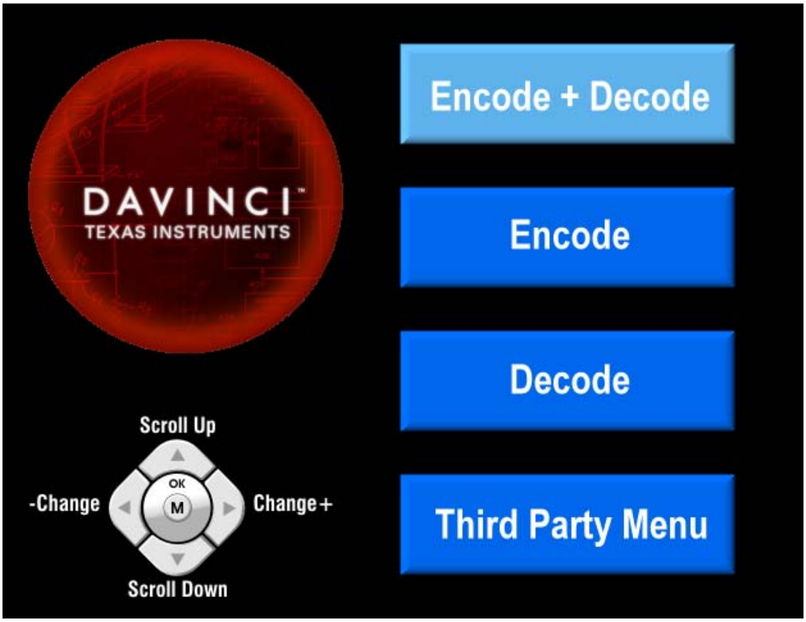
Once the DVEVM board has booted, the LCD display should show a picture of the remote control. You use the IR remote to control the demos.

To use the demos in standalone mode, follow these steps:

1) Check to make sure the batteries are installed in your IR remote.
2) The initial screen shows a diagram of the IR remote, which you use to run the standalone demos. Take a minute to look at the functions of the various buttons.
3) Since this is a universal remote, you may need to set it to use the codes necessary to run the DVEVM demos. To do this, hold down the "Code Search" button until the red light on the remote stays lit. Then press the "DVD" button and enter "020" as the code.
4) If you accidentally put the remote in TV or some other mode, press "DVD" to return the remote to the correct mode.
5) If the remote does not accept the DVD+020 code, do a full reset by removing the batteries, pressing the power key for at least a minute, then reinserting the batteries. Then program the remote as in Step 3.

3.3 Running the Standalone Demos

1) Press "Play" or "OK" on the remote to move from the remote control diagram to the main menu screen, which looks like this:

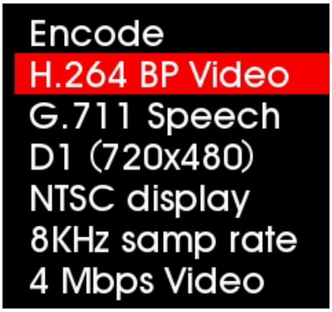


The Encode + Decode demo allows you to record and playback video. The Encode demo records audio/speech and video in the formats you select. The Decode demo plays audio/speech and video files you select. The Third-Party Menu can be used to add additional demos (see Section A.3, *Putting Demo Applications in the Third-Party Menu*).

2) Use the up and down arrows to change which demo is selected. Then, press "OK" or "Play" to switch to the selected demo. (You can quit out of the demos completely at this point by pressing "Power".)

3) Within a demo, you start at the settings screen, where you see the controls you can use to run the demo at the bottom of the screen and the current settings in the upper-right.

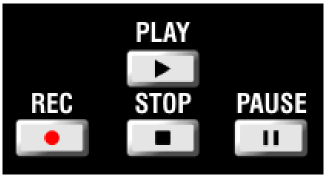
   For example, the Encode demo allows you to set the video format and the bit rate at which video should be encoded. Fixed settings are also shown here.



Encode
H.264 BP Video
G.711 Speech
D1 (720x480)
NTSC display
8KHz samp rate
4 Mbps Video

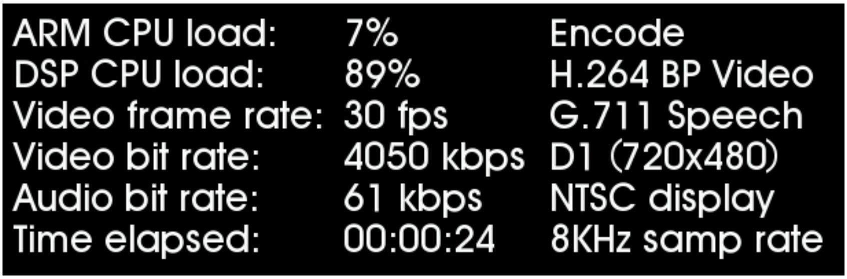
4) Use the up and down arrows to move to a setting you want to change.

5) Use the left and right arrows to cycle through the options until the setting you want is shown.

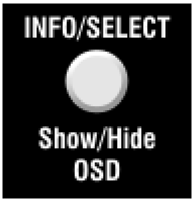


6) Press "Play" to begin the EncodeDecode and Decode demos. Press "Rec" (record) twice to begin the Encode demo. Press "Stop" to return to the main menu.

7) While the demo runs, data about the settings, processor load, and rates are shown. Static settings are on the right. Dynamic data reporting is on the left. For example:



```
ARM CPU load:       7%           Encode
DSP CPU load:       89%          H.264 BP Video
Video frame rate:   30 fps       G.711 Speech
Video bit rate:     4050 kbps    D1 (720x480)
Audio bit rate:     61 kbps      NTSC display
Time elapsed:       00:00:24     8KHz samp rate
```



8) This information overlays the video; as a result the video you see is darker than the actual video. To hide the information display so that you can better see the video, press the "Info/Select" button on the IR remote. You can change the transparency of the OSD (overlay) while running a demo by using the left and right arrows on the remote.

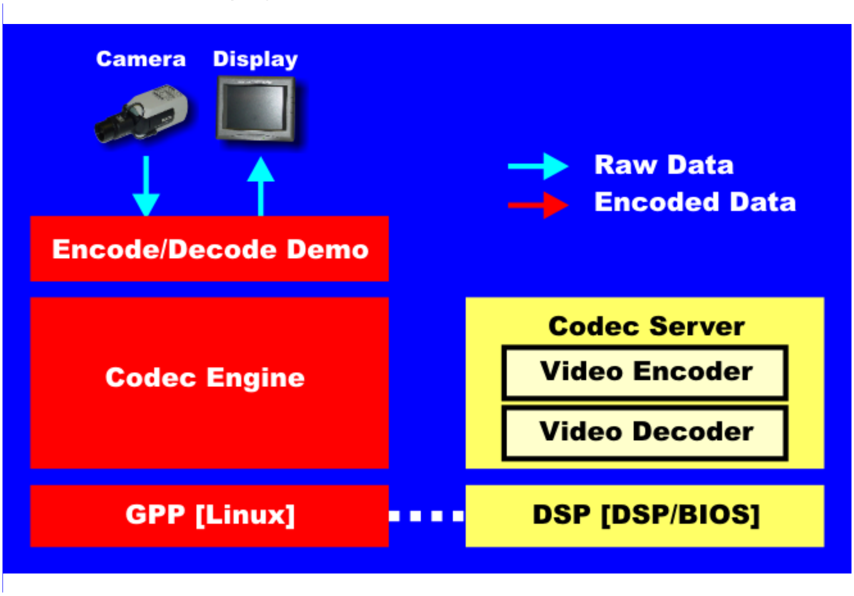
9) Press "Stop" or "Pause" when you want to end or pause a demo. Press "Stop" from the settings screen, you go back to the main menu.

The demos use the Codec Engine to allow GPP-side applications to run algorithms transparently on the DSP.

You may notice that the DSP CPU load is initially high, even if the DSP is not running any algorithms. The CPU load starts at 100% while the DSP is booting and then decreases while the DSP waits for work to be requested by the GPP. Even if DSP is idle, it may take a short amount of time (several seconds) for the CPU load to settle to zero. This is because the Codec Engine’s CPU load calculation includes a small amount of history.

3.3.1 About the Encode + Decode Demo

The Encode + Decode demo allows you to record and playback video. Video input comes from the camera, it is encoded, then decoded, and sent to the LCD display.



The Encode + Decode does only video processing; it does not encode and decode audio or speech. The supported video algorithm is H.264 Baseline Profile (.264 file extension).

Table 3–1 IR Remote Buttons for Encode + Decode Demo

IR Remote Button	Mode	Action Performed
Up/Down	--	-- no action --
Left/Right	Setup	Change resolution (ZOOM, CIF, D1)
Play or OK	Setup	Begin demo
Record	--	-- no action --
Info/Select	Setup	Show / hide block diagram for demo
Info/Select	Run	Toggle information display
Left/Right	Run	Change information transparency level
Pause	Run	Pause demo (press Play to resume)
Stop	Setup / Run	Return to previous screen

The application runs on the ARM using Linux. The video signal is passed to video encoders and decoders on the DSP by the Codec Engine. Shared memory is used when passing data.

