TMS320DM357 DVEVM v2.05
Getting Started Guide

Literature Number: SPRUGH0
December 2008

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	
Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DSP	dsp.ti.com
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
Low Power Wireless	www.ti.com/lpw

Applications	
Audio	www.ti.com/audio
Automotive	www.ti.com/automotive
Broadband	www.ti.com/broadband
Digital Control	www.ti.com/digitalcontrol
Military	www.ti.com/military
Optical Networking	www.ti.com/opticalnetwork
Security	www.ti.com/security
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

Copyright © 2008, Texas Instruments Incorporated

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

Copyright © 2008, Texas Instruments Incorporated

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 DaVinci architecture and lets users evaluate the power and performance of DaVinci as a Multimedia engine.

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

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.

December 15, 2008

Contents

1 DVEVM Overview 1-1
This chapter introduces the DVEVM (Digital Video Evaluation Module).
1.1 What's in this Kit? 1-2
1.2 What's on the Board? 1-3
1.3 What's Next? 1-4

2 EVM Hardware Setup 2-1
This chapter tells you how to set up the EVM hardware.
2.1 Setting Up the Hardware 2-2
2.2 Connecting to a Console Window 2-6

3 Running the Demonstration Software 3-1
This chapter explains how to run the software demos provided with the DVEVM kit.
3.1 Default Boot Configuration 3-2
3.2 Starting the Standalone Demos 3-2
3.3 Running the Standalone Demos 3-5
3.3.1 Shutting Down the Demos 3-6
3.3.2 About the Encode + Decode Demo 3-7
3.3.3 About the Encode Demo 3-7
3.3.4 About the Decode Demo 3-9
3.4 Running the Demos from the Command Line 3-10
3.5 Running the Network Demo 3-11

4 DVEVM Software Setup 4-1
This chapter explains how to use the software provided with the DVEVM.
4.1 Software Overview 4-2
4.1.1 Command Prompts in This Guide 4-3
4.1.2 Software Components 4-4
4.2 Preparing to Install 4-5
4.3 Installing the Software 4-6
4.3.1 Installing the Target Linux Software 4-6
4.3.2 Installing the DVSDK Software 4-7
4.3.3 Installing the A/V Demo Files 4-8
4.3.4 Exporting a Shared File System for Target Access 4-8
4.3.5 Testing the Shared File System 4-10
4.3.6 Notes on Using Evaluation/Production Codecs 4-11

4.4 Setting Up the Build/Development Environment 4-12
4.4.1 Writing a Simple Program and Running it on the EVM 4-12
4.5 Building a New Linux Kernel 4-13
4.6 Rebuilding the DVEVM Software for the Target 4-14
4.7 Building with DSPLink 4-15
4.8 Booting the New Linux Kernel 4-16
4.9 Using the Digital Video Test Bench (DVTB) 4-17

A Additional Procedures A-1
This appendix describes optional procedures you may use depending on your setup and specific needs.
A.1 Changing the Video Input/Output Methods A-2
A.2 Putting Demo Applications in the Third-Party Menu A-3
A.3 Setting Up a TFTP Server A-5
A.4 Alternate Boot Methods A-6
A.5 Updating/Restoring the Bootloaders A-9
A.6 Restoring the NAND Flash A-12

Chapter 1

DVEVM Overview

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

Topic **Page**

1.1 What's in this Kit? 1–2
1.2 What's on the Board? 1–3
1.3 What's Next? 1–4

1.1 What's in this Kit?

Your TMS230DM357 DVEVM kit contains the following hardware items. Section 2.1, *Setting Up the Hardware* tells how to connect these components.

- ❏ **EVM Board** This board contains a DaVinci TMS320DM357 Digital Media System-on-Chip.
- ❏ **Universal Power Supply.** Both U.S. and European power are supported.
- ❏ **Cables.** Serial and Ethernet cables are included to allow for host development.
- ❏ **IR Remote Control** (Phillips). This universal remote control is included to provide a user interface to the demo applications.

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

- ❏ DaVinci Digital Software Developer's Kit, including TI DaVinci Demonstration Version of MontaVista Linux Pro v5.0. (2 DVDs)
- ❏ Spectrum Digital EVM Tools

1.2 What's on the Board?

The EVM comes loaded with peripherals your multimedia applications may need to make use of. The following block diagram shows the major hardware components.

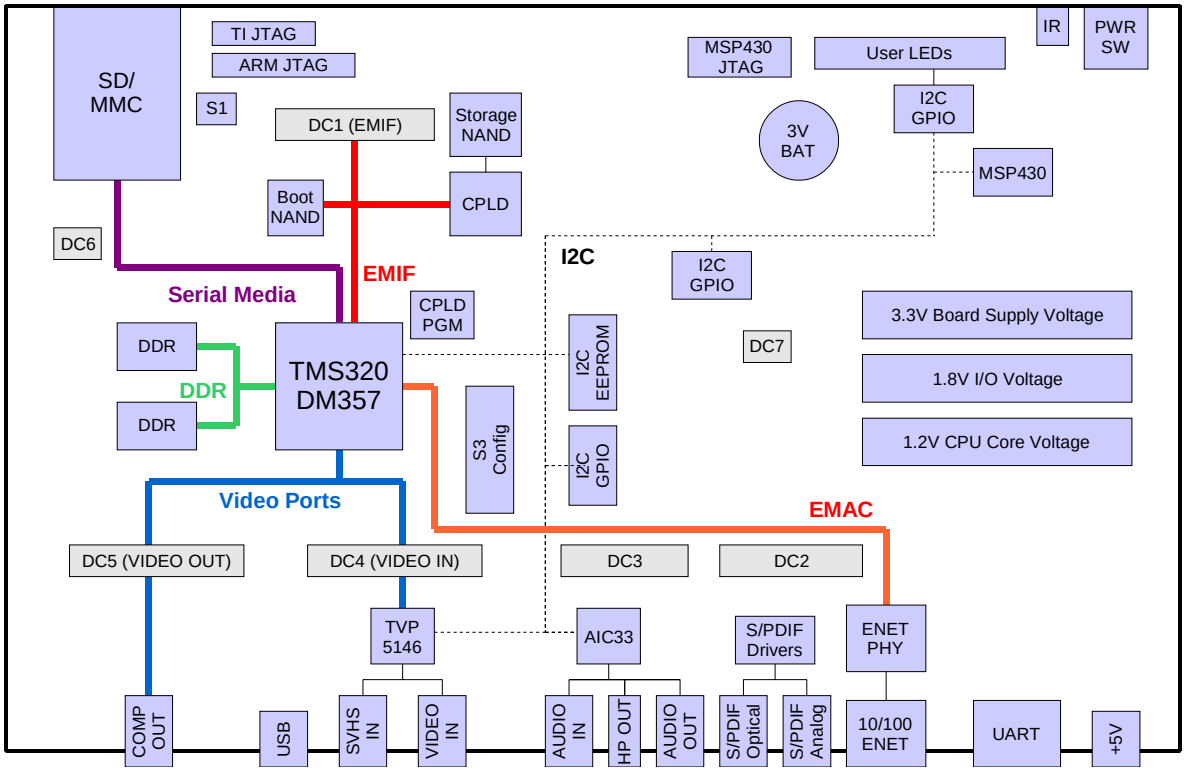


Diagram provided courtesy of Spectrum Digital Inc.

Figure 1–1 DM357 Hardware Block Diagram

For more information about the hardware, see the Spectrum Digital website at http://support.spectrumdigital.com/boards/evmdm357.

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.

