TMS320C6201/6701 Evaluation Module User's Guide

PRINTED WITH
SOY INK™

TEXAS INSTRUMENTS

Printed on Recycled Paper

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 that 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.

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265

Read This First

About This Manual

This manual tells you how to install and operate the TMS320C6x ('C6x) evaluation module (EVM). This module contains two boards, 'C6201 (fixed-point) and 'C6701 (floating-point).

The 'C6x EVM is a peripheral component interconnect (PCI) plug-in card that is compliant with the *PCI Local Bus Specification Revision 2.1*. The 'C6x EVM helps you evaluate characteristics of the 'C6x digital signal processor (DSP) to determine if it meets your application requirements. The 'C6x EVM is intended for use in a PCI expansion slot inside the PC™. It can also be operated outside the PC on a desktop or lab bench with the use of an external power supply and emulator (XDS510 or XDS510WS). The power supply and emulator are not included in the kit.

This manual assumes that you are familiar with working in a Windows™ 95 or a Windows NT™ environment and understand general and technical PC terminology. This manual specifically addresses the 'C6x EVM and revision 1.3 support software. Detailed information about the 'C6x DSP and TI code development support tools is provided separately (see the *Related Documentation From Texas Instruments* section in this *Preface* for a list of documents and ordering information). For up-to-date information on the 'C6x EVM, as well as related products, visit the 'C6x EVM website at **http://www.ti.com/sc/docs/dsps/tools/c6000/c62xevm/index.htm**.

How to Use This Manual

This users guide provides the following types of information about the 'C6x EVM:

❑ **Introductory information**, consisting of Chapter 1, provides a high-level overview of the 'C6x EVM hardware and support software and describes how the product can be used during code development and debugging.

❑ **Installation information**, consisting of Chapters 2 and 3, provides hands-on information to help you get started using the 'C6x EVM.

■ Chapter 2 identifies the system requirements and 'C6x EVM kit contents. It also provides step-by-step hardware installation procedures.

■ Chapter 3 provides instructions for running the EVM confidence test.

■ Chapter 4 describes the 'C6x EVM host software utilities.

Notational Conventions

This document uses the following conventions:

❑ Program listings, program examples, and interactive displays are shown in a `special typeface`. Some examples use a **bold version** for emphasis; interactive displays use a **bold version** to distinguish commands that you enter from items that the system displays (such as prompts, command output, error messages, etc.).

Here is a sample program listing with the evm6x_close() function highlighted for emphasis:

```
#include <windows.h>
#include <evm6xdll.h>
. . .
    HANDLE    h_board;
    h_board = evm6x_open( 0, FALSE );
    if ( h_board == INVALID_HANDLE_VALUE )
    {
        exit(-1);
    }
. . .
    evm6x_close( h_board );
```

❑ In syntax descriptions, the instruction or command is in a **bold face**, and parameters are in *italics*. Portions of a syntax that are in **bold** should be entered as shown; portions of a syntax that are in *italics* describe the type of information that should be entered. Here is an example of a command syntax:

evm6xldr *filename*

❑ Square brackets (**[** and **]**) identify an optional parameter. If you use an optional parameter, you specify the information within the brackets; you do not enter the brackets themselves. Here is an example of a command that has optional parameters.

evm6xtst [*options*] [*log_filename*]

evm6xtst is the command. This command has two optional parameters, indicated by *options* and *log_filename*.

❑ Device pins often are represented in groups. Device pin group notation consists of the pin name followed by brackets containing the range of pins included in the group. A colon separates the numbers in the range. For ex-

ample, PDB[7:0] represents the 8 peripheral data bus pins (PDB7 through PDB0) on a device.

❑ The TMS320C6x family of devices is referred to as the 'C6x. The following abbreviations may be used in this manual for TI devices on the 'C6x EVM:

Abbreviation	Device Definition
'C6x01	TMS320C6201/6701
'ALB16244	SN74ALB16244
'CBT3257	SN74CBT3257
'CBTD3384	SN74CBTD3384
'LVT125	SN74LVT125
'LVTH162245	SN74LVTH162245

Information About Cautions and Warnings

This book contains cautions and warnings.

This is an example of a caution statement.

A caution statement describes a situation that could potentially damage your software or equipment.

This is an example of a warning statement.

A warning statement describes a situation that could potentially cause harm to <u>you</u>.

The information in a caution or a warning is provided for your protection. Please read each caution and warning carefully.

Related Documentation From Texas Instruments

The following books describe the 'C6x processor and related support tools. To obtain a copy of any of these TI documents, call the Texas Instruments Literature Response Center at (800)477–8924. When ordering, please identify the book by its title and literature number.

TMS320C6x Assembly Language Tools User's Guide (literature number SPRU186) describes the assembly language tools (assembler, linker, and other tools used to develop assembly language code), assembler directives, macros, common object file format, and symbolic debugging directives for the 'C6x generation of devices.

TMS320C6x C Source Debugger User's Guide (literature number SPRU188) tells you how to invoke the 'C6x simulator and emulator versions of the C source debugger interface. This book discusses various aspects of the debugger, including command entry, code execution, data management, breakpoints, profiling, and analysis.

TMS320C6x Optimizing C Compiler User's Guide (literature number SPRU187) describes the 'C6x C compiler and the assembly optimizer. This C compiler accepts ANSI standard C source code and produces assembly language source code for the 'C6x generation of devices. The assembly optimizer helps you optimize your assembly code.

TMS320C62x/C67x CPU and Instruction Set Reference Guide (literature number SPRU189) describes the 'C62x/C67x CPU architecture, instruction set, pipeline, and interrupts for these digital signal processors.

TMS320C6201/C6701 Peripherals Reference Guide (literature number SPRU190) describes common peripherals available on the TMS320C6201/C6701 digital signal processors. This book includes information on the internal data and program memories, the external memory interface (EMIF), the host port, multichannel buffered serial ports, direct memory access (DMA), clocking and phase-locked loop (PLL), and the power-down modes.

TMS320C62x/C67x Programmer's Guide (literature number SPRU198) describes ways to optimize C and assembly code for the TMS320C62x/C67x DSPs and includes application program examples.

TMS320C62x/C67x Technical Brief (literature number SPRU197) gives an introduction to the 'C62x/C67x digital signal processors, development tools, and third-party support.

XDS51x Emulator Installation Guide (literature number SPNU070) describes the installation of the XDS510™, XDS510PP™, and XDS510WS™ emulator controllers. The installation of the XDS511™ emulator is also described.

TMS320 DSP Development Support Reference Guide (literature number SPRU011) describes the TMS320 family of digital signal processors and the tools that support these devices. Included are code-generation tools (compilers, assemblers, linkers, etc.) and system integration and debug tools (simulators, emulators, evaluation modules, etc.). Also covered are available documentation, seminars, the university program, and factory repair and exchange.

TMS320C6x Peripheral Support Library Programmer's Reference (literature number SPRU273) describes the contents of the 'C6x peripheral support library of functions and macros. It lists functions and macros both by header file and alphabetically, provides a complete description of each, and gives code examples to show how they are used.

TMS320C6201 Digital Signal Processor Data Sheet (literature number SPRS051) describes the features of the TMS320C6201 fixed-point DSP and provides pinouts, electrical specifications, and timings for the device.

TMS320C6701 Digital Signal Processor Data Sheet (literature number SPRS067) describes the features of the TMS320C6701 floating-point DSP and provides pinouts, electrical specifications, and timings for the device.

TMS320C6201/6701 Evaluation Module Technical Reference, (literature number SPRU305) can be accessed through the following URL:

http: //www.ti.com/sc/docs/dsps/tools/c6000/c6xevm/index.htm

Related Documentation

You can use the following specification to supplement this reference guide:

PCI Local Bus Specification Revision 2.1, PCI Special Interest Group, June 1, 1995.

Trademarks

320 Hotline On-line, VelociTI, XDS510, XDS510PP, XDS510WS, and XDS511 are trademarks of Texas Instruments Incorporated.

IBM and PC are trademarks of International Business Machines Corporation.

Intel and Pentium are trademarks of Intel Corporation.

Microsoft, Windows, and Windows NT are registered trademarks of Microsoft Corporation.

OpenWindows, Solaris, and SunOS are trademarks of Sun Microsystems, Inc.

SPARCstation is trademark of SPARC International, Inc., but licensed exclusively to Sun Microsystems, Inc.

Obtaining Technical Support

Before contacting Texas Instruments Technical Support, please have the following information available:

❏ Assembly number of your 'C6x EVM (e.g., D600830-0001) located on the bottom side of the board

❏ Revision number of your 'C6x EVM located in parentheses next to the assembly number on the bottom side of the board

❏ Serial number located on the bottom side of the board

❏ Record of the 'C6x EVM confidence test utility results that identifies potential problems and other revision numbers (software, EEPROM, CPLD). Chapter 3, *Running the Board Confidence Test*, explains how to run the test.

❏ Computer's PCI BIOS brand name and version number

❏ Amount of memory in your computer system

❏ Version of the software and operating environment you are using such as Windows NT 4.0

❏ Version of the code generation tools your are using

❏ Version of the debugger you are using

❏ If you are using Windows 95, print out a report of your system configuration by performing the following steps:

  1) Right click on the My Computer icon on the desktop.

  2) Select the Properties menu item.

  3) Select the Device Manager tag.

  4) Select the Print button.

  5) Select the System summary radio button.

  6) Click on the OK button to print a system resource summary.

Have this system resource summary available when you contact technical support.

> **Note:**
>
> Check the system resource summary to see if the IRQ assigned to TI TMS320C6x EVM is shared with another device. If it is, this is probably the problem. See Appendix A, *Troubleshooting*, for the corrective action.

Once you have this information ready, contact Texas Instruments Technical Support as specified in the *If You Need Assistance* section that follows.

If You Need Assistance . . .

❏ **World-Wide Web Sites**

TI Online	http://www.ti.com
Semiconductor Product Information Center (PIC)	http://www.ti.com/sc/docs/pic/home.htm
DSP Solutions	http://www.ti.com/dsps
320 Hotline On-line ™	http://www.ti.com/sc/docs/dsps/support.htm

❏ **North America, South America, Central America**

Product Information Center (PIC)	(972) 644-5580		
TI Literature Response Center U.S.A.	(800) 477-8924		
Software Registration/Upgrades	(214) 638-0333	Fax: (214) 638-7742	
U.S.A. Factory Repair/Hardware Upgrades	(281) 274-2285		
U.S. Technical Training Organization	(972) 644-5580		
DSP Hotline	(281) 274-2320	Fax: (281) 274-2324	Email: dsph@ti.com
DSP Modem BBS	(281) 274-2323		
DSP Internet BBS via anonymous ftp to ftp://ftp.ti.com/pub/tms320bbs			

❏ **Europe, Middle East, Africa**

European Product Information Center (EPIC) Hotlines:

Multi-Language Support	+33 1 30 70 11 69	Fax: +33 1 30 70 10 32
Email: epic@ti.com		
Deutsch	+49 8161 80 33 11 or +33 1 30 70 11 68	
English	+33 1 30 70 11 65	
Francais	+33 1 30 70 11 64	
Italiano	+33 1 30 70 11 67	
EPIC Modem BBS	+33 1 30 70 11 99	
European Factory Repair	+33 4 93 22 25 40	
Europe Customer Training Helpline		Fax: +49 81 61 80 40 10

❏ **Asia-Pacific**

Literature Response Center	+852 2 956 7288	Fax: +852 2 956 2200
Hong Kong DSP Hotline	+852 2 956 7268	Fax: +852 2 956 1002
Korea DSP Hotline	+82 2 551 2804	Fax: +82 2 551 2828
Korea DSP Modem BBS	+82 2 551 2914	
Singapore DSP Hotline		Fax: +65 390 7179
Taiwan DSP Hotline	+886 2 377 1450	Fax: +886 2 377 2718
Taiwan DSP Modem BBS	+886 2 376 2592	
Taiwan DSP Internet BBS via anonymous ftp to ftp://dsp.ee.tit.edu.tw/pub/TI/		

❏ **Japan**

Product Information Center	+0120-81-0026 (in Japan)	Fax: +0120-81-0036 (in Japan)
	+03-3457-0972 or (INTL) 813-3457-0972	Fax: +03-3457-1259 or (INTL) 813-3457-1259
DSP Hotline	+03-3769-8735 or (INTL) 813-3769-8735	Fax: +03-3457-7071 or (INTL) 813-3457-7071
DSP BBS via Nifty-Serve	Type "Go TIASP"	

❏ **Documentation**

When making suggestions or reporting errors in documentation, please include the following information that is on the title page: the full title of the book, the publication date, and the literature number.

Mail: Texas Instruments Incorporated   Email: dsph@ti.com
      Technical Documentation Services, MS 702
      P.O. Box 1443
      Houston, Texas   77251-1443

Note: When calling a Literature Response Center to order documentation, please specify the literature number of the book.

Contents

Figures

Tables

Introduction

This chapter lists the key features of the TMS320C6x evaluation module (EVM), provides functional overviews of the EVM hardware and software, and describes the EVM operating scenarios.

The TMS320C6x evaluation module (EVM) is a low-cost, general-purpose platform for the development, analysis, and testing of 'C6x digital signal processor (DSP) algorithms and applications. The 'C6x EVM allows you to evaluate the 'C6x DSP and algorithms to determine if they meet your application requirements. The 'C6x EVM hardware design information and software application programming interfaces (APIs) also provide a reference design that facilitates your own 'C6x-based hardware and software development.

The EVM is bundled with Code Composer, Windows 95 and NT drivers, host PC and DSP software APIs, example applications with source code, and various utility applications. The hardware and software bundle provides an integrated package that allows you to quickly evaluate the 'C6x DSP's performance and develop custom applications.