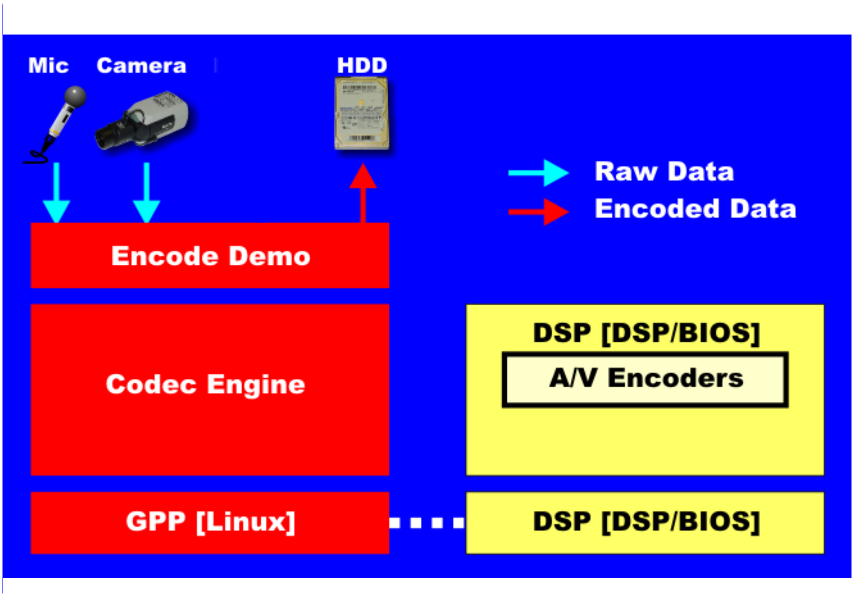
To use this demo from the command line, see Section A.1.1, *Encode/Decode Demo Command Line*.

3.3.2 About the Encode Demo

Like the Encode + Decode demo, the Encode demo also encodes video. In addition, it also encodes audio or speech. The audio/speech source is the microphone.

The encoded data is written to files on the DVEVM's hard disk drive. The possible filenames are demo.264, demo.mpeg4, demompeg4.g711, and demo264.g711. Older versions of these files are overwritten as needed.

Output is not decoded and sent to the LCD display or speakers other than to show the settings and dynamic data collected about the load and rates.



Note that you can use only a speech encoder, not an audio encoder. The supported video algorithms are MPEG4 (.mpeg4 file extension) and H.264 (.264 file extension). The supported speech algorithm is G.711 (.g711 extension).

Table 3–2 IR Remote Buttons for Encode Demo

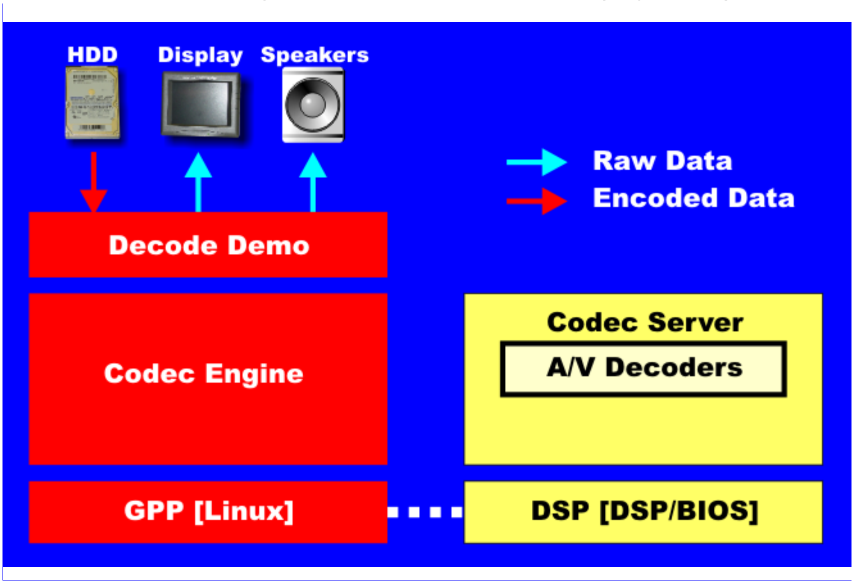
IR Remote Button	Mode	Action Performed
Up/Down	Setup	Change option selection
Left/Right	Setup	Change setting of selected option
Play	Setup	Switch to decode demo setup
Record (twice) or OK	Setup	Begin encode demo
Info/Select	Setup	Show / hide block diagram for demo
Info/Select	Run	Toggle information display
Left/Right	Run	Change information transparency level (There is no display for encode demo behind the information.)
Pause	Run	Pause demo (press Play to resume)
Stop	Setup / Run	Return to previous screen

The application runs on the ARM using Linux. The video and audio signals are passed to encoders on the DSP by the Codec Engine. Shared memory is used when passing data.

To use this demo from the command line, see Section A.1.2, *Encode Demo Command Line*.

3.3.3 About the Decode Demo

The Decode demo plays audio/speech and video files you select. You can select a source video file and a source audio or speech file. Use the left and right arrow buttons to choose from the demo files and the files created by the Encode demo, which are stored on the DVEVM's hard disk drive. The decoded signals are sent to the LCD display and speakers.



The supported video algorithms are MPEG4 (.mpeg4 file extension), H.264 (.264 file extension) and MPEG2 (.m2v file extension).

The supported audio algorithms are AAC (.aac file extension) and MPEG1 Layer 2 (.mp2 file extension). The supported speech algorithm is G.711 (.g711 file extension).

Table 3–3 IR Remote Buttons for Decode Demo

IR Remote Button	Mode	Action Performed
Up/Down	--	-- no action --
Left/Right	Setup	Select a different file combination
Play or OK	Setup	Begin decode demo
Record	--	-- no action --
Info/Select	Setup	Show / hide block diagram for demo
Info/Select	Run	Toggle information display

Table 3–3 IR Remote Buttons for Decode Demo

IR Remote Button	Mode	Action Performed
Left/Right	Run	Change information transparency level
Pause	Run	Pause demo (press Play to resume)
Stop	Setup / Run	Return to previous screen

The application runs on the ARM using Linux. The video and audio signals are passed to decoders on the DSP by the Codec Engine. Shared memory is used when passing data.

To use this demo from the command line, see Section A.1.3, *Decode Demo Command Line*.

3.3.4 About the Third Party Menu

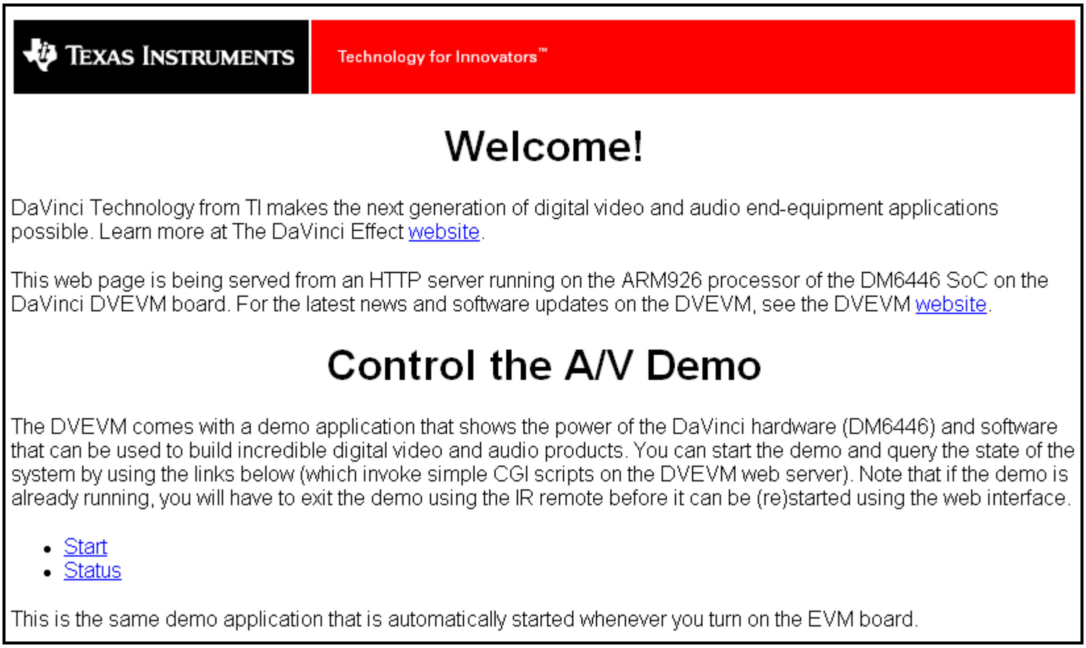
The Third-Party Menu can be used to add additional demos. See Section A.3, *Putting Demo Applications in the Third-Party Menu*.

3.4 Running the Network Demo

As an example of standard TCP/IP networking support, the DVEVM examples include a small HTTP web server. This web server is started on the GPP-side as part of the Linux startup sequence. It configured to service requests from web browsers on the standard TCP/IP port 80.

After the DVEVM board has booted, connect a PC to the same network to which the DVEVM board is connected. Enter a URL of the form "http://ip-address-of-dvevm" in a web browser (for example, Internet Explorer, Firefox, or Opera). The IP address of the board is shown in the lower-right corner of the main menu of the A/V demos.

You should see a web page with information about DaVinci technology and the DVEVM software.



Welcome!

DaVinci Technology from TI makes the next generation of digital video and audio end-equipment applications possible. Learn more at The DaVinci Effect website.

This web page is being served from an HTTP server running on the ARM926 processor of the DM6446 SoC on the DaVinci DVEVM board. For the latest news and software updates on the DVEVM, see the DVEVM website.

