TMS320DM6446 DVEVM v1.30
Getting Started Guide

Literature Number: SPRUE66D
April 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 TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

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

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

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

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

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
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
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
Low Power Wireless	www.ti.com/lpw	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) kit is an evaluation platform that showcases the DM644x architecture and lets users evaluate the power and performance of the DM644x 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 release notes 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:

- Spectrum Digital website: http://c6000.spectrumdigital.com/davincievm/
- TI Linux Community for DaVinci Processors: http://linux.davincidsp.com
- TI DaVinci Software Updates: http://www.ti.com/dvevmupdates
- TI DaVinci Technology Developers Wiki: http://wiki.davincidsp.com
- *Codec Engine Application Developer's Guide* (SPRUE67)
- Other PDF documents on the CDs included with the DVEVM kit
- Section 4.11 lists documentation in the DVSDK software installation.
- SoC Analyzer Help menu

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.

April 15, 2008

Contents

Chapter 1

DVEVM Overview

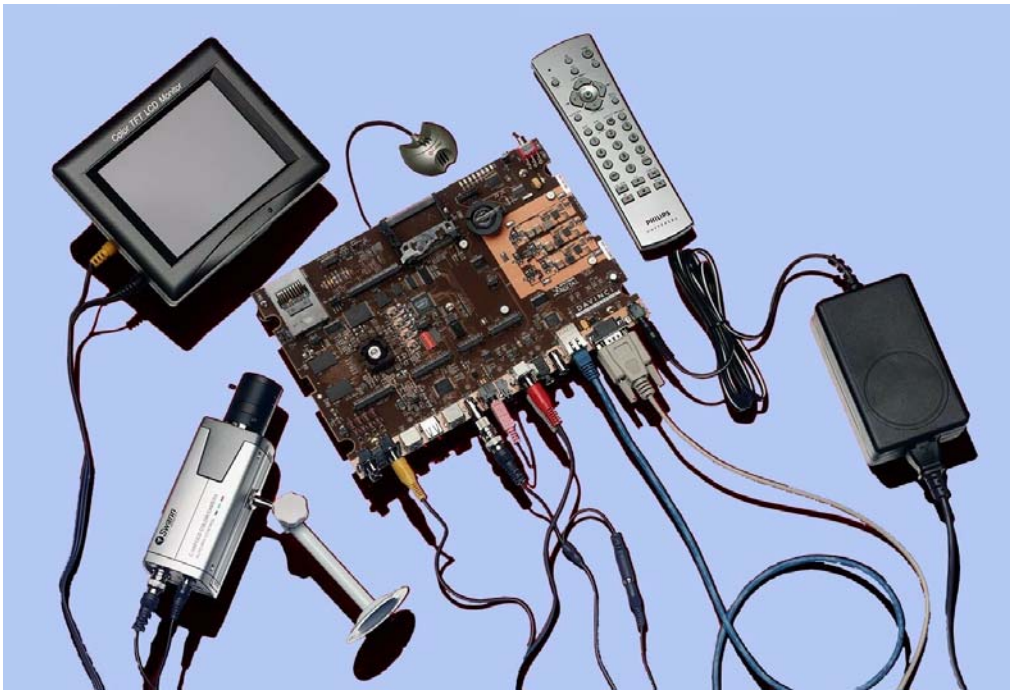
This chapter introduces the DVEVM (Digital Video Evaluation Module) kit.

Topic **Page**

1.1 Welcome!

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

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



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.

- **EVM Board**. 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**. The hard drive provided with the EVM 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 has been preloaded on this EVM board's hard disk drive.
- **CCD Camera**. This camera provides NTSC or PAL video imaging for DaVinci applications.
- **LCD Display**. The 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 has a 110 VAC power supply. The PAL version has a 220 VAC power supply.
- **PC Desktop Microphone**. The microphone provides a way to capture audio for use by DaVinci applications.
- **IR Remote Control**. This universal remote control is included to provide a user interface to the demo applications.
- **Cables.** Cables used to connect the EVM 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 v4.0.1 Target
- TI DaVinci Demonstration Version of MontaVista Linux Pro v4.0.1 Tools
- A/V Media Clips
- Spectrum Digital EVM Tools
- SoC Analyzer

1.3 What's on the Board?

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

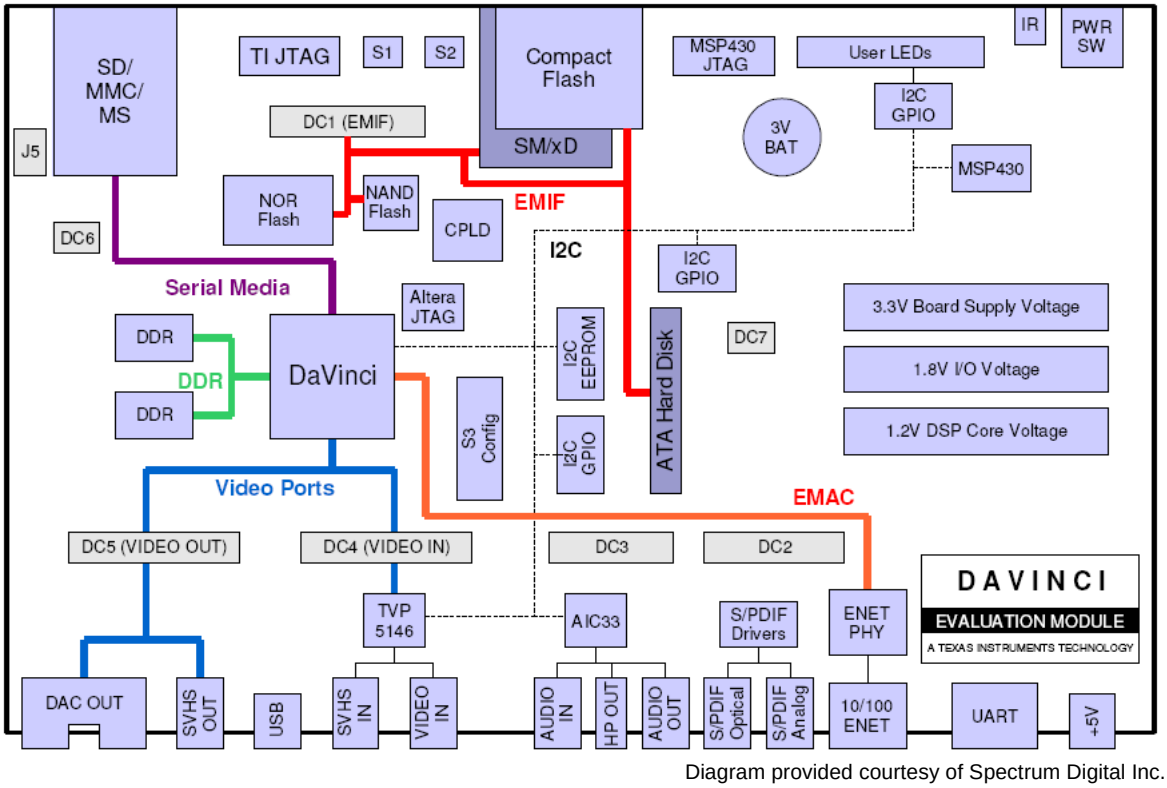


Diagram provided courtesy of Spectrum Digital Inc.