1.1 Key Features of the TMS320C6x EVM

The 'C6x EVM has the following key features:

❑ 'C6201 fixed-point or 'C6701 floating-point DSP

❑ Quad clock support (see Table 1–1)

Table 1–1. Quad Clock Support Frequencies

	OSC A		OSC B	
	x1	x4	x1	x4
'C6201 EVM	33.25 MHz	133 MHz	50 MHz	200 MHz
'C6701	25 MHz	100 MHz	33.25 MHz	133 MHz

❑ Peripheral component interconnect (PCI) interface with initiator/target (master/slave) support

❑ 256K bytes of 133-MHz synchronous burst static random-access memory (SBSRAM)

❑ 32M bytes of 100-MHz synchronous dynamic RAM (SDRAM)

❑ Embedded JTAG emulation via PCI or external XDS510 support

❑ Access to all DSP memory from the PCI bus via Host port interface

❑ Stereo 16-bit audio codec that supports sample rates from 5.5 kHz–48 kHz

❑ Onboard switching voltage regulators for 1.8/2.5 volts direct current (V_{DC}) and 3.3 V_{DC}

❑ Onboard linear voltage regulator for analog 5 V_{DC}

❑ Three light emitting diode (LED) indicators (one power-on indicator and two user-defined indicators)

❑ Internal PCI operation

❑ External desktop operation (requires external power supply and XDS510 or XDS510WS emulator, which are not included in the 'C6x EVM kit)

❑ Expansion memory and DSP peripheral connectors for daughterboard support

❑ Plug-and-play PCI device

1.1.1 User Controls and Indicators

The 'C6x EVM has the following user controls and indicators:

❏ JTAG emulation (embedded or external)

❏ DSP oscillator clock selection (OSC A or OSC B – see Table 1–1)

❏ Clock mode (multiply-by-1 or multiply-by-4 for either oscillator)

❏ DSP boot mode selection:

■ None (no-boot mode)

■ HPI boot mode

■ Read-only memory (ROM) boot mode (requires optional daughter-board)

❏ Little- or big-endian memory addressing

❏ Three user-defined dual in-line package (DIP) switches

❏ Manual reset

❏ Power-on LED indicator

❏ Two user-defined LED indicators

1.1.2 External Interfaces

The 'C6x EVM has the following external interfaces:

❏ PCI revision 2.1-compliant interface (5 V, 32-bit)

❏ Molex 4-pin external power connector (5 V, 12 V, –12 V, and GND)

❏ 2-pin power connector for cooling fan (5 V)

❏ 14-pin external JTAG header

❏ 10-pin complex programmable logic device (CPLD) in-system program-mable (ISP) header (not installed)

❏ Three 3.5-mm audio jacks (line in, line out, and microphone)

❏ Two low-profile 80-pin (.050-inch) daughterboard connectors (EMIF and DSP peripheral expansion)

1.2 The TMS320C6x EVM Board

The 'C6x EVM is a full-size PCI board with approximate dimensions of 4.2 inches wide, 12.28 inches long, and 0.49 inches high (excluding its bracket and retainer). The 'C6x EVM is intended for use in a PCI expansion slot on the computer's motherboard. The 'C6x EVM can be operated stand-alone on a desktop with the use of an external power supply and XDS510 or XDS510WS emulator. (The power supply and the emulator required for stand-alone operation are not included in the kit.)

Note:

If you are using the 'C6x EVM in a PCI expansion slot, make sure there is an unobstructed slot open for the full-length PCI card. See section 2.3.2, *Avoiding Obstructions to the EVM Board*, for more information.

The 'C6x EVM has a 'C6201 or 'C6701 DSP onboard that allows full-speed verification of 'C6x code with the included source debugger. The 'C6x EVM provides a PCI interface, SBSRAM and SDRAM, a 16-bit audio codec, and embedded JTAG emulation support. Connectors on the 'C6x EVM provide DSP external memory interface (EMIF) and peripheral signals that enable its functionality to be expanded with custom or third-party daughterboards.

The EVM provides a 'C6x hardware reference design that can assist you in the development of your own 'C6x-based products. In addition to providing a reference for interfacing the DSP to various types of memories and peripherals, the design also addresses power, clock, JTAG, and PCI controller interfaces.

Figure 1–1 and Figure 1–2 depict the top and bottom sides, respectively, of the 'C6201 EVM. The figures identify the major components on the 'C6x EVM board.

Figure 1–1. TMS320C6x EVM Layout (Top Side)
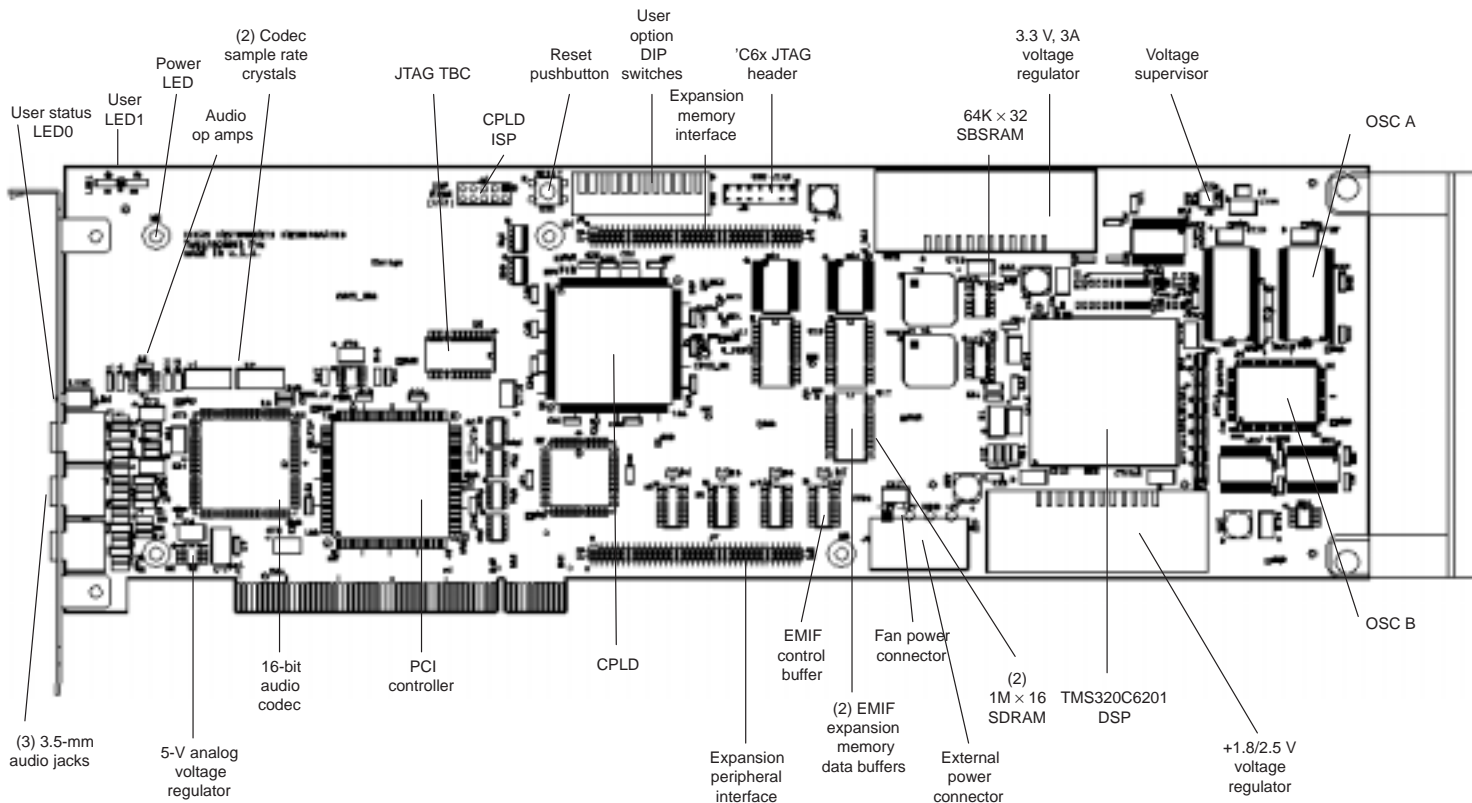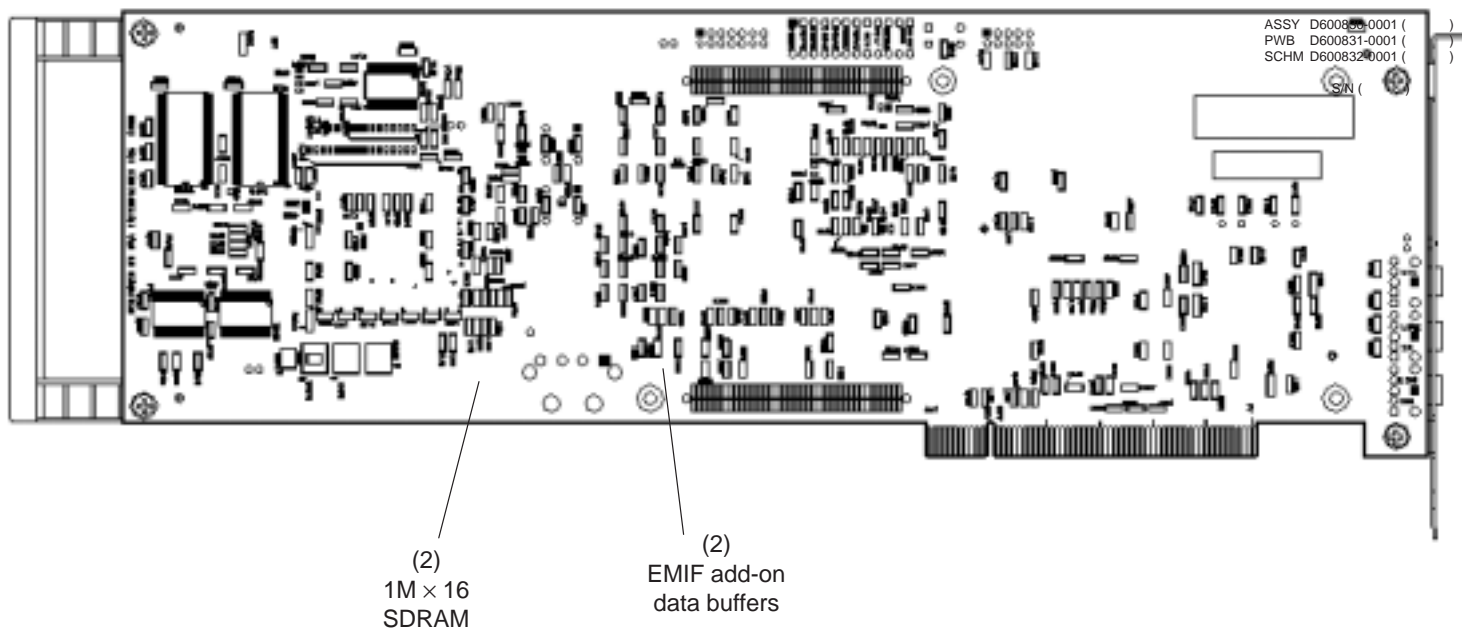
Figure 1–1. TMS320C6x EVM Layout (Top Side)

Figure 1–2. TMS320C6x EVM Layout (Bottom Side)

ASSY D600850-0001 (   )
PWB   D600831-0001 (   )
SCHM D600832-0001 (   )

S/N (   )

(2)
1M × 16
SDRAM

(2)
EMIF add-on
data buffers

1.3 TMS320C6x EVM Hardware Functional Overview

Figure 1–3 shows the basic functional block diagram and interfaces of the 'C6x EVM.

Figure 1–3. TMS320C6x EVM Block Diagram
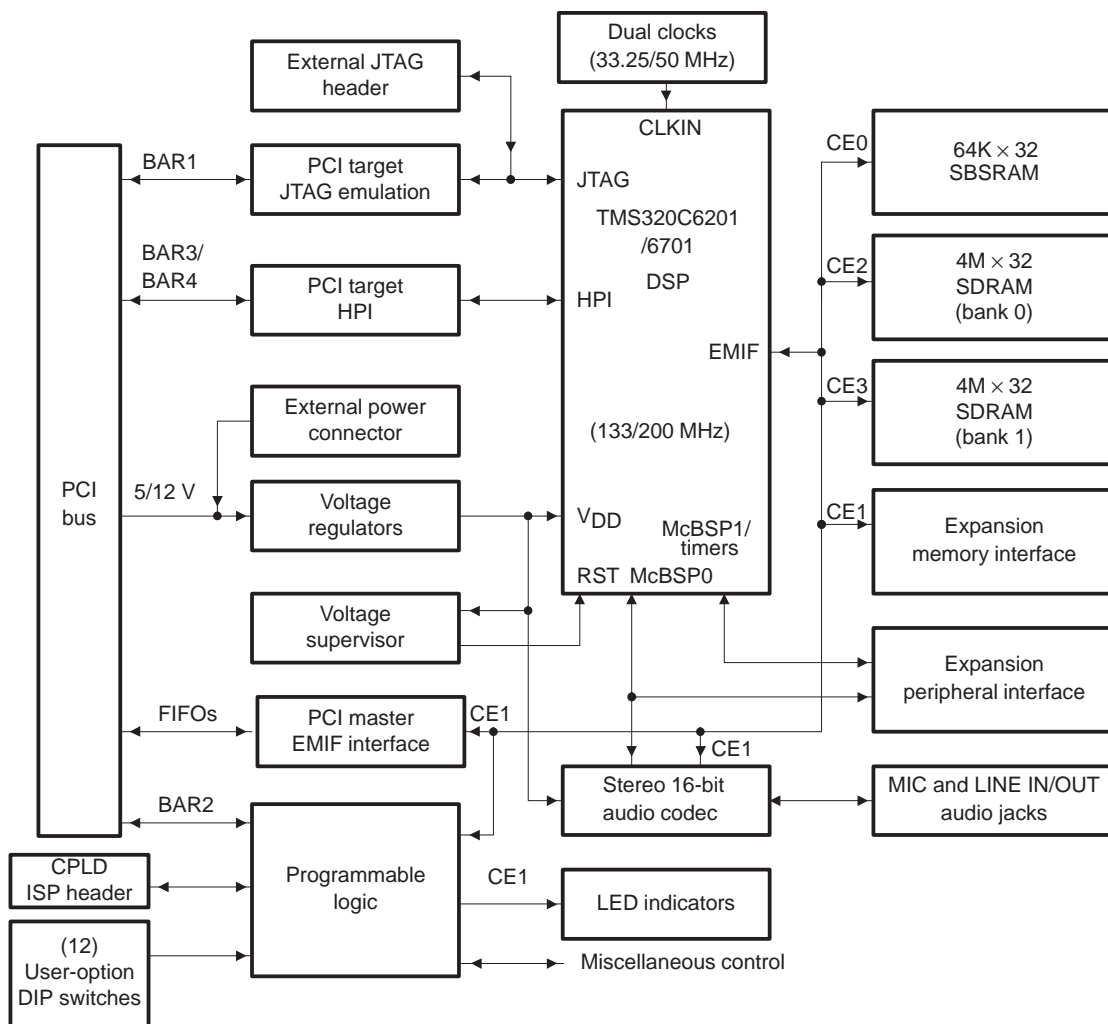
The 'C6x EVM hardware can be divided into 12 functional areas. This section provides an overview of each of these 12 areas. Detailed descriptions of the

EVM hardware are provided in Chapter 1, *TMS3206201/6701 Evaluation Module Technical Reference*.