Control the A/V Demo

The DVEVM comes with a demo application that shows the power of the DaVinci hardware (DM6446) and software that can be used to build incredible digital video and audio products. You can start the demo and query the state of the system by using the links below (which invoke simple CGI scripts on the DVEVM web server). Note that if the demo is already running, you will have to exit the demo using the IR remote before it can be (re)started using the web interface.

- Start
- Status

This is the same demo application that is automatically started whenever you turn on the EVM board.

Use this web page to interact with the board and run the A/V demos described in Section 3.3, *Running the Standalone Demos*. Two simple CGI scripts on the DVEVM enable you to start the demos (assuming they are not already running) and see what processes are running on the board. If you want to see the demo started from the web page, be sure to exit the demo first (use the Power button from the main menu).

The web server software is an open-source package called THTTPD (http://www.acme.com/software/thttpd/). It is designed to be small, fast, and portable. The source code is included with the DVEVM software. You can get the latest version directly from the web. The web server and CGI scripts are installed on the target in the /opt/dvevm/web directory.

Chapter 4

DVEVM Software Setup

This chapter explains how to use the software provided with the DVEVM.

Topic **Page**

4.1 Software Overview

To begin developing applications, you need to install the DVEVM development environment. This section outlines the steps required to load the DVEVM software onto the development host. You will need the three DVEVM distribution CDs or the files they contain to get started.

The DaVinci software approach provides interoperable, optimized, production-ready video and audio codecs that leverage DSP and integrated accelerators. These codecs are built into configurable frameworks, and are presented via published APIs within popular operating systems (such as Linux) for rapid software implementation.

The DVEVM provides for development on the ARM side. Developers treat the DSP-side as a black box that is accessible through a set of APIs. If you want to program the DSP-side directly, the DVSDK (Digital Video Software Development Kit) supports this capability.

The following software is provided with the DVEVM:

- ❏ **Standalone demonstration software.** This is provided on the hard drive on the DVEVM. The hard-wired examples encode and decode audio, video, and speech. Another demo shows the board’s network capabilities. See Section 3.2, *Starting the Standalone Demos*.
- ❏ **CD 1: Getting Started Guide**. This CD includes demo applications, Codec Engine software, example codec servers, and DVEVM documentation. Contains the following files:
  - ■ sprue66a.pdf (this manual)
  - ■ dvevm_setuplinux_#_##_##_##.bin
  - ■ dvevm_setupwin32_#_##_##_##.exe (Windows installer)
  - ■ mvl_lsp_setuplinux_#_##_##_##.bin
  - ■ restore directory (Contains files used for hard drive recovery. Contact TI Technical Worldwide Support if you need details.)
- ❏ **CD 2: MontaVista Linux Pro v4.0 Target File System**. The DVEVM provides a preliminary demonstration version. Contains the file:
  - ■ mvl_target_setuplinux_#_##_##_##.bin. This installation file contains the MontaVista target file system.
- ❏ **CD 3: MontaVista Linux Pro v4.0 System Tools**. The version provided with the DVEVM is the preliminary demonstration version. Contains the following file:
  - ■ mvl_setuplinux_#_##_##_##.bin. This installation file contains the MontaVista Tool development tool chain.
- ❏ **CD 4: A/V Data**. Contains sample A/V data in data.tar.gz.

Texas Instruments, in agreement with MontaVista Software Inc., is providing a demonstration version of the Linux Professional Edition v4.0 embedded operating system and development tools. The base DVEVM kit includes a preliminary release of this demonstration version. The demo version is a subset of what MontaVista provides with the full Professional Edition. Tools such as DevRocket^{TM} and the Professional Edition documentation are not included, but it is otherwise fully functional and useful for customers evaluating the DaVinci platform. Also, please note that this release does not include a MontaVista user license, and no direct customer support, warranty, or indemnification from MontaVista Software Inc. is provided.

You may choose to order the DVSDK, which includes the production release of this demonstration version of MontaVista Linux. This includes a full MontaVista license and the DevRocket IDE.

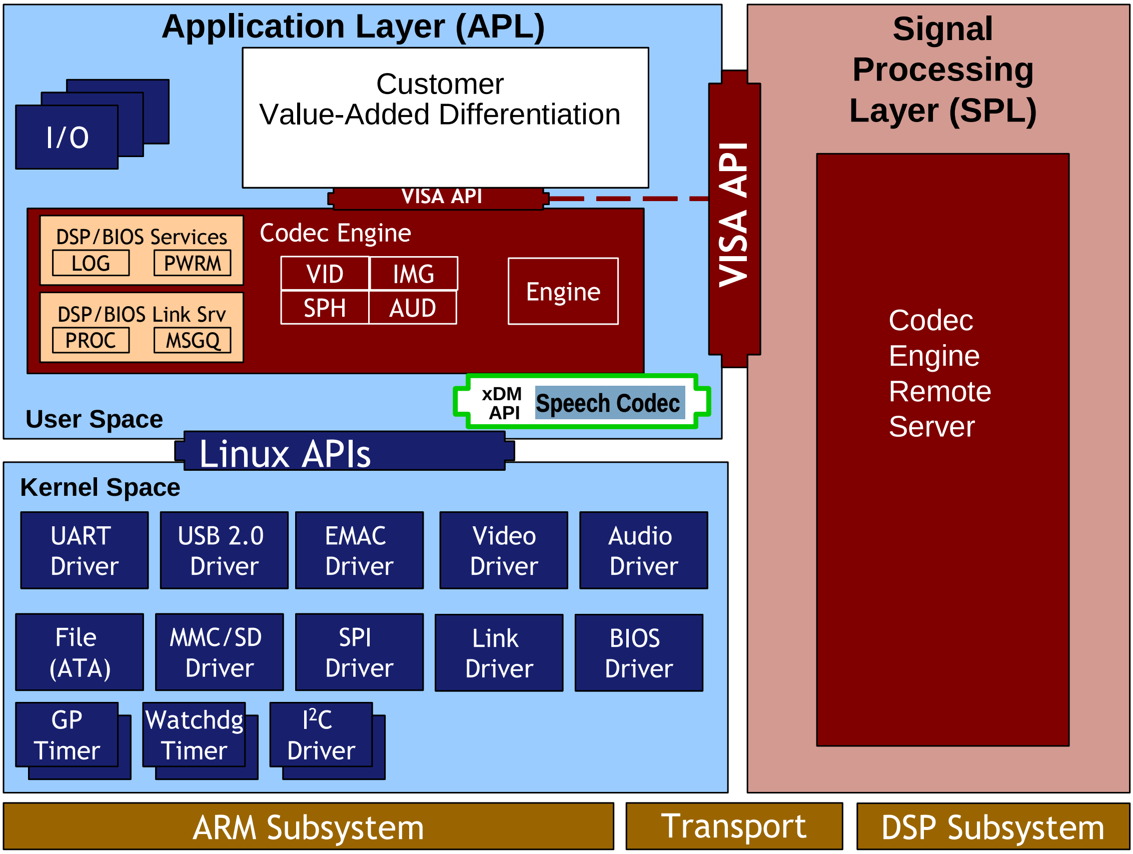
4.1.1 Command Prompts in This Guide

In this guide, commands are preceded by prompts that indicate the environment where the command is to be typed. For example:

- ❑ **`host $`**
  Indicates command to be typed into the shell window of the host Linux workstation.
- ❑ **`EVM #`**
  Indicates commands to be typed into the U-Boot shell in a console window connected to the EVM board's serial port. (Section 2.2)
- ❑ **`target $`**
  Indicates commands to be typed into the Linux shell in the terminal window connected to the EVM board's serial port.

4.1.2 Software Components

The following figure shows the software components used for application development on the DVEVM:



In the previous figure, your application runs on the ARM side. It handles I/O and application processing. To process video, image, speech, and audio signals, it uses the VISA APIs provided by the Codec Engine. The Codec Engine, in turn, uses services such as DSP/BIOS Link and protocols such as xDAIS and xDM to communicate with a pre-configured Codec Engine Remote Server on the DSP side. The DSP handles signal processing and the results are available to the ARM side in shared memory. For more information, see the *Codec Engine Application Developer's Guide* (SPRUE67).

In addition, Linux on the ARM side makes a large number of APIs available to your application. These include drivers and timers.

4.2 Preparing to Install

On a host system, mount the three DVEVM demonstration CDs and copy the following .bin files to a temporary location with at least 1.2 GB available space. Since you can delete the installation files after installing the software, a directory like /tmp is recommended.

❏ mvl_setuplinux_#_##_##_##.bin

❏ mvl_target_setuplinux_#_##_##_##.bin

❏ mvl_lsp_setuplinux_#_##_##_##.bin

❏ dvevm_setuplinux_#_##_##_##.bin or
dvevm_setupwin32_#_##_##_##.exe (See Section 4.3.2)

Ensure that an X graphical display is available, and point your DISPLAY environment variable to this value. For example:

csh:

```
host $ setenv DISPLAY cnabc0314159d1:0
```

ksh:

```
host $ export DISPLAY=cnabc0314159d1:0
```

4.3 Installing the Software

Installing the software used by the DVEVM involves performing the following steps:

❏ Section 4.3.1, *Installing the Target Linux Software*

❏ Section 4.3.2, *Installing the DVEVM Software*

❏ Section 4.3.3, *Installing the A/V Demo Files*

❏ Section 4.3.4, *Exporting a Shared File System for Target Access*

❏ Section 4.3.5, *Testing the Shared File System*

❏ Section 4.3.6, *Configuring the Boot Setup for PAL Video Users*

4.3.1 Installing the Target Linux Software

This section explains how to install Linux for use on the target board. This is a demonstration version of MontaVista Linux Pro v4.0.