Figure 1–1 EVM Hardware Block Diagram

For more information about the EVM 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 kit 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:

- Spectrum Digital website: http://c6000.spectrumdigital.com/davincievm/
- TI Linux Community for DaVinci Processors: http://linux.davincidsp.com
- TI DaVinci Software Updates: http://www.ti.com/dvevmupdates
- TI DaVinci Technology Developers Wiki: http://wiki.davincidsp.com
- *Codec Engine Application Developer's Guide* (SPRUE67)
- Other PDF documents on the CDs included with the DVEVM kit
- Section 4.11 lists documentation in the DVSDK software installation.
- SoC Analyzer Help menu

Chapter 2

EVM Hardware Setup

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

Topic **Page**

2.1 Setting Up the Hardware

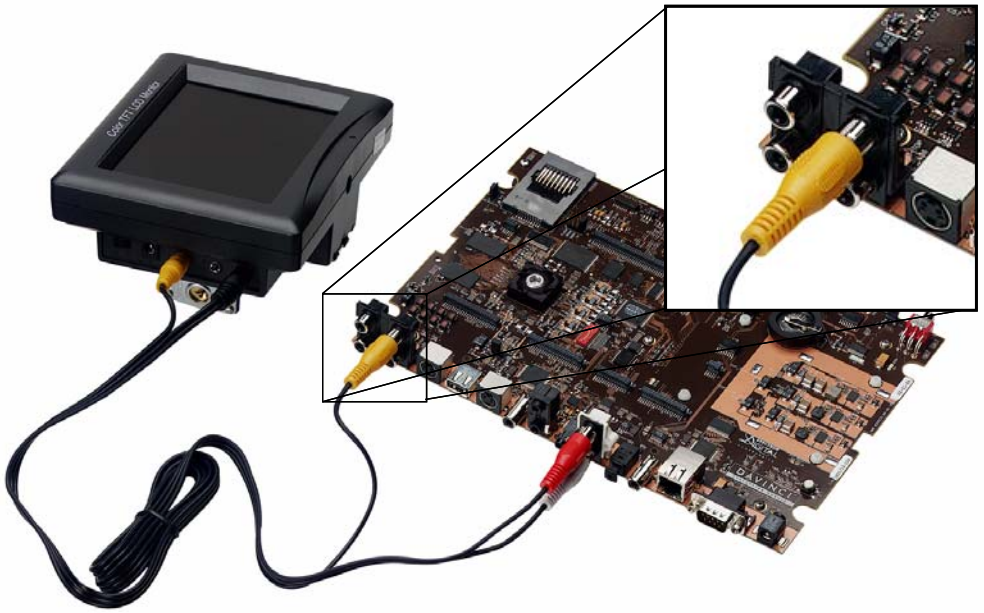
To set up the hardware provided with the EVM, use the steps in the sections that follow. You may skip sections if you do not need to access a particular peripheral. For example, if you do not need to use the serial cable, skip that section.

1) The EVM 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.

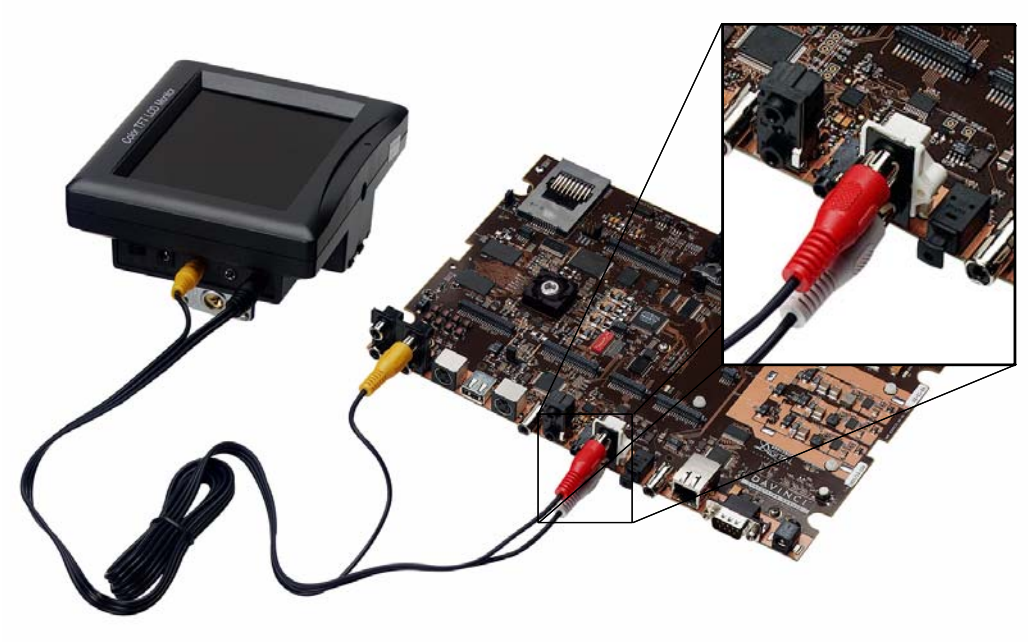
2) If you use PAL video, set switch 10 on the S3 (USER) bank of switches to On. If you use NTSC video, set this switch to Off. See Figure 1–1 for S3 switch bank location.

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



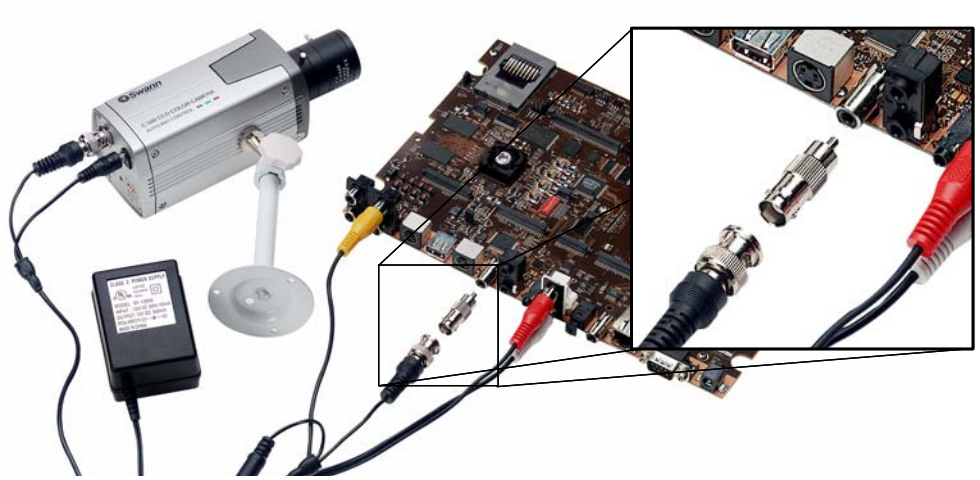
See Section A.1, *Changing the Video Input/Output Methods* for information about using S-Video or Component video.

4) Connect the red and white audio cables to the EVM 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 EVM Video Input.

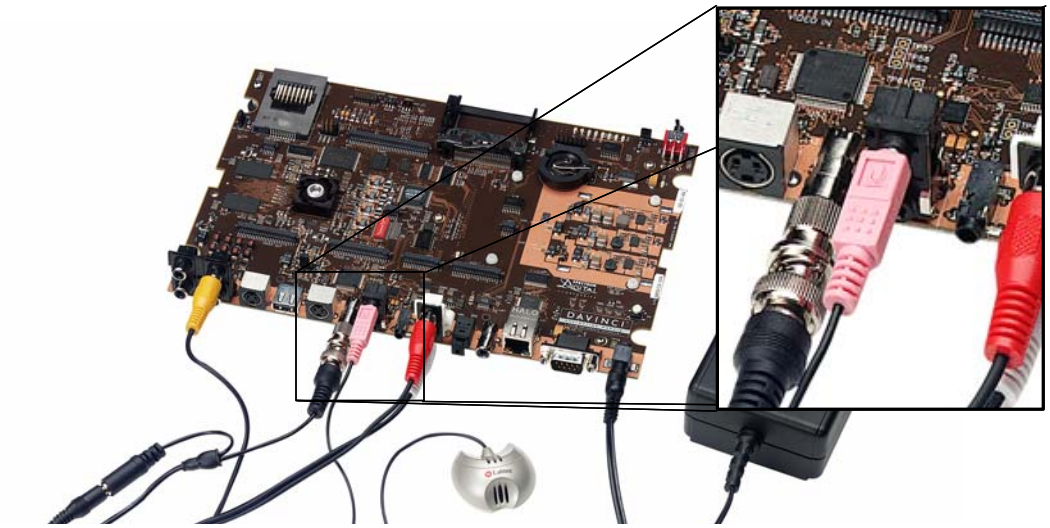
6) Connect the power jack for the video camera. To be ESD safe, do not plug in the other end of the camera power cord until the later step that instructs you to do so.