❑ **DSP.** The 'C6x EVM is built around the 'C6201 or 'C6701 DSP, which operates up to 1600 MIPS with a CPU clock rate of 200 MHz. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.2, *TMS320C6201/6701 DSP.*

❑ **DSP clocks**. The 'C6x EVM supports operation with two different onboard clock sources (OSC A and OSC B) and two clock modes (multiply-by-1 and multiply-by-4). As a result, the DSP can operate at four different clock rates, including OSC A, OSC A x 4, OSC B, or OSC B x 4. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.3, *DSP Clocks.*

❑ **External memory**. The 'C6x EVM provides one bank of 64K $\times$ 32, 133-MHz SBSRAM and two banks of 4M $\times$ 32, 100-MHz SDRAM. Additional asynchronous memory can be added with a daughterboard using the expansion memory interface. All external memory devices are byte-addressable. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference,* Chapter 1, section 1.4*, External Memory.*

❑ **Expansion interfaces**. The 'C6x EVM provides external memory interface and external peripheral interface connectors that enable the use of a custom or third-party daughterboard. The memory interface supports asynchronous memory transfers, and the peripheral interface provides daughterboard use of the DSP internal peripherals. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.5*, Expansion Interfaces.*

❑ **PCI interface**. The 'C6x EVM includes a *PCI Local Bus Revision 2.1*-compliant interface that enables host access to the onboard JTAG controller, DSP host port interface (HPI), and board control/status registers. The 'C6x EVM's PCI interface allows source debugging with the 'C6x EVM without requiring an XDS510 emulator, as well as host software access to all of the DSP memory space via the PCI bus. The 'C6201/6701 DSP can also master the PCI bus to transfer data to and from the host memory. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.6*, PCI Interface.*

❑ **JTAG emulation**. The 'C6x EVM provides embedded JTAG emulation, using an onboard test bus controller (TBC), as well as a header to support an XDS510 JTAG emulator. This allows source debugging over the PCI bus without requiring an emulator or by using an XDS510 emulator when operating stand-alone on a desktop. For more information, see

TMS3206201/6701 Evaluation Module Technical Reference, Chapter 1, section 1.7, *JTAG Emulation.*

❏ **Programmable logic**. The 'C6x EVM's CPLD provides the board's glue-logic and control/status registers. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.8*, Programmable Logic.*

❏ **Audio interface**. The 'C6x EVM includes a CD-quality, 16-bit audio interface with stereo microphone and line-level inputs and a stereo line-level output. A multimedia audio codec is used that supports all popular sample rates from 5.5kHz–48 kHz. The audio circuit includes an op amp based microphone preamplifier. Three 3.5-mm audio jacks are located on the board's mounting bracket. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.9, *Stereo Audio Interface.*

❏ **Power supplies**. The 'C6x EVM uses voltage regulators to provide 1.8 V or 2.5 V for the DSP core; 3.3 V for the DSP I/O, memories, CPLD, and buffers; and 5 V for audio components. The PCI bus or external power connector's 5 V is used for all other digital components. The PCI bus or external power connector's 12 V is used for the input to the 5 V regulator. Efficient switching regulators are used for the 1.8/2.5 V and 3.3 V, and a low-power linear regulator is used to generate the analog 5 V. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference,* Chapter 1, section 1.10, *Power Supplies.*

❏ **Voltage supervision and reset control**. The 'C6x EVM uses a voltage supervisor to monitor the board's voltages and provide a board reset signal. The CPLD also includes logic related to reset control with inputs from a manual reset pushbutton, the PCI controller, and software reset control signals. For more information, see T*MS3206201/6701 Evaluation Module Technical Reference*, Chapter 1, section 1.11, *Voltage Supervision.*

❏ **User options**. The 'C6x EVM supports user option control via 12 onboard DIP switches or with direct control by host software via the PCI bus. The user options include the boot mode, clock mode, clock select, JTAG select, and endian mode. Three user-defined options are also provided. For more information, see Chapter 2, section 2.4, Table 2–1, *TMS320C6x User Options DIP Switches.*

❏ **LED indicators**. The 'C6x EVM provides three LED indicators. A single green LED is illuminated whenever 5 V is applied to the board. Two red LEDs can be used for user-defined status, with one located on the board's mounting bracket and the other located at the top of the board. For more information, see section 2.5, *Indicators.*

1.4 TMS320C6x EVM Software Functional Overview

The 'C6x EVM software consists of *host support software* and *DSP support software*.

The host support software supplied with the 'C6x EVM board includes the following Win32 host utilities and libraries. The host utilities and host libraries run on an Intel™ compatible PC under either Windows 95 or Windows NT 4.0.

❏ **Board configuration utility (evm6xrst.exe).** This utility is used to reset and configure the board. For more information, see section 4.3, *EVM Board Configuration*.

❏ **Code Composer.** Code composer is the new software debugger that is used to debug 'C6x software on the board. For more information, see SPRU296, *Code Composer User's Guide*.

❏ **EVM confidence test utility (evm6xtst.exe).** This utility tests the basic operation of the board. For more information about the EVM confidence test, see section 4.2, *EVM Confidence Test*. To use the EVM confidence test to verify the hardware and software installation, see Chapter 3, *Running the Board Confidence Test*.

❏ **'C6x COFF loader utility (evm6xldr.exe).** This utility is used to load and execute 'C6x software on the board. For more information, see section 4.1, *EVM COFF Loader*.

❏ **EVM Win32 DLL (evm6x.dll).** The Win32 host libraries consist of a Windows 95 and a Windows NT version of evm6x.dll, which provides user software access for control and communication with the EVM board. For more information, see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 2, *TMS320C6x EVM Host Support Software*.

❏ **Example source code.** Example code that illustrates how to use the host Win32 dynamic link library (DLL) functions is provided with the 'C6x EVM. For more information, see see *TMS3206201/6701 Evaluation Module Technical Reference*, Chapter 2, section 2.4, *EVM Host Support Software Example*.

Figure 1–4 provides a block diagram of the EVM host software components and their relationships. The left side of the figure shows the Code Composer. The debugger makes direct calls to the low-level (ring 0) drivers to access the JTAG TBC device on the EVM board. The board configuration utility also makes direct calls to the drivers to access the board option registers on the EVM board.The remainder of the figure shows the Win32 host utilities and the components they use to access the hardware. The Win32 applications call the Windows 95 or Windows NT DLL that implement a consistent user-mode (ring 3) API. These DLLs make calls to the low-level (ring 0) drivers that provide access to the hardware.

Figure 1–4. TMS320C6x EVM Host Support Software Block Diagram
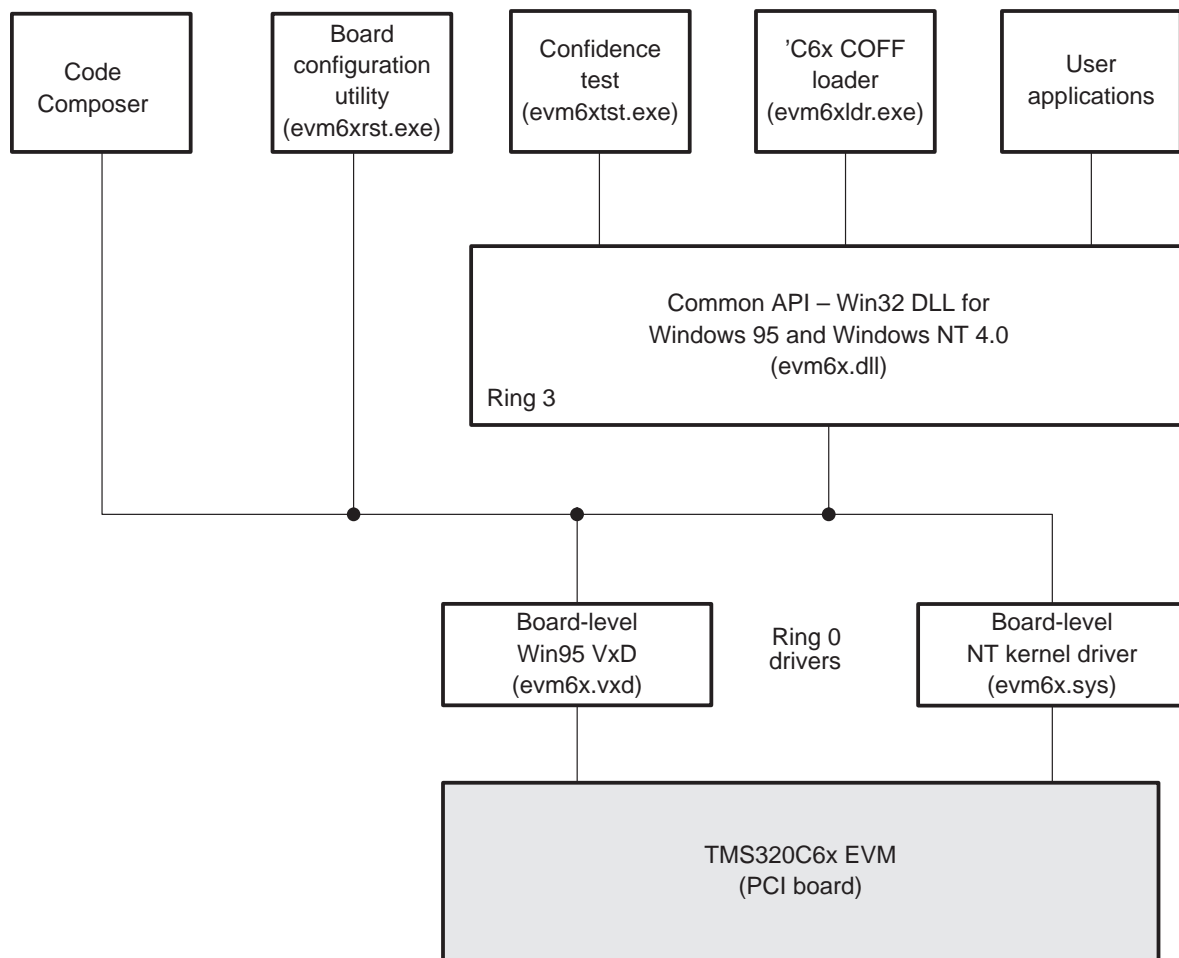
The DSP support software supplied with the EVM board includes the following components:

❑ **McBSP driver (mcbspdrv.c and mcbspdrv.h).** This driver is for the multichannel buffered serial port (McBSP) peripheral. For more information, see *TMS320C6201/6701 Ealuation Module Technical Reference*, section 3.3, *McBSP Driver API*.

❑ **Codec library (codec.c and codec.h).** This library is a collection of routines that configures and controls the operation of the CS4231A audio codec device. For more information, see *TMS320C6201/6701 Evaluation Module Technical Reference,* section 3.4, *Codec Library API*.

❑ **Board support library (board.c and board.h).** This library provides 'C6x board-specific routines for EVM configuration and control. For more information, see *TMS320C6201/6701 Evaluation Module Technical Reference,* section 3.5, *Board Support Library API*.

❑ **Example source code.** Code examples are provided to demonstrate the use of the DSP McBSP, codec, and board support functions. For more information, see *TMS320C6201/6701 Evaluation Module Technical Reference,* section 3.6, *DSP Support Software Examples*.

Figure 1–5 provides a block diagram of the EVM 'C6x software components and their relationships. For more information, see *TMS320C6201/6701 Evaluation Module Technical Reference,* Chapter 3, *TMS320C6x EVM DSP Support Software*.