1.3 What's Next?

To get started evaluating the DVEVM kit and developing applications for the DM357, 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 DM357 architecture, see the following:

- Spectrum Digital website: http://support.spectrumdigital.com/boards/evmdm357
- TI DaVinci Software Updates: http://www.ti.com/dvevmupdates
- TI Linux Community for DaVinci Processors: http://linux.davincidsp.com
- *Codec Engine Application Developer's Guide* (SPRUE67)
- TI DaVinci Technology Developers Wiki: http://wiki.davincidsp.com
- Other PDF documents on the DVDs included with the DVEVM kit

Chapter 2

EVM Hardware Setup

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

Topic **Page**

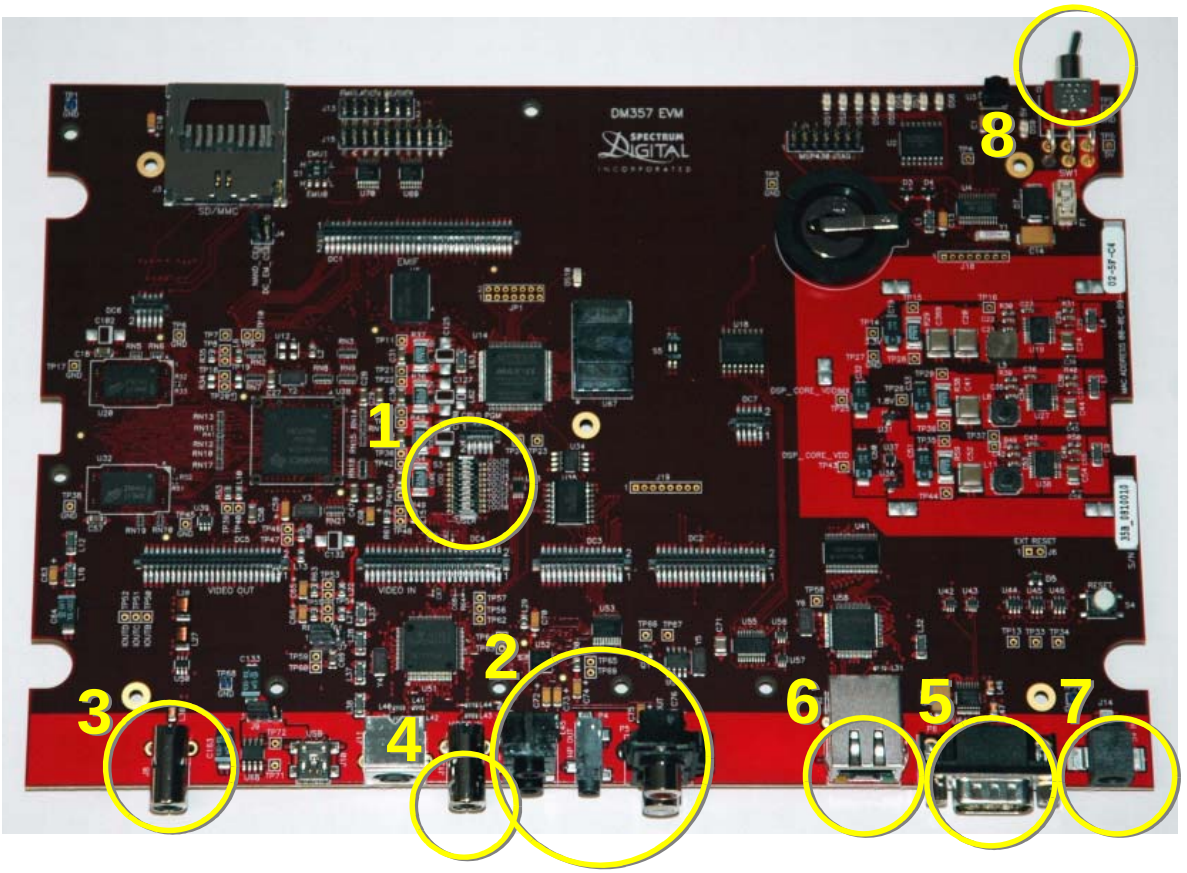
2.1 Setting Up the Hardware 2–2

2.2 Connecting to a Console Window 2–6

2.1 Setting Up the Hardware

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

For reference, the numbers in the following photo of the DM357 EVM target board correspond to the steps in the procedure.

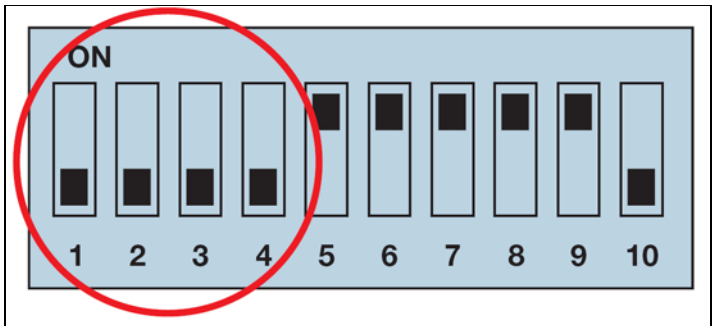


Important: The EVM board is sensitive to static discharges. Use a grounding strap or other device to prevent damaging the board. Be sure to connect communication cables before applying power to any equipment.

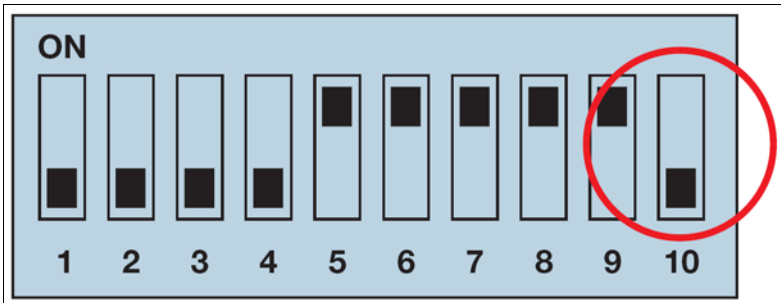


1) On the S3 mini-dip switch, verify that positions 1 through 4 are configured to boot from on-board NAND as in the following figure, where the black rectangle is the switch location.



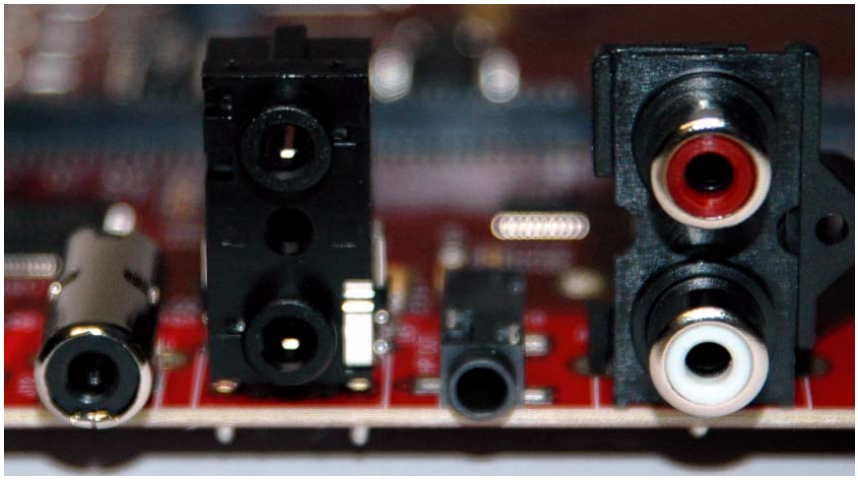


Also on the S3 mini-dip switch, verify that position 10 selects the correct video format—NTSC or PAL. The following diagram shows the switch in the NTSC setting.

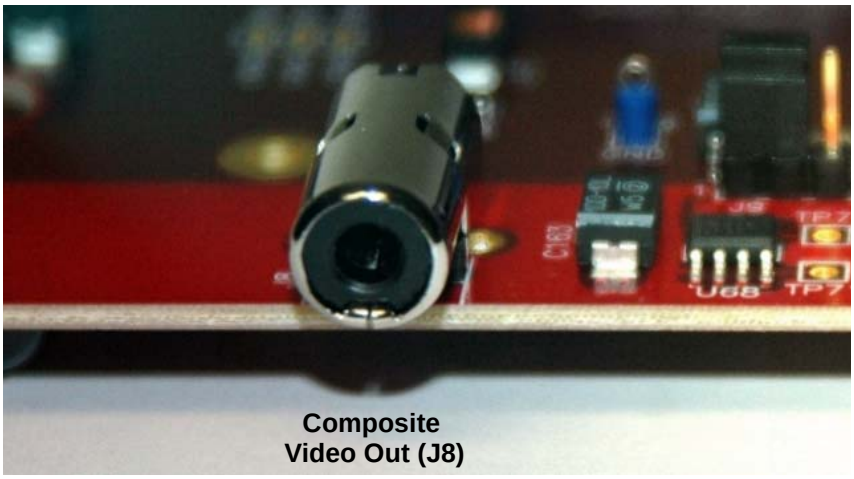


2) Connect an audio speaker to Stereo Line Out (P5) and an audio source to Stereo Line In (P3).

Audio In (P3) **Audio Out (P5)**

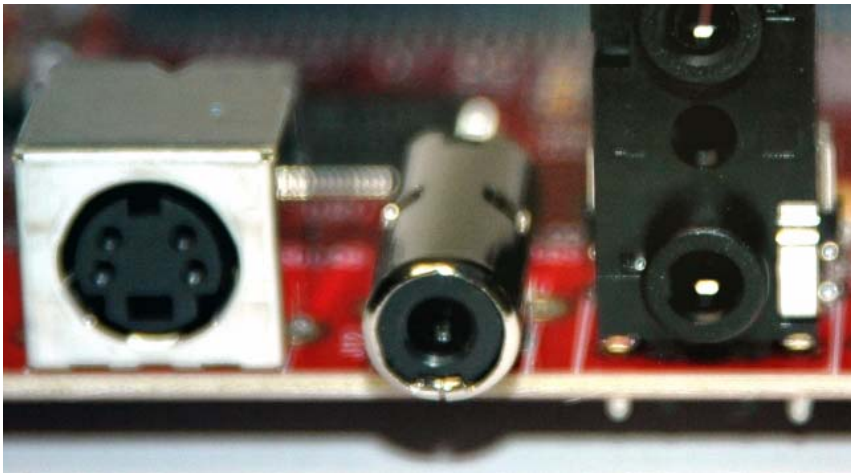


3) Connect your video display to the composite video-out RCA connector (J8).



