

Board Support Package for TMS67xx DSPs and Windows XP

SOFTWARE MANUAL

745-01-003-4000

Revision A

October 2006

ALPHI TECHNOLOGY CORPORATION

6202 S. Maple Avenue #120

Tempe, AZ 85283 USA

Tel: (480) 838-2428

Fax: (480) 838-4477

NOTICE

The information in this document has been carefully checked and is believed to be entirely reliable. While all reasonable efforts to ensure accuracy have been taken in the preparation of this manual, ALPHI TECHNOLOGY assumes no responsibility resulting from omissions or errors in this manual or from the use of information contain herein.

ALPHI TECHNOLOGY reserves the right to make any changes, without notice, to this or any of ALPHI TECHNOLOGY's products to improve reliability, performance, function or design.

ALPHI TECHNOLOGY does not assume any liability arising out of the application or use of any product or circuit described herein; nor does ALPHI TECHNOLOGY convey any license under its patent rights or the rights of others.

ALPHI TECHNOLOGY CORPORATION

All Rights Reserved

This document shall not be duplicated, nor its contents used for any purpose, unless express permission has been granted in advance.

TABLE OF CONTENTS

1.	GENERAL DESCRIPTION	1
1.1	INTRODUCTION	1
1.2	FEATURE SUMMARY	1
1.3	FUNCTIONAL DESCRIPTION	2
1.4	REFERENCE MATERIALS LIST	3
2.	INSTALLATION INSTRUCTIONS	4
2.1	Hardware and Software Requirements	4
2.2	Pre-Installation	4
2.3	Installation	4
2.4	Notes	4
3.	GENERAL ACCESS BY THE HOST	5
3.1	ALPH9056 - XP Device Drivers for PLX9056 based PCI cards	5
3.2	ALPHIDLL – General access DLL for all PCI cards	6
3.3	ALPHIERRORCODE – DLL to help display meanings of error codes	7
3.4	GETSYMBOLSFROMCOFF – DLL to read DSP COFF file and get symbol values	Error! Bookmark
4.	HOST UTILITIES	8
4.1	BUSMASTERDMA – Utility to test and demonstrate ability to transfer data between DSP and host	8
4.2	DOWNLOAD – Download and execute DSP code	8
4.3	HOSTBUG – Utility to access and test card resources, enumerate system	9
4.4	INTERRUPT – Utility to test and demonstrate ability to interrupt host	12
4.5	PFLASH – FLASH programming utility	13
4.6	PLX9056 - PLX NVRAM Image Editor and Programmer	15
4.7	REPORTVERSION – Software configuration management tool	18
4.8	SHAREDHOSTMEMORY – Utility to demonstrate sharing HOST memory with a PLX based design	19
4.9	SYMBOLS – Demonstrate the GetSymbolsFromCoff DLL	Error! Bookmark not defined.
5.	DSP SUPPORT LIBRARY (Lib6713)	20
5.1	PLX / TMS67xx based hardware	20
6.	GENERAL PURPOSE DSP UTILITIES AND EXAMPLES	21
6.1	DSP BOOTLOADER / DEBUGGER	21
6.2	TESTBOOTROM	Error! Bookmark not defined.
6.3	DMAEXAMPLE	27

6.4	INTLOOP	27
6.5	SERIALEX	Error! Bookmark not defined.
7.	DSP SPECIFIC EXAMPLES	<i>Error! Bookmark not defined.</i>
7.1	ADDA	Error! Bookmark not defined.
7.2	ADA08	Error! Bookmark not defined.
7.3	CIO32	Error! Bookmark not defined.
7.4	SCC04	Error! Bookmark not defined.
8.	BUILDING CUSTOM DSP CODE	28
8.1	Development tools for the DSP	28
8.2	Lib6713 Library	28
8.3	Compiling and Linking	28
8.4	Memory Map	28
8.5	Creating TEKTRONIX Files	29
9.	STAND ALONE OPERATION	30
9.1	Boot Mode	30
9.2	Field Update through Serial Cable	30
9.3	Update by HOST Computer	30
9.4	Required External Connections	30

1. GENERAL DESCRIPTION

1.1 INTRODUCTION

The **Board Support Package for TMS67xx DSPs and Windows XP** is a collection of necessary software and examples offering the customer the ability to quickly create custom applications under the Windows XP operating system on INTEL platforms.

Additionally, the software examples offer the ability to quickly ensure that the hardware is operating, and offer a starting point for custom applications with the supplied source.

Software utilities allow the user to configure the card for certain modes of operation, quickly access the card resources, and on cards with DSPs, download and run DSP code, and program the FLASH memory.

For the users of other operating systems than Windows XP, this package provides an example of accessing the cards with source code both for the HOST and the DSP.

1.2 FEATURE SUMMARY

INCLUDES:

- Windows device drivers
- 32-bit DLLs
- DSP Support Library
- Demo and example programs
- Full source code (except device driver) in C/C++

FEATURES:

- Download and control DSP via local PCI/CPCI bus
- Download and control DSP via serial interface
- Communicate between DSP and HOST by mailbox and FIFO registers
- Data transfer between HOST and DSP by bus master DMA
- Full access (on PLX products) to PCI/CPCI bus by DSP
- Auto-configure for all ALPHI PCI products
- Full interrupt support on HOST and DSP
- Direct communication with board I/O
- Object oriented design (C/C++)

SUPPORTS:

- Hosted and stand alone option
- Multiple I/O boards
- Most 32 bit software tools on XP

Board Support Package for TMS67xx DSPs and Windows XP SOFTWARE MANUAL

ALPHI Technology cards with DSPs are supported by a DSP support library called **Lib6713** providing the following features:

- Common access routines for communications with the host by Mailbox and FIFO
- Mapping C language standard input and output by the DSP to the onboard 8530 serial chip
- Identify the applicable card resources and parameters
- On PLX designs, direct access to the PCI/CPCI bus

This library is consistent across ALPHI Technology cards. A boot-loader is provided to allow for control by the HOST and for independent operation in stand-alone operation. User code can be downloaded to FLASH memory and booted automatically on reset. Access to the card can be made both by the HOST bus and serially through the 8530.

1.3 FUNCTIONAL DESCRIPTION

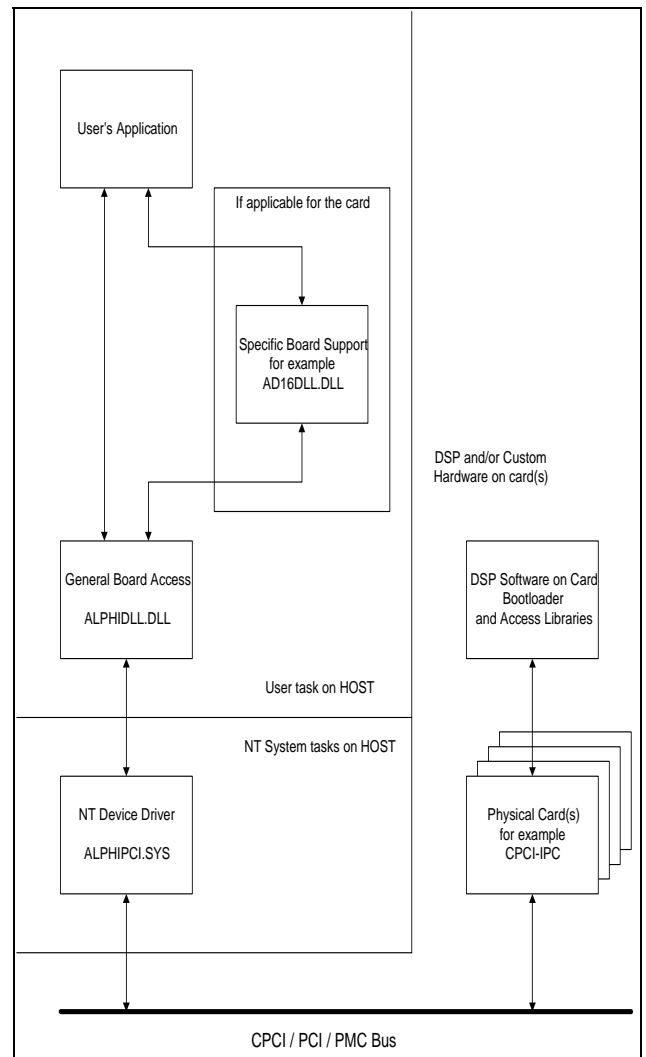
The **Board Support Package for TMS67xx DSPs and Windows XP** is composed of many subcomponents, some of which are applicable to many of the cards offered by ALPHI Technology, some of which are specific to a particular card.