Note that separate versions of Linux are used by the target and your host Linux workstation. The following Linux host operating systems are supported for use with the DVEVM.

❏ Red Hat Enterprise Linux v3

❏ Red Hat Enterprise Linux v4

❏ SuSe v10.0 Workstation

To install the Linux software, follow these steps:

1) Log in as **root** on your host Linux workstation. This will allow you to successfully run the graphical installer to install MontaVista Linux.

2) Execute each of the following bin files (where **#_##_##_##** is the current version number) from the temporary location that they were copied to in order to install the Linux tools, Linux kernel, and the file system. These installation instructions assume you use the default installation directory, /opt.

```
host $ ./mvl_setuplinux_#_##_##_##.bin
host $ ./mvl_target_setuplinux_#_##_##_##.bin
host $ ./mvl_lsp_setuplinux_#_##_##_##.bin
```

It may take up to several minutes per file to start up InstallShield from these files.

3) After you execute these files, make sure the following files are located in /opt/mv_pro_4.0 (or in the /mv_pro_4.0 subdirectory of the directory you chose in place of the default):

   - mvltools4.0-no-target.tar.gz
   - mvl4.0-target_path.tar.gz
   - DaVinciLSP#.#.#.tar.gz

4) Go to the location where you will unpack the tar files. For example:

```
host $ cd /opt/mv_pro_4.0
```

5) Unpack the tar files (as root) by using the following commands:

```
host $ tar zxf mvltools4.0-no-target.tar.gz
host $ tar zxf mvl4.0-target_path.tar.gz
host $ tar zxf DaVinciLSP#.#.#.tar.gz
```

   This creates the MontaVista directory structure under the /opt/mv_pro_4.0/montavista/ directory.

4.3.2 Installing the DVEVM Software

The DVEVM software includes Codec Engine components, DSP/BIOS Link, sample data files, xDAIS and xDM header files, and a contiguous memory allocator for Linux (CMEM).

❑ Codec Engine provides a framework for creating and interacting with A/V codecs running on the DSP via a reflection of their xDM interfaces on the ARM through a Linux C-callable API.

❑ DSP/BIOS Link provides a GPP/DSP interface utilized by the Codec Engine to control and communicate with the DSP from Linux.

To install the DVEVM software using the Linux installer, follow these steps:

1) Log in using a **user account**. In the following steps, we refer to the home user directory as "~".

2) Install the software from the DVEVM CD. For example:

```
host $ cd /tmp
host $ ./dvevm_setuplinux_1_00_00_bb.bin
```

3) When you are prompted, *do not* use the default installation location. Instead, install the software in the home directory for the account you are using.

   For example, if your home directory is /home/useracct, enter that in the installation location dialog. The DVEVM software would then be installed under /home/useracct/dvevm_#_##, where #_## is the version number.

4) You can now delete the .bin files that you loaded into the temporary directory.

> **Note:** You can uninstall one of these components by using the `rm -rf` command on its directory. You should ignore the _uninstall directories created by InstallShield.

Some Microsoft Windows installers are provided for customers who already have the Windows version of MontaVista Linux Tools. Use the installer that corresponds to your version of MontaVista Linux. See Section A.6, *Installing Components Under Windows* for more information.

4.3.3 Installing the A/V Demo Files

The fourth CD contains the A/V files used by the demos. After following the instructions in the previous section, follow these instructions to install the A/V files:

1) Go to the DVEVM directory that you set up previously. For example:

```
host $ cd ~/dvevm_1_xx
```

2) Mount the A/V data CD and copy the file to your DVEVM directory. For example:

```
host $ cp /mnt/cdrom/data.tar.gz .
```

3) Extract the A/V data files. For example:

```
host $ tar xfz data.tar.gz
```

4.3.4 Exporting a Shared File System for Target Access

Although the board’s hard drive contains a file system, during development it is more convenient to have the target board NFS mount a file system on a host Linux workstation. Once you have tested the application, you can store it on the board’s hard drive for a standalone demonstration.

Before the board can mount a target file system, you must export that target file system on the host Linux workstation. The file system uses an NFS (Network File System) server. The exported file system will contain the target file system and your executables.

To export the file system from your NFS server, perform the following steps. You only need to perform these steps once.

1) Log in with a **user** account on the host Linux workstation. (In the following steps, we refer to the home user directory as "~".)

2) Perform the following commands to prepare a location for the MontaVista file system:

```
host $ cd ~
host $ mkdir -p workdir/filesys
host $ cd workdir/filesys
```

3) Switch user to "**root**" on the host Linux workstation.

```
host $ su root
```

4) Perform the following commands to create a copy of the target file system with permissions set for writing to the shared area as <*useracct*>. Substitute your user name for <*useracct*>. If you installed in a location other than /opt/mv_pro_4.0, use your location in the cp command.

```
host $ cp -a /opt/mv_pro_4.0/montavista/pro/devkit/arm/v5t_le/target/* .
 host $ chown -R <useracct> opt
```

5) Edit the /etc/exports file on the host Linux workstation. Add the following line for exporting the filesys area, substituting your user name for <*useracct*>. Use the full path from root; ~ may not work for exports on all file systems.

```
/home/<useracct>/workdir/filesys *(rw,no_root_squash,no_all_squash,sync)
```

6) Still as root, use the following commands to make the NFS server aware of the change to its configuration and to invoke an NFS restart.

```
host $ /usr/sbin/exportfs -a
host $ /sbin/service nfs restart
```

4.3.5 Testing the Shared File System

To test your NFS setup, follow these steps:

1) Get the IP address of your host Linux workstations as follows. Look for the IP address associated with the eth0 Ethernet port.

```
host $ /sbin/ifconfig
```

2) Open a terminal emulation window to connect to the DVEVM board via RS-232. If you have a Windows workstation, you can use HyperTerminal. If you have a Linux workstation, you might use Minicom.

3) Power on the DVEVM board, and abort the automatic boot sequence by pressing a key in the console window (Section 2.2).

4) Set the following environment variables in the console window:

```
EVM # setenv nfshost <IP ADDRESS OF YOUR NFS HOST>
EVM # setenv rootpath <DIRECTORY_TO_MOUNT>
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
     ip=dhcp root=/dev/nfs
     nfsroot=$(nfshost):$(rootpath),nolock mem=120M
```

5) Save the environment so that you don't have to retype these commands every time you cycle power on the EVM board:

```
EVM # saveenv
```

6) Boot the board using NFS:

```
EVM # boot
```

See Section A.5, *Alternate Boot Methods* for information about booting with TFTP or NFS and using flash or the board's hard drive.

4.3.6 Configuring the Boot Setup for PAL Video Users

You can configure the DVEVM to select either the NTSC or PAL video standard during the default boot sequence. To select PAL, set switch 10 on the S3 (USER) user bank of switches to On. For NTSC, set this switch to Off. The switch causes the U-Boot environment variable "videostd" to be set to "pal" or "ntsc".

Using the "videostd" variable in the "bootargs" environment variable passed to the Linux kernel causes the corresponding video standard to be used by the display (VPBE) driver. The default "bootcmd" environment variable accomplishes this task as follows:

```
bootargs="mem=120M console=ttyS0,115200n8 root=/dev/hda1 rw
noinitrd ip=dhcp"

bootcmd="setenv setboot setenv bootargs \$(bootargs)
video=dm64xxfb:output=\$(videostd);run setboot;bootm
0x2050000"
```

When the "boot" command is run in U-Boot, the value of "videostd" is substituted based on the setting of the switch.

The bootargs examples in the rest of this manual assume the default NTSC video output is being used. If you are using PAL video, you will need to modify the examples accordingly.

See Section A.2, *Changing the Video Input/Output Methods* for information about switching to S-Video and Component video.

4.4 Setting Up the Build/Development Environment

To set up the GPP-side development and build environment, follow these steps:

1) Log in to your **user** account (and not as root) on the NFS host system.

2) Set your PATH so that the MontaVista tool chain host tools and cross compiler (arm_v5t_le-gcc) can be found. For example, in a default installation of the MontaVista LSP, you should add a definition like the following to your shell resource file (for example, ~/.bashrc). If you installed in a location other than /opt/mv_pro_4.0, use your location in the PATH.

```
PATH="/opt/mv_pro_4.0/montavista/pro/devkit/arm/v5t_le/bin:
   /opt/mv_pro_4.0/montavista/pro/bin:
    /opt/mv_pro_4.0/montavista/common/bin:$PATH"
```

4.4.1 Writing a Simple Program and Running it on the DVEVM

Make sure you have performed the steps in Section 4.3.4, *Exporting a Shared File System for Target Access* and Section 4.4, *Setting Up the Build/Development Environment*.

Perform the following steps on the NFS host system as user (not as root):

1) `host $ mkdir ~/workdir/filesys/opt/hello`

2) `host $ cd ~/workdir/filesys/opt/hello`

3) Create a file called hello.c with the following contents:

```
#include <stdio.h>

int main() {
   printf("Buongiorno DaVinci!\n");
   return 0;
}
```

4) `host $ arm_v5t_le-gcc hello.c -o hello`

Perform the following steps on the target board. You may use either the target's console window (Section 2.2) or a telnet session.