Figure 1–5. TMS320C6x EVM DSP Support Software Block Diagram
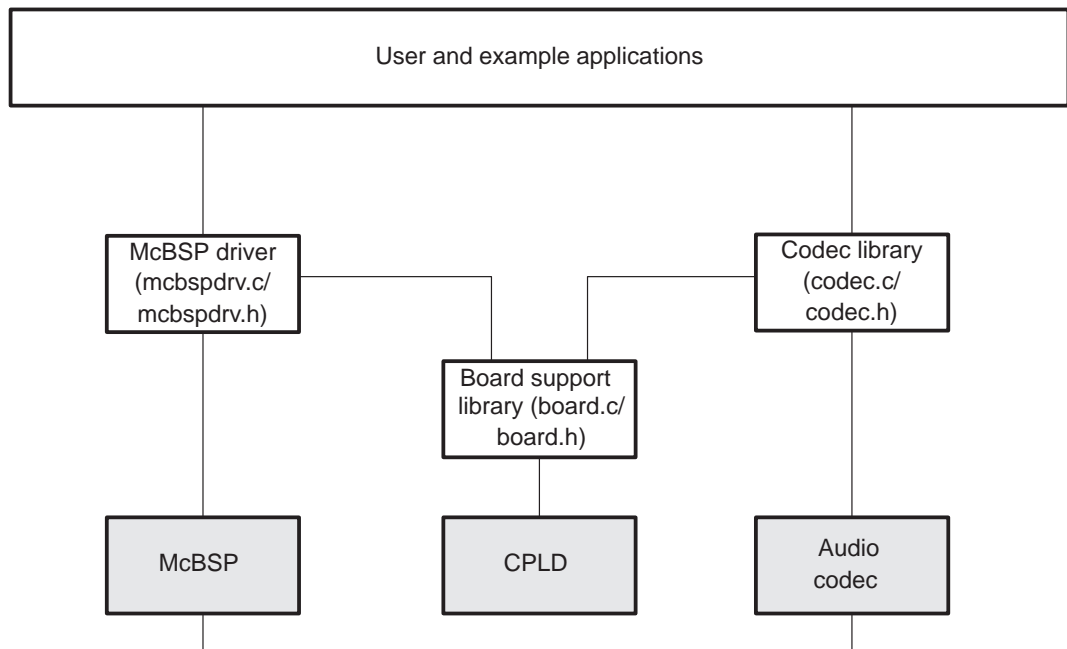
1.5 Operating Scenarios

The 'C6x EVM can operate in a computer motherboard's full-size PCI slot or stand alone on a desktop or lab bench. Because the 'C6x EVM does not have onboard ROM, applications are loaded by host software using the board's embedded JTAG or DSP HPI interfaces over the PCI bus or by an XDS510 emulator connected to the board's 14-pin JTAG header. A daughterboard with nonvolatile memory can be designed to provide the 'C6x EVM with a ROM boot capability that enables it to boot an application itself upon reset in both operating scenarios.

Table 1–2 summarizes the various ways applications can be loaded into the 'C6x in the PCI and stand-alone operating scenarios.

Table 1–2. Operating Scenarios Application Loading Summary

Operating Scenario	JTAG TBC Via PCI Bus	DSP HPI Via PCI Bus	XDS510 Emulator	ROM Daughterboard
PCI slot	Yes	Yes	Yes	Yes
Stand-alone	No	No	Yes	Yes

The following paragraphs summarize the PCI and stand-alone operating scenarios.

❏ **PCI operation.** When the 'C6x EVM is installed in a PCI slot, host software can control and monitor it via memory-mapped registers. The host software can also access the onboard JTAG test bus controller (TBC) to download code and access the DSP, as well as transfer data to and from the DSP using the 'C6201/6701 HPI. Code Composer uses the JTAG TBC to access the board, and the 'C6x EVM COFF loader uses the HPI. An XDS510 can be used with the board in a PCI slot, but it is not needed because onboard emulation control is provided. An optional ROM-boot mode can be used in PCI operation if a daughterboard with nonvolatile memory is installed.

❏ **Stand-alone operation.** When the 'C6x EVM is operated in a stand-alone configuration on a desktop or lab bench, an XDS510 or XDS10WS emulator with the standard 'C6x Code Composer (not the 'C6x EVM debugger included in the kit) is typically used to load applications and control the board. In stand-alone operation, the PCI interface is disabled, so the onboard JTAG TBC and HPI interfaces are not available. An optional ROM-boot mode can be used in stand-alone operation if a daughterboard with nonvolatile memory is installed.

Getting Started With the TMS320C6x EVM

This chapter explains how to get started using the 'C6x EVM. It provides a list of EVM kit contents, system requirements, and step-by-step hardware and software installation instructions.

Before you begin the installation, verify that you received all of the EVM kit components listed in section 2.1 and that your PC hardware and software platforms meet the minimum requirements described in section 2.2.

2.1 EVM Kit Contents

The following checklist details items that are shipped with the 'C6x EVM kit.

☐   **PCI plug-in card**   TMS320C6x EVM PCI plug-in card

☐   **CD-ROMs**   Two CD-ROMs are included in the kit:

TMS320C6x EVM Support Software. This CD-ROM contains the following software:

❏ **Device driver for Windows 95**. Includes common Win32 user-mode dynamic link library (DLL) with programmer API and low-level VxD driver

❏ **Device driver for Windows NT 4.0**. Includes common Win32 user-mode DLL with programmer API and low-level kernel mode driver

❏ **Host example code**. Example code that uses the Win32 user-mode DLL to access the EVM board resources

❏ **Code Composer**. Includes a C source debugger for Windows 95 and another for Windows NT 4.0

❏ **'C6x COFF loader**. A command-line utility that provides a means to configure the EVM board and load an application via the DSP HPI

❏ **Confidence test utility**. A command-line utility that verifies proper operation of the EVM by testing the major components on the board

❏ **Board configuration utility.** A command-line utility that resets and configures the EVM board.

❏ **'C6x peripheral support library**. 'C6x device library for DMA, timers, McBSPs, EMIF, HPI, and interrupt control

❏ **'C6x McBSP driver**. Stand-alone 'C6x McBSP driver in object and source formats

❏ **'C6x codec device library**. Audio codec device library in object and source formats

❏ **'C6x board support library**. EVM board support library in object and source formats.

❏ **'C6x example code**. Example DSP application using the McBSP driver, codec device library, and board support library

**CD-ROMS
(continued)**

TMS320C6x DSP CD-ROM. This interactive CD-ROM features the 'C6x DSP generation. It also contains full information about DSP Solutions from the Texas Instruments website at http://www.ti.com/dsps, including 165M bytes of DSP technical documentation.

Documentation

The following documentation is included:

❏ *TMS320C6201/6701 Evaluation Module User's Guide* (this book)

❏ *Microprocessor Development Systems TI Customer Support Guide*

❏ *Government Compliance Card*

❏ *TMS320C6x EVM Registration Card*

The following documentation is included on the CD–ROM:

❏ *TMS320C6x Peripheral Support Library Programmer's Reference*

❏ *TMS320C62x/C67x CPU and Instruction Set Reference Guide*

❏ *TMS320C6201/C6701 Peripherals Reference Guide*

❏ *TMS320C6x Code Composer*

❏ *TMS320C6201 Digital Signal Processor Data Sheet*

❏ *TMS320C6701 Digital Signal Processor Data Sheet*

2.2   System Requirements

The following sections list the system requirements for operating the 'C6x EVM in a PCI slot and in a stand-alone configuration.

2.2.1   PCI Operation

The following checklists detail the items that are required for operating the 'C6x EVM in a PCI slot.

Hardware checklist

❏	**Host**	Pentium™ PC or Pentium-compatible PC
❏	**Memory**	Minimum of 16M bytes (32M bytes recommended)
❏	**Hard disk space**	Minimum of 20M bytes
❏	**Monitor**	Color
❏	**Slot**	One full-length revision 2.1-compliant PCI local bus slot (5 V, 32 bits)
❏	**Board**	TMS320C6x EVM board (included in the kit)
❏	**Other hardware**	CD-ROM drive
❏	**Cables**	3.5-mm audio cables (optional)

Software checklist

❏	**Operating system**	Windows 95 or Windows NT 4.0
❏	**PC software tools**	Microsoft™ Visual C++ version 4.2 or greater (required for host-side user application development)
❏	**Files**	All other files are installed by the 'C6x EVM support software.

2.2.2 Stand-alone Operation

The following checklists detail the items that are required for operating the 'C6x EVM in a stand-alone configuration.

Hardware checklist

❑ **Host** One of the following:

    ❑ A 32-bit 486 IBM PC or compatible PC

    ❑ Minimum of a SPARCstation™ compatible system equivalent to a SPARCstation 2

❑ **Memory** Minimum of 16M bytes of RAM

❑ **Hard disk space** Minimum of 32M bytes of hard disk space for temp files and 15M bytes of hard disk space for software tools

❑ **Monitor** Color

❑ **Board** TMS320C6x EVM board (included in the kit)

❑ **Other hardware** CD-ROM drive

❑ **Power supply** 5 V_{DC} at 4 A

12 V_{DC} at 500 mA

−12 V_{DC} at 100 mA (optional for daughterboard)

❑ **Cables** ❑ Power cable with 4-pin Molex (disk-drive) connector

    ❑ 3.5-mm audio cables (optional)

❑ **Emulator** XDS510 or XDS510WS

Software checklist

❑ **Operating system** One of the following:

    ❑ For a PC platform, you need Windows NT 4.0 or higher or Windows 95

    ❑ For a SPARCstation platform, you need SunOS™ version 4.1.x or SunOS version 5.x (or Solaris™ 2.x) using an X Window System™ based window manager, such as OpenWindows™ version 3.0.

❑ **Debugger** TMS320C6x debugger for an XDS510 or XDS510WS (not the EVM 'C6x debugger included in the kit)

2.3 Before Starting the Installation Process

Before you begin installing the 'C6x EVM, perform the following tasks:

❏ Verify that you have received the items listed as part of the 'C6x EVM kit in section 2.1.

❏ Make sure that the system in which you are installing the EVM meets the hardware and software requirements listed in section 2.2.

❏ To help avoid equipment damage, read and follow the instructions in this section for controlling static electricity and avoiding obstructions that could prevent you from installing the 'C6x EVM.

Note:

If any items in the EVM kit are missing or damaged, contact Texas Instruments immediately. See *If You Need Assistance* in the *Preface* for the appropriate contact information.

2.3.1 Controlling Static Electricity

Minimizing Static Shock

Special handling methods and materials should be used to prevent equipment damage. You should be familiar with identification and handling of ESD-sensitive devices before attempting to perform the procedures identified in this manual.

To help avoid problems resulting from static electricity, follow these guidelines:

❏ Store the board in its antistatic bag until you are ready to use it.

❏ Keep the antistatic bag for storing or transferring the board between computers.

❏ Wear a properly connected ground strap at all times when handling the 'C6x EVM.

2.3.2 Avoiding Obstructions to the EVM Board

Avoiding Obstructions

To minimize the risk of damage to the 'C6x EVM, avoid physical obstructions that may prevent insertion of the board. You must install the EVM in a full-length PCI slot only.

Before installing the EVM, make sure that your PC has a full-length PCI slot. Otherwise, obstructions could prevent you from fully inserting the EVM in the motherboard PCI-bus connector. For example, in some computer systems, the location or height of the central processing unit (CPU) or the location of the PCI-bus slot may prevent you from inserting the EVM. Possible obstructions also include memory modules, processor fans, power supply circuits, and cable connections. Failure to avoid obstructions when installing the EVM could permanently damage the card.

2.4 EVM Hardware Installation for PCI Operation

Minimizing Personal Injury Risk

To minimize the risk of personal injury, always turn off the power to your PC and unplug the power cord before installing the 'C6x EVM board.

Note: You must have a full-length PCI slot available to install an EVM.

To install the 'C6x EVM board into your PC, follow these steps:

1) Turn the PC's power off and unplug the power cord.

2) Remove the cover from your PC.

3) Locate an existing unused PCI slot. Make sure that this slot allows the installation of a full-length PCI card. If the slot is a shared ISA/PCI slot that shares mechanical space, make sure that an ISA card is not already installed.

4) Remove the PCI slot's cover from the back of the computer by unscrewing it from the chassis.

5) Remove the 'C6x EVM from the antistatic bag. Keep the antistatic bag for storing or transferring the board in the future.

6) Insert the 'C6x EVM board into the empty, full-length PCI slot (see Figure 2–1). Secure its mounting bracket to the chassis with the previously removed screw. You do not have to set any jumpers or switches on the board for PCI operation.

Figure 2–1. TMS320C6x EVM PCI Installation
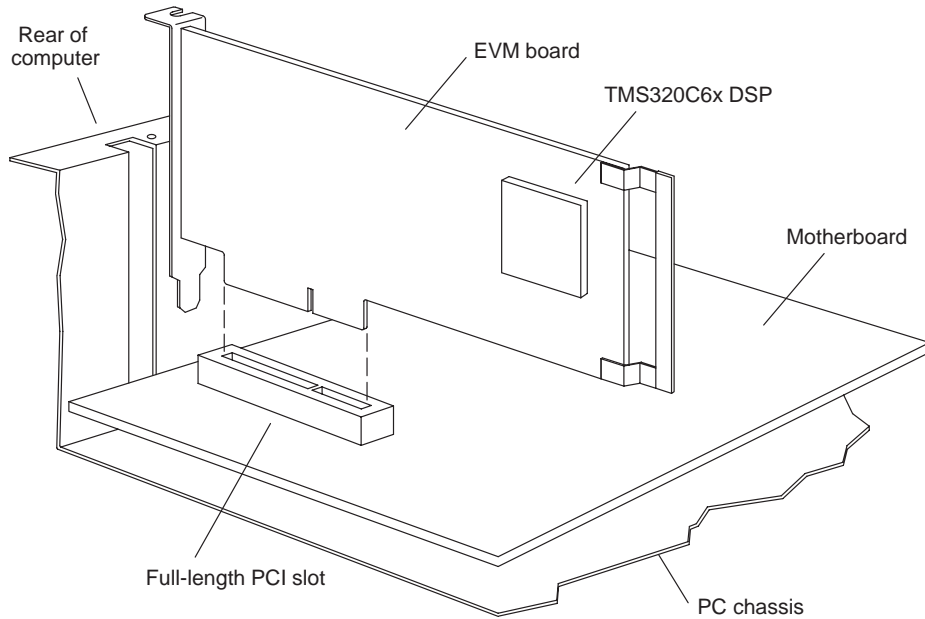
7) Add the dip switches. There are 12 dual in-line package (DIP) switches (SW2-1 through SW2-12) on the TMS320C6x EVM, as shown in Table 2–1. Table 2–2 lists the valid boot mode selections for the EVM. The highlighted (boldface) items in both tables denote the factory default switch settings.

