User's Guide

AIA (LVDS / RS422) and Camera Link framegrabbers for PCI, cPCI, and PMC

High-speed image capture for multiple form factors

Doc. 008-00966-17
Rev. 2013 November 20

Engineering Design Team (EDT), Inc.
1400 NW Compton Drive, Suite 315
Beaverton, OR 97006
p 503-690-1234 / 800-435-4320
f 503-690-1243
www.edt.com

EDT™ and Engineering Design Team™ are trademarks of Engineering Design Team, Inc. All other trademarks, service marks, and copyrights are the property of their respective owners†.

© 1997-2011 Engineering Design Team, Inc. All rights reserved.

FCC Compliance: EDT devices described herein are in compliance with part 15 of the FCC Rules. Operation is subject to two conditions: (1) The device may not cause harmful interference, and (2) the device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used as described in the user's guide, may cause harmful interference to radio communications. Operating this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his or her own expense.

Caution: Changes or modifications not expressly approved by Engineering Design Team, Inc. could void your warranty to operate this equipment.

Terms of Use Agreement

Definitions. This agreement, between Engineering Design Team, Inc. ("Seller") and the user or distributor ("Buyer"), covers the use and distribution of the following items provided by Seller: a) the binary and all provided source code for any and all device drivers, software libraries, utilities, and example applications (collectively, "Software"); b) the binary and all provided source code for any and all configurable or programmable devices (collectively, "Firmware"); and c) the computer boards and all other physical components (collectively, "Hardware"). Software, Firmware, and Hardware are collectively referred to as "Products." This agreement also covers Seller's published Limited Warranty ("Warranty") and all other published manuals and product information in physical, electronic, or any other form ("Documentation").

License. Seller grants Buyer the right to use or distribute Seller's Software and Firmware Products solely to enable Seller's Hardware Products. Seller's Software and Firmware must be used on the same computer as Seller's Hardware. Seller's Products and Documentation are furnished under, and may be used only in accordance with, the terms of this agreement. By using or distributing Seller's Products and Documentation, Buyer agrees to the terms of this agreement, as well as any additional agreements (such as a nondisclosure agreement) between Buyer and Seller.

Export Restrictions. Buyer will not permit Seller's Software, Firmware, or Hardware to be sent to, or used in, any other country except in compliance with applicable U.S. laws and regulations. For clarification or advice on such laws and regulations, Buyer should contact: U.S. Department of Commerce, Export Division, Washington, D.C., 20230, U.S.A.

Limitation of Rights. Seller grants Buyer a royalty-free right to modify, reproduce, and distribute executable files using the Seller's Software and Firmware, provided that: a) the source code and executable files will be used only with Seller's Hardware; b) Buyer agrees to indemnify, hold harmless, and defend Seller from and against any claims or lawsuits, including attorneys' fees, that arise or result from the use or distribution of Buyer's products containing Seller's Products. Seller's Hardware may not be copied or recreated in any form or by any means without Seller's express written consent.

No Liability for Consequential Damages. In no event will Seller, its directors, officers, employees, or agents be liable to Buyer for any consequential, incidental, or indirect damages (including damages for business interruptions, loss of business profits or information, and the like) arising out of the use or inability to use the Products, even if Seller has been advised of the possibility of such damages. Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitations may not apply to Buyer. Seller's liability to Buyer for actual damages for any cause whatsoever, and regardless of the form of the action (whether in contract, product liability, tort including negligence, or otherwise) will be limited to fifty U.S. dollars ($50.00).

Limited Hardware Warranty. Seller warrants that the Hardware it manufactures and sells shall be free of defects in materials and workmanship for a period of 12 months from date of shipment to initial Buyer. This warranty does not apply to any product that is misused, abused, repaired, or otherwise modified by Buyer or others. Seller's sole obligation for breach of this warranty shall be to repair or replace (F.O.B. Seller's plant, Beaverton, Oregon, USA) any goods that are found to be non-conforming or defective as specified by Buyer within 30 days of discovery of any defect. Buyer shall bear all installation and transportation expenses, and all other incidental expenses and damages.

Limitation of Liability. *In no event shall Seller be liable for any type of special consequential, incidental, or penal damages, whether such damages arise from, or are a result of, breach of contract, warranty, tort (including negligence), strict liability, or otherwise.* All references to damages herein shall include, but not be limited to: loss of profit or revenue; loss of use of the goods or associated equipment; costs of substitute goods, equipment, or facilities; downtime costs; or claims for damages. Seller shall not be liable for any loss, claim, expense, or damage caused by, contributed to, or arising out of the acts or omissions of Buyer, whether negligent or otherwise.

No Other Warranties. Seller makes no other warranties, express or implied, including without limitation the implied warranties of merchantability and fitness for a particular purpose, regarding Seller's Products or Documentation. Seller does not warrant, guarantee, or make any representations regarding the use or the results of the use of the Products or Documentation or their correctness, accuracy, reliability, currentness, or otherwise. All risk related to the results and performance of the Products and Documentation is assumed by Buyer. The exclusion of implied warranties is not permitted by some jurisdictions. The above exclusion may not apply to Buyer.

Disclaimer. Seller's Products and Documentation, including this document, are subject to change without notice. Documentation does not represent a commitment from Seller.

Contents

AIA (LVDS / RS422) and Camera Link Framegrabbers for PCI, cPCI, and PMC

Overview

This guide covers EDT AIA (LVDS / RS422) and Camera Link framegrabbers, both standard and fiberoptic, for PCI, cPCI, and PMC. These framegrabbers provide fast image capture and direct memory access (DMA) between an external camera and a host computer.

The products covered in this guide include:

PCI DV and PCI DVK	Legacy standard PCI framegrabbers for AIA (LVDS / RS422)
PCI DVa	Standard PCI framegrabber for AIA (LVDS / RS422)
PCI DV C-Link	Standard PCI framegrabber for Camera Link
cPCI DV C-Link	Standard cPCI framegrabber for Camera Link
PMC DV C-Link	Standard PMC framegrabber for Camera Link
PCI DV FOX	Fiberoptic PCI framegrabber for Camera Link
PMC DV FOX	Fiberoptic PMC framegrabber for Camera Link

EDT provides a common application programming interface (API) for all supported operating systems, so an application written for one board will work with others in the same family with little or no modification.

For remote operation, you can add one or more EDT extenders (RCX, RCX C-Link, or RCX C-Link Coax2) for fiberoptic or coaxial cable, as noted below. The number of extenders needed depends on what your application is and which EDT board you are using. The FOX boards have one or more fiberoptic transceivers built in for direct input from RCX or RCX C-Link extenders.

Related Products

EDT PCI, cPCI, and PMC framegrabbers can be used with (or in some cases, such as for faster applications, replaced by or upgraded to) various other EDT products, all for Camera Link. These include:

PCIe framegrabbers, such as PCIe4 or PCIe8 DVa C-Link (standard) or PCIe4 DV FOX (fiberoptic)

PCI or PCIe simulators, such as PCI DV CLS or PCIe8 DVa CLS

Fiber extenders, such as RCX C-Link

Coax extenders, such as RCX C-Link Coax2

Record-playback systems, such as RAPID-V

For user's guides to these companion products – as well as addenda covering camera configuration and EDT firmware – see link information under Related Resources, below.

Related Resources

The resources below may be helpful or necessary for your applications.

NOTE To find information on any EDT product, go to www.edt.com and find the product page for links to that product's datasheet (specifications) and user's guide.

NOTE To find technical information that is not related to a specific EDT product (for example, cable pinouts for multiple products), go to www.edt.com and look in Product Documentation.

EDT Resources

Description	*Detail*	*Web link*
• Documentation for each particular product	Datasheets and user's guides	www.edt.com (find product page)
• User's guide for camera configuration (setup)	Camera configuration guide	" (Product Documentation)
• User's guide for firmware (setup)	Firmware guide	" (Product Documentation)
• User's guides for cabling and pinouts	Cabling and pinouts for such form factors as PCI and PMC	" (Product Documentation)
• Application Programming Interface (API)	HTML and PDF versions	" (Product Documentation)
• Installation packages: Windows, Linux, Mac (Solaris, VxWorks)	Software / firmware downloads	" (Product Documentation)

Standards / Specifications

Description	*Pertains to*	*Documentation*	*Web link*
• PCI	PCI bus	PCI Special Interest Group (PCI SIG)	www.pcisig.com
• Camera Link	Camera Link	Machine Vision Online (MVO)	www.machinevisiononline.org
• IRIG-B	IRIG-B time code	Inter-Range Instrumentation Group, mod B	irigb.com

Requirements

EDT framegrabbers are high-speed DMA devices that require adequate bandwidth for reliable operation. As different cameras run at different speeds, the system requirements vary by camera; therefore, you should select and configure your camera and system with these requirements in mind.

For requirements related to input / output (I/O), bus type, and throughput, see Table 1. For details on cabling and pinouts, see link information under Related Resources on page 2.

Table 1. Requirements for I/O, Bus Type, Throughput, and Cabling

Product name	Input / Output	Bus type*	Typical maximum	Cabling
PCI DV C-Link	Camera Link in	PCI (66 MHz)		Camera Link from EDT or 3rd party
PMC DV C-Link		PMC (66 MHz)		
cPCI DV C-Link		cPCI (66 MHz)		
PCI DV FOX	Camera Link / AIA in (over fiber)	PCI (66 MHz)		Fiber-optic
PMC DV FOX		PMC (66 MHz)		
PCI DV CLS	Camera Link out	PCI (66 MHz)		Camera Link
PCI DVa	AIA in (LVDS / RS422)	PCI (66 MHz)		Camera-specific from EDT
PCI DV		PCI (33 MHz)	Up to 90 MB/s	
PCI DV44				Standard DVC 44-pin
PCI DVK				68-68 Princeton / Redlake / Kodak

* If you use a bus that is slower than the bus specified, the board can work, but at reduced bandwidth. For details, see Bandwidth Problems on page 20.