See Section A.1, *Changing the Video Input/Output Methods* for information about using S-Video or Component video.

7) Connect the microphone to the EVM.

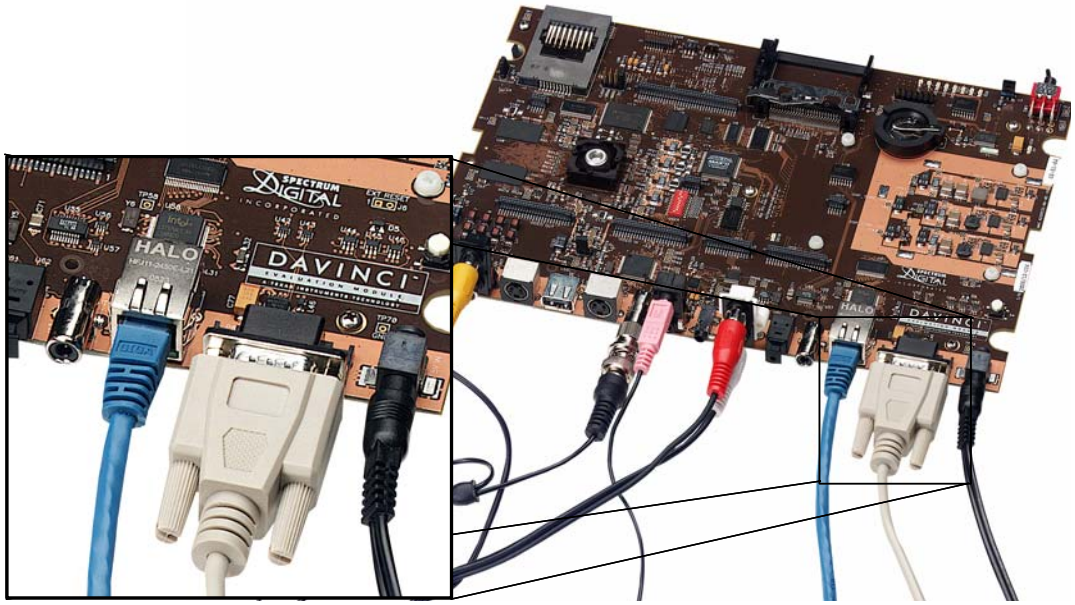
8) Connect the power cable to the EVM power jack on the board. To be ESD safe, do not plug in the other end of the cable yet.



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

   Note that the U-Boot bootargs must include "ip=dhcp" to enable the network connection.

10) If you plan to use the UART port for a console window, connect the RS-232 null modem cable to the EVM UART port and 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) Connect the power cable to the EVM power jack on the board. To be ESD safe, plug in the other end of the power cable only after you have connected the power cord to the board.

14) Power on the LCD display.

15) Power on the EVM 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.

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 HyperTerminal session on the PC 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

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 following are alternate ways you may want to boot the board:

- ❏ TFTP boot with hard drive 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 4.3.7)

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.

3.2 Starting the Standalone Demos

When you connect the EVM 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. This mode is 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 EVM'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, the LCD 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 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 (for older remotes shipped with the kit, the code is "020").

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 key for at least a minute, then reinserting the batteries. Then program the remote as in Step 3.

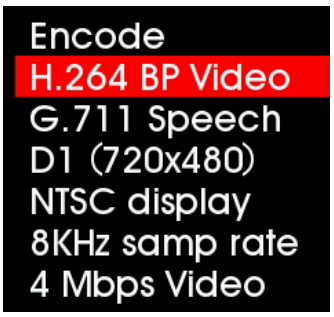
3.3 Running the Standalone Demos

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.2, *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.

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. 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.

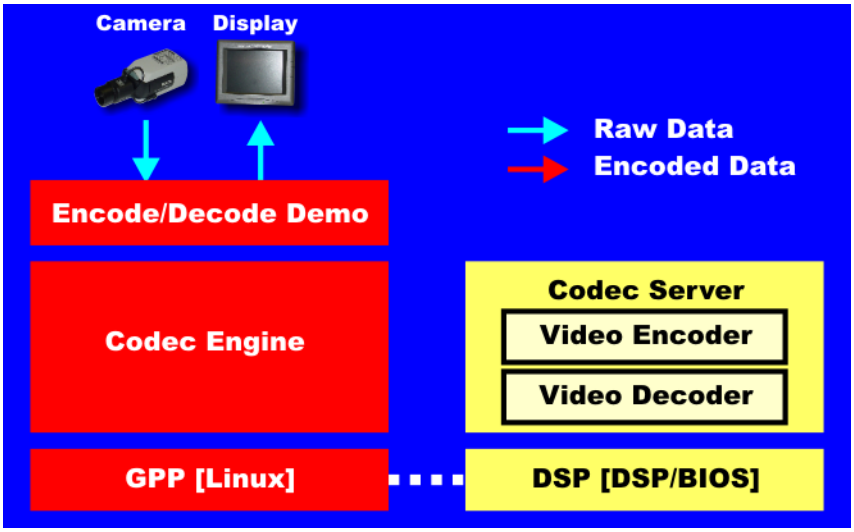
9) Press "Stop" or "Pause" when you want to end or pause a demo. Press "Stop" from the settings screen, you go back to the main menu.

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 comes from the camera, is encoded, then decoded, and then sent to the LDC 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 --
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.

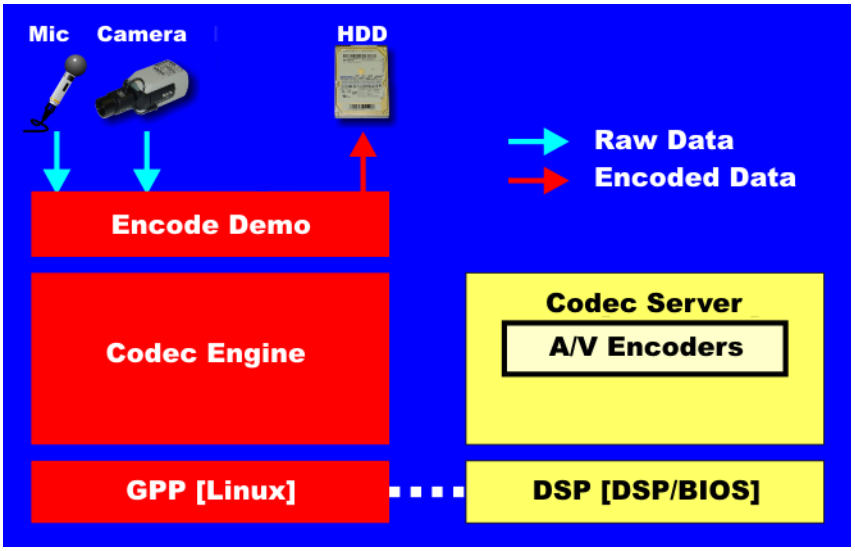
To use this demo from the command line, see Section 3.4, *Running the Demos from the Command Line*.