*Table 2–1. TMS320C6x EVM User Options DIP Switches*

Switch Number	Name	OFF Selection	ON Selection
SW2-1– SW2-5	BOOTMODE4– BOOTMODE0	See Table 2–2	See Table 2–2
SW2-6	CLKMODE	Multiply-by-1 mode (PLL bypassed)	**Multiply-by-4 mode**
SW2-7	CLKSEL	OSC B	**OSC A**
SW2-8	ENDIAN	Big endian	**Little endian**
SW2-9	JTAGSEL	Internal (TBC)	**External (XDS510)**
SW2-10	USER2	1	**0**
SW2-11	USER1	1	**0**
SW2-12	USER0	1	**0**

Table 2–2. TMS320C6x EVM Boot Mode DIP Switches

Boot Mode	Map	Memory at Address 0	BM4 (SW2-1)	BM3 (SW2-2)	BM2 (SW2-3)	BM1 (SW2-4)	BM0 (SW2-5)
None	MAP 0	1/2-rate SBSRAM	ON	ON	ON	OFF	OFF
None	MAP 0	1×-rate SBSRAM	ON	ON	OFF	ON	ON
None	MAP 1	Internal	**ON**	**ON**	**OFF**	**ON**	**OFF**
HPI	MAP 0	External	ON	ON	OFF	OFF	ON
HPI	MAP 1	Internal	ON	ON	OFF	OFF	OFF

8) Replace the PC cover.

9) Connect the 3.5-mm audio cables to the EVM's microphone, line in, and line out connectors as needed. The connectors are labeled with MIC IN, LINE IN, and LINE OUT designations (see Figure 2–2).

Figure 2–2. Audio Cable Connectors on the EVM Board Mounting Bracket
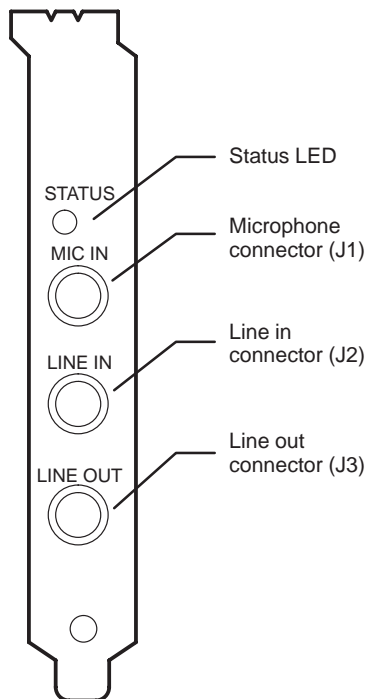
10) Plug in the PC's power cord and turn on the PC's power.

11) If your operating system is Windows NT, the hardware installation is finished. If your operating system is Windows 95, proceed to step 12.

12) When you turn on the PC's power, Windows 95 detects a newly installed EVM board and prompts you for the driver disk. Insert the EVM support software CD-ROM and follow the instructions on the screen. After the driver is installed, reboot the PC. For additional information about moving an EVM board to another PCI slot, adding another EVM board, and removing an EVM board, see section 2.6, *Notice to Windows 95 Users*.

2.5 Indicators

The EVM has three LEDs that provide a power-on indication and two user-defined status indications.

A green LED is hardwired on the 5-V supply, which can originate from the PCI bus during internal operation or the power connector during external operation. When the green LED is illuminated, it indicates that power is applied to the board.

Two red LEDs are controlled by CPLD register bits that are memory mapped in the DSP's CE1 memory space. One LED is located on the mounting bracket to allow you to see it when the PC cover is on. The other LED is located on top of the board to allow you to see it when the cover is off, without having to look at the back of the machine. Both LEDs are under user DSP control and are application dependent.

Because the LEDs are memory mapped in the DSP memory space, the host software can also control them from the HPI via the PCI bus.

2.6 Notice to Windows 95 Users

If an EVM board is moved from one slot to another one that has never had an EVM board installed in it, or if a second EVM is installed into a PCI slot that has never had an EVM board installed in it, Windows 95 recognizes the board as new hardware and installs the driver for the EVM board using the previously installed driver files. In this situation, Windows 95 does not prompt for a system reboot, although one is required for proper operation with the EVM board in the new slot. Therefore, whenever an EVM board is installed in a PCI slot for the first time, you must reboot the system after the driver is installed.

At initial installation, if you cancel the driver installation when Windows 95 prompts for a driver disk, you must perform the following steps to reinstall the driver properly. The board is then accessible via the driver.

1) Perform a complete driver uninstall.
2) Reboot the system.
3) When prompted, reinstall the driver using the driver disk.
4) Reboot the system.

If you remove an EVM board from your system, or if you want to move it from one PCI slot to another one, you must perform a *remove hardware* operation as follows:

1) From the Windows 95 desktop, right click on the My Computer icon.

2) Select the Properties menu item.

3) Select the Device Manager tab.

4) Select the Other devices item in the list of devices in the system.

5) Select the TI TMS320C6x EVM device under the Other devices item.

6) Click on the Remove button to remove the EVM driver associated with the current PCI slot.

2.7 EVM Hardware Installation for Stand-Alone Operation

Minimizing the Risk of Equipment Damage

Never disconnect or reconnect any cables or other hardware devices while power is applied to the emulator or EVM. Doing so can cause damage to your EVM.

In stand-alone operation, the 'C6x EVM must be connected to an external power supply (which is not included in the 'C6x kit) and an emulator. To install the 'C6x EVM for stand-alone operation as depicted in Figure 2–3, follow these steps:

1) Install your emulation hardware and software.

2) Turn the power supply off and unplug its power cord from the wall outlet.

3) Connect the power supply's DC output cable with the 4-pin female connector to the 'C6x EVM external power connector (J8).

4) Connect the XDS510 or XDS510WS emulator's 14-pin connector to the EVM's JTAG header (J5) located at the top-center of the EVM board.

Note:

Pin 6 of the 14-pin XDS510 emulator connector is filled in and the corresponding pin 6 of the JTAG header (J5) on the EVM is missing to ensure proper connection.

5) Configure the 'C6x EVM DIP switches (SW2) for the desired mode of operation. See Table 2–1 and Table 2–2 for DIP switch settings. The JTAG selection must be used to select external emulation (SW2-9 ON).

6) Connect the 3.5-mm audio cables to the EVM's microphone, line in, and line out connectors as needed. The connectors are labeled with MIC IN, LINE IN, and LINE OUT designations (see Figure 2–2).

7) Plug the power supply's power cord into the wall outlet.

8) Turn on the power supply and verify that the 'C6x EVM's green power indicator is illuminated.

9) Power up the emulator.

Figure 2–3. TMS320C6x EVM Stand-Alone Installation
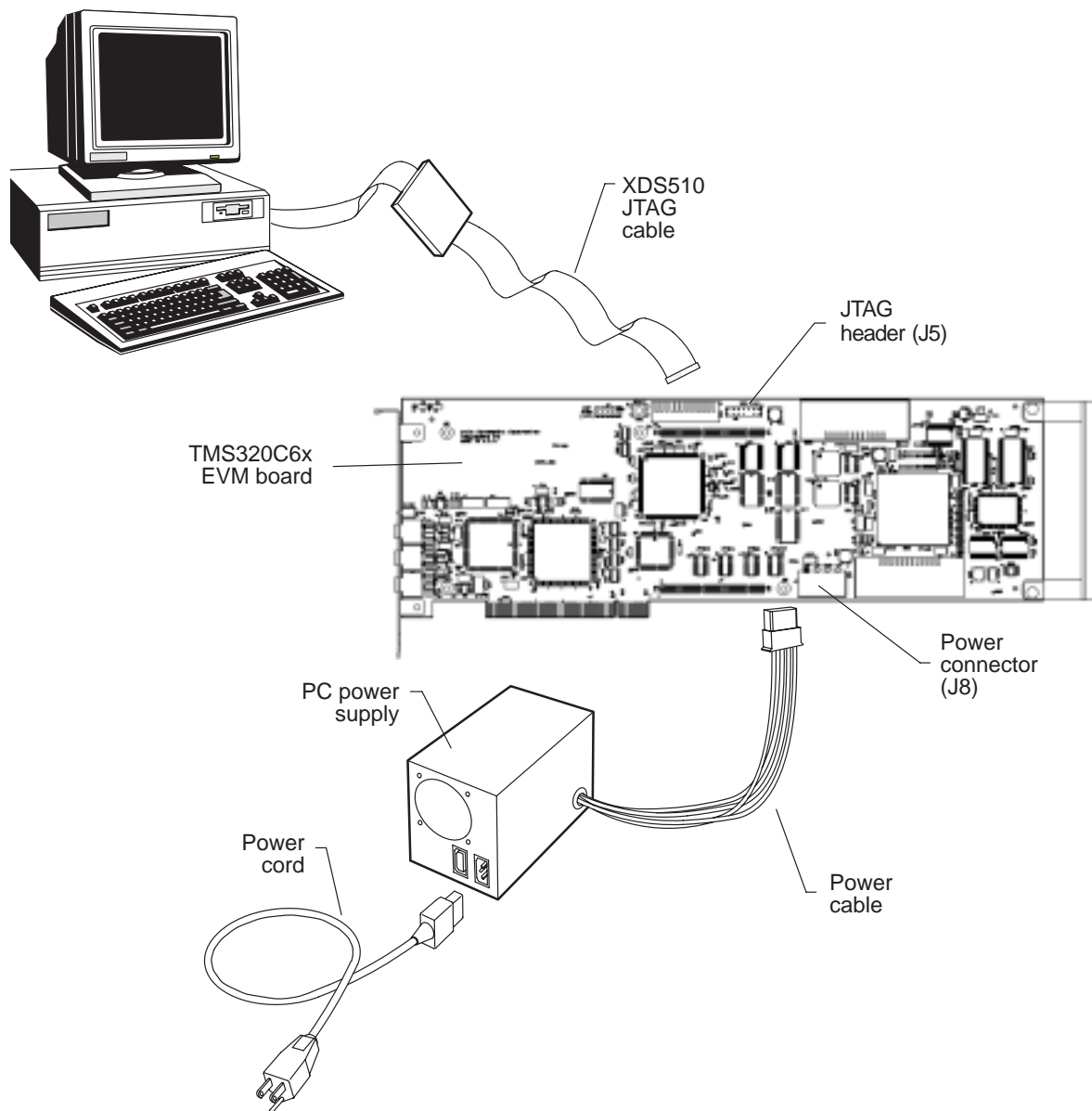
Running the Board Confidence Test

This chapter explains how to run the board confidence test included with the TMS320C6x EVM board. This test can only be run after the EVM board hardware and software have been installed. See Chapter 2, *Getting Started With the TMS320C6x EVM*, for installation instructions.

3.1 About the Board Confidence Test

The confidence test verifies proper operation of the major components of the EVM board and its support software. The test also displays board configuration information. These EVM hardware components are tested:

❑ SBSRAM
❑ SDRAM
❑ 'C6x interrupts
❑ 'C6x timers
❑ 'C6x DMA
❑ 'C6x McBSP serial ports
❑ PCI controller NVRAM (EEPROM)
❑ Audio codec
❑ Audio inputs and output
❑ LEDs
❑ Endian mode selection

The installation of these EVM software components is tested:

❑ Win32 DLL
❑ Host low-level board driver

The following section explains how to run the board confidence test utility to verify that the board and software have been installed properly. For additional information about the confidence test utility, see section 4.2, *EVM Confidence Test*.

3.2 Verifying the EVM Installation

To run the board confidence test utility to verify the installation of the board and software, follow these steps:

1) Install the 'C6x EVM board and supplied software as described in Chapter 2, *Getting Started With the TMS320C6x EVM*. If you want to verify the stereo audio capabilities of the EVM board during the confidence test, you must connect input sources to the microphone and line inputs and an amplified speaker to the line output.

2) Open a command window and change the directory to the conftest subdirectory of the EVM installation directory. This directory is created during the installation process and contains additional files needed for the execution of the confidence test utility.

3) Execute the confidence test utility in a command window by entering the following command on the command line:

```
evm6xtst test.log
```

This command starts the confidence test utility for the first EVM board and logs the test results to a file named test.log.

4) The output from the confidence test should be similar to the sample output provided in Example 3–1. If the test results in a failure message, see Appendix A, *Troubleshooting*.

Example 3–1. Sample Confidence Test Output

```
02:58:08 PM   Oct 09, 1998

Board Index              : 0
Board Type               : C6x01 EVM, Rev 0

Resetting Board.
DIP Switch Settings ...
Board Clock              : Osc. A's frequency multiplied by four
Endian Mode              : Little Endian
JTAG Selection           : External XDS510
User Defined Option bits : 0

Beginning Confidence Test ...

Testing Win32 DLL and Low-Level Driver access................PASSED.

***** Memory Tests.
CPU Clock                : Osc. A's frequency multiplied by four
Endian Mode              : Little Endian

Loading little endian memory test..............................PASSED.
SBSRAM A5 Test      : address 0x00000000; length 0x00010000...PASSED.
SDRAM0 A5 Test      : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 A5 Test      : address 0x03000000; length 0x00100000...PASSED.
SBSRAM Pulse Test   : address 0x00000000; length 0x00010000...PASSED.
SDRAM0 Pulse Test   : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 Pulse Test   : address 0x03000000; length 0x00100000...PASSED.

***** Memory Tests.
CPU Clock                : Osc. B's frequency multiplied by four
Endian Mode              : Little Endian

Loading little endian memory test..............................PASSED.
SBSRAM A5 Test      : address 0x00000000; length 0x00010000...PASSED.
SDRAM0 A5 Test      : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 A5 Test      : address 0x03000000; length 0x00100000...PASSED.
SBSRAM Pulse Test   : address 0x00000000; length 0x00010000...PASSED.
SDRAM0 Pulse Test   : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 Pulse Test   : address 0x03000000; length 0x00100000...PASSED.

***** Peripheral Confidence Tests.
CPU Clock                : Osc. B's frequency multiplied by four
Endian Mode              : Little Endian

Loading little endian peripheral confidence test..............PASSED.
Interrupt Tests...............................................PASSED.
Device Tests..................................................PASSED.
Timer Tests...................................................PASSED.
LED Tests...PASSED.
DMA Tests.....................................................PASSED.
McBSP Loopback Test...........................................PASSED.
NVRAM Checksum..(12AB)........................................PASSED.

***** Codec Confidence Tests.
CPU Clock                : Osc. A's frequency multiplied by four
Endian Mode              : Little Endian
```

```
Loading little endian codec confidence test...................PASSED.
Codec Left Channel Tone Test................................PASSED.
Codec Right Channel Tone Test...............................PASSED.
Codec Left/Right Channel Tone Test..........................PASSED.
Codec Line In Loopback Test.................................PASSED.
Codec Mic In Loopback Test..................................PASSED.
Resetting Board.
```

TMS320C6x EVM Host Utilities

This chapter describes and tells you how to use the following TMS320C6x EVM host command-line utilities:

❑ EVM COFF loader utility
❑ EVM confidence test utility
❑ EVM board configuration utility

4.1   EVM COFF Loader

The EVM COFF loader command-line utility provides a means of displaying and/or loading and executing a 'C6x application program image stored in a preexisting COFF file. These functions are accomplished via the PCI bus and the 'C6x host port interface (HPI), without the need for an emulator and without the use of the JTAG port.