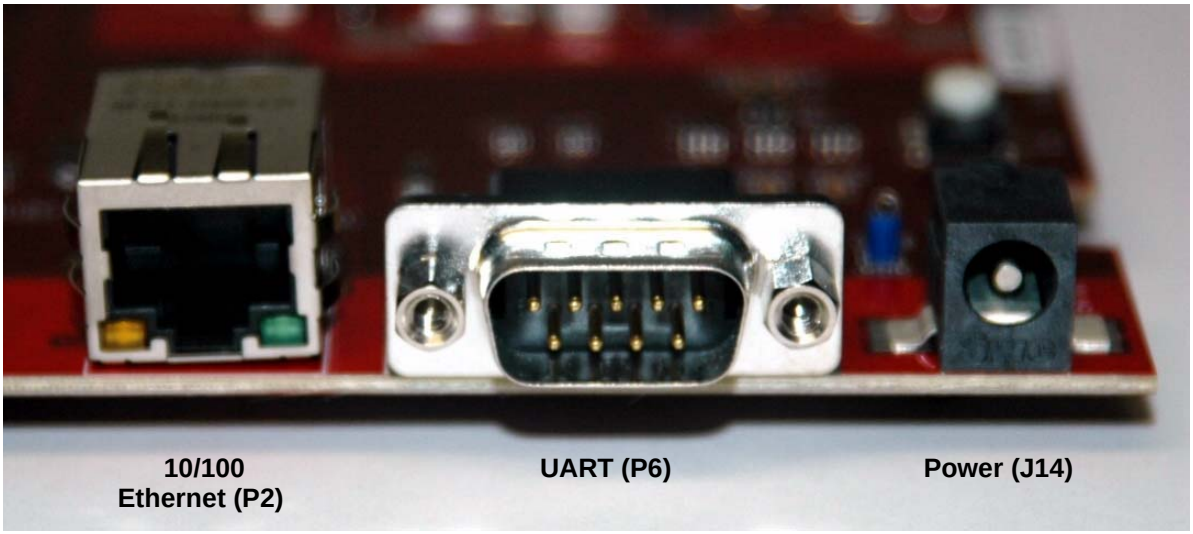
Composite Video Out (J8)

4) Connect a video source (such as a camera or DVD player) to the composite video-in RCA connector (J12). Then power on your video input and output devices.



Composite Video In (J12)

5) (Optional) If you plan to use the UART port for a console window, connect the provided RS-232 null modem cable to the EVM UART port (P6) and to a COM port on your host Linux workstation.

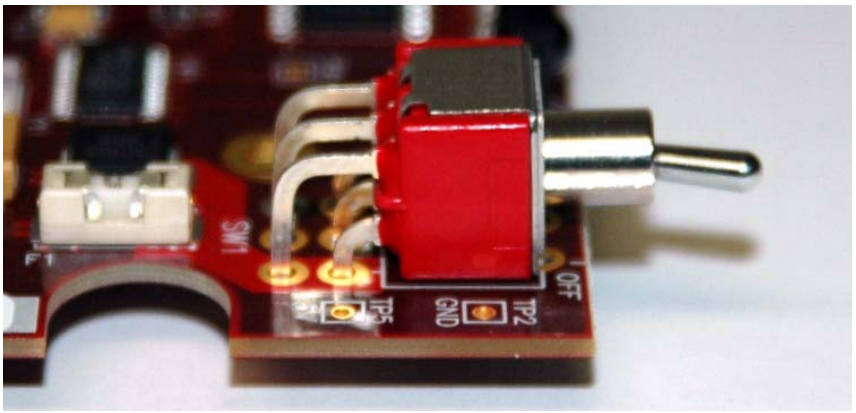


6) (Optional) If you will be using an Ethernet connection, connect the provided Ethernet cable to the Ethernet port (P2) on the EVM board and to an Ethernet network port.

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

7) Connect the provided power cable to the DVEVM power jack (J14) on the board. To be ESD safe, plug in the other end of the cable only after you have connected the power cord to the board.

8) Power on the EVM board by flipping the power switch (SW1).



9) You should see the initial screen of the demo software on your video display. Use the IR remote to run the software as described in Chapter 3.

2.2 Connecting to a Console Window

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

1) Connect a serial cable between the serial port on the EVM and the serial port (for example, COM1) on a PC.

2) Run a terminal session (such as Minicom on Linux or HyperTerminal on Windows) on the workstation and configure it to connect to that serial port with the following characteristics:

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

3) When you power on the EVM, 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 kit.

Topic **Page**

3.1 Default Boot Configuration 3–2
3.2 Starting the Standalone Demos 3–2
3.3 Running the Standalone Demos 3–5
3.4 Running the Demos from the Command Line 3–10
3.5 Running the Network Demo 3–11

3.1 Default Boot Configuration

Out of the box, the EVM boots from flash and 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.

Note: The default U-Boot bootargs definition sets "ip=off", which disables the Ethernet connection.

The out-of-the-box boot parameters are listed in Section A.4.1. The following are alternate ways you may want to boot the board:

- ❏ TFTP boot with NAND flash file system (Section A.4.2)
- ❏ Flash boot with NFS file system (Section A.4.3)
- ❏ TFTP boot with NFS file system (Section A.4.4)
- ❏ PAL video mode vs. NTSC video mode (Section 2.1)

To abort the standard boot, press any key in the console window (see Section 2.2). Also see Section A.4, *Alternate Boot Methods* if you want to change the boot configuration.

> **Note:** It is best to power down the board cleanly in order to reboot, rather than using the reset button or power switch. We recommend that you use the shutdown command to shut down the operating system and unmount the file system before removing power from the board. This will help prevent corruption of the flash-based root file system.

3.2 Starting the Standalone Demos

When you connect the EVM hardware, the pre-loaded examples run automatically on your video 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. This is the mode documented in the rest of this chapter.

  The standalone demo was set up by the DVSDK, which copies the file /examples/dvevmdemo to the directory /etc/rc.d/init.d (the central repository for startup scripts). This file is symbolically linked to /etc/rc.d/rc3id/S88demo. When the board boots up and enters runlevel 3, this file is executed to start the demo web server and the demo interface.

- **Command line.** Once you have connected the EVM 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. For further information on running the demos from the command line, see the demo documentation that is linked to by the DVSDK release notes.

  Note: When you run the demos from the command line, make sure the *interface* process used by the standalone mode demos is not running. Otherwise you will see error messages raised when device drivers fail to open.

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

The order of the buttons on the actual remote may be different from the picture; if your remote looks different, find the buttons with the same labels on your remote.

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 kit 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 "0020" 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+0020 code, do a full reset by removing the batteries, pressing the Power button 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.

2) Use the up and down arrows to change which demo is selected. Then, press "OK" or "Play" to move to the selected demo.

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.

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 Encode+Decode and Decode demos. Press "Rec" (record) twice to begin the Encode demo.

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.



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. The first time you press "Stop", you return to the settings screen. Press "Stop" from the settings screen to go back to the main menu.

For information about running the individual demos, see Section 3.3.2 through Section 3.3.4.

The demos use the Codec Engine to allow applications to run algorithms.

3.3.1 Shutting Down the Demos

You can quit out of the demos completely while at the main menu screen by pressing "Power" on the remote.

> **Note:** It is best to power down the board cleanly in order to reboot rather than using the reset button or power switch. We recommend that you use the shutdown command to shut down the operating system and unmount the file system before removing power from the board. This will help prevent corruption of the flash-based root file system.

To restart the demos, you can reboot the board or run the demos from the command line as described in Section 3.4.

3.3.2 About the Encode + Decode Demo

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

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

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

IR Remote Button	Mode	Action Performed
Up/Down	--	-- no action --
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 video signal is passed to video encoders and decoders by the Codec Engine.

To use this demo from the command line, see Section 3.4, *Running the Demos from the Command Line*.

3.3.3 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 EVM's NAND flash. The possible filenames are demo.264, demo.mpeg4, demo264.g711, and demompeg4.g711. Older versions of these files are overwritten as needed.

The encode demo has a five minute time limit to prevent the demo from filling up the NAND file system.

