ADLINK
TECHNOLOGY INC.

PCIe-CML64FB/FP Series

PCI Express x4 Full configuration Frame Grabber Series

User's Manual

Manual Rev.	2.00
Revision Date:	January 17, 2008
Part No:	50-1C003-1000

Recycled Paper

Advance Technologies; Automate the World.

The information in this document is subject to change without prior notice in order to improve reliability, design, and function and does not represent a commitment on the part of the manufacturer.

In no event will the manufacturer be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the product or documentation, even if advised of the possibility of such damages.

Trademarks

NuDAQ, NuIPC, DAQBench are registered trademarks of ADLINK TECHNOLOGY INC.

Product names mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective companies.

Getting Service from ADLINK

Customer Satisfaction is top priority for ADLINK Technology Inc. Please contact us should you require any service or assistance.

ADLINK TECHNOLOGY INC.

Web Site:	http://www.adlinktech.com
Sales & Service:	Service@adlinktech.com
TEL:	+886-2-82265877
FAX:	+886-2-82265717
Address:	9F, No. 166, Jian Yi Road, Chungho City, Taipei, 235 Taiwan

Please email or FAX this completed service form for prompt and satisfactory service.

Company Information		
Company/Organization		
Contact Person		
E-mail Address		
Address		
Country		
TEL	FAX:	
Web Site		
Product Information		
Product Model		
Environment	OS: M/B: Chipset:	CPU: BIOS:

Please give a detailed description of the problem(s):

Table of Contents

List of Tables

1 Introduction

ADLINK's PCIe-CML64FB/FP is a Camera Link frame grabber that is based on the PCI Express x4 interface, and supports one channel Base/Medium/Full configuration, multi-tap area and line scan color and monochrome camera Link cameras.

The PCIe-CML64FB/FP series each have FPGA based (Field Programmable Gate Array) designs, and bring image acquisition flexibility, high performance, and pre-processing functionality. The advance technology of the PCIe-CML64FB/FP allows simultaneous image acquisition between cameras.

Camera Link is an industrial high-speed serial data and cabling standard. Created for easy connectivity between the PC and the camera, Camera Link provides simple, flexible cabling for high-speed, high-resolution digital cameras. A Camera Link cable is a slender 26-pin cable with 28-bit data, clock, and enable and control signals.

PCI-express (or PCIe), the latest standard serial interconnect architecture offering high performance on standard PCs, is now being used with vision applications for higher data transfer rates.

1.1 Features

- Support one Channel Base/Medium/Full Configuration Camera Link PCI Express x4 interface
- High-speed image transfer rates up to 640 MB/sec per channel
- Acquisition pixel clock rates up to 85 MHz
- 128 MB of 200MHz DDR SDRAM for acquisition
- RoHS compliant
- 2 programmable GPIO, RS422 level A,B,Z phase encoder Input, Line trigger Input, Line trigger Output, External Page trigger Input
- Supports Windows XP/XP embedded
- Supports VC++ 6.0, VB 6.0, BCB 6.0

2 Hardware Reference

2.1 PCIe-CML64FB/FP Specifications

Video Input

- Camera Link LVDS differential signals
- Base Configuration: Using Data1 MDR26 pins connector
- Medium and Full Configuration: Using Data1 and Data2 MDR26 pins connector
- Maximum camera link data rate: 85MHz

Camera Control

- RS-422 signal: CC1~CC4 control signal in Data1 MDR26 pins connector

External Signal Input

- RS-422 signal: External A, B, Z phase differential signal input, maximum frequency: 1 MHz
- Line trigger input
- Line trigger bypass output
- External page trigger input
- One channel Digital input , one channel Digital output

Form Factor

- PCI-express x4 interface

Dimension

- W x L : 174.62 x 111.15 mm

Operating Environment

- Temperature: 0 to 45°C
- Humidity: 5 to 90% RHNC

Storage Environment

- Temperature: 0 to 70°C
- Humidity: 0 to 95% RHNC

Power Requirements

- PCIe-CML64FB: +12 V max 0.6 A, +3.3 V max 3 A
- PCIe-CML64FP: +12 V max 1 A, +3.3 V max 4 A

2.2 PCIe-CML64FB/FP Layout

2.2.1 PCIe-CML64FB/FP Connectors & Pin Definitions

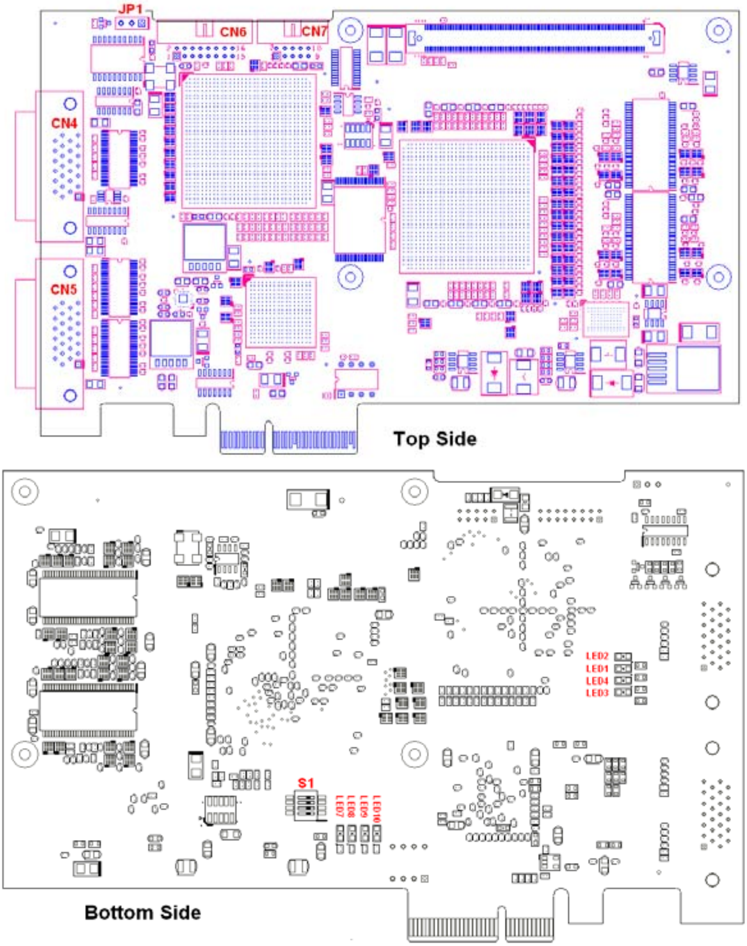
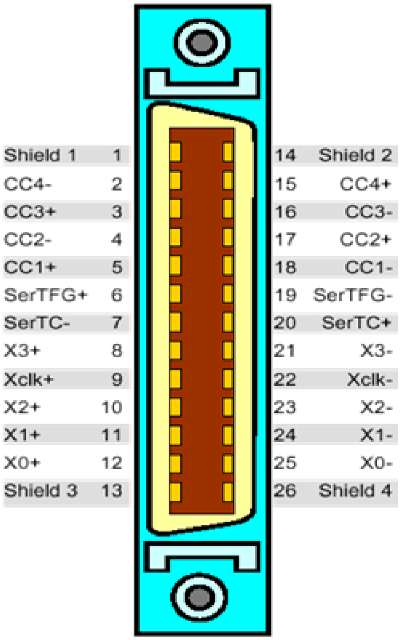


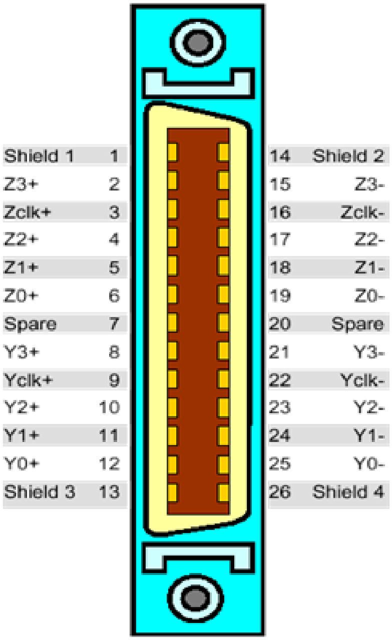
Figure 2-1: PCIe-CML64FB/FP Layout

CN4,CN5 Video Inputs



Shield 1	1	14	Shield 2
CC4-	2	15	CC4+
CC3+	3	16	CC3-
CC2-	4	17	CC2+
CC1+	5	18	CC1-
SerTFG+	6	19	SerTFG-
SerTC-	7	20	SerTC+
X3+	8	21	X3-
Xclk+	9	22	Xclk-
X2+	10	23	X2-
X1+	11	24	X1-
X0+	12	25	X0-
Shield 3	13	26	Shield 4

Base Configuration



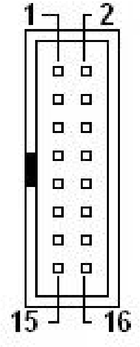
Shield 1	1	14	Shield 2
Z3+	2	15	Z3-
Zclk+	3	16	Zclk-
Z2+	4	17	Z2-
Z1+	5	18	Z1-
Z0+	6	19	Z0-
Spare	7	20	Spare
Y3+	8	21	Y3-
Yclk+	9	22	Yclk-
Y2+	10	23	Y2-
Y1+	11	24	Y1-
Y0+	12	25	Y0-
Shield 3	13	26	Shield 4

Medium & Full Configuration

CN4: Base Configuration Connector

CN5: Medium & Full Configuration Connector

CN6 Encoder & GPIO

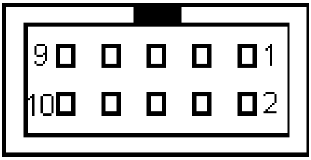


PIN	PIN NAME	TYPE	PIN	PIN NAME	TYPE
1	Encoder A phase A+ (Line Trigger +)	IN RS422	2	Encoder A phase A- (Line Trigger -)	IN RS422
3	Encoder B phase B+	IN RS422	4	Encoder B phase B-	IN RS422
5	Encoder Z phase Z+	IN RS422	6	Encoder Z phase Z-	IN RS422
7	GND	--	8	Page Trigger IN	IN TTL
9	Line Trigger in	IN TTL	10	Line Trigger start	IN TTL
11	Line Trigger out	OUT TTL	12	Digital Output	OUT TTL
13	GND	--	14	Digital Input	IN TTL
15	+5V Output	--			

Table 2-1: CN6 Encoder & GPIO

CN7 External COM port

Bypass camera control from system COM port.



PIN	PIN NAME	Note
1	--	
2	--	
3	RX	Connect with your system COM port TX signal
4	--	
5	TX	Connect with your system COM port RX signal
6	--	
7	--	
8	--	
9	GND	Connect with your system COM port GND
10	--	

Table 2-2: CN7 External COM port

JP1 Digital OUT mode setting

Digital out is transistor open collect type.

