TMS320C62x Multichannel Evaluation Module User's Guide

Literature Number: SPRU285A
February 1999

Printed on Recycled Paper

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Preface

Read This First

About This Manual

This manual tells you how to install and operate the TMS320C62x (C62x) multichannel evaluation module (McEVM). It also provides support software documentation, application programming interface (API) references, and technical reference material for the C62x McEVM.

The C62x McEVM is a peripheral component interconnect (PCI) plug-in card that is compliant with the *PCI Local Bus Specification Revision 2.1*. The C62x McEVM helps you evaluate characteristics of the C62x digital signal processor (DSP) to determine if it meets your application requirements. It is a high-performance platform targeted for multichannel telecom and datacom applications. The C62x McEVM 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). The power supply and emulator are not included in the kit.

C62x McEVM support software and APIs enable you to use the board to create applications for the C62x. Software utilities are provided with the C62x McEVM for board diagnostics, board configuration, and common object file format (COFF) DSP application loading. McEVM schematics and logic equations are included in the companion TMS320C62x McEVM Technical Reference (SPRU308) manual to ease your hardware development efforts and reduce time to market.

This manual assumes that you understand general and technical PC terminology. This manual specifically addresses the installation and operation of the C62x McEVM and its support software. Detailed information about the C62x 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 C62x McEVM, as well as related products, visit the C6000 website at **http://www.ti.com/sc/docs/dsps/tools/c6000/index.htm.**

How to Use This Manual

This reference guide provides the following types of information about the C62x McEVM:

- **Introductory information**, Chapter 1, provides a high-level overview of the C62x McEVM hardware and support software and describes how the product can be used during code development and debugging.
- **Installation information**, Chapters 2 and 3, provides hands-on information to help you get started using the C62x McEVM.
  - Chapter 2 identifies the system requirements and C62x McEVM kit contents. It also provides step-by-step hardware and software installation procedures.
  - Chapter 3 provides instructions for running the McEVM confidence test.
  - Chapter 4 describes the C62x McEVM 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 example, GD[31:0] represents the global data bus pins on a device.

- The TMS320C62x family of devices is referred to as the C62x. The following abbreviations are used in this manual for TI devices on the C62x McEVM:

Abbreviation	Device Definition
C6201	TMS320C6201
ALVCH16244	SN74ALVCH16244
CBT3257	SN74CBT3257
CBTD3384	SN74CBTD3384
LVT125	SN74LVT125
ALVCH16245	SN74ALVCH16245

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 you.

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 C62x 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.

TMS320C6000 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 C6000 generation of devices.

TMS320C6000 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 C6000 generation of devices.

TMS320C6000 Optimizing C Compiler User's Guide (literature number SPRU187) describes the C6000 C compiler and the assembly optimizer. This C compiler accepts ANSI standard C source code and produces assembly language source code for the C6000 generation of devices. The assembly optimizer helps you optimize your assembly code.

TMS320C6000 CPU and Instruction Set Reference Guide (literature number SPRU189) describes the C6000 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.

TMS320C6000 Programmer's Guide (literature number SPRU198) describes ways to optimize C and assembly code for the TMS320C6000 DSPs and includes application program examples.

TMS320C6000 Technical Brief (literature number SPRU197) gives an introduction to the C6000 platform of 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.

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.

TMS320C6201/6701 Evaluation Module Reference Guide (literature number SPRU269C) provides instructions for installing and operating the C6x evaluation module. It also includes support software documentation, application programming interfaces, and technical reference material.

TMS320C6201, TMS320C6201B Digital Signal Processors Data Sheet (literature number SPRS051) describes the features of the TMS320C6201 and TMS320C6201B fixed-point DSPs and provides pinouts, electrical specifications, and timings for the devices.

TMS320C62x McEVM Technical Reference, (literature number SPRU308) can be accessed through the following URL:

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

Related Documentation

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

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

http://www.pcisig.com/specs.html

MVIP-90 Reference Manual, GO-MVIP, Inc. 1990
http://www.mvip.org/MemOrder.htm

The MVIP Book, GO-MVIP, Inc., ISBN 0-936648-76-7.

Trademarks

Obtaining Technical Support

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

- ❑ Assembly number of your C62x McEVM (D600860-0001) located on the bottom side of the board
- ❑ Revision number of your C62x McEVM 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 C62x McEVM 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 you are using
- ❑ Version of the debugger you are using
- ❑ If you are using Windows 95/98, 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.
- ❑ If you are using Windows NT, perform the following steps to get system information:
  1) Select the Run... menu item from the Windows NT Start menu.
  2) Type **winmsd** at the Open prompt, and press Enter.
  3) Select the Resources tab of the Windows NT Diagnostics window.
  4) Click on the Print button to get a report.
  5) Click on OK at the Create Report window.

  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 TMS320C62x McEVM 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.

Contents

Figures

Tables

Notes, Cautions, and Warnings

Chapter 1

Introduction

This chapter lists the key features of the TMS320C62x multichannel evaluation module (McEVM), provides functional overviews of the McEVM hardware and software, and describes the McEVM operating scenarios.

The C62x McEVM is a high-performance, multichannel telephony platform for the development, analysis, and testing of C62x digital signal processor (DSP) algorithms and applications. The C62x McEVM allows you to evaluate the C62x DSP and algorithms to determine if they meet your application requirements. The C62x McEVM hardware design information and software application programming interfaces (APIs) also provide a reference design that facilitates your own C62x-based hardware and software development.

The McEVM is bundled with TMS320C62x source debugger tools, Windows 95/98™ 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 C62x DSP's performance and develop custom applications.

Topic **Page**

1.1 Key Features of the TMS320C62x McEVM

The C62x McEVM has the following key features:

- ❑ 200-MHz C6201 DSP capable of executing 1600 million instructions per second (MIPS)
- ❑ Quad clock support (33.25 MHz, 50 MHz, 133 MHz, and 200 MHz) (see Table 1–1)

Table 1–1. Quad Clock Support Frequencies

	OSC A		OSC B	
	x1	x4	x1	x4
C6201 McEVM	33.25 MHz	133 MHz	50 MHz	200 MHz

- ❑ Peripheral component interconnect (PCI) interface with initiator/target (master/slave) support
- ❑ 512K 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 and external XDS510 support
- ❑ Host port interface (HPI) access to all DSP memory via the PCI bus
- ❑ Enhanced Multi-Vendor Integration Protocol (MVIP)-90 telephony bus
- ❑ Nonblocking 384 $\times$ 384 channel switching
- ❑ T1/E1 interface
- ❑ Handset interface that supports μ-Law and A-Law companding
- ❑ Onboard switching voltage regulators for 1.8 volts direct current (V_{DC}) and 3.3 V_{DC}
- ❑ Seven light emitting diode (LED) indicators (one power-on indicator, two user-defined indicators, and four T1/E1 activity indicators)
- ❑ Internal PCI operation
- ❑ External desktop operation (requires external power supply and XDS510 emulator, which are not included in the C62x McEVM kit)
- ❑ Expansion memory and DSP peripheral connectors for daughterboard support
- ❑ Plug-and-play PCI device

1.1.1 User Controls and Indicators

The C62x McEVM has the following user controls and indicators:

- ❑ JTAG emulation (embedded or external)
- ❑ DSP oscillator clock selection (33.25 MHz or 50 MHz)
- ❑ Clock mode (multiply-by-1 or multiply-by-4 for 33.25-MHz, 50-MHz, 133-MHz, or 200-MHz DSP CLKOUT1)
- ❑ DSP boot mode selection:
  - ■ None (no-boot mode)
  - ■ HPI boot mode
  - ■ Read-only memory (ROM) boot mode (requires optional daughterboard)
- ❑ Little- or big-endian memory addressing
- ❑ Twelve dual in-line package (DIP) switches
- ❑ Manual reset
- ❑ Power-on LED indicator
- ❑ Two user-defined LED indicators
- ❑ Four T1/E1 activity LED indicators
- ❑ MVIP C2 and /C4 clock termination
- ❑ Daughterboard control of MVIP /C4 and /F0 signals

1.1.2 External Interfaces

The C62x McEVM 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 (5 V) for cooling fan (not used)
- ❑ 14-pin external JTAG header
- ❑ 10-pin complex programmable logic device (CPLD) in-system programmable (ISP) header (not installed)

- 40-pin enhanced MVIP-90 connector
- Two 3.5mm audio jacks for handset microphone and earphone
- RJ-48C modular T1/E1 interface connector
- Two low-profile 80-pin (.050-inch) daughterboard connectors (EMIF and DSP peripheral expansion)

1.2 The TMS320C62x McEVM Board

The C62x McEVM 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 C62x McEVM is intended for use in a PCI expansion slot on your computer's motherboard. The C62x McEVM can be operated stand-alone on a desktop with the use of an external power supply and XDS510 emulator. (The power supply and the emulator required for stand-alone operation are not included in the kit.)