Output is not decoded and sent to your video 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 H.264 (.264 extension) and MPEG4 (.mpeg4 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
Record (twice) or OK	Setup / Run	Begin encode demo, send unencoded data to display
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 Record 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 by the Codec Engine.

To use this demo from the command line, see Section 3.4, *Running the Demos from the Command Line*.

3.3.4 About the Decode Demo

> **Note:** You must run the Encode demo before you can run the Decode demo unless you have placed appropriately-named audio and video files on the EVM's NAND flash storage device. If you see a message that says "File Not Found", please run the Encode 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 EVM's NAND flash. The decoded signals are sent to your video display and speakers.

The supported video algorithms are H.264 (.264 extension) and MPEG4 (.mpeg4 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
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 by the Codec Engine.

To use this demo from the command line, see Section 3.4, *Running the Demos from the Command Line*.

3.4 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*.

Before running demo applications from the command line, the CMEM and accelerator kernel modules must be loaded. Use "lsmod" to see if they are loaded. If not, use the following commands to load these modules:

```
Target $ cd /opt/dvsdk/dm357
Target $ ./loadmodules.sh
```

To see the command-line options for the demos, use one of the following commands with the -h or --help option:

```
Target $ ./encodedecode -h

Target $ ./encode -h

Target $ ./decode -h
```

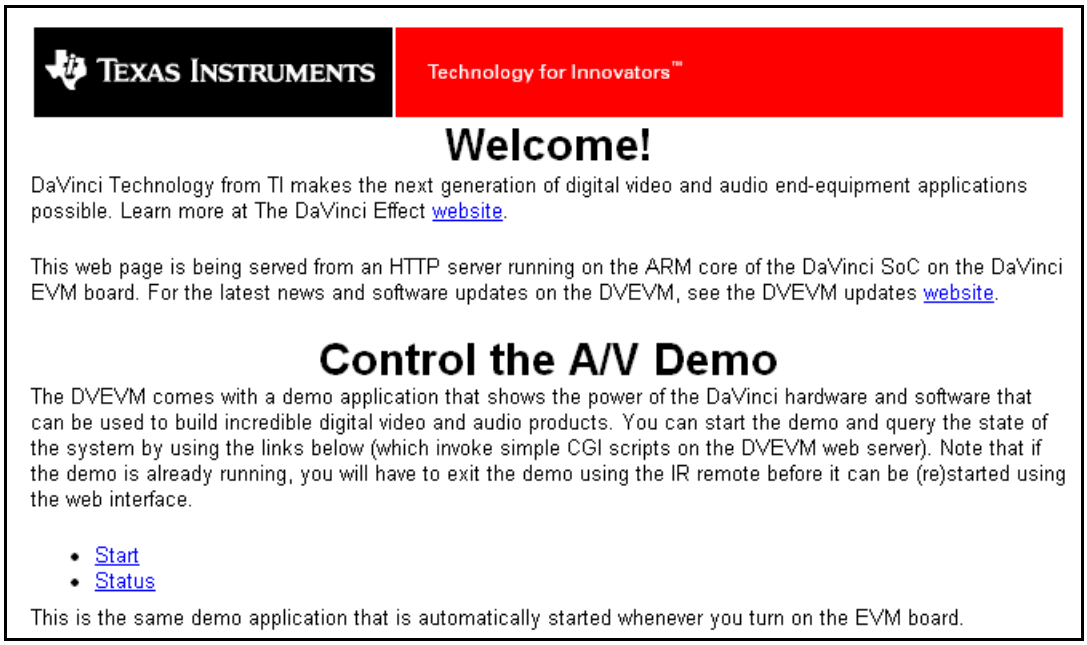
You can also find the list of command-line options in encode.txt, decode.txt, and encodedecode.txt in the respective demo directories of the DVSDK package on the host.

3.5 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 as part of the Linux startup sequence. It configured to service requests from web browsers on the standard TCP/IP port 80.

After the EVM board has booted, connect a PC to the same network to which the EVM board is connected. Enter a URL of the form "http://ip-address-of-evm" 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.



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 EVM 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/dvsdk/dm357/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 4–2
4.2 Preparing to Install 4–5
4.3 Installing the Software 4–6
4.4 Setting Up the Build/Development Environment 4–12
4.5 Building a New Linux Kernel 4–13
4.6 Rebuilding the DVEVM Software for the Target 4–14
4.7 Building with DSPLink 4–15
4.8 Booting the New Linux Kernel 4–16
4.9 Using the Digital Video Test Bench (DVTB) 4–17

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 distribution disks or the files they contain to get started.

The DaVinci software approach provides interoperable, optimized, production-ready video and audio codecs that leverage 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 following software is provided with the ARM-side DVEVM software.

- **Standalone demonstration software.** This is provided on the EVM's NAND flash. 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* and Section 3.5, *Running the Network Demo*.
- **Disk 1: MontaVista Linux Pro v5.0 System Tools and Target File System.** The version provided with the DVEVM kit is the demonstration version. It contains the following file:
  - mvl_5_0_demo_sys_setuplinux.bin. This installation file contains the MontaVista Tool development tool chain and the target file system.
- **Disk 2: TI DVSDK Software.** This DVD includes demo applications, Codec Engine software, example codec servers, and DVEVM documentation. It contains the following files:
  - this manual in PDF format
  - dvsdk_dm357_setuplinux_#_#_#_#.bin (DVSDK installer)
  - mvl_5_0_0_demo_lsp_setuplinux_#_#_#_#.bin
  - xdctools_setuplinux_#_#_#.bin (XDCtools installer)
  - data.tar.gz (Contains A/V data files for use by the demos)
  - restore/dm357_flash_image_#_#_#_#.tar (Contains files for NAND flash recovery. Contact TI Technical Worldwide Support if you need details.)
  - restore/overlay.tar.gz (Contains demo files in case they are needed for recovery. You can ignore this file.)
  - bios_setuplinux_#_#_#.bin (You can ignore this DSP/BIOS installer because the DM357 EVM contains no DSP.)
- **Disk 3: SDI Board Support Software.** It contains EVM board utilities.

Texas Instruments, in agreement with MontaVista Software Inc., is providing a demonstration version of the Linux Professional Edition v5.0 embedded operating system and development tools. The base DVEVM kit includes a demonstration version. The demo version is a subset of what MontaVista provides with the full Professional Edition. Tools such as DevRocket™ 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 DaVinci Software Production Bundle (DVSPB), which includes the production release of this demonstration version of MontaVista Linux. This includes a full MontaVista license and the DevRocket IDE.

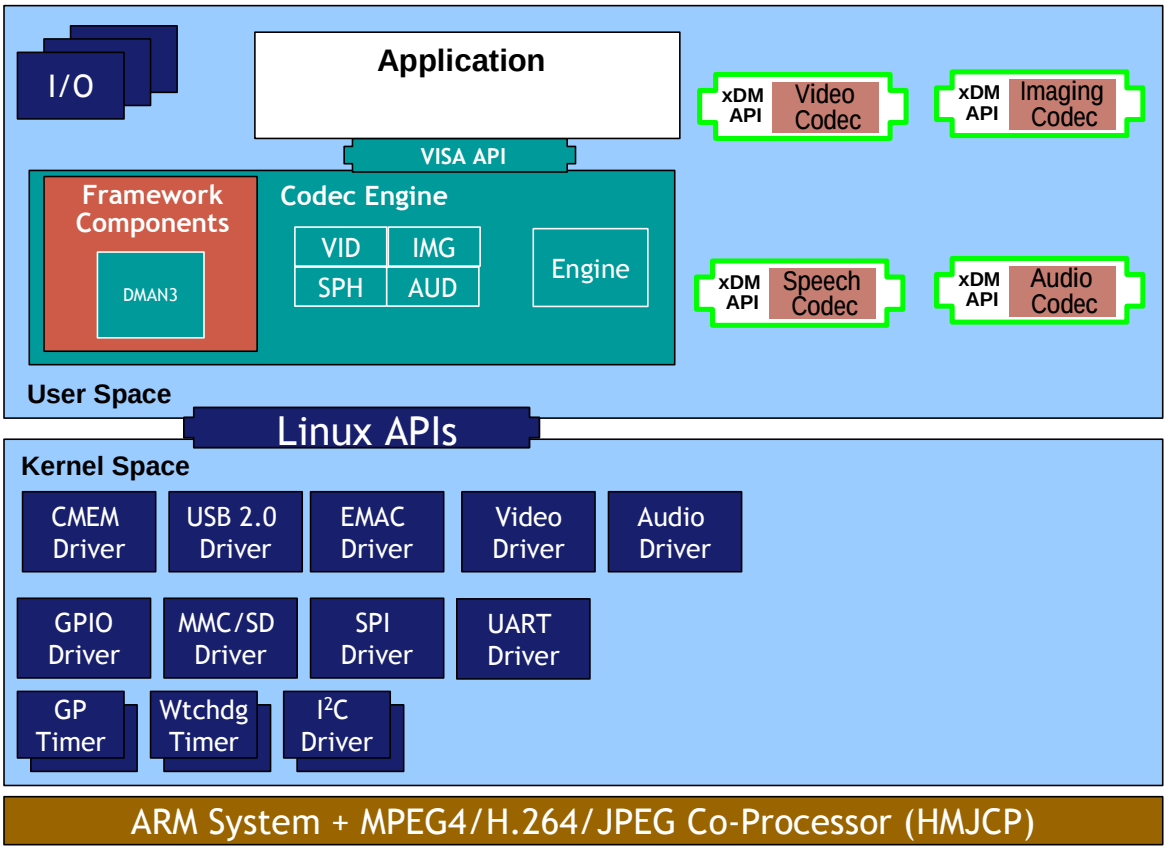
4.1.1 Command Prompts in This Guide

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 in the DVEVM kit:



In the previous figure, everything runs on the ARM. The application 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 xDM-based codecs. For more information, see the *Codec Engine Application Developer's Guide* (SPRUE67).