Installation

EDT provides installation packages for all supported operating systems (Windows, Linux, Mac and, upon request, Solaris or VxWorks). These packages are available from two sources:

- The common installation disk that ships with all EDT products (with instructions on the CD sleeve); or
- Our frequently-updated current and archived downloads (see Related Resources on page 2).

NOTE To be sure that you have the latest version of the appropriate installation package, and to avoid future issues with version compatibility, EDT recommends:

— For a new application, download the latest package from the EDT website.

— For an existing application, use the same package that was used to build it, or recompile / relink the application with the latest package from the EDT website.

To install your framegrabber:

1. Uninstall any previously installed EDT installation packages.
2. Do one of the following:

   — For Windows, Linux, Solaris, or Mac OS, follow the instructions on your EDT installation disk sleeve or download the latest package and instructions from the EDT website.

   — For VxWorks, see Appendix B: VxWorks or www.edt.com/support.html, or contact EDT.
3. For non-Camera Link boards, check board jumper settings; see Setting the Control Signal Levels on page 4 for details.
4. Follow your hardware manufacturer's instructions to install the board into your computer.
5. Cable the board to the camera, using the cabling specified in Requirements on page 3.

Setting the Control Signal Levels

If you are using a Camera Link board and camera, you can skip this section.

If you are using an AIA (non-Camera Link) board and camera, be aware that many AIA cameras have serial transmit and receive lines (either RS232 or differential LVDS or RS422) coming from the digital interface connector to provide serial control of the camera directly from the board. The PCI DVa and RCX products are configured at the factory for differential serial signal levels.

Many EDT cables have signal level converters built in. If you are using such a cable, leave the board at the default differential setting even if the camera outputs RS232 serial. EDT cables that pass RS232 straight through are labeled as such; when using those cables, set the jumpers for RS232.

PCI DV and PCI DVK legacy boards are fixed for differential levels. The EDT cables for these products come with RS232-to-differential converters built in; therefore, the default differential setting on the PCI DVa is the correct setting if you are using any cable built for the PCI DV or PCI DVK.

For the appropriate jumper settings for the PCI DVa, see Appendix A: Board Diagrams on page 23.

Included Files

Your EDT installation package includes a capture-and-display application with a graphical user interface (GUI), as well as C source and executables for command-line example, diagnostic, and utility programs.

Applications mentioned in this user's guide include:

`pdvshow`	Capture and display application with a GUI for Windows.
`pdv_flshow`	Capture and display application with a GUI for Linux, Solaris, or Mac OS.
`take`	Command-line application; captures images from the installed camera and (optionally) writes them to files. Also includes many diagnostic options.
`simple_take`	A simplified version of `take` to use as an example application, optimized to acquire and process data in parallel.
`simplest_take`	An even simpler example, with no parallel optimization of data acquisition.
`simple_*`	More examples of other functionalities.
`serial_cmd`	Sends serial commands to an AIA serial camera in the appropriate format.
`dvinfo`	Runs several image capture and other diagnostic tests and collects the results in a file, which you can send to EDT for troubleshooting help; see dvinfo on page 13.
`cl_speed`	A utility application for measuring the maximum bus bandwidth on Camera Link boards. For details, see Simulation and Testing on page 15.

For details on these applications, see Example and Utility Applications on page 10. For a descriptive list of examples and utilities, see the `README` file in your installation package.

Programming Interface

To interface to the framegrabber, use either the EDT Digital Video library provided with the EDT installation package (see Related Resources on page 2), or a combination of the EDT Digital Video Library and the EDT DMA Library.

The EDT Digital Video Library provides a C language interface to your framegrabber; it also handles error recovery, bookkeeping, and camera shutter triggering and timing. Therefore, we recommend using it for all programming specific to EDT products, and using the lower-level DMA library only where it provides needed functionality not provided in the EDT Digital Video Library. Routines in both libraries are documented in the applied programming interface (see Related Resources on page 2).

Building or Rebuilding an Application

By default, EDT's installation package is copied into the directory `C:\EDT\pdv` (Windows) or `/opt/EDTpdv` (Linux , Solaris, Mac OS). The package includes executables and C source code for all examples, diagnostics, and utilities. If you want to rebuild a program, you'll need to use a compiler and (for Windows) the `nmake` application or Visual Studio 6.0 or later, or (for Linux, Solaris, and Mac OS) the `make` utility, as described below.

1. Do one of the following:
   - For Windows, run *Pdv Utilities*.
   - For Linux, Solaris, or Mac OS, navigate to the installation directory in a terminal window.

2. Enter:

   `make file`

   ...where `file` is the name of the example program you wish to build.

3. To build sample programs, either set up a project in Windows Visual C++ or, to build and install all the example programs, use the included Visual Studio project or enter:

   `make`

Setting the Camera Model

After installing the board and its driver, configure it for the camera you will use.

Your EDT installation package provides example configuration files for various camera models; if no file is provided for your camera, or if you wish to modify the directives of an existing configuration file, consult the Camera Configuration Guide on the EDT website (see Related Resources on page 2).

NOTE For a medium- or full-mode camera, you may need to first reprogram the flash memory for medium- or full-mode operation. For details, see Table 3 on page 18.

Next, initialize (configure) the driver for your camera model, using one of the methods in Table 2.

Table 2. Initialization Methods by Operating System

OS	PdvShow	Command line

Windows	To configure the driver for your camera:
	1. Double-click the *PdvShow* desktop icon.
	2. In dialog box, select your camera model.
	3. Click *OK*.
	
	To reconfigure for a different camera model or operating mode:
	1. Double-click the *PdvShow* desktop icon.
	2. Execute *Camera > Setup*.
	3. In dialog box, select your camera model.
	4. Click *OK*.
		Use the `initcam` utility. At a command prompt, enter:
		`initcam -f camera_config/file.cfg`
		...where `file` is the name of the camera configuration file that matches your camera model and operating mode.
		This utility optionally lets you specify a unit (`-u` flag) and channel (`-c` flag). Thus, if you are using two base-mode cameras and wish to configure the camera on channel 1, but not the camera on channel 0, enter:
		`initcam -u 0 -c 1 -f camera_config/file.cfg`
		See Addressing the Board and Camera (in this guide) and the Camera Configuration Guide (Related Resources).

Linux, Solaris, Mac OS	To configure the driver for your camera: 1. Navigate to the installation directory `/opt/EDTpdv`. 2. At the prompt, enter: `camconfig` 3. In dialog box, select your camera model. 4. Click *OK*. To reconfigure for a different camera model or operating mode, rerun `camconfig`.	Follow the above procedure for Windows. NOTE: If you do not have "." in your path, you'll need to precede commands with "./" as in the example below. Example: `./initcam -f camera_config/file.cfg`
VxWorks	For instructions for your setup, see Appendix B: VxWorks or www.edt.com/support.html, or contact EDT.	

Image Capture and Display

For capturing and displaying images, your EDT software contains a GUI in an application called *PdvShow.* To experiment with example code in this application, use the source code indicated below.

- For Windows, look in `pdvshow` in the appropriate Visual Studio project subfolder. For details, see the `README` file in the `pdvshow` subfolder of the main distribution folder (by default, `C:\EDT\Pdv`).

NOTE The `pdvshow` executable comes already built on Windows distributions, so you need not compile it unless you wish to experiment with the source code.

- For a cross-platform, FLTK-based version of *PdvShow* called `pdv_flshow,` see `README` in the `pdv_flshow` subdirectory of the main distribution directory (by default, `/opt/EDTpdv`). For Linux, Solaris, and Mac OS, run `make pdvshow` to build the FLTK application.

Use *PdvShow* to capture an image from the camera or simulator, display the image on the host computer monitor, or manipulate the displayed image in a few basic ways.

To start the application:

1. Double-click the *PdvShow* icon on your desktop, or enter `pdvshow` at a command line prompt.
2. The command line invocation allows you to specify options:

   `pdvshow -pdv`*`n_c`*

   ...where *`n`* is the unit number (useful if you have more than one device) and *`c`* is the channel number for multichannel devices. For example:

   `pdvshow -pdv0_1`

   ...runs *PdvShow* using board 0, channel 1. This is useful if, for example, you are using one DV board with two base-mode cameras, and you wish *PdvShow* to access the camera on channel 1.

NOTE The command line is a property of the icon.To use an icon to access a unit or channel other than 0 (the default): copy and rename the icon; then change its shortcut properties to use the command line with the option `-pdv`*`n_c`* where *`n`* is the unit and *`c`* is the channel.

For demonstration or debugging purposes, you can run this application when no board is installed in the system; the image window then shows a test pattern. To do so, at the command line, enter:

`initcam -u dmy0 -f` *`configuration_file`*

`pdvshow -dmy0`

If you have not yet initialized the driver, select your camera or simulator from the list and click *OK*.

If the image window shows incorrect data (usually because the camera model has been changed since the last driver initialization), select *Camera* > *Setup* to select the correct camera model.

To access camera controls, use the *PdvShow* toolbar and menus. For details, see *PdvShow Help*.