Note:

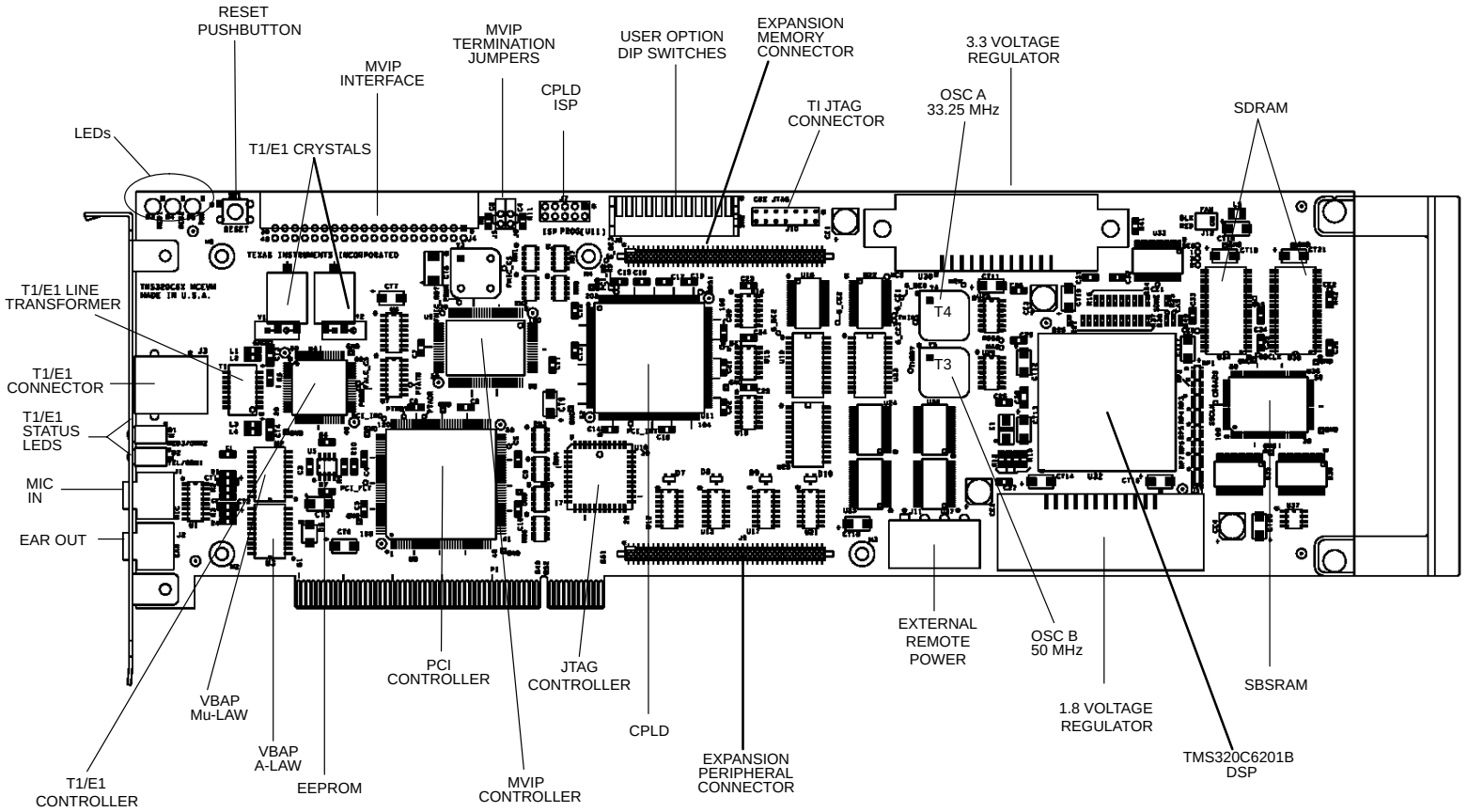
If you are using the C62x McEVM in a PCI expansion slot, make sure there is an unobstructed slot open for the full-length PCI card. See section 2.2.2, *Avoiding Obstructions to the McEVM Board*, for more information.

The C62x McEVM has a C6201 DSP onboard that allows full-speed verification of C62x code with Code Composer. The C62x McEVM provides a PCI interface, SBSRAM and SDRAM, MVIP switching, T1/E1 interface, μ-Law/A-Law handset interface, and embedded JTAG emulation support. Connectors on the C62x McEVM provide DSP external memory interface (EMIF) and peripheral signals that enable its functionality to be expanded with custom or third-party daughterboards.

The McEVM provides a C62x hardware reference design that can assist you in the development of your own C62x-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 C62x McEVM. The figures identify the major components on the C62x McEVM board.

Figure 1–1. TMS320C62x McEVM Layout (Top Side)



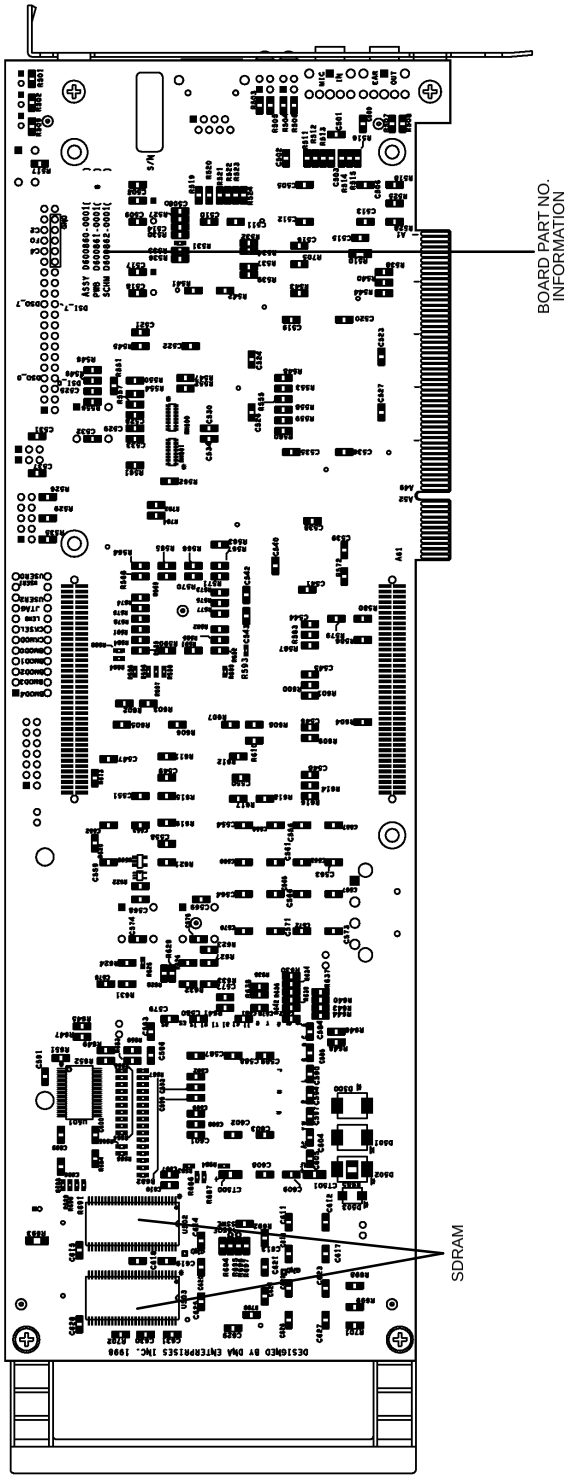
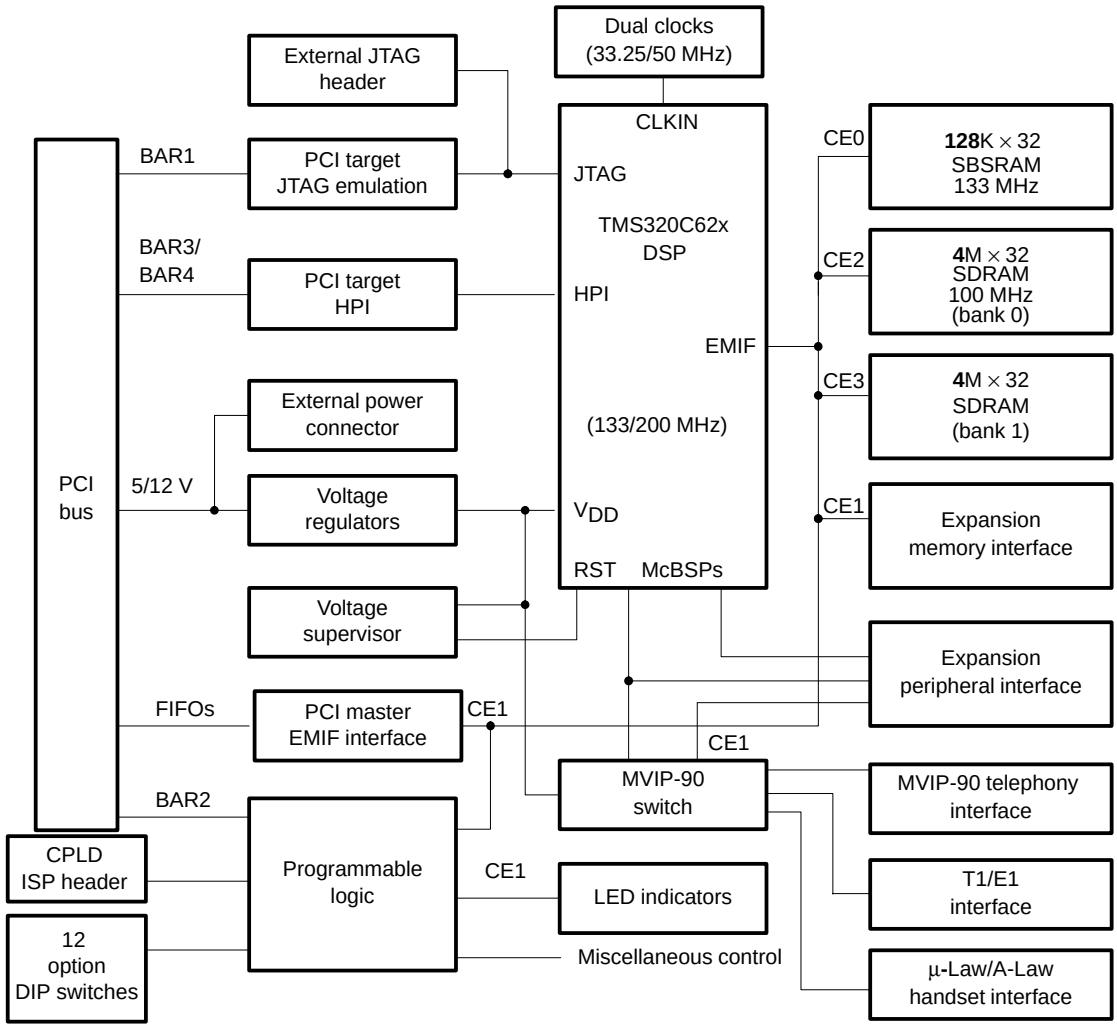


Figure 1–2. TMS320C62x McEVM Layout (Bottom Side)

1.3 TMS320C62x McEVM Hardware Functional Overview

Figure 1–3 shows the basic functional block diagram and interfaces of the C62x McEVM.