The software installation disk can install the appropriate software and examples for each product, as well as support for all the products.

The components described in this document fit into one of the following categories.

- General Access by the HOST
- HOST Utilities
- Specific HOST Examples and Access to Specific Cards
- DSP Support Library
- DSP Boot-loader / Debugger
- DSP Specific Examples

A functional block diagram of the **Board Support Package for TMS67xx DSPs and Windows XP** is presented to the right.



1.4 REFERENCE MATERIALS LIST

PCI Local Bus Specification:

PCI Special Interest Group
Tel: (800) 433-5177
Tel: (503) 797-4207
Fax: (503) 234-6762

PLX PCI 9056 PCI Controller data book:

PLX Technology Corporation
Tel: (408) 744-9060
<http://www.plxtech.com>
apps@plxtech.com

DSP (TMS320C31/C32) C Compiler, Assembler, Linker:

Part Number TMDU3243855-02

Texas Instruments
Tel: (410) 312-7900
<http://www.ti.com>

DSP Debugger and Integrated Development Environment:

Code Composer

Texas Instruments
Tel: (410) 312-7900
<http://www.ti.com>

DSP Emulators suitable for use with *Code Composer*:

XDS150PP MPD

Spectrum Digital Incorporated
(281) 561-6952
<http://www.spectrumdigital.com>

2. INSTALLATION INSTRUCTIONS

This chapter explains how to install the **Board Support Package for TMS67xx DSPs and Windows XP**.

2.1 Hardware and Software Requirements

Although the target environment of this software is Windows XP, it is possible to use Windows 9x as a development platform. Therefore, the requirements list is separated into Development Machine and Target Machine. Of course, they can be the same machine.

2.1.1 Target Machine

PC	Intel Pentium 133 or later, with appropriate bus for hardware.
Operating System	Windows XP
Disk Space	500 MB. Or more

2.1.2 Development Machine

PC	Intel Pentium 133 or later.
Operating System	Windows XP.
Disk Space	Less than 5 MB.
HOST Software Development Tools	Any compiler capable of writing native Windows applications. Must be capable of calling system DLLs directly. A C or C++ compiler is suggested. Other languages require that the user create custom header files.
Optional DSP Software Development Tools	TI C Compiler and Assembler/Linker for C3x processors <i>Part Number TMDU3243855-02</i> See the Reference Material List for details.

2.2 Pre-Installation

If you are installing the product on Windows XP, verify that you have an account with administrator privileges.

If you have previously installed the product on this machine, be sure to uninstall the previous version and then reboot. The earlier version can be uninstalled by using the **Add/Remove Programs** icon in the **Control Panel**. Select "*Board Support for ALPHI Technology PCI Hardware on Window XP*" and **Add/Remove**. You will need to reboot to unload the earlier device driver.

Exit all Windows programs prior to installation.

If you are installing to a target machine, install the card in the system before installing the software.

2.3 Installation

Log in with administrator privileges.

Insert the CD-ROM into your CD drive and run the setup program (setup.exe).

Follow the on screen instructions.

2.4 Notes

Any errors related to driver installation are usually the result of not removing the previous installation of the software. Occasionally, errors are valid configuration problems. If you are still having problems, please contact the factory.

If the customer desires to make this software part of an installation script for the end application, please be sure to read the **Installation** section of the ALPHIPCI driver.

3. GENERAL ACCESS BY THE HOST

3.1 ALPH9056 - XP Device Drivers for PLX9056 based PCI cards

Applicable To: All PLX9056 based PCI, CPCI, and PMC Products
Executable File: ALPH9056.SYS
Source Location: Not included in package
Include File: C:\ALPHI67xx\INCLUDE\DDALPHIP.H

3.1.1 Description

This Windows XP device driver allows the customer to directly access the card resources. The Application Programmer's Interface (API), provided by this device driver, allows the customer to perform the following tasks:

- Determine which card(s) are available
- Get the capabilities of a card
- Memory map the card resources for direct access by the customer application
- Hook the interrupt line of the card, and wait for an interrupt
- Setup and use Bus Master DMA
- Read and write the NVRAM configuration of the card

Although the customer can use the API directly, an access DLL (AlphiDll.dll) wraps the necessary IOCTL calls into a more useful form for the programmer.

ALPH9056.SYS is common to all PLX9056 based designs. They both will communicate with multiple instances of cards in a system, and with multiple types of cards.

3.1.2 API

The interface to each device driver is provided via a series of IOCTL calls and structures passed to the device driver. The interface is described in the documentation for the AlphiDll provided in manual form and a Windows Help file.

3.1.3 Device Names

User applications wishing to utilize the resources of ALPHI Technology's PCI cards tend to fall into two categories. There are applications which know the exact types of cards required in the host system in order to operate correctly and those which do not really care about what type of card is present or the quantity, or will enumerate all the cards in a host system.

In order to facilitate both types of applications, the device drivers publish two different names for each card, a generic name and a specific name. Both names are followed by a decimal digit to uniquely identify the card. All the functionality of the driver for a particular card type is available regardless of which name was used to open the device.

The generic name is of the form ALPHIPCI[n] or ALPHIPLX[n] where n is a digit starting at 0. Every card handled by the driver will have a name of this form. An application wishing to enumerate the resources on the host, or interested in performing a task which is common to many different cards, (such as downloading and running DSP code, or burning a new DSP code into FLASH) can use the generic name in the Win32 API call to CreateFile. By opening each device in sequence (0, 1, 2...) until one fails for not existing, the application can enumerate every card in the host system. By using the IOCTL IOCTL_ALPHIPCI_GET_DEVICE_CAPABILITIES, the application can get the equivalent specific name (or true name) of the card, as well as the resources available.

The specific name, or true name, is of similar form and depends upon the device of interest. For example, the CPCI-IPC card will have a specific name starting with CPCI-IPC0. This allows an application expecting a specific type of board to find it quickly.

3.2 ALPHIDLL – General access DLL for all PCI cards

Applicable To: All PCI, CPCI, and PMC Products
Executable File: C:\ALPHI67xx\ALPHIDLL.DLL
Source Location: C:\ALPHI67xx\ALPHIDLL
Include File: C:\ALPHI67xx\INCLUDE\ALPHIDLL.H
Export Library Files: C:\ALPHI67xx\LIBRARY\ALPHIDLL.LIB

3.2.1 Description

AlphiDll is a 32-bit DLL for Windows™, allowing for high level control of all of ALPHI Technology's PCI cards. It is provided as a more understandable interface than that of the IOCTL interface to one of the drivers.

The DLL is intended to provide the base functionality, which is common to all of our PCI cards. Specific functionality applicable to only certain cards is provided by separate DLLs, which rely upon this DLL.

This DLL allows the following tasks to be performed.

- Finding out what cards are present and their capabilities.
- Downloading DSP code to RAM, executing and stopping under HOST control.
- Communicating with the DSP via mailbox registers and FIFO registers.
- Supporting HOST access to card resources
- Downloading DSP code and burning into FLASH for standalone BOOT.
- Interrupting the HOST processor.
- Uniquely identifying board resources (by a user ID) when identical boards exist in a HOST.
- Utilizing PCI Bus Mastering DMA to stream data between the DSP and the HOST.
- On PLX designs, allowing the DSP to use HOST memory for a shared buffer.

3.2.2 API

The interface to the AlphiDll is provided via a set of exported functions. The interface is described in the documentation for AlphiDll, provided both in manual form and as a Windows Help file.

The DLL is intended to be accessible by almost any programming language which supports the Windows™ API. In particular, functions are exported in C++ and C, and the appropriate header files are included. Other languages may require that a header file be created for that language.

This DLL was compiled under Microsoft Visual C/C++ Version 6.0.

3.2.3 Linking

An import library named ALPHIDLL.LIB can be found in the library directory. By reading the documentation from your development system, you should be able to add this file to your list of files to link with. This should resolve any function references.

In some cases (notably Borland compilers) the COFF file format is not compatible with the linker. In this case, use the documentation provided (and the DEF file) to create a compatible OMB lib file.