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 EVM'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 / Run	Begin encode demo, send unencoded data to display

Table 3–2 IR Remote Buttons for Encode Demo

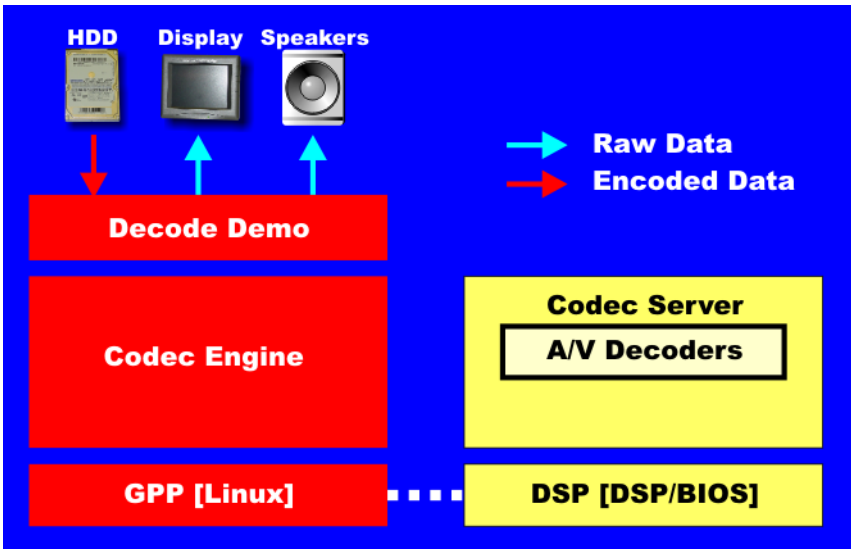
IR Remote Button	Mode	Action Performed
Info/Select	Setup	Show / hide block diagram for demo
Info/Select	Run	Toggle information display
Left/Right	Run	Change information transparency level (There is no display for encode demo behind the information.)
Pause	Run	Pause demo (press Record to resume)
Stop	Setup / Run	Return to previous screen

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

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

3.3.4 About the Third Party Menu

The Third-Party Menu can be used to add additional demos. See Section A.2, *Putting Demo Applications in the Third-Party Menu*.

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/dm6446
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.

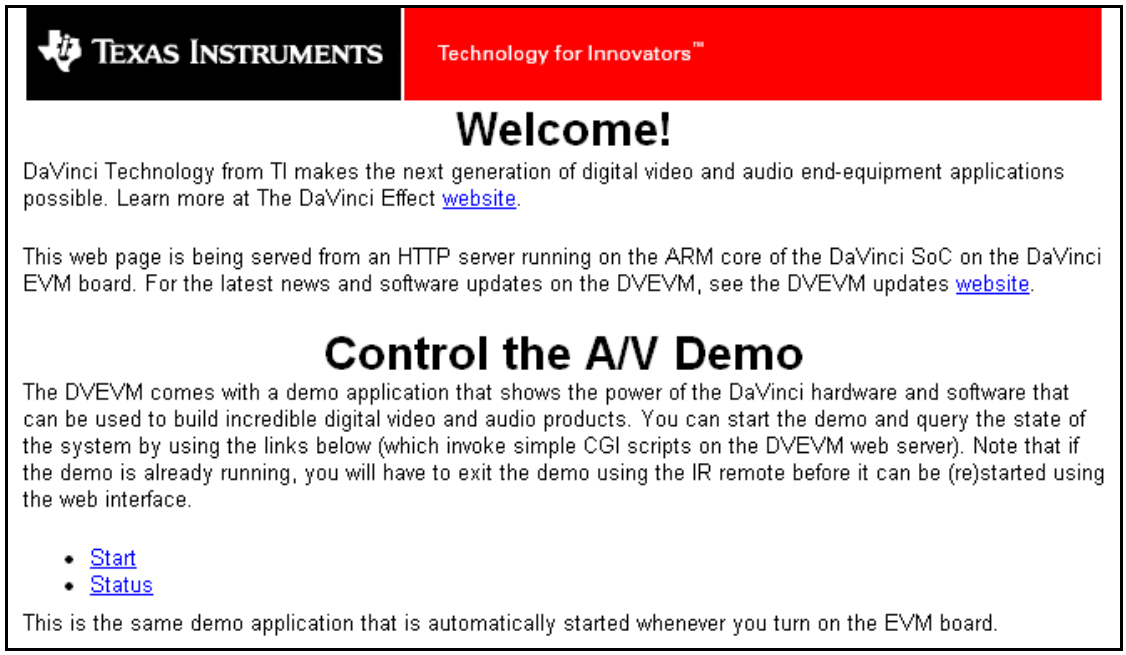
Note: When you run loadmodules.sh, you may see warnings about the kernel being tainted by cmemk and dsplinkk. You can ignore these warnings. They occur because DSP/BIOS Link does not use the kernel build system and is not under the same license as the kernel itself.

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 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 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/dm6446/web directory.

Chapter 4

DVEVM Software Setup

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

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 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.

- **Standalone demonstration software.** This is provided on the hard drive on the EVM. The hard-wired examples encode and decode audio, video, and speech. Another demo shows the EVM's network capabilities. See Section 3.2, *Starting the Standalone Demos*.
- **CD 1: MontaVista Linux Pro v4.0.1 System Tools.** The version provided with the DVEVM kit is the preliminary demonstration version. It contains the following file:
  - mvl_4_0_1_demo_sys_setuplinux.bin. Installation file for the MontaVista Tool development tool chain.
- **CD 2: MontaVista Linux Pro v4.0.1 Target File System.** The DVEVM kit provides a preliminary demonstration version. It contains the file:
  - mvl_4_0_1_demo_target_setuplinux.bin. Installation file for the MontaVista target file system.
- **CD 3: TI DVSDK Software.** This CD includes demo applications, Codec Engine software, example codec servers, and DVSDK documentation. It contains the following files:
  - sprue66#.pdf (this Getting Started Guide; # is a letter indicating the version)
  - dvsdk_setuplinux_#_#_#_#.bin. DVSDK installer.
  - mvl_4_0_1_demo_lsp_setuplinux_#_#_#_#.bin.
  - xdc_setuplinux_#_#_#_#.bin. XDC Tools installer.
  - restore directory (Contains files used for hard drive recovery. See Section A.6 for details.)
- **CD 4: A/V Data.** It contains sample A/V data in the data.tar.gz file.
- **CD 5: SDI Board Support Software.** It contains ARM-based test code, CCStudio GEL files, and other EVM support software.
- **CD 6: SoC Analyzer 1.2.** The SoC Analyzer is a graphical tool that runs on a Windows development host and uses data collected from

Linux, DSP/BIOS, and Codec Engine to provide system-level execution and performance analysis for debugging and profiling DVEVM software execution.

Texas Instruments, in agreement with MontaVista Software Inc., is providing a demonstration version of the Linux Professional Edition v4.0.1 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™ 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.