Figure 1–3. TMS320C62x McEVM Block Diagram



The C62x McEVM hardware can be divided into 14 functional areas. This section provides an overview of each of these areas. Detailed descriptions of the McEVM hardware are provided in Chapter 1, *TMS320C62x McEVM Technical Reference*.

- **DSP.** The C62x McEVM is built around the C6201 DSP, which operates up to 1600 MIPS with a CPU clock rate of 200 MHz. For more information, see *TMS320C62x McEVM Technical Reference*, Chapter 1, section 1.2, *TMS320C62x DSP.*

- **DSP clocks**. The C62x McEVM supports operation with two different onboard clock sources (33.25 MHz and 50 MHz) and two clock modes (multiply-by-1 and multiply-by-4). As a result, the DSP can operate at four different clock rates, including 33.25 MHz, 50 MHz, 133 MHz, or 200 MHz. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.3*, DSP Clocks.*

- **External memory**. The C62x McEVM provides one bank of 128K $\times$ 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 *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.4*, External Memory.*

- **Expansion interfaces**. The C62x McEVM 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 *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.5*, Expansion Interfaces.*

- **PCI interface**. The C62x McEVM 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 C62x McEVM's PCI interface allows source debugging with the C62x McEVM without requiring an XDS510 emulator, as well as host software access to all of the DSP memory space via the PCI bus. The C6201 DSP can also master the PCI bus to transfer data to and from the host memory. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.6*, PCI Interface.*

- **JTAG emulation**. The C62x McEVM 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 *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.7, *JTAG Emulation.*

- **Programmable logic**. The C62x McEVM's CPLD provides the board's glue-logic and control/status registers. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.8, *Programmable Logic.*

- **MVIP interface and MVIP switch**. The C62x McEVM includes an enhanced MVIP-90 interface that allows interoperation with a wide range of third-party telephony boards. The MVIP interface consists of a 40-pin connector located at the top of the McEVM board. The flexible MVIP interface circuit (FMIC) provides 384 × 384 channel nonblocking connectivity between all telephony devices on the board and the DSP. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.9, *MVIP Interface and MVIP Switch.*

- **T1/E1 interface.** The McEVM provides a common T1/E1 interface that allows it to connect to T1, E1, or Integrated Services Digital Network (ISDN) primary rate trunks operating at 1.544 megabits per second (Mbps) or 2.048 Mbps. The T1/E1 interface provides a multichannel, digital telephone interface for the C6201 DSP. The T1/E1 output is presented as an RJ-48C twisted-pair modular jack on the board's mounting bracket. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.10, *T1/E1 Interface*.

T1/E1 Interface Compatibility

This interface is electrically compatible with T1, E1, and ISDN services provided by the phone company. It is NOT certified or approved for direct connections.

- **μ-Law/A-Law telephone handset interface.** The McEVM includes a handset interface that allows voice data to be processed by the DSP or routed to the T1/E1 and MVIP interfaces. Two TI voice-band audio processor (VBAP) devices are used to provide μ-Law and A-Law companding. The VBAPs are single-supply devices with glueless interfaces to an electret microphone, earphone, and the MVIP switch. The telephone handset interface is brought out to two 3.5mm audio jacks. One jack provides a mono microphone input and the other jack provides a mono earphone output. The two audio jacks are located on the board's mounting bracket. For

more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.11, *Handset Interface*.

- **Power supplies**. The C62x McEVM uses voltage regulators to provide 1.8 V for the C6201 core; 3.3 V for the C6201 I/O, memories, CPLD, and buffers. The PCI bus or external power connector's 5 V is used for all other digital components. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.12, *Power Supplies.*

- **Voltage supervision and reset control**. The C62x McEVM 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, *TMS320C62x McEVM Technical Reference,* Chapter 1, see section 1.13, *Voltage Supervision.*

- **User options**. The C62x McEVM 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. Option straps are also provided to select MVIP clock termination and daughterboard control of MVIP framesync and clock signals. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.14, *User Options.*

- **LED indicators**. The C62x McEVM provides seven LED indicators. A single green LED is illuminated whenever 5 V is applied to the board. Two red LEDs, located at the top of the board, can be used for user-defined status. Four LEDs, located on the mounting bracket, provide T1/E1 status indications. Of the four LEDs, two are green, one is red, and one is yellow. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 1, section 1.15, *Indicators.*

1.4 TMS320C62x McEVM Software Functional Overview

The C62x McEVM software consists of *host support software* and *DSP support software*.

The host support software supplied with the C62x McEVM board includes the following Win32 host utilities and libraries. The host utilities and host libraries run on an Intel™ PC under either Windows 95/98 or Windows NT 4.0.

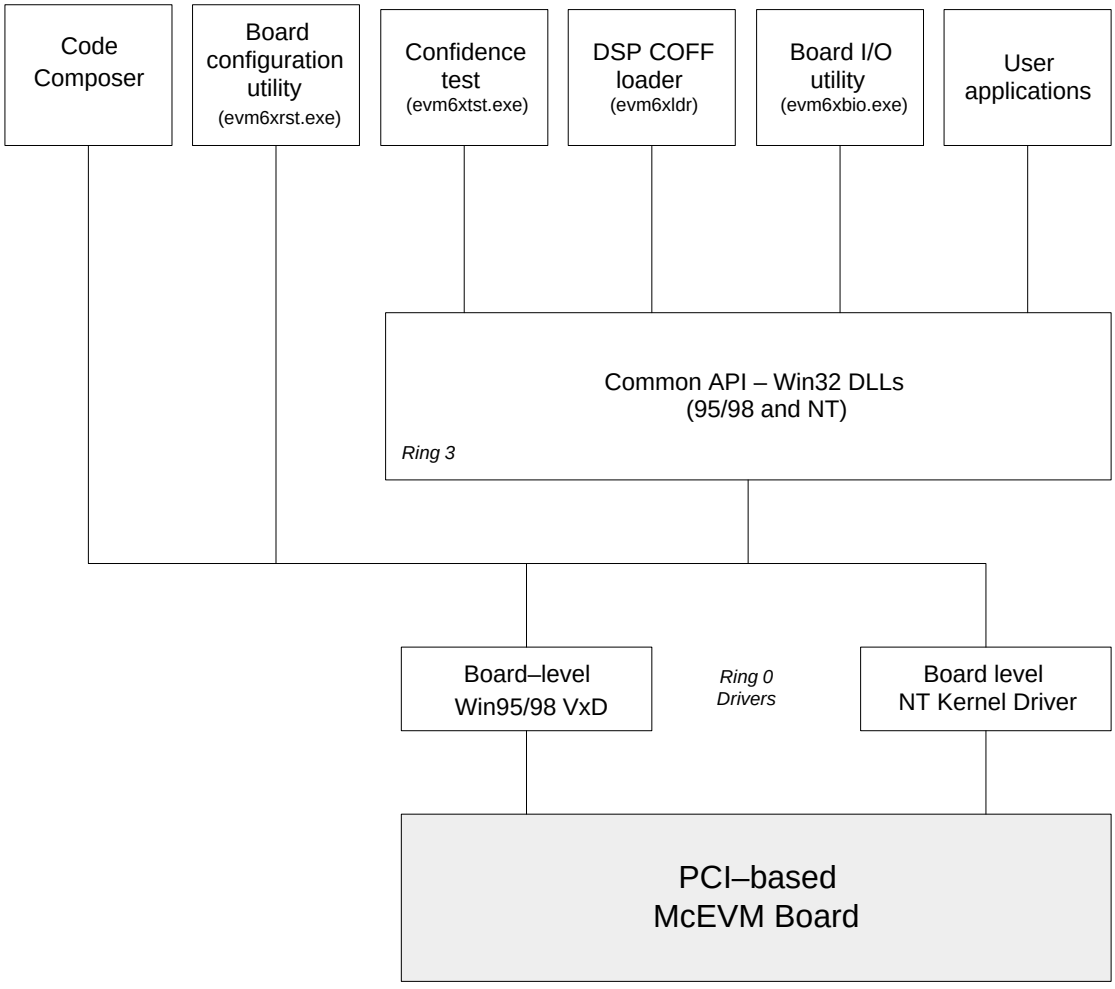
- **Code Composer.** Code Composer is the software debugger that is used to debug C6x software on the board. For more information, see SPRU296, *Code Composer User's Guide.*
- **Board configuration utility (evm6xrst.exe).** This utility is used to reset and configure the board. For more information, see section 4.3, *McEVM Board Configuration*.
- **McEVM confidence test utility (evm6xtst.exe).** This utility tests the basic operation of the board. For details about the McEVM confidence test, see section 4.2, *McEVM Confidence Test*. To use the McEVM confidence test to verify the hardware and software installation, see Chapter 3, *Running the Board Confidence Test*.
- **C62x COFF loader utility (evm6xldr.exe).** This utility is used to load and execute C62x software on the board. For more information, see section 4.1, *McEVM COFF Loader*.
- **McEVM board I/O utility (evm6xbio.exe)**. This utility allows a user to configure all EVM and McEVM I/O devices. Simple script files or direct command line options are used to configure the board without having to write any code. For more information, see section 4.4, EVM/McEVM Board I/O Control Utility.
- **McEVM Win32 DLL (evm6x.dll).** The Win32 host libraries consist of a Windows 95/98 and a Windows NT version of evm6x.dll, which provides user software access for control and communication with the McEVM board. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 2, *TMS320C62x McEVM 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 C62x McEVM. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 2, section 2.4, *McEVM Host Support Software*.

Figure 1–4 provides a block diagram of the McEVM host software components and their relationships. The left side of the figure shows Code Composer. Code Composer makes direct calls to the low-level (ring 0) drivers to access the JTAG TBC device on the McEVM board. The board configuration utility also

makes direct calls to the drivers to access the board option registers on the McEVM 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 95/98 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. TMS320C62x McEVM Host Support Software Block Diagram



The DSP support software supplied with the McEVM 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 *TMS320C62x McEVM Technical Reference*, Chapter 3, section 3.3, McBSP Driver API.