3.3 ALPHIERRORCODE – DLL to help display meanings of error codes

Applicable To: All PCI, CPCI, and PMC Products
Executable File: C:\ALPHI67xx\ALPHIERRORCODE.DLL
Source Location: C:\ALPHI67xx\HOST EXAMPLES\ALPHIERRORCODE
Include File: C:\ALPHI67xx\INCLUDE\DISPLAYERRORCODE.H,
C:\ALPHI67xx\INCLUDE\ERRORCODE.H
Export Library File: C:\ALPHI67xx\LIBRARY\ALPHIERRORCODE.LIB

3.3.1 Description

ALPHIERRORCODE is a DLL which helps to display the text message associated with a particular HRESULT returned from the AlphiDll. There are also some macros in DisplayErrorCode.h to assist with this. Some of the examples demonstrate how to do this, such as HostBug.exe.

The actual list of error codes which are defined is in ERRORCODE.H along with the text output from that message.

Any system defined error codes will output the associated system message.

3.3.2 Linking

An import library named ALPHIERRORCODE.LIB can be found in the library directory. By reading the documentation from your development system, you should be able to add this file to your list of files to link with. This should resolve any function references.

In some cases (notably Borland compilers) the COFF file format is not compatible with the linker. In this case, use the documentation provided (and the DEF file) to create a compatible OMB lib file.

3.3.3 API

The API for this DLL is documented formally in the documentation for AlphiDll. Some of the examples demonstrate how to do this, such as HostBug.exe.

Look for DisplayXxxMessageBox(), DisplayXxxToConsole(), and GetResultText().

4. HOST UTILITIES

4.1 BUSMASTERDMA – Utility to test and demonstrate ability to transfer data between DSP and host

<i>Applicable To:</i>	<i>All DSP cards</i>
<i>Executable File:</i>	<i>C:\ALPHI67xx\BUSMASTERDMA.EXE</i>
<i>Source Location:</i>	<i>C:\ALPHI67xx\HOST EXAMPLES\BUSMASTERDMA, C:\ALPHI67xx\DSP EXAMPLES\DMAEXAMPLE</i>

4.1.1 Description

BUSMASTERDMA is a software example demonstrating how to program the HOST and DSP to exchange data using bus master DMA.

BUSMASTERDMA works closely with a small DSP program called DMAEXAMPLE.OUT. The source for it is located in the DSP EXAMPLES folder.

The programs work together to simulate a source of data (like an A/D converter) and a sink of data (like a D/A converter). You can set the transfer size, rate, and such to simulate the expected performance on your system.

In the DSP, the data is created and destroyed inside interrupt routines, triggered off programmable timers. At each call, the specified number of DWORDs is processed. Because of the time to enter/leave the interrupt routine on the C3X, this time can have significant effect on throughput. This example can help make the appropriate tradeoffs in design.

A software FIFO is implemented in the DSP (in both directions) to help mitigate the effects of latency in processing the DMA transfers.

All in all, this example outlines how to stream data to/from the card.

4.1.2 Syntax

BUSMASTERDMA

4.2 DOWNLOAD – Download and execute DSP code

<i>Applicable To:</i>	<i>All DSP cards</i>
<i>Executable File:</i>	<i>C:\ALPHI67xx\DOWNLOAD.EXE</i>
<i>Source Location:</i>	<i>C:\ALPHI67xx\HOST EXAMPLES\DOWNLOAD</i>

4.2.1 Description

DOWNLOAD is a command line utility to download and execute DSP code on the target card. It also serves as an example of how to perform the same tasks in the customer's code.

4.2.2 Syntax

DOWNLOAD [0-9] filename

Where *filename* is the name of the DSP OUT file to be downloaded, and the digit specifies which board of several to download to. If the digit is not specified, board 0 is used.

4.2.3 Notes

Download works with any of the boards with DSPs by opening the device by number *n* where *n* is the digit passed on the command line.

4.3 HOSTBUG – Utility to access and test card resources, enumerate system

Applicable To: All PCI, CPCI, and PMC Products
Executable File: C:\ALPHI67xx\HOSTBUG.EXE
Source Location: C:\ALPHI67xx\HOST EXAMPLES\HOSTBUG

4.3.1 Description

HOSTBUG is a command line debugger-type utility to allow for direct access to card resources by the host. It also serves as an example of how to perform some of the same tasks in the customer's code, including enumerating available cards on the HOST. It is also used extensively at the factory for shaking out new hardware and compatibility issues.

It has a debugger type interface, in the spirit of the old DOS DEBUG.COM debugger. It allows for access to any one pass-through region of any one card. Commands allow for the changing of values at locations, displaying memory, several memory tests and tight access loops.

The debugger interface is closely related to that of the Bootloader.

4.3.2 Syntax

HOSTBUG

HOSTBUG will enumerate the devices found on the system, and present the command prompt.

4.3.3 Available Commands

4.3.3.1 Select – Select a device to connect to

The select command allows for the connection of several cards in the system and to display a list of available cards.

SELECT [n]

Where *n* is the device number to switch to. If *n* is not given, only list the available devices. The current device is part of the command line prompt.

4.3.3.2 AS - Address Space of this card to connect to

The Address Space command allows for the connection to any of the available card resources available on the current card and to display a list of available resources.

AS [n]

Where *n* is the region number to switch to. If *n* is not given, only list the available regions. The current region is part of the command line prompt.

Available regions depend upon the type of device, but include the following:

- PLX Registers
- Dual Port RAM
- IP ID, IO, and Memory spaces for up to 4 IPs
- The SUMMIT registers (on 1553 cards)
- HOST Control Regions of many cards.

4.3.3.3 PAGE – Select method of decoding IP memory on IPM designs

The Page command is used on IPM style boards to switch the IP mapping of memory space. Some improvements have been made on this second generation IP carrier allowing software control of 16/32 memory path, as well as mapping multiple IP memory spaces into the available space.

PAGE [*n*]

Where *n* selects one of the following:

1	IP A memory space is mapped into the unified memory space in 16 bit mode.
2	IP B memory space is mapped into the unified memory space in 16 bit mode.
3	IP C memory space is mapped into the unified memory space in 16 bit mode.
4	IP D memory space is mapped into the unified memory space in 16 bit mode.
5	IPs A and B memory space is mapped into the unified memory space in 32 bit mode.
6	IPs C and D memory space is mapped into the unified memory space in 32 bit mode.
7	The first 2 MB of IPs A - D memory space is mapped into the unified memory space in 16 bit mode.
8	The first 4 MB of IPs A - D memory space is mapped into the unified memory space in 32 bit mode.

When two IPs are used in 32 bit mode, IP B or D is the upper 16 bits, IP A or C is the lower.

In mode 7, only the first 2 MB of each IP space is available. IP B is mapped at offset 0x400000, IP C is mapped at 0x800000, and IP D is at 0xC00000.

In mode 8, only the first 4 MB of each IP space is available. IP A – B is mapped at offset 0, IP C – D is mapped at 0x800000.

4.3.3.4 WL – Write Loop

The Write Loop command is a convenient way of outputting a series of values to the card in a tight loop. This is useful for checking the hardware timing of a card or an IP. It is also useful in testing some IPs. Access can be made in BYTE, WORD, and DWORD modes. Up to 10 locations can be written in the loop.

WL [*dwb*] *addr data* [[*dwb*] *addr data* ...]

Where *addr* is the hex address to write and *data* is the value to be written.

'd' is a 32-bit write, 'w' is a 16-bit write, and 'b' is an 8 bit write. 'd' is the default.

Address is based on offset into the appropriate access region and is decoded by bytes. For example, the following command will write the first two DWORDS in a loop:

WL 0 ffffffff 4 ffffffff

Press any key to exit the loop.

4.3.3.5 RL – Read Loop

The Read Loop command is a convenient way of inputting a series of values from the card in a tight loop. This is useful for checking the hardware timing of a card or an IP. Access can be made in BYTE, WORD, and DWORD modes. Up to 10 locations can be read in the loop. The read values are not displayed.

RL [*dwb*] *addr* [[*dwb*] *addr* ...]

Where *addr* is the hex address to write. 'd' is a 32-bit write, 'w' is a 16-bit write, and 'b' is an 8-bit write. 'd' is the default.

Press any key to exit the loop.

4.3.3.6 MM – Memory Modify

The Memory Modify command is useful for directly accessing and modifying locations in the card resources.