To invoke the EVM COFF loader utility, use this syntax:

evm6xldr   *filename* [*options*]

evm6xldr	Command that invokes the EVM COFF loader utility
filename	Name of the COFF file to be processed and loaded on the EVM board. You must supply a filename.
options	Options that affect the way the EVM COFF loader utility behaves. Options are not case sensitive and can appear anywhere on the command line following the command. The options are listed and described in Table 4–1. Each option is disabled by default.

Table 4–1. EVM COFF Loader Options

Option	Effect
–?	Displays command-line help information; all other options and parameters are ignored when you specify the –? option.
–c	Clears the bss section by setting all bss section variables to 0
–d	Displays all COFF section data in ASCII via the standard output stream. This can be a large amount of data. The –d option enables the –v (verbose) option.
–m *map*	Selects the DSP's memory map. If the *map* parameter is 0, memory map 0 (MAP 0) is selected. If the *map* parameter is 1, memory map 1 (MAP 1) is selected. The default value is 1 (MAP 1).
–p *port*	Selects a specific EVM target board; *port* is a zero-based relative board index, ranging from 0 (for 1 board) to n–1, where n EVM boards are installed. The default is 0.
–q	Suppresses output to the display (quiet option)
–r *reset*	Controls the EVM board reset. If the *reset* parameter is 0, the board is not reset. If the *reset* parameter is 1, the board is reset. The default is 1 (reset board).

Table 4–1. EVM COFF Loader Options (Continued)

Option	Effect
–s	Shows only; writes data to the standard output stream but does not write any data to DSP memory. The –s option enables the –v (verbose) option.
–v	Enables verbose mode, in which basic COFF section information is displayed via the standard output stream
–z	Displays loader status and waits for key press after each step

The loader generally performs the following steps:

1) If the –s (show) option is specified, the loader does not actually load an image but only reads the COFF file, sends output to stdout, and then stops. The information displayed depends on whether the –d (dump) option is selected:

■ If the –d option is not selected, the information sent to stdout consists of a descriptive line for each section, including name, size, and flags.

■ If the –d option is selected, all of the data in each section is displayed. This can be a very large amount of data and can take a long while to display.

If the –s option is not specified, the loader continues to step 2.

2) Opens a driver connection to a specific EVM target board

3) Performs a board reset (unless the –r0 option is selected)

4) Performs an HPI boot mode DSP reset, which permits HPI operation while holding the DSP in a halted state. The memory map mode is set by the –m0 or –m1 option.

5) Opens an HPI connection

6) Uses the HPI to load an executable image from a 'C6x COFF file. Optionally, clears the bss section (if the –c option is specified)

7) Closes the HPI connection

8) Releases the DSP from the halted reset state, thus starting program execution

9) Closes the driver connection to the EVM target board

An example load file, examples.out, is provided for rudimentary demonstration of loader operation. To load and execute it, simply enter the following command:

```
evm6xldr examples.out
```

4.2 EVM Confidence Test

The confidence test is a command-line utility that allows you to confirm the proper installation of the EVM board and drivers and performs several quick tests of the onboard hardware. A successful completion of this test assures you that the board is installed and working properly. If a hardware error is reported, see Appendix A, *Troubleshooting*.

This test must be executed from the directory into which it is installed. Several support files are required for the test operations. This directory is the conftest directory within the EVM installation directory.

To invoke the EVM confidence test, use this syntax:

evm6xtst   [*options*] [*log_filename*]

evm6xtst	Command that invokes the EVM confidence test
options	Options that affect the way the EVM confidence test utility behaves. Options are not case sensitive and can appear anywhere on the command line following the command. The options are listed and described in Table 4–2. Each option is disabled by default.
log_filename	Filename used for the log file. If no filename is supplied, a log file is not created.

Table 4–2. EVM Confidence Test Options

Option	Effect
–?	Displays command-line help information; all other options and parameters are ignored when you specify the –? option.
–p *port*	Specifies the board to be tested; *port* is the relative index of the board, ranging from 0 (for 1 board) to n–1, where n EVM boards are installed. The default is 0.
–i	Enables the information-only mode; in this mode, only the board configuration information is displayed, bypassing the board tests

The confidence test checks for proper installation of the Win32 DLL and low-level driver by opening a connection to the board. With successful access to the board, the board's configuration information is retrieved and displayed to the command window. If the information-only mode is requested (using the –i option), the program terminates at this time.

The configuration information includes:

❏ Board index
❏ Board type and revision
❏ Board clock
❏ Endian mode
❏ JTAG selection
❏ User-defined options
❏ CPLD revision

The target board being tested is reset and configured into SBSRAM clock mode and big-endian mode. While the board is configured in this manner, a 'C6x memory test module is loaded to the 'C6x and executed. This verifies proper operation of the host port interface (HPI), which is used to load the 'C6x executable code, and the onboard memory while in SBSRAM clock mode and big-endian mode.

The board is then reset and configured into the standard setup of NORMAL clock mode and little-endian mode. While the board is in the standard configuration, a sequence of tests are loaded to the 'C6x and executed. These tests include:

❏ Interrupt tests
❏ Device tests
❏ Timer tests
❏ LED tests
❏ DMA tests
❏ McBSP tests
❏ NVRAM checksum test

After these tests are finished, the board is reset and configured into SBSRAM clock mode and little-endian mode. While the board is in this configuration, a sequence of audio codec tests are performed. You can set up for these tests by connecting powered speakers to the LINE OUT connector, a microphone to the MIC IN connector, and a source such as the output from a CD-ROM drive playing an audio CD to the LINE IN connector. These connections, however, are not required for the successful completion of the following tests:

❏ Left channel tone test
❏ Right channel tone test
❏ Left/right channel tone test
❏ Line in loopback test
❏ Mic in loopback test

When the audio codec tests are finished, the board is reset and is allowed to remain in the configuration set by the onboard DIP switches.

All of the configuration and test information is written to stdout and optionally recorded to a file specified on the command line.

4.3 EVM Board Configuration

The EVM 'C6x board configuration utility resets the EVM board and, optionally, configures the board's clock speed, endian mode, memory map, JTAG selection, and user bits. If none of the configuration parameters are used, the board configuration is taken from the hardware DIP switches. With evm6xrst and options selected, you can override the hardware DIP switch configuration of a board.

If you do not specify any of the clock speed, endian mode, memory map, JTAG selection, or user configuration parameters, the utility performs a board reset and the board configuration is determined by the EVM's hardware DIP switches.

If you specify any of the clock speed, endian mode, memory map, JTAG selection, or user configuration parameters, the utility modifies the board configuration and the EVM's hardware DIP switches are overridden. All unspecified configuration parameters are set to the defaults described in Table 4–3.

To invoke the EVM 'C6x board configuration utility, use this syntax:

evm6xrst   [*options*]

evm6xrst	Command that invokes the EVM 'C6x board configuration utility
options	Options that affect the way the EVM board configuration utility behaves. Options are not case sensitive and can appear anywhere on the command line following the command. The options are listed and described in Table 4–3. Each option is disabled by default.

Table 4–3. EVM Board Configuration Options

Option	Effect
–?	Displays command-line help information; all other options and parameters are ignored when you specify the –? option.
–e *endian*	Configures the DSP's endian mode. A value of 0 for the *endian* parameter selects little-endian mode; any other value selects big-endian mode. The default is 0.
–i *init*	Controls the initialization of the EVM's memory located at address 0 with simple loop code to ensure valid code execution upon the release of reset. If the DSP is executing invalid code when the debugger application is invoked, the debugger may not be able to communicate with the DSP. If the *init* parameter is 0, memory initialization is not performed. If the *init* parameter is 1, memory initialization is performed. The default value is 1 (initialize memory).

Table 4–3. EVM Board Configuration Options (Continued)

Option	Effect
–je	Selects the external JTAG connection to the DSP for emulation. This configuration requires that an XDS510 or XDS510WS emulator, along with the TI C source debugger for the XDS510, be used. If this option is not used, the internal JTAG connection is selected by default.
–ji	Selects the internal JTAG connection to the DSP. This configuration uses the EVM's JTAG test bus controller for embedded emulation and requires the use of the 'C6x EVM C source debugger. This is the default selection.
–m *map*	Selects the DSP's memory map. If the *map* parameter is 0, memory map 0 (MAP 0) is selected. If the *map* parameter is 1, memory map 1 (MAP 1) is selected. The default value is 1 (MAP 1).
–p *port*	Specifies the board to be reset and optionally configured; *port* is a zero-based relative board index, ranging from 0 (for 1 board) to n–1, where n EVM boards are installed. The default is 0.
–q	Suppresses output to the display (quiet option)
–r *reset*	Controls the EVM board reset. If the *reset* parameter is 0, the board is not reset. If the *reset* parameter is 1, the board is reset. The default value is 1 (reset board).
–s *speed*	Configures the DSP clock speed based on the *speed* parameter as follows:

speed	Selected DSP Clock Speed
0	53.25 MHz
1	40 MHz
2	133 MHz
3	200 MHz

The default is 2 (133 MHz).

Option	Effect
–u *user*	Configures the user bits; *user* is a value from 0 to 7 that is used to set the three user bits on the EVM board. If the *user* value is greater than 7 or if the user option is not used, the user bits default to the hardware DIP switch settings. (See Table 2–1 on page 2-9 for more information about the user bits.)
–x	Ignores relative board index selected in the D_OPTIONS environment variable
–z	Displays configuration status and waits for key press after each step

Troubleshooting

This chapter lists common error conditions and explains how to correct them.

If you experience difficulty when installing or using the TMS320C6x EVM board, follow these guidelines before seeking technical support:

❑ Read the topic in this chapter that relates to the problem.

❑ Read the portion of this manual that discusses the procedures you followed or tasks you performed before the problem occurred.

The following error conditions may occur during or after installation of the EVM and its software. To correct the problem, take the action specified here for that message or condition. If the problem persists, seek technical support.

At least one service driver failed during system startup

Description   This message indicates one of the following problems:

❏ The drivers were installed incorrectly.

❏ The motherboard peripheral component interface (PCI) configuration is incorrect.

Action   Make sure the EVM is fully seated in the PCI connector. If the EVM is not fully seated, power down the PC and correct the problem. Rerun the setup.exe program on the CD-ROM to re-install the EVM support software.

Make sure that the PCI BIOS autoconfiguration is enabled. (Ask your system administrator.) Enable the PCI BIOS auto-configuration, if necessary, and rerun the setup.exe program on the CD-ROM to reinstall the EVM support software.

Audio codec failure

Description   If the DSP software does not configure the 'C6x EVM hardware as required or communicate with the audio codec properly, the audio codec may not operate. In this situation, no audio is output from the audio codec, and no audio samples are received by the DSP.

This problem should not occur if the included audio codec library is used to control the audio codec, because it configures the hardware and communicates with the codec as required.

Action   First, verify that the McBSP0 multiplexer is configured for a serial connection between the DSP and the audio codec. By default, McBSP0 is connected to the expansion peripheral interface, not the audio codec. Therefore, the SP0SEL bit in the DSP's memory-mapped CPLD CNTL register must be set to 1 to connect the DSP's serial port to the audio codec. When the audio codec is not connected, all reads from the audio codec's parallel bus via the EMIF return 0x80. This is an indication that the audio codec is busy or in its power-down state.

If the DSP's McBSP0 is connected to the audio codec and the audio codec still does not work, verify that the DSP's CE1 memory space control register mapped at address 0x01800004 is configured for a read and write strobe periods of 3 CLKOUT1 cycles. See *TMS320C6201/6701 EVM Technical Reference*, Chapter 1, Table 1-4, for CE1 control register values for the four possible CLKOUT1 speeds.

If the audio codec still does not work, use the audio codec library functions provided with the EVM since they have been fully tested and work at all CPU clock rates.

Could not open handle

Description   If an EVM application displays a message similar to:

**Error: could not open handle to \\.\Evm6xDev0 (2)"

then the EVM board or its Windows driver is not installed correctly.

Action   Make sure that the EVM is fully seated in the PCI connector. If the EVM is not fully seated, power down the PC and correct the problem.

If the EVM is fully seated, reboot the PC and check if the problem is solved. If you move an EVM board from one PCI slot to another or install it into a PCI slot that previously held an EVM board, Windows 95 may not prompt you to reboot the PC, which is required for proper driver operation. By simply rebooting the machine, the driver may be configured properly, and the problem may be solved.

If the problem persists, rerun the setup.exe program on the 'C6x EVM support software CD-ROM to reinstall the EVM support software.

If you continue to receive the error message, make sure that the PCI BIOS autoconfiguration is enabled. Enable the PCI BIOS autoconfigure, if necessary, and rerun the setup program on the CD-ROM to reinstall the EVM support software.

In some systems, it may be necessary to reboot the computer two times after the support software is installed.

Driver not loaded

Description   The most common reason the EVM device driver does not load once the drivers and software files are properly installed is that your PC has an improper motherboard BIOS configuration.

Action   Ask your system administrator to verify the following items:

❏ PCI interrupts are enabled and one is allocated to the EVM.

❏ PCI autoconfiguration is enabled.

❏ The motherboard/chipset supports the *PCI Local Bus Specification Revision 2.1*. Motherboards with chipsets supporting earlier versions do not correctly configure the EVM, and, therefore, the device driver will not load.