- **MVIP library (fmic.c, mt90810.c, fmic.h and mt90810.h).** This library is a collection of routines that configure and control the operation of the MT90810 flexible MVIP interface circuit (FMIC) device. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 3, section 3.4, *FMIC Support Library API*.

- **Board support library (board.c and board.h).** This library provides C62x board-specific routines for McEVM configuration and control. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 3, section 3.5, *Board Support Library API*.

- **T1/E1 framer library. (t1e1.c, falc.c, t1e1.h and falc_1h.h).** This library is a collection of routines that configure and control the operation of the integrated T1/E1 framer and line interface unit (LIU) Bt8370 device. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 3, section 3.6, *T1/E1 Framer Library*.

- **VBAP library (vbap.c and vbap.h).** This library is a collection of routines that configure and control the operation of the TCM320AC36 and TCM320AC37 VBAP devices. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 3, section 3.7, *VBAP Driver Library API*.

- **PCI library (pci.c and pci.h).** This library is a collection of routines that configure and control the operation of the AMCC S5933 PCI controller. It provides functions for passing data between the host and DSP using mailboxes and the FIFO, as well as for assessing the board's serial EEPROM. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 3, section 3.8, *PCI/AMCC Driver Library API*.

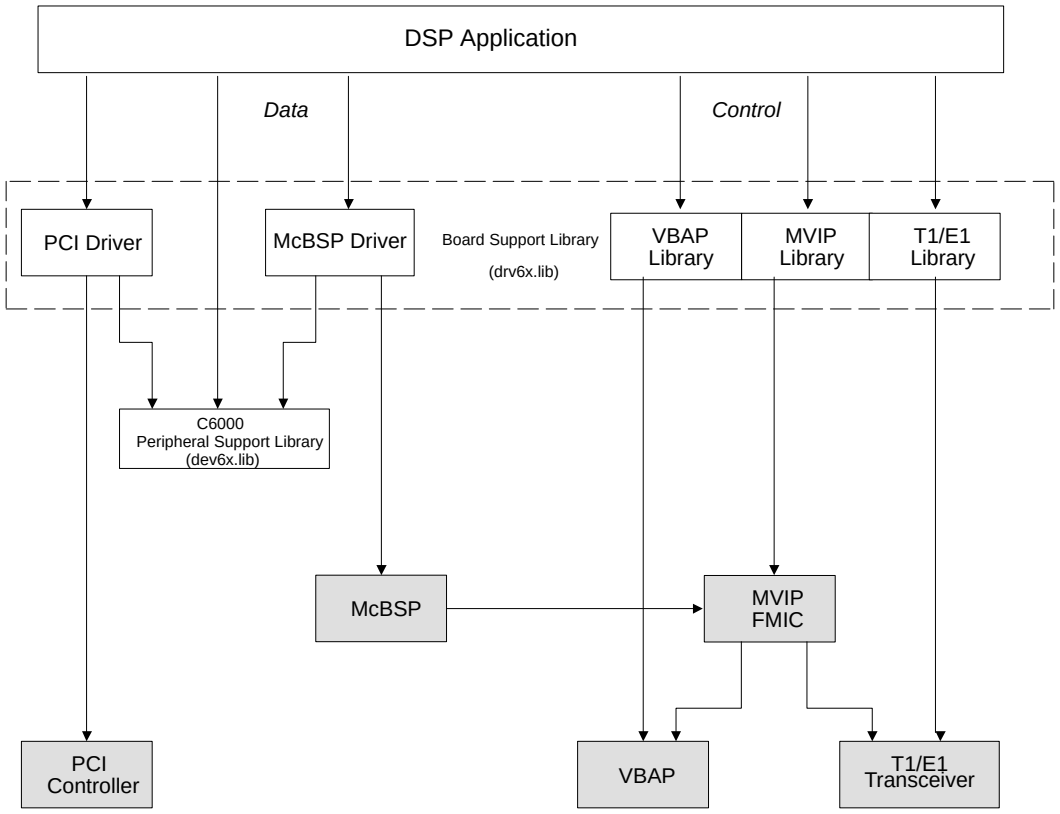
- **C I/O interface library (cio_fifo.c and cio_fifo.h).** This library is a collection of routines that provide support for C I/O by interfacing the TI C compiler's C I/O support to the PCI driver. For more information, see

TMS320C62x McEVM Technical Reference, Chapter 3, section 3.9, *C I/O Interface 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 *TMS320C62x McEVM Technical Reference,* Chapter 3, section 3.10 , *DSP Support Software Examples*.

Figure 1–5 provides a block diagram of the McEVM C62x software components and their relationships. For more information, see *TMS320C62x McEVM Technical Reference,* Chapter 3, *DSP Support Software*.

Figure 1–5. TMS320C62x McEVM DSP Support Software Block Diagram



1.5 Operating Scenarios

The C62x McEVM can operate in a computer motherboard's full-size PCI slot or stand alone on a desktop or lab bench. Because the C62x McEVM 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 C62x McEVM 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 C62x 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 C62x McEVM 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 HPI. Code Composer uses the JTAG TBC to access the board, and the C62x McEVM 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 C62x McEVM is operated in a stand-alone configuration on a desktop or lab bench, an XDS510 emulator with C6000 Code Composer (not C6000 Code Composer for EVMs 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.

Chapter 2

Getting Started With the TMS320C62x McEVM

This chapter explains how to get started using the TMS320C62x McEVM. It provides a list of McEVM 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 McEVM kit components listed in section 2.1.

Topic **Page**

2.1 McEVM Kit Contents

The following checklist details items that are shipped with the C62x McEVM kit.

☐ **PCI plug-in card** TMS320C62x McEVM PCI plug-in card

☐ **CD-ROM** This CD-ROM contains the following software:

- ❑ **Device driver for Windows**. Includes common Win32 user-mode dynamic link library (DLL) with programmer API and low-level VxD driver
- ❑ **Host example code**. Example code that uses the Win32 user-mode DLL to access the McEVM board resources
- ❑ **Code Composer for EVMs**. Supports Windows 95/98 and Windows NT 4.0
- ❑ **C62x COFF loader**. A command-line utility that provides a means to configure the McEVM board and load an application via the DSP HPI
- ❑ **Confidence test utility**. A command-line utility that verifies proper operation of the McEVM by testing the major components on the board
- ❑ **Board configuration utility.** A command-line utility that resets and configures the McEVM board.
- ❑ **Board I/O control utility.** A command-line utility that configures McEVM I/O devices and supports streaming of data between the McEVM and host I.O files.
- ❑ **C6000 peripheral support library**. C6000 device library for DMA, timers, McBSPs, EMIF, HPI, and interrupt control

CD-ROM (continued)

- ❑ **C62x board support library**. McEVM board support library in object and source formats. Library and driver routines are included that support the McBSP, MVIP FMIC, T1/E1, VBAP, and PCI capabilities of the McEVM board.
- ❑ **C62x example code**. Example DSP applications using the C62x board support library
- ❑ *TMS320C6x Peripheral Support Library Programmer's Reference*
- ❑ *Code Composer User's Guide*
- ❑ *TMS320C62x/C67x CPU and Instruction Set Reference Guide*
- ❑ *TMS320C6201/C6701 Peripherals Reference Guide*
- ❑ *TMS320C6201 Digital Signal Processor Data Sheet*
- ❑ *TMS320C62x Multichannel Evaluation Module Technical Reference*
- ❑ ***TMS320C6x DSP CD-ROM.*** This interactive CD-ROM features the C6000 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:

- ❑ *TMS320C62x Multichannel Evaluation Module User's Guide* (this book)
- ❑ *Microprocessor Development Systems TI Customer Support Guide*
- ❑ *Government Compliance Card*
- ❑ *TMS320C62x McEVM Registration Card*

2.2 Before Starting the Installation Process

Before you begin installing the C62x McEVM, perform the following tasks:

- ❑ Verify that you have received the items listed as part of the C62x McEVM kit in section 2.1.
- ❑ 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 C62x McEVM.

Note:

If any items in the McEVM kit are missing or damaged, contact Texas Instruments immediately.

2.2.1 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 C62x McEVM.

2.2.2 Avoiding Obstructions to the McEVM Board

Avoid Obstructions

To minimize the risk of damage to the C62x McEVM, avoid physical obstructions that may prevent insertion of the board. You must install the McEVM in a full-length PCI slot only.

Before installing the McEVM, make sure that your PC has a full-length PCI slot. Otherwise, obstructions could prevent you from fully inserting the McEVM 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 McEVM. Possible obstructions also include memory modules, processor fans, power supply circuits, and cable connections. Failure to avoid obstructions when installing the McEVM could permanently damage the card.

2.3 McEVM 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 C62x McEVM board.

To install the C62x McEVM board into your PC, follow these steps:

1) Turn off your PC's power 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 cover from an unused PCI slot at the back of the computer.

5) Remove the C62x McEVM from the antistatic bag. Keep the antistatic bag for storing or transferring the board in the future.

Preventing Power Supply Damage

To prevent power supply damage, DO NOT connect power to the external power connector on the McEVM board when it is installed in a PCI slot on your computer. The external power connector is on ly for use in standalone operation.

6) Insert the C62x McEVM board into the empty, full-length PCI slot (see Figure 2–1). Secure its mounting bracket to the chassis with the previously removed screw.

7) If the McEVM will be installed at the end of your system's MVIP bus, then you should select MVIP C2o and C4b clock termination. This is selected by connecting pins 1 and 2 on the J5 header, and pins 1 and 2 on the J6 header. If the McEVM is the only MVIP card in the system, or it is in the middle of the MVIP bus, the factory defaults of no termination should be used (pins 2 and 3 on the J5 and J6) headers should be connected. Refer to Figure 2–3 for the location of the J5/J6 headers.

8) The McEVM DIP switches (SW2-1 through SW2-12) should be placed in their default settings. All switches should be in the ON (down) position ex-

cept switches SW2-3, SW2-4, and SW2-5, which should be in the OFF (up) position. Refer to Figure 2–1 for the location of the SW2 DIP switches.