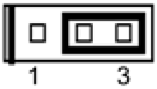
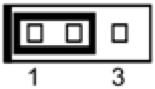
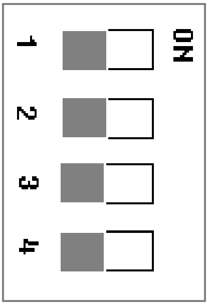
JP1	Setting	Function
	Pin 2-3 Short/Closed	Open collect; no pull to high
	Pin 1-2 Short/Closed	Open collect; pull high +5V (Default)

Table 2-3: JP1 Digital OUT mode setting

S1 Card ID Select



Pin	Signal Name	Default
1	Board ID Select 0	OFF
2	Board ID Select 1	OFF
3	System Jumper	OFF
4	System Jumper	OFF

Table 2-4: S1 Card ID Select

Card ID	Board ID Select 0	Board ID Select 1
0	OFF	OFF
1	ON	OFF
2	OFF	ON
3	ON	ON

Table 2-5: Card ID select table

Status LED

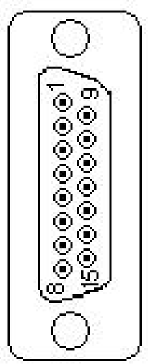
LED no.	Function
LED 1	No function (always ON)
LED 2	Frame grabber DMA run
LED 3	Frame grabber FIFO Empty
LED 4	Frame grabber FIFO Full
LED 7	PCI Express Lane 0 Status Indicators , Lane active (LED is On)
LED 8	PCI Express Lane 1 Status Indicators , Lane active (LED is On)
LED 9	PCI Express Lane 2 Status Indicators , Lane active (LED is On)
LED10	PCI Express Lane 3 Status Indicators , Lane active (LED is On)

Table 2-6: Status LED

2.2.2 Encoder & GPIO Extension cable

Extension cable connect CN6

Extension cable connector is D-sub 15 pin female



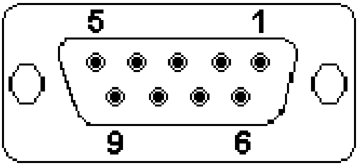
PIN	PIN NAME	TYPE	PIN	PIN NAME	TYPE
1	Encoder A phase A+ (Line Trigger +)	IN RS422	9	Encoder A phase A- (Line Trigger -)	IN RS422
2	Encoder B phase B+	IN RS422	10	Encoder B phase B-	IN RS422
3	Encoder Z phase Z+	IN RS422	11	Encoder Z phase Z-	IN RS422
4	GND	--	12	Page Trigger IN (Falling Edge Trigger)	IN TTL
5	Line Trigger in (Rising Edge Trigger)	IN TTL	13	Line Trigger start (Active Low)	IN TTL
6	Line Trigger out	OUT TTL	14	Digital Output	OUT TTL
7	GND	--	15	Digital Input	IN TTL
8	+5V Output	--			

Table 2-7: Extension cable connect CN6

2.2.3 External COM port extension cable

Extension cable connect CN7

Extension cable connector is D-sub 9 pin female



PIN	PIN NAME	Note
1	--	
2	RX	Connect with your system’s COM port TX signal
3	TX	Connect with your system’s COM port RX signal
4	--	
5	GND	Connect with your system’s COM port GND
6	--	
7	--	
8	--	
9	--	
10	--	

Table 2-8: External COM port extension cable

2.2.4 Trigger modes

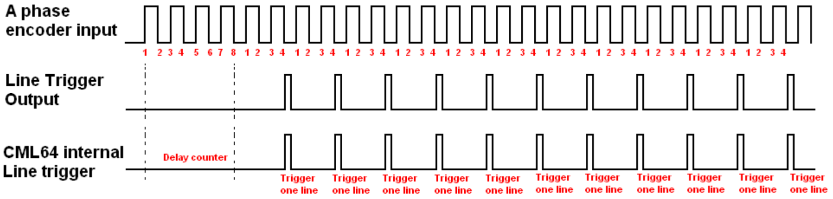
PCIe-CML64FB/FP supported trigger mode list

Trigger Mode	Input Signal RS-422 Level	Input signal TTL Level	Line Trigger Start	Line Trigger Out	Encoder Counter	Line Counter	Plus Delay Counter	Plus Step Counter
A phase	O	X	X	O	O	O	O	O
A,B phase	O	X	X	O	O	O	O	O
A,B,Z phase	O	X	X	O	O	O	O	O
Line Trigger	O	O	O	O	X	O	X	X
Page Trigger	X	O	X	X	X	X	X	X

Table 2-9: PCIe-CML64FB/FP supported trigger modes

- Line trigger start (CN6 pin10): When this function is enabled, line trigger start signal can be used to control line trigger
- Line trigger out (CN6 pin11): Bypass CML64 internal Line trigger output
- Encoder counter: Count encoder input plus number
- Line counter: Count line trigger input plus number
- Plus Delay counter: Setting encoder delay number
- Plus Step counter: Setting encoder step number

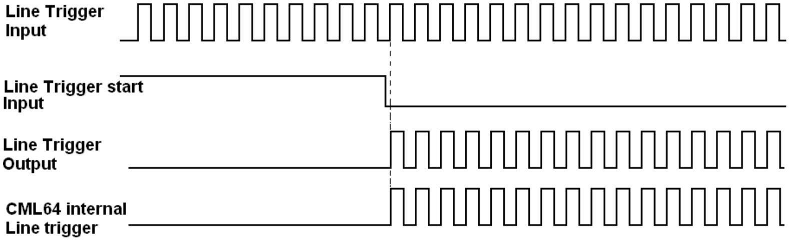
Example timing 1. A phase trigger mode



Plus delay counter setting :8

Plus step counter setting : 4

Example timing 2. Line trigger mode



Enable Line trigger start function

3 Installation Guide

3.1 Hardware Installation

Use the following steps to install the PCIe-CML64FB/FP series board on the PCI express bus:

1. Remove the computer cover using the instructions from the computer manual.
2. Check that there is an empty PCI express slot to accommodate the PCIe-CML64FB/FP board. If there is not an empty slot, remove a PCI express board to make room and take note of the chosen slot number.
3. Remove the blank metal plate located at the back of the selected slot (if any). Keep the removed screw to fasten the PCIe-CML64FB/FP board after installation.
4. Carefully position the PCIe-CML64FB/FP in the selected PCI express slot as illustrated below. If using a tower computer, orient the board to suit the board slots.
5. Once perfectly aligned with an empty slot, press the board firmly but carefully into the connector.
6. Anchor the board by replacing the screw.

Note: PCIe-CML64FB/FP can install at PCI express X4,X8,X16 slots.

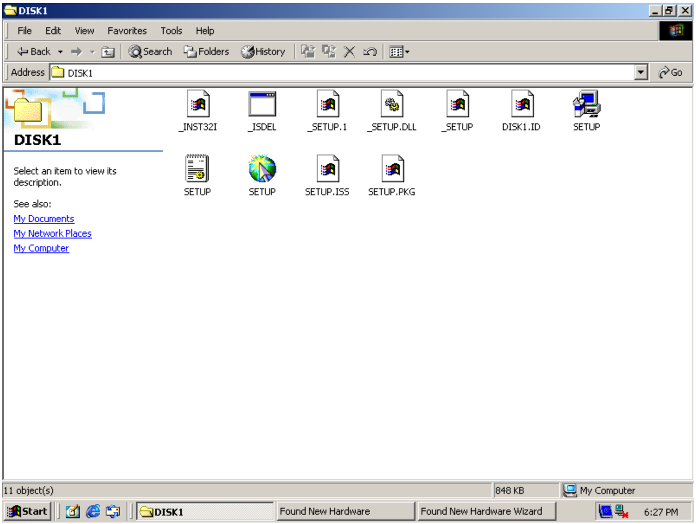
Some motherboard PCI Express x16 slots only support VGA cards. When installing A PCIe-CML64FB/FP on one of these slots, the speed will be decreased to a PCI express X1 speed.

3.2 Driver Installation

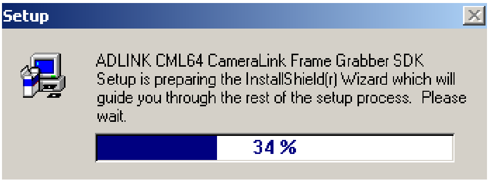
3.2.1 WDM Driver Installation

Note:Do not plug in the hardware before installing the software driver.

1. Click **Setup**



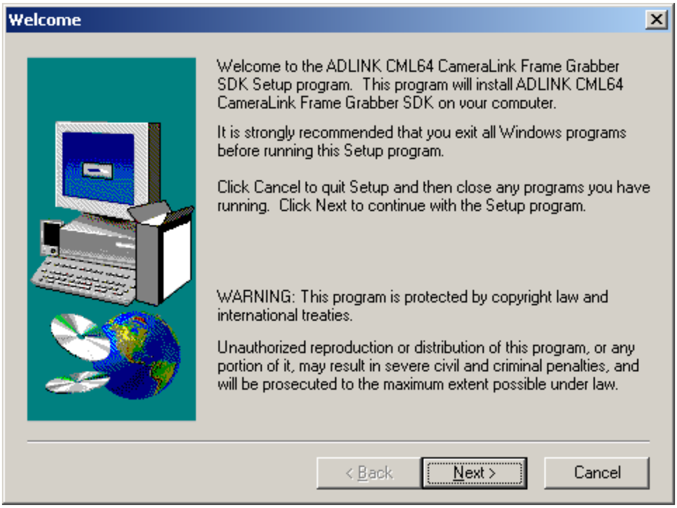
2. The driver will begin installing.



ADLINK CML64 CameraLink Frame Grabber SDK Setup is preparing the InstallShield(r) Wizard which will guide you through the rest of the setup process. Please wait.

34 %

3. Click next until driver installation is complete.



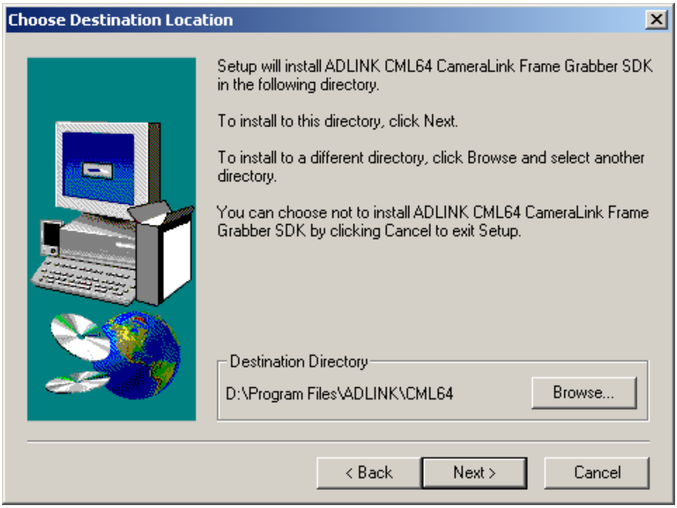
Welcome to the ADLINK CML64 CameraLink Frame Grabber SDK Setup program. This program will install ADLINK CML64 CameraLink Frame Grabber SDK on your computer.

It is strongly recommended that you exit all Windows programs before running this Setup program.

Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.



Setup will install ADLINK CML64 CameraLink Frame Grabber SDK in the following directory.