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

4.2 Preparing to Install

On a host system, mount the DVEVM demonstration DVD and copy the following 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_5_0_demo_sys_setuplinux.bin (disk 1)
- ❏ mvl_5_0_0_demo_lsp_setuplinux_#_#_#_#.bin (disk 2)
- ❏ dvsdk_dm357_setuplinux_#_#_#_#.bin (disk 2)
- ❏ xdctools_setuplinux_#_#_#.bin (disk 2)

Updates to these installers may be available on the TI DaVinci Software Updates website listed in Section 1.3.

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 or bash:

```
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 DVSDK 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*

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 v5.0.

Note that separate versions of Linux are used by the target and your host Linux workstation. The following Linux host operating system is tested with the DVEVM:

- ❏ Red Hat Enterprise Linux v4 (Server Edition)

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 in order to extract the installers for the Linux tools, Linux kernel, and the file system. If a bin file does not run, make sure these files are executable (use `chmod +x *.bin`).

   Instead of the default installation directory, we suggest that you change the installation folder to /opt/mv_pro_5.0 directory.

   ```
   host $ ./mvl_5_0_demo_sys_setuplinux.bin
   host $ ./mvl_5_0_0_demo_lsp_setuplinux_#_#_#_#.bin
   ```

3) After you execute these .bin files, make sure the following files are located in /opt/mv_pro_5.0 (or in the /mv_pro_5.0 subdirectory of the directory you chose in place of the default):

   - mvltools5.0-########.tar.gz
   - DaVinciLSP-#_#_#_#.tar.gz

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

   ```
   host $ cd /opt/mv_pro_5.0
   ```

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

```
host $ tar zxf mvltools5.0-#######.tar.gz
host $ tar zxf DaVinciLSP-#_#_#_#.tar.gz
```

This creates the MontaVista directory structure under the /opt/mv_pro_5.0/montavista/ directory.

Note that unpacking these tar files will overwrite any existing files that were previously installed.

Note: The LSP shipped with the DVSDK is a multi-platform LSP and is not configured for a particular platform. As shipped, this LSP cannot be used to build the demo or example applications. It must first be copied to a user area and configured/built for the EVM. Please see Section 4.5 for instructions.

4.3.2 Installing the DVSDK Software

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

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

1) Log in using a **user account**. The user account must have execute permission for the dvsdk_dm357_setuplinux_#_#_#_#.bin and xdctools_setuplinux_#_#_#.bin files.

2) Execute the DVSDK installer that you previously copied from the DVSDK DVD. For example:

```
host $ cd /tmp
host $ ./dvsdk_dm357_setuplinux_#_#_#_#.bin
```

This installs the DVSDK in /home/<*useracct*>/dvsdk_#_#.

3) Execute the XDC installer that you previously copied from the DVSDK DVD. For example:

```
host $ cd /tmp
host $ ./xdctools_setuplinux_#_#_#.bin
```

When you are prompted, *do not* use the default installation location. Instead, install the software in the directory created in Step 2. For example, /home/<*useracct*>/dvsdk_#_#.

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

Note: You can uninstall these components by using the `rm -rf` command on its directory. You should ignore the uninstall files created by the installer.

4.3.3 Installing the A/V Demo Files

The fourth disk 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 demos directory in the DVSDK directory that you set up previously. For example:

```
host $ cd /home/<useracct>/dvsdk_#_#/dvsdk_demos_#_#_#
```

2) Mount disk 2 and copy the data.tar.gz file to your DVSDK directory. For example:

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

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

```
host $ tar zxf data.tar.gz
```

4.3.4 Exporting a Shared File System for Target Access

Although the board's NAND flash 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 flash 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.

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

```
host $ cd /home/<useracct>
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_5.0, use your location in the cp command.

```
host $ cp -a /opt/mv_pro_5.0/montavista/pro/devkit/arm/v5t_le/target/* .
host $ chown -R <useracct> opt
```

5) Edit the /etc/exports file on the host Linux workstation (not the exports file on the target filesystem). 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)
```

Note: Make sure you do not add a space between the * and the (in the above command.

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 -av
host $ /sbin/service nfs restart
```

Note: Use `exportfs -rav` to re-export all directories. Use `/etc/init.d/nfs status` to verify that the NFS status is running.

7) Verify that the server firewall is turned off:

```
host $ /etc/init.d/iptables status
```

If the firewall is running, disable it:

```
host $ /etc/init.d/iptables stop
```

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 EVM board via RS-232 using the instructions in Section 2.2. If you have a Windows workstation, you can use HyperTerminal. If you have a Linux workstation, you might use Minicom. (You may need to turn on line wrap.)

3) Power on the EVM 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 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=232M
 video=davincifb:vid0=0,2025K:vid1=0,
 1350K:osd0=720x576x16,
 2025K davinci_enc_mgr.ch0_output=COMPOSITE
 davinci_enc_mgr.ch0_mode=ntsc
```

Note that the **setenv bootargs** command should be typed on a single line. Also note that you should avoid using the numeric keypad to enter numbers, as it can sometimes insert extra invisible characters.

The *<directory to mount>* must match what you specified in Step 5 of Section 4.3.4. For example, /home/<*useracct*>/workdir/filesys.

Hints: You may want to use the printenv command to print a list of your environment variables. You can also save these setenv commands in a .txt file from which you can paste them in the future.

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
```

7) You can now log in as "root" with no password required.

See Section A.4, *Alternate Boot Methods* for information about booting with TFTP, NFS, or the board's NAND flash.

4.3.6 Notes on Using Evaluation/Production Codecs

As part of the DM357 DVSDK installation, you received a number of codecs:

- Sequential JPEG Decoder
- Sequential JPEG Encoder
- MPEG4 Restricted Simple Profile Decoder
- MPEG4 Simple Profile Encoder
- H.264 Base Profile Decoder
- H.264 Base Profile Encoder
- G.711 Decoder (not a TI codec)
- G.711 Encoder (not a TI codec)

These codecs are provided under a "for demonstration-only" license agreement. If you wish to use these codecs in a production development environment, you can go to the DVEVM Updates web site at http://www.ti.com/dvevmupdates to download the latest production versions, along with the appropriate license agreement.

4.4 Setting Up the Build/Development Environment

To set up the 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):

```
PATH="/opt/mv_pro_5.0/montavista/pro/devkit/arm/v5t_le/bin:
    /opt/mv_pro_5.0/montavista/pro/bin:
     /opt/mv_pro_5.0/montavista/common/bin:$PATH"
```

   If you installed in a location other than /opt/mv_pro_5.0, use your own location in the PATH.

3) Remember to use the following command after modifying your .bashrc file:

```
host $ source ~/.bashrc
```

4.4.1 Writing a Simple Program and Running it on the EVM

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 /home/<useracct>/workdir/filesys/opt/hello`
2) `host $ cd /home/<useracct>/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 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 EVM 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 EVM* before attempting to build a new kernel.

To rebuild the Linux Kernel, follow these steps:

1) Log in to your user account (not as root).

2) Set the **PLATFORM** variable in the Rules.make file as described in Section 4.6.

3) 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 2.6.18 kernel plus the DaVinci drivers. If you installed in a location other than /opt/mv_pro_5.0, use your location in the cp command.

```
host $ cd /home/<useracct>
host $ mkdir -p workdir/lsp
host $ cd workdir/lsp
host $ cp -R /opt/mv_pro_5.0/montavista/pro/devkit/lsp/ti-davinci .
```

4) 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/linux-2.6.18_pro500
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- davinci_dm357_defconfig
```

5) To modify the kernel options, you will need to use a configuration command such as "make menuconfig" or "make xconfig". To enable the MontaVista default kernel options, use the following command:

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

6) Compile the kernel using the following command:

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

7) If the kernel is configured with any loadable modules (that is, selecting <M> for a module in menuconfig), use the following commands to rebuild and install these modules:

```
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- modules
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le-
   INSTALL_MOD_PATH=/home/<useracct>/workdir/filesys modules_install
```

8) Use the following command to copy 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. (Perform these commands as root or use a `chown uImage` command to get ownership of the file.)