Figure 2–1. TMS320C62x McEVM PCI Installation

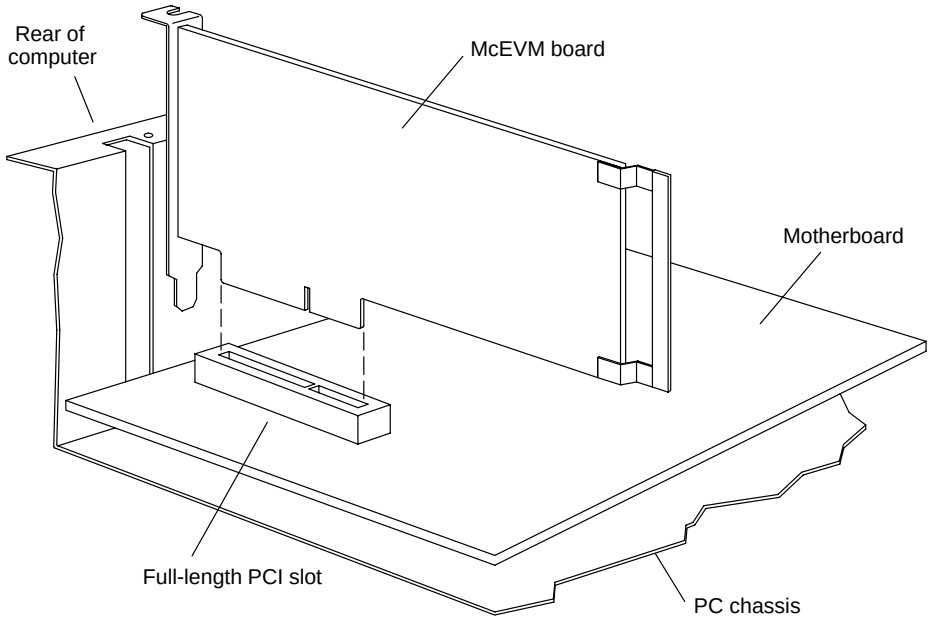


Table 2–1. TMS320C62x McEVM User Options DIP Switches

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 (50 MHz)	**OSC A (33.25 MHz)**
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. TMS320C62x McEVM 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**

9) Replace the PC cover.

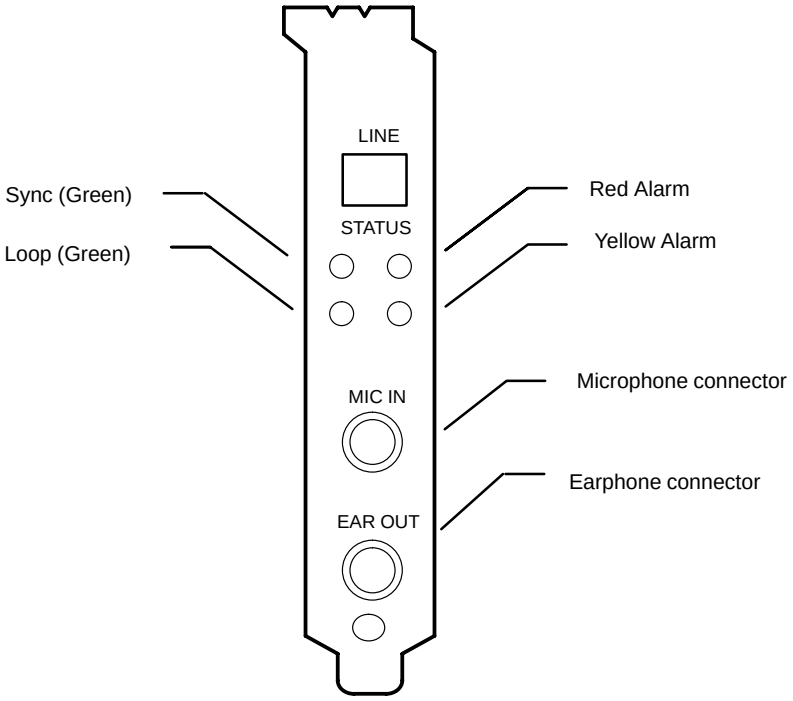
10) The connectors on the bracket are labeled with MIC IN, EAR OUT, and LINE. The MVIP-90 connector is located on the top edge of the card. Connect the handset cable (or microphone and speaker cables), T1/E1 cable, and MVIP-90 cable as needed.

The McEVM has four LEDs that display T1/E1 status. The following provides a summary of these LEDs and Figure 2–2 provides a graphical display.

Table 2–3. LED Summary Table

Bracket Location	Name	Color	Intended Indication
Bottom Left	GRN1	Green	Sync (Green)
Top Left	GRN2	Green	Loop (Green)
Top Right	RED3	Red	Red Alarm
Bottom Right	YEL	Yellow	Yellow Alarm

Figure 2–2. McEVM Board Mounting Bracket



11) Plug in the PC's power cord and turn on the PC's power.

12) If your operating system is Windows NT or later, the hardware installation is finished. Proceed to section 2.5, *McEVM Software Installation*. If your operating system is Windows 95/98, proceed to step 13.

13) When you turn on the PC's power, Windows 95/98 detects a newly installed McEVM board and prompts you for the driver disk. Insert the CD-ROM and follow the instructions on the screen. After the driver is installed, reboot the PC. To install the McEVM software, proceed to section 2.5, *McEVM Software Installation*.

2.4 McEVM 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 McEVM. Doing so can cause damage to your McEVM.

In stand-alone operation, the C62x McEVM must be connected to an external power supply and an emulator (both of which are not included in the C62x kit). To install the C62x McEVM 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 C62x McEVM external power connector (J15).

4) Connect the XDS510 emulator's 14-pin connector to the McEVM's JTAG header (J14) located at the top-center of the McEVM board.

Note:

Pin 6 of the 14-pin XDS510 emulator connector is filled in and the corresponding pin 6 of the JTAG header (J14) on the McEVM is missing to ensure proper connection.

5) Configure the C62x McEVM 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).

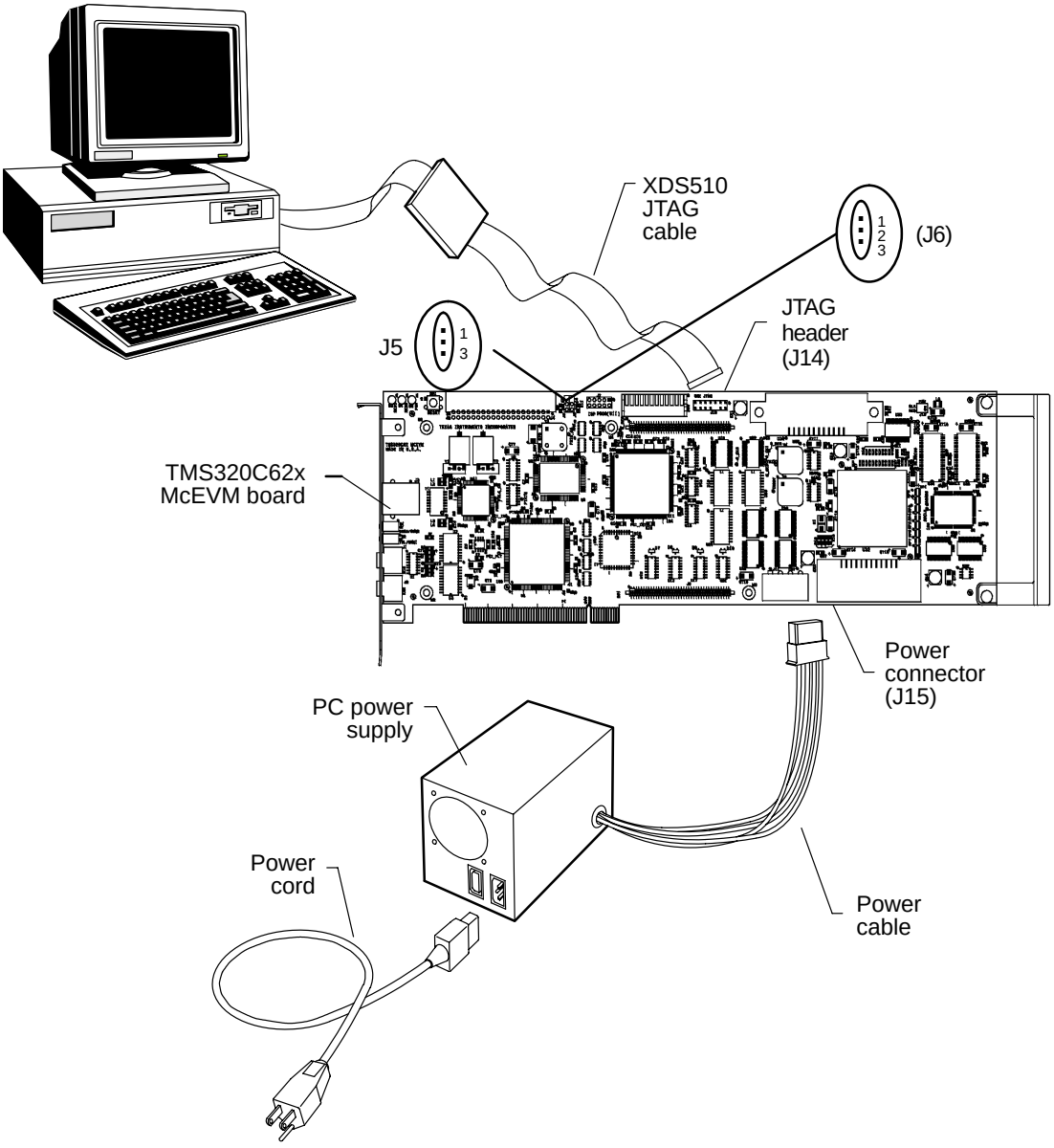
6) The connectors on the bracket are labeled with MIC IN, EAR OUT, and LINE (see Figure 2–2). Connect the handset cable (or microphone and speaker cables), T1/E1 cable, and MVIP-90 cable as needed.