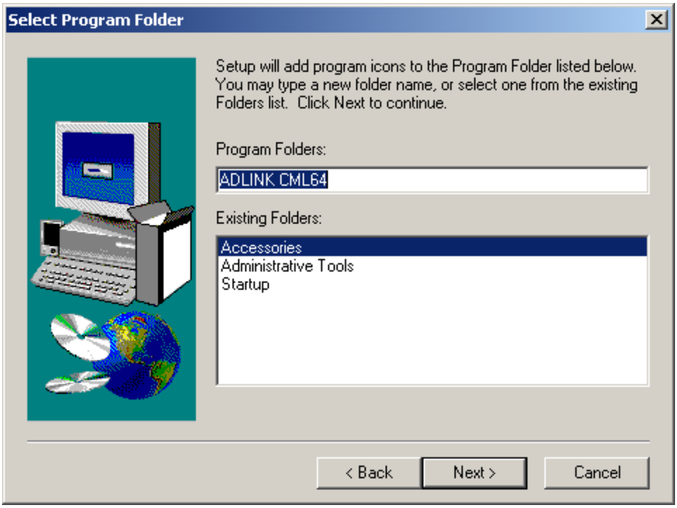
To install to this directory, click Next.

To install to a different directory, click Browse and select another directory.

You can choose not to install ADLINK CML64 CameraLink Frame Grabber SDK by clicking Cancel to exit Setup.

Destination Directory

D:\Program Files\ADLINK\CML64



Select Program Folder

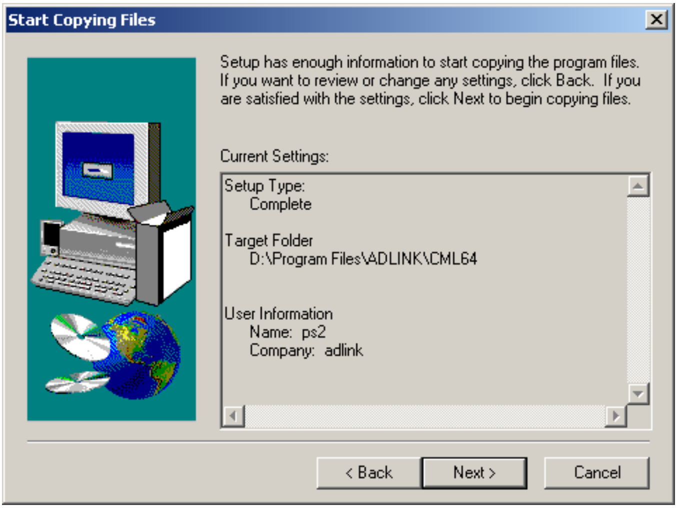
Setup will add program icons to the Program Folder listed below. You may type a new folder name, or select one from the existing Folders list. Click Next to continue.

Program Folders:

ADLINK CML64

Existing Folders:

Accessories
Administrative Tools
Startup



Start Copying Files

Setup has enough information to start copying the program files. If you want to review or change any settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

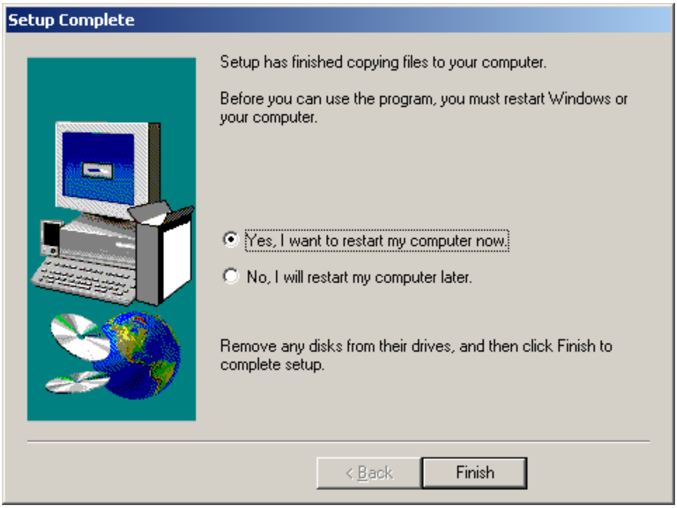
Current Settings:

Setup Type:
Complete

Target Folder
D:\Program Files\ADLINK\CML64

User Information
Name: ps2
Company: adlink

4. Click finish and restart system.



Setup Complete

Setup has finished copying files to your computer.

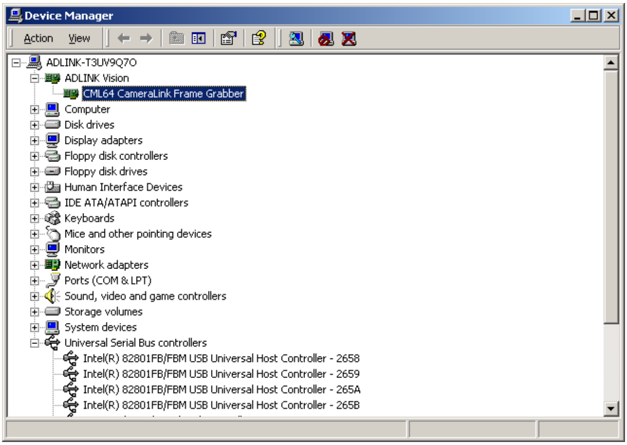
Before you can use the program, you must restart Windows or your computer.

● Yes, I want to restart my computer now.

○ No, I will restart my computer later.

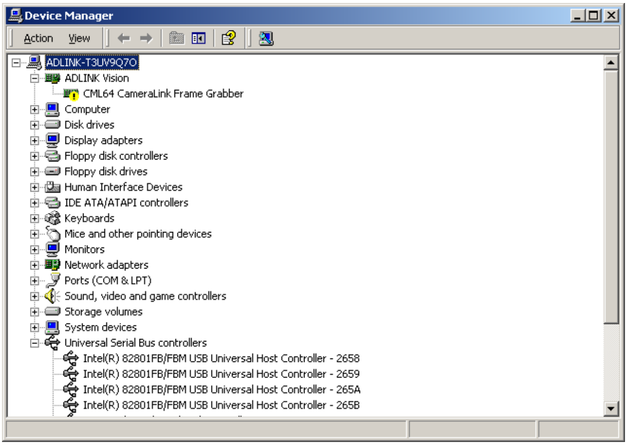
Remove any disks from their drives, and then click Finish to complete setup.

5. Go to system control panel and check multimedia devices. The system control panel should be as follows:



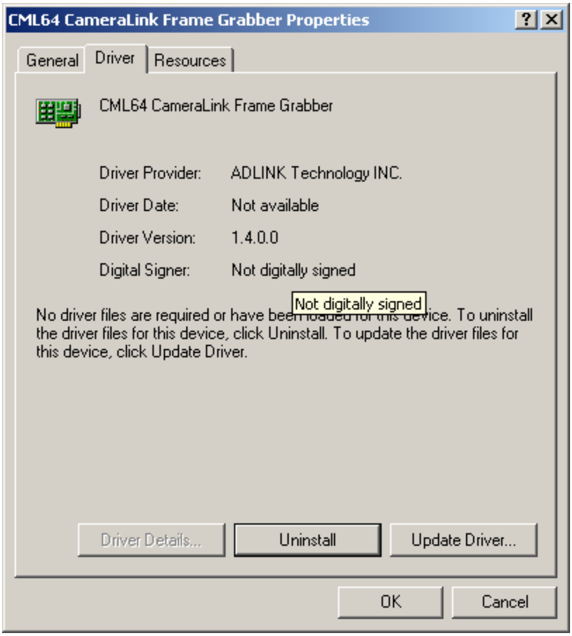
- ADLINK-T3UV9Q7O
  - ADLINK Vision
    - CML64 CameraLink Frame Grabber
  - Computer
  - Disk drives
  - Display adapters
  - Floppy disk controllers
  - Floppy disk drives
  - Human Interface Devices
  - IDE ATA/ATAPI controllers
  - Keyboards
  - Mice and other pointing devices
  - Monitors
  - Network adapters
  - Ports (COM & LPT)
  - Sound, video and game controllers
  - Storage volumes
  - System devices
  - Universal Serial Bus controllers
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 2658
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 2659
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 265A
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 265B

6. If you see a yellow marker in front of the new driver name, you need to setup the driver manually.



- ADLINK-T3UV9Q7O
  - ADLINK Vision
    - CML64 CameraLink Frame Grabber
  - Computer
  - Disk drives
  - Display adapters
  - Floppy disk controllers
  - Floppy disk drives
  - Human Interface Devices
  - IDE ATA/ATAPI controllers
  - Keyboards
  - Mice and other pointing devices
  - Monitors
  - Network adapters
  - Ports (COM & LPT)
  - Sound, video and game controllers
  - Storage volumes
  - System devices
  - Universal Serial Bus controllers
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 2658
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 2659
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 265A
    - Intel(R) 82801FB/FBM USB Universal Host Controller - 265B

7. Right click on Multimedia Controller (which is an audio device), then select Properties from the popup menu. Follow the steps to complete the driver installation.



CML64 CameraLink Frame Grabber

Driver Provider: ADLINK Technology INC.

Driver Date: Not available

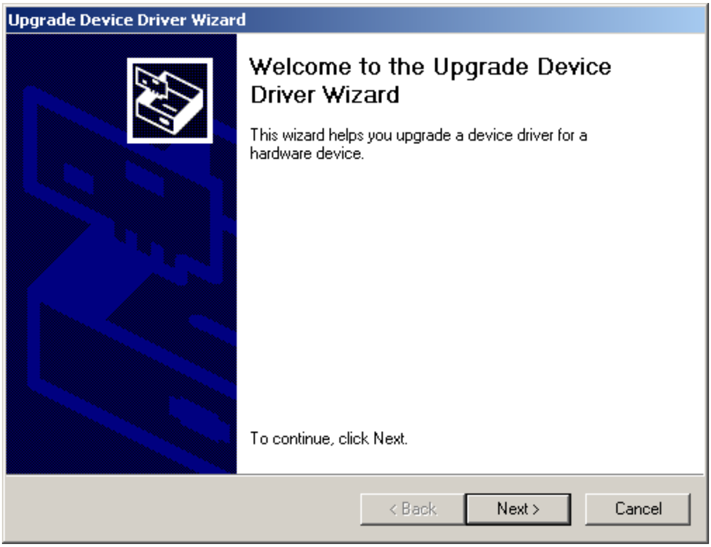
Driver Version: 1.4.0.0

Digital Signer: Not digitally signed

Not digitally signed

No driver files are required or have been loaded for this device. To uninstall the driver files for this device, click Uninstall. To update the driver files for this device, click Update Driver.