Addressing the Board and Camera

The number of data paths in a given board or host system will depend upon:

- the number of *units* — that is, the number of boards in the host;
- the number of *channels* per board — that is, the number of cameras connected to each board; and
- the number of *taps* in each camera.

For details on taps, consult the EDT user's guide for camera configuration (see Related Resources on page 2). For details on units and channels, see the information below.

Units

The number of units (boards) in your system is limited only by the number of available bus slots and the required bandwidth. For details on bandwidth issues, see Requirements on page 3.

When a system includes more than one board, each board is addressed by a unit number. Most frequently, this is an argument to the `-u` flag when invoking an application (such as `take`), or an argument passed into one of the EDT digital video subroutines.

Boards are numbered starting with unit 0, which indicates the first board in the system.

The sequence is system-dependent.

Channels

In this guide, a channel is a single data path that passes from one camera, through one connector, to a framegrabber, to output to one channel of the host computer.

Some cameras have multiple taps, which their manufacturers may sometimes refer to as channels, in order to double their throughput. These taps are not what is meant, in this guide, by the term *channel.*

For example, the PCI DV C-Link has two connectors and can support two cameras; therefore it has two channels, regardless of the number of taps possessed by either or both of the cameras connected to it. Similarly, the PCI DV FOX can be ordered with one or two connectors; therefore it provides one or two channels, again regardless of the characteristics of the camera or cameras connected to it.

In software, channels are also numbered starting with 0. Channel 0 always uses the connector closest to the PCI bus connector; the next connector accesses channel 1. If more connectors are present, the channels are numbered from the bottom up.

NOTE The physical connectors are labeled 0 and 1 on some boards, but 1 and 2 on others. Regardless of the labels, the software always addresses the first connector as 0, the second as 1, and so on.

Different Camera Link specifications are implemented for different modes. Base mode uses only one connector, while medium and full mode use two connectors and cables with a single camera.

EDT Camera Link framegrabbers allow the two channels to be used for:

- two base-mode cameras, or
- one medium-mode camera with two connectors.

NOTE Due to bandwidth requirements, currently the only EDT board that supports full-mode camera operation is the PCIe8 DV C-Link.

When you use an EDT Camera Link board with two base-mode cameras, the software assigns each camera its own DMA channel, serial channel, and set of registers. It creates two devices, named `pdv0_0` and `pdv0_1` — each device having a unit number of 0 and a channel number of 0 or 1, respectively. Any application that seeks to access these cameras, whether one of our example applications or your own, must therefore specify a channel number.

To see the triggering for all channels on each board, see LED Status Indicators (FOX Boards Only) on page 17.

Application Interface to Units and Channels

If you use two base-mode cameras connected to one PCI DV C-Link board: Open each device with the EDT Digital Video Library subroutine `pdv_open_channel()`, which takes a channel number as an argument, instead of the more generic `pdv_open()`, which always opens channel 0 of the unit selected.

Device open routines return a pointer to the structure that represents the opened device (unit and channel); this pointer appears in EDT examples and documentation as `pdv_p`. Each unit or channel can be opened and manipulated separately by passing the appropriate pointer to the library subroutines.

Serial Communication

Most cameras have a manufacturer-defined serial command set for camera control and status. To make use of this capability, EDT digital video boards implement serial transmit and receive using standard serial lines as defined by the Camera Link specification, or — for non-Camera Link cameras — RS232 or RS422 signaling as defined by the camera manufacturer's specification. You can use serial communications in a number of ways, as discussed below.

At Initialization

As mentioned in Setting the Camera Model on page 6, the initialization process uses directives in a configuration file to set board registers and driver variables to match your camera model and your operating mode. Additional directives (especially `serial_baud`, `serial_init`, `serial_binit`, and other `serial_*` directives) can be used to send serial commands when the system is initialized. These are described in the EDT user's guide for camera configuration (see Related Resources on page 2).

EDT provides several example configuration files that contain the serial commands needed to put a camera into the desired mode. You can edit these commands or copy them to a new configuration file.

For suggestions, see comments in the example configuration files `camera_config/genericXcl.cfg` (where *X* is replaced by various values – for example, `generic8cl.cfg`) in the installation directory.

From Command Line

The command-line utility `serial_cmd`, described in serial_cmd on page 12, allows you to send serial commands to a camera and receive its response, in either ASCII or hexadecimal format. Command-line parameters also can be accessed by entering `serial_cmd --help`.

If you wish to incorporate this functionality in your own application, see the source code provided in `serial_cmd.c` in the EDT installation directory.

From PdvShow

From the *Camera* menu of the *PdvShow* application, select *Programming*. The resulting dialog enables you to send and receive serial commands from the camera. This application is described in Image Capture and Display on page 7.

From a Camera Manufacturer's Application

Most Camera Link camera manufacturers supply a Windows-based graphical camera control application that allows you to send and receive serial commands using a framegrabber-specific serial dynamic link library (DLL). Your EDT installation package provides a DLL named `clseredt.dll`, installed in the `C:\cameralink\serial` folder (a location where many camera control applications automatically search for it). Manufacturers typically provide some method for specifying the DLL pathname; for details, see your camera documentation.

If it becomes necessary to rename the file or copy it to a different location, be sure to recopy any newer versions of the file to the appropriate location when you reinstall the EDT installation package.

NOTE To avoid conflicting commands, do not run any third-party camera application while also running an EDT initialization operation (for example, `initcam`, `camconfig`, or the *PdvShow* camera setup dialog), as EDT initialization may send commands to the camera to put it into the expected state.

From Your Application

To see all of the routines needed for user applications to send and receive serial commands:

In the API (see Related Resources on page 2), follow the link to the EDT Digital Video Library, and then — under Modules at the bottom of the page — to Communications / Control.

Example and Utility Applications

EDT provides a variety of example, utility, and diagnostic applications. All can be run from the command line, using Unix-style options and arguments.

For those developing custom applications for EDT boards, C source code is provided for all the examples. The source code file is the name of the application with a `.c` extension (e.g., "`take.c`").

For those just beginning, we recommend starting with the source for `simple_take` or `simplest_take`, as those applications are the easiest to understand.

The most commonly useful options for these programs are described below. Placeholders shown in italics should be replaced with your own values.

For a complete list of usage options, at the command line, enter the application name with the `--help` option to display the help message.

NOTE When running under VxWorks, be sure to enclose each argument in double quotes (" ").

take

`take` is used to acquire images and (optionally) save them to files. Though it does not display the images, it does provide many other options, making it a useful diagnostic tool. The source also shows how to change camera settings such as integration time; tune image acquisition in certain ways; and detect errors.

Several of the most useful options are:

`-b` *`filename`*	The base name of the file in which to save the image, in Sun raster format (on Unix systems) or Windows bitmap format (on Windows systems). If only one image is taken, this is the filename; otherwise a two-digit number is appended to each file, starting with 00. The appropriate suffix is automatically added.
`-c` *`channel`*	On multichannel boards, selects the channel to access — by default, 0.
`-f` *`filename`*	The name of the file in which to save the image, in raw format. The file includes only raw image data, with no formatting information.
`-l` *`loopcount`*	The number of consecutive pictures you wish to take. The default is one.
`-N` *`numbufs`*	The number of ring buffers — by default, one. (A *ring buffer* is a portion of host memory preallocated for DMA and used in round-robin fashion.) A setting of four optimizes pipelining — one ring buffer currently acquiring data, one ready for data, one getting ready, and one backup.
`-u` *`unit number`*	The unit number, when multiple boards are installed in the host. The default is 0, indicating the first board.
`-v`	Turns on verbose mode. The default is off.

Example: To acquire 100 images as quickly as possible using four ring buffers without saving to files, enter:

```
take -N 4 -l 100
```

Example: If you have one PCI DV C-Link board connected to two base-mode cameras, and you wish `take` to use the camera on channel 1, enter:

```
take -u 0 -c 1 -N 4 -l 100
```

simple_take

`simple_take` shows how to use the API to acquire images from a camera connected to an EDT framegrabber and (optionally) save the images to files.

To add image acquisition to an application, EDT recommends starting with this example, which shows how to use the ring buffer routines to improve performance by pipelining image acquisition and processing.

Several of the most useful options are:

`-b` *`filename`*	The base name of the file in which to save the image, in Sun raster format (on Unix systems) or Windows bitmap format (on Windows systems). If only one image is taken, this is the filename; otherwise a two-digit number is appended to each file, starting with 00. The appropriate suffix is automatically added.
`-c` *`channel`*	On multichannel boards, selects the channel to access — by default, 0.
`-l` *`loopcount`*	The number of consecutive pictures you wish to take. The default is one.
`-N` *`numbufs`*	The number of ring buffers — by default, one. (A *ring buffer* is a portion of host memory preallocated for DMA and used in round-robin fashion.) A setting of four optimizes pipelining — one ring buffer currently acquiring data, one ready for data, one getting ready, and one backup.
`-u` *`unit number`*	The unit number, when multiple boards are installed in the host. The default is 0, indicating the first board.

Example: To acquire four images as fast as possible using four ring buffers, saving each to files named `picture00.bmp` through `picture03.bmp` on Windows (or `.ras` on Linux, Solaris, or Mac OS), enter:

```
simple_take -N 4 -l 4 -b picture
```

simplest_take

`simplest_take` is the simplest example application. It sets up four ring buffers and acquires a single image, with no pipelining.

`simplest_take` accepts an optional argument of a file name to which to save the image. If no name is supplied, it reports a successful image acquisition, or any errors that occurred — useful for testing.