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

8) Turn on the power supply and verify that the C62x McEVM's green power indicator is illuminated.

9) Power up the emulator.

Figure 2–3. TMS320C62x McEVM Stand-Alone Installation



2.5 McEVM Software Installation

These instructions help you install the Code Composer for TMS320C62x McEVM Software. This version of Code Composer only functions with the McEVM. It does not function with the XDS510 or other equipment. You will a need Pentium PC running Windows™ with a CD–ROM drive and one full–length revision 2.1–compliant PCI local bus slot (5V, 32 bits). Also, a minimum of 16M bytes of RAM (32M bytes recommended) and a minimum of 20M bytes to install the McEVM Support Software and Code Composer – TMS320C6x McEVM Edition.

Note:

You must be the administrator to install in some versions of Windows.

1) Shutdown the PC, install the McEVM and reboot the PC
2) When prompted about the driver file select “Driver from disk provided by hardware manufacturer” and click OK.
3) Specify the CD–ROM and click OK.
4) Restart the PC when prompted.
5) Insert the Code Composer for McvEVM CD–ROM into your CD–ROM drive.

On most Windows systems, Setup automatically starts when the CD–ROM is loaded. However, if AutoPlay is not enabled on your system, you need to run the setup.exe program in the top-level directory of the CD–ROM following the instructions below:

6) Choose Run from the Start menu.
7) In the Open box, type the following command:
   `d:\setup` where d: is the letter assigned to your CD–ROM drive.
8) Click on OK.

You are given several software installation options in a master setup window. You must install Device Drivers & Support Software for TMS320C6x McEVM and Code Composer Version 4.0. Installation of the Adobe Acrobat Reader software is optional.

9) Install the Device Drivers & Support Software for TMS320C6x McEVM software
10) Click on Device Drivers & Support Software for TMS320C6x McEVM to invoke the setup program for this software component and wait (this can be slow on some machines).

11) Setup will install the software in `c:\ti\EVM6x01` by default. You may change the destination folder but it is advised to leave the path as `d:\ti\EVM6x01` where d: is the drive of choice.

12) When Setup is complete choose No, I will restart my computer later and click Finish. This will return you to the master setup window.

13) Install the Code Composer Version 4.0 software

14) Click on Code Composer Version 4.0 to invoke the setup program for this software component and wait.

    Setup will install the software in `c:\ti` by default. You may change the destination folder but it is advised to leave the path as `d:\ti` where d: is the drive of choice.

15) When Setup is complete choose Launch Code Composer Setup and click Finish. This will invoke Code Composer Setup before returning to the master setup window.

16) Click on Install Device Driver in the right column of the Code Composer Setup window. Edit the Select Device Driver File dialog to select `c:\ti\c6000\drivers\tievm6xnt.dll` (assuming Code Composer was installed in the default location).

17) Drag the newly installed tievm6xnt icon in the center column to the left column. Click Next.

18) Edit the I/O Port Setting to change it to `0x0`. Click Finish.

19) Click the tievm6xnt icon under My System in the left column. This reveals components in the center column. Drag a TMS320C6x from the center column to the left column. Click OK. Save and Exit.

20) Install the Adobe Acrobat Reader software if desired. All the documentation contained on this C-ROM is in portable document format (PDF). As such the user will need an Adobe Acrobat Reader to view the documentation.

21) Restart the PC. Your newly installed software and EVM are ready for use.

Chapter 3

Running the Board Confidence Test

This chapter explains how to run the board confidence test included with the TMS320C62x McEVM board. This test can only be run after the McEVM board hardware and software have been installed. See Chapter 2, *Getting Started With the TMS320C62x McEVM*, for installation instructions.

Topic **Page**

3.1 About the Board Confidence Test

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

- ❑ SBSRAM
- ❑ SDRAM
- ❑ C62x interrupts
- ❑ C62x timers
- ❑ C62x DMA
- ❑ C62x McBSP serial ports
- ❑ PCI controller NVRAM (EEPROM)
- ❑ T1/E1 framer
- ❑ VBAP (voice-band audio processor)
- ❑ FMIC (flexible MVIP interface circuit)
- ❑ LEDs
- ❑ Endian mode selection

The installation of these McEVM 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, *McEVM Confidence Test*.

3.2 Verifying the McEVM Installation

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

1) Install the C62x McEVM board and supplied software as described in Chapter 2, *Getting Started With the TMS320C62x McEVM*.

2) Open a command window and change the directory to the conftest subdirectory of the McEVM 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 McEVM 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 shown in Example 3–1. If the test results in a failure message, see Appendix A, *Troubleshooting*.

Example 3–1. Sample Confidence Test Output

```
01:03:51 PM   Nov 17, 1998

Board Index                          : 0
Board Type                           : C6x McEVM, Rev 0

Resetting Board.
DIP Switch Settings ...
Board Clock                          : 33.25 MHz, x4 mode
Endian Mode                          : Little Endian
JTAG Selection                       : External XDS510
User Defined Option bits             : 3

Beginning Confidence Test ...

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

***** Memory Tests.
CPU Clock                            : 133 MHz
Endian Mode                          : Little Endian

Loading little endian memory test.............................PASSED.
SBSRAM A5 Test      : address 0x00012000; length 0x0001b800...PASSED.
SDRAM0 A5 Test      : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 A5 Test      : address 0x03000000; length 0x00100000...PASSED.
SBSRAM Pulse Test   : address 0x00012000; length 0x0001b800...PASSED.
SDRAM0 Pulse Test   : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 Pulse Test   : address 0x03000000; length 0x00100000...PASSED.

***** Memory Tests.
CPU Clock                            : 200 MHz
Endian Mode                          : Little Endian

Loading little endian memory test.............................PASSED.
SBSRAM A5 Test      : address 0x00012000; length 0x0001b800...PASSED.
SDRAM0 A5 Test      : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 A5 Test      : address 0x03000000; length 0x00100000...PASSED.
SBSRAM Pulse Test   : address 0x00012000; length 0x0001b800...PASSED.
SDRAM0 Pulse Test   : address 0x02000000; length 0x00100000...PASSED.
SDRAM1 Pulse Test   : address 0x03000000; length 0x00100000...PASSED.

***** Peripheral Confidence Tests.
CPU Clock                            : 200 MHz
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.
McBSP Asynchronous Send Synchronous Receive Test..............PASSED.
McBSP Synchronous Send Asynchronous Receive Test..............PASSED.
McBSP Asynchronous Send Asynchronous Receive Test.............PASSED.
NVRAM Checksum..(12AC)........................................PASSED.

***** McEvm Specific Confidence Tests.
CPU Clock                            : 200 MHz
Endian Mode                          : Little Endian
```

```
Loading little endian T1 Loop confidence test.................PASSED.
T1 Loop Test..................................................PASSED.
Loading little endian VBAP confidence test....................PASSED.
Gen Tone thru VBAP Test.......................................PASSED.
VBAP Loop Back Test...........................................PASSED.
Resetting Board.

***** C6x EVM/McEVM Confidence Test Complete – No Errors.
```

Chapter 4

TMS320C62x McEVM Host Utilities

This chapter describes how to use the following TMS320C62x McEVM host command-line utilities:

- McEVM COFF loader utility
- McEVM confidence test utility
- McEVM board configuration utility
- McEVM board I/O control utility

Topic **Page**

4.1 McEVM COFF Loader

The McEVM COFF loader command-line utility provides a means of displaying and/or loading and executing a C62x application program image stored in a pre-existent COFF file. In addition, the utility handles the host side of the C62x application's file I/O once execution begins. This is accomplished using the PCI bus interface between the host and the target DSP, but without the use of the emulator JTAG port.

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

evm6xldr *filename* [*options*]

evm6xldr	Command that invokes the McEVM COFF loader utility
filename	Name of the COFF file to be processed and loaded on the McEVM board. You must supply a filename.
options	Options that affect the way the McEVM 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. McEVM 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.
–i *filename*	Filename of the host file to be read and sent as file input data to the DSP. Host to DSP stream is not opened when omitted. The default filename for this option is "input.dat."
–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).
–ls *bytes*	Output file size limit (bytes). Saving data to the output file will terminate when this number of bytes is reached. The default value for this option is 0, which indicates "no limit."
–lt *seconds*	Output file time limit (sec). Saving data to the output file will terminate when this time limit is reached. The default value for this option is 0, which indicates "no limit."

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

Option	Effect
–o *filename*	Filename of the host file to be written with data received from the DSP as file output data. DSP to host stream is not opened when the –o option, the DSP to host stream, defaults to be piped to stdout.
–p *port*	Selects a specific McEVM target board; *port* is a zero-based relative board index, ranging from 0 (for 1 board) to n–1, where n McEVM boards are installed. The default is 0.
–q	Suppresses output to the display (quiet option)
–r *reset*	Controls the McEVM 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).
–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 McEVM 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 C62x 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) Host handles I/O data streams to output file and/or input file.

10) File I/O continues until a <Ctrl-Z> is entered or until file size or time limit restrictions have been met.

11) Closes input file, output file, and the driver connection to the EVM target board, and the application terminates.

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

```
evm6xldr evm.out
```

4.2 McEVM Confidence Test

The confidence test is a command-line utility that allows you to confirm the proper installation of the McEVM 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 McEVM installation directory.

To invoke the McEVM confidence test, use this syntax:

evm6xtst [*options*] [*log_filename*]

evm6xtst	Command that invokes the McEVM confidence test
options	Options that affect the way the McEVM 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. McEVM 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 McEVM 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
- Boot mode
- Board clock
- Endian mode
- JTAG selection
- User-defined options

The target board being tested is reset and configured into SBSRAM clock mode and little-endian mode. While the board is configured in this manner, a C62x memory test module is loaded to the C62x and executed. This verifies proper operation of the host port interface (HPI), which is used to load the C62x executable code, and the onboard memory while in SBSRAM clock mode and little-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 C62x and executed. These tests include:

- Memory tests
- 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 NORMAL clock mode and little-endian mode. While the board is in this configuration, a sequence of external peripheral tests are performed to check the MVIP, T1/E1, and VBAP circuits on the McEVM board. You can set up for these tests by connecting powered speakers to the EAR OUT connector and a microphone to the MIC IN connector. These connections are not required for successful completion of the following tests:

- T1/E1 loop test
- VBAP tone test
- VBAP loop test

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

4.3 McEVM Board Configuration

The McEVM C62x board configuration utility resets the McEVM board and, optionally, configures the board's clock speed, endian mode, memory map, JTAG selection, and user bits. If none of the configuration parameters is used, the board configuration is taken from the hardware DIP switches. With this utility, 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 McEVM'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 McEVM's hardware DIP switches are not used. All unspecified configuration parameters are set to the defaults described in Table 4–3.