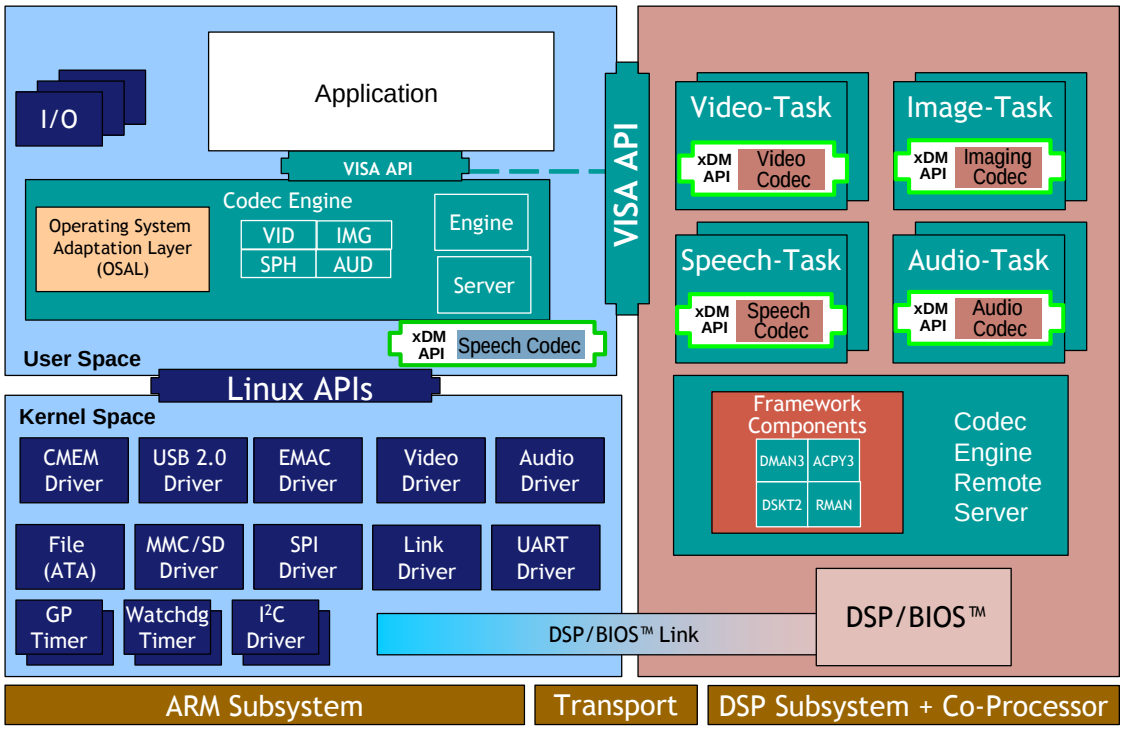
4.1.1 Command Prompts in This Guide

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 on the EVM:



In the previous figure, your application runs on the ARM subsystem. 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 subsystem. The DSP handles signal processing and the results are available to the ARM subsystem in shared memory. 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 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_4_0_1_demo_sys_setuplinux.bin
- ❏ mvl_4_0_1_demo_target_setuplinux.bin
- ❏ mvl_4_0_1_demo_lsp_setuplinux_#_#_#_#.bin
- ❏ dvsdk_setuplinux_#_#_#_#.bin
- ❏ xdc_setuplinux_#_#_#_#.bin
- ❏ bios_setuplinux_#_#_#_#.bin
- ❏ TI-C6x-CGT-v#.#.#.#.bin

Updates to these installers may be available on the TI DaVinci Software Updates website listed in Section 1.4.

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 kit 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, *Installing the SoC Analyzer*
- ❏ Section 4.3.5, *Exporting a Shared File System for Target Access*
- ❏ Section 4.3.6, *Testing the Shared File System*
- ❏ Section 4.3.7, *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 v4.0.1.

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

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 directory, we suggest that you install in the /opt/mv_pro_4.0.1 directory.

   ```
   host $ ./mvl_4_0_1_demo_sys_setuplinux.bin
   host $ ./mvl_4_0_1_demo_target_setuplinux.bin
   host $ ./mvl_4_0_1_demo_lsp_setuplinux_#_#_#_#.bin
   ```

3) After you execute these .bin files, make sure the following files are located in /opt/mv_pro_4.0.1 (or in the /mv_pro_4.0.1 subdirectory of the directory you chose in place of the default):

   - mvltools4.0.1-no-target.tar.gz
   - mvl4.0.1-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_4.0.1
   ```

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

   ```
   host $ tar zxf mvltools4.0.1-no-target.tar.gz
   host $ tar zxf mvl4.0.1-target_path.tar.gz
   host $ tar zxf DaVinciLSP-#_#_#_#.tar.gz
   ```

   This creates the MontaVista directory structure under the /opt/mv_pro_4.0.1/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; it 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, DSP/BIOS Link, sample data files, xDAIS and xDM header files, and a contiguous memory allocator for Linux (CMEM).

Note: The installers for DSP/BIOS and Code Generation Tools (codegen) have a different default installation location. However, we strongly recommend that you change the default installation locations to place the components together (if you have not already installed the Linux versions of these components elsewhere). This simplifies the build setup steps.

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

1) Log in using a **user account**.

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

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

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

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

```
host $ ./xdc_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) Execute the DSP/BIOS installer. For example:

```
host $ ./bios_setuplinux_5_#_#_#.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_#_#.

5) Execute the Code Generation Tools installer. For example:

```
host $ ./TI-C6x-CGT-v#.#.#.#.bin
```

When the installer prompts for an installation location, *do not* use the default location. Instead, use the **entire** path to the dvsdk_#_# codegen directory. You will need to manually create the folder "cg6x_6_0_15". For example:

/home/<*useracct*>/dvsdk_#_#/cg6x_6_0_15.

Remember to set the environment variable as directed by the installer. For example:

```
C6X_C_DIR="/home/<useracct>/dvsdk_#_#/cg6x_6_0_15/include;
/home/<useracct>/dvsdk_#_#/cg6x_6_0_15/lib"
```

6) 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 directories created by InstallShield.

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

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

2) Mount the A/V data CD and copy the 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 Installing the SoC Analyzer

SoC Analyzer is a graphical tool that runs on a Windows development host and uses data collected from Linux, DSP/BIOS, and Codec Engine to provide system-level execution and performance analysis for debugging and profiling DVEVM software execution. Follow these instructions to install SoC Analyzer:

1) Insert the SoC Analyzer CD into the Windows development host PC.
2) The installer should start automatically. If the installer does not start after 30 seconds, browse to the CD and double-click the dvt.exe file.
3) Follow the installer's prompts to install the software.

4.3.5 Exporting a Shared File System for Target Access

Although the EVM'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 EVM'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.
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_4.0.1, use your location in the cp command.

```
host $ cp -a /opt/mv_pro_4.0.1/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)
```

Note: Make sure you do not add a space between the * and (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
```

4.3.6 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.

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 video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
   console=ttyS0,115200n8 noinitrd rw ip=dhcp
   root=/dev/nfs nfsroot=$(nfshost):$(rootpath),nolock
   mem=120M
```

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 the exports. 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 or NFS and using flash or the EVM's hard drive.

4.3.7 Configuring the Boot Setup for PAL Video Users

You can configure the EVM 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. For example:

```
EVM # setenv bootargs video=davincifb:vid0=720x576x16,
    2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
    davinci_enc_mngr.ch0_output=COMPOSITE
    console=ttyS0,115200n8 noinitrd rw ip=dhcp
    root=/dev/hda1 mem=120M

EVM # setenv bootcmd 'setenv setboot setenv bootargs
   \$(bootargs) davinci_enc_mngr.ch0_mode=\$(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.