1) `target $ cd /opt/hello`

2) Run ./hello. The output should be:

```
Buongiorno DaVinci!
```

4.5 Rebuilding the DVEVM Software for the Target

To place demo files in the /opt/dvevm directory, you need to rebuild the DVEVM software. To do this, follow these steps:

1) Change directory to ~/dvevm_#_##.
2) Edit the ~/dvevm_#_##/Rules.make file. Make sure that EXEC_DIR points to the opt directory on the NFS exported file system as follows and that LINUXKERNEL_INSTALL_DIR is defined as follows:

```
EXEC_DIR=/home/<useracct>/workdir/filesys/opt/dvevm
LINUXKERNEL_INSTALL_DIR=/opt/mv_pro_4.0/montavista/pro/devkit/lsp/ti-davinci
```

3) While in the same directory that contains Rules.make, use the following commands to build the DVEVM product and put the resulting binaries on the target file system (for example, /opt/dvevm).

```
host $ make
host $ make install
```

4.6 Building a New Linux Kernel

If you modify the target’s Linux kernel sources, you will need to rebuild it and then boot it up by either replacing the kernel that comes installed on the DVEVM board’s flash or by having the U-Boot utility use TFTP to boot the kernel over a network connection.

Make sure you have completed Section 4.4, *Setting Up the Build/Development Environment* and Section 4.4.1, *Writing a Simple Program and Running it on the DVEVM* before attempting to build a new kernel. You must at a minimum have the MontaVista tools in your path:

```
PATH="/opt/mv_pro_4.0/montavista/pro/devkit/arm/v5t_le/bin:
/opt/mv_pro_4.0/montavista/pro/bin:$PATH"
```

To rebuild the Linux Kernel, follow these steps:

1) Log in to your user account (not as root).
2) Use commands like the following to make a local working copy of the MontaVista Linux Support Package (LSP) in your home directory. This copy contains the embedded Linux 2.6.10 kernel plus the DaVinci drivers. If you installed in a location other than /opt/mv_pro_4.0, use your location in the cp command.

```
host $ cd ~
host $ mkdir -p workdir/lsp
host $ cd workdir/lsp
host $ cp -R /opt/mv_pro_4.0/montavista/pro/devkit/lsp/ti-davinci .
```

3) Use the following commands to configure the kernel using the DaVinci defaults. Note that CROSS_COMPILE specifies a prefix for the executables that is used during compilation:

```
  host $ cd ti-davinci
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- davinci_dm644x_defconfig
```

4) To modify the kernel options, you will need to use the 'make menuconfig' command. See the MontaVista documentation for information on how to do this. To enable the MontaVista default kernel options, use the following command:

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

5) Compile the kernel using the following command:

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

6) Use the following command to copy the uImage to a place where U-Boot can use TFTP to download it to the EVM. These commands assume you are using the default TFTP boot area, which is /tftpboot. If you use another TFTP root location, please change /tftpboot to your own TFTP root location:

```
host $ cp ~/workdir/lsp/ti_davinci/arch/arm/boot/uImage /tftpboot
host $ chmod a+r /tftpboot/uImage
```

See a standard Linux kernel reference book or online source for more about Linux build configuration options.

4.7 Booting the New Linux Kernel

After building the new kernel, in order to use it to boot the DaVinci board, you must transfer it to the board via TFTP. It is assumed you have completed the steps in Section 4.6, *Building a New Linux Kernel* and the boot file, uImage has been copied to /tftpboot (or some other site-specific TFTP accessible location).

1) Power on the DVEVM board, and abort the automatic boot sequence by pressing a key in the console window (Section 2.2).

2) Set the following environment variables. (This assumes you are starting from a default, clean U-Boot environment. See Section 3.1, *Default Boot Configuration* for information on the U-Boot default environment.)

```
EVM # setenv bootcmd 'dhcp;bootm'
EVM # setenv serverip <YOUR TFTP SERVER IP ADDRESS>
EVM # setenv bootfile uImage
EVM # setenv bootargs mem=120M console=ttyS0,115200n8
  root=/dev/hda1 rw noinitrd ip=dhcp
```

This configuration boots a new Linux kernel via TFTP with a hard drive based file system. Please see Section A.5.4, *Booting via TFTP Using NFS File System* for information on using TFTP with an NFS file system.

4.8 Installing Upgrades

To install software from an upgrade package, following the instructions provided with that package.

Appendix A

Additional Procedures

This appendix describes optional procedures you may use depending on your setup and specific needs.

A.1 Running the Demos from the Command Line

You can run the demo applications from the Linux shell in a terminal window connected to the EVM board's serial port. These are the same demos described in Section 3.2, *Starting the Standalone Demos*.

The command line syntax and options for the demo applications are provided in the following subsections.

Before running demo applications from the command line, you must load the DSP/BIOS Link and CMEM kernel modules. Use the following command from the directory that contains the demos to load these modules:

```
Target $ ./loadmodules.sh
```

The supported file extensions for A/V files are:

- ❏ Video. H.264 Baseline Profile (*.264 files)
- ❏ Video. MPEG4 (*.mpeg4 files)
- ❏ Video. MPEG2 (*.m2v files)
- ❏ Speech. G.711 (*.g711 files)
- ❏ Audio. AAC (*.aac files)
- ❏ Audio. MPEG1 Layer 2 (*.mp2 files)

A.1.1 Encode/Decode Demo Command Line

This demo uses Codec Engine to encode data from the capture device (V4L2) using the H.264 algorithm into an intermediate buffer before the data is decoded to the display frame buffer.

```
Target $ ./encodedecode [options]
```

Options:

- ❏ -r | --resolution
  Resolution of demo. Specify CIF, ZOOM, or D1. The default is D1.
  - ■ CIF. Captures and displays at 352x240 on an NTSC system and 352x288 on a PAL system. The image is centered on the screen.
  - ■ ZOOM. Captures video at 352x240 on an NTSC system and 352x288 on a PAL system. After the data has been encoded and decoded, it is zoomed by the display to D1 resolution (720x480 on NTSC and 720x576 on PAL).
  - ■ D1. Capture and displays at 720x480 on an NTSC system and 720x576 on a PAL system.

❏ -t | --time
Number of seconds to run the demo. By default, there is no time limit.

❏ -i | --interface
If used, causes the main demo interface to launch when this demo exits. By default, this is off.

❏ -x
Select S-Video input format. This flag is available only with DVEVM 1.1 greater. (DVEVM 1.0 supports only composite video.)

❏ -h | --help
Print this help message.

The following example uses ZOOM resolution and quits after 20 seconds.

```
Target $ ./encodedecode -r ZOOM -t 20
```

For more information about this demo, see the encodedecode.txt file and Section 3.3.1, *About the Encode + Decode Demo*.

A.1.2 Encode Demo Command Line

This demo encodes data from peripheral device drivers to files. Video and speech files are supported. The files created are raw frames of encoded data with no headers.

```
Target $ ./encode [options]
```

Options:

❏ -s | --speechfile
Specify the filename to which speech output should be sent. The file extension identifies the format to use. The supported speech algorithm is G.711 (.g711 extension). The file will be created if it does not exist, and truncated if it does exist.

❏ -v | --videofile
Specify the filename to which video output should be sent. The file extension identifies the format to use. The supported video algorithms are MPEG4 (.mpeg4 file extension) and H.264 (.264 file extension). The file will be created if it does not exist, and truncated if it does exist.

❏ -b | --bitrate
Specify the bit rate at which video should be encoded. The default is 4000000 bps.

❏ -t | --time
Number of seconds to run the demo. By default, there is no time limit.

- -l, --linein
  Changes the input device for sound recording to the "Line In" as opposed to the "Mic In", which is the default.
- -i | --interface
  If used, causes the main demo interface to launch when this demo exits. By default, this is off.
- -x
  Select S-Video input format. This flag is available only with DVEVM 1.1 or greater. (DVEVM 1.0 supports only composite video.)
- -h | --help
  Print this help message.

The following example uses MPEG4 video encode, no audio encode, and quits after 20 seconds.

```
Target $ ./encode -v test.mpeg4 -t 20
```

The following example uses H.264 video encode at 1 Mbps and no audio encode:

```
Target $ ./encode -v test.264 -b 1048576
```

The following example uses H.264 video encode and G.711 speech encode:

```
Target $ ./encode -v test.264 -s test.g711
```

You must supply at least a video or a speech file or both with appropriate extensions for the file formats. For more information about this demo, see the encode.txt file and Section 3.3.2, *About the Encode Demo*.

A.1.3 Decode Demo Command Line

This demo uses Codec Engine to decode data from files. It outputs the uncompressed data using peripheral device drivers. Video, audio, and speech files are supported. All files must consist of raw frames of data.

```
Target $ ./decode [options]
```

Options:

- -a | --audiofile
  Specify the filename of the audio file to play. The file extension identifies the format to use. The supported audio algorithms are AAC (.aac file extension) and MPEG1 Layer 2 (.mp2 file extension). You cannot play both an audio file and a speech file at the same time.

- -s | --speechfile
  Specify the filename of the speech file to play. The file extension identifies the format to use. The supported speech algorithm is G.711 (.g711 file extension). You cannot play both an audio file and a speech file at the same time.
- -v | --videofile
  Specify the filename of the video to play. The file extension identifies the format to use. The supported video algorithms are MPEG4 (.mpeg4 file extension), H.264 (.264 file extension) and MPEG2 (.m2v file extension).
- -t | --time
  Number of seconds to run the demo. By default, there is no time limit.
- -l | --loop
  If used, causes a loop back to the beginning of the files when they are finished.
- -i | --interface
  If used, causes the main demo interface to launch when this demo exits. By default, this is off.
- -h | --help
  Print this help message.