Example: To acquire an image and save it to a file named `pic.bmp,` enter:

```
simplest_take -b pic.bmp
```

serial_cmd

`serial_cmd` sends serial commands to a camera through your EDT framegrabber(s), using calls to routines in the API.

By default, the application starts in command mode: the final argument to `serial_cmd` is the command to send to the camera. Delimit this command with single or double quotation marks — either works, as long as you're consistent. For example:

```
serial_cmd "MDE?"
```

If you omit the command argument, the application enters interactive mode, in which you can type one command per line. To quit the application, enter Control-C.

Several of the most useful options are:

`-c` *`channel`*	The channel to access, on multichannel boards; default is 0 (first channel).
`-u` *`unit number`*	The unit number, if multiple boards are installed; default is 0 (first board).
`-x`	Treats the command argument as a hexadecimal number, which is sent to the camera without terminating nulls or carriage returns. The default is ASCII with a terminating carriage return added.

Example (command mode usage):

`% serial_cmd "MDE?"`	(Redlake "Query Mode" command)
`MDE TR`	(camera response)
`%`	

Example (interactive mode usage):

`% serial_cmd`	
`>MDE?`	(Redlake "Query Mode" command)
`MDE TR`	(camera response)
`>`	
`% serial_cmd -x`	(for hexadecimal arguments)
`> 03 06 02`	(camera-dependent command)
`> Control-C`	(end the program)

To access a camera on channel 1 (for example, if you have two base-mode cameras connected to one PCI DV C-Link), enter:

```
% serial_cmd -u 0 -c 1 "MDE?"        (Redlake "Query Mode" command)
MDE TR                               (camera response)
%
```

dvinfo

`dvinfo` should be run with the camera connected and powered on and the board initialized for your camera with `initcam` or `pdvshow`, if possible. The application gathers technical data that is specific to your EDT board, runs tests, and writes the results to `dvinfo.out` in the current directory. Use the resulting ASCII text file to diagnose problems yourself, or send the file to EDT for technical support.

One useful option is:

`-u` *`unit number`*	The unit number, when multiple boards are installed in the host. The default is 0, indicating the first board.

Triggering

This section describes the most common triggering methods for your EDT framegrabber, as well as each mode's configuration file directives, serial control commands, and software considerations. You can change trigger modes either directly in your application, or by using configuration file directives.

- For details on camera configuration directives, see Related Resources (Camera Configuration Guide).
- For details on specific serial control commands, see your camera manual.

By default, most cameras power up in continuous (also called *freerun*) mode, sending images continuously. For most cameras, EDT provides configuration files for freerun mode. For some cameras, EDT also provides configuration files for internal triggered, external triggered, or pulse-width mode. All of these modes and configuration details are described below.

Freerun (Continuous)

In this mode, the camera acquires images continuously without waiting for a trigger signal.

Configuration file directives	`MODE_CNTL_NORM: 00` (the default)
	`serial_init:` as needed to set freerun mode (usually not necessary because most cameras power up in freerun by default)

Triggered by Your EDT Board

In this mode, the camera waits for a trigger signal from your EDT board before acquiring an image.

Configuration file directives	`MODE_CNTL_NORM: 10`
	`serial_init:` as needed to put the camera in triggered mode

Pulse-width Triggered (Controlled or Level)

In this mode, exposure duration is determined by how long the EXPOSE line is held TRUE (high).

Configuration file directives	`MODE_CNTL_NORM: 10`
	`method_camera_shutter_timing: AIA_MCL`
	`serial_init:` as needed to put the camera in pulse-width mode

API subroutine	`pdv_set_exposure,` millisecond units, range 0–25500

Some cameras have very fast shutters and can accept exposure times in microseconds. For such cameras, a different configuration file directive tells `pdv_set_exposure` to use microseconds instead of milliseconds:

Configuration file directives	`MODE_CNTL_NORM: 10`
	`method_camera_shutter_timing: AIA_MCL_100US`
	`serial_init:` as needed to put the camera in pulse-width mode

API subroutine	`pdv_set_exposure,` microsecond units, range 0–25500

External Trigger Direct to Camera

With this method, the trigger is sent from an external source directly to the camera, bypassing the board. The camera and board can be configured as in freerun mode; however, depending on timing, your application may time out and fail to receive images. You can avoid timeouts in either of two ways:

- If the maximum period of time between triggering signals is known, configure the `user_timeout` period in the software for a large enough value to avoid timeouts.
- If application blocking is acceptable, configure the `user_timeout` period in the software for an infinite period (`user_timeout=0`) to ensure that the application waits until an image arrives.

Configuration file directive	`MODE_CNTL_NORM: 00` `user_timeout` as needed to ensure that your application does not time out while waiting for an image
API subroutine	`pdv_set_timeout,` millisecond units

NOTE To help prevent timeouts and data loss, be sure to comply with Requirements on page 3.

External Trigger Pass-through

With this method, a trigger is sent from an external device to the board, and from the board to the camera. A TTL signal is input to the board, which in turn sends out an LVDS or RS422 signal (depending on the board and its configuration) to the camera trigger line, typically CC1.

Configuration file directive	`MODE_CNTL_NORM: A0` `CL_CFG2_NORM` as needed to set separate or combined triggers from trigger 0 pins `user_timeout` as needed to ensure that your application does not time out while waiting for an image
API subroutine	`pdv_set_timeout,` millisecond units

The software timeout considerations are the same as those in the section External Trigger Direct to Camera.

NOTE To help prevent timeouts and data loss, be sure to comply with Requirements on page 3.

External Triggering Pins

The pins to which you connect the trigger source are shown in Appendix A: Board Diagrams on page 23.

With two cameras, trigger input 0 can trigger both cameras, or triggers 0 and 1 can trigger cameras 0 and 1 independently; for details, see configuration directives (above).

Fire the trigger by applying a TTL signal lasting at least 10 microseconds to these pins, which in turn send a signal of the appropriate level to the camera trigger line, typically CC1.

The two pins of each trigger drive an optocoupler (Fairchild HCPL062N) through a 130-Ω series resistor. This optocoupler is provided to allow coupling to electromechanical systems in which major ground spikes can occur when electrical devices such as motors, for example, turn on or off.

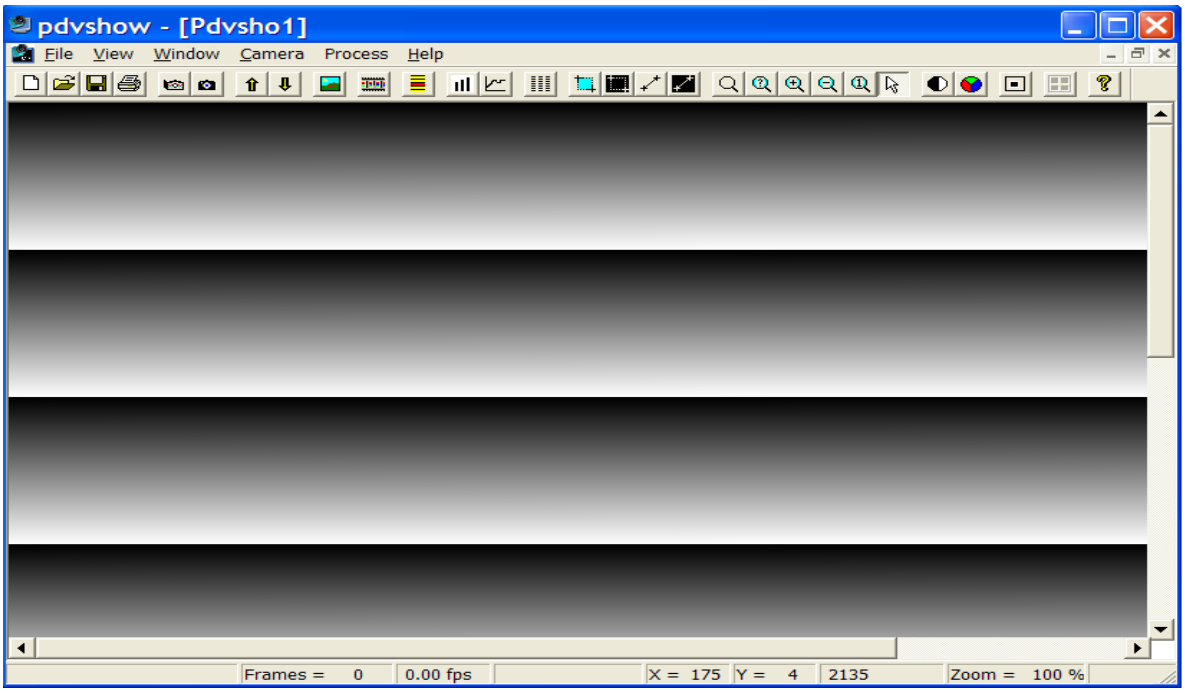
Simulation and Testing

In addition to the example applications `pdvshow` and `take` (see Example and Utility Applications on page 10), several EDT boards offer special simulation capabilities (below) for more in-depth testing.

Camera Link Boards

EDT's Camera Link boards include a "phantom" channel 2 that lets you generate sample data with no camera attached to the board (useful for testing the board hardware). This channel uses a simple counter to generate 16-bit pixel data; pixels start black and fade to white, as shown in Figure 1.

Figure 1. Simulated Channel 2 Test Image in PdvShow



Data is generated as fast as the bus speed allows:

- On a 33 MHz PCI bus with no contention, 90–120 MB per second;

- On a 66 MHz PCI bus with no contention, 190–220 MB per second.