If a correction is made, you must rerun the setup.exe program to reinstall the EVM support software.

Interrupt (IRQ) conflict

Description   If a PC will not boot properly after an EVM is installed or an EVM application locks up the PC or causes a Windows exception screen to be displayed, there is probably an interrupt conflict between the EVM and another board in the system. The EVM's plug-and-play functionality provides problem-free interrupt assignments in most cases, but conflicts can arise.

The PC's plug-and-play BIOS automatically determines the appropriate interrupt assignment for each board on the PCI bus. Sometimes this is a difficult task for a computer to do without human intervention. As boards are added and removed, the BIOS can get confused and may make inappropriate decisions that can result in an interrupt being shared between the EVM and another board.

The EVM's support software properly handles shared inter-rupts. However, other boards or their drivers may not be able to share interrupts. This is the root of the IRQ conflict, which is quite common with PCs. PCI boards are supposed to be able to share interrupts, so many times the PC's BIOS assigns multiple devices to the same interrupt, such as IRQ9, even if there are other IRQs available. When a board that does not support shared interrupts is assigned the same IRQ as the EVM, the system does not operate properly. In most cases the PC does not boot at all. This problem is a system-level problem, not a problem with your EVM.

PCI network interface cards (NICs) and video adapters tend to be the source of interrupt conflict problems. Two popular boards that have been found to cause this problem are the 3COM 3C905 Etherlink NIC and the STB Velocity 128 (PCI and AGP) video adapter. STB explicitly states on their support web page that this board cannot share an IRQ.

Action   Unfortunately, there is not one solution that applies to all PCs because there are different types of BIOS and releases of the Windows operating systems. The following actions can be taken to address the problem, but every one may not work for your particular system. It is recommended that you read all the items before you take action so that you understand all of the potential actions that may apply to your situation.

❑ It may be helpful to check your system's current interrupt assignments to determine exactly which device is con-flicting with the EVM's interrupt.

To observe the interrupt assignments, perform the follow-ing steps within the applicable operating system:

Windows 95:

1) From the Windows 95 desktop, right click on the My Computer icon.

2) Select the Properties menu item.

3) Select the Device Manager tab.

4) Double click on the Computer menu item.

5) Select the Interrupt request (IRQ) radio button in the View Resources tab to display the interrupt assign-ments.

Windows NT:

1) Select the Run... menu item from the Windows NT Start menu.

2) At the Open: prompt type **winmsd** and press Enter.

3) Select the Resources tab of the Windows NT Diagnostics window.

4) Click on the IRQ button to display the interrupt assignments.

❏ The easiest way to resolve an interrupt conflict that works in some cases is to simply move the EVM to another PCI slot. The system's plug-and-play BIOS sometimes assigns a different interrupt to the board when you move it. If the new assignment does not conflict with any other boards, the problem is solved.

❏ If moving the EVM to another PCI slot does not solve the problem or there is not another full-size PCI slot available, either remove the other board or have its interrupt physically changed to an unused IRQ, if possible.

❏ Sometimes IRQs are set to Reserved in the system's BIOS setup. If these IRQs are not required to be reserved, it is recommended that they be changed to Available to make IRQs available. This feature may not be available with your particular BIOS.

❏ If all IRQs are already being used in the system, it is recommended that any unused IDE and USB interfaces in the system be disabled in the system BIOS, or in the Windows Device Manager, to free up interrupts. This may allow the BIOS to assign the EVM to another available interrupt.

❏ If your system BIOS allows you to manually assign interrupts to specific PCI slots, the problem can be easily solved in this manner by assigning the EVM's slot to an unused IRQ in the BIOS setup.

❏ Sometimes if you remove and reinstall all of the PCI boards one at a time (rebooting in between each addition), the interrupt conflict may disappear. This procedure allows the system BIOS to start from scratch and reallocate interrupts for each board.

❑ In some cases, a new version of your system BIOS may be available from the BIOS vendor that supports control over IRQ allocation. Check with your BIOS vendor for further information. Most BIOS vendors provide free BIOS upgrades that are available from their support web pages.

❑ If you are running under Windows 95, it is possible to manually set the EVM's IRQ assignment. In the BIOS set-up, enable the PnP OS option if it is available. This allows Windows 95 to control the plug-and-play information and manually assign interrupts in the Device Manager.

If your machine will not boot Windows at all, you can force a boot in Safe mode. This should allow you to boot and get to the Device Manager.

Perform the following steps:

1) From the Windows 95 desktop, right click on the My Computer icon.

2) Select the Properties menu item.

3) Select the Device Manager tab.

4) Double-click on the Other devices item.

5) Double-click on the TI TMS320C6x EVM item.

6) The General tab should be active. Click on the Disable in this hardware profile check box so that it is checked.

Once the system is rebooted, perform the following steps:

1) From the Windows 95 desktop, right click on the My Computer icon.

2) Select the Properties menu item.

3) Select the Device Manager tab.

4) Double-click on the Other devices item.

5) Double-click on the TI TMS320C6x EVM item.

6) Click on the Disable in this hardware profile check box so that it is not checked.

7) Click on the Resources tab.

8) Click on the Use automatic settings check box so that it is not checked.

9) Double click on the Interrupt Request entry in the Resource settings list.

10) At the Value: prompt, select the desired IRQ that does not conflict with other devices, and click on the OK button.

11) Reboot the system.

❑ Because Windows NT 4.0 does not provide plug-and-play support, you are at the mercy of the BIOS interrupt allocation. If your BIOS setup does not provide control over interrupt allocation, you must resort to vendor or BIOS utilities.

■ Check with your computer manufacturer. Some manufacturers, such as Compaq, include or can provide a utility that allows users to reallocate IRQs as needed.

■ Some BIOS vendors also offer utilities that allow users to manually control IRQs. One of them is available from Intel called ICU configuration utility. It can be used to assign unique IRQs to PCI cards. This action should be attempted as a last resort. This utility is available from the following Intel web page:

http://developer.intel.com/design/motherbd/gen_indx.htm

PCI bus mastering failure

Description To use the EVM's PCI bus mastering capability, the EVM must be installed in a PCI slot that supports bus mastering. Not all PCI slots allow bus mastering.

Action Check your system's documentation or contact your computer's manufacturer to make sure that you are installing the EVM into a PCI slot that supports bus mastering.

Source debugger cannot be started successfully

Description If a message similar to the following one displays in a pop-up error window, the C source debugger application cannot be started successfully:

```
Function:  fatal_msg()

 Init error –1
 CANNOT INITIALIZE THE EVM!!
       –  Check I/O configuration

                     [OK]
```

After you click on the [OK] button, the debugger command window displays a red error message as follows:

––– Processor access timeout at 00000000

This condition occurs when the debugger is started when the DSP is running invalid code, which almost always is true the first time the debugger is invoked because the EVM does not have nonvolatile program memory storage. When the DSP executes invalid code, the debugger may not be able to communicate with the DSP via its JTAG interface. If this occurs, the debugger displays the identified error message and exits.

Action If you are using the 'C6x EVM debugger with the EVM board installed in a PCI slot, invoke the evm6xrst command with the –i1 option. This prevents the problem by initializing the DSP's memory at address 0 to known code. The evm6xrst command is described in section 4.3, *EVM Board Configuration,* on page 4-6.

IIf you are using the 'C6x EVM in a stand-alone configuration with an external XDS510 or XDS510WS emulator and the standard TI 'C6x source debugger, press the EVM's manual reset pushbutton after the debugger displays the blue Texas Instruments banner in the command window and before the error message is displayed to successfully start the debugger.

You have insufficient privilege to add or remove a driver

Description If you did not log onto Windows NT as a user with administrative privileges, this message appears.

Action Log onto Windows NT as a user with administrative privileges and then rerun the setup.exe program to reinstall the EVM support software.

Glossary

A

A/D: See *analog-to-digital.*

adaptive differential pulse code modulation (ADPCM): A speech coding method that calculates the difference between two consecutive speech samples and encodes it using an adaptive filter to transmit at a lower rate than the standard 64-kbps pulse code modulation technique.

ADC: See *analog-to-digital converter.*

address: The logical location of program code or data stored in memory.

administrative privileges: Authority to set software and hardware access; includes access and privileges to install, manage, and maintain system and application software and directories on a network server or individual computer systems.

ADPCM: See *adaptive differential pulse code modulation.*

A-Law companding: See *compress and expand (compand).*

ALU: See *arithmetic logic unit.*

American National Standards Institute (ANSI): A standards-setting, non-government organization that develops and publishes standards for voluntary use in the United States.

American Standard Code for Information Interchange (ASCII): A standard computer code for representing and exchanging alphanumeric information.

analog-to-digital (A/D): Conversion of continuously variable electrical signals to discrete or discontinuous electrical signals.

analog-to-digital converter (ADC): A converter with internal sample-and-hold circuitry used to translate an analog signal to a digital signal.

ANSI C: A version of the C programming language that conforms to the C standards defined by the American National Standards Institute.

application programming interface (API):   A set of standard software function calls and data formats that application programs use to interact with other applications, device-specific drivers, or the operating system.

application-specific integrated circuit (ASIC):   A custom chip designed for a specific application. It is designed by integrating standard cells from a library.

arithmetic logic unit (ALU):   The section of the computer that carries out all arithmetic operations (addition, subtraction, multiplication, division, or comparison) and logic functions.

ASCII: See *American Standard Code for Information Interchange*.

ASIC: See *application-specific integrated circuit*.

assembler: A software program that creates a machine language program from a source file that contains assembly language instructions, directives, and macros. The assembler substitutes absolute operation codes for symbolic operation codes, and absolute or relocatable addresses for symbolic addresses.

assert: To make a digital logic device pin active. If the pin is active low, then a low voltage on the pin asserts it. If the pin is active high, then a high voltage asserts it.

B

ball grid array (BGA):   An integrated circuit package in which the input and output connections are solder balls arranged in a grid pattern.

base address register (BAR):   A device configuration register that defines the start address, length, and type of memory space required by a peripheral component interconnect (PCI) device. The value written to this register during device configuration programs its memory decoder to detect accesses within the indicated range.

basic input/output system (BIOS):   A firmware program that is responsible for power-on testing and initialization of a computer. In addition, it may provide runtime services for operating systems.

BBS: See *bulletin board service*.

benchmarking: A type of program execution that allows you to track the number of CPU cycles consumed by a specific section of code.

BGA: See *ball grid array*.

big endian: An addressing protocol in which bytes are numbered from left to right within a word. More significant bytes in a word have lower numbered addresses. Endian ordering is specific to hardware and is determined at reset. See also *little endian.*

BIOS: See *basic input/output system*.

bit: A binary digit, either 0 or 1.

boot: The process of loading a program into program memory.

boot mode: The method of loading a program into program memory. The 'C6x DSP supports booting from external ROM or the host port interface (HPI).

bulletin board service (BBS):   An electronic bulletin board that allows users to post and read messages and download software.

bus master: A device capable of initiating a data transfer with another device.

byte: A sequence of eight adjacent bits operated upon as a unit.

C

CBT: See *crossbar technology*.

CD-ROM: See *compact disc read-only memory*.

central processing unit (CPU):   The CPU is the portion of the processor involved in arithmetic, shifting, and Boolean logic operations, as well as the generation of data- and program-memory addresses. The CPU includes the central arithmetic logic unit (CALU), the multiplier, and the auxiliary register arithmetic unit (ARAU).

clock cycle: A cycle based on the input from the external clock.

clock mode (clock generator):   One of the modes that sets the internal CPU clock frequency to a fraction or multiple of the frequency of the input clock signal CLKIN.

clock modes: Options used by the clock generator to change the internal CPU clock frequency to a fraction or multiple of the frequency of the input clock signal.

CMOS: See *complementary metal oxide semiconductor*.

coder-decoder or compression/decompression (codec): A device that codes in one direction of transmission and decodes in another direction of transmission.

common object file format (COFF): A system of object files configured according to a standard developed by AT&T. These files are relocatable in memory space. The 'C6x code generation tools generate COFF files.

compact disc read-only memory (CD-ROM): A 4.7-inch optical disk that can hold as much as 660M bytes of digital data. A CD-ROM can store digitized audio, image, video, text, and application data.

compiler: A translation program that converts a high-level language set of instructions into a target machine's assembly language.

complementary metal oxide semiconductor (CMOS): An integrated circuit technology that uses complementary transistors to efficiently charge and discharge capacitive loads in both the positive and negative directions and dissipates power only on transitions.

complex programmable logic device (CPLD): A digital, user-configurable integrated circuit used to implement custom logic functions.

compress and expand (compand): A quantization scheme for audio signals in which the input signal is compressed and then, after processing, is reconstructed at the output by expansion. There are two distinct companding schemes—A-law, used in Europe, and μ-law, used in the United States.

CPLD: See *complex programmable logic device*.

crossbar technology (CBT): High-speed bus-connect devices that are useful for bus isolation, multiplexing, and voltage translation. These devices have an on-state resistance of 5 ohms and a propagation delay of 250 ps.

CPU: See *central processing unit*.

D

D/A: See *digital-to-analog.*

DAC: See *digital-to-analog converter*.

daughterboard: A circuit board that connects to a motherboard to provide additional capabilities and/or interfaces. See also *motherboard*.

dB: See *decibels*.

debugger: A software interface used to identify and eliminate mistakes in a program.

decibels (dB): A unit for measuring the level of signal relative to a defined reference signal that follows it. For example, the notation dBm indicates a signal power level relative to a 1 milliwatt reference signal.

device driver: Software that enables computer hardware to communicate with a device. A device driver may also translate data and call other drivers to actually send data to a device.

device ID: Every peripheral component interconnect (PCI) device must have a device ID configuration register to identify itself.

digital signal processor (DSP): A semiconductor that turns analog signals—such as sound or light—into digital signals, which are discrete or discontinuous electrical impulses, so that they can be manipulated.

digital-to-analog (D/A): Conversion of discrete or discontinuous electrical signals to continuously variable signals. See also *digital-to-analog converter*.

digital-to-analog converter (DAC): A device that converts a signal represented by a series of numbers (digital) to a continuously varying signal (analog). See also *digital-to-analog*.