MM [dwb] addr

Where *addr* is the hex address to display. 'd' is a 32-bit write, 'w' is a 16-bit write, and 'b' is an 8-bit write. 'd' is the default.

HOSTBUG displays the address and read value at the address.

If '+' or ENTER is pressed, HOSTBUG does not modify the location, and goes to the next location.

If '-' is pressed, HOSTBUG does not modify the location, and goes to the previous location.

If the space bar is pressed, HOSTBUG does not modify the location, and stays at the same location, rereading the location.

If a new hexadecimal value is entered before any of the above characters, the location is changed to the new value and confirmed first.

If the updated contents do not match, a warning is given.

A '.' alone will exit the command.

4.3.3.7 MD – Memory Display

The Memory Display command is useful to display a series of locations in a convenient format.

MD addr count

Where *addr* is the starting address and *count* is the number of DWORDS to display. Values are entered in hexadecimal.

Memory contents are displayed in hexadecimal and ASCII equivalent.

4.3.3.8 MT – Memory Test

The Memory Test command is useful for testing card resources such as Dual Port RAM or IP memories to ensure correct operation.

There are 4 different memory tests available, and they can be run in DWORD, WORD, and BYTE modes.

Tests can be repeated until stopped, and they can be run in quiet mode with only summaries printed at test termination.

The first incorrect read will terminate that particular test.

The four tests are as follows:

- Walking Ones and Zeros. Useful for finding stuck data bits.
- Stray Write. Will find if writing to one location inadvertently changes another location too. Takes significantly longer to execute than other tests.
- Power Supply. Alternately writes 0 and 0xffffffff to ensure adequate power supply bypass is available. Also ensures that back to back writes work correctly.
- Address Read back. Helps to localize stuck address lines.

The syntax of the command is as follows:

MT [1spa*bwkrq] addr size

Where *addr* is the starting address and *size* is the number of DWORDs, WORDs or BYTEs depending on the size.

The other parameters have the following meaning:

- 1 Enable the Walking Ones and Zeros test.
- s Enable the Stray Write test.

- p Enable the Power Supply test.
- a Enable the Address Read back test.
- * Enable all four memory tests.
- b BYTE mode.
- w WORD mode.
- k Skip alternate targets (useful when testing IP memory in 16 bit mode)
- r Repeat until stopped.
- q Run quietly only reporting statistics at the end of the test.

4.3.3.9 MTR – Memory Test; Stop All Tests Immediately On Failure

This command functions almost identically to MT above, except that at the first failure, the test ends. This way the factory can catch rare problems on the logic analyzer.

4.3.3.10 QUIT - Exit the utility

Typing

QUIT

Will exit the utility.

4.3.3.11 HELP or ? – Display help screen

Typing

HELP

or

?

will list the available commands. Specific help for a command is given by

HELP *command*

or

? *command*

Where *command* is the name of the command to display help on.

4.4 INTERRUPT – Utility to test and demonstrate ability to interrupt host

Applicable To: All DSP cards
Executable File: C:\ALPHI67xx\INTERRUPT.EXE
Source Location: C:\ALPHI67xx\HOST EXAMPLES\INTERRUPT, C:\ALPHI67xx\DSP EXAMPLES\INTLOOP

4.4.1 Description

INTERRUPT is a software example demonstrating how to program the HOST to receive interrupts generated by the DSP.

INTERRUPT works closely with a small DSP program called INTLOOP.OUT. The source for it is MAIN.C. INTLOOP.OUT stays in a loop checking for any values written by the HOST to any of the mailboxes. If the host writes to mailbox 1, the value is then written back to mailbox 1. Similarly, any value written to mailbox 2 is written back to mailbox 2.

INTERRUPT operates in the following manner. The INTLOOP.OUT program is downloaded to the DSP and started. A pointer to the interrupt handler InterruptReceived is passed to HookMailboxInterrupt function of AlphaDll. When the DSP writes to mailbox 1, a HOST interrupt will be generated and InterruptReceived will be called to handle it.

Finally, a thread is created to run the function Ping, which writes to mailbox 1 once a second.

When InterruptReceived is called, the passed value is printed out. Before the value is printed out, though, it is resent to the DSP in mailbox 2. The latency time was measured by Code Composer between the two writes to the mailboxes in the DSP.

4.4.2 Syntax

INTERRUPT

4.4.3 Performance issues

I have profiled the interrupt latency on a Pentium 166 INTEL motherboard at less than 400 μ S. This requires that the task priority class be elevated to real-time. Otherwise, other events and tasks running on XP will cause the thread calling the user's completion routine to not run immediately after the XP driver unblocks it.

Make sure that the time to execute the completion routine is less than the time between interrupts, or XP will hang.

4.4.4 Notes

It is important to note that the completion routine must return before the thread can call it again. No interrupts will be lost as a result of taking too long, but the latency will obviously be high.

As written, this example will only operate with device 0.

4.5 PFLASH – FLASH programming utility

Applicable To: All DSP cards
Executable File: C:\ALPHI67xx\PFLASH.EXE
Source Location: C:\ALPHI67xx\HOST EXAMPLES\PFLASH

4.5.1 Description

PFLASH is a command line utility to download DSP code to the FLASH device on the target card. It also serves as an example of how to perform the same tasks in the customer's code.

The image to be downloaded must be in a TEKTRONIX file format, such as that generated by TI's HEX6X utility. See the chapter on compiling DSP code in this manual for more details.

4.5.2 Syntax

PFLASH [-[b][0-9]] filename

where *filename* is the name of the DSP image file to be downloaded, and the digit specifies which board of several to download to. If the digit is not specified, board 0 is used.

By default, the user area of the FLASH is programmed. The card can be configured to automatically start the customer's application at reset. See the chapter on the PLX9056 editor and programmer for more details.

If the -b option is used, the boot area of the FLASH device is burned instead of the user area. It is not recommended that customers replace the boot program on the card. This option can be used, however, to update the bootloader in the field.

4.5.3 Notes

The PFLASH utility outputs a series of periods as a progress indicator. Boot loaders inform PFLASH when the programming is complete and return the status of the burn

Board Support Package for TMS67xx DSPs and Windows XP SOFTWARE MANUAL

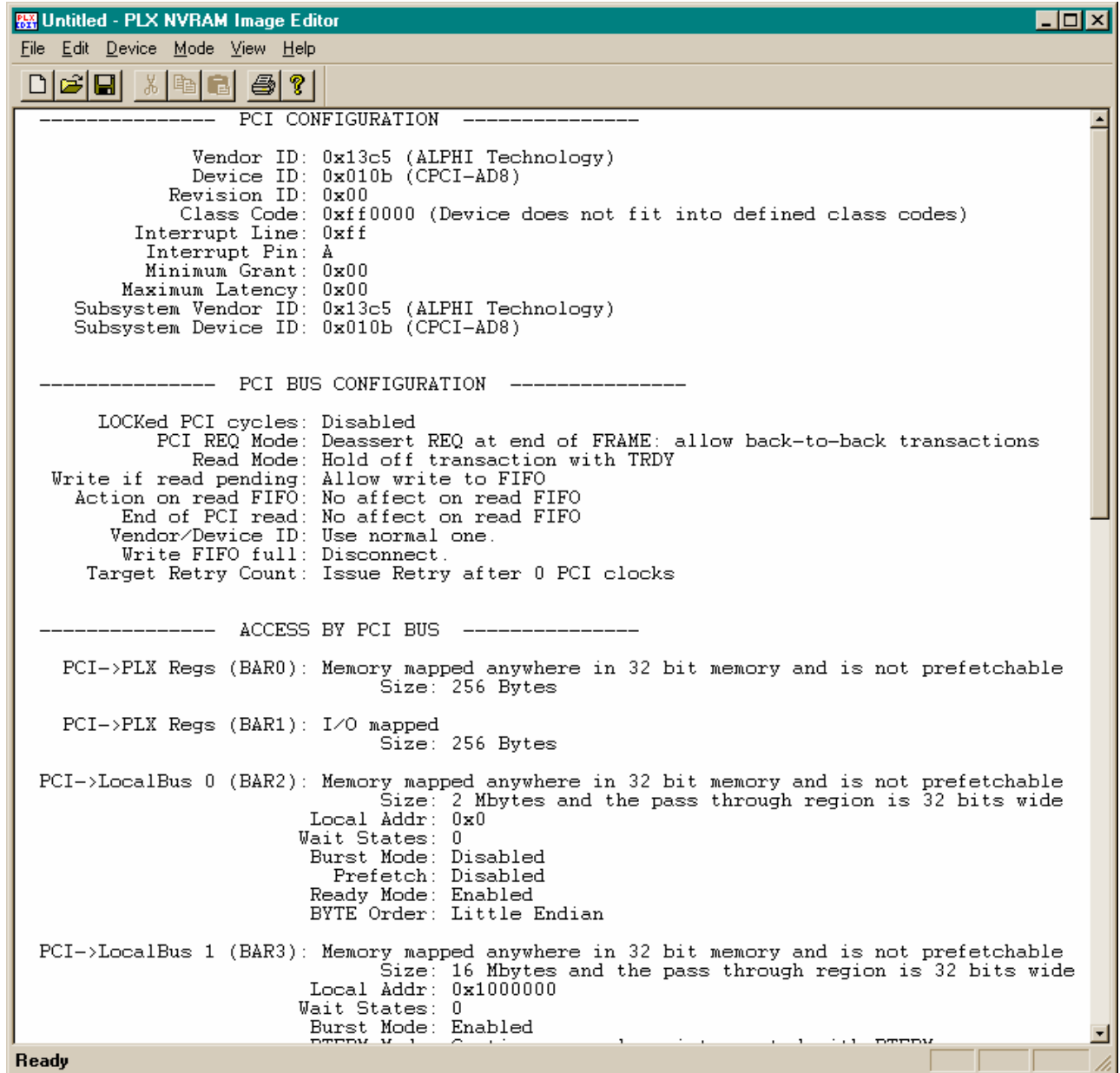
If the customer is using a PCI system to update a card normally run in stand-alone mode, remember to set the jumpers correctly before placing the card in the HOST.