```
host $ cp /home/<useracct>/workdir/lsp/ti-davinci/linux_2.6.18_pro500/arch/arm/boot/uImage /tftpboot
host $ chmod a+r /tftpboot/uImage
```

For more information on setting up a TFTP server, see Section A.3.

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

4.6 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) If you have not already done so, rebuild the Linux kernel as described in Section 4.5.

2) Change directory to dvsdk_#_#.

3) Edit the dvsdk_#_#/Rules.make file.

- Set PLATFORM to match your EVM board as follows:

```
PLATFORM=dm357
```

- Set DVSDK_INSTALL_DIR to the top-level DVEVM installation directory as follows:

```
DVSDK_INSTALL_DIR=/home/<useracct>/dvsdk_#_#
```

- Make sure EXEC_DIR points to the opt directory on the NFS exported file system as follows:

```
EXEC_DIR=/home/<useracct>/workdir/filesys/opt/dvsdk/dm357
```

- Make sure MVTOOL_DIR points to the MontaVista Linux tools directory as follows:

```
MVTOOL_DIR=/opt/mv_pro_5.0/montavista/pro/devkit/arm/v5t_le
```

- Make sure LINUXKERNEL_INSTALL_DIR is defined as follows:

```
LINUXKERNEL_INSTALL_DIR=/home/<useracct>/workdir/lsp/ti-davinci/linux-2.6.18_pro500
```

- Modify the following environment variable as needed to match the location of XDCtools on your Linux host. We recommend that XDCtools be installed in the /home/<*useracct*>/dvsdk_#_# directory, but you may have installed it elsewhere.

```
XDC_INSTALL_DIR=/home/<useracct>/dvsdk_#_#/xdctools_#_#
```

4) While in the same directory that contains Rules.make, use the following commands to build the DVSDK demo applications and put the resulting binaries on the target file system specified by EXEC_DIR.

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

5) You can test the rebuilt DVEVM software by booting your NFS file system and running the demos from the command line as described in Section 3.4.

4.7 Building with DSPLink

The DSPLink build system now requires a GNU make version of 3.81 or greater. Currently, versions of GNU make that qualify are versions 3.81, 3.81beta1, 3.90, and 3.92. Red Hat Linux 3 and 4 usually have GNU make 3.80 pre-installed, which will not build DSPLink.

To work around this, either install and build a version of make 3.81+ or use the make 3.81beta1 distributed with XDCtools. The 3.81beta1 version of make is located in XDC_TOOLS_DIR/gmake.

See the DSPLink documentation for further information about building.

4.8 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 4.5, *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 EVM 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 <tftp server ip address>
EVM # setenv bootfile uImage
EVM # setenv bootargs mem=232M console=ttyS0,115200n8
   root=/dev/mtdblock0 rw rootfstype=yaffs2 ip=dhcp
   video=davincifb:vid0=0,2025K:vid1=0,
   1350K:osd0=720x576x16,
   2025K davinci_enc_mgr.ch0_output=COMPOSITE
   davinci_enc_mgr.ch0_mode=ntsc
EVM # saveenv
```

Note that the **setenv bootargs** command should be typed on a single line.

3) Boot the board:

```
EVM # boot
```

This configuration boots a new Linux kernel via TFTP with a NAND flash based file system. To boot using an NFS file system, see Section A.4.4.

For instructions on how to verify that your host workstation is running a TFTP server, and for instructions on what to do if it isn't, see Section A.3.

For more details on booting, see Section A.4.

4.9 Using the Digital Video Test Bench (DVTB)

The Digital Video Test Bench (DVTB) is a Linux utility that was developed to execute end-to-end data flows using the DVSDK for any platform. DVTB uses the Codec Engine VISA APIs and Linux driver peripheral APIs to encode and decode video, image, audio and speech streams.

Using DVTB, you can configure codecs and/or peripherals before starting a data flow. This enables you to try different use case scenarios and evaluate the system.

The DVSDK installation places DVTB in the /home/<*useracct*>/dvsdk_#_#/dvtb_#_#_# directory, where #_#_# is the DVTB version number.

To install DVTB to the target file system, perform the following steps on the host machine where the DVSDK has been installed:

1) Make sure the Rules.make file defines PLATFORM correctly as described in Section 4.6.

2) Perform the following commands:

```
host $ cd /home/<useracct>/dvsdk_#_#/dvtb_#_#_#
host $ make clean CONFIGPKG=dm357
host $ make CONFIGPKG=dm357
```

3) Copy the binaries "dvtb-d" and "dvtb-r" to /opt/dvsdk/dm357 on the device's target filesystem and run it there. It must be in the same directory as the DSP executables.

For further details on the DVTB, see the following documents:

- **Release Notes.**
  /home/<useracct>/dvsdk_#_#/dvtb_#_#_#/docs/dvtb_release_notes.pdf
- **User Guide.**.
  /home/<useracct>/dvsdk_#_#/dvtb_#_#_#/docs/dvtb_user_guide.pdf

Appendix A

Additional Procedures

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

Topic **Page**

A.1 Changing the Video Input/Output Methods. A–2
A.2 Putting Demo Applications in the Third-Party Menu A–3
A.3 Setting Up a TFTP Server A–5
A.4 Alternate Boot Methods A–6
A.5 Updating/Restoring the Bootloaders. A–9
A.6 Restoring the NAND Flash A–12

A.1 Changing the Video Input/Output Methods

U-Boot reads the S3 mini-dip switch, position 10 setting on boot-up and stores the results in the videostd environment variable. As long as your U-Boot bootcmd sets the video output using the videostd variable (as the example bootcmds in Section A.4, *Alternate Boot Methods* do), you can switch between NTSC and PAL by simply changing the S3 switch setting as shown in Section 2.1, *Setting Up the Hardware*.

To automatically update the bootargs based on the S3 switch setting, please use the following options:

```
EVM # setenv bootargs 'mem=232M console=ttyS0,115200n8
   root=/dev/mtdblock0 rw rootfstype=yaffs2 ip=dhcp
   video=davincifb:vid0=0,2025K:vid1=0,
   1350K:osd0=720x576x16,
   2025K davinci_enc_mgr.ch0_output=COMPOSITE

EVM # setenv bootcmd 'setenv setboot setenv bootargs
   \$(bootargs) davinci_enc_mngr.ch0_mode=\$(videostd);run
   setboot;nboot 0x80700000 0 0x400000;bootm'
```

If you do not want to use the videostd variable in your bootcmd, use the following options within your bootargs setting. The difference between the NTSC and PAL settings is shown in bold.

NTSC

```
video=davincifb:vid0=0,2025K:vid1=0,
1350K:osd0=720x576x16,
2025K davinci_enc_mgr.ch0_output=COMPOSITE
davinci_enc_mgr.ch0_mode=ntsc
```

PAL

```
video=davincifb:vid0=0,2025K:vid1=0,
1350K:osd0=720x576x16,
2025K davinci_enc_mgr.ch0_output=COMPOSITE
davinci_enc_mgr.ch0_mode=pal
```

A.2 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".

```
file <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.3 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 tftp-server-#.#-#.rpm
   ```

   You should see the following output:

   ```
    warning: 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 | grep tftp
   ```

If you want to turn on the TFTP server, use this command:

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

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

A.4 Alternate Boot Methods

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

- ❏ TFTP boot with NAND flash file system (Section A.4.2)
- ❏ Flash boot with NFS file system (Section A.4.3)
- ❏ TFTP boot with NFS file system (Section A.4.4)

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

   Set the environment variables indicated in the following subsections for the boot mode you want to use. (Note that the **`setenv bootargs`** command should be typed on a single line.)