The following example uses MPEG2 video decode, no speech or audio decode, and quits after 20 seconds.

```
Target $ ./decode -v data/videos/mpeg2ts_part1.m2v -t 20
```

The following example uses AAC audio decode only:

```
Target $ ./decode -a data/sounds/l9_44.aac
```

The following example uses MPEG2 video and AAC audio decode:

```
Target $ ./decode -a data/sounds/l9_44.aac
   -v data/videos/mpeg2ts_part1.m2v
```

The following example uses MPEG4 video and G.711 speech decode:

```
Target $ ./decode -v data/sounds/dlp.mpeg4
   -s data/sounds/Input1_Alaw.g711
```

You must specify at least a video or speech or audio file in order to run the decode demo. You may specify both video and speech or both video and audio. You cannot specify all three at once.

For more information about this demo, see the decode.txt file and Section 3.3.3, *About the Decode Demo*.

A.2 Changing the Video Input/Output Methods

The DVEVM can input video using the following methods:

❑ Composite [default]

❑ S-Video (best quality)

In addition, there are three types of video output:

❑ Composite [default] (lowest quality)

❑ S-Video (medium quality)

❑ Component (best quality)

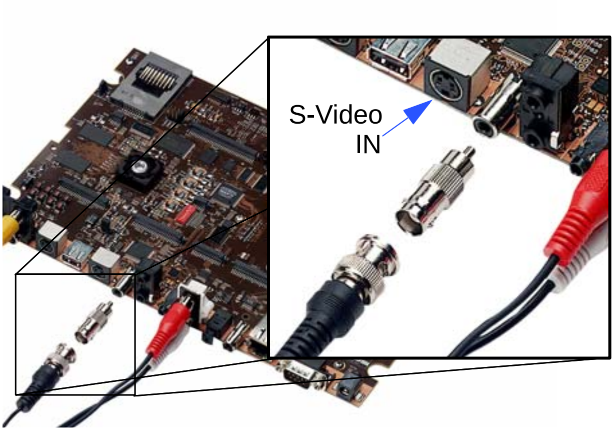
There is a significant quality difference between the different inputs and outputs. However, the cables in the DVEVM kit support only composite video. You will need to get S-Video or Component video cables from another source.

A.2.1 Using S-Video Input

To switch to higher-quality S-Video input, follow these steps:

1) Connect your S-Video connector to the S-Video input port, which is directly to the left of the currently-used composite video input port.

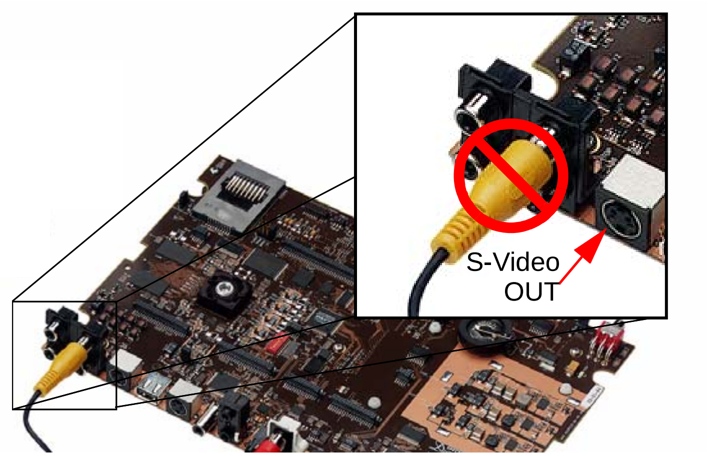
2) Select S-Video input on the command line when you execute the encode or encodedecode demo using the '-x' flag. This flag is available only with DVEVM 1.1 or greater. (DVEVM 1.0 supports only composite video.)



A.2.2 Using S-Video Output

To switch to higher-quality S-Video output, follow these steps:

1) Unplug the composite video connector. Then, connect your S-Video connector to the S-Video output port, which is to the right of the currently-used composite video output port.



The DVEVM kit does not include an S-Video cable. In addition, you will need a video display with an S-Video input.

2) On the kernel command line, you can configure the DVEVM to select both NTSC vs. PAL and the S-Video output format (see Section 4.3.6, *Configuring the Boot Setup for PAL Video Users*). For example, if you want both NTSC and S-Video output, use the following:

```
video=dm64xxfb:output=ntsc:format=s-video
```

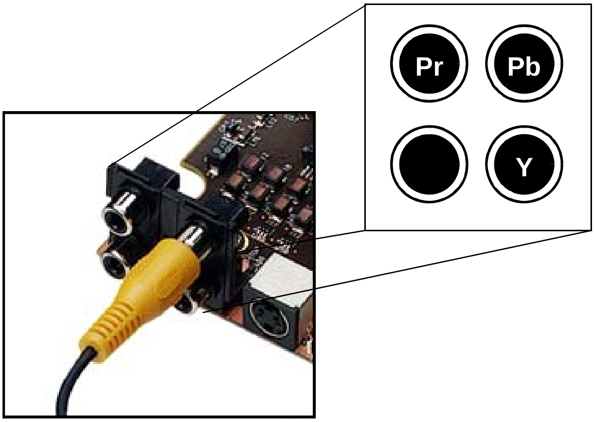
If you want both PAL and S-Video, use the following:

```
video=dm64xxfb:output=pal:format=s-video
```

A.2.3 Using Component Video Output

To switch to highest-quality component video output, follow these steps:

1) Connect your component video connectors to the connectors in a square on the far left of the board. Instead of connecting one connector as with composite video, connect the YPrPb connectors as shown here.



The DVEVM kit does not include a 3-connector cable used for component (YPrPb) video. In addition, you will need a video display with component video inputs.

2) On the kernel command line, you can configure the DVEVM to select both NTSC vs. PAL and the component video output format (see Section 4.3.6, *Configuring the Boot Setup for PAL Video Users*). For example, if you want both NTSC and component video output, use the following:

```
video=dm64xxfb:output=ntsc:format=component
```

If you want both PAL and component video, use the following:

```
video=dm64xxfb:output=pal:format=component
```

A.3 Putting Demo Applications in the Third-Party Menu

You can add your own demos to the Third-Party Menu by following the steps in this section. Only four demos can be shown at once in the user-interface. If you add more than four demos, the first four in alphabetical order are shown.

1) Create the following files for your demo:

   - **logo.jpg.** This is the logo of the third party company which will be showed next to the demo description. The picture needs to be in JPEG format and of size 50x50.
   - **readme.txt.** This is a text file. The first 40 characters of the file should briefly describe the demo. The demo interface displays up to 40 characters, but stops if it encounters a new line character. For example, the file might contain "Video Phone demo" or "Network Audio demo".
   - **app.sh.** This is an executable that launches your demo. It can either be the demo executable itself or a shell script that executes the executable. (If this is a shell script, make sure its executable bit is set for all). A script could look something like:

     ```
     #!/bin/sh
     exec ./mydemoname
     ```

   - **other files.** If app.sh is a shell script, your demo executable will have some other name. You may also need to include data files or other files used by the executable.

> **Note:** The demo application must use relative paths to access any files it needs at runtime. This because the archive is extracted to another location from which the demo is executed.

2) Create a gzipped tar file (ends with .tar.gz) that archives all the files in the previous list. For example, if your files are logo.jpg, readme.txt, and app.sh, you could use the following command:

   ```
   tar cvzf ti_videophone.tar.gz logo.jpg readme.txt app.sh
   ```

   Name the tar file using <company>_<demoname>.tar.gz (with no spaces in the file name) as the convention. For example, a video phone demo created by Texas Instruments would be named ti_videophone.tar.gz. The name must be unique since all demos are installed in the same directory.

   The three required files must be in the top-level directory of the archive. Other files may be in subdirectories, so long as the demo uses relative references to access them. For example, the following directory structure might be used in the archive:

```
|-- app.sh
|-- data
|   |-- datafile1
|   `-- datafile2
|-- logo.jpg
`-- readme.txt
```

To check the format of the file you create, execute the following command in Linux. The result should say "gzip compressed data".

```
<filename>.tar.gz
```

3) Put your archive in the "thirdpartydemos" subdirectory of the target installation directory. This is where the DVEVM software was installed on the target file system. The default target installation directory is /opt/dvevm, so the default location for demo archives is /opt/dvevm/thirdpartydemos. Do not extract the contents of the archive in this location. Extraction is performed behind-the-scenes each time the demo is run.

A.4 Setting Up a TFTP Server

You can check to see if a TFTP server is set up with the following command:

```
host $ rpm -q tftp-server
```

If it is not set up, you can follow these steps:

1) If you have not yet installed MontaVista Linux Demo Edition (see Section 4.3.1), you can download a TFTP server for your Linux host from many locations on the Internet. Search for "tftp-server".

2) To install TFTP, use this command, where `-#.##-#` is the version number portion of the filename:

   ```
   host $ rpm -ivh /db/ztree/useracct/tftp-server-#.##-#.rpm
   ```

   You should see the following output:

   ```
    warning: /db/ztree/useracct/tftp-server-#.##-#.rpm:
    V3 DSA signature: NOKEY, key ID 4f2a6fd2
   Preparing... #################################### [100%]
   1:tftp-server ################################## [100%]
   ```

3) Confirm that TFTP is installed with this command:

   ```
   host $ /sbin/chkconfig --list
   ```

   You should see the following output. Note that TFTP is at the end of the list.