To invoke the McEVM C62x board configuration utility, use this syntax:

evm6xrst [*options*]

evm6xrst	Command that invokes the McEVM C62x board configuration utility
options	Options that affect the way the McEVM 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. McEVM 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 McEVM'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. McEVM 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 Code Composer 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 McEVM's JTAG test bus controller for embedded emulation and requires the use of the C62x McEVM 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 McEVM boards are installed. The default is 0.
–q	Suppresses output to the display (quiet option)
–r *reset*	Controls the McEVM 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. The default is 2 (133 MHz).
–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 McEVM 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.
–x	Ignores relative board index selected in the D_OPTIONS environment variable
–z	Displays configuration status and waits for key press after each step

4.4 McEVM Board I/O Control Utility

The McEVM ‘C62x board I/O utility allows a user to configure all McEVM I/O devices. It also allows for data to be streamed to and from the McEVM via an input and output file on the host. These functions are accomplished via the PCI and the ‘C62x host port interface (HPI), without the need for an emulator and without the use of the JTAG port.

For example, the utility can be used to monitor a particular MVIP channel on the VBAP. A user could also record data from an MVIP channel to a disk file.

The command line for this application supports the input of a script file, which lists the device driver and device library calls in ASCII just as they are to be sent to the C62x. The scripting supports one call per line and ignores lines beginning with a ‘#’ character for commenting. The scripting support reads each line of ASCII data and sends it to the C62x as it would any other device driver or device library call.

This application also supports a limited number of direct commands, simple device operations that do not require a script file. The utility makes all the calls required to initialize the board I/O based on command line options, then continues by handling the requested file I/O.

One benefit of the script support is that the user has a chance to use the C6x device driver and device library calls from the host environment without having to write any C62x code. This utility could be used to test out a board I/O configuration before writing C62x code.

To invoke the McEVM C62x board I/O utility, use this syntax:

evm6xbio *@script_file_name* [*options*]

-or-

evm6xbio *direct_command_name* [*optional_parameters*] [*options*]

evm6xbio	Command that invokes the EVM C6x board I/O utility.
script_file_name	Name of the script file to be executed by the host utility.
direct_command_name	Name of the direct command to be executed by the host utility (see section 4.4.1).

optional_parameters	Optional parameters that are specific to the direct command to be executed by the host utility. These parameters are not case sensitive and can appear anywhere on the command line following the command. Parameter syntax and default values are identified in the direct command description.
options	Options that affect the way the McEVM board I/O 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 4–4. Each option is disabled by default.

Table 4–4. McEVM Board I/O Utility Options

Option	Effect
–?	Displays command-line help information; all other options and parameters are ignored when you specify the –? option.
-p *port*	Selects a specific target board from 0-3. The default value for this option is 0.
-i *filename*	Filename of the host file to be read and sent to the DSP. This file will be used as input for the operation indicated by the script file or the direct command. This parameter will override the *play* filename indicated in a script file. The default value for this option is “input.dat.”
-o *filename*	Filename of the host file to be written with data received from the DSP. This file will record the output for the operation indicated by the script file or the direct command. This parameter will override the *record* filename indicated in a script file. The default value for this option is “output.dat.”
-ls *bytes*	Output file size limit (bytes). Recording of data to the output file will terminate when this number of bytes is reached. This parameter will override the *record max_size* indicated in a script file. The default value for this option is 0, which indicates “no limit.”
-lt *seconds*	Output file time limit (sec). Recording of data to the output file will terminate when this time limit is reached. This parameter will override the *record max_time* indicated in a script file. The default value for this option is 0, which indicates “no limit.”
-z	Displays board I/O status and waits for key press after each step.

The board I/O generally performs the following steps:

1) If executing with script file input, performs a pre-execution syntax check of scripting commands.
2) Opens a driver to a specific Mc EVM target board.
3) Performs board reset.
4) Performs an HPI boot mode DSP reset, which permits HPI operation while holding the DSP in a halted state.
5) Opens an HPI connection.
6) Uses the HPI to load the board I/O DSP executable image.
7) Closes the HPI connection.
8) Releases the DSP from the halted state, thus starting program execution.
9) If executing with script file input, reads and processes each driver and library call within the script file and waits for a response from the DSP between calls. If executing a direct command, executes a series of pre-defined driver and library calls to perform the direct command operation.
10) Sends a command to the DSP indicating all setup operations have been performed and to begin execution.
11) Launches file I/O processes to play to and record from data files as indicated by the script or the direct command.
12) File I/O continues until a <Ctrl-Z> is entered or until file size or time limit restrictions have been met.
13) Closes input file, output file and the driver connection to the EVM target board and the application terminates.

4.4.1 Direct commands

The following commands are executed directly from the command line instead of via script file processing. These operations provide easy access to basic I/O operation of the board.

VBAP Record and Playback

These commands record or play audio data between the VBAP device and a host file.

Direct command names:

vbap_record
vbap_play

Command specific parameters:

-dv *mode*	companding mode. Supported values are μ-law and a-law. The default value is μ-law.

T1/E1 Channel Record and Playback

These commands record or play data between a single T1/E1 channel and a host file.

Direct command names:

t1e1_record

t1e1_play

Command specific parameters:

-dt channel	selected T1/E1 channel. Supported values are 1 – 24 for T1 and 1 – 32 for E1. The default value is 1.
-dm mode	T1 or E1 mode selection. Supported values are t1 or e1. The default value is t1.

MVIP Channel Record and Playback

These commands record or play data between a single MVIP channel and a host file.

Direct command names:

mvip_record
mvip_play

Command specific parameters:

-dc *channel*	selected MVIP channel. Supported values are 0 - 383. The default value is 0.

VBAP to T1/E1 Channel Connection

This command creates a connection between the VBAP device and a selected T1/E1 channel.

Direct command name:

monitor_t1e1

Command specific parameters:

-dt *channel*	selected T1/E1 channel. Supported values are 1 – 24 for T1 and 1 – 32 for E1. The default value is 1.
-dm mode	T1 or E1 mode selection. Supported values are t1 or e1. The default value is t1.
-dv mode	companding mode. Supported values are u-law and a-law. The default value is μ-law.

VBAP to MVIP Channel Connection

This command creates a connection between the VBAP device and a selected MVIP channel.

Direct command name:

monitor_mvip

Command specific parameters:

-dc channel	selected MVIP channel. Supported values are 0 - 383. The default value is 0.
-dv mode	companding mode. Supported values are u-law and a-law. The default value is μ-law.

T1/E1 Channel to MVIP Channel Connection

This command creates a connection between a T1/E1 channel and a selected MVIP channel.

Direct command name:

t1e1_to_mvip

Command specific parameters:

-dt channel	selected T1/E1 channel. Supported values are 1 – 24 for T1 and 1 – 32 for E1. The default value is 1.
-dm mode	T1 or E1 mode selection. Supported values are t1 or e1. The default value is t1.
-dc channel	selected MVIP channel. Supported values are 0 - 383. The default value is 0.

4.4.2 Script Command Protocol

The script command protocol represents the subset of driver interfaces necessary to specify the behavior and connectivity of the devices on the McEVM. In

addition, the command protocol may specify the host files to use for playback and record of data streams.

The command syntax is not case sensitive. Whitespace is optional, but parentheses, underscores and commas are required. Blank lines, or lines beginning with # are ignored.

All script commands with the exception of *play* and *record* translate directly to driver calls.

Host Commands

Play(filename) – Specifies the file to use as the source of the data stream to send to the McEVM board at completion of setup commands.

Record(filename, max_size, max_time) – Specifies the file to use to record the data stream sent to the host by the control program on the McEVM board at completion of setup commands. Max_size specifies the maximum size of the recorded stream (in bytes). 0 indicates no size limit. Max_time specifies the maximum time to record a stream of data (in seconds). 0 indicates no time limit.

FMIC Commands

Fmic_timing(fmic_instance, timing_mode)	Setup timing mode for the FMIC
fmic_instance [0,1]	Indicates which FMIC is being adjusted (0–McEVM, 1–daughterboard)
fmic_timing_mode [FMIC_MVIP_BUS_TIMING, FMIC_MOTHER_BD_FREE_RUN, FMIC_DAUGHTER_BD_FREE_RUN, FMIC_MOTHER_BD_RCV_TIMING, FMIC_DAUGHTER_BD_RCV_TIMING]	Determines FMIC timing setup
Fmic_connect(fmic_instance, out_channel, in_channel)	Connect an input channel to an output channel
fmic_instance [0,1]	indicates which FMIC is being adjusted (0–McEVM, 1–daughterboard)
out_channel [0–383]	indicates the destination channel whose data stream source is being determined (see note on channel organization)
in_channel [0–383]	indicates the source channel of a data stream (see note on channel organization)
Fmic_direction(fmic_instance, channel, direction)	Set the direction of the MVIP bus line for the indicated channel.

fmic_instance [0,1]	indicates which FMIC is being adjusted (0–McEVM, 1–daughterboard)
channel [0–255]	indicates the channel whose direction is being determined
direction [FMIC_DSo_IN, FMIC_DSi_IN]	indicates which MVIP bus line is used as input for the indicated channel
Fmic_output_control(fmic_instance, channel, enable)	enables or disables an output channel.
fmic_instance [0,1]	indicates which FMIC is being adjusted (0–McEVM, 1–daughterboard)
channel [0–255]	indicates the channel whose output is enables/disabled
enable [TRUE, FALSE]	indicates whether indicated channel is to be enabled

Note on channel organization:

The FMIC is a 384 non-blocking TDM switch. These channels are divided among 8 MVIP streams and 4 local streams. Each stream has 32 channels. The MVIP streams are represented by channels 0-255. Local stream 0 (channels 256-287) is mapped to the McBSP0 of the C62x DSP. The first channel of this group is used as the conduit for host file record and playback; the other channels are ignored. Local stream 1 (channels 288-319) is mapped to the VBAP. Local stream 2 (channels 320-351) is mapped to the T1/E1 framer. Local stream 3 (channels 352-358) is mapped to the daughter board TDM interface.