DIP: See *dual in-line package*.

direct memory access (DMA): A mechanism whereby a device other than the host processor contends for, and receives, mastery of the memory bus so that data transfers can take place independent of the host.

DLL: See *dynamic link library*.

DMA: See *direct memory access*.

doubleword (DWORD): The PCI (host) defines a doubleword as a 32-bit value. See also *word* and *halfword*.

driver: See *device driver*.

DSP: See *digital signal processor.*

dual in-line package (DIP): A common rectangular chip housing with leads (pins) on both long sides.

DWORD: See *doubleword*.

dynamic link library (DLL):   A Windows software library that is linked dynamically at run time, rather than statically at compile time. DLLs can be shared among multiple applications and be replaced with newer versions without requiring the applications to be recompiled.

E

EEPROM: See *electrically-erasable programmable read-only memory*.

electret microphone: A condenser microphone that requires an external power source.

electrically-erasable programmable read-only memory (EEPROM):   A nonvolatile memory device that can be programmed in-circuit and have its contents selectively changed. Although its name includes *read-only*, it supports both read and write accesses.

electrostatic discharge (ESD):   Discharge of a static charge on a surface or body through a conductive path to ground, which can be damaging to integrated circuits.

EMIF: See *external memory interface*.

erasable programmable read-only memory (EPROM):   A nonvolatile memory device that can be erased with exposure to ultraviolet light. The device can be randomly accessed, but it is read only.

ESD: See *electrostatic discharge*.

evaluation module (EVM):   A board and software tools that allow the user to evaluate a specific device.

expansion interface: An interface that allows additional capabilities to be added to a base product.

external interrupt: A hardware interrupt triggered by a specific value on a pin.

external memory interface (EMIF):   The boundary between the CPU and external memory through which information is conveyed.

F

first in, first out (FIFO):   A queue; a data structure or hardware buffer from which items are taken out in the same order they were put in. A FIFO is useful for buffering a stream of data between a sender and receiver that are not synchronized; that is, the sender and receiver are not sending and receiving at exactly the same rate. If the rates differ by too much in one direction for too long, the FIFO becomes either full (blocking the sender) or empty (blocking the receiver).

flag: A binary status indicator whose state indicates whether a particular condition has occurred or is in effect.

Flash memory: Nonvolatile read-only memory that is electronically erasable and programmable.

H

halfword: The 'C6x DSP defines a halfword as a16-bit data value. See also *doubleword* and *word*.

handle: An identifier used by software to reference a file or device.

high-level language (HLL):   A general-purpose language that can be used to program a microprocessor rather than using a low-level, machine-dependent language.

host: A device to which other devices (peripherals) are connected and that generally controls those devices.

host port interface (HPI):   A 16-bit parallel interface that the host uses to access the DSP's memory space.

I

identifier (ID):   A field that contains a resource-table index, a sequence number, and a resource-type code; it identifies a kernel resource such as a port, semaphore, or task.

IEEE 1149.1 standard: "IEEE Standard Test Access Port and Boundary-Scan Architecture", first released in 1990. See also *JTAG*.

Industry Standard Architecture (ISA):   An industry-standard 8/16-bit bus used in IBM™ compatible desktops. It provides a theoretical maximum data transfer rate of 8.33M bytes per second.

initiator: When a PCI bus master has arbitrated for and won access to the PCI bus, it becomes the initiator of a transaction.

Institute of Electrical and Electronic Engineers (IEEE):   A publishing and standards-making body focused on advancing the theory and practice of electrical, electronics, computer engineering, and computer science.

in-system programmable (ISP):   The ability to program and reprogram a device on a circuit board.

integrated switching regulator (ISR):   A complete switch-mode power supply in a modular, board-mounted package.

internal interrupt: A hardware interrupt caused by an on-chip peripheral.

interrupt: A signal sent by hardware or software to a processor requesting attention. An interrupt tells the processor to suspend its current operation, save the current task status, and perform a particular set of instructions. Interrupts communicate with the operating system and prioritize tasks to be performed.

interrupt service routine: A module of code that is executed in response to a hardware or software interrupt.

ISP: See *in-system programmable*.

ISR: See *integrated switching regulator*.

J

Joint Test Action Group (JTAG):   The Joint Test Action Group was formed in 1985 to develop economical test methodologies for systems designed around complex integrated circuits and assembled with surface-mount technologies. The group drafted a standard that was subsequently adopted by IEEE as IEEE Standard 1149.1-1990, "IEEE Standard Test Access Port and Boundary-Scan Architecture".

K

kilohertz (kHz):   One thousand hertz, or cycles per second, used to indicate the frequency of a clock signal.

L

latch phase: The phase of a CPU cycle during which internal values are held constant.

light emitting diode (LED):   A semiconductor chip that gives off visible or infrared light when activated.

linker: A software tool that combines object files to a form an object module, which can be loaded into memory and executed.

little endian: An addressing protocol in which bytes are numbered from right to left within a word. More significant bytes in a word have higher numbered addresses. Endian ordering is specific to hardware and is determined at reset. See also *big endian*.

load: To enter data into storage or working registers.

loader: A device that places an executable module into system memory.

M

mA: See *milliamp.*

macro: A sequence of statements or instructions that is represented by a symbolic symbol.

μF: See *microfarad*.

μ-Law companding: See *compress and expand (compand)*.

mailbox: A 32-bit register that provides a simple communication method to pass messages between the host and DSP software. Multiple mailboxes are typically available to support bidirectional, multiword message transfers.

maskable interrupt: An interrupt that can be enabled or disabled through software.

master clock output signal (CLKOUT1):   The output signal of the on-chip clock generator. The CLKOUT1 high pulse signifies the CPU's logic phrase (when internal values are changed), while the CLKOUT1 low pulse signifies the CPU's latch phase (when the values are held constant).

McBSP: See *multichannel buffered serial port.*

megahertz (MHz): One million hertz, or cycles per second, used to indicate the frequency of a clock signal.

memory map: A graphical representation of a computer system's memory, showing the locations of program space, data space, reserved space, and other memory-resident elements.

memory-mapped register: An on-chip register mapped to an address in memory. Some memory-mapped registers are mapped to data memory, and some are mapped to input/output memory.

MHz: See *megahertz*.

microfarad (μF): One-millionth of a farad, which is the basic unit of capacitance.

microsecond (μs): One-millionth of a second.

milliamp (mA): One-thousandth of an ampere, which is the basic unit of current.

millimeter (mm): One-thousandth of a meter.

million instructions per second (MIPS): A unit of instruction execution speed of a computer.

millisecond (ms): One-thousandth of a second.

millivolts root mean square (mV$_{rms}$): One-thousandth of a volt root mean square. See also *volts root mean square.*

MIPS: See *million instructions per second*.

mm: See *millimeter*.

most significant byte (MSbyte): The byte in a multibyte word that has the most influence on the value of a word.

motherboard: The main circuit board that contains the processor, main memory, circuitry, bus controller, connectors, and primary components of the computer. See also *daughterboard*.

μs: See *microsecond*.

ms: See *millisecond*.

MSbyte: See *most significant byte*.

multichannel buffered serial port (McBSP): A standard serial port interface found on 'C6x devices. It provides full-duplex communication, double-buffered data registers, independent transmit and receive framing and clocking, direct interface to industry-standard serial devices, internal and external clock support, and an autobuffering capability using a DMA controller.

multiplexing: A process of transmitting more than one set of signals at a time over a single wire or communications link. (Also known as muxing.)

mutex: A mutual exclusion semaphore used to restrict access to a resource.

mV$_{rms}$:   See *millivolts root mean square*.

N

nanosecond (ns):   One-billionth of a second, the basic unit of time.

nonmaskable interrupt (NMI):   An interrupt that uses the same logic as the maskable interrupts, but can be neither masked nor disabled. It is often used as a soft reset.

nonvolatile random access memory (NVRAM):   A type of random access memory that retains its data when its power source is turned off, providing nonvolatile storage.

O

object file: A file that has been assembled or linked and contains machine language object code.

off chip: A device external to the device.

on chip: An element or module internal to the device.

P

parallel debug manager (PDM):   A program used for creating and controlling multiple debuggers for the purpose of debugging code in a parallel-processing environment.

PC: *Personal computer.*

PCI: See *peripheral component interconnect*.

PCM: See *pulse code modulation*.

PDM: See *parallel debug manager*.

peripheral component interconnect (PCI):   A high-speed local bus that supports data-transfer speeds of up to 132M bytes per second at 33 MHz.

phase-locked loop (PLL):   A unit within a system that uses phase to lock on to a signal to ensure synchronous clocking of digital signals.

pitch: The distance between successive centers of leads of a component package.

plastic quad flat pack (PQFP):   A low-profile, surface-mount integrated circuit package that is plastic and has leads (pins) on all four sides.

PLL: See *phase-locked loop.*

poll: A continuous test used by the program until a desired condition is met.

PQFP: See *plastic quad flat pack*.

profiling environment: A special debugger environment that provides a method for collecting execution statistics about specific areas in application code.

pulse code modulation (PCM):   The most common method of encoding an analog voice signal into digital data. Voice signals are encoded into 8-bit data samples at an 8-kHz sample rate, resulting in a 64-kbps digital data stream.

R

random-access memory (RAM):   A memory element that can be written to as well as read from.

read-only memory (ROM):   A semiconductor storage element containing permanent data that cannot be changed.

ready: A task state indicating that the task either is currently executing or is able to execute as soon as it acquires the processor.

real time: The actual time during which the physical process of computation transpires in order that results of the computation interact with a physical process.

reduced instruction set computer (RISC):   A computer whose instruction set and related decode mechanism are much simpler than those of microprogrammed complex instruction set computers. The result is a higher instruction throughput and a faster real-time interrupt service response from a smaller, cost-effective chip.

register: A small area of high-speed memory, located within a processor or electronic device, that is used for temporarily storing data or instructions. Each register is given a name, contains a few bytes of information, and is referenced by programs.

reset: A means to bring processors to known states by setting registers and control bits to predetermined values and signaling execution to start at a specified address.

ring 0:   Highest level of privilege available on an Intel processor that defines what data can be accessed, what code in memory can be executed, and what machine instructions can be executed by a program. Low-level device drivers run at ring 0. See also *ring 3.*

ring 3:   Lowest level of privilege available on an Intel processor that defines what data can be accessed, what code in memory can be executed, and what machine instructions can be executed by a program. High-level user-mode applications and DLL run at ring 3. See also *ring 0.*

RISC: See *reduced instruction set computer.*

ROM: See *read-only memory.*

S

sample rate: The rate at which the audio codec samples audio data. Usually specified in hertz (samples per second).

SBSRAM: See *synchronous burst static random-access memor*y.

SDRAM: See *synchronous dynamic random-access memory.*

slave: Another name for the target being addressed during a PCI transaction.

static random-access memory (SRAM):   Fast memory that does not require refreshing, as DRAM does. It is more expensive than DRAM, though, and is not available in as high a density as DRAM.

structure: A collection of one or more variables grouped together under a single name.

surface-mount technology: A method of assembling printed wiring boards where components are mounted onto the surface rather than through holes.

synchronous burst static random-access memory (SBSRAM):   High-performance SRAM device with accesses that are synchronized to a microprocessor clock and includes a burst address counter.

synchronous dynamic random-access memory (SDRAM):   High-performance DRAM device with accesses that are synchronized to a microprocessor clock and support for page bursts.

syntax: The grammatical and structural rules of a language. All higher-level programming languages possess a formal syntax.

T

target: When related to PCI, it is the device that is the target of a PCI transaction initiated by a PCI bus master. When related to the debugger, the DSP is the target of an emulation access.

target memory: Physical memory in a device into which executable object code is loaded.

test bus controller (TBC):   Application-specific integrated circuit that controls an IEEE 1149.1–1990 (JTAG) serial-test bus to support production testing and in-system microprocessor emulation. The TBC provides control of the DSP and access to all of its registers and memory.

thin quad flat pack (TQFP):   A very low-profile, surface-mount integrated circuit package that is plastic and has leads (pins) on all four sides.

thread of execution: A schedulable unit of execution in a multitasking system. The term refers specifically to the progressive execution of a program element; it excludes other attributes, such as the system resources allocated to a task or process.

timer: A programmable peripheral used to generate pulses or to time events.

TQFP: See *thin quad flat pack.*

transistor-transistor logic (TTL):   A family of logic devices that are made with bipolar junction transistors and resistors. A TTL low level is defined as a voltage level below 0.4 volts. A TTL high level is defined as a voltage level above 2.4 volts.

V

V: See *volt.*

V_{dc}:   See *volts direct current.*

VelociTI: Architecture developed by Texas Instruments that features very long instruction words.

vendor ID: Every PCI device must have a vendor ID configuration register that identifies the vendor of the device.

very long instruction word (VLIW):   Architecture using words between the sizes of 256 bits and 1024 bits.

virtual device driver (VxD):   A 32-bit, ring-0 module that virtualizes hardware for ring-3 modules to provide a primary interface to hardware or specialized software services.

VLIW: See *very long instruction word*.

volt (V):   The unit of voltage, or potential difference.

volts direct current (V_{dc}):   The voltage measurement of a direct current signal.

volts peak-to-peak (V_{pp}):   A measurement of a signal that indicates the difference between its maximum and minimum voltage values.

volts root mean square (V_{rms}):   A measurement of a periodic signal that indicates the effective voltage at which a direct current voltage would deliver the same average power. This measurement provides a method for comparing the power delivered by different waveforms.

VxD: See *virtual device driver*.

W

Win32:   The 32-bit application programming interface for both Windows 95 and Windows NT.

word: A character or bit string considered as an entity. The length of the word is machine-dependent. The 'C6x DSP defines a word as a 16-bit data value. The PCI (host) defines a word as a 16-bit data value. See also *doubleword* and *halfword*.

X

XDS510:   A hardware emulator that provides a scan-path connection to a DSP for source code debugging.

Index