It is not necessary to reconfigure the card's boot mode to run PFLASH. When the card is opened by AlphiDII, the DSP is reset in a special mode to force command mode.

When the -b option is used, the DSP is not reset, to facilitate the initial loading of the bootloader by emulator at the factory.

4.6 PLX9056 - PLX NVRAM Image Editor and Programmer

Applicable To: All PLX9056 based PCI, CPCI, and PMC Products
Executable File: C:\ALPHI67xx\PLX9056.EXE
Source Location: Not included in package



4.6.1 Description

PLX9056 is a windowed utility to edit and program the NVRAM device which configures the PLX 9056 PCI interface chip and provides certain other options.

In addition to the configuring the PLX PCI interface, it allows for configuring the following hardware and DSP features:

- User's Identification for the board. A 32 bit decimal integer is available for sole use by the customer to identify multiple instances of a particular card uniquely.
- Hardware Serial Number and Revision. These fields allow the customer to track configuration and testing issues more easily.
- Identify custom options for the card. Certain card options for specific customers, such as the size of the Dual Port Ram on applicable boards, or the clock frequencies of the DSP or the PCLK given to the SCC8530 serial interface are configurable. The DSP is able to determine these values through the Lib6713 library.
- Mode of operation at reset. The DSP can be set to automatically boot a customer's application stored in FLASH, along with several other options.

4.6.2 Starting the Editor

The PLX editor either by selecting the entry in the Start menu or by typing the following at a command prompt:

```
PLX9056
```

When the editor is started in this manner, the image settings default to a common starting point for creating new images. The customer should immediately select either **Device / Read from Device** or **File / Open** to load an image. It is recommended that the customer use either the image in the device or a default file image as a starting point for any changes.

An alternative way to start the editor is to double click on a PLX image file (*.PLX), or to type the following at a command prompt:

```
PLX9056 filename
```

Where *filename* is the desired image file to load.

4.6.3 Display

The editor displays a report based on the image loaded in the editor. The report is divided into six sections. A brief description of each section is as follows:

4.6.3.1 PCI CONFIGURATION

This section includes most of the PCI configuration space required by the PCI specification. Of primary interest to the **Board Support Package for TMS67xx DSPs and Windows XP** is the settings for the Vendor ID and the Device ID. These two settings serve as the primary means of identifying the board type. Other parameters are important to the HOST system configuration, and do not need to be changed. See the PLX documentation and the PCI specification for details.

4.6.3.2 PCI BUS CONFIGURATION

There are some configuration parameters that may be of some use in fine-tuning a system to match the PCI bus. However, for most applications, there is no need to adjust any of these parameters.

4.6.3.3 ACCESS BY PCI BUS

This section includes the options required to communicate with the card. The device driver and the card hardware depend upon these setting being correct.

The **Board Support Package for TMS67xx DSPs and Windows XP** depends upon the card resources being memory mapped in order to allow the customer the fastest means of communication with the card. However, for customers who are using alternative operating systems or hardware which may not fully support the PCI specification and map memory resources, it is possible to I/O map the card resources in many cases. Contact the factory for more details.

Part of these settings are dependent on the card hardware. Other parameters are important to the HOST system configuration. See the PLX documentation and the PCI specification for details.

4.6.3.4 LOCAL BUS->PCI

The initial values for the passthru from DSP to the PCI bus are defined here. Most of the decoding is set up in the DSP library Lib6713 library calls (defined in Pci.c). Some of the parameters may have an effect on this behavior.

4.6.3.5 LOCAL BUS CONFIGURATION

This section includes some parameters which affect how the DSP Local Bus is utilized by the PLX. For the most part, these settings do not need to be changed. Of interest, the latency and pause timers can affect the percentage of time that the PLX has to use the local bus. See the PLX documentation for details.

4.6.3.6 DSP AND HARDWARE OPTIONS

This section includes the new image fields which are defined by ALPHI Technology. Some of the fields which are configurable are listed in the **Description** section above. These fields are also available to the HOST software via AlphiDll and to the DSP via the Lib6713 library. Certain modes affect the processing of the Bootloader, and are documented in that chapter.

4.6.4 Printing and File Options

The editor loads and saves PLX9056 images in a binary file format. **File / New**, **File / Open**, **File / Save**, and **File / Save As** work as they would in any Windows application.

A text copy of the report displayed can be saved via the **File / Save Report As** option. This file can be E-mailed to Technical Support or included in user documentation.

The report can be sent to a Windows printer via the **File / Print** option.

4.6.5 Mode of Operation

The editor operates in one of three modes. The mode of operation is selected by the menu **Mode**.

4.6.5.1 Basic Mode

Basic mode allows the user to change the most useful fields configuring the card without risking changing anything which will grossly affect the card operation.

The only fields which are configurable are in the DSP and Hardware Options section and are the User's ID and the Boot Option at Reset.

A customer desiring to change either of these fields should read the present image from the device, change the desired field, and write the image back to the device.

4.6.5.2 Advanced Mode

Advanced mode allows the user to change almost all of the fields configuring the card. There is a possible risk of changing something which will grossly affect the card operation.

The only fields which are locked out are in the DSP and Hardware Options section and are the Serial Number and the Hardware Revision fields.

A sophisticated customer desiring to change a PCI configuration field should start with the present image from the device or an image from the file as a starting point for changes.

4.6.5.3 Factory Only

Factory Only mode adds to **Advanced** mode the ability to set the Serial Number and Hardware Revision fields. There is a possible risk of changing something which will grossly affect the card operation. **Factory Only** mode is locked out from use by the customer.

4.6.6 Changing values

The field values are easily changed via a set of dialog boxes under the **Edit** menu. The dialog boxes correspond to the sections. Some fields of the dialog boxes may be disabled depending upon the mode of operation.

Depending upon the field, documentation can be found in the following sources:

- PLX PCI 9080 Data Sheet
- PCI Specification
- Bootloader chapter of this document
- AlphiDll chapter of this document

4.6.7 Reading from and writing to the device

If a card is present on the system and was found by the AlphiPci device driver, it is possible to read and write the NVRAM of the device. Otherwise the menu entries under **Device** will be disabled.

The user can select one card from a list of available devices. The user can choose to read the image from the NVRAM device, or to write the image from the editor into the NVRAM device.

When writing, special processing is performed based upon the Serial Number and Hardware Revision fields.

In order to not overwrite the Serial Number and Hardware Revision fields set at the factory, the editor will compare the settings loaded in the editor with those already in the card. A serial number of 0 indicates that no serial number or hardware revision is set.