8. Click Update Driver.



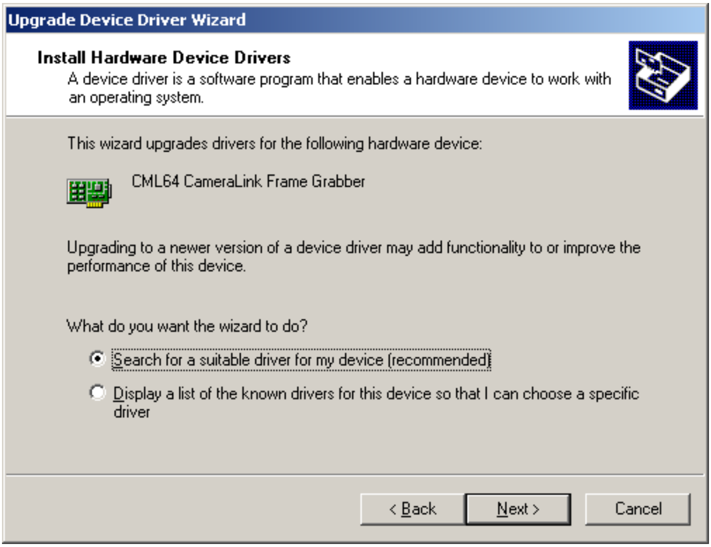
Upgrade Device Driver Wizard

Welcome to the Upgrade Device Driver Wizard

This wizard helps you upgrade a device driver for a hardware device.

To continue, click Next.

9. Click Next.



Upgrade Device Driver Wizard

Install Hardware Device Drivers

A device driver is a software program that enables a hardware device to work with an operating system.

This wizard upgrades drivers for the following hardware device:

CML64 CameraLink Frame Grabber

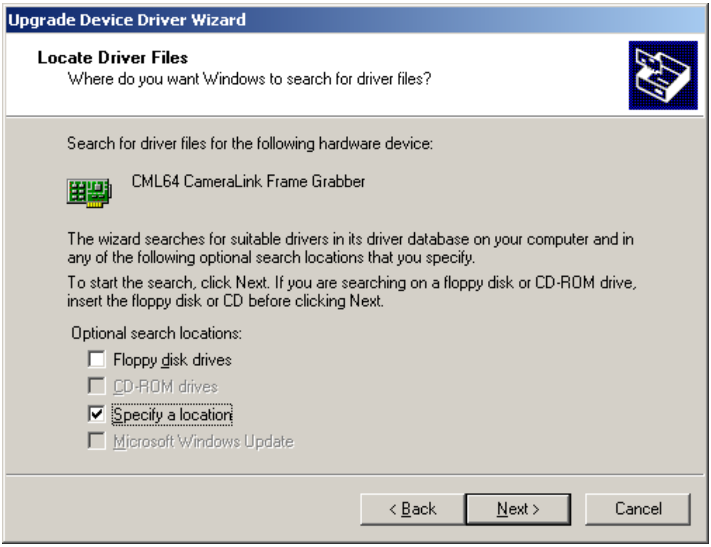
Upgrading to a newer version of a device driver may add functionality to or improve the performance of this device.

What do you want the wizard to do?

● Search for a suitable driver for my device (recommended)

○ Display a list of the known drivers for this device so that I can choose a specific driver

10.Click Next.



Upgrade Device Driver Wizard

Locate Driver Files

Where do you want Windows to search for driver files?

Search for driver files for the following hardware device:

CML64 CameraLink Frame Grabber

The wizard searches for suitable drivers in its driver database on your computer and in any of the following optional search locations that you specify.

To start the search, click Next. If you are searching on a floppy disk or CD-ROM drive, insert the floppy disk or CD before clicking Next.

Optional search locations:

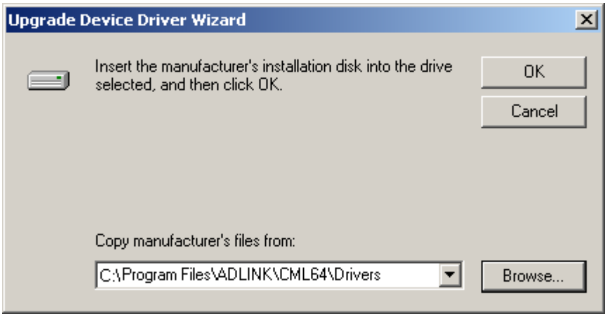
□ Floppy disk drives

□ CD-ROM drives

☑ Specify a location

□ Microsoft Windows Update

11.Specify a location and then click next.



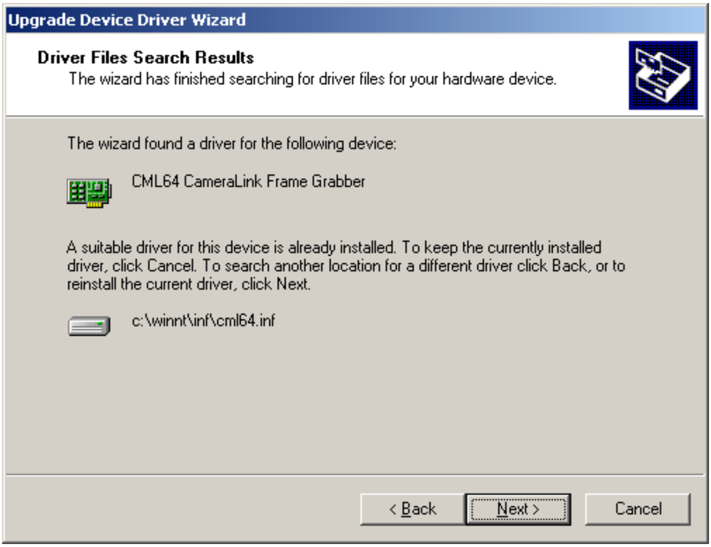
Upgrade Device Driver Wizard

Insert the manufacturer's installation disk into the drive selected, and then click OK.

Copy manufacturer's files from:

C:\Program Files\ADLINK\CML64\Drivers

12. Input the location of driver installed in step 3, for example, “C:\Program Files\ADLINK\CML64\Drivers”. Click OK.



Upgrade Device Driver Wizard

Driver Files Search Results

The wizard has finished searching for driver files for your hardware device.

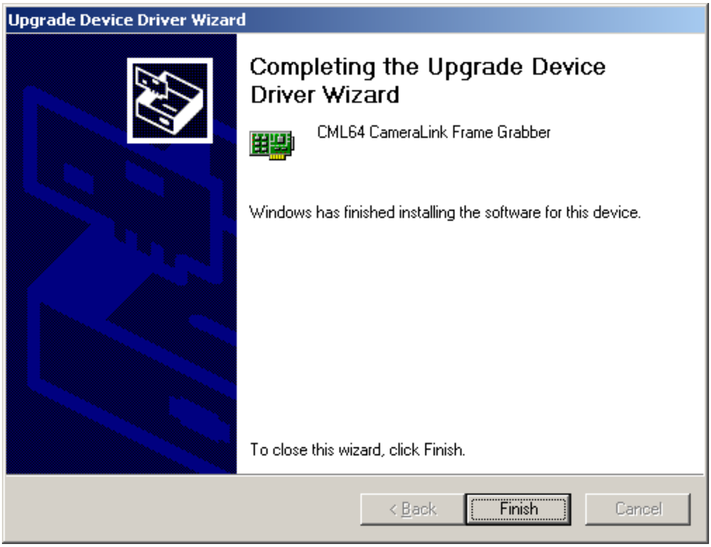
The wizard found a driver for the following device:

CML64 CameraLink Frame Grabber

A suitable driver for this device is already installed. To keep the currently installed driver, click Cancel. To search another location for a different driver click Back, or to reinstall the current driver, click Next.

c:\winnt\inf\cml64.inf

13. Click Next.



Upgrade Device Driver Wizard

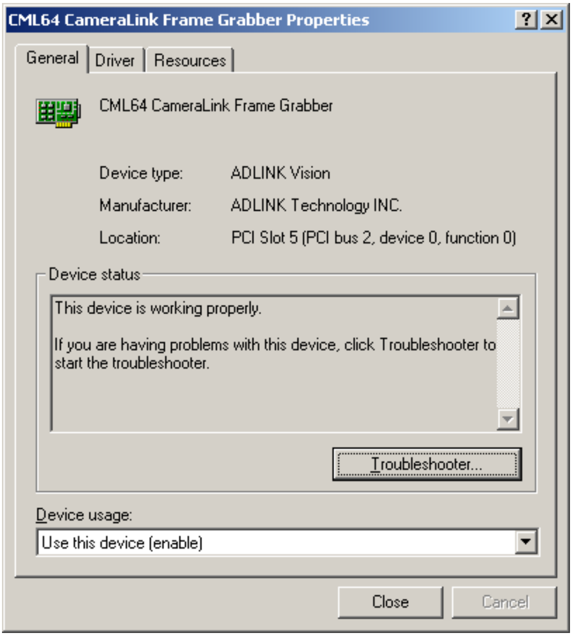
Completing the Upgrade Device Driver Wizard

CML64 CameraLink Frame Grabber

Windows has finished installing the software for this device.

To close this wizard, click Finish.

14. Click Finish to complete this wizard.



CML64 CameraLink Frame Grabber Properties

CML64 CameraLink Frame Grabber

Device type: ADLINK Vision

Manufacturer: ADLINK Technology INC.

Location: PCI Slot 5 (PCI bus 2, device 0, function 0)

Device status

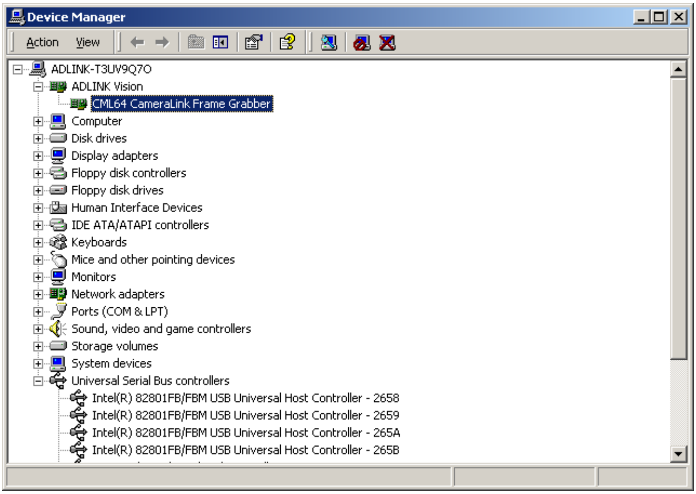
This device is working properly.

If you are having problems with this device, click Troubleshooter to start the troubleshooter.

Device usage:

Use this device (enable)

15. This device should be working properly.



Device Manager

```
ADLINK-T3UV9Q7O
  ADLINK Vision
    CML64 CameraLink Frame Grabber
  Computer
  Disk drives
  Display adapters
  Floppy disk controllers
  Floppy disk drives
  Human Interface Devices
  IDE ATA/ATAPI controllers
  Keyboards
  Mice and other pointing devices
  Monitors
  Network adapters
  Ports (COM & LPT)
  Sound, video and game controllers
  Storage volumes
  System devices
  Universal Serial Bus controllers
    Intel(R) 82801FB/FBM USB Universal Host Controller - 2658
    Intel(R) 82801FB/FBM USB Universal Host Controller - 2659
    Intel(R) 82801FB/FBM USB Universal Host Controller - 265A
    Intel(R) 82801FB/FBM USB Universal Host Controller - 265B
```

4 CamCreator Utility

Before running the CamCreator utility, ensure that all hardware installed is configured correctly. This chapter outlines how to establish a vision system and how to manually control PCIe-CML64 cards to verify correct operation. CamCreator provides a simple yet powerful means to setup, configure, test, and debug the system from an easy-to-use point and click interface.