   ```
   xinetd based services:
       krb5-telnet:    off
       rsync:          off
       eklogin:        off
       gssftp:         off
       klogin:         off
       ...
       sgi_fam:        on
       rexec:          off
       rlogin:         off
       rsh:            on
       telnet:         on
       tftp:           off
   ```

If you need to turn off the TFTP server, use this command:

```
/sbin/chkconfig tftp on
```

The default root location for servicing TFTP files is /tftpboot.

A.5 Alternate Boot Methods

The default configuration for the DVEVM is to boot from flash with the file system on the board's hard drive. The following are alternate ways you may want to boot the board:

❑ TFTP boot with hard drive file system (Section A.5.2)

❑ Flash boot with NFS file system (Section A.5.3)

❑ TFTP boot with NFS file system (Section A.5.4)

Section 4.3.6 discusses booting in PAL video mode vs. NTSC video mode.

The subsections that follow show the environment variable settings used to enable each boot method.

To boot in one of these modes, follow these steps:

1) Power on the DVEVM board, and abort the automatic boot sequence by pressing a key in the console window (Section 2.2).
2) Set the environment variables indicated in the following subsections for the boot mode you want to use.
3) If you want to use these settings as the default in the future, save the environment:

```
EVM # saveenv
```

4) Boot the board using the settings you have made:

```
EVM # boot
```

A.5.1 Booting from Flash Using Board's Hard Drive File System

This is the default, out-of-the-box boot configuration.

To boot in this mode, set the following parameters after you abort the automatic boot sequence:

```
EVM # setenv bootcmd bootm 0x2050000
EVM # setenv bootargs mem=120M console=ttyS0,115200n8
    root=/dev/hda1 rw noinitrd ip=dhcp
```

For example, the environment variables set for this mode might be:

```
bootargs=mem=120M console=ttyS0,115200n8 root=/dev/hda1 rw
noinitrd ip=dhcp
bootdelay=3
baudrate=115200
bootfile=library/davinci/0.4.2/uImage (NOT USED)
serverip=192.168.160.71 (NOT USED)
bootcmd=bootm 0x2050000
stdin=serial
stdout=serial
stderr=serial
ethaddr=00:0e:99:02:51:46
```

When you boot, look for the following line that confirms the boot mode:

```
## Booting image at 02050000 ...
```

A.5.2 Booting via TFTP Using Board’s Hard Drive File System

To boot in this mode, set the following parameters after you abort the automatic boot sequence:

```
EVM # setenv bootcmd 'dhcp;bootm'
EVM # setenv bootargs mem=120M console=ttyS0,115200n8
   root=/dev/hda1 rw noinitrd ip=dhcp
EVM # setenv serverip <your_tftp_server_ip_address>
EVM # setenv bootfile <path_on_tftpserver>/uImage
```

For example, the environment variables set for this mode might be:

```
bootargs=mem=120M console=ttyS0,115200n8 root=/dev/hda1 rw
noinitrd ip=dhcp
bootdelay=3
baudrate=115200
stdin=serial
stdout=serial
stderr=serial
ethaddr=00:0e:99:02:51:46
serverip=xxx.xxx.xxx.xxx (your tftp server)
bootcmd=dhcp;bootm
bootfile=tftp_path/uImage
```

When you boot, look for the following lines that confirm the boot mode:

```
TFTP from server 192.168.160.71; our IP address is
192.168.161.186
Filename 'library/davinci/0.4.2/uImage'.
...
## Booting image at 80700000 ...
```

A.5.3 Booting from Flash Using NFS File System

To boot in this mode, set the following parameters after you abort the automatic boot sequence:

```
EVM # setenv bootcmd 0x2050000
EVM # setenv nfshost 192.168.160.57
EVM # setenv rootpath /exports/useracct
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
   ip=dhcp
   root=/dev/nfs nfsroot=$(nfshost):$(rootpath),nolock
   mem=120M
```

For example, the environment variables set for this mode might be:

```
bootdelay=3
baudrate=115200
bootfile=library/davinci/0.4.2/uImage
serverip=192.168.160.71
stdin=serial
stdout=serial
stderr=serial
ethaddr=00:0e:99:02:51:46
bootcmd=0x2050000
nfshost=192.168.160.57
rootpath=/exports/useracct
bootargs=console=ttyS0,115200n8 noinitrd rw ip=dhcp
root=/dev/nfs nfsroot=192.168.160.57:/ex
ports/useracct,nolock mem=120M eth=00:0e:99:02:51:46
```

When you boot, look for the following lines that confirm the boot mode:

```
## Booting image at 02050000 ...
...
Starting kernel ...
...
VFS: Mounted root (nfs filesystem).
```

A.5.4 Booting via TFTP Using NFS File System

To boot in this mode, set the following parameters after you abort the automatic boot sequence:

```
EVM # setenv bootcmd 'dhcp;bootm'
EVM # setenv serverip 192.168.160.71
EVM # setenv bootfile library/davinci/0.4.2/uImage
EVM # setenv rootpath /exports/useracct
EVM # setenv nfshost 192.168.160.57
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
   ip=dhcp
   root=/dev/nfs nfsroot=$(nfshost):$(rootpath),nolock
   mem=120M
```

For example, the environment variables set for this mode might be:

```
bootdelay=3
baudrate=115200
stdin=serial
stdout=serial
stderr=serial
ethaddr=00:0e:99:02:51:46
bootcmd=dhcp;bootm
serverip=192.168.160.71
bootfile=library/davinci/0.4.2/uImage
rootpath=/exports/useracct
nfshost=192.168.160.57
bootargs=console=ttyS0,115200n8 noinitrd rw ip=dhcp
root=/dev/nfs nfsroot=192.168.160.57:/ex
ports/useracct,nolock mem=120M eth=00:0e:99:02:51:46
```

When you boot, look for the following lines that confirm the boot mode:

```
TFTP from server 192.168.160.71; our IP address is
192.168.161.186
Filename 'library/davinci/0.4.2/uImage'.
...
Starting kernel ...
...
VFS: Mounted root (nfs filesystem).
```

A.6 Installing Components Under Windows

A Linux host is the recommended development platform for DVEVM development. However, it is possible to use a Microsoft Windows host for such development. Since the expected host is a Linux platform, less documentation is provided for Windows host build issues.

The DVEVM provides the following Microsoft Windows installers for customers who already have the Windows version of MontaVista Linux Tools:

❑ DVEVM: mvl_lsp_setupwin32_#_##_##_##.exe

❑ DVEVM: dvevm_setupwin32_#_##_##_##.exe

The DVEVM does not include the Windows version of the MontaVista Linux Tools. If you want to use a Windows host for DaVinci development, you will need to obtain the Windows version of the MontaVista Linux Tools from Texas Instruments. You must also have the Windows version of the MVL Target (cross compilers that run on Windows to generate target-side code).

To set up a Windows host for DaVinci development, follow these steps:

1) Run all the Windows installers in the sequence listed above. The DVEVM software installers expect the MontaVista Linux tools to have already been installed.

2) When running the installers (even if you are in a MVL bash shell), specify a full Windows-style path with backslashes. For example, C:\mvcyg4.0\home\useracct.

3) Use a text editor to edit the Rules.make file. Make the following changes:

   ■ MVTOOL_PREFIX: Specify the prefix to be added before the GNU compiler tools. Use UNIX format (forward slashes) with a preceding C:. For example, you might use:

```
C:/mvcyg4.0/opt/mv_pro_4.0/montavista/pro/devkit/arm/v5t_le/bin/arm_v5t_le-
```

   ■ EXEC_DIR: Specify the location to which the resulting executables and data should be copied using UNIX format (forward slashes). For example, C:/mvcg4.0/opt/dvevm.

A.7 Rebuilding DSP/BIOS Link

If you want to rebuild the DSP/BIOS Link package, follow these steps (assuming you are using the bash shell):

1) Edit the davinci_mvlpro4.0.mk file, which is in the ~/dvevm_#_##/dsplink_1_##_##_##/packages/dsplink/make/Linux/ directory, to make sure the BASE_BUILDOS and BASE_CGTOOLS variables correctly point to the correct locations.

2) Define the DSPLINK environment variable to be the absolute path to the "dsplink" directory. (Use export for bash shell, and setenv for tcsh shell.) For example:

```
host $ export DSPLINK=/home/useracct/dvevm_#_##/dsplink_1_30_08_02/packages/dsplink
```

3) Move to the Linux build script directory:

```
host $ cd $DSPLINK/etc/host/scripts/Linux
```

4) Build DSP/BIOS Link as follows:

```
host $ sh -f buildmodule.sh
```

5) The rebuilt kernel module is called dsplinkk.ko. It is located in the $DSPLINK/gpp/export/BIN/Linux/Davinci/RELEASE/ directory.

A.8 Restoring and Updating the DVEVM Hard Disk Drive

This section describes how to restore and update all the files on the DVEVM hard disk drive (HDD), including the MontaVista file system and the demos. Using these restore procedures, you can return your board to a known state, should anything happen to the data on the board's HDD.

This section assumes that you have configured a host Linux workstation with the software necessary to perform an NFS root mount with the DVEVM as described in Section 4.3.4 and Section 4.3.5.

In this section, U-Boot is always located at the start of flash memory (address 0x02000000) on the target. Similarly, uImage, the Linux kernel program, is booted from the target flash memory address of 0x02050000.