If the image loaded in the editor has a valid non-zero serial number and it does not match the valid serial number previously written into the card, the user is prompted on write to choose a Serial Number and Hardware Revision to use. If the user is deliberately updating the Serial Number and Revision, he should select the image. Otherwise, the value on the card should be kept.

4.6.8 Notes

On some systems, it is necessary to cycle the power after changing the PLX configuration in order to force a reload of the NVRAM image. This is related to a hardware issue on these systems, which do not propagate PCI Reset across a PCI-PCI bridge correctly.

4.7 REPORTVERSION – Software configuration management tool

Applicable To: All PCI, CPCI, and PMC Products
Executable File: C:\ALPHI67xx\REPORTVERSION.EXE
Source Location: C:\ALPHI67xx\HOST EXAMPLES\REPORTVERSION

4.7.1 Description

REPORTVERSION is a command line utility to help technical support deal with software configuration issues at customer sites. It also serves as an example of how to perform the same tasks in the customer's code.

The software components which are executables on the HOST that are distributed with the **Board Support Package for TMS67xx DSPs and Windows XP** are tagged with a version identification resource to help keep track of the configuration on the target machine. Windows XP knows about this version resource and has an API designed to manipulate it that is part of the operating system.

For example, if you open the C:\ALPHIPCI directory in Explorer, and right click on AlphiDll.dll, and select Properties, there is a tab called Version, which displays the contents of this resource.

REPORTVERSION uses the Windows API functions to retrieve the version resources of the subcomponents of the package and displays it to the screen.

Additionally, any hardware cards present in the system are enumerated, and the bootloader version, hardware serial number, and hardware version information is displayed, if present.

4.7.2 Syntax

`REPORTVERSION | MORE`

to display to the screen a page at a time.

`REPORTVERSION > output.txt`

to copy to the file output.txt to display in an editor or E-mail to the factory for support.

4.7.3 Notes

ALPHIPCI.SYS, ALPHIDLL.DLL, and AD16DLL.DLL have always had a version resource. Starting with revision D installation disks, all the examples have version resources. Bootloader version 1.7 and later support querying the version via a new command. Serial numbers and hardware revisions are a recent addition to the NVRAM of the cards.

New cards and cards returned for repair will be programmed with serial numbers, and hardware version information. If desired by the customer, it is possible to field upgrade this information on old cards.

4.8 SHAREDHOSTMEMORY – Utility to demonstrate sharing HOST memory with a PLX based design

Applicable To: All PLX based cards
Executable File: C:\ALPHI67xx\SHAREDHOSTMEMORY.EXE
Source Location: C:\ALPHI67xx\HOST EXAMPLES\ SHAREDHOSTMEMORY

4.8.1 Description

SHAREDHOSTMEMORY is a software example demonstrating how to program the HOST to allow the DSP to use a portion of the HOST memory, as well as make that memory shared with the HOST.

SHAREDHOSTMEMORY works by creating a page-aligned buffer that is to be shared. It then makes the memory available to the DSP. Then it loads a modified version of the BOOTROM called (not surprisingly) TESTBOOTROM. This version adds extra commands to access the PCI space and the HOST shared buffer.

By utilizing the SM command on the serial port of the card, this shared buffer can be accessed by the DSP.

4.8.2 Syntax

`SHAREDHOSTMEMORY`

5. DSP SUPPORT LIBRARY (Lib6713)

5.1 PLX / TMS67xx based hardware

<i>Applicable To:</i>	<i>All DSP based cards using PLX and TMS320C32</i>
<i>Library File:</i>	<i>C:\ALPHI67xx\LIB6713\C32_PLX\LIB6713.LIB, C:\ALPHI67xx\LIB6713\C32_PLX\LIB6713G.LIB, C:\ALPHI67xx\LIB6713\C32_PLX\LIB6713R.LIB, C:\ALPHI67xx\LIB6713\C32_PLX\LIB6713GR.LIB</i>
<i>Source Location:</i>	<i>C:\ALPHI67xx\LIB6713\C32_PLX\LIB6713.SRC</i>

The Lib6713 library is a collection of DSP routines, which provide a common base of functionality. It provides the following features:

- Common access routines for communications with the host by Mailbox, FIFO, and Bus Master DMA
- Mapping C language standard input and output by the DSP to the onboard 8530 or equivalent serial chip
- Identify the applicable card resources and parameters
- On PLX designs, allow direct access by DSP to PCI bus, including shared memory buffer on HOST.

Full source code is provided in a TI compatible source library suitable for processing by the MK30 utility supplied by TI, and it comes precompiled for stack and register passing models, in debug and release forms. It can easily be recompiled for other models as well. A batch file called build_lib.bat demonstrates how to recompile the DSP library.

The Standard IO routines require that the TI compiler 5.0 or later be used, as there was no support for Standard IO prior to that release.

The R versions of the library support the register passing model of the DSP C Compiler. The non R versions support stack passing. The G versions incorporate debugging symbols.

The source is provided in a source library. Use TI's AR30 utility with the x command to extract the files.

The interface to the library is fully described in separate documentation for Lib6713, provided both in manual form and as a Windows Help file.

6. GENERAL PURPOSE DSP UTILITIES AND EXAMPLES

6.1 DSP BOOTLOADER / DEBUGGER

Applicable To: All DSP based cards
Executable File: C:\ALPHI67xx\DSP EXAMPLES\BOOTROM \Release\BOOTROM.out
Source Location: C:\ALPHI67xx\DSP EXAMPLES\BOOTROM

6.1.1 Description

The Bootloader is the first DSP program run when the card comes out of reset. It is loaded into static RAM from either the boot area of the FLASH device or the EPROM, depending on the type of card.

The Bootloader provides the following major features:

- Ability to download and execute a TI COFF formatted DSP program via HOST interface.
- Ability to download and execute a TEKTRONIX formatted DSP program via HOST interface or serial cable.
- Ability to download and burn into the user area of FLASH a TEKTRONIX formatted DSP program for stand-alone operation via HOST interface or serial cable.
- Ability to automatically load and execute a user's program from FLASH at RESET.
- Ability to directly access card resources using a debugger interface via the serial port.

6.1.2 HOST Control Mode and the HOST Interface

The Bootloader communicates with the HOST by means of a series of messages passed through the mailboxes of the PLX PCI interface chip. Details about the types of messages and their format are described in the documentation for ALPHiDll, provided both in manual form and as a Windows Help file.

6.1.3 Serial Port

Many ALPHI Technology cards have an SCC8530 chip for use as a serial port. Select boards have implemented a serial port within the programmed FPGA. Presently, all cards with DSPs except the CPCI-AD16 have a serial port available.

This port can be used to control the card using the Command Mode interface.

The bootloader (by means of the Lib6713 Library) communicates with the user using the following parameters:

19200, N, 8, 1

Neither hardware nor software flow control is used.

Appropriate cables for the serial connection are available, and are normally provided with the card. Contact the factory for more information.

6.1.4 Boot Modes and Reset

One field in the NVRAM Image tells the bootloader what to do at RESET. The following modes are available for default behavior at RESET:

- Command Prompt (with HOST Control)
- Menu Mode (with HOST Control)
- User's application in FLASH after 5 second delay
- User's application in FLASH without delay
- HOST Control only (no output to serial port)

This Boot Mode can be set in one of two ways:

- Via the PLX9056 Editor and Programmer
- Via the Command Mode command BM (Boot Mode)

For details on the PLX Editor, see the appropriate chapter in this manual.

There is a mechanism by which AlphaDII and the Bootloader can force the Bootloader into a mode which will accept HOST commands and not boot the user code in FLASH, even if that is the setting.

6.1.5 BootROM Command Mode

Command mode is a command line debugger-type interface to allow for direct access to card resources by the DSP. It is also used extensively at the factory for shaking out new hardware, locating compatibility issues, and for testing production boards.

It has a debugger type interface. Commands allow for the changing of values at locations, displaying memory, several memory tests and tight access loops. User programs in TEKTRONIX format can be downloaded, executed, and burned into FLASH memory.

The debugger interface is closely related to that of HOSTBUG. The following commands are available.

6.1.5.1 D – Download TEKTRONIX file into RAM

The Download command is the only means of loading a DSP program via the serial port. It is most useful when the card is used in stand-alone mode, and it is inconvenient to bring the card to an Windows™ host for programming.