2) If you want to use these settings as the default in the future, save the environment:

```
EVM # saveenv
```

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

```
EVM # boot
```

A.4.1 Booting from Flash Using Board's NAND Flash 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 'nboot 0x80700000 0 0x400000;bootm'
EVM # setenv bootargs console=ttyS0,115200n8 ip=dhcp
   root=/dev/mtdblock0 rw rootfstype=yaffs2 mem=232M
   video=davincifb:vid0=0,2025K:vid1=0,
   1350K:osd0=720x576x16,
   2025K davinci_enc_mgr.ch0_output=COMPOSITE
   davinci_enc_mgr.ch0_mode=ntsc
EVM # boot
```

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

```
## Booting image at 80700000 ...
```

A.4.2 Booting via TFTP Using Board's NAND Flash 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 console=ttyS0,115200n8 ip=dhcp
   root=/dev/mtdblock0 rw rootfstype=yaffs2 mem=232M
   video=davincifb:vid0=0,2025K:vid1=0,
   1350K:osd0=720x576x16,
   2025K davinci_enc_mgr.ch0_output=COMPOSITE
   davinci_enc_mgr.ch0_mode=ntsc
EVM # setenv serverip <tftp server ip address>
EVM # setenv bootfile <kernel image>
EVM # boot
```

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.4.3 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 'nboot 0x80700000 0 0x400000;bootm'
EVM # setenv nfshost <ip addr of 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=232M
   video=davincifb:vid0=0,2025K:vid1=0,
   1350K:osd0=720x576x16,
   2025K davinci_enc_mgr.ch0_output=COMPOSITE
   davinci_enc_mgr.ch0_mode=ntsc
EVM # boot
```

*For example, <directory to mount> might be /home/<useracct>/workdir/filesys.

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

```
## Booting image at 80700000 ...
...
Starting kernel ...
...
VFS: Mounted root (nfs filesystem).
```

A.4.4 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 <ip addr of tftp server>
EVM # setenv bootfile <name of kernel image>
EVM # setenv rootpath <root directory to mount>
EVM # setenv nfshost <ip addr of nfs host>
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
  ip=dhcp root=/dev/nfs nfsroot=$(nfshost):$(rootpath),nolock
  mem=232M video=davincifb:vid0=0,2025K:vid1=0,
  1350K:osd0=720x576x16,
  2025K davinci_enc_mgr.ch0_output=COMPOSITE
  davinci_enc_mgr.ch0_mode=ntsc
EVM # boot
```

The <*root directory to mount*> must match the filesystem that you set up on your workstation. For example, /home/<*useracct*>/workdir/filesys.

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.5 Updating/Restoring the Bootloaders

The DM357 EVM board contains 2 GB of NAND flash memory. During the boot process 64 MB of NAND is used. After booting, 2 GB are used for the file system.

When the EVM board is reset, the ROM bootloader (RBL) executes, initializing the board and then loading a small program called UBL (User Bootloader) from NAND flash memory into internal memory for execution. UBL in turn loads the U-Boot bootloader program from NAND flash memory. The U-Boot bootloader is responsible for loading and starting the Linux kernel.

Therefore, there are two bootloader images that need to be stored in the EVM's NAND flash memory: UBL and U-Boot. This section describes how to flash UBL and U-Boot in case these images are corrupted or need to be updated.

If the U-Boot image is intact in the EVM flash memory, you can use it to update itself. If working U-Boot (or UBL) images are not present in flash, you will need to restore these images using Code Composer Studio (CCStudio) and an emulator. The subsections that follow explain both of these procedures.

You can find UBL, U-Boot, and the NAND programmer pre-built binaries in your DVSDK installation. The location for these is typically the /home/<useracct>/dvsdk_#_#/PSP_#_#_#_#/bin/dm357 directory. Alternatively, the source code for the Bootloader components can be found in the /home/<useracct>/dvsdk_#_#/PSP_#_#_#_#/board_utilities directory.

For further information about upgrading and flashing, see the TI DaVinci Technology Developers Wiki at http://wiki.davincidsp.com.

A.5.1 Updating U-Boot Using U-Boot

If the U-Boot image is intact in the EVM flash memory, you can use it to update itself by following these steps:

1) After aborting the automatic boot sequence, assign an IP address to the EVM board using one of these methods:

   - If you are on a standalone network or using a network cross cable to your workstation, you can assign a static IP address to the EVM as follows:

     ```
     EVM # setenv ipaddr <static IP address>
     ```

- To assign a dynamic address, use the following:

```
EVM # dhcp
EVM # setenv ipaddr <IP address returned by dhcp>
```

2) Set the TFTP server IP address:

```
EVM # setenv serverip <TFTP server IP address>
```

3) Save these settings to the flash memory:

```
EVM # saveenv
```

4) Load U-Boot. To load U-Boot, the U-Boot image must be copied to the TFTP directory (usually /tftpboot), and the tftp command must specify both the image name and the destination address. In this case, the destination is in DDR memory starting at address 0x80700000, chosen arbitrarily in the DDR space.

```
EVM # tftp 0x80700000 <u-boot file name>
```

NAND Device	Total Size	Sector Size	U-Boot Load Address
Boot device U6 (SAMSUNG K9F1208R0B-JIB0)	512 MB	64 MB x 8	0x18000
Storage Device U67 (MICRON MT29F16G08FAAWC:A)	2 GB		not accessible from U-Boot

5) Erase the U-Boot location at the "U-Boot Load Address" shown in the previous table for your NAND device with a size that is larger than the "Bytes transferred" value from Step 4.

```
EVM # nand erase <U-Boot Load Address> 0x20000
```

6) Flash the new U-Boot from 0x80700000 to the U-Boot Load Address shown in the previous table for the corresponding NAND device.

```
EVM # nand write 80700000 <U-Boot Load Address> 20000
```

7) Power cycle the board or type "reset" to reboot. Verify that the recently flashed U-Boot is working by inspecting the build date on the serial terminal console output.

A.5.2 Updating UBL and U-Boot Bootloaders Using an Emulator and CCStudio

If working U-Boot (or UBL) images are not present in flash, you will need to restore these images using Code Composer Studio (CCStudio) and an emulator. Follow these steps:

1) Find the NAND programmer utilities. The NAND Programmer binary (NANDWriter.out) is in /home/<useracct>/dvsdk_#_##/PSP_#_#_#_#/bin/dm357.
   Alternatively, the source for the NAND programmer utilities can be extracted from your DVSDK installation at /home/<useracct>/dvsdk_#_##/PSP_#_#_#_#/board_utilities.
   Extract them onto a PC workstation that has CCStudio 3.3 (or higher) and an XDS560 or XDS510 emulator installed.
2) Configure CCStudio to connect to the DM357 EVM board using CCStudio Setup and the DM357 GEL files. The .ccs and .gel files are not included in the PSP package. You can download them from http://support.spectrumdigital.com/boards/evmdm357.
3) Connect an emulator to the EVM board's JTAG connector and power up the EVM board.
4) Open CCStudio and connect to the device (Alt+C).
5) Load the program NANDWriter.out and run it (F5).
6) Enter the full UBL path and file name for ubl_DM357_nand.bin in the dialog box.
7) Enter the full U-Boot path and file name for u-boot-1.2.0-dm357-nand.bin in the dialog box.
8) At the next two prompts, enter 0x82080000.
9) Wait until NAND programming is complete.
10) Cycle power on the EVM board and press any key on the EVM's monitor window to get the U-Boot prompt.

A.6 Restoring the NAND Flash

You can restore the contents of the DVSDK NAND flash memory on the EVM board via NFS (Section A.6.2) or via RAM Disk and an SD card (Section A.6.3). These contents include the Linux kernel and filesystem and the demo application software. However, you must first update the kernel on the NAND flash as described in Section A.6.1.

The DVSDK NAND image is included on the DVSDK (disk #2) restore directory (or the http://www.ti.com/dvevmupdates extranet) and is called "dm357_flash_image_#_#_#_#.tar", where #_#_#_# is the version.

A.6.1 Updating the Kernel

The Linux kernel (uImage) can be loaded to the NAND flash via TFTP. This step is required before you restore the NAND flash, whether you plan to do that via NFS (Section A.6.2) or via RAM Disk and an SD card (Section A.6.3).

To load the kernel, the file name of the kernel image that is in the server's tftp directory (usually /tftpboot) and the destination address need to be specified. Execute the following commands to download the kernel image and write to the NAND partition.

1) Copy the uImage-dm357 file to /tftpboot/uImage to rename the file as required by the following steps.

2) Assign an IP address to the EVM board using one of these methods:

   - If you are on a standalone network or using a network cross cable to your workstation, you can assign a static IP address to the EVM as follows:

     ```
     EVM # setenv ipaddr <static IP address>
     EVM # setenv serverip <tftp server IP address>
     EVM # tftp 0x80700000 uImage
     ```

   - To assign a dynamic address, use the following commands:

     ```
     EVM # setenv bootfile uImage
     EVM # setenv serverip <tftp server IP address>
     EVM # dhcp
     ```

3) Download the kernel image and write to the NAND flash as follows:

   ```
   EVM # nand erase 0x400000 0x200000
   EVM # nand write 0x80700000 0x400000 0x200000
   ```

A.6.2 Restoring the NAND Flash Using NFS

Once you have loaded the kernel binary to the corresponding NAND partition as described in Section A.6.1, you can use NFS to populate the YAFFS2 image (dm357_flash_image_#_#_#_#.tar) to the NAND partition. The YAFFS2 image should reside on the NFS Server root directory. Follow these steps:

1) Copy the dm357_flash_image_#_#_#_#.tar file from the DVSDK disk to the NFS mounted root directory. For example, /home/<*useracct*>/workdir/filesys.