To use the phantom channel 2 with `pdvshow`:

1. Start the application with:

   ```
   pdvshow -pdv0_2
   ```

2. Select the *Camera Setup* menu item.
3. Select the desired configuration.

To use the phantom channel 2 with your own application:

1. Use the initialization application `initcam`, specifying channel 2 and the camera configuration file for your camera. From a command prompt, enter:

   ```
   initcam -c 2 -f camera_config/yourCamera.cfg
   ```

   ...replacing *`yourCamera`* with the appropriate configuration file name for your camera and application.

NOTE For general testing, use one of the `genericXXcl.cfg` files provided with your board.

2. Open the device with a call to `pdv_open_channel`, passing it `NULL` for the device, your value for the board unit number, and 2 for the channel – as in this example (showing a board with a unit number of 0):

   ```
   PdvDev *pdv_p = pdv_open_channel(NULL, 0, 2);
   ```

   The pointer returned from this call points to the simulated data; any image acquisition calls, such as `pdv_wait_image`, that pass this pointer will access the simulated data.

Because data is generated as fast as possible, you can also use the channel 2 simulator to measure the maximum bus bandwidth, using the utility application `cl_speed`. This utility takes the unit number as an argument and begins channel 2 simulation on the specified board; it then reports the data speeds achieved.

Legacy PCI DV and PCI DVK Boards

For legacy PCI DV / DVK boards only, you can use the files below to generate sample data at a fixed rate.

NOTE This method will not work with Camera Link or PCI DVa boards, but only with PCI DV / DVK boards.

`camsim.cfg`	Provides simulated 1024 x 1024 x 8 camera data at 5 MHz.
`camfst.cfg`	Provides simulated 1024 x 1024 x 8 camera data at 20 MHz.

You can generate this sample data using the image capture and display application `pdvshow`, the configuration application `camconfig`, or the initialization application `initcam`.

To do so using `pdvshow`:

1. Select the *Camera Setup* menu item.
2. From the resulting list, select *EDT Simulator 8 bit 5 MHZ* or *EDT Generic Simulator 20 MHz*.
3. Click *OK*.

To do so using `camconfig`:

1. In the initial dialog, select *EDT Simulator 8 bit 5 MHZ* or *EDT Generic Simulator 20 MHz*.
2. Click *OK*.

To use `initcam`, from a command prompt, enter either one of the following:

```
initcam -f camera_config/camsim.cfg
```

```
initcam -f camera_config/camfst.cfg
```

LED Status Indicators (FOX Boards Only)

Each FOX board (PCI or PMC DV FOX) has up to four LED status indicators – one for each transceiver you ordered. For the location of these LEDs, see the diagrams under Fiberoptic (FOX) Framegrabbers – for PCI and PMC on page 26.

When a FOX board is first powered up, it is likely that all of its LEDs will be on, because there is no firmware in the FPGA and all outputs of the FPGA have weak internal pullups.

After the FPGA firmware is loaded by bitload (which is called by `initcam` when you configure the board for a specific camera), each LED shows the current status of its corresponding transceiver, as follows:

- The bottom LED is lit when Channel 0 is receiving a valid fiberoptic signal, and dark when it is not.
- The next LED up is lit when Channel 1 is receiving a valid fiberoptic signal, and dark when it is not.
- The upper two LEDs typically are not used and are blocked by the panel. Currently the third LED is not present or permanently dark, and the uppermost fourth LED is permanently lit.

Firmware

At times, you may need to reprogram the PCI interface flash memory using `pciload` – for instance:

- if you want to switch from one mode to another (base, medium, full) on certain EDT boards;
- if you need to use an FPGA configuration file that has special functionality;
- if you upgrade to a new installation package that includes a required update for your board; or
- if the firmware becomes corrupted.

To do so, follow the instructions in the sections below.

Checking and Loading the Firmware

The following procedure applies to standard firmware only. If you are running a custom firmware file and need to update it, first get a custom firmware configuation file from EDT.

NOTE Do not upgrade the firmware simply because you see a newer firmware file with a new driver; instead, consult the `README` file in the package, which will tell you if there is a mandatory upgrade.

- For Windows, `pciload` is an application in `\EDT\Pdv`. Double-click the *Pdv Utilities* icon to bring up a command shell in the installation directory `\EDT\Pdv`.
- For Linux, Solaris, and Mac OS, `pciload` is an application in `/opt/EDTpdv`.

To see currently installed and recognized EDT boards and firmware, enter:

```
pciload
```

The program outputs the date and revision number of the firmware in the flash memory.

To compare the PCI firmware in the package to the firmware loaded in flash on the board, enter:

```
pciload verify
```

If the two match, the firmware on the board is the same as the firmware in the installation package. If they differ, you'll see error messages, but these do not necessarily indicate problems; if your application is working correctly, you probably do not need to upgrade the firmware.

If you do wish to update the standard firmware, enter:

```
pciload update
```

To upgrade or switch to a specific firmware file, enter:

```
pciload firmware
```

...replacing *firmware* with the filename of the desired firmware, up to but not including the `.bit` extension.

If the filename ends in `_3v.bit` or `_5v.bit,` include only the part before the "_".

Example: To load medium mode firmware `pdvcamlk_pir_3v.bit` and `pdvcamlk_pir_5v.bit`, enter:

```
pciload pdvcamlk_pir
```

The board reloads the firmware from the flash memory only during powerup – so after running `pciload,` the old firmware is in the PCI FPGA until the system has power-cycled.

NOTE Updating the firmware requires cycling power, not simply rebooting.

For a list of all `pciload` options, see Table 3 (below) or enter:

```
pciload --help
```

Table 3. Arguments to pciload for specific boards

Board name	Command
PCI DVa	`pciload dva1`
PCI or PMC DVK	`pciload pdvk`
PCI or PMC DV FOX	`pciload dvtlk4`
PMC DV C-Link	`pciload pdvcamlk` (base mode) `pciload pdvcamlk_pir` (medium mode)
cPCI DV C-Link	`pciload pdvcamlk` (base mode) `pciload pdvcamlk_pir` (medium mode)
PCI DV C-Link (rev. 51 or lower)	`pciload pdvcamlk` (base mode) `pciload pdvcamlk_pir` (medium mode)
PCI DV C-Link (rev. 52 or higher)	`pciload pdvcamlk2` (base mode) `pciload pdvcamlk_pir` (medium mode)
PCI DV CLS	`pciload dvc14`

Corrupted Firmware

In rare cases, the board firmware may become corrupted. Typically, the symptom is that the board is not recognized by the operating system, or the computer itself will not boot with the board in it. In such cases, booting from the protected (backup) sector will allow the board to be seen so that you can reprogram the programmable sector.

Each EDT DV board has both a protected (backup) and a programmable flash memory boot sector. The sector from which the board will boot is determined by a jumper, which is preset to the programmable sector. If that sector becomes corrupted, you can move the jumper so the board will boot from the protected sector.

Most often, firmware corruption is the result of an interrupted load process or an unanticipated interaction with the host computer; if so, following the procedure below should solve the problem.

However, if the firmware file itself has become corrupted, first contact EDT for the current firmware you'll need to replace it, and then follow this procedure.

To reboot from the protected sector:

1. If needed, move the new firmware file to the directory in which you installed the EDT software.
2. Power down the host and board.
3. To avoid later confusion, remove any other EDT boards from the host.
4. On the EDT board with the corrupted firmware, move the jumper from its programmable to its protected setting (to locate this setting on your board, see Appendix A: Board Diagrams on page 23).
5. Power up the host and board.
6. Navigate to the directory in which you installed the EDT software.
7. At the command prompt, enter:

   `pciload`

   The program outputs the date and revision number of the firmware in the flash memory — in this case, the date and revision number that shipped as of your purchase date. If no errors are reported, you have successfully booted from the protected sector.
8. With the system still powered up, move the jumper back to its original position.
9. Enter:

   `pciload` *`firmware`*

   ...replacing *`firmware`* with the correct filename, as indicated in Table 3 (above). If the feedback shows no errors, the new firmware has been installed, although it is not yet running.
10. Power down the host and board again.
11. Power up the system. The new firmware is now running.

Troubleshooting

This section offers suggestions to help you with various difficulties you may encounter. To submit a bug report or to ask for further technical help, go to the support section of the EDT website.

A number of diagnostic utilities are provided with the EDT installation package. The most useful of these are briefly described in Example and Utility Applications on page 10.

When contacting EDT for technical support, provide the output from the applications that are most relevant, as well as the output from the utility `dvinfo`.

Diagnostic Programs

The `dvinfo` program gathers board- and system-specific information and runs tests; then it writes the results to the file `dvinfo.out` in the current directory. For details, see dvinfo on page 13.

If you are having trouble, run `dvinfo` and send the output file (`dvinfo.out`) to EDT with your query, or have the file available when you call.

Problems Installing Software

To address software installation problems, download the latest installation package from the EDT website and follow the instructions below.

NOTE Be sure to remove previous software releases before installing updates.

On Windows, if you have more than one board in the host, reboot the computer after installing the software (you'll be prompted to do so when it's necessary). If a board isn't recognized after installation:

1. On the *System Properties* control panel, under the *Hardware* tab, use the *Device Manager* to uninstall the unknown EDT device.
2. From the *Action* menu, select *Scan for Hardware Changes*.

Do not extract and install files by hand; doing so will circumvent the installation process, which automatically updates system files and creates links as well as creating new files.