VBAP Commands

Vbap_set(law)	select a-law or μ-law encoding and decoding for the VBAP.
Law – [VBAP_A_LAW, VBAP_U_LAW]	selects a–law or μ–law encoding and decoding

T1/E1 Commands

T1e1_init(t1e1_instance, line_mode, line_code_mode, framing_mode, sync_mode, loopback_mode)	Initialize the T1/E1 framer operating mode.
t1e1_instance [0,1]	indicates which T1/E1 is being adjusted (0–McEVM, 1–daughterboard)
line_mode [T1E1_T1, T1E1_E1]	determines whether the T1/E1 operates as T1 or E1

line_code_mode [T1E1_T1_AMI, T1E1_T1_B8ZS] or [T1E1_E1_AMI, T1E1_E1_HDB3]	determines the line code operation available during T1 or E1 operation respectively
framine_mode [T1E1_T1_D4, T1E1_T1_ESF] or [T1E1_E1_CRC4_MULTIFRAME, T1E1_E1_DOUBLE_FRAME]	determines the framing behavior during T1 or E1 operation respectively
sync_mode [T1E1_MVIP_BUS_TIMING, T1E1_RCV_TIMING, T1E1_MOTHER_BD_FREE_RUN, T1E1_DAUGHTER_BD_FREE_RUN, T1E1_MOTHER_BD_RCV_TIMING, T1E1_XMIT_NMOTHER_BD_RCV_TIMING, T1E1_SMIT_DAUGHTER_BD_RCV_TIMING, T1E1_DAUGHTER_BD_RCV_TIMING]	determines the sync behavior during both T1 and E1 operation
T1e1_channel_loopback (t1e1_instance, channel, control, idle_code)	Enables or disables loopback mode for the indicated channel.
t1e1_instance [0,1]	indicates which T1/E1 is being adjusted (0–McEVM, 1–daughterboard)
channel [0–31]	indicates which channel is being controlled
control [enable, disable]	determines whether loopback is being enabled or disabled
idle_code [0–255]	code to send on outgoing side of loopback channel

Appendix A

Troubleshooting

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

If you experience difficulty when installing or using the TMS320C62x McEVM 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 McEVM 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 interconnect (PCI) configuration is incorrect.

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

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

Could not open handle

Description If an McEVM application displays a message similar to:

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

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

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

If the McEVM is fully seated, reboot the PC and check if the problem is solved. If you move a McEVM board from one PCI slot to another or install it into a PCI slot that previously held a McEVM board, some versions of Windows 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 C62x McEVM support software CD-ROM to reinstall the McEVM 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 McEVM 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 McEVM 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 McEVM.
- ❑ 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 McEVM, and, therefore, the device driver will not load.

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

Interrupt (IRQ) conflict

Description If a PC will not boot properly after a McEVM is installed or a McEVM application locks up the PC or causes a Windows exception screen to be displayed, there is probably an interrupt conflict between the McEVM and another board in the system. The McEVM'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 re-

moved, the BIOS can get confused and may end up making inappropriate decisions that can result in an interrupt being shared between the McEVM and another board.

The McEVM's support software handles shared interrupts; 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 ends up assigning multiple devices to the same interrupt, such as IRQ9, even though there are other IRQs available. When a board that does not share interrupts is assigned the same IRQ and the McEVM, the system will not operate properly. In most cases the PC will not boot at all. This problem is a system-level problem, not a problem with your McEVM.

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 since 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. We recommend that you read all the items before you take action so that you understand all the potential actions they may apply to your situation.

❑ It may be helpful to check your system's current interrupt assignments to determine exactly which device is conflicting with the McEVM's interrupt.

To observe the interrupt assignments, perform the following steps in 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 assignments.

■ **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 McEVM 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 McEVM 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, then change them to Available to make IRQs available. This feature may not be accessible with your particular BIOS.

❑ If all IRQs are already being used in the system, any unused IDE and USB interfaces in the system should be disabled in the system BIOS, or in the Windows Device Manager, to free up interrupts. This may allow the BIOS to assign the McEVM 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 McEVM'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.
    1) Power off the computer and remove all PCI cards.
    2) Power on the computer momentarily (do not worry about any boot errors because of missing vital cards, such as VGA card, network card, etc.). A few seconds is all that is needed to clear the plug and play BIOS resource allocations.
    3) Power off the computer.
    4) Power on the computer momentarily, then power off again.
    5) Continue this process of powering on and off the computer, reinstalling each remaining PCI card in between, until all PCI cards have been reinstalled.
    6) Boot the system and check for conflicts. If the McEVM is still in conflict, you may need to disable the offending IRQ via the BIOS setup and try resetting the plug and play BIOS again, reenabling the IRQ if needed between the reinstall of two of the PCI cards.
- 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 available from their support web page.
- If you are running under Windows 95, it is possible to manually set the McEVM's IRQ assignment. In the BIOS setup, enable the PnP OS option if it is available. This will allow Windows 95 to be able to control the plug and play information and manually assign interrupts in the Device Manager.

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. The following steps should be performed:

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 McEVM item.
6) The General tab should be active. Click on the Disable in this hardware profile check box so that it is checked.
7) Click on the OK button and reboot.

❑ Once the system is rebooted, the following steps should be performed:

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 McEVM 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 checkbox 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.

❑ Since 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, then you must resort to vendor or BIOS utilities.

- Check with your computer manufacturer. Some of them, such as Compaq, include or can provide a utility that allows users to reallocate IRQs if needed.
- Some BIOS vendors also offer utilities that allow you to manually control IRQs. One of them, called ICU configuration utility, is available from Intel. 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 Intel web page below:

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

PCI bus mastering failure

Description To use the McEVM's PCI bus mastering capability, the McEVM 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 McEVM 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 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 McEVM 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 C62x McEVM debugger with the McEVM 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, *McEVM Board Configuration,* on page 4-7.

If you are using the C62x McEVM in a stand-alone configuration with an external XDS510 emulator and the standard TI C62x source debugger, press the McEVM'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 McEVM support software.

Appendix B

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 ’C62x 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).

channel service unit (CSU): A device used to connect a digital phone line, such as T1/E1, coming in from the phone company to another device producing a digital signal.

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 ’C62x 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*.

CSU: See *channel service 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.

flexible MVIP integrated circuit (FMIC): Device which provides a complete MVIP-compliant interface between the MVIP bus and a variety of processors, telephony interfaces and other circuits. It also provides a 384 x 384 switching matrix between the MVIP bus and four local data streams.

H

halfword: The 'C62x 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 Services Digital Network (ISDN): A worldwide digital communications network evolving from existing telephone services. Its goal is to replace current telephone lines that require DA conversions with totally digital switching and transmission facilities capable of carrying a variety of data—from voice to computer transmissions, music, and video. The ISDN is built on two main types of communications channels: a B channel, which carries data at 64 Kb/s, and a D channel, which carries control information at either 16 or 64 Kb/s. Computers and other devices connect ISDN lines through simple, standardized interfaces.

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.

ISDN: See *Integrated Services Digital Network*.

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.

LBO: See *line build out*.

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

line build out (LBO): Selectable output attenuation with typical loss of 0.0, 7.5 and 15 dB at 772 kHz.

line interface unit (LIU): A device or a part of a device responsible for interfacing to a digital network (T1/E1) line.

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*.

LIU: See *line interface unit*.

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).

Mbps: See *megabit per second*.

McBSP: See *multichannel buffered serial port.*

McEVM: A multichannel communications development board and software tools that allow the user to evaluate a specific device.

megabit per second (Mbps): A million bits of data per second.

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 'C62x 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.)

Multi-Vendor Integration Protocol (MVIP): A family of standards that allows products from different vendors to interoperate withina computer or group of computers. The MVIP bus is a telephony bus that provides 256 full-duplex voice channels (16 streams of 32 timeslots) over a ribbon cables between PC boards.

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

MVIP: See *Multi-Vendor Integration Protocol*.

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 memory*.

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

T1/E1: T1 is a digital transmission link with a capacity of 1.544 Mbps. It uses two pairs of normal twisted-wires and can handle 24-voice conversations, each digitized using mu-law coding at 64 kbps. T1 is used in USA, Canada, Hong Kong, and Japan. E1 is a digital transmission link with a capacity of 2.048 Mbps. It is the European equivalent of T1. It can handle 30-voice conversations, each digitized using A-law coding at 64 kbps.

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.

TDM: See *time-division multiplexed*.

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.

time-division multiplexed (TDM): The process by which a single serial bus is shared by multiple devices with each device taking turns to communicate on the bus. The total number of time slots (channels) depends on the number of devices connected. During a time slot, a given device may talk to any combination of devices on the bus.

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.

tri-state: High impedance.

V

V: See *volt.*

VBAP: See *voice-band audio processor*.

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*.

voice-band audio processor (VBAP): A voice codec device that provides filtering, analog-to-digital and digital-to-analog conversion with interfaces to a microphone, speaker, and serial, digital itnerface. It can operate in either mu-law or A-law companding or 13-bit linear modes.

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 ’C62x 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.

xDSL: A generic name (x is the generic) for digital subscriber line equipment and services. Asymmetric DSL (ADSL) is one of the more popular types, providing up to 6 Mbsp downstream and 640 kbps upstream data rates using standard twisted pair wiring.

Index

A

B

C

D

E

F

H

I

J

U

V

W

X

Z