See Section A.1, *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):

```
PATH="/opt/mv_pro_4.0.1/montavista/pro/devkit/arm/v5t_le/bin:
     /opt/mv_pro_4.0.1/montavista/pro/bin:
      /opt/mv_pro_4.0.1/montavista/common/bin:$PATH"
```

   If you installed in a location other than /opt/mv_pro_4.0.1, 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 4.3.5, *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 2.6.10 kernel plus the DaVinci drivers. If you installed in a location other than /opt/mv_pro_4.0.1, 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_4.0.1/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-2.6.10_mvl401
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- davinci_dm644x_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) If you want to enable Linux Trace for the SoC Analyzer, follow these substeps. Otherwise, skip to Step 7.

   a) Use this command to go to the configuration menu for the ARM:

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

   b) Navigate to Device Drivers->Filesystems->Pseudo Filesystems.

   c) Set **Relayfs** file system support to "built-in".

   d) Return to the main menu and navigate to **General Setup**.

   e) Enable **Linux Trace Toolkit Support** as "built-in".

   f) Select Exit to save your changes and exit the configuration screen.

7) Compile the kernel using the following command:

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

8) 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
```

9) 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. (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-2.6.10_mvl401/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 DVSDK Software for the Target

To place demo files in the /opt/dvsdk/dm6446 directory, you need to rebuild the DVSDK software. To do this, follow these steps:

1) Change directory to dvsdk_#_#.

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

- Set PLATFORM to match your EVM board as follows:

```
PLATFORM=dm6446
```

- Set DVSDK_INSTALL_DIR to the top-level DVSDK 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/dm6446
```

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

```
MVTOOL_DIR=/opt/mv_pro_4.0.1/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-2.6.10_mvl401
```

- If necessary, modify the following environment variables to match the locations of these components on your Linux host. We recommend that these components be in the /home/<*useracct*>/dvsdk_#_# directory, but you may have installed them elsewhere.

  BIOS_INSTALL_DIR
  FC_INSTALL_DIR
  XDC_INSTALL_DIR

3) 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
```

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 4.5, *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 video=davincifb:vid0=720x576x16,
2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
davinci_enc_mngr.ch0_output=COMPOSITE
davinci_enc_mngr.ch0_mode=$(videostd)
console=ttyS0,115200n8 noinitrd rw ip=dhcp
root=/dev/hda1 mem=120M
```

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

3) Boot the board:

```
EVM # bootm
```

This configuration boots a new Linux kernel via TFTP with a hard drive based file system. For details on booting via TFTP 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, *Setting Up a TFTP Server*.

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

4.8 Testing the Build Environment

To test your DVSDK software installation, you can build one of the Codec Engine servers. This server is a DSP-side application. Building it tests the installation of DSP-side development components.

To build the video_copy server, follow these steps:

1) Go to /home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/examples and open the build_instructions.html file.
2) Follow the step-by-step instructions for building examples. When you are editing the xdcpaths.mak file, note that the DVSDK installation *does not* include the cetools directory, so you will need to set the USE_CETOOLS_IF_EXISTS variable to 0, and modify additional variables to point to the locations of xDAIS, DSP/BIOS Link, CMEM, and Framework Components.
3) Go to the video_copy server directory. (/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/examples/servers/video_copy) to inspect the built server.

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. For example, 1_23_000).