Note: CamCreator is available for Windows-based operating system with a recommended screen resolution higher than 800x600.

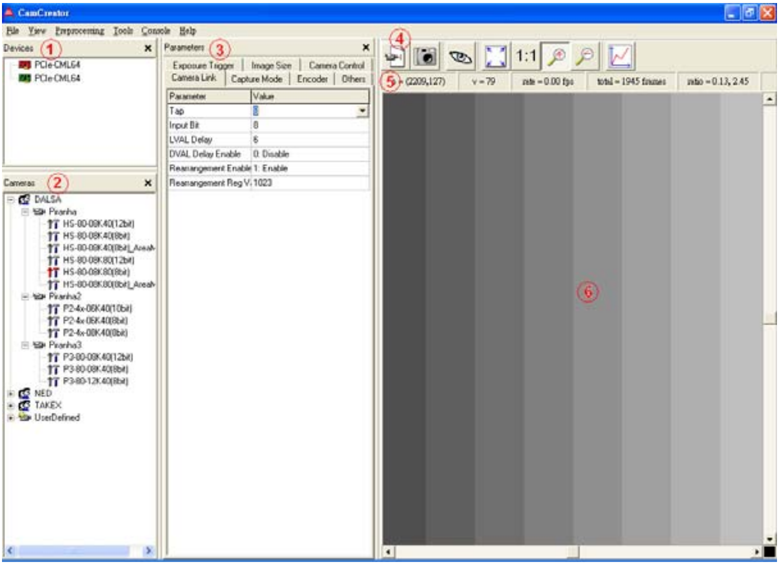
4.1 Overview

CamCreator offers the following features:

1. Automatic detection of acquisition hardware
2. Creation and modification of camera file
3. Instant modification of camera parameters
4. Serial communication
5. Opening and saving operation of still image file
6. Direct access to general purpose I/Os
7. Captured Image viewing

4.2 Component Description

Start the utility:



Parameter	Value
Tap	8
Input Bit	8
LVAL Delay	6
DVAL Delay Enable	0: Disable
Rearrangement Enable	1: Enable
Rearrangement Reg V:	1023

(2209,127) | v = 79 | rate = 0.00 fps | total = 1945 frames | ratio = 0.13, 2.45

1.Devices panel

The list window lists the PCIe-CML64 cards on the local computer.

2. Cameras panel

The tree browser window lists all available cameras and their camera files.

3. Parameters panel

The tab window lists all adjustable parameters.

4. Toolbar

The toolbar helps simplify operations.

5. Status bar

The status bar shows information about the captured image.

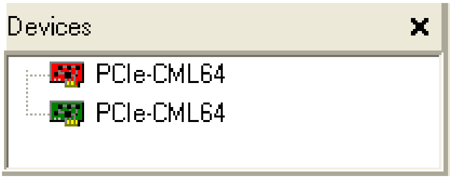
6. Display panel

This window shows a captured image.

4.3 Operation theory

CamCreator provides many functions for the PCIe-CML64 card as described below:

4.3.1 Devices panel



Devices ✕

PCIe-CML64

PCIe-CML64

🟥 **Current active device**

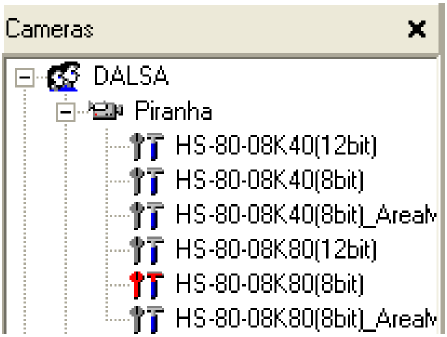
All operations apply to this device.

🟩 **Inactive device**

Click the device name after this icon to activate the device. Only one device can be activated at a time. If you select a standby device, the device currently active will become inactive.

✕ **Close this panel**

4.3.2 Cameras panel



Cameras
- DALSA
  - Piranha
    - HS-80-08K40(12bit)
    - HS-80-08K40(8bit)
    - HS-80-08K40(8bit)_Area
    - HS-80-08K80(12bit)
    - HS-80-08K80(8bit)
    - HS-80-08K80(8bit)_Area

Camera provider

Camera type

Camera file

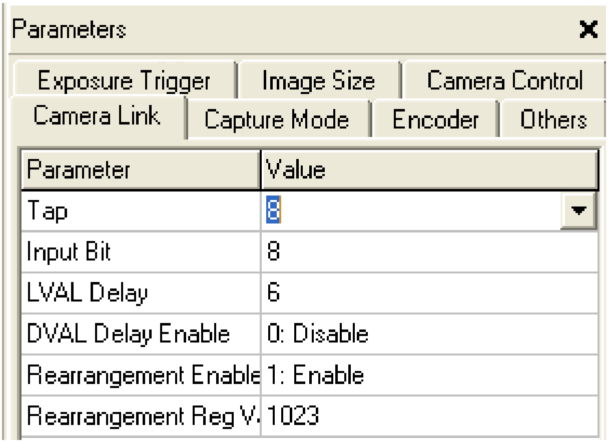
Current loaded camera file

Select an appropriate camera file according to your camera. CamCreator will load this camera file to the device and show its parameters in the parameters panel.

✖ Close this panel

4.3.3 Parameters panel

Parameters panel shows the parameters of camera file and other adjustable parameters that are aggregated on “Others” tab. Any modification instantly applies to the current active device. Users can save the camera file with the modification parameters by “Save Cam File” command in the main menu.



Parameter	Value
Tap	8 ▾
Input Bit	8
LVAL Delay	6
DVAL Delay Enable	0: Disable
Rearrangement Enable	1: Enable
Rearrangement Reg V.	1023

▾ **Pick list button**

Clicking this button shows an available list.

... **Ellipsis button**

Clicking this button opens an editing window for inputing a value.

✖ **Close this panel**

4.3.4 Toolbar

Continue Grab

Start to grab images and display them on the display panel. Click it again to stop the grab. This is a toggle button.

Snap Shot

Capture an image and display it on the display panel.

Hide Image

Hide or unhide image. This is a toggle button.

Fit Size

Fit the image to the whole display panel.

1:1

Original Size

Restore to original size.

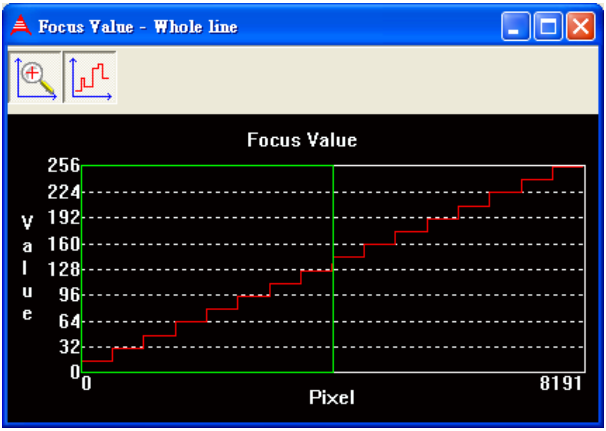
Zoom In

Zoom Out

Focus Value

Open a chart to see the pixel values of a selected horizontal line on the image. The display image shows a red horizontal line on it. Click the display image to move the selected line.

Focus value window is shown below:



Zoom in

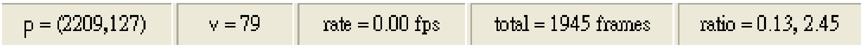
Open a window to zoom in the green rectangle region.

Differential

Open a window to show the slop of the line for the green rectangle region.

Drag the vertical green line to resize the green rectangle.

4.3.5 Status bar



p = (2209,127) | v = 79 | rate = 0.00 fps | total = 1945 frames | ratio = 0.13, 2.45

From left to right, the panel items are: cursor position, pixel value, frame rate, total captured frames, and magnification (horizontal ratio, vertical ratio).

4.3.6 Main menu

File menu

Open Cam File

Load camera parameters from a camera file.

Save Cam File

Save current camera parameters to the current loaded camera file.

Save Cam File As

Save current camera parameters to a new camera file.

Delete Cam File

Delete the currently loaded camera file.

Open Image

Open an image from a file and display it in the display panel.

Save Image

Save current displaying image to a bitmap file.

Exit

Terminate CamCreator.

View menu

Devices

Hide or unhide Devices panel.

Cameras

Hide or unhide Cameras panel.

Parameters

Hide or unhide Parameters panel.

Tools menu

Counter

Open a window for displaying the encoder counter and the line counter.

I/Os

Open a window for reading and writing the general purpose I/Os.

Console menu

Setup

Open a window for setting serial communication.

Connection

Open a terminal window that allows user to communicate with camera.

5 Function Library

This chapter describes APIs of the CML64 CameraLink frame grabber.

5.1 Function List

Table 5-1 lists all CML64 API functions and provides short descriptions for them. For the details, please reference to the corresponding sections.

Function name	Description
System functions	
CML64_GetBoardNum	Get the number of CML64 cards that are on your system. You can call this function before opening any CML64 card.
CML64_OpenDevice	Open CML64 and initialize FPGA to default status. You should call this function before any other CML64 API except CML64_GetBoardNum().
CML64_CloseDevice	Close CML64 and release all resource that created by CML64 library. Call this function before your application closed.
CML64_GetDLLVersion	Get the version of current dll.
CML64_GetPSMFPGAVersion	Get the version of FPGA on the image pre-processing module.
CML64_GetMainFPGAVersion	Get the version of main FPGA on CML64 card.
CML64_GetImageFPGAVersion	Get the version of image FPGA on CML64 card.
Image grabbing functions	
CML64_SetCallback	Set callback function that CML64 library will call when one image frame is ready.
CML64_Snap	Call this function to grab one image frame.
CML64_Live	Call this function to grab images continuously. CML64 will stop grabbing images when it grabs enough frame numbers assigned by users or when CML64_Stop() is called.
CML64_Stop	Call this function to stop grabbing images. Usually, this function will be called after using CML64_Live().
CML64_IsGrabbing	Check whether CML64 is grabbing images.
Configuration functions	
CML64_LoadRBFFile	Load a RBF file to set FPGA registers on the image pre-processing module.
CML64_IsRBFLoaded	Check whether the RBF file is loaded.
CML64_LoadCameraFile	Load a camera file and set the parameters in the camera file to FPGA registers.
CML64_SetCamFileParameter	Set CamFile parameters to FPGA registers.
CML64_GetCamFileParameter	Get CamFile parameters from FPGA registers.