A.8.1 System Setup

You should make sure the following system setup steps have been performed before you attempt to restore or update the hard disk drive:

1) Inspect jumper J4, which is labeled "CS2 SELECT". Make sure FLASH is selected.
2) Connect the Ethernet port of the host workstation to a router. Configure the host Ethernet port to obtain IP address dynamically via a DHCP server running inside the router.
3) Connect the Ethernet port of the target DVEVM to another port on the same router. This establishes a network connection with your host workstation.
4) Connect an RS-232 cable from the UART0 port of the target DVEVM board to the host workstation.
5) On the host workstation, open a terminal session to the target DVEVM board with the following characteristics:

   - Bits per Second: 115200
   - Data Bits: 8
   - Parity: None
   - Stop Bits: 1
   - Flow Control: None

   For example, you can create a terminal session with HyperTerminal or TeraTerm on MS Windows, and Minicom or C-Kermit on Linux.

6) Start an NFS server on the host workstation. This document assumes the host path /home/user/workdir/filesys contains a file system that the target DVEVM can use for root mounting.

A.8.2 Configure DVEVM for NFS Root Mount

Follow these steps to configure your DVEVM for an NFS root mount:

1) Configure the Boot Switches (S3) to 1011111110. This is the red bank of switches in the middle of the DVEVM.

2) Power on the DVEVM and hit any key to enter U-Boot.

3) Configure bootcmd as follows to boot the Linux kernel via Flash.

```
EVM # setenv bootcmd bootm 0x2050000
```

4) Configure bootargs as follows to root mount the file system from NFS:

```
EVM # setenv bootargs 'console=ttyS0,115200n8 noinitrd rw ip=dhcp
root=/dev/nfs nfsroot=192.168.1.102:/home/user/workdir/filesys,nolock mem=120M'
```

The nfsroot option in this command uses the host workstation IP address. Make sure to replace the IP address 192.168.1.102 with the actual address of your host Linux workstation.

5) Optional: Print the U-Boot parameters

```
EVM # printenv
```

6) Save the U-Boot parameters

```
EVM # saveenv
```

7) Boot DVEVM from NFS on the host Linux workstation

```
EVM # boot
```

8) Log into MontaVista Linux as "root".

A.8.3 Restore the DVEVM Hard Disk Drive

The DVEVM hard disk drive (HDD) can be restored from a target DVEVM HDD partition or from the host Linux workstation file system. It is not necessary to follow both methods; they both achieve the same result.

Restoring the DVEVM HDD takes 10 to 15 minutes. The restore script must uncompress 600 MB of compressed data and load it to the /dev/hda1 partition.

After the hard drive restore process has completed, make sure to restart the DVEVM and configure U-Boot to root mount via the local HDD. The steps for this type of boot are provided in Section A.5.1, *Booting from Flash Using Board’s Hard Drive File System*.

A.8.3.1 Restoring From Target DVEVM HDD Partition

Follow these steps to restore the HDD from the restore partition on the HDD itself:

1) Make a directory for mounting the HDD restore partition:

```
EVM # mkdir /mnt/restore
```

2) Mount the HDD restore partition:

```
EVM # mount -t ext3 /dev/hda2 /mnt/restore
```

3) Set the Linux date variable to today's date. If the date is too far off, the target file system installation generates a bunch of warnings.

```
EVM # date MMDDHHMMCCYY
```

For example, for 9:00 am on April 18th, 2006, enter 041809002006.

4) Change directory to /mnt/restore:

```
EVM # cd /mnt/restore
```

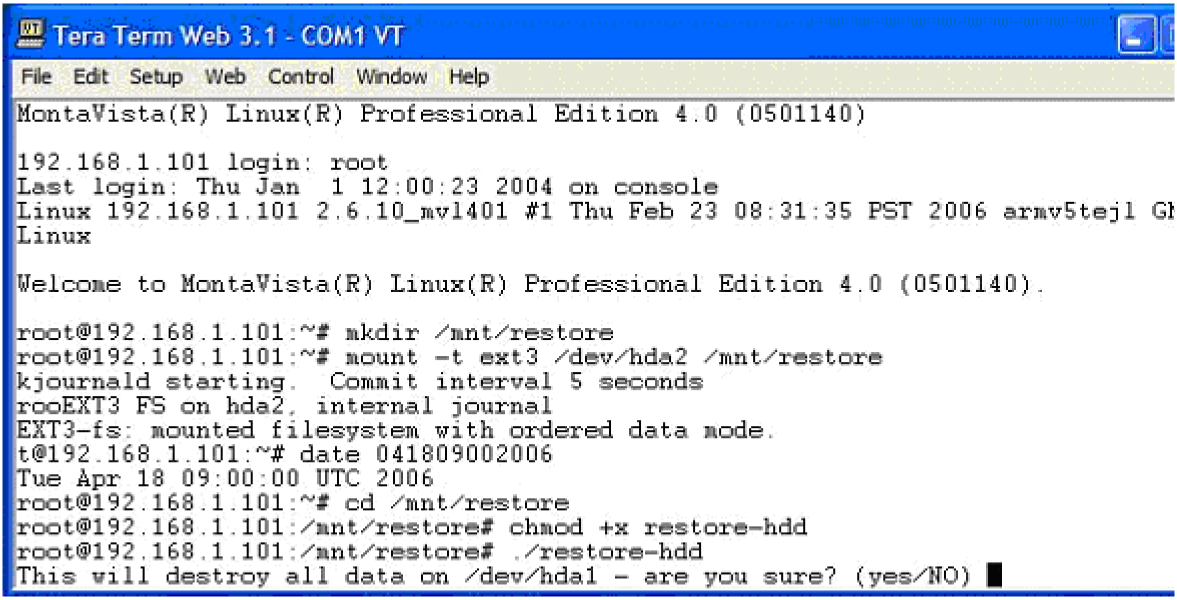
5) Add execute permissions for the script:

```
EVM # chmod +x restore-hdd
```

6) Run the restore script:

```
EVM # ./restore-hdd
```

7) The script will ask for confirmation: "This will destroy all data on /dev/hda1 - are you sure?" Type **yes**.


Tera Term Web 3.1 - COM1 VT
File Edit Setup Web Control Window Help

```
MontaVista(R) Linux(R) Professional Edition 4.0 (0501140)

192.168.1.101 login: root
Last login: Thu Jan  1 12:00:23 2004 on console
Linux 192.168.1.101 2.6.10_mvl401 #1 Thu Feb 23 08:31:35 PST 2006 armv5tejl GN
Linux

Welcome to MontaVista(R) Linux(R) Professional Edition 4.0 (0501140).

root@192.168.1.101:~# mkdir /mnt/restore
root@192.168.1.101:~# mount -t ext3 /dev/hda2 /mnt/restore
kjournald starting.  Commit interval 5 seconds
rooEXT3 FS on hda2, internal journal
EXT3-fs: mounted filesystem with ordered data mode.
t@192.168.1.101:~# date 041809002006
Tue Apr 18 09:00:00 UTC 2006
root@192.168.1.101:~# cd /mnt/restore
root@192.168.1.101:/mnt/restore# chmod +x restore-hdd
root@192.168.1.101:/mnt/restore# ./restore-hdd
This will destroy all data on /dev/hda1 - are you sure? (yes/NO) █
```

8) After the HDD restore is complete, shutdown the DVEVM:

```
EVM # halt
```

9) When the "Power down" message is printed in the terminal window, it is safe to power down the DVEVM.

10) Restart the DVEVM and configure U-Boot to root mount via the local HDD. Follow the steps in Section A.5.1, *Booting from Flash Using Board’s Hard Drive File System*.

A.8.3.2 Restoring From Host Linux Workstation File System

This section assumes that you have installed the DVEVM software to the host Linux /workdir/filesys/restore directory. After an NFS mount, this is equivalent to /restore for the target DVEVM.

Follow these steps to restore the HDD from the host Linux workstation restore directory:

1) Login to the DVEVM as root.

2) Go to the /restore directory.

```
EVM # cd /restore
```

3) Set the Linux date variable to today's date. If the date is too far off, the target file system installation generates a bunch of warnings.

```
EVM # date MMDDHHMMCCYY
```

For example, for 9:00 am on April 18th, 2006, enter 041809002006.

4) Add execute permissions on the restore-hdd script.

```
EVM # chmod +x restore-hdd
```

5) Run the script in the /restore directory.

```
EVM # ./restore-hdd
```

6) The script will ask for confirmation: "This will destroy all data on /dev/hda1 - are you sure?" Type **yes**.

7) After the HDD restore is complete, shutdown the DVEVM:

```
EVM # halt
```

8) When the "Power down" message is printed in the terminal window, it is safe to power down the DVEVM.

9) Restart the DVEVM and configure U-Boot to root mount via the local HDD. Follow the steps in Section A.5.1, *Booting from Flash Using Board’s Hard Drive File System*.

A.8.4 Update DVEVM Hard Disk Drive from Update Website

You can download and install updated DVEVM HDD software from the DVEVM Update website. To do this, follow these steps:

1) Download the latest DVEVM Software Update (dvevm_setuplinux_#_##_##_##.bin) from the following location to your host file system:

```
https://www-a.ti.com/extranet/cm/product/dvevmsw/dspswext/general/homepage.shtml
```

2) Follow the steps in Section A.8.3.2, *Restoring From Host Linux Workstation File System* to restore the target DVEVM HDD from the the host file system.

Index