Once the program image has downloaded to the RAM, it can either be burned into FLASH memory using the BURN command or it can be immediately executed via the G command.

To use this command, type “D” and return.

The Bootloader will prompt for the image. Using your terminal program, download the X0 file to the card. The command prompt will return when the download is complete.

The Bootloader will type a series of periods to the screen, one for each line of the file, as a status indicator.

The user can then immediately execute the program with the “G” command or burn it into the FLASH with the “BURN” command.

6.1.5.2 BURN – Burn downloaded file into user area of FLASH

The BURN command is the only means of updating the FLASH contents via the serial port. It is most useful when the card is used in stand-alone mode, and it is inconvenient to bring the card to an XP host for programming.

After the user has downloaded a TEKTRONICS file to the RAM, this command will burn the image into the user area of the FLASH.

To use this command, type “BURN” and return. The Bootloader will ask you to confirm the command.

The Bootloader will type a series of periods to the screen, one for each FLASH sector programmed, as a status indicator.

The command prompt will come back after programming. Use the “B” command to load the newly burned program.

6.1.5.3 BTBURN – Burn downloaded file into boot area of FLASH

The BTBURN command is used at the factory for the initial loading of the Bootloader code. The customer may also use it to update the Bootloader version in the field. Be prepared with an emulator in case the FLASH burn fails, or the board may become inoperable.

After the user has downloaded the new version of the bootloader file to the RAM, this command will burn the image into the boot area of the FLASH.

To use this command, type “BTBURN” and return. The Bootloader will ask you to confirm the command.

The Bootloader will type a series of periods to the screen, one for each sector programmed, as a status indicator.

The command prompt will come back after programming. Cycle the power, or short the RESET pins to load the new Bootloader version.

6.1.5.4 B – Boot the user’s program stored in FLASH user area

The B command is used to tell the Bootloader to load the program stored in the user area of the FLASH and to begin executing it.

To use this command, type “B” and return. The Bootloader will immediately load and execute the program.

6.1.5.5 G – Execute the TEKTRONICS program downloaded into RAM

The G command is used to tell the Bootloader to load the program downloaded into RAM and to begin executing it.

To use this command, type “G” and return. The Bootloader will immediately load and execute the program.

6.1.5.6 R – Read Memory

The Read Memory command allows a simple means to read one DSP location without using the Memory Modify command.

R *addr*

Where *addr* is the address to read.

6.1.5.7 W – Write Memory

The Write Memory command allows a simple means to write one DSP location without using the Memory Modify command.

W *addr value*

Where *addr* is the address to write, and *value* is the value to written. No additional cycles (such as confirming reads) are performed.

6.1.5.8 RL – Read Loop

The Read Loop command is a convenient way of inputting a series of values from the card in a tight loop. This is useful for checking the hardware timing of a card or an IP. Since the DSP always deals with 32 bit DWORDS, this is the only access mode supported. Up to 10 locations can be read in the loop. The read values are not displayed.

RL *addr [addr ...]*

Where *addr* is the hex address to read.

Press any key to exit the loop.

6.1.5.9 WL – Write Loop

The Write Loop command is a convenient way of outputting a series of values to the card in a tight loop. This is useful for checking the hardware timing of a card or an IP. It is also useful in testing some IPs. Since the DSP always deals with 32 bit DWORDS, this is the only access mode supported. Up to 10 locations can be written in the loop.

WL *addr data [addr data ...]*

Where *addr* is the hex address of the DSP location to write and *data* is the value to be written.

Address is based on offset into the appropriate access region and is decoded by bytes. For example, the following command will write the first two DWORDS of the user area of the static RAM in a loop:

```
WL 6000 ffffffff 6001 ffffffff
```

Press any key to exit the loop.

6.1.5.10 RNV – Read NVRAM location

The Read NVRAM Location command is most useful as a tool for software debugging. Note that the PLX9056 Editor provides a much better means of programming and viewing the NVRAM.

The syntax is as follows

```
RNV addr [addr ...]
```

where *addr* is the hex address of the NVRAM 16-bit location to read. The address 0 is the first 16-bit word, 1 is the second, etc...

6.1.5.11 WNV – Write NVRAM location

The Write NVRAM Location command is useful as a tool for software debugging. Note that the PLX9056 Editor provides a much better means of programming the NVRAM.

The syntax is as follows

```
WNV addr data [addr data ...]
```

where *addr* is the hex address of the NVRAM 16-bit location to write and *data* is the 16-bit value to be written. The address 0 is the first 16-bit word, 1 is the second, etc...

WARNING: the values in this NVRAM are used by the PLX and modifying the wrong values can prevent the board to be recognized by the system, and can even prevent it from booting.

6.1.5.12 EEPROMDEFAULT – Program the EEPROM with default values

This command detects which kind of board is being used and program the NVRAM with the default values for that board.

The syntax is as follows

```
EEPROMDEFAULT
```

6.1.5.13 INFO – Display a report about the board configuration

The INFO command is useful for outputting a report about the current hardware configuration. All of this information is extracted by the Lib6713 library.

The report includes the following information:

- Vendor ID
- Device ID
- Size of the static RAM
- Size and location of the Dual Port RAM (if present)
- DSP Clock speed
- Type and Speed of any serial device present
- Number of IP slots available to the DSP, and the locations of their access regions
- The boot option from the last RESET
- The User's ID value
- Device serial number, and hardware and FPGA revision numbers.

6.1.5.14 MM – Memory Modify

The Memory Modify command is useful for directly accessing and modifying locations in the DSP.

MM *addr*

Where *addr* is the hex address to display.

The Bootloader displays the address and read value at the address.

If '+' or ENTER is pressed, the Bootloader does not modify the location, and goes to the next location.

If '-' is pressed, the Bootloader does not modify the location, and goes to the previous location.

If the space bar is pressed, the Bootloader does not modify the location, and stays at the same location, rereading the location.

If a new hexadecimal value is entered before any of the above characters, the location is changed to the new value and confirmed first.

If the updated contents do not match, a warning is given.

A '.' alone will exit the command.

6.1.5.15 MD – Memory Display

The Memory Display command is useful to display a series of locations in a convenient format.

MD *addr count*

where *addr* is the starting address and *count* is the number of DWORDS to display. Values are entered in hexadecimal.

Memory contents are displayed in hexadecimal and ASCII equivalent.

6.1.5.16 MF – Memory Fill

The Memory Search command is useful to fill a memory area with a pattern.

MS *beginaddr endaddr data*

where *beginaddr* is the starting address, *endaddr* is the ending address and *data* is the value to search for. Values are entered in hexadecimal.

If the data field contains a "*", the memory location is filled with its own address. This can be useful to find addressing problems in the memory.

6.1.5.17 MS – Memory Search

The Memory Search command is useful to find a pattern in a memory area.

MS *beginaddr endaddr data*

where *beginaddr* is the starting address, *endaddr* is the ending address and *data* is the value to search for. Values are entered in hexadecimal.

6.1.5.18 MT – Memory Test

The Memory Test command is useful for testing card resources such as Dual Port RAM or IP memories to ensure correct operation.

There are 4 different memory tests available, and they can be masked off for DWORD, WORD, and BYTE accesses.

Tests can be repeated until stopped, and they can be run in quiet mode with only summaries printed at test termination.

The first incorrect read will terminate that particular test.

The four tests are as follows:

- Walking Ones and Zeros. Useful for finding stuck data bits.
- Stray Write. Will find if writing to one location inadvertently changes another location too. Takes significantly longer to execute than other tests.
- Power Supply. Alternately writes 0 and 0xffffffff to ensure adequate power supply bypass is available. Also ensures that back to back writes work correctly.
- Address Readback. Helps to localize stuck address lines.

The syntax of the command is as follows:

```
MT [lspa*bwrq] addr size [trig]
```

where *addr* is the starting address and *size* is the number of locations to test. If an error is found, read from *trig* to trigger logic analyzer.

The other parameters have the following meaning:

- 1 Enable the Walking Ones and Zeros test.
- s Enable the Stray Write test.
- p Enable the Power Supply test.
- a Enable the Address Readback test.
- * Enable all four memory tests.
- b BYTE mode. Use 0xff as a mask for memory comparisons.
- w WORD mode. Use 0xffff as a mask for memory comparisons.
- r Repeat until stopped.
- q Run quietly only reporting statistics at the end of the test.