Function name	Description
CML64_SetImageSizeForLine	Set image dimension.
CML64_GetImageSizeForLine	Get current image dimension.
CML64_SetExposureTriggerForLine	Set CCD exposure time and related configuration.
CML64_GetExposureTriggerForLine	Get current CCD exposure time and related configuration.
CML64_SetCameraControl	Set camera control signal configuration.
CML64_GetCameraControl	Get current camera control signal configuration.
CML64_SetCameraLinkCfg	Set CameraLink related setting by CCD setting.
CML64_GetCameraLinkCfg	Get current CameraLink related setting.
CML64_SetCaptureMode	Set capture mode and timeout.
CML64_GetCaptureMode	Get current capture mode and timeout.
CML64_SetEncoder	Set encoder signal configuration. This function is used when capture mode is not “Normal”.
CML64_GetEncoder	Get current encoder signal configuration.
CML64_GetEncoderCounter	Get the value of encoder counter. It means the total number of received triggers.
CML64_ResetEncoderCounter	Reset the encoder counter to 0.
CML64_SetLineTriggerInLevel	Set line trigger input level : TTL or RS422.
CML64_GetLineTriggerInLevel	Get line trigger input level : TTL or RS422.
CML64_GetLineCounter	Get the value of line counter. It means the total number of captured lines.
CML64_ResetLineCounter	Reset the line counter to 0.
CML64_LineTriggerStartEnable	Start grabbing images when the line trigger start signal is high.
Functions for image buffer setting	
CML64_SetDMABufferAddr	Change the image buffer address that will be used by CML64_Snap().
CML64_SetDMARingBufferNum	Change the ring buffer number.
CML64_SetDMARingBufferAddr	Change the ring buffer number and address. User should create the ring buffer by himself.
CML64_SetDMARingBufferDefault	Set the ring buffer to default setting.
Functions for image pre-processing module setting	
CML64_PSMReset	Reset image pre-processing module to default status.
CML64_PSMSetRegValue	Set value to the image pre-processing module register.
CML64_PSMGetRegValue	Read value from the image pre-processing module register.
CML64_PSMWriteArray	Write a byte array to the image pre-processing module.
CML64_PSMPPEnable	Enable PSM pre-processing function. If enabled, images will go through the image pre-processing module, and you should process these images manually. This function is only effective in CML64 FP, and it must be called right after CML64_LoadCameraFile. In CML64FB, do not use this function.
Functions for serial port communication	

Function name	Description
CML64_SRLInitialize	Initial the serial port communication device on the CML64 card.
CML64_SRLClose	Close the serial port communication device and release resource.
CML64_SRLSetBaudRate	Set serial port communication baud rate. Default is 9600.
CML64_SRLWriteCommand	Write control command from CML64 card to CameraLink CCD.
CML64_SRLReadCommand	Read the CameraLink CCD response message.
CML64_SRLRxIsEmpty	Check whether the receive buffer of serial port communication device is empty.
Functions for GPIO	
CML64_SetDOStatus	Set general purpose digital output status.
CML64_GetDIStatus	Get general purpose digital input status.

Table 5-1: CML64 API function list

5.2 Camera File

This section describes the context of camera file.

5.2.1 Camera File Introduction

5.2.1.1 CameraInfo Section

Name

The CCD name.

5.2.1.2 ExposureTrigger Section

This section describes line trigger signal and PRIN signal setting. **Parameters** of this section are useful when the CCD operates in exposure mode.

LineTriggerPeriod

This parameter is the EXSYNC signal from CML64 to CCD. It is used to control the line period of CCD to grab images.

Value: 0 - 6553 (microsecond)

LineTriggerPolarity

The line trigger signal level.

0: Positive

1: Negative

PRINPeriod

This parameter is used to control the exposure time of CCD to grab images. The PRINPeriod should be less than or equal to the LineTriggerPeriod.

Value: 0 - 6553 (microsecond)

PRINPolarity

The PRIN signal level.

0: Positive

1: Negative

5.2.1.3 ImageSize Section

This section contains the configuration information of image frame dimension.

Width

Width of an image frame. For each CameraLink CCD, the maximal value ofth is bounded by the resolution of the CCD. The sum of this parameter and XOffset should be less than or equal to the resolution of CCD.

Value: 0 - 16368

Height

Height of an image frame. It means the number of lines you want to grab for one image frame.

Value: 0 - 65535

XOffset

This parameter describes how many pixels you want delay for grabbing one line. The sum of this parameter and Width should be less than or equal to the resolution of CCD.

Value: 0 - 16368

YOffset

This parameter describes how many lines you want delay for grabbing one image frame.

Value: 0 - 65536

5.2.1.4 CameraControl Section

This section contains configuration information for camera control signals.

CameraControlOutputPort

This parameter indicates which output port the camera control signal uses.

CameraControlConnecter

This parameter indicates which connecter the camera control signal pass through.

0: Base Connector

1: Medium/Full Connector

CC1Polarity

Camera control 1 (CC1) signal level. 0: Low, 1: High

CC2Polarity

Camera control 2 (CC1) signal level. 0: Low, 1: High

CC3Polarity

Camera control 3 (CC1) signal level. 0: Low, 1: High

CC4Polarity

Camera control 4 (CC1) signal level. 0: Low, 1: High

5.2.1.5 CameraLink Section

This section indicates the configuration of CameraLink mode of CCD and the corresponding configuration for CML64.

Tap

This parameter indicates the number of taps of CCD current setting.

Value: 1 - 8

InputBit

This parameter indicates the number of bits of CCD current setting.

Value: 8/10/12/14/16/24/32/40

LVALDelay

Define the LVAL delay for CML64 to receive and rearrange the image data.

Value: 0 - 255

DVALDelayEnable

This parameter indicates whether the DVAL delay function is enable or not.

0: Disable

1: Enable

RearrangementEnable

Enable the rearrangement function of CML64 to rearrange the image data from camera.

0: Disable

1: Enable

RearrangementRegValue

This parameter tells CML64 how to rearrange the image data from camera. The value will be different because of different setting of Tap and InputBit.

Value: 0 - 0xFFFFFFFF

5.2.1.6 CaptureMode Section

This section indicates with which capture mode CML64 grabs images.

Mode

Capture mode for CML64 to grab images.

Mode	Value	Description
Normal	1	Normally grab images.
Encoder Scan	2	Grab images according to encoder pulses and the encoder setting listed in section 5.2.1.7. The trigger input level can only be RS422. Users can get the number of input encoder pulses by encoder counter.
Line Trigger	3	Grab images when receiving input line triggers. The trigger input level can be TTL or RS422. Users can get the number of input line triggers by line counter.
Page Trigger	4	Grab images (a page of frame) according to Height defined in the camera file.

Timeout

The time-out interval in millisecond for grabbing one image frame. It occurs a time-out error if the interval had elapsed.

Value: 1 - 4294967295

5.2.1.7 Encoder Section

This section contains the configuration information of encoder. CML64 can receive A, B, Z phase from GPIO connector.

EncoderStartMode

This parameter indicates when to start counting input encoder pulses. There are three ways to start.

Mode	Value	Description
Normal start	0	Capture image normally starts to count input encoder pulses.
Z phase start	2	Capture image starts to count input encoder pulses when it receives a Z phase input pulse.

PulsePhase

This parameter indicates whether the input encoder pulse is AB phase or A phase.

0: AB phase

1: A phase

PulseDirection

This parameter indicates the direction of input encoder pulse when PulsePhase is AB phase.

0: Clockwise rotation (CW)

1: Counter Clockwise rotation (CCW)

PulseDelay

This parameter is used to delay grabbing images. It means that CML64 will start to grab images after PulseDelay counts of encoder pulse. CML64 increases the pulse counts at both raising edge and falling edge.

Value: 0 - 4294967295

PulseStep

CML64 grabs one line according to this parameter. CML64 increases the PulseStep count at both raising edge and falling edge of an encoder pulse. For example, if PulseStep equals 4, CML64 will grab one line every 2 encoder pulse.

Value:2 - 65535 (A phase)

4 - 65535 (AB phases)

5.2.2 Supported CameraLink CCDs

The following table lists the CameraLink CCD supported now and their camera files.

Company	CCD	Camera File
DALSA	Piranha2 P2-4x-06K40	DALSA Piranha2 P2-4x-06K40(8bit).ini
		DALSA Piranha2 P2-4x-06K40(10bit).ini
	Piranha2 P2-4x-08K40	DALSA Piranha2 P2-4x-08K40(8bit).ini
	Piranha3 P3-80-08K40	DALSA Piranha3 P3-80-08K40(8bit).ini
		DALSA Piranha3 P3-80-08K40(12bit).ini
	Piranha3 P3-80-12K40	DALSA Piranha3 P3-80-12K40(8bit).ini
	Piranha HS-80-08K40	DALSA Piranha HS-80-08K40(8bit).ini
		DALSA Piranha HS-80-08K40(12bit).ini
	Piranha HS-80-08K80	DALSA Piranha HS-80-08K80(8bit).ini
		DALSA Piranha HS-80-08K80(12bit).ini
NED	CLISBEE XCM8060	NED CLISBEE XCM8060(8bit).ini
		NED CLISBEE XCM8060(10bit).ini
	Eagle e7450D	NED Eagle e7450D(8bit).ini
TAKEX	TL-7400RCL	TAKEX TL-7400RCL(8bit).ini
		TAKEX TL-7400RCL(10bit).ini

Table 5-2: Supported CameraLink CCDs

5.2.3 Examples

The following is the camera file from ADLINK for DALSA Piranha2 (CL mode: 8Tap/8Bit, Resolution: 8 k).

```
[CameraInfo]
Name=Piranha2-8K(8bit/4taps)

[ExposureTrigger]
LineTriggerPeriod=300
LineTriggerPolarity=1
PRINPeriod=300
PRINPolarity=1

[ImageSize]
Width=8192
Height=512
XOffset=0
YOffset=0

[CameraControl]
CameraControlOutputPort=1
CameraControlConnecter=0
CC1Polarity=0
CC2Polarity=0
CC3Polarity=0
CC4Polarity=0

[CameraLink]
Tap=8
InputBit=8
LVALDelay=6
DVALDelayEnable=0
RearrangementEnable=1
RearrangementRegValue=537528318

[CaptureMode]
Mode=1
ExternalTriggerEnable=0
Timeout=1000

[Encoder]
EncoderStartMode=0
PulsePhase=0
```

```
PulseDirection=0
PulseDelay=0
PuleStep=4
```

5.3 Starting to Program

In this section, two simple codes are provided. One is for CML64_Snap() that will grab one image frame only. The other is for CML64_Live() that will grab images continuously and stop grabbing when CML64_Stop() is called.