Similarly, do not remove files by hand. Instead, use `pkgrm`. (Solaris) or the *Add or Remove Programs* facility (Windows). On Linux:

- If the package was installed using *`package`*`.run`, remove it with `./uninstall.sh`.
- If the package was installed using `rpm --install package`, remove it with `rpm -erase package`.

Corrupted Images, Timeouts, or Slow Acquisition

Corrupted images, repeated timeouts, or slow acquisition rates usually indicate that the bus is too slow or the driver is misconfigured. To correct this:

- Verify that the camera configuration file is the correct one for your camera and operating mode; see Setting the Camera Model on page 6.

NOTE To help prevent timeouts and data loss, be sure to comply with Requirements on page 3.

When updating to a new device driver, always recompile and relink applications that use EDT libraries.

- Eliminate other devices, if any, that may be reducing the available bus bandwidth.
- Determine the bandwidth required by your camera, and then ensure that both the board and the host can sustain the required throughput. For faster cameras or multiple camera installations, select a host that has one or more PCI or PCI-X buses of 66 MHz or faster, with a faster board such as the PCIe8 DVa C-Link. For details, see Bandwidth Problems on page 20.

Bandwidth Problems

To address bandwidth problems, you'll need to determine the bandwidth required by your camera and verify that both the board and the host system can sustain the required throughput.

NOTE To help prevent timeouts and data loss, be sure to comply with Requirements on page 3.

Bandwidth requirements depend on the camera's pixel clock speed, the number of bytes per pixel, the number of taps, and the number of channels. To calculate bandwidth requirements, use this equation:

```
pixel clock speed * taps * bytes/pixel = total bandwidth
```

NOTE The pixel clock speed, not the frame rate, affects bandwidth requirements. If data loss is a problem, increasing the delay between frames probably will not help.

Cameras that output 10–16 bits per pixel will require two bytes per pixel of bandwidth. Therefore, a camera that has a 40 MHz pixel clock, two taps, and 12 bits per pixel requires system bandwidth of:

```
40 * 2 * 2 = 160 MB per second
```

You can ease this constraint by running the camera in a less demanding mode — for example, you can often run two-tap cameras in single-tap mode, or 12-bit cameras in 8-bit mode.

A multichannel board with multiple cameras also may exceed bandwidth limits. The bus bandwidth may be sufficient for slower cameras, but not for two or more fast cameras. Other tips include:

- Ensure that the board is not sharing the bus with other devices that reduce the available bus bandwidth. PCI video boards or other image acquisition boards — including more than one EDT board on the same bus — can all reduce bandwidth, sometimes considerably.
- One 33 MHz board on a system will slow a 66 MHz (or faster) bus down to 33 MHz for all devices. To avoid this problem, isolate such boards on separate, independent PCI buses, or select a different bus (such as AGP or PCI Express Video) for the other devices.
- A typical PCI desktop system provides one 33 MHz bus and supports one camera, with output of less than 90 MB per second. For faster or multiple cameras, select either a workstation or server system that has one or more PCI or PCI-X buses of 66 MHz or faster, or a PCI Express host and board.

Problems Acquiring Images Using EDT Applications

If you have problems with image acquisition when using EDT applications, try the suggestions below.

- Make sure the camera device is on and is receiving power.
- Verify that all interface cable connections are working properly: Turn off the camera, unplug the cable at both ends, reattach the cable at both ends, and turn on the camera again.
- Try a different cable, if available.
- Try a different camera, if available.
- Verify that the EDT installation package was installed and the driver is running and recognizes the board. In the EDT installation directory, go to the command line and enter:

  ```
  pciload
  ```

  If you see no information about your EDT board, then a problem occurred during installation. To resolve it, one at a time, try:

  — cycling system power;

  — moving the board to a different slot;

  — uninstalling and reinstalling the driver.

For best performance, install your EDT board in a bus that is not shared with any other devices and that meets the board's speed requirements, as shown in Requirements on page 3.

Problems With Your Applications

A mismatch between driver and library software versions can cause application or system failures in your applications that use EDT libraries. When updating to a new device driver, always recompile and relink applications that use EDT libraries.

Use the PDVDEBUG and EDTDEBUG environment variables to enable debug console output from the EDT libraries. Both libraries are documented on the EDT website.

Applications that use the EDT Digital Video library can set the PDVDEBUG environment variable to 1 or 2; a value of 1 turns on call trace information from most (though not all) library routines; a value of 2 enables more verbose trace information. A value of 0 turns off debug output.

Similarly, applications that use the EDT DMA library can set the EDTDEBUG environment variable to 1 or 2 for less or more verbose debug output from the DMA library.

The EDT Message Handler Library provides generalized error- and message-handling for EDT software libraries and can be helpful for debugging your programs. See the EDT Message Handler Library in the API for specific routines and usage.

Before calling for technical support, reproduce the problem with one of our applications, such as `take`, `simple_take`, or `pdvshow`, if possible. If you cannot reproduce the problem with an EDT application, compare your code with ours to see if you can spot the difficulty.

A simple example program that demonstrates the problem is also helpful.

Problems with Threads

The driver and libraries for your EDT product use threads. Make sure all PDV acquisition calls (for example, `pdv_start_image, pdv_wait_image, pdv_multibuf`) are in the same thread.

Some third-party software packages may not be thread-safe. If this causes problems, contact your third-party vendor to determine if a thread-safe version is available.

Firmware Problems

You may need to update the PCI interface flash memory with `pciload` under these conditions:

- If you install a new device driver or switch to an FPGA configuration file with special functionality;
- If your firmware becomes corrupted;
- If the board is not seen in the system or is missing functionality that is in newer boards;
- If the camera is full mode and the board is running base mode firmware (or vice versa).

For details, see Simulation and Testing on page 15.

Appendix A: Board Diagrams

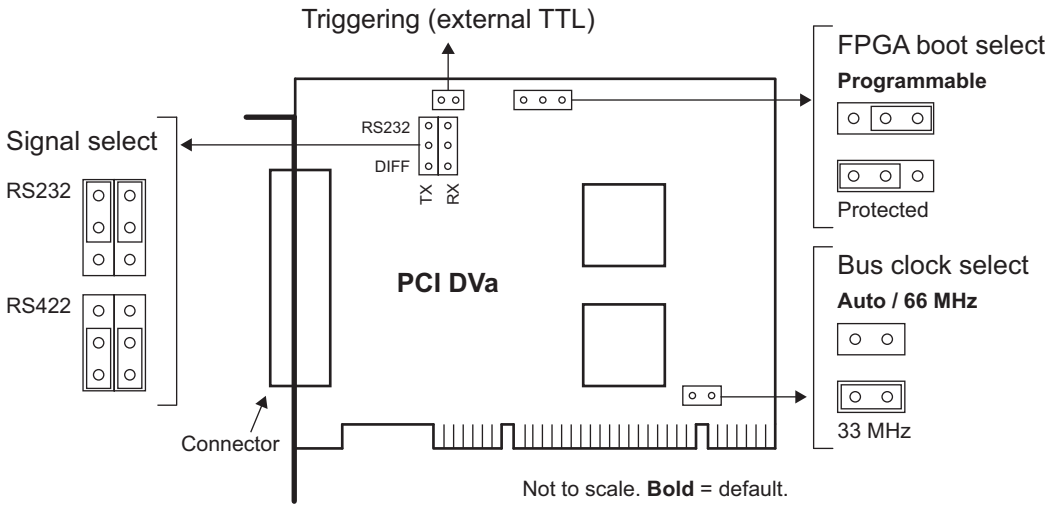
This section shows board diagrams of most EDT framegrabbers, excluding some legacy versions. The diagrams show configurations for external triggering, FPGA boot select (flash memory), bus clock select, and – in some cases – signal select.

Key features of these products are as follows:

- The fiberoptic (FOX) framegrabbers have fiberoptics built in; the standard framegrabbers do not.
- However, all of these framegrabbers can be used with EDT RCX-series extenders for long-range applications. For a link to the RCX-series user documentation, see Related Resources on page 2.
- All framegrabbers work with Camera Link except the PCI DVa, which is for AIA (LVDS / RS422) only.