6.1.5.19 BM – Select Boot Mode

The BM command allows the user to select a different Boot Mode than the current one. The PLX9056 Editor can also perform this functionality. This command is useful in stand-alone operations where it is not possible to use an XP host to configure the card.

The syntax of the command is as follows:

```
BM n
```

where *n* is the new Boot Mode. It can be one of the following:

- 0 Command Prompt (with HOST Control)
- 1 Menu Mode (with HOST Control)
- 2 User's application in FLASH after 5 second delay
- 3 User's application in FLASH without delay
- 4 HOST Control only (no output to serial port)

6.1.5.20 USER – Configure User ID

The USER command allows the user to select a unique identifier to the card. The PLX9056 Editor can also perform this functionality. This command is useful in stand-alone operations where it is not possible to use an XP host to configure the card.

The syntax of the command is as follows:

USER *n*

where *n* is the new user identifier.

6.1.6 Menu Mode

Menu mode is not supported at this time. Any cards set for menu mode will default to Command Mode.

6.1.7 User's Application Modes

When the user has selected the boot mode for User's Application after 5 seconds, the user has 5 seconds to press any key on the serial port to default the bootloader to Command Mode. If the user fails to press any key, the Bootloader will load and execute the customer's application from FLASH.

If the user has selected the boot mode for User's Application without Delay, the Bootloader will immediately load and execute the customer's application from FLASH. There will be no serial output by the Bootloader.

Be aware that in order to recover from the User's Application without Delay mode, a HOST computer running XP will be necessary. If you must perform field updates of your code, then use the 5 second delay mode.

In either case, if there is no valid boot image, Command mode will be defaulted. In either of these modes, if AlphiDll performs a RESET, then Command mode will be selected.

6.2 DMAEXAMPLE

This program is used in conjunction with the HOST example program BusMasterDma.exe. See that documentation for details about this program.

6.3 INTLOOP

This program is used in conjunction with the Interrupt.exe example program. See that documentation for details about this program.

6.4 BLINKING

Applicable To: All DSP based cards
Executable File: C:\ALPHI67xx\BLINKING.OUT
Source Location: C:\ALPHI67xx\DSP EXAMPLES\BLINKING

This very simple program outputs a single line of text to the serial port on the card. It serves as a very simple start to setting up a build system at customer sites. It also demonstrates how simple it is to use the Lib6713 library to output text for debugging.

7. BUILDING CUSTOM DSP CODE

It is fairly easy to write and debug custom applications to run on the DSP. This chapter serves as a collection of topics to be aware of when writing custom code.

7.1 Development tools for the DSP

7.1.1 Compiler and Assembler/Linker

For middle to complex DSP projects, ALPHI Technology recommends that the customer purchase a debugger and emulator. The emulator is a hardware device installed in a development machine that connects to the DSP's emulator port and allows for fully controlling the processor. The debugger is a software program, which allows for full source debugging of the DSP code via the emulator.

ALPHI Technology strongly recommends TI's Code Composer debugger running with a Spectrum Digital DSP emulator (model XDS510PP MPD). See the **Reference Material List** for more details.

For small DSP coding efforts, it is possible to use the serial port and the Lib6713 library as a standard output stream for "printf" type debugging. See the BLINKING for the typical C "Hello, World!" example.

7.2 Lib6713 Library

The Lib6713 library is a collection of DSP routines, which provide a common base of functionality. It provides the following features:

- Common access routines for communications with the host by Mailbox and FIFO
- Mapping C language standard input and output by the DSP to the onboard 8530 serial chip
- Identify the applicable card resources and parameters
- On PLX designs, allow direct access by DSP to PCI bus, including shared memory buffer on HOST.

Full source code is provided, and it comes precompiled for stack and register passing models, in debug and release forms. It can easily be recompiled for other models as well.

7.3 Compiling and Linking

There are several batch files distributed with the DSP examples, which offer a starting point for choosing compiler and linker options. Any of the examples are good for compiling to a .OUT file. The best example for compiling an X0 file is provided in the BLINKING source.

If an X0 is the end product, you must determine the starting address from the MAP file (created by the linker), and then change the appropriate line in the .ROM file to the same value. See **Creating TEKTRONIX Files** below.

7.4 Memory Map

The DSP memory map is described below. This memory map is implemented in the linker file LNK.CMD file provided in the C:\ALPHI67xx\LIB6713 directory..

This command file makes use of the BOOT area for the heap. It also uses the internal RAM 1 for fast access to the STACK and any local variables.

Range	Possible User Use	Bootloader Use
0x000000- 0x0003ff	All user segments	This area is downloaded automatically by the DSP at boot time. The very short program downloaded will program the 6713 memory interface and clock registers, and it will copy the initial bootloader from the FLASH to its memory location
0x000400 – 0x003fff	.bss, .sysmem, .stack These areas cannot be initialized during program load.	The Bootloader uses this area for all segments. This area cannot be initialized at user program load, or the Bootloader will crash.
0x4000 – 0x02ffff	All user segments if the BootROM software is not used and the bootloader loads directly the user software (Boot mode 2 or 3)	BootROM software.
0x08000000 – 0x087FFFFFFF	(Size depends on the amount of SDRAM) All user segments	Not used.

The Bootloader uses only the lowest available locations in the static RAM. The area used is from 0x00000 to 0x003FFF. User code must not preset any locations in this area during program load. It is available to the user for the uninitialized BSS segment, the STACK segment, or the SYSMEM (heap) segment.

If it is desired to use the board in stand-alone mode, and to download and test without burning the program into FLASH, then the RAM transfer area from 0x10000 to 0x1FFFF must be treated in the same manner.

7.5 Creating TEKTRONIX Files

The TI Assembler/Linker comes with a utility called HEX6X. This tool is used to create the X0 files appropriate for the PFLASH utility or for downloading via serial port.

The Blinking.bat and Blinking.cmd files provided in the C:\ALPHI67xx\DSP Examples\Blinking directory demonstrate how this is done.

Note that the program entry point must be specified in the ROM file. This entry point can be found in a linker MAP file or, if debug symbols are present in the OUT file, with the SYMBOLS utility described in this document.

The entry point is called C_INT00.

8. STAND ALONE OPERATION

It is possible to operate the ALPHI Technology cards with DSPs in a stand-alone or embedded mode without a HOST computer to provide bus timing or signals. In these applications, the user's application can be stored in the FLASH memory on the card.

Using the serial port, it is also possible to update the user software in the field without requiring removal of the card from the target system by using the serial port.

Alternatively, a HOST computer running XP can be used to update the user's application.

There usually is some jumpers that must be changed, or possibly an adapter card to supply power for a particular card. See the appropriate hardware manual for details.

8.1 Boot Mode

If desired, the card can be set to automatically load the user's application stored in FLASH at RESET by setting a parameter called Boot Mode. This parameter can be set by one of two methods.

If the card is installed in a HOST computer running Windows XP, then the PLX9056 Editor will allow the user to set the Boot Mode. See the chapter on the PLX9056 Editor and Programmer for more details.

If the card is not installed in a HOST computer running Windows XP, then this parameter can be changed through the Command interpreter running on the serial port. See the chapter on the Bootloader for more details.

Note that, in order to support field updates of the user's code using the serial cable, the Boot Mode cannot be set to load immediately without delay, or there will be no means of aborting the load of the user application to get to the Command interpreter. Updates via HOST computer do not suffer from this limitation.

8.2 Field Update through Serial Cable

This assumes that the card is set to Load User Application after 5 seconds.

With a terminal program connected to the serial port, reset the card. Press the space bar to abort the load of the user program. This will put the card into Command mode.

Using the DOWNLOAD command (see the Bootloader chapter), transfer the new X0 file to the card. Use the BURN command to program the FLASH. Reset the card, and wait the 5 seconds for booting the user program.

8.3 Update by HOST Computer

Modify any jumpers necessary to put the card into hosted mode. Insert the card into the HOST computer.

Using the PFLASH utility, update the FLASH contents with the new X0 file.

Remove the card from the HOST and set the jumpers back to stand-alone mode.

8.4 Required External Connections

If the card is installed in an enclosure that makes it difficult to extract the card for software updates, at a minimum, the following connections must be brought outside of the enclosure to update the FLASH.

- Serial connection.
- RESET line from the jumper as documented in the hardware manual.