5.3.1 Snap

```
int rtn;
int CardID = 0;
unsigned long pBuffer;
CML64_ImageSize ImageSize;

rtn = CML64_GetBoardNum();
rtn = CML64_OpenDevice(CardID);
rtn = CML64_LoadCameraFile(CardID, “C:\\Program
     Files\\ADLINK\\CML64\\CameraFile\\DALSA
     Piranha2_8k(8bit).ini”);

// Change the CamFile configuration here
// For example:
rtn = CML64_GetImageSizeForLine(CardID,
     &ImageSize);
ImageSize.Height = 5000;  // Change the line
     number of image frame
rtn = CML64_SetImageSizeForLine(CardID,
     ImageSize);

rtn = CML64_Snap(CardID, &pBuffer);

// Image is ready in buffer with address
     “pBuffer”
// Draw image or do inspection here

rtn = CML64_CloseDevice(CardID);
```

5.3.2 Live

For API CML64_Live(), you should have a callback function and set callback function in the main code.

```
/
     *******************************************
     **************/
// Callback function
void __stdcall MyCallback(int CardID, Param_Info
     *pParamInfo)
{
   // You can get the frame number, that you have
     grabbed after CML64_Live() called, by
     pParamInfo->FrameNo.

    // Image is ready in buffer with address
     "pParamInfo->pImageStartAddress"
// Draw image or do inspection here
}

/
     *******************************************
     **************/
// Main code
int rtn;
int CardID = 0;
CML64_ImageSize ImageSize;

rtn = CML64_GetBoardNum();
rtn = CML64_OpenDevice(CardID);
rtn = CML64_LoadCameraFile(CardID, "C:\\Program
     Files\\ADLINK\\CML64\\CameraFile\\DALSA
     Piranha2_8k(8bit).ini");

// Change the CamFile configuration here
// For example:
rtn = CML64_GetImageSizeForLine(CardID,
     &ImageSize);
ImageSize.Height = 5000;  // Change the line
     number of image frame
rtn = CML64_SetImageSizeForLine(CardID,
     ImageSize);
```

```
rtn = CML64_Live(CardID, 0);

/
      *******************************************
      **************/
// When you want stop grabbing image
rtn = CML64_Stop(CardID);

/
      *******************************************
      **************/
// When leave your owen application
rtn = CML64_CloseDevice(CardID);
```

5.4 Functions

5.4.1 CML64_GetBoardNum

Description

Get the number of CML64 cards that are on your system. You can call this function before opening any CML64 card.

Syntax

```
int CML64_GetBoardNum(void);
```

5.4.2 CML64_OpenDevice

Description

Open CML64 and initialize FPGA to default status. You should call this function before any other CML64 API except CML64_GetBoardNum().

Syntax

```
int CML64_OpenDevice(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.3 CML64_CloseDevice

Description

Close CML64 and release all resource that created by CML64 library. Call this function before your application closed.

Syntax

```
int CML64_CloseDevice(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.4 CML64_GetDLLVersion

Description

Get the version of current dll.

Syntax

```
int CML64_GetDLLVersion(char *VersionString, int
     StringSize);
```

Parameters

VersionString

[out] Pointer to a character array which contains the dll version.

StringSize

[in] Size of the character array that will be returned.

5.4.5 CML64_GetPSMFPGAVersion

Description

Get the version of FPGA on the image pre-processing module.

Syntax

```
int CML64_GetPSMFPGAVersion(int CardID, char
     *VersionString, int StringSize);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

VersionString

[out] Pointer to a character array which contains the FPGA version on the image pre-processing module.

StringSize

[in] Size of the character array that will be returned.

5.4.6 CML64_GetMainFPGAVersion

Description

Get the version of main FPGA on CML64 card.

Syntax

```
int CML64_GetMainFPGAVersion(int CardID, char
     *VersionString, int StringSize);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

VersionString

[out] Pointer to a character array which contains the main FPGA version on CML64 card.

StringSize

[in] Size of the character array that will be returned.

5.4.7 CML64_GetImageFPGAVersion

Description

Get the version of image FPGA on CML64 card.

Syntax

```
int CML64_GetImageFPGAVersion(int CardID, char
     *VersionString, int StringSize);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

VersionString

[out] Pointer to a character array which contains the image FPGA version on CML64 card.

StringSize

[in] Size of the character array that will be returned.

5.4.8 CML64_SetCallback

Description

Set callback function that CML64 library will call when one image frame is ready.

Syntax

```
int CML64_SetCallback(int CardID, void (__stdcall
      *CallbackAddress)(int CardID, Param_Info
      *pParamInfo));
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CallbackAddress

[in] Pointer to a callback function. CML64 library will call the callback function when one image frame ready. The callback function prototype may be similar to the following:

```
void __stdcall MyCallback(int CardID, Param_Info
      *pParamInfo);
```

The first parameter CardID in the callback function indicates which card the image is from. The second parameter Param_Info structure has two members FrameNo and pImage-StartAddress. FrameNo indicates the number of current frame, and pImageStartAddress is the address that points to the image buffer. For an example, please reference to section 5.3.2.

5.4.9 CML64_Snap

Description

Call this function to grab one image frame.

Syntax

```
     int CML64_Snap(int CardID, unsigned long
          *pImageStartAddress);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

pImageStartAddress

[out] This parameter stores the address that points to the image buffer when the function returns.

5.4.10 CML64_Live

Description

Call this function to grab images continuously. CML64 will stop grabbing images when it grabs enough frame numbers assigned by users or when CML64_Stop() is called.

Syntax

```
int CML64_Live(int CardID, unsigned int
     FrameNum=0);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

FrameNum

[in] The number of frames that user wants to grab. If this value is 0, frame grabbing will not stop until CML64_Stop() is called.

5.4.11 CML64_Stop

Description

Call this function to stop grabbing images. Usually, this function will be called after using CML64_Live().

Syntax

```
int CML64_Stop(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.12 CML64_IsGrabbing

Description

Check whether CML64 is grabbing images.

Syntax

```
int CML64_IsGrabbing(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.13 CML64_LoadRBFFile

Description

Load a RBF file to set FPGA registers on the image pre-processing module.

Syntax

```
int CML64_LoadRBFFile(int CardID, char
     *FileName);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

FileName

[in] The file path to a RBF file.

5.4.14 CML64_IsRBFLoaded

Description

Check whether the RBF file is loaded.

Syntax

```
int CML64_LoadRBFFile(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.15 CML64_LoadCameraFile

Description

Load a camera file and set the parameters in the camera file to FPGA registers.

Syntax

```
int CML64_LoadCameraFile(int CardID, char
     *FileName);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

FileName

[in] The file path to a camera file.

5.4.16 CML64_SetCamFileParamter

Description

Set CamFile parameters to FPGA registers.

Syntax

```
int CML64_SetCamFileParamter(int CardID,
     CML64_CamFile CamFile);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CamFile

[in] A CML64_CamFile structure which contains all of the configuration information of FPGA on CML64. CML64_CamFile has 6 members:

```
CML64_ExposureTrigger
CML64_ImageSize
CML64_CameraControl
CML64_CameraLink
CML64_CaptureMode
CML64_Encoder
```

The members of CML64_CamFile are the same with the context of a camera file. For more details about the members of CML64_CamFile structure, please reference to the following functions.

5.4.17 CML64_GetCamFileParamter

Description

Get CamFile parameters from FPGA registers.

Syntax

```
int CML64_GetCamFileParamter(int CardID,
     CML64_CamFile *CamFile);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0..

CamFile

[out] Pointer to a CML64_CamFile structure which receives the setting of FPGA on CML64. For the information of CML64_CamFile structure, please reference to the parameter descriptions of CML64_SetCamFileParameter function.

5.4.18 CML64_SetImageSizeForLine

Description

Set image dimension.

Syntax

```
int CML64_SetImageSizeForLine(int CardID,
     CML64_ImageSize ImageSize);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

ImageSize

[in] A CML64_ImageSize structure which contains the dimension of an image frame. For more details about the members of CML64_ImageSize structure, please reference to section 5.2.1.3 ImageSize Section.

5.4.19 CML64_GetImageSizeForLine

Description

Get current image dimension.

Syntax

```
int CML64_GetImageSizeForLine(int CardID,
     CML64_ImageSize *ImageSize);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

ImageSize

[out] Pointer to a CML64_ImageSize structure which receives the dimension of an image frame. For more details about the members of CML64_ImageSize structure, please reference to section 5.2.1.3 ImageSize Section.

5.4.20 CML64_SetExposureTriggerForLine

Description

Set CCD exposure time and related configuration.

Syntax

```
int CML64_SetExposureTriggerForLine(int CardID,
     CML64_ExposureTrigger ExposureTrig);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

ExposureTrig

[in] A CML64_ExposureTrigger structure which contains the configuration information of exposure trigger signals. For more details about the members of CML64_ExposureTrigger structure, please reference to section 5.2.1.2 ExposureTrigger Section.

5.4.21 CML64_GetExposureTriggerForLine

Description

Get current CCD exposure time and related configuration.

Syntax

```
int CML64_GetExposureTriggerForLine(int CardID,
    CML64_ExposureTrigger *ExposureTrig);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

ExposureTrig

[out] Pointer to a CML64_ExposureTrigger structure which receives the setting of exposure trigger signals. For more details about the members of CML64_ExposureTrigger structure, please reference to section 5.2.1.2 ExposureTrigger Section.

5.4.22 CML64_SetCameraControl

Description

Set camera control signal configuration.

Syntax

```
int CML64_SetCameraControl(int CardID,
     CML64_CameraControl CameraControl);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CameraControl

[in] A CML64_CameraControl structure which contains the configuration information of camera control signals. For more details about the members of CML64_CameraControl structure, please reference to section 5.2.1.4 CameraControl Section.

5.4.23 CML64_GetCameraControl

Description

Get current camera control signal configuration.

Syntax

```
int CML64_GetCameraControl(int CardID,
     CML64_CameraControl *CameraControl);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CameraControl

[out] Pointer to a CML64_CameraControl structure which receives the setting of camera control signals. For more details about the members of CML64_CameraControl structure, please reference to section 5.2.1.4 CameraControl Section.

5.4.24 CML64_SetCameraLinkCfg

Description

Set CameraLink related setting by CCD setting.

Syntax

```
int CML64_SetCameraLinkCfg(int CardID,
     CML64_CameraLink CameraLink);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CameraLink

[in] A CML64_CameraLink structure which contains the configuration information of CCD setting. For more details about the members of CML64_CameraLink structure, please reference to section 5.2.1.5 CameraLink Section.

5.4.25 CML64_GetCameraLinkCfg

Description

Get current CameraLink related setting.

Syntax

```
int CML64_GetCameraLinkCfg(int CardID,
     CML64_CameraLink *CameraLink);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CameraLink

[out] Pointer to a CML64_CameraLink structure which receives the CCD setting. For more details about the members of CML64_CameraLink structure, please reference to section 5.2.1.5 CameraLink Section.

5.4.26 CML64_SetCaptureMode

Description

Set capture mode and timeout.

Syntax

```
int CML64_SetCaptureMode(int CardID,
     CML64_CaptureMode CaptureMode);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CaptureMode

[in] A CML64_CaptureMode structure which contains the configuration information of capture mode. For more details about the members of CML64_CaptureMode structure, please reference to section 5.2.1.6 CaptureMode Section.

5.4.27 CML64_GetCaptureMode

Description

Get current capture mode and timeout.

Syntax

```
int CML64_GetCaptureMode(int CardID,
     CML64_CaptureMode *CaptureMode);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CaptureMode

[out] Pointer to a CML64_CaptureMode structure which receives the setting of capture mode. For more details about the members of CML64_CaptureMode structure, please reference to section 5.2.1.6 CaptureMode Section.

5.4.28 CML64_SetEncoder

Description

Set encoder signal configuration. This function is used when capture mode is not “Normal”.

Syntax

```
int CML64_SetEncoder(int CardID, CML64_Encoder
        Encoder);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Encoder

[in] A CML64_Encoder structure which contains the configuration information of encoder setting. For more details about the members of CML64_Encoder structure, please reference to section 5.2.1.7 Encoder Section.

5.4.29 CML64_GetEncoder

Description

Get current encoder signal configuration.