Standard Framegrabbers – for PCI, cPCI, and PMC

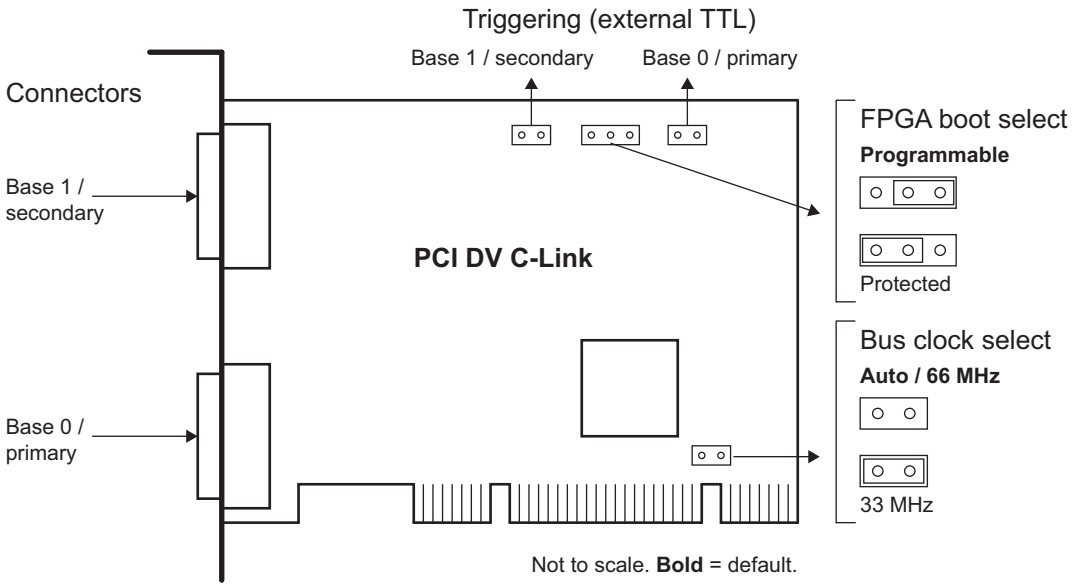
PCI DVa – for AIA (LVDS / RS422) Only



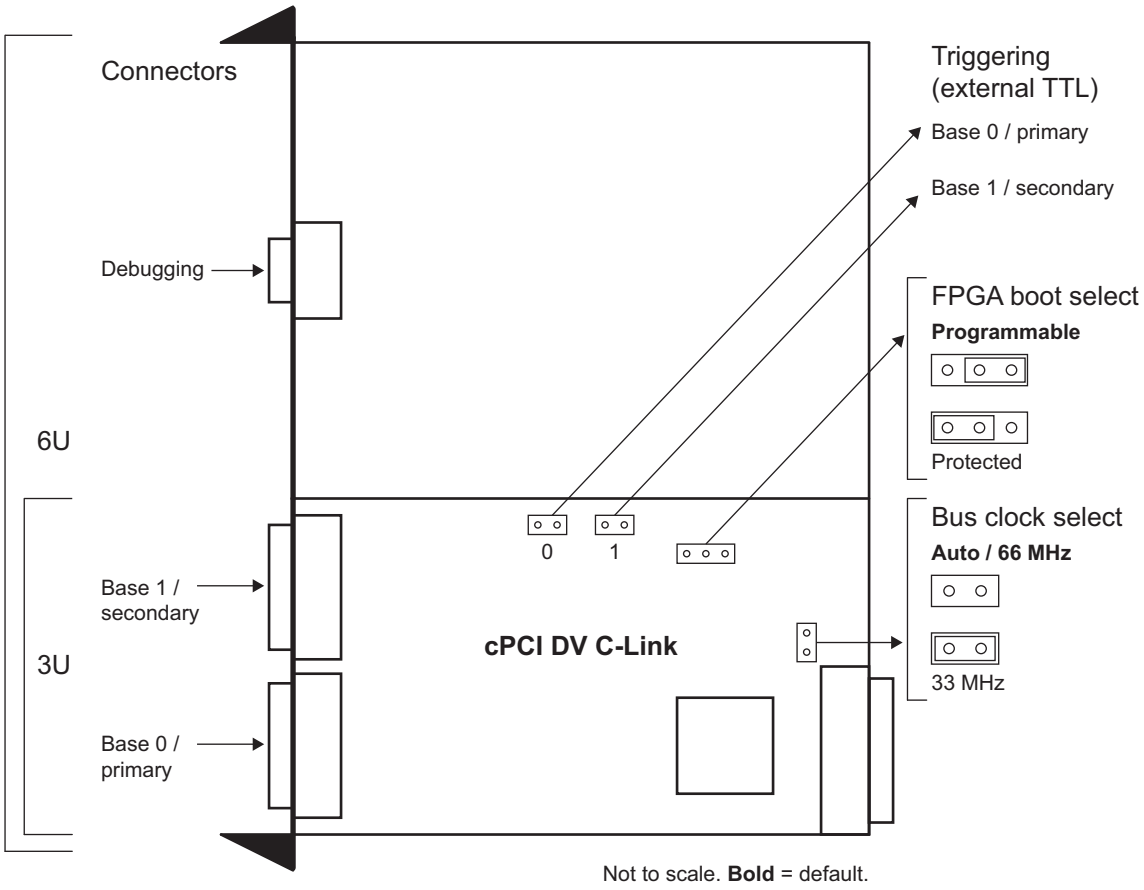
PCI DV C-Link

NOTE To use two input triggers going to two separate cameras with a PCI DV C-Link board, you'll need revision 36 or higher of `pdvcamlk` (available in your EDT installation package, version 4.1.7.7 or higher). You must also set a bit in a register to separate the two trigger inputs. To do so, use the configuration file directive:

```
CL_CFG2_NORM: 01
```



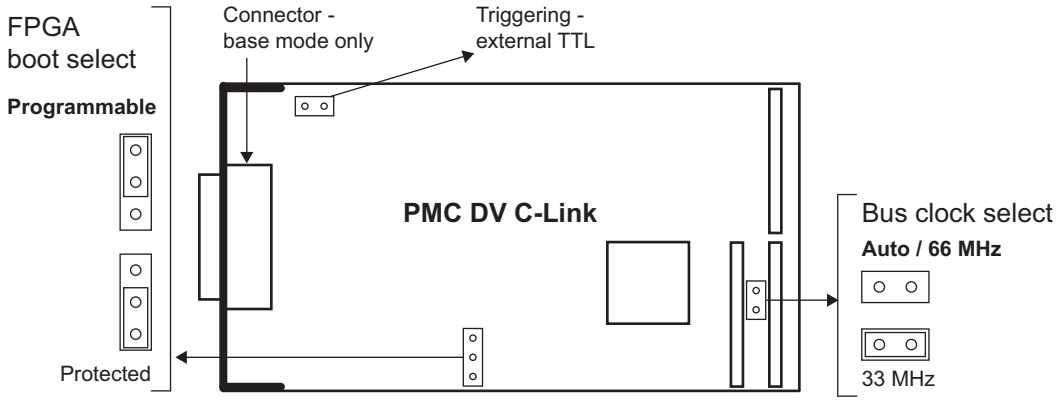
cPCI DV C-Link



Not to scale. **Bold** = default.

PMC DV C-Link

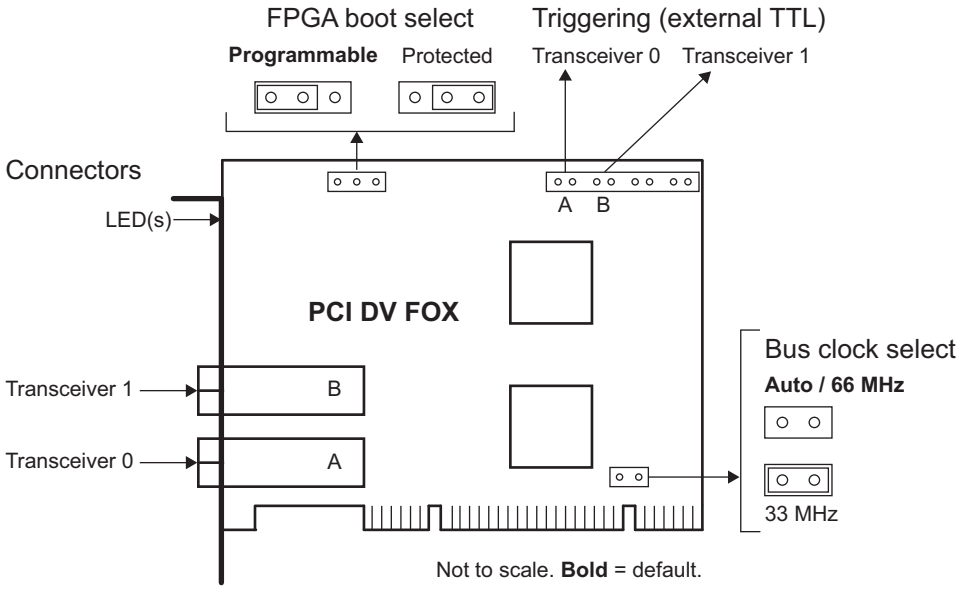
NOTE The jumpers on the PMC DV C-Link are on the side of the board that faces your main PMC bus board; to change the jumpers, you must separate the two boards. The exposed side of the PMC DV C-Link indicates the protected setting with a small bracket labeled *SEL PROTECTED*.



Not to scale. **Bold** = default.

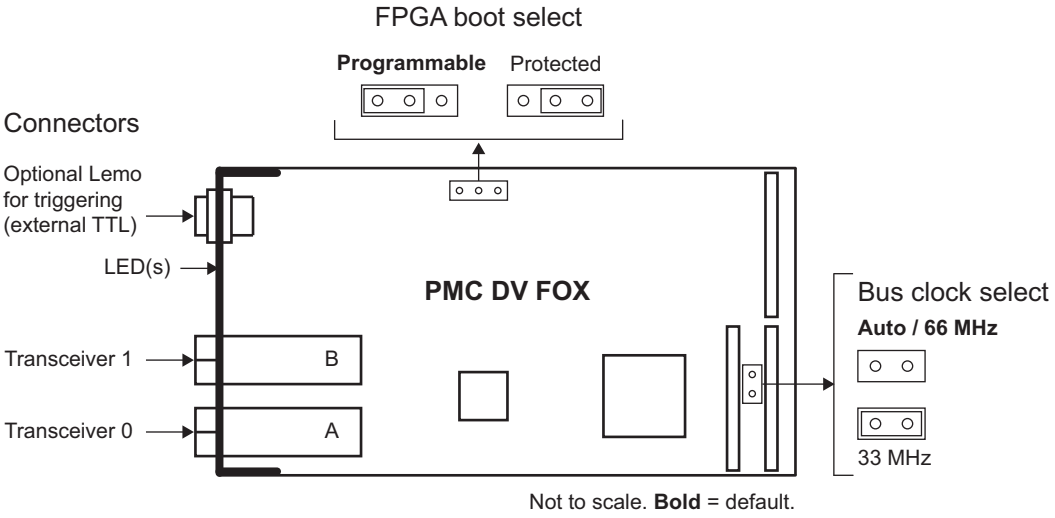
Fiberoptic (FOX) Framegrabbers – for PCI and PMC

PCI DV FOX



PMC DV FOX

NOTE The jumpers on the PMC DV FOX are on the side of the board that faces your main PMC bus board; to change the jumpers, you must separate the two boards. On the exposed side of the PMC DV FOX, for FPGA boot select, there are two small labels: *USR* (for programmable) and *BKUP* (for protected).