2) Set the bootcmd environment variable to boot to kernel and mount to NFS. (Alternatively use the 'dhcp' command for the EVM IP Address.)

```
EVM # setenv bootcmd 'nboot 0x80700000 0 0x400000; bootm'
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd
  ip=dhcp root=/dev/nfs rw nfsroot=<nfs_host_ip>:<nfs_root_path> mem=232M
  video=davincifb:vid0=0,2025K:vid1=0,1350K:osd0=720x576x16,
  2025K davinci_enc_mgr.ch0_output=COMPOSITE
  davinci_enc_mgr.ch0_mode=ntsc
```

Note: These variables need not be saved, because NFS is just a temporary filesystem.

3) Execute the 'boot' command to boot the Linux kernel.

4) Login to the EVM as root and execute the following set of U-Boot commands to mount the NAND partition and populate the YAFFS2 image:

```
EVM # mkdir /mnt/nand
EVM # flash_eraseall /dev/mtd0
EVM # mount -t yaffs2 /dev/mtdblock0 /mnt/nand/
EVM # cd /mnt/nand
EVM # tar xf /dm357_flash_image_#_#_#_#.tar
EVM # cd
EVM # umount /mnt/nand
EVM # reboot
```

5) When the EVM comes up after rebooting from the previous steps, press Esc to get back to U-Boot prompt. You can now restore the out-of-the-box U-Boot environment variables as described in Section A.4.1.

A.6.3 Restoring the NAND Flash Using RAM Disk and a 2 GB SD Card

This procedure assumes the TFTP setup in Section A.3 has been performed.

Once you have loaded the kernel binary to the corresponding NAND partition as described in Section A.6.1, you can load the Ramdisk image (ramdisk.gz) to the DDR memory via TFTP. The YAFFS2 image resides on the 2 GB SD card. An MMC/SD card reader should be used to copy the YAFFS2 image on the 2 GB SD card. Follow these steps:

1) Find the RAM disk image at /home/<useracct>/dvsdk_#_#/PSP_#_#/bin and copy it to the host's /tftpboot directory.

2) After aborting the boot sequence, download the RAM disk image to the RAM as follows:

```
EVM # tftp 0x82000000 ramdisk.gz
```

3) Set the following environment variables to boot to Kernel and mount to the RAM disk.

```
EVM # setenv bootcmd 'nboot 0x80700000 0 0x400000; bootm'
EVM # setenv bootargs mem=232M console=ttyS0,115200n8
  root=/dev/ram0 rw initrd=0x82000000,4M ip=off
  video=davincifb:vid0=0,2025K:vid1=0,
  1350K:osd0=720x576x16,
  2025K davinci_enc_mgr.ch0_output=COMPOSITE
  davinci_enc_mgr.ch0_mode=ntsc
```

Note: These variables need not be saved to the NAND flash, as the RAM disk is just a temporary filesystem.

4) Plug the 2GB SD Card into the MMC/SD slot on the DM357 EVM.

Note: If the card is not plugged in when the kernel boots the kernel will lock-up when the card is hot-plugged later.

5) Execute the following command to boot to Kernel:

```
EVM # boot
```

6) Login to the EVM using the root username. Note that "root" does not have a password.

7) Execute the following commands to mount the MMC/SD and NAND partitions and populate the YAFFS2 image:

```
EVM # mkdir /mnt/mmc
EVM # mkdir /mnt/nand
EVM # mount -t vfat /dev/mmcblk0 /mnt/mmc/
EVM # flash_eraseall /dev/mtd0
EVM # mount -t yaffs2 /dev/mtdblock0 /mnt/nand/
EVM # cd /mnt/nand
EVM # tar xf /mnt/mmc/dm357_flash_image_#_#_#_#.tar
EVM # cd
EVM # umount /mnt/nand
EVM # reboot
```

The first mount command assumes you have a VFAT partition. The tar command will take about a minute to run.

The filename of the dm357_flash_image_#_#_#_#.tar image will be in DOS 8.3 format if you are using a vfat filesystem. That is, dm357_fl.tar.

8) When the EVM comes up after rebooting from the previous steps, press Esc to get back to U-Boot prompt. You can now restore the out-of-the-box U-Boot environment variables as described in Section A.4.1.

Index

A

application 4-4
arrow buttons 3-5
audio 2-3

B

battery 1-3, 3-4
bin files 4-6
block diagram 1-3
boot configurations A-6
  flash with NAND flash A-6
  flash with NFS A-7
  NFS 4-10
  standard 3-2
  TFTP with NAND flash A-7
  TFTP with NFS A-8
boot sequence A-6
bootloader A-9
build environment 4-12

C

cables
  connecting 2-2
CDs 4-2
clock battery 1-3
Code Search button 3-4
Codec Engine 3-6, 3-7, 4-4
COM port 2-6
command line demos 3-10
command prompts 4-3
composite video-in (J12) 2-4
composite video-out (J8) 2-4
console window 2-6
  UART port 2-5
contents of kit 1-2

D

data files 4-8
DaVinci technology
  community 1-4
Decode demo 3-5, 3-9
  command line 3-10
demos 3-2
  command line 3-10
Digital Video Test Bench (DVTB)
  building 4-17
  documentation 4-17
disks 1-2
  file contents 4-2
DISPLAY environment variable 4-5
DSPLink 4-15
DVD button 3-4
DVDs 4-2
DVDs, mounting 4-5
DVEVM
  installing software 4-7
DVEVM software
  rebuilding 4-14
DVSPB 4-3

E

electrostatic precautions 2-2
Encode + Decode demo 3-5, 3-7
  command line 3-10
Encode demo 3-5, 3-7
  command line 3-10
environment variables
  XDC_INSTALL_DIR 4-15
Ethernet port (P2) 2-5
EVM # prompt 2-6, 4-3
examples 3-2
exit demo 3-6
exports file 4-9

F

file extensions 3-8
file system 4-8
files
  Decode demo 3-9

Encode demo 3-7
on disks 4-2
flash memory
boot configuration A-6, A-7

G

G.711 speech 3-8, 3-9
GNU make, version 4-15

H

H.264 video 3-7, 3-8, 3-9
host $ prompt 4-3

I

Info/Select button 3-6
installing
DVEVM software 4-7
hardware 2-2
Linux software 4-6
IR remote 1-2, 3-4
resetting code 3-4

K

kit contents 1-2

L

Linux 4-4
installing 4-6
kernel 4-13
versions supported 4-6
Linux Support Package 4-13

M

make utility, version 4-15
MontaVista Linux
demo version 4-3
full version 4-3
MPEG4 video 3-8, 3-9
multimedia peripherals 1-3

N

NAND flash A-9
boot configuration A-6, A-7
restoring A-12
NAND programmer utilities A-11
NFS server 4-8
boot configuration A-7, A-8
testing 4-10
NTSC video 2-3

O

OSD show and hide 3-6
OSD toggle 3-6

P

PAL video 2-3
PATH environment variable 4-12
Pause button 3-6
peripherals 1-3, 2-2
Play button 3-5
ports 2-5
Power button 3-6
power cable 2-5
power jack 2-5
power supply 1-2
power switch 2-5
prompts 4-3

Q

quit demo 3-6

R

RCA connectors 2-4
rebuilding
DVEVM software 4-14
Linux kernel 4-13
Record button 3-5
Red Hat Enterprise Linux 4-6
remote control 1-2, 3-4
resetting code 3-4
RS-232 2-5
Rules.make file 4-14
running applications 3-5

S

S3 mini-dip switch 2-3
serial cable 2-6
software 4-2

components 1-2, 4-4
installing 4-6
speaker 2-3
Spectrum Digital website 1-3
standalone demos 3-2
static precautions 2-2
Stereo Line In (P3) 2-3
Stereo Line Out (P5) 2-3
Stop button 3-6
SuSe Workstation 4-6

T

target $ prompt 4-3
terminal session 2-6
test program 4-12
TFTP
boot configuration A-7, A-8
server A-5
transfer files to board 4-16
Third-Party Menu A-3
transparency of OSD 3-6

U

UART port (P6) 2-5
UBL A-9
U-Boot A-9
U-Boot utility 4-13
uImage boot file 4-16

V

video-in (J12) 2-4
video-out (J8) 2-4
VISA APIs 4-4

X

XDC_INSTALL_DIR environment variable 4-15

Y

YAFFS2 image A-13

Spectrum Digital, Inc.
511458-0001B