Syntax

```
int CML64_GetEncoder(int CardID, CML64_Encoder
     *Encoder);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Encoder

[out] Pointer to a CML64_Encoder structure which receives the encoder setting. For more details about the members of CML64_Encoder structure, please reference to section 5.2.1.7 Encoder Section.

5.4.30 CML64_GetEncoderCounter

Description

Get the value of encoder counter. It means the total number of received triggers.

Syntax

```
int CML64_GetEncoderCounter(int CardID, int
     *EncoderCounter);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

EncoderCounter

[out] Pointer to a 32-bit integer variable to store read out data.

5.4.31 CML64_ResetEncoderCounter

Description

Reset the encoder counter to 0.

Syntax

```
int CML64_ResetEncoderCounter(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.32 CML64_SetLineTriggerInLevel

Description

Set line trigger input level: TTL or RS422.

Syntax

```
int CML64_SetLineTriggerInLevel(int CardID, int
    Level);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Level

[in] A 32-bit integer variable which represents the input line trigger level.

0: TTL

1: RS422

5.4.33 CML64_GetLineTriggerInLevel

Description

Get line trigger input level : TTL or RS422.

Syntax

```
int CML64_GetLineTriggerInLevel(int CardID, int
     *Level);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Level

[out] Pointer to a 32-bit integer variable to store read out data.

5.4.34 CML64_GetLineCounter

Description

Get the value of line counter. It means the total number of captured lines.

Syntax

```
int CML64_GetLineCounter(int CardID, int
     *LineCounter);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

LineCounter

[out] Pointer to a 32-bit integer variable to store read out data.

5.4.35 CML64_ResetLineCounter

Description

Reset the line counter to 0.

Syntax

```
int CML64_ResetLineCounter(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.36 CML64_LineTriggerStartEnable

Description

Start grabbing images when the line trigger start signal is high.

Syntax

```
int CML64_LineTriggerStartEnable(int CardID, BOOL
     Enable);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Enable

[in] A boolean variable representing enable this function or not.

5.4.37 CML64_SetDMABufferAddr

Description

Change the image buffer address that will be used by CML64_Snap().

Syntax

```
int CML64_SetDMABufferAddr(int CardID, void
     *pBufferAddr);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

pBufferAddr

[in] Pointer to a buffer for CML64_Snap() function. This buffer is created by user himself and it is only used by CML64_Snap.

5.4.38 CML64_SetDMARingBufferNum

Description

Change the ring buffer number.

Syntax

```
int CML64_SetDMARingBufferNum(int CardID,
     unsigned int BufferNum);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

BufferNum

[in] The number of image buffers.

5.4.39 CML64_SetDMARingBufferAddr

Description

Change the ring buffer number and address. User should create the ring buffer by himself.

Syntax

```
int CML64_SetDMARingBufferAddr(int CardID, void
     **ppBufferAddr, unsigned int BufferNum);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

ppBufferAddr

[in] Pointer to a pointer array that stores the addresses of image buffers. These image buffers and their addresses are created and recorded by user himself. Here, the size of a pointer array is equal to the third parameter BufferNum.

BufferNum

[in] The number of image buffers.

5.4.40 CML64_SetDMARingBufferDefault

Description

Set the ring buffer to default setting.

Syntax

```
int CML64_SetDMARingBufferDefault(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.41 CML64_PSMReset

Description

Reset image pre-processing module to default status.

Syntax

```
int CML64_PSMReset(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.42 CML64_PSMSetRegValue

Description

Set value to the image pre-processing module register.

Syntax

```
int CML64_PSMSetRegValue(int CardID, unsigned int
     RegAddress, unsigned int RegData);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

RegAddress

[in] Address of a register in the image pre-processing module which will be written data in. Image pre-processing module has 256 registers, and the previous 32 registers are reserved. Therefore, the range of RegAddress is from 32 to 255.

RegData

[in] A 32-bit variable that will be wrote in the register at address RegAddress.

5.4.43 CML64_PSMGetRegValue

Description

Read value from the image pre-processing module register.

Syntax

```
int CML64_PSMGetRegValue(int CardID, unsigned int
     RegAddress, unsigned int *pRegData);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

RegAddress

[in] Address of a register in the image pre-processing module which will be read data out. Image pre-processing module has 256 registers, and the previous 32 registers are reserved. Therefore, the range of RegAddress is from 32 to 255.

pRegData

[out] Pointer to a 32-bit variable to store read out data.

5.4.44 CML64_PSMWriteArray

Description

Write a byte array to image pre-processing module.

Syntax

```
int CML64_PSMWriteArray(int CardID, unsigned int
      *ArrayData, unsigned int ArraySize);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

ArrayData

[in] Pointer to an array with 32-bit data which will be upload to the image pre-processing module.

ArraySize

[in] Size of the data array that will be upload to the image pre-processing module.

5.4.45 CML64_PSMPPEnable

Description

Enable PSM pre-processing function. If enabled, images will go through the image pre-processing module, and you should process these images manually. This function is only effective in CML64 FP, and it must be called right after CML64_LoadCameraFile. In CML64FB, do not use this function.

Syntax

```
int CML64_PSMPPEnable(int CardID, bool Enable);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Enable

[in] A boolean variable which indicates that enable this function or not.

5.4.46 CML64_SRLInitialize

Description

Initial the serial port communication device on the CML64 card.

Syntax

```
int CML64_SRLInitialize(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.47 CML64_SRLClose

Description

Close the serial port communication device and release resource.

Syntax

```
int CML64_SRLClose(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.48 CML64_SRLSetBaudRate

Description

Set serial port communication baud rate. Default is 9600.

Syntax

```
int CML64_SRLSetBaudRate(int CardID, long
     Baudrate);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

Baudrate

[in] The baud rate at which CML64 card and CCD operate, and it should match the configuration of the CCD. This parameter can be an actual baud rate, or one of the following indexes.

Index	Value	Meaning
CML64_BR_9600	9600	9600 bps
CML64_BR_19200	19200	19200 bps
CML64_BR_38400	38400	38400 bps
CML64_BR_57600	57600	57600 bps
CML64_BR_115200	115200	115200 bps
CML64_BR_230400	230400	230400 bps
CML64_BR_460800	460800	460800 bps
CML64_BR_921600	921600	921600 bps

5.4.49 CML64_SRLWriteCommand

Description

Write control command from CML64 card to CameraLink CCD.

Syntax

```
int CML64_SRLWriteCommand(int CardID, char
     *CmdBuffer, unsigned long *CmdBufferSize,
     int unsigned long TimeOut_ms);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

CmdBuffer

[in] Pointer to a buffer which stores the commands that will be sent to CCD.

CmdBufferSize

[in/out] This parameter indicates the size of command in byte. It also indicates how many bytes of data the function has sent when it returns.

TimeOut_ms

[in] The time-out interval in millisecond. The function returns if the interval had elapsed.

5.4.50 CML64_SRLReadCommand

Description

Read the CameraLink CCD response message.

Syntax

```
int CML64_SRLReadCommand(int CardID, char
     *RecvBuffer, unsigned long *RecvBufferSize,
     unsigned long TimeOut_ms);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

RecvBuffer

[out] Pointer to a buffer which stores received data. This buffer is created by user himself.

RecvBufferSize

[in/out] Size of the buffer which RecvBuffer points to. If received data is more than RecvBufferSize, the function will return since the received buffer is full. Otherwise, the function will return when data receive completely. This parameter also indicates how many bytes of data the function has received when it returns.

TimeOut_ms

[in] The time-out interval in millisecond. The function returns if the interval had elapsed.

5.4.51 CML64_SRLRxIsEmpty

Description

Check whether the receive buffer of serial port communication device is empty.

Syntax

```
int CML64_SRLRxIsEmpty(int CardID);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

5.4.52 CML64_SetDOStatus

Description

Set general-purpose digital output status.

Syntax

```
int CML64_SetDOStatus(int CardID, int DOStatus);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

DOStatus

[in] A 32-bit integer variable which represents the status of digital output.

0: Low

1: High

5.4.53 CML64_GetDIStatus

Description

Get general-purpose digital input status.

Syntax

```
int CML64_GetDIStatus(int CardID, int *DIStatus);
```

Parameters

CardID

[in] Card ID of CML64. The card ID could be 1, 2, 3 and 4. It is defined by DIP switch on CML64 main board. Please reference to chapter 2 - Hardware Reference. The default card ID is 0.

DIStatus

[out] Pointer to a 32-bit integer variable to store read out digital input status.

5.5 Error Codes

Table 5-2 lists all the error codes of CML64 API functions.

Error code	Meaning
0	Err_NoError
-2	Err_Open_CameraFile
-4	Err_TimeOut
-5	Err_Open_File
-6	Err_Grab
-500	Err_Card_Not_Initial
-501	Err_Invalid_Card_ID
-502	Err_Psm_Card_Not_Initial
-701	Err_COMM_Not_Initial
-702	Err_INIT_COMM
-703	Err_BaudRate
-705	Err_COMM_BUFF_OVERFLOW
-801	Err_Ring_Buffer_Overflow

Table 5-3: Error Codes

Warranty Policy

Thank you for choosing ADLINK. To understand your rights and enjoy all the after-sales services we offer, please read the following carefully.

1. Before using ADLINK's products please read the user manual and follow the instructions exactly. When sending in damaged products for repair, please attach an RMA application form which can be downloaded from: http://rma.adlinktech.com/policy/.

2. All ADLINK products come with a limited two-year warranty, one year for products bought in China:

   - The warranty period starts on the day the product is shipped from ADLINK's factory.
   - Peripherals and third-party products not manufactured by ADLINK will be covered by the original manufacturers' warranty.
   - For products containing storage devices (hard drives, flash cards, etc.), please back up your data before sending them for repair. ADLINK is not responsible for any loss of data.
   - Please ensure the use of properly licensed software with our systems. ADLINK does not condone the use of pirated software and will not service systems using such software. ADLINK will not be held legally responsible for products shipped with unlicensed software installed by the user.
   - For general repairs, please do not include peripheral accessories. If peripherals need to be included, be certain to specify which items you sent on the RMA Request & Confirmation Form. ADLINK is not responsible for items not listed on the RMA Request & Confirmation Form.

3. Our repair service is not covered by ADLINK's guarantee in the following situations:
   - Damage caused by not following instructions in the User's Manual.
   - Damage caused by carelessness on the user's part during product transportation.
   - Damage caused by fire, earthquakes, floods, lightening, pollution, other acts of God, and/or incorrect usage of voltage transformers.
   - Damage caused by unsuitable storage environments (i.e. high temperatures, high humidity, or volatile chemicals).
   - Damage caused by leakage of battery fluid during or after change of batteries by customer/user.
   - Damage from improper repair by unauthorized ADLINK technicians.
   - Products with altered and/or damaged serial numbers are not entitled to our service.
   - This warranty is not transferable or extendible.
   - Other categories not protected under our warranty.
4. Customers are responsible for shipping costs to transport damaged products to our company or sales office.
5. To ensure the speed and quality of product repair, please download an RMA application form from our company website: http://rma.adlinktech.com/policy. Damaged products with attached RMA forms receive priority.

If you have any further questions, please email our FAE staff: service@adlinktech.com.