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=dm6446
host $ make CONFIGPKG=dm6446
```

3) Copy the binaries "dvtb-d" and "dvtb-r" to /opt/dvsdk/dm6446 on the device's target filesystem and run it there. It must be in the same directory as the DSP executables.

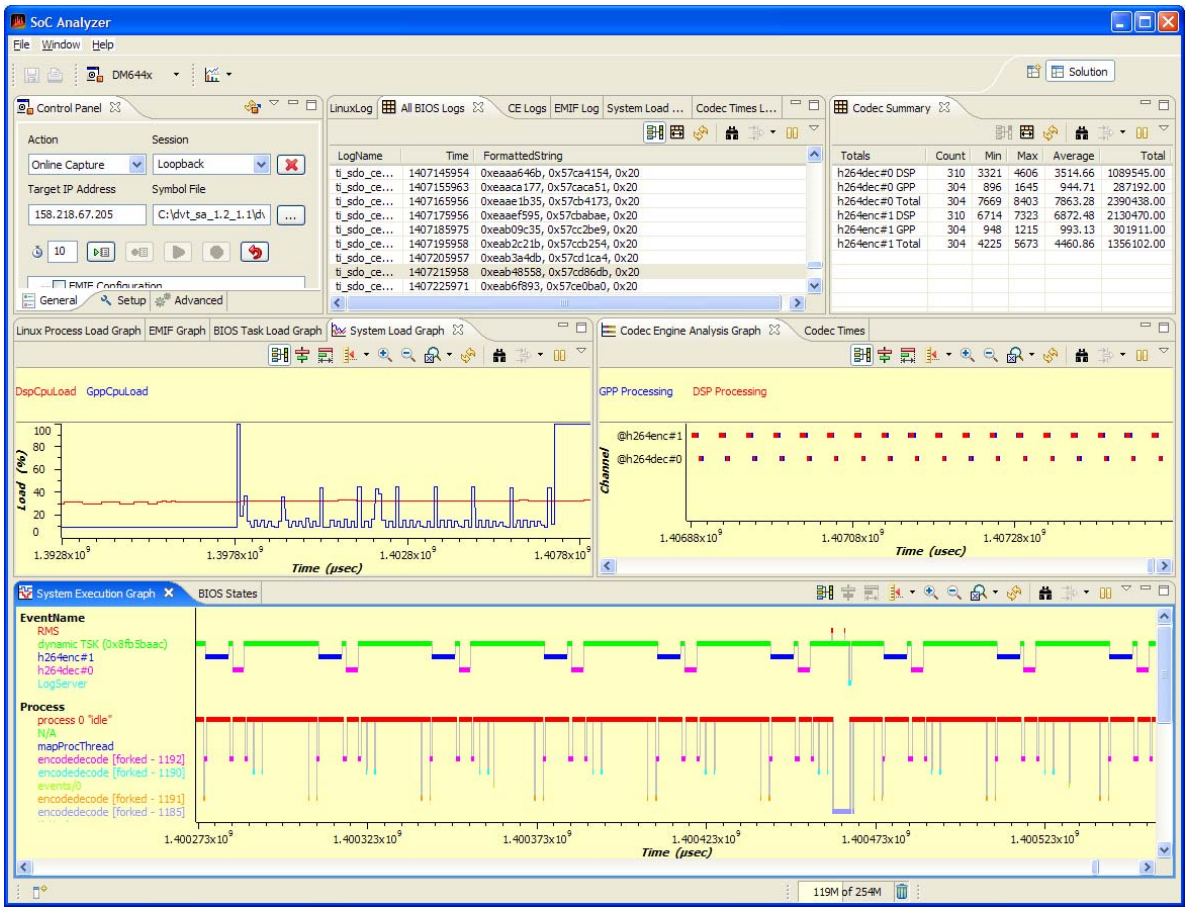
For further details on the DVTB, see the following documents:

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

4.10 Running The SoC Analyzer

Built upon Texas Instruments' eXpressDSP data visualization technology (DVT), the SoC Analyzer simplifies debugging, analysis, and optimization of DVEVM applications. It collects execution, interaction, and resource utilization logs from Linux, DSP/BIOS, Codec Engine, and drivers and presents system-level analysis and graphical visualization such as:

- ❏ System execution flow
- ❏ System performance
- ❏ Resource utilization
- ❏ Processor interaction



Install the SoC Analyzer as described in Section 4.3.4.

To run the SoC Analyzer, double-click the SoC Analyzer icon on the Windows Desktop or select it from the Windows Start menu under Texas Instruments.

The SoC Analyzer comes with online help, which can be accessed from the SoC Analyzer Help menu (choose **Help->Help Contents**). Select the *DM644x SoC Analyzer User Guide*.

Follow the "Getting Started" chapter of the *DM644x SoC Analyzer User Guide*, which includes steps on how to run the tool with demonstration logs shipped with the product. This "Getting Started" chapter also contains the steps you should perform to enable your applications to collect logs for the SoC Analyzer. The DVEVM Demo Application has such logging enabled by default. The uImage must be rebuilt to enable logging.

4.11 Documentation for DSP-Side Development

After you have installed the DVSDK software, you can begin to create and modify DSP-side applications for your DM644x.

The following table lists places to look for documentation on using each component of the DVSDK. Documents in PDF, HTML, and text format are included in the installations with each product.

Table 4-1. Documentation for DVSDK Components

Component	Title	Location
DSP/BIOS	*TMS320C6000 DSP/BIOS API Reference* (SPRU403)	/home/<*useracct*>/dvsdk_#_#/bios_5_#/packages/ti/bios/doc
	Application Notes	www.dspvillage.com
Code Generation Tools	*TMS320C6000 Optimizing C Compiler User's Guide* (SPRU187)	www.dspvillage.com
	TMS320C6000 Programmer's Guide (SPRU189)	www.dspvillage.com
Framework Components	Release Notes	/home/<*useracct*>/dvsdk_#_#/framework_components_#_#
Digital Video Test Bench	README.txt	/home/<*useracct*>/dvsdk_#_#/dvtb/docs

Table 4-1. Documentation for DVSDK Components

Component	Title	Location
	Section 4.9, *Using the Digital Video Test Bench (DVTB)*	this document
Codec Engine	*Codec Engine Application Developer User's Guide* (SPRUE67)	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/docs
	Codec Engine Server Integrator User's Guide (SPRUED5)	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/docs
	Codec Engine Algorithm Creator User's Guide (SPRUED6)	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/docs
	Example Build and Run Instructions	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/examples/build_instructions.html
	Codec Engine API Reference	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/docs/html/index.html
	Codec Engine SPI Reference Guide	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/docs/spi/html/index.html
	Configuration Reference	/home/<*useracct*>/dvsdk_#_#/codec_engine_#_#/xdoc/index.html
XDC Tools (used by Codec Engine)	Documentation Links	/home/<*useracct*>/dvsdk_#_#/xdctools_#_#/docs/index.html
xDAIS	*xDAIS-DM (Digital Media) User Guide* (SPRUEC8)	/home/<*useracct*>/dvsdk_#_#/xdais_5_00/packages/ti/xdais/dm/docs

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

The EVM can input video using the following methods:

- ❏ Composite [default]
- ❏ S-Video (best quality)

In addition, there are three types of video output:

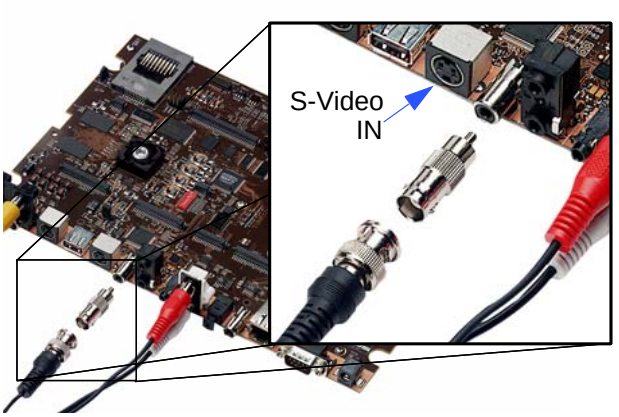
- ❏ Composite [default] (lowest quality)
- ❏ S-Video (medium quality)
- ❏ Component (best quality)

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.1.1 Using S-Video Input

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

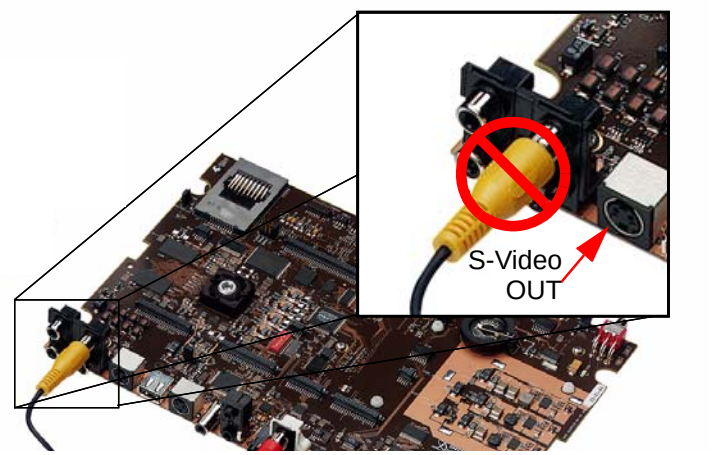
1) Connect your S-Video connector to the S-Video input port, which is directly to the left of the currently-used composite video input port.
2) Select S-Video input on the command line when you execute the encode or encodedecode demo using the '-x' flag.



A.1.2 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 EVM to select both NTSC vs. PAL and the S-Video output format (see Section 4.3.7, *Configuring the Boot Setup for PAL Video Users*). For example, if you want both NTSC and S-Video output, use the following:

```
davinci_enc_mngr.ch0_output=SVIDEO davinci_enc_mngr.ch0_mode=NTSC
```

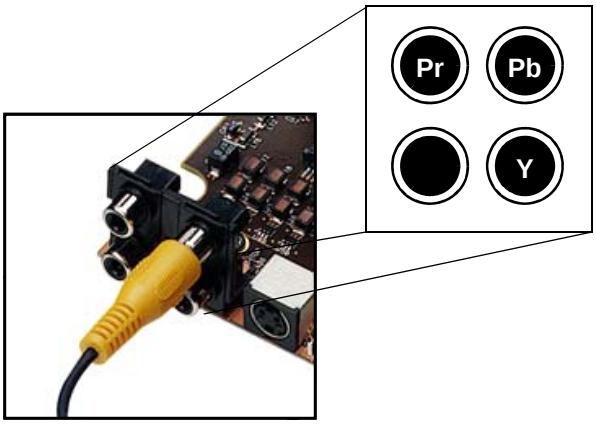
If you want both PAL and S-Video, use the following:

```
davinci_enc_mngr.ch0_output=SVIDEO davinci_enc_mngr.ch0_mode=PAL
```

A.1.3 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. The cable you use may have red, green, and blue connectors, but you should be aware that the Y, Pr, and Pb signals used by component video do not correspond to RGB signals. In addition, you will need a video display with component video inputs.

2) On the kernel command line, you can configure the EVM to select both NTSC vs. PAL and the component video output format (see Section 4.3.7, *Configuring the Boot Setup for PAL Video Users*). For example, if you want both NTSC and component video output, use the following:

```
davinci_enc_mngr.ch0_output=COMPONENT davinci_enc_mngr.ch0_mode=NTSC
```

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

```
davinci_enc_mngr.ch0_output=COMPONENT davinci_enc_mngr.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 DVSDK software was installed on the target file system. The default target installation directory is /opt/dvsdk/dm6446, so the default location for demo archives is /opt/dvsdk/dm6446/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 EVM's hard drive. The following are alternate ways you may want to boot the board:

- ❏ TFTP boot with hard drive 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)

Section 4.3.7 discusses booting in PAL mode vs. NTSC 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 EVM 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.4.1 Booting from Flash Using the EVM'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 video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
   console=ttyS0,115200n8 noinitrd rw ip=dhcp
   root=/dev/hda1 mem=120M
```

Note: If you have flashed a new kernel to the NOR Flash at an address other than 0x2050000, modify the bootcmd definition accordingly.

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

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

A.4.2 Booting via TFTP Using the EVM'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 video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
   console=ttyS0,115200n8 noinitrd rw ip=dhcp
   root=/dev/hda1 mem=120M
EVM # setenv serverip <tftp server ip address>
EVM # setenv bootfile <path on tftpserver>/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.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 0x2050000
EVM # setenv nfshost <ip address of nfs host>
EVM # setenv rootpath <directory to mount*>
EVM # setenv bootargs video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
   console=ttyS0,115200n8 noinitrd rw ip=dhcp root=/dev/nfs
   nfsroot=$(nfshost):$(rootpath),nolock mem=120M
```

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

Note: If you have flashed a new kernel to the NOR Flash at an address other than 0x2050000, modify the bootcmd definition accordingly.