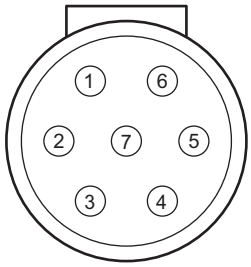
External Signal Inputs

With Berg Pins or Optional Lemo Connector

NOTE All current EDT framegrabbers have Berg connectors for external triggers and other external inputs, and the PMC DV FOX has an optional seven-pin Lemo connector on the front panel for the same purpose. Below is the Lemo pinout.

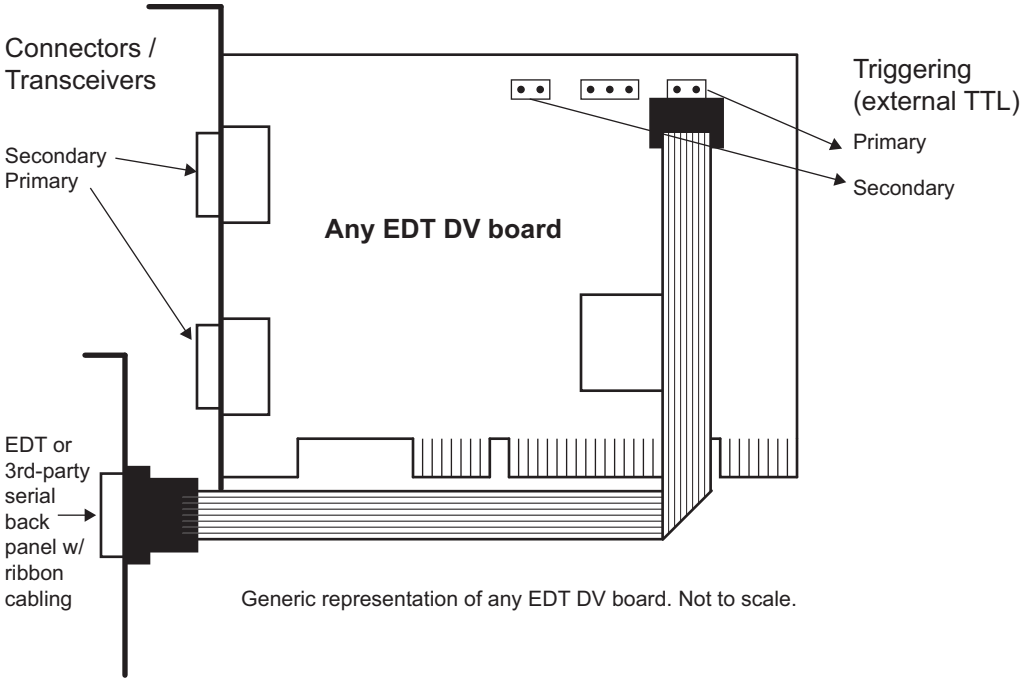
Table 4. Lemo Pinout (PMC DV FOX)

Lemo Pin	Function
1	Trigger 0+
2	Reserved
3	Trigger 1-
4	Trigger 1+
5	Reserved
6	Trigger 0-
7	Ground



With Ribbon Cabling and D9 Connectors

NOTE If you do not have the optional Lemo connector (PMC DV FOX only), you can use a serial backpanel equipped with ribbon cabling and one or two 9-pin D-connectors. The diagram below illustrates the single D-connector setup.



Generic representation of any EDT DV board. Not to scale.

Appendix B: VxWorks

To run EDT products on VxWorks, you'll need to contact EDT for instructions customized for your setup.

NOTE VxWorks requires program arguments to be enclosed in double quotes (“ ”). Otherwise, all EDT command-line applications, utilities, and examples work the same as on other operating systems.

Initialization

The program `edtInstall` initializes the driver. To initialize the board for your camera, run `camconfig` and select your camera model.

Applications With and Without File Systems

EDT's VxWorks software can be configured to run with or without a file system.

If your target system has a file system, all programs work as for other operating systems, as described in the rest of this manual.

If your target system does not have a file system, build the required camera configuration files into the kernel module at the time of compilation. To save images to disk, you must set up a remote (e.g., FTP) connection to your development host. However, even with no file system, you can test whether image acquisition is working by running:

`camconfig` (if you have not yet initialized the board)

`take "-l 10"` (to acquire 10 images and store them in main memory until the program exits).

Display Applications

Because VxWorks applications typically include hardware-dependent code, EDT's GUI applications (such as `pdvshow`) do not work with VxWorks out of the box. However, EDT provides source code for `pdvshow` and the FLTK/OpenGL version `pdv_flshow` for developing a GUI.

If you wish to write your own image acquisition application, start with the example on the first page of the EDT Digital Library in the EDT API (see Related Resources on page 2). If not, simply use `take` to capture images and transfer the images to your PC for display.

Portability

If you want to develop a cross-platform application (for example, to develop and test a program on Windows or Linux before deploying it on VxWorks), be aware that for VxWorks, the operating system, applications, and libraries are linked together into one binary that is loaded into memory and run as a single process, so applications cannot have a `main()` function.

To address this, your application can check whether the `NO_MAIN` name is defined and, if it is, replace `MAIN` with the application name. VxWorks passes the command line as a single string, so the code should use `opt_create_argv()` to split the string into an argument array and count.

Example (from `xtest.c`):

```
#ifdef NO_MAIN

  #include "opt_util.h"
```

```
 int xtest(char *command_line)
#else
 int main(int argc, char **argv)
#endif
{
 #ifdef NO_MAIN
     char **argv  = 0 ;
     int argc = 0 ;
     opt_create_argv("xtest",command_line,&argc,&argv);
 #endif
```

The rest of the code in the `xtest.c` file deals with `argv` and `argc` and works the same on VxWorks as on other operating systems.

Note also these other portability issues:

- Stack size is set by the user; it does not grow automatically as it would in other operating systems.
- To prevent namespace conflicts, application functions should be declared static.
- Global variables should not be used; if they are used, declare them static.

Any globals and statics will have correct values only the first time the program is run; after that, the variables are still present and if nothing resets them, problems will occur when the program runs again.

Example: If global `init_done = 0` and `my_prog()` sets the global `init_done = 1,` then next time the program is run, `init_done` will be 1. Thus, `my_prog()` should reset `init_done = 0` right before exit.

Revision Log

Below is a history of modifications to this guide.

Date	By	Rev.	Pp	Detail
20131120	PH,DB	17	15, 27	Per customer comments... p. 15 - External Triggering Pins: Deleted the statement, “The trigger cable can drive either pin high with respect to the other; no particular polarity is required” and changed optocoupler from Vishay SFH6206 to Fairchild HCPL062N. p. 27 - Corrected the Lemo pinouts as follows: 1 = trigger 0+ 2 = reserved 3 = trigger 1- 4 = trigger 1+ 5 = reserved 6 = trigger 0- 7 = ground
20131120	PH	17	All	Corrected a mispopulated variable (name of product family) on all pages.
20111201	PH,RH	16	17,26	- Added section on LED status indicators for FOX boards. - Added pointer to LEDs on FOX board diagrams.
20110706	PH,RH	15	All	Split the original user’s guide for digital video framegrabbers into two: - This guide, for PCI, cPCI, and PMC framegrabbers (deleting most PCIe info) - A new, separate guide, for PCI Express framegrabbers
20110526	PH,RH	14c	18	In Table 3, added new products and updated firmware file information: - PCIe4 DVa C-Link - PCIe8 DVa C-Link - PCIe8 DVa CLS - PCIe4 DVa FOX
20100903	PH	14.11(b)	1	Under “Related Products,” added RAPID-V.
20100902	PH, RH, TL	14.10(a)	3	Under “Requirements,” added this paragraph: “Starting with PCIe 2.0 and depending on your BIOS, a PCIe bus ‘maximum payload size’ setting may be available in your CMOS setup pages (consult your computer manufacturer’s documentation for availability and usage). If so, EDT recommends a setting of 256 bytes to increase maximum bandwidth up to 2-3% over the bandwidth associated with a typical default setting of 128. However, if your PCIe bus is running another device with a smaller maximum payload size, this setting may adversely affect that device’s performance.”
20100800	PH,RH	14	All	Implemented multiple minor updates, corrections, and improvements.
20100700	PH,RH	14	All	Updated templates (width = 42 pc), formats, and pp. i-3. Changed “PROM” to “flash memory.” Incorporated IRIG-B document into this user’s guide as Appendix B.
20100408	PH,SC	14	[App. B]	Created stand-alone IRIG-B document for timestamping (with PCIe8 DV C-Link).
20070000	LW	0	All	Created guide for new app note (IRIG-B) on PCIe8 DV C-Link.