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.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 <tftp server ip address>
EVM # setenv bootfile <name of kernel image>
EVM # setenv rootpath <root directory to mount>
EVM # setenv nfshost <ip address of nfs host>
EVM # setenv bootargs video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
   console=ttyS0,115200n8 noinitrd rw ip=dhcp root=/dev/nfs
   nfsroot=$(nfshost):$(rootpath),nolock mem=120M
```

The <*root directory to mount*> must match the file system 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 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 /home/<*useracct*>/dvsdk_#_#/dsplink_#/packages/dsplink/make/Linux/ directory, to make sure the BASE_BUILDOS and BASE_CGTOOLS variables correctly point to the correct locations. For example:

```
BASE_BUILDOS := /home/<useracct>/workdir/lsp/ti-davinci/linux-2.6.10_mvl401
 BASE_CGTOOLS := /opt/mv_pro_4.0.1/montavista/pro/devkit/arm/v5t_le/bin
```

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>/dvsdk_#_#/dsplink_#/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/DM6446/RELEASE/ directory.

A.6 Restoring and Updating the EVM Hard Disk Drive

This section describes how to restore and update all the files on the EVM hard disk drive (HDD), including the Linux file system and the demos. Using these restore procedures, you can return your board to a known state if anything happens to the data on the EVM 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 EVM as described in Section 4.3.5 and Section 4.3.6.

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

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.

Note: If you have flashed a new kernel to the NOR Flash at an address other than 0x2050000, then the flash memory address will be different.

A.6.1 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 EVM 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 EVM board to the host workstation.
5) On the host workstation, open a terminal session to the target EVM 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 EVM can use for root mounting.

A.6.2 Configure EVM for NFS Root Mount

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

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

2) Power on the EVM 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
```

Note: If you have flashed a new kernel to the NOR Flash at an address other than 0x2050000, modify this command accordingly.

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

```
EVM # setenv rootpath <root directory to mount>
EVM # setenv nfshost <ip address of nfs host>
EVM # setenv bootargs video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
   console=ttyS0,115200n8 noinitrd rw ip=dhcp root=/dev/nfs
   nfsroot=$(nfshost):$(rootpath),nolock mem=120M
```

The nfsroot option in this command uses the host workstation IP address. Make sure to replace the <*host ip addr*> 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 EVM from NFS on the host Linux workstation

```
EVM # boot
```

8) Log into MontaVista Linux as "root".

A.6.3 Restore the EVM Hard Disk Drive

The EVM hard disk drive (HDD) can be restored from a target EVM HDD partition or from the host Linux workstation file system. Either method will achieve the same result.

Restoring the EVM 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 EVM and configure U-Boot to root mount via the local HDD. The steps for this type of boot are provided in Section A.4.1, *Booting from Flash Using the EVM's Hard Drive File System*.

A.6.4 Restoring From Target EVM 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
```

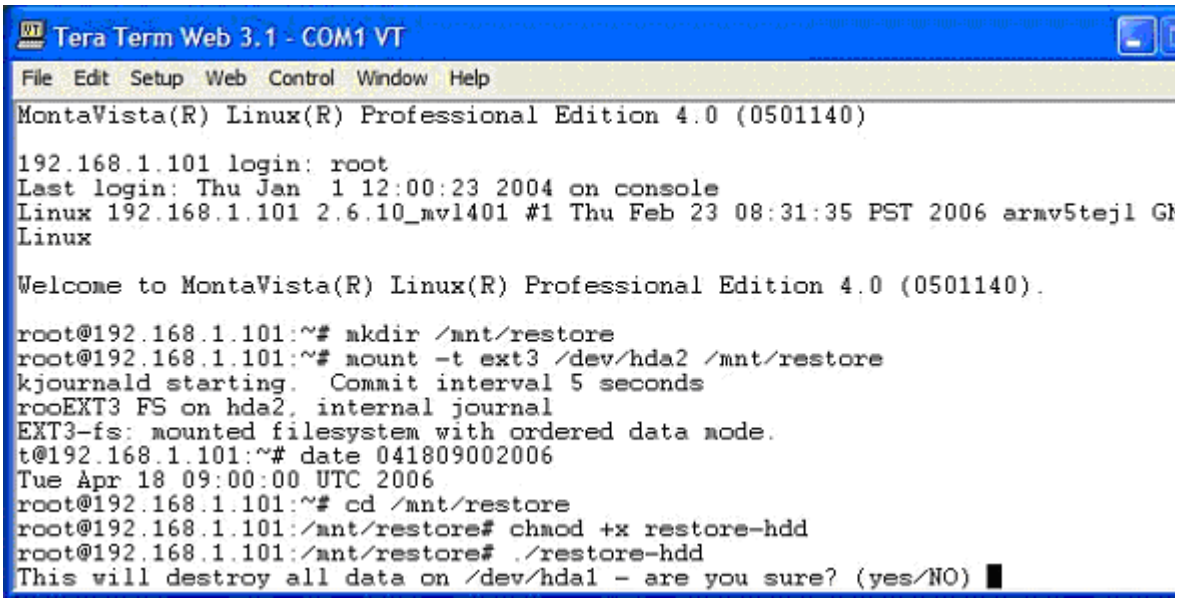
5) Add execute permissions for the script:

```
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**.



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

```
EVM # halt
```

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

10) Restart the EVM and configure U-Boot to root mount via the local HDD. Follow the steps in Section A.4.1, *Booting from Flash Using the EVM's Hard Drive File System*.

A.6.5 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 EVM.

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

1) On your host workstation, copy the restore directory from the DVSDK CD #3 to /home/<*useracct*>/workdir/filesys/restore. After NFS mounting this file system, this will appear as the /restore directory on the EVM.

2) Login to the EVM as root.

3) Go to the /restore directory.

```
EVM # cd /restore
```

4) Set the Linux date variable to today's date. If the date is too far off, the target file system installation generates warnings for each file it installs.

```
EVM # date MMDDHHMMCCYY
```

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

5) Add execute permissions on the prep-hdd script.

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

6) Run the prep-hdd script in the /restore directory.

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

7) The script will ask for confirmation: "Are you sure you want to partition and format /dev/hda1?" Type **yes**.

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

```
EVM # halt
```

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

10) Restart the EVM and configure U-Boot to root mount via the local HDD. Follow the steps in Section A.4.1, *Booting from Flash Using the EVM's Hard Drive File System*.

If you instead manually restore the overlay.tar.gz file from CD 1 on an NFS file system, the demos will likely produce a cryptic error. Although such manual restores are not documented or recommended, you can correct the problem by turning off the s-bit as follows:

```
EVM # cd /opt/dvsdk/dm6446
EVM # chmod u-s ${install}/encode ${install}/decode ${install}/encodedecode \
      ${install}/interface
```

Index

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

X

Y

Spectrum Digital, Inc.